Advanced BGP Features

Foreword

- Border Gateway Protocol (BGP) is usually deployed on large-scale networks. Compared with an Interior Gateway Protocol (IGP), BGP has more flexible route control capabilities. Each BGP route can carry multiple path attributes. Special route matching tools, such as AS_Path and community filters, are available for matching the route attributes. Routing policies can be used to control route advertisement and acceptance according to the actual networking requirements.
- In addition, BGP provides various advanced features and networking deployment solutions to improve network performance.
- This course describes the fundamentals and configurations of BGP route control, common advanced BGP features, such as ORF, peer group, and security, as well as the networking modes of BGP route reflectors (RRs).

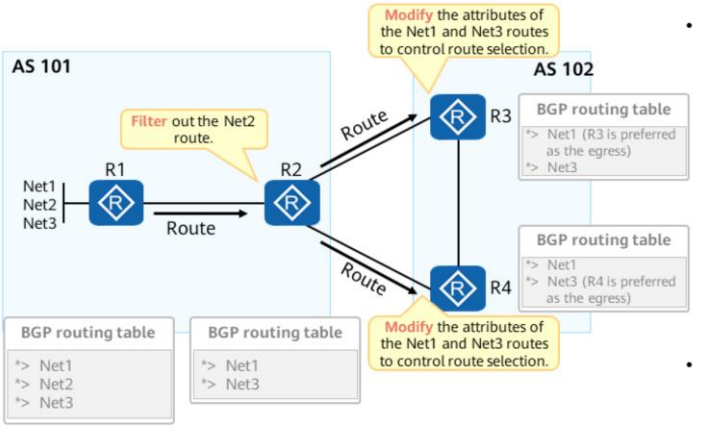
Objectives

- On completion of this course, you will be able to:
  - Use regular expressions during the configurations of AS_Path and community filters.
  - Use AS_Path and community filters to control BGP routes.
  - Apply the ORF and peer group functions of BGP.
  - Perform basic configurations of BGP security.
  - Learn about the networking modes of BGP RRs.

Contents

Overview of BGP Route Control

BGP route control involves controlling the advertisement and acceptance of routes.



- Generally, BGP route control is implemented using routing policies. Specifically, a route matching tool is used to match specific routes, and then a routing policy tool is used to control route advertisement and acceptance.
  - Route matching tools: access control list (ACL), IP prefix list, AS_Path filter, and community filter
  - Routing policy tools: filter-policy and route-policy
- BGP route control usually affects the AS_Path and community attributes.

  HUAWEI

- As shown in the figure:
  - R1 and R2 reside in AS 101, and establish an IBGP peer relationship with each other. R3 and R4 reside in AS 102, and each establishes an EBGP peer relationship with R2.
  - R1 is directly connected to three network segments: Net1, Net2, and Net3. R1 advertises routes to the three network segments to its BGP routing table.
  - R2 can filter out the Net2 route through BGP route control so that R2's BGP routing table does not contain the Net2 route.
  - R3 and R4 can implement BGP route control by modifying the attributes of the Net1 and Net3 routes, respectively. In this way, when a device in AS 102 accesses Net1, R3 is preferentially selected as the egress device; when the device accesses Net3, R4 is preferentially selected as the egress device.
- Note: For details about the ACL, IP prefix list, filter-policy, route-policy, and BGP path attributes, see the "HCIP-Datacom-Core Technology" course.

Regular Expression

- A regular expression (regex) is a formula used to match character strings based on a specific template. It consists of common characters (for example, letters from a to z) and special characters.
- Common characters are used to match themselves in a string:
  - Include all upper-case and lower-case letters, digits, punctuations, and special symbols.
  - For example, the regex a matches the letter "a" in "abc", the regex 10 matches the digits "10" in "10.113.25.155", and the regex @ matches the symbol "@" in "xxx@xxx.com".
- Special characters, together with common characters, are used to match complex or special character strings:
  - A special character that precedes or follows a common character is used to restrict or extend the independent control character or placeholder of the common character.
  - A special character describes how the character that precedes the special character is reused.
  - A special character specifies a complete range.

- A regex has the following functions:
  - Checks and obtains the sub-character string that matches a specific rule in the character string.
  - Replaces the character string based on matching rules.

Example of Special Characters (1)

- Type 1

.	Matches any single character, including a space.	0.0 matches 0x0, 020, and so on.
^	Matches the start position of a character string.	^10 matches 10.1.1.1 but not 20.1.1.1.
$	Matches the end position of a character string.	1$ matches 10.1.1.1 but not 10.1.1.2.
_	Underscore, which matches any separator. Matches a comma (,), left brace ({), right brace (}), left parenthesis ((), or right parenthesis ()). (Same as ^) Matches the start position of an input character string. (Same as $) Matches the end position of an input character string. Matches a space.	_10 matches (10, {10, space10, and so on. 10_ matches 10), 10}, 10space, and so on.
|	Pipeline character, which is a logical OR operator. x|y matches x or y.	100|200 matches 100 or 200.
\	Defines an escape character, which is used to mark the next character (common or special character) as a common character.	* matches *.

- Type 2

*	Matches a sub-regular expression that it follows zero or multiple times.	10* matches 1, 10, 100, 1000, and so on.	(10) * matches null, 10, 1010, 101010, and so on.
+	Matches a sub-regular expression that it follows once or multiple times.	10+ matches 10, 100, 1000, and so on.	(10)+ matches null, 10, 1010, 101010, and so on.
?	Matches a sub-regular expression that it follows zero times or once.	10? matches 1 or 10.	(10)? matches null or 10.

- Note: The parentheses () can be used to define the scope and priority of an operator. For example, gr(a|e)y is equivalent to gray|grey.

Example of Special Characters (2)

- Type 3

[xyz]	Matches any character contained in a regex.	[123] matches the character 2 in 255.
[^xyz]	Matches any character that is not contained in a regex.	[^123] matches any character except 1, 2, and 3.
[a-z]	Matches any character within a specified range in a regex.	[0-9] matches all digits in the range from 0 to 9.
[^a-z]	Matches any character beyond the range specified in a regex.	[^0-9] matches all non-digit characters (matching any characters except digits 0 to 9).

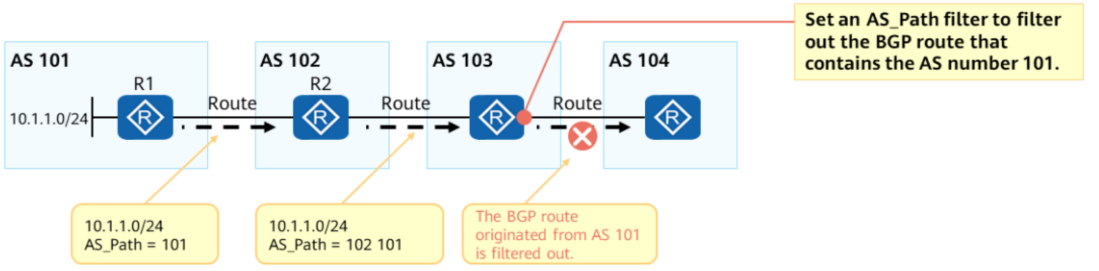
- Think about character strings that can be matched against the regexes of the following types.

Type 1	Type 2	Type 3
^a.$	abc*d	[abcd]
^100_	abc+d	[a-c 1-2]$
^100$	abc?d	[^act]$
100$\|400$	a(bc)?d	[123].[7-9]
^\(65000\)$		

- Quiz:
- Type 1:
  - **^a.$**: matches a character string that starts with the character a and ends with any single character, for example, a0, a!, ax, and so on.
  - **^100_**: matches a character string starting with 100, for example, 100, 100 200, 100 300 400, and so on.
  - **^100$**: matches only 100.
  - **100$|400$**: matches a character string ending with 100 or 400, for example, 100, 1400, 300 400, and so on.
  - **^\(65000\)$**: matches (65000) only.
- Type 2:
  - **abc*d**: matches the character c zero or multiple times, for example, abd, abcd, abccd, abcccd, abccccdef, and so on.
  - **abc+d**: matches the character c once or multiple times, for example, abcd, abccd, abcccd, abccccdef, and so on.
  - **abc?d**: matches the character c zero times or once, for example, abd, abcd, abcdef, and so on.
  - **a(bc)?d**: matches the character string bc zero times or once, for example, ad, abcd, aaabcdef, and so on.
- Type 3:
  - **[abcd]**: matches any character in the string abcd, for example, ax, b!, abc, d0, and so on.
  - **[a-c 1-2]$**: matches a character string ending with any of a, b, c, 1, and 2, for example, a, a1, 62, xb, 7ac, and so on.
  - **[^act]$**: matches a character string that does not end with a, c, or t, for example, ax, b!, d, and so on.
  - **[123].[7-9]**: matches the character strings such as 1 7, 2x9, and 348.

Route Matching Tool: AS_Path Filter

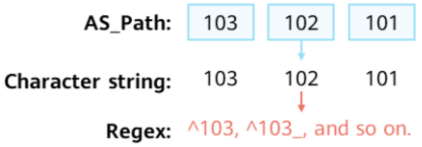
- An AS_Path filter uses the AS_Path attribute of BGP routes as a matching condition to filter BGP routes.
- If you do not want routes from certain ASs, you can use an AS_Path filter to filter out the routes that carry the associated AS numbers.



- The AS_Path attribute is a well-known mandatory attribute of BGP. All BGP routes must carry this attribute. This attribute records the numbers of all the ASs that a BGP route traversed during transmission.
- The value of the AS_Path attribute can be 0, 1, or a set of multiple AS numbers.

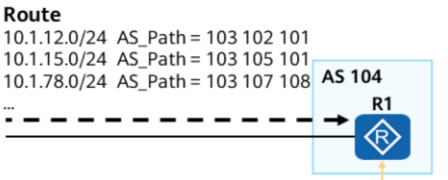
Using a Regex to Match the AS_Path Attribute

- A regex can be used to match the AS_Path attribute of routes.
  - For example, a regex can match the AS number 103 in AS_Path = 103 102 101.



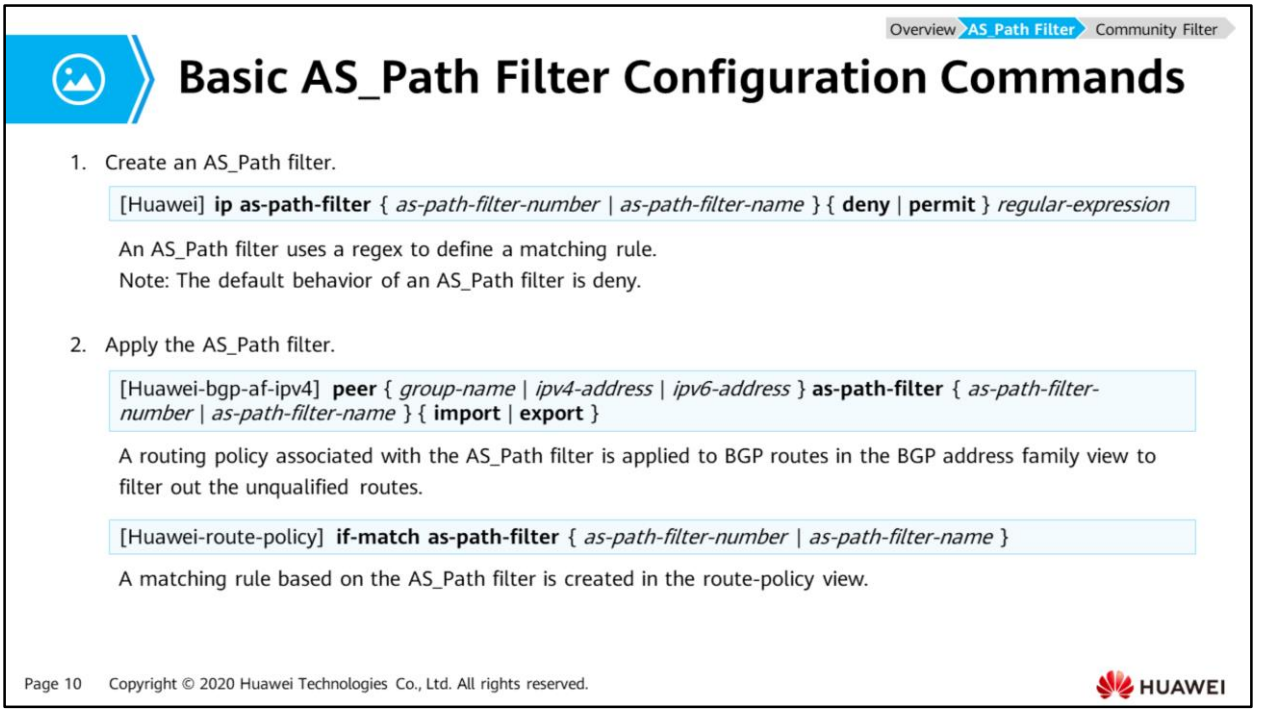
- Other examples:

^$	Matches the AS_Path list that does not contain any AS number, that is, matching routes in the local AS.
.*	Match all or any routes.
^10[012349]$	Matches the AS numbers 100, 101, 102, 103, 104, and 109.
^10[^0-6]$	Matches the AS_Path lists that exclude the AS numbers 100 to 106.
^10.	Matches the AS numbers 100 to 109 and 10.
^12(_34)?_56$	Matches the AS_Path lists 12 56 and 12 34 56.



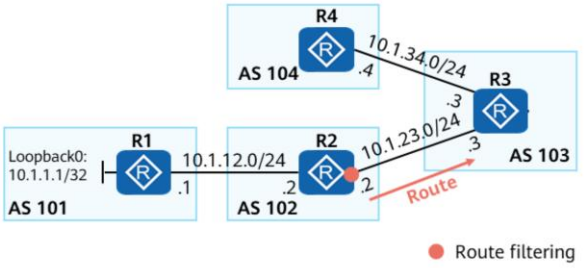
R1 receives many BGP routes, each of which has its own AS_Path value.
To meet a specific requirement, R1 needs to apply a policy to the routes carrying the AS number 101 in their AS_Path attribute. In this case, you can associate an AS_Path filter with a regex to match routes, regardless of the route prefixes.

You can use different regexes to meet different requirements, for example, using ^101$ to match AS 101, and _101$ to match the AS_Path lists ending with AS 101.



- Multiple matching rules (each in permit or deny mode) can be specified in an AS_Path filter. These rules are in the OR relationship, which means that if a route matches one of the matching rules, the route is considered to match the AS_Path filter.
- Command: [Huawei] **ip as-path-filter** {*as-path-filter-number* | *as-path-filter-name*} {**deny** | **permit**} *regular-expression*
    - *as-path-filter-number*: specifies the number of an AS_Path filter. The value is an integer ranging from 1 to 256.
    - *as-path-filter-name*: specifies the name of an AS_Path filter. The value is a string of 1 to 51 case-sensitive characters. It cannot be comprised of only digits. If spaces are used, the string must start and end with double quotation marks (").
    - **deny:** sets the matching mode of the AS_Path filter to deny.
    - **permit**: sets the matching mode of the AS_Path filter to permit.
    - *regular-expression*: specifies a regex for the AS_Path filter. The value is a string of 1 to 255 characters and can contain spaces.
- The default behavior of an AS_Path filter is deny. That is, if a route is not permitted in a filtering, the route fails to match the AS_Path filter. If all matching rules in an AS_Path filter work in deny mode, all BGP routes are denied by the filter. To prevent this problem, configure a matching rule in permit mode after one or more matching rules in deny mode so that the routes except for those denied by the preceding matching rules can match the filter.

Examples for Configuring AS_Path Filters (1)



R2 transmits EBGP routes to R3. Among these routes, some are locally originated by R2, and others are transmitted from AS 101 to R2 and then updated by R2 to R3.

A routing policy is configured on R2 to deny the routes originated from AS 101.

1. Create an AS_Path filter.

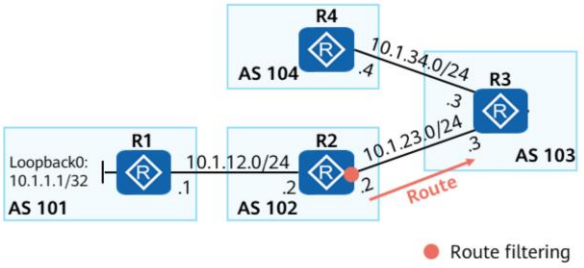
```
[R2] ip as-path-filter 1 deny _101$
[R2] ip as-path-filter 1 permit .*
```

The AS_Path filter is configured to filter out the routes originated from AS 101 and permit the other routes.

2. (Direct reference) Apply the AS_Path filter.

```
[R2] bgp 102
[R2-bgp] peer 10.1.23.3 as-number 103
[R2-bgp] ipv4-family unicast
[R2-bgp-af-ipv4] peer 10.1.23.3 as-path-filter 1 export
```

Examples for Configuring AS_Path Filters (2)



R2 transmits EBGP routes to R3. Among these routes, some are locally originated by R2, and others are transmitted from AS 101 to R2 and then updated by R2 to R3.

A routing policy is configured on R2 to deny the routes originated from AS 101.

1. Create an AS_Path filter.

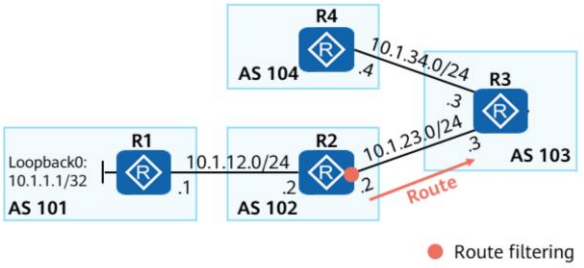
```
[R2] ip as-path-filter 1 deny _101$
[R2] ip as-path-filter 1 permit .*
```

The AS_Path filter is configured to filter out the routes originated from AS 101 and permit the other routes.

2. (Through a route-policy) Apply the AS_Path filter.

```
[R2] route-policy AS_Path permit node 10
[R2-route-policy] if-match as-path-filter 1
[R2-route-policy] quit
[R2] bgp 102
[R2-bgp] peer 10.1.23.3 as-number 103
[R2-bgp] ipv4-family unicast
[R2-bgp-af-ipv4] peer 10.1.23.3 route-policy AS_Path export
```

Checking the AS_Path Filter Information



Route filtering

1. Check the AS_Path filter.

```
[R2]display ip as-path-filter 1
As path filter number: 2
        deny        _101$
        permit      .*
```

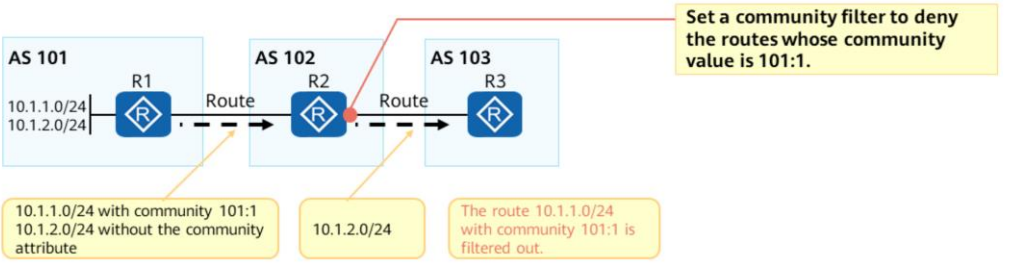
2. Check all the routes whose AS_Path lists match the specified regex in the BGP routing table.

```
[R2]display bgp routing-table regular-expression _101$

 Total Number of Routes: 1

 BGP Local router ID is 10.1.12.2
 Status codes: * - valid, > - best, d - damped,
               h - history,  i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
    Network         NextHop    MED  LocPrf  PrefVal  Path/Ogn
*>  10.1.1.1/32     10.1.12.1  0                0        101i
```

The route 10.1.1.1/32 originated from AS 101 is filtered by the AS_Path filter.

Route Matching Tool: Community Filter

- The community filter and community attribute can be used together to simplify route management when the IP prefix list and AS_Path filter are not available.
- There are two types of community filters:
  - Basic community filter, which matches the community number or well-known community attribute.
  - Advanced community filter, which uses a regex to match the community number.



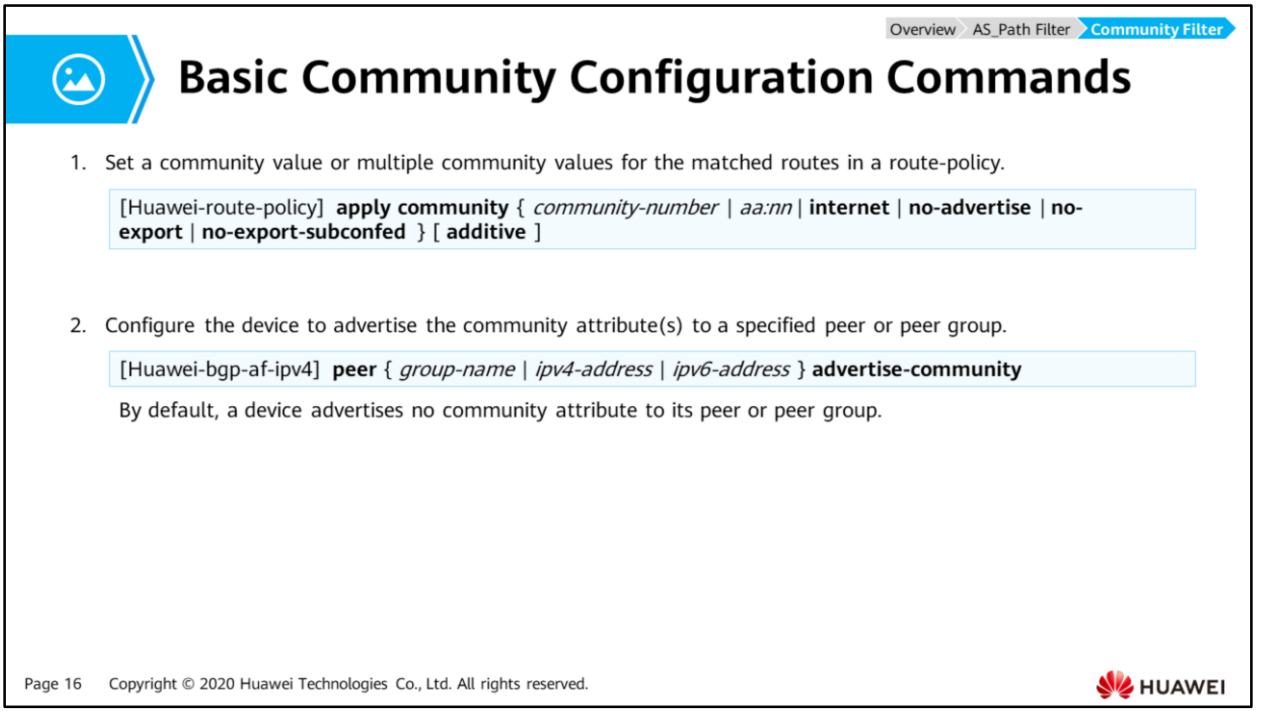
- The community attribute is an optional transitive attribute. It can identify the routes with the same characteristics, regardless of the scattered route prefixes and various AS numbers. That is, a specific community value can be assigned to some routes so that these routes can be matched against the community value instead of the network number or mask. Then, a corresponding routing policy can be applied to the matched routes.

Community Attributes

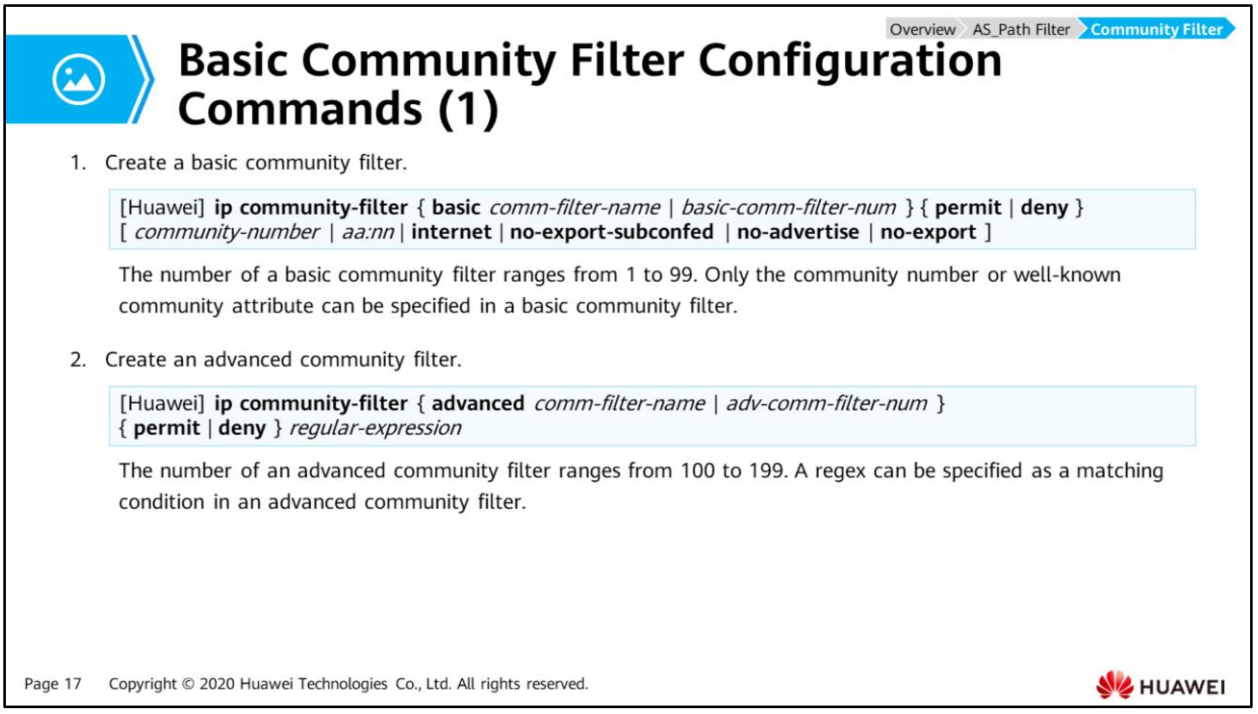
- Well-known community attributes

Name	Number	Description
Internet	0 (0x00000000)	After a device receives a route with this attribute, the device can advertise this route to any BGP peer. By default, all routes belong to the Internet community.
No_Advertise	4294967042 (0xFFFFFF02)	After a device receives a route with this attribute, the device does not advertise this route to any BGP peer.
No_Export	4294967041 (0xFFFFFF01)	After a device receives a route with this attribute, the device does not advertise this route outside the local AS.
No_Export_Subconfed	4294967043 (0xFFFFFF03)	After a device receives a route with this attribute, the device does not advertise this route outside the local AS. If a confederation is used, the device does not advertise this route to the other sub-ASs in the confederation.

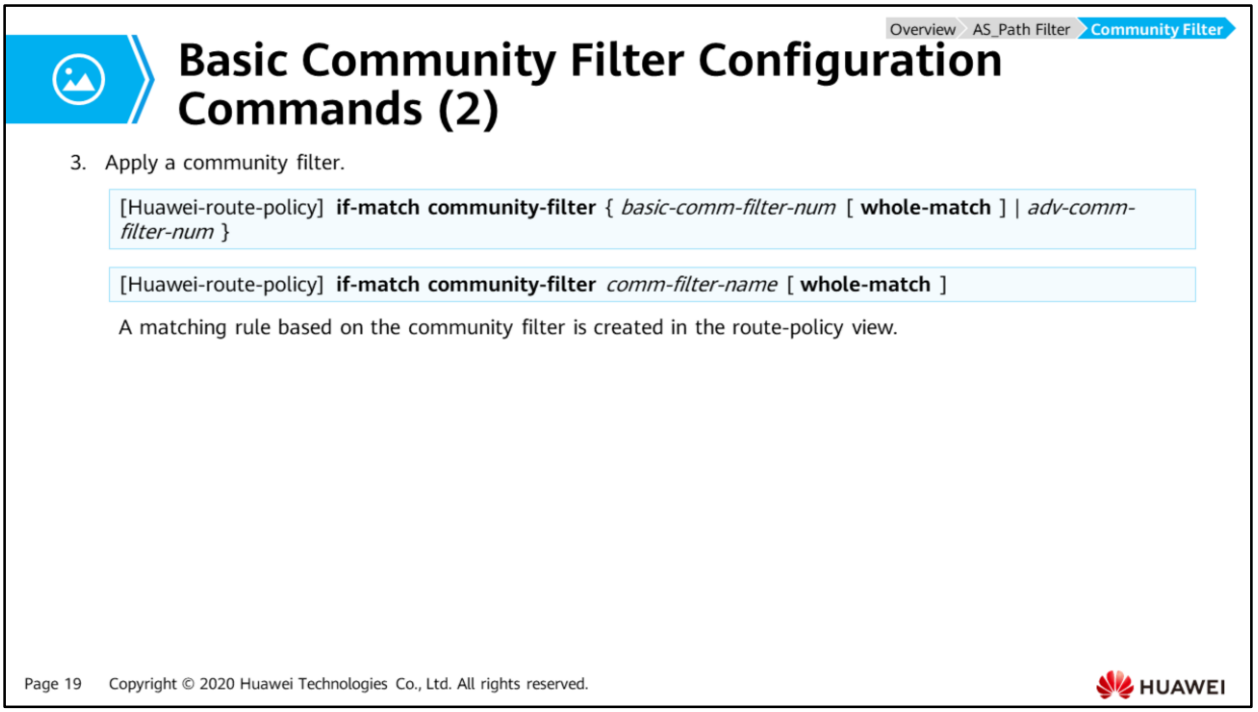
- Community attribute format:
  - The length of a community attribute is 32 bits, which can be presented in either of the following formats:
    - Decimal integer
    - In the *AA*:*NN* format, *AA* indicates an AS number, and *NN* is a user-defined number.



- Command: [Huawei-route-policy] **apply community** { *community-number* | *aa:nn* | **internet** | **no-advertise** | **no-export** | **no-export-subconfed** } [**additive**]
  - *community-number* | *aa:nn*: specifies a community number for a community attribute. A maximum of 32 community numbers can be specified at a time using this command. The value of *community-number* is an integer ranging from 0 to 4294967295. The values of *aa* and *nn* are also integers ranging from 0 to 65535.
  - **internet**: allows the matched routes to be advertised to any peers. By default, all routes belong to the Internet community.
  - **no-advertise**: prevents the matched routes from being advertised to any peer. After a device receives a route with this attribute, it cannot advertise this route to any other BGP peers.
  - **no-export**: prevents the matched routes from being advertised outside the local AS but allows them to be advertised to other sub-ASs in the local AS. After a device receives a route with this attribute, it cannot advertise this route outside the local AS.
  - **no-export-subconfed**: prevents the matched routes from being advertised outside the local AS or to other sub-ASs in the local AS. After a device receives a route with this attribute, it cannot advertise this route to any other sub-ASs.
  - **additive:** adds community attributes to the routes that match the filtering conditions.



- Command: [Huawei] **ip community-filter** { **basic** *comm-filter-name* | *basic-comm-filter-num* } { **permit** | **deny** } [*community-number* | *aa:nn* | **internet** | **no-export-subconfed** | **no-advertise** | **no-export**]
  - **basic** *comm-filter-name*: specifies the name of a basic community filter. The value is a string of 1 to 51 case-sensitive characters. It cannot be comprised of only digits.
  - *basic-comm-filter-num*: specifies the number of a basic community filter. The value is an integer ranging from 1 to 99.
  - **deny**: sets the matching mode of the community filter to deny.
  - **permit**: sets the matching mode of the community filter to permit.
  - *community-number*: specifies a community number. The value is an integer ranging from 0 to 4294967295.
  - *aa:nn*: specifies a community number. A maximum of 20 community numbers can be specified at a time using this command. The values of *aa* and *nn* are integers ranging from 0 to 65535.
  - **internet**: allows the matched routes to be advertised to any peers.
  - **no-export-subconfed**: prevents the matched routes from being advertised outside the local AS. If a confederation is used, the matched routes will not be advertised to the other sub-ASs in the confederation.
  - **no-advertise**: prevents the matched routes from being advertised to any other peers.
  - **no-export**: prevents the matched routes from being advertised outside the local AS. If a confederation is used, the matched routes will not be advertised outside the confederation but will be advertised to the other sub-ASs in the confederation.



- Command: [Huawei-route-policy] **if-match community-filter** { *basic-comm-filter-num* [**whole-match**] | *adv-comm-filter-num* }
- Command: [Huawei-route-policy] **if-match community-filter** *comm-filter-name* [**whole-match**]
    - *basic-comm-filter-num*: specifies the number of a basic community filter. The value is an integer ranging from 1 to 99.
    - *adv-comm-filter-num*: specifies the number of an advanced community filter. The value is an integer ranging from 100 to 199.
    - *comm-filter-name*: specifies the name of a community filter. The value is a string of 1 to 51 case-sensitive characters. It cannot be comprised of only digits. If spaces are used, the string must start and end with double quotation marks (").
    - **whole-match**: indicates complete matching. That is, all the community attributes in the specified community filter must be matched. This parameter applies only to basic community filters.

Examples for Configuring Community Filters

1. Example for configuring a basic community filter

   Match the routes carrying community values [100:1, 200:1, 300:1]. (The relationship between multiple community values in a command configuration is AND.)

   ```
   ip community-filter 1 permit 100:1 200:1 300:1
   ```

   Match the routes carrying community value [100:1] or community values [200:1, 300:1]. (The relationship between different groups of community values is OR. A set of community values in a command configuration is called a group.)

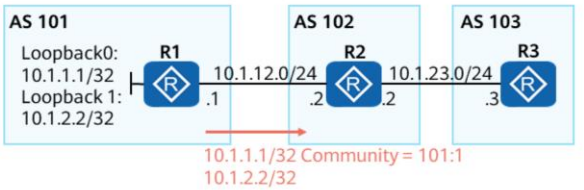
   ```
   ip community-filter 1 permit 100:1
   ip community-filter 1 permit 200:1 300:1
   ```

2. Example for configuring an advanced community filter

   Match the routes carrying community values starting with 10.

   ```
   ip community-filter 100 permit ^10
   ```

Configuring Community Attributes (1)



Configure a routing policy on R1 to allow the community value 101:1 to be carried only in the BGP route 10.1.1.1/32 when R1 advertises this route.

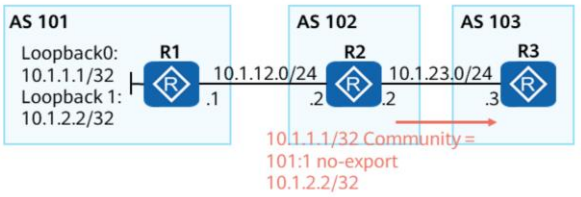
1. Configure a routing policy on R1 to add a community attribute to matched routes, and enable R1 to advertise the community attribute to the EBGP peer R2.

```
[R1] ip ip-prefix 1 permit 10.1.1.1 32
[R1] route-policy Community permit node 10
[R1-route-policy] if-match ip-prefix 1
[R1-route-policy] apply community 101:1
[R1-route-policy] quit
[R1] route-policy Community permit node 20
[R1-route-policy] quit
[R1] bgp 101
[R1-bgp] peer 10.1.12.2 as-number 102
[R1-bgp] peer 10.1.12.2 route-policy Community export
[R1-bgp] peer 10.1.12.2 advertise-community
[R1-bgp] network 10.1.1.1 32
[R1-bgp] network 10.1.2.2 32
```


HUAWEI

- Command: [R1] **route-policy** Community **permit node** 20
- Run this command to allow the route 10.1.2.2/32 to be advertised properly.

Configuring Community Attributes (2)



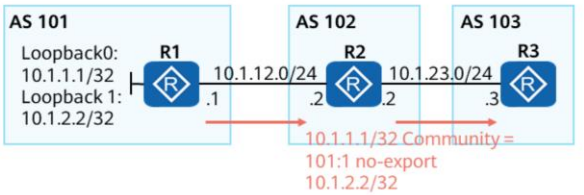
Configure a routing policy on R1 to allow the community value 101:1 to be carried only in the BGP route 10.1.1.1/32 when R1 advertises this route.

Configure a routing policy on R2 to allow the community attribute no-expert to be added to the BGP route 10.1.1.1/32 when R2 advertises this route.

2. Configure R2 to advertise the community attribute to the EBGP peer R3.

```
[R2] ip ip-prefix 1 permit 10.1.1.1 32
[R2] route-policy Community permit node 10
[R2-route-policy] if-match ip-prefix 1
[R2-route-policy] apply community no-expert additive
[R2-route-policy] quit
[R2] route-policy Community permit node 20
[R2-route-policy] quit
[R2] bgp 102
[R2-bgp] peer 10.1.12.1 as-number 101
[R2-bgp] peer 10.1.23.3 as-number 102
[R2-bgp] peer 10.1.23.3 advertise-community
[R2-bgp] peer 10.1.23.3 route-policy Community export
```


Configuring Community Attributes (3)



Configure a routing policy on R1 to allow the community value 101:1 to be carried only in the BGP route 10.1.1.1/32 when R1 advertises this route.

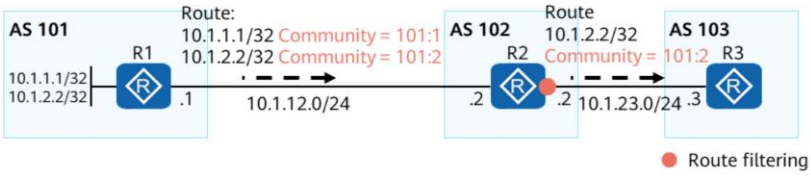
Configure a routing policy on R2 to allow the community attribute no-expert to be added to the BGP route 10.1.1.1/32 when R2 advertises this route.

3. Check the BGP routing information on R3.

```
[R3] bgp 103
[R3-bgp] peer 10.1.23.2 as-number 102
[R3-bgp] quit

[R3] display bgp routing-table 10.1.1.1
 BGP local router ID : 10.1.23.3
 Local AS number : 103
 Paths:   1 available, 1 best, 1 select
 BGP routing table entry information of 10.1.1.1/32:
 From: 10.1.23.2 (10.1.12.2)
 Route Duration: 00h00m21s
 Direct Out-interface: GigabitEthernet0/0/2
 Original nexthop: 10.1.23.2
 Qos information : 0x0
 Community:<101:1>, no-export
 AS-path 102 101, origin igp, pref-val 0, valid, external,
best, select, active, pre 255
 Not advertised to any peer yet
```

Check the route 10.1.1.1/32 on R3. The command output shows that the route carries two community values: 101:1 and no-export.

Configuring a Community Filter (1)



R2 transmits routes to the EBGP peer R3. Configure a routing policy on R2 to filter out the route carrying the community value 101:1.

1. Configure a community filter to match the route carrying the community value 101:1.

```
[R2] ip community-filter 1 permit 101:1
```

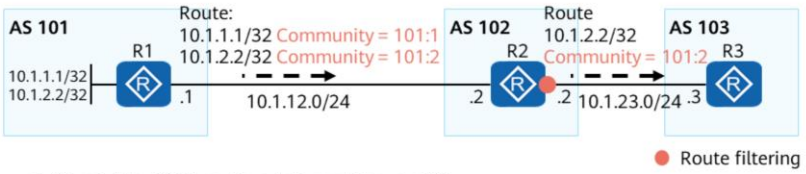
2. Configure a matching rule that is based on the community filter.

```
[R2] route-policy Community deny node 10
[R2-route-policy] if-match community-filter 1
[R2-route-policy] quit
[R2] route-policy Community permit node 20
[R2-route-policy] quit
[R2] bgp 102
[R2-bgp] peer 10.1.23.3 route-policy Community export
```

3. Check the community filter information on R2.

```
[R2]display ip community-filter 1
Community filter Number: 1
        permit 101:1
```

Configuring a Community Filter (2)



R2 transmits routes to the EBGP peer R3. Configure a routing policy on R2 to filter out the route carrying the community value 101:1.

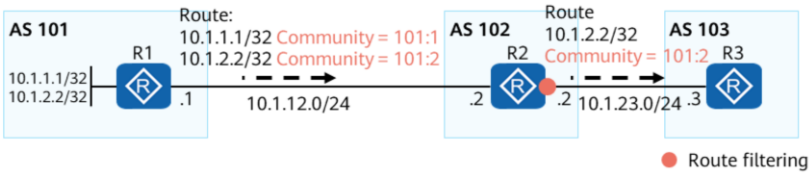
4. Check the BGP routing information on R2.

```
[R2]dis bgp routing-table 10.1.1.1
 BGP local router ID : 10.1.12.2
 Local AS number : 102
 Paths:   1 available, 1 best, 1 select
 BGP routing table entry information of 10.1.1.1/32:
 From: 10.1.12.1 (10.1.1.1)
 Route Duration: 00h13m39s
 Direct Out-interface: GigabitEthernet0/0/1
 Original nexthop: 10.1.12.1
 Qos information : 0x0
 Community:<101:1>
 AS-path 101, origin igp, MED 0, pref-val 0, valid, external, best, select, active, pre 255
 Not advertised to any peer yet
```

Check the route 10.1.1.1/32 on R2. The command output shows that the route carries the community value 101:1.

Use the community filter to filter out this route.

Configuring a Community Filter (3)



R2 transmits routes to the EBGP peer R3. Configure a routing policy on R2 to filter out the route carrying the community value 101:1.

5. Check the BGP routing table on R3.

```
[R3]display bgp routing-table
 BGP Local router ID is 10.1.23.3
 Status codes: * - valid, > - best, d - damped,
               h - history,  i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
 Total Number of Routes: 1
      Network       NextHop       MED        LocPrf    PrefVal  Path/Ogn
 *>   10.1.2.2/32   10.1.23.2                          0        102 101i
```

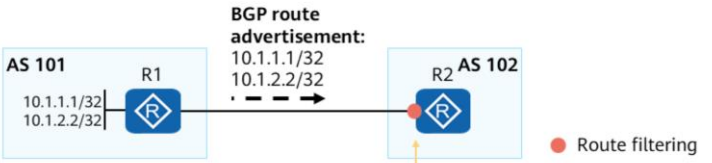
R3 does not receive the BGP route 10.1.1.1/32.

Contents

HUAWEI

On-Demand Route Advertisement by BGP Peers

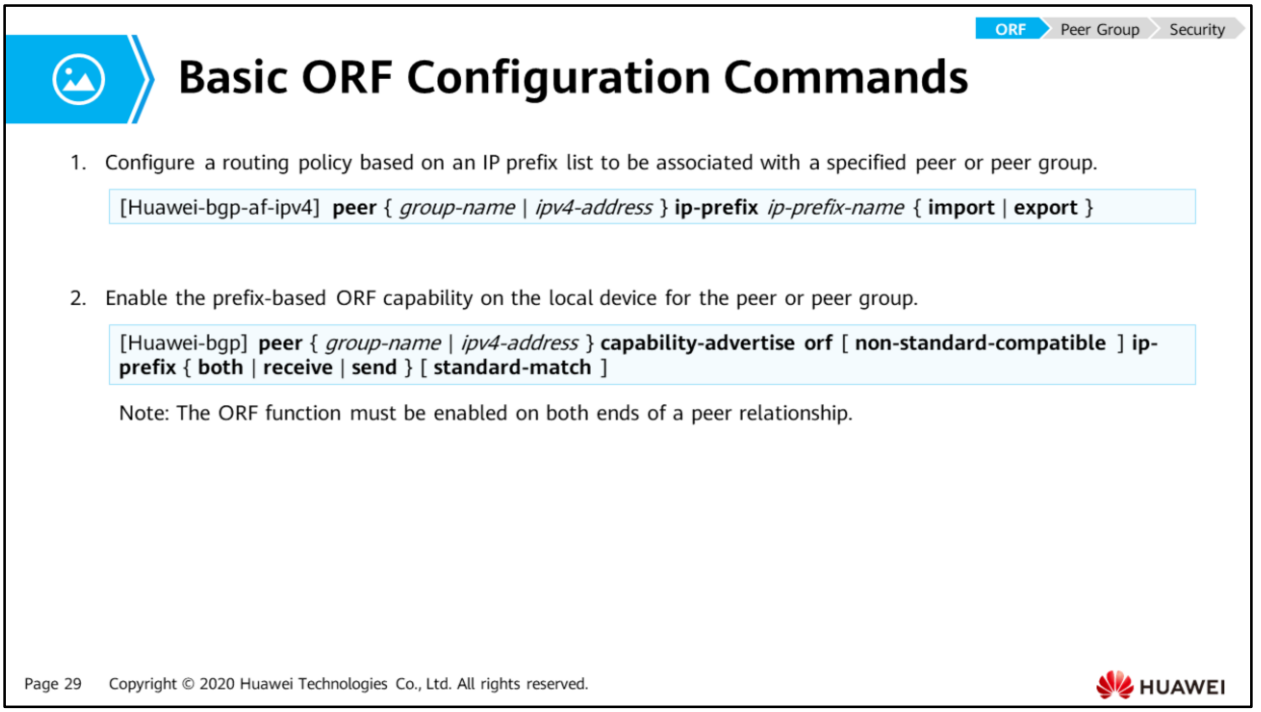
- If a device expects to receive only required routes from a remote device, and the remote device cannot maintain a separate outbound routing policy for each connected peer, you can configure a prefix-based outbound route filter (ORF) to implement on-demand route advertisement.



R2 filters routes on the inbound interface and accepts only the route 10.1.1.1/32.

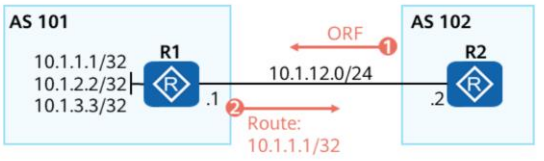
For the routes that are filtered out (for example, route 10.1.2.2/32), actually R1 does not need to advertise them to R2.

- Prefix-based ORF enables a device to send locally configured prefix-based import policies to its BGP peers through Route-refresh messages. The BGP peers construct export policies based on the received policies (in Route-refresh messages) to filter routes to be advertised.
- This prevents the local device from receiving a large number of unwanted routes, lowers the CPU usage of the local device, reduces the configuration workload on BGP peers, and lowers the link bandwidth usage.



- Command: [Huawei-bgp-af-ipv4] **peer** { *group-name* | *ipv4-address* } **ip-prefix** *ip-prefix-name* { **import** | **export** }
  - **import**: applies the routing policy to the routes received from the peer or peer group.
  - **export**: applies the routing policy to the routes to be advertised to the peer or peer group.
- Command: [Huawei-bgp] **peer** { *group-name* | *ipv4-address* } **capability-advertise orf** [**non-standard-compatible**] **ip-prefix** { **both** | **receive** | **send** } [**standard-match**]
  - **non-standard-compatible**: indicates that the ORF capability supported by the Huawei device is compatible with that supported by a non-Huawei device.
  - **both**: enables the local device to both send and accept ORF packets.
  - **receive**: enables the local device to only accept ORF packets.
  - **send**: enables the local device to only advertise ORF packets.
  - **standard-match**: matches routes according to the prefix matching rules defined in an RFC standard.

Example for Configuring ORF



R2 expects R1 to only advertise the route 10.1.1.1/32, and sends an ORF packet to R1 to meet this expectation.

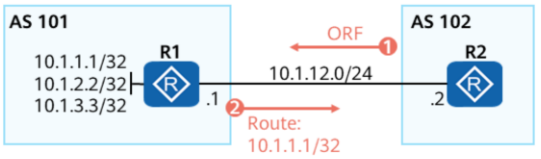
1. Configure a routing policy on R2 to filter the route 10.1.1.1/32, and enable ORF on R2 to allow R2 to send ORF packets.

```
[R2] ip ip-prefix 1 permit 10.1.1.1 32
[R2] bgp 102
[R2-bgp] peer 10.1.12.1 as-number 101
[R2-bgp] peer 10.1.12.1 ip-prefix 1 import
[R2-bgp] peer 10.1.12.1 capability-advertise orf ip-prefix send
```

2. Enable ORF on R1 to allow R1 to accept ORF packets.

```
[R1] bgp 101
[R1-bgp] peer 10.1.12.2 as-number 102
[R1-bgp] peer 10.1.12.2 capability-advertise orf ip-prefix receive
[R1-bgp] network 10.1.1.1 32
[R1-bgp] network 10.1.2.2 32
[R1-bgp] network 10.1.3.3 32
```

Verifying the ORF Configuration



1. On R1, check the prefix-based ORF information sent by R2.

```
[R1]display bgp peer 10.1.12.2 orf ip-prefix
 Total number of ip-prefix received: 1
 Index      Action     Prefix          MaskLen  MinLen  MaxLen
 10         Permit     10.1.1.1        32
```

2. Check the BGP routing information on R2.

```
[R2]display bgp routing-table peer 10.1.12.1 received-routes

 BGP Local router ID is 10.1.12.2
 Status codes: * - valid, > - best, d - damped,
               h - history,  i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete

 Total Number of Routes: 1
    Network        NextHop        MED        LocPrf    PrefVal  Path/Ogn
 *> 10.1.1.1/32    10.1.12.1      0                    0        101i
```

Check the ORF information on R1. The command output shows that the route 10.1.1.1/32 is required.

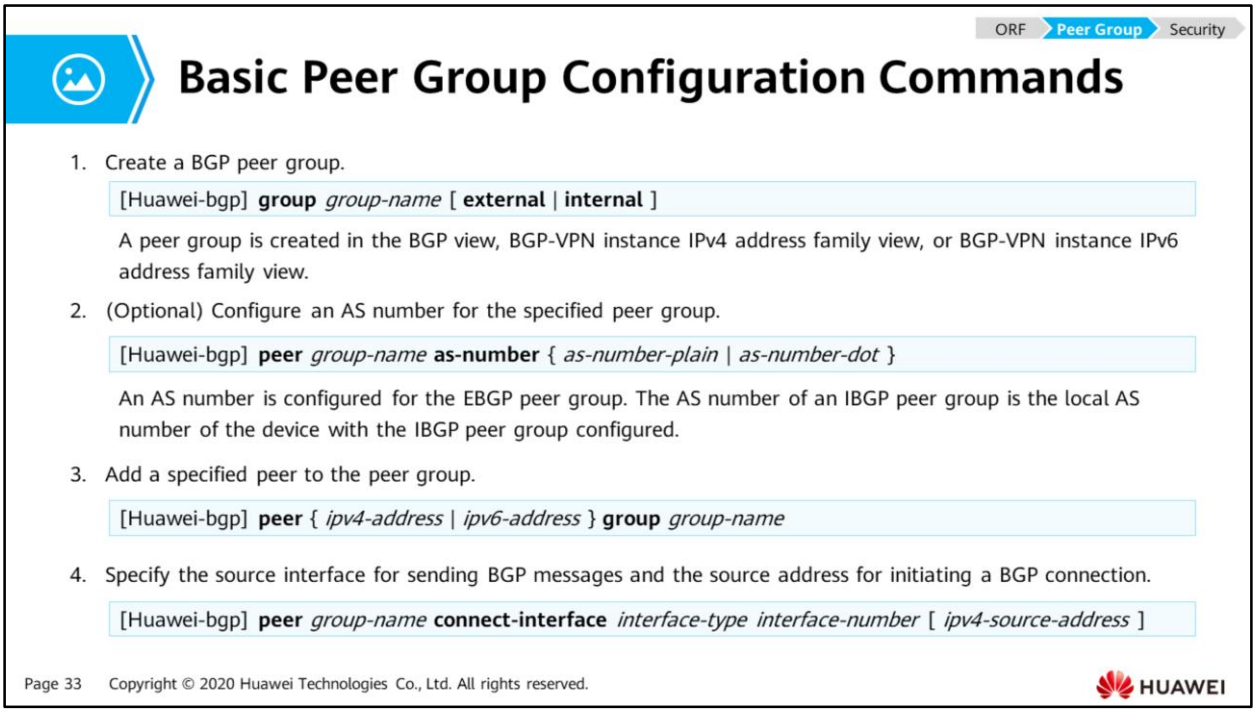
Check the BGP routing information on R2. The command output shows that R2 has received only the route 10.1.1.1/32.

BGP Peer Group

- A peer group is a set of peers with the same policies. When a peer is added to a peer group, it inherits the configurations of the peer group. If the configurations of the peer group change, the configurations of all the peers in the group change accordingly.
- A large number of BGP peers may exist on a large-scale BGP network, many of which need the same policies, you can configure a peer group to simplify configuration.

HUAWEI

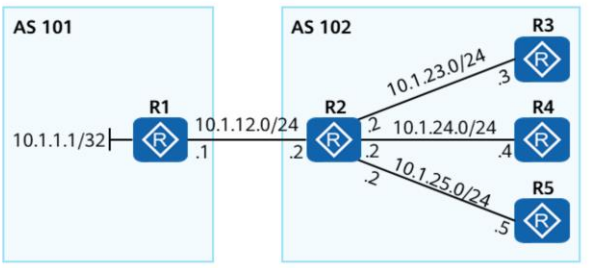
- Each peer in a peer group can be configured with its own policies for route advertisement and acceptance.



Command: [Huawei-bgp] **group** *group-name* [**external** | **internal**]

- *group-name*: specifies the name of a peer group. The value is a string of 1 to 47 case-sensitive characters. If spaces are used, the string must start and end with double quotation marks (").
- **external**: creates an EBGP peer group.
- **internal**: creates an IBGP peer group.

Example for Configuring a BGP Peer Group



After R2 (ASBR) receives a route from its EBGP peer (R1), it advertises the route to all its IBGP peers (R3, R4, and R5). If R2 supports the BGP peer group function, its BGP forwarding performance will be greatly improved.

Device	Interface	Interface Address	Device	Interface	Interface Address
R1	Loopback0	10.1.1.1/32	R4	Loopback0	10.1.4.4/32
R2	Loopback0	10.1.2.2/32	R5	Loopback0	10.1.5.5/32
R3	Loopback0	10.1.3.3/32	/		

1. Complete the basic BGP configuration on R1.

```
[R1] bgp 101
[R1-bgp] peer 10.1.12.2 as-number 102
[R1-bgp] network 10.1.1.1 32
```

2. Complete the basic EBGP configuration and basic IBGP peer group configuration on R2.

```
[R2] bgp 102
[R2-bgp] peer 10.1.12.1 as-number 101
[R2-bgp] group in internal
[R2-bgp] peer 10.1.3.3 group in
[R2-bgp] peer 10.1.4.4 group in
[R2-bgp] peer 10.1.5.5 group in
[R2-bgp] peer in connect-interface Loopback 0
```

3. Complete the basic IBGP configuration on R3.

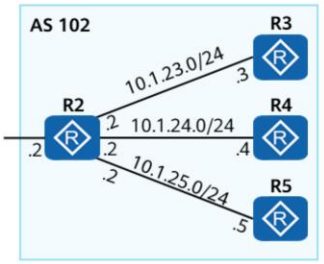
```
[R3] bgp 102
[R3-bgp] peer 10.1.2.2 as-number 102
[R3-bgp] peer 10.1.2.2 connect-interface Loopback 0
```

The configurations of R4 and R5 are similar to the configuration of R3, and are not provided here.

HUAWEI

- As shown in the figure, assume that static routes are used or OSPF is used to ensure internal network reachability in AS 102. The configuration details are not provided here.

Verifying the BGP Peer Group Configuration



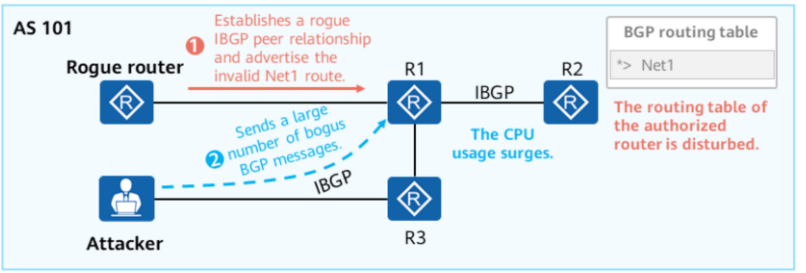
1. Check the BGP peer group information on R2.

```
[R2]display bgp group in
 BGP peer-group: in
 Remote AS: 102
 Authentication type configured: None
 Type : internal
 Configured hold timer value: 180
 Keepalive timer value: 60
 Connect-retry timer value: 32
 Minimum route advertisement interval is 15 seconds
 Connect-interface has been configured
 PeerSession Members:
  10.1.3.3        10.1.4.4        10.1.5.5

 Peer Preferred Value: 0
 No routing policy is configured
 Peer Members:
  Peer      V    AS  MsgRcvd  MsgSent  OutQ  Up/Down   State       PrefRcv
  10.1.3.3  4   102        5        7     0  00:03:33  Established       0
  10.1.4.4  4   102        5        6     0  00:03:11  Established       0
  10.1.5.5  4   102        4        6     0  00:02:52  Established       0
```

The command output shows that the peer group created on R2 is named in and has three IBGP peer members: 10.1.3.3, 10.1.4.4, and 10.1.5.5. All these members have established peer relationships with R2.

BGP Security

- Common BGP attacks are as follows:
  - A rogue BGP peer relationship is established, and invalid routes are advertised to disturb the normal routing table.
  - A router receives a large number of bogus BGP messages and sends them to the CPU. As a result, the CPU usage goes excessively high.

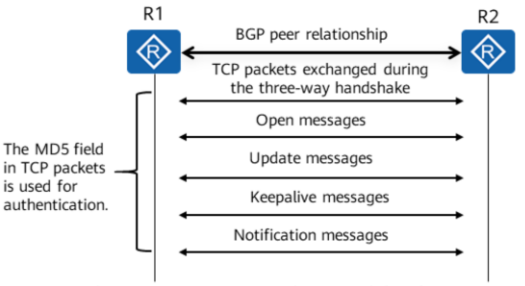


- BGP uses authentication and Generalized TTL Security Mechanism (GTSM) to ensure the security of message exchange between BGP peers.

BGP Authentication

BGP authentication is classified as MD5 authentication or keychain authentication. Authenticating BGP peer relationships can prevent rogue BGP peer relationships from being established.

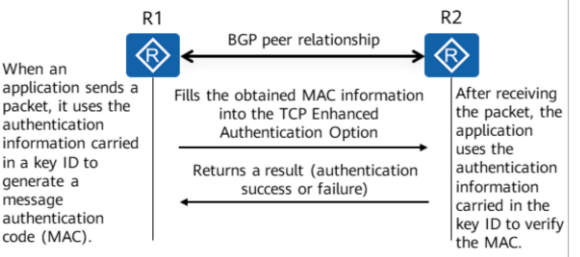
MD5 Authentication



BGP uses the Transmission Control Protocol (TCP) as its transport layer protocol. To enhance BGP security, MD5 authentication can be implemented when a TCP connection is established. You can set an MD5 authentication password for a TCP connection so that TCP implements MD5 authentication for BGP.

Keychain Authentication

- Keychain is a set of key IDs. A key ID uniquely identifies authentication information, including the authentication password and algorithm.



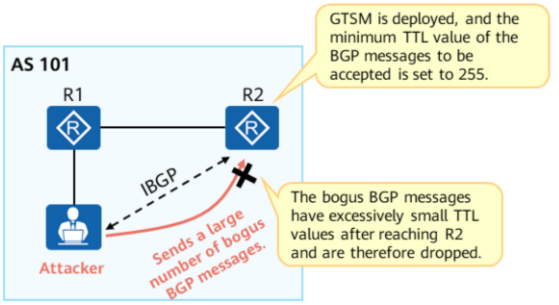
- You must configure keychain authentication for TCP-based applications on both BGP peers. Note that encryption algorithms and passwords configured for keychain authentication on both peers must be the same; otherwise, a TCP connection cannot be set up between BGP peers and BGP messages cannot be exchanged.

HUAWEI

- BGP uses TCP as its transport layer protocol and considers a TCP packet valid only if the source IP address, destination IP address, source port number, destination port number, and TCP sequence number in the packet are correct. Most of the preceding parameters in a TCP packet can be easily obtained by attackers. To protect BGP from attacks, use MD5 authentication or keychain authentication between BGP peers to reduce the possibility of attacks.
  - The MD5 algorithm is easy to configure and generates a single password, which can only be manually changed.
  - The keychain algorithm is complex to configure and generates a set of passwords. Keychain authentication allows passwords to be changed automatically based on configurations. Therefore, keychain authentication is applicable to networks requiring high security.
- Note: BGP MD5 authentication and BGP keychain authentication are mutually exclusive.

BGP GTSM

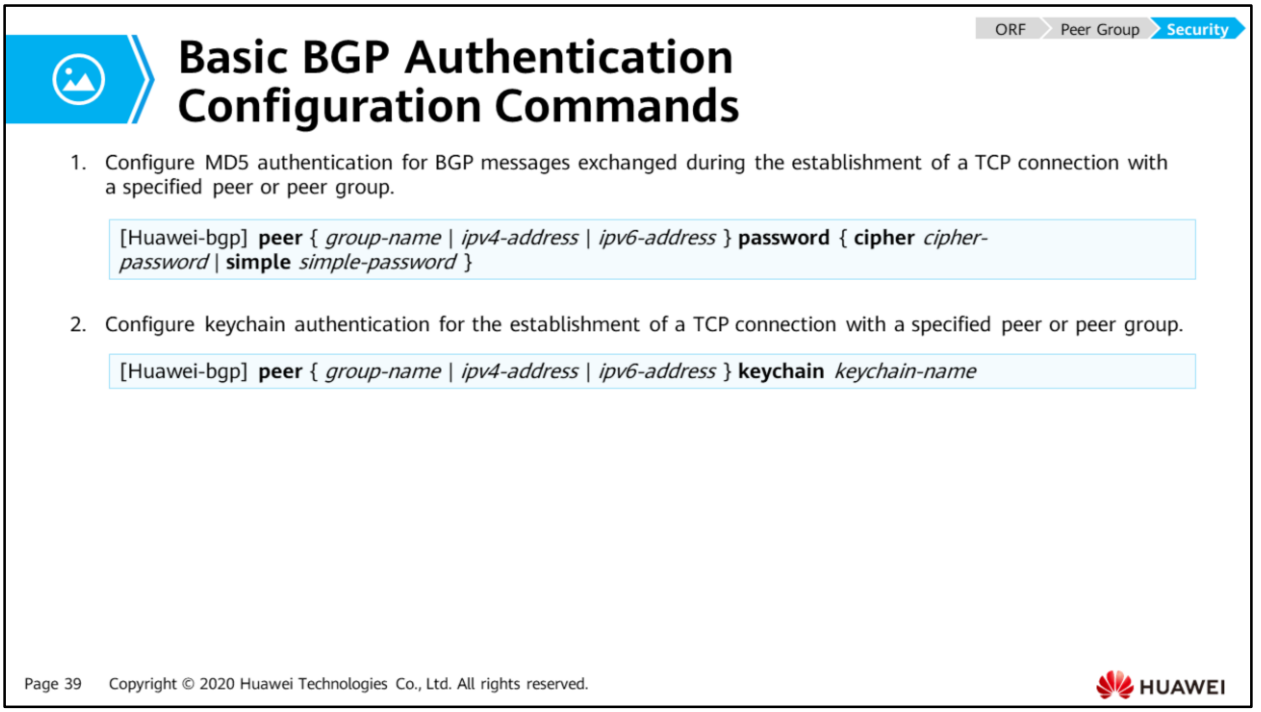
BGP GTSM can check whether the time to live (TTL) value in the IP packet header is within a preset range, and drop the packets whose TTL values are not within the preset range. This prevents bogus BGP messages from attacking the device.



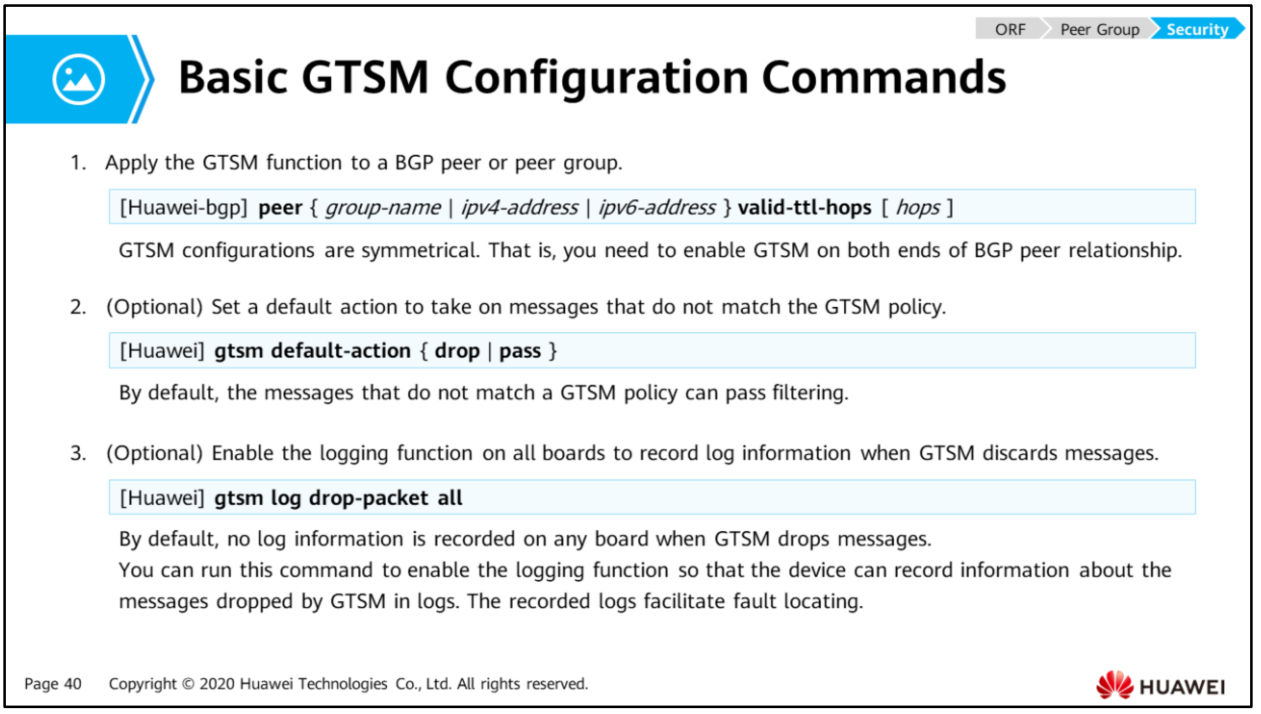
- When an attacker continuously sends bogus BGP messages to attack R2, the TTL values of such messages are smaller than 255.
- If BGP GTSM is enabled on R2 and the valid TTL range for messages sent by the IBGP peer is set to [255, 255], the system checks the TTL values of all BGP messages and drops the bogus messages whose TTL values are smaller than 255. This prevents the bogus messages from consuming CPU resources.

HUAWEI

- As shown in the figure, if BGP GTSM is not enabled, the device finds that the received numerous bogus BGP messages are destined for itself, and directly sends them to the control plane for processing. As a result, the control plane has to process a large number of bogus messages, causing the CPU usage to go excessively high and the system to be unexpectedly busy.

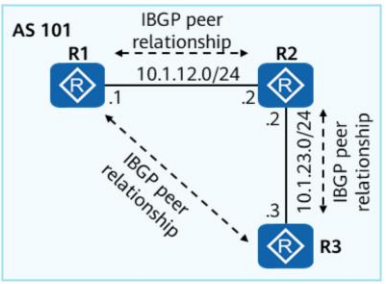


- Command: [Huawei-bgp] **peer** { *group-name* | *ipv4-address* | *ipv6-address* } **keychain** *keychain-name*
  - *keychain-name*: specifies the name of a keychain. The value is a string of 1 to 47 case-insensitive characters. It cannot contain question marks (?). If spaces are used, the string must start and end with double quotation marks (").



- Command: [Huawei-bgp] **peer** { *group-name* | *ipv4-address* | *ipv6-address* } **valid-ttl-hops** [*hops*]
  - *hops*: specifies the number of TTL hops to be checked. The value is an integer ranging from 1 to 255. The default value is 255. If you specify *hops*, the valid range of TTL values in the messages to be checked is [255 – *hops* + 1, 255].
- Command: [Huawei] **gtsm default-action** { **drop** | **pass** }
  - **drop**: indicates that the messages that do not match the GTSM policy cannot pass filtering and are dropped.
  - **pass**: indicates that the messages that do not match the GTSM policy can pass filtering.
- Command: [Huawei] **gtsm log drop-packet all**
  - **all**: indicates all boards.

Example for Configuring GTSM (1)



R1, R2, and R3 all belong to AS 101 and use loopback0 interfaces to establish full-mesh IBGP connections. GTSM needs to be enabled on them to prevent CPU attacks.

Device	Interface	Interface Address
R1	Loopback0	10.1.1.1/32
R2	Loopback0	10.1.2.2/32
R3	Loopback0	10.1.3.3/32

1. Establish full-mesh IBGP connections.

```
[R1] bgp 101
[R1-bgp] peer 10.1.2.2 as-number 101
[R1-bgp] peer 10.1.2.2 connect-interface Loopback 0
[R1-bgp] peer 10.1.3.3 as-number 101
[R1-bgp] peer 10.1.3.3 connect-interface Loopback 0
[R1-bgp] network 10.1.1.1 32
```

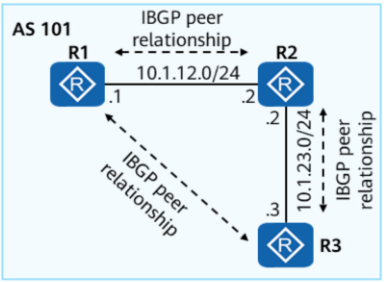
```
[R2] bgp 101
[R2-bgp] peer 10.1.1.1 as-number 101
[R2-bgp] peer 10.1.1.1 connect-interface Loopback 0
[R2-bgp] peer 10.1.3.3 as-number 101
[R2-bgp] peer 10.1.3.3 connect-interface Loopback 0
```

```
[R3] bgp 101
[R3-bgp] peer 10.1.1.1 as-number 101
[R3-bgp] peer 10.1.1.1 connect-interface Loopback 0
[R3-bgp] peer 10.1.2.2 as-number 101
[R3-bgp] peer 10.1.2.2 connect-interface Loopback 0
```


HUAWEI

- As shown in the figure:
  - Assume that static routes are used or OSPF is used to ensure internal network reachability in AS 101. The configuration details are not provided here.
  - R1 advertises the route destined for the IP address of its loopback0 interface to the BGP routing table.

Example for Configuring GTSM (2)



R1, R2, and R3 all belong to AS 101 and use loopback0 interfaces to establish full-mesh IBGP connections. GTSM needs to be enabled on them to prevent CPU attacks.

Device	Interface	Interface Address
R1	Loopback0	10.1.1.1/32
R2	Loopback0	10.1.2.2/32
R3	Loopback0	10.1.3.3/32

2. Enable GTSM between R1 and R2. As the two routers are directly connected, the valid TTL range of the messages from one router to the other is [255, 255]. In this case, the value of **valid-ttl-hops** is 1.

```
[R1-bgp] peer 10.1.2.2 valid-ttl-hops 1
```

```
[R2-bgp] peer 10.1.1.1 valid-ttl-hops 1
```

3. Enable GTSM between R2 and R3. As the two routers are directly connected, the valid TTL range of the messages from one router to the other is [255, 255]. In this case, the value of **valid-ttl-hops** is 1.

```
[R2-bgp] peer 10.1.3.3 valid-ttl-hops 1
```

```
[R3-bgp] peer 10.1.2.2 valid-ttl-hops 1
```

4. Enable GTSM between R1 and R3. As the two routers are connected through R2, the valid TTL range of the messages from one end to the other is [254, 255]. In this case, the value of **valid-ttl-hops** is 2.

```
[R1-bgp] peer 10.1.3.3 valid-ttl-hops 2
```

```
[R3-bgp] peer 10.1.1.1 valid-ttl-hops 2
```

Verifying the GTSM Configuration

```
[R1]display bgp peer 10.1.3.3 verbose
  BGP Peer is 10.1.3.3,  remote AS 101
  Type: IBGP link
  BGP version 4, Remote router ID 10.1.3.3
  Update-group ID: 1
  BGP current state: Established, Up for 00h02m17s
  BGP current event: KATimerExpired
  BGP last state: OpenConfirm
  BGP Peer Up count: 1
  Received total routes: 0
  Received active routes total: 0
  Advertised total routes: 1
  Port:  Local - 179          Remote - 51077
  Configured: Connect-retry Time: 32 sec
  Configured: Active Hold Time: 180 sec  Keepalive Time:60 sec
  Received  : Active Hold Time: 180 sec
  Negotiated: Active Hold Time: 180 sec  Keepalive Time:60 sec
  Peer optional capabilities:
  Peer supports bgp multi-protocol extension
  Peer supports bgp route refresh capability
  Peer supports bgp 4-byte-as capability
```

To be continued on the right

```
Received: Total 5 messages
                        Update messages     0
                        Open messages       1
                        KeepAlive messages  3
                        Notification messages 0
                        Refresh messages    1
Sent: Total 8 messages
                        Update messages     2
                        Open messages       2
                        KeepAlive messages  3
                        Notification messages 0
                        Refresh messages    1
Authentication type configured: None
Last keepalive received: 2020/06/22 17:34:13 UTC-08:00
Last keepalive sent   : 2020/06/22 17:34:13 UTC-08:00
Last update   sent    : 2020/06/22 17:34:02 UTC-08:00
Minimum route advertisement interval is 15 seconds
Optional capabilities:
Route refresh capability has been enabled
4-byte-as capability has been enabled
Connect-interface has been configured
GTSM has been enabled, valid-ttl-hops: 2
Peer Preferred Value: 0
Routing policy configured:
No routing policy is configured
```

GTSM is enabled on R1, and the number of valid hops between R1 and its IBGP peer 10.1.3.3 (R3) is 2.

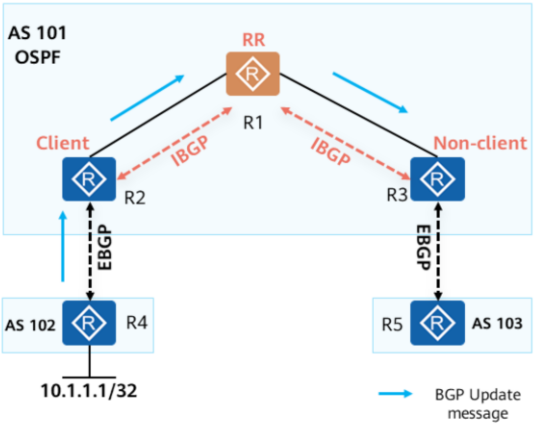
Run the **display bgp peer** command to check BGP peer information.

By specifying the **verbose** parameter, you can check information about the BGP timer, numbers of received and sent routes, capabilities supported by the peer, and enabled functions.

Contents

RR

Using RRs can avoid the need for full-mesh IBGP connections and reduce the burden on the network and CPU.



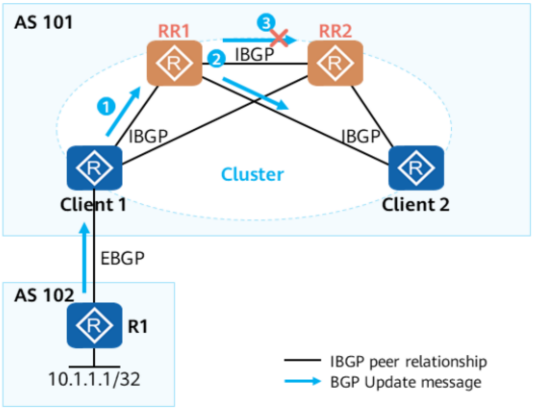
- After an RR is introduced, three roles exist:
  - Route reflector
  - Client
  - Non-client
- The RR reflects the learned routes so that IBGP routes are advertised in the AS, avoiding the need to establish full-mesh IBGP connections.
- After the RR receives routes from its peers, it selects the optimal route based on BGP route selection rules and advertises the optimal route to IBGP peers based on certain rules.

HUAWEI

- RR-related roles:
  - RR: BGP device that reflects the routes learned from an IBGP peer to other IBGP peers. An RR is similar to the designated router (DR) on an OSPF network.
  - Client: IBGP peer whose routes are reflected by the RR to other IBGP peers. In an AS, clients only need to be directly connected to the RR.
  - Non-client: IBGP device that is neither an RR nor a client. In an AS, full-mesh connections still must be established between non-clients and RRs, and between all non-clients.
  - Originator: device that originates routes in an AS. The Originator_ID attribute is used to prevent routing loops in a cluster.
  - Cluster: a set of RRs and their clients. The Cluster_List attribute is used to prevent routing loops between clusters.
- When configuring a BGP router as an RR, you also need to specify a client of the RR. A client does not need to be configured because it is not aware that an RR exists on the network.
- Rules for an RR to advertise routes:
  - After learning routes from non-clients, the RR selects and advertises the optimal route to all its clients.
  - After learning routes from clients, the RR selects and advertises the optimal route to all its non-clients and clients (except the originating client).
  - After learning routes learned from EBGP peers, the RR selects and advertises the optimal route to all its clients and non-clients.

Common Networking Modes: Backup RR Networking

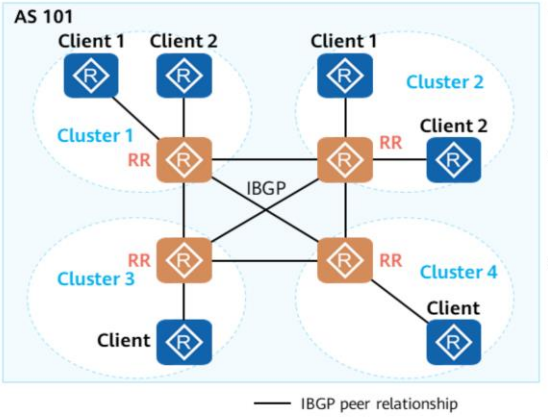
- To improve network reliability and prevent a single point of failure (SPOF) from affecting the network, multiple RRs need to be configured in a cluster.
- Routers on a forwarding path establish IBGP peer relationships with all RRs. Any of the RRs has complete BGP routes.



- RR1 and RR2 are in the same cluster and configured with the same cluster ID.
- Route reflection in single-cluster RR networking (RR1 is used as an example):
  1. When client 1 receives an updated route from an EBGP peer, it advertises this route to RR1 and RR2 through IBGP peer relationships.
  2. After RR1 and RR2 receive this route, they add the local cluster ID to the top of the cluster list of the route, and then reflect the route to client 2 and to each other.
  3. After RR1 and RR2 receive the reflected route, they check the cluster list of the route, finding that the cluster list contains their local cluster ID. RR1 and RR2 discard this route to prevent routing loops.

Common Networking Modes: Multi-Cluster RR Networking (1)

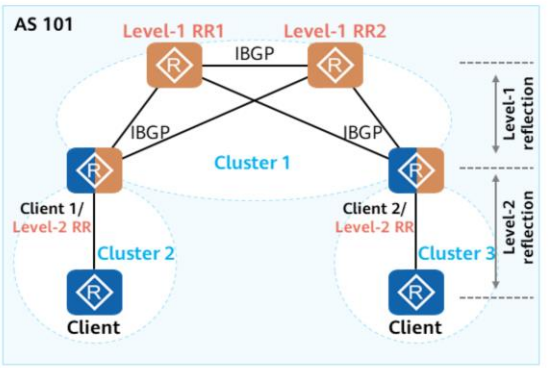
- If multiple clusters exist in an AS, RRs of the clusters establish IBGP peer relationships with each other.
- When RRs reside at the same network layer, RRs of different clusters can establish full-mesh connections to implement flat RR.



- An AS on a backbone network may be divided into multiple clusters. RRs of the clusters are non-clients of each other and establish full-mesh IBGP connections with each other.
- Although each client in a cluster establishes an IBGP connection only with the RR in the same cluster, each RR and client can receive all the routing information.
- As shown in the figure, four RRs reside in cluster 1, cluster 2, cluster 3, and cluster 4. IBGP connections are established between the four RRs. Each client in a cluster establishes an IBGP connection only with the RR in the same cluster.

Common Networking Modes: Multi-Cluster RR Networking (2)

- If multiple clusters exist in an AS, RRs of the clusters establish IBGP peer relationships with each other.
- When the RRs reside at different network layers, the RRs at the lower network layer can be configured as clients to implement hierarchical RR.

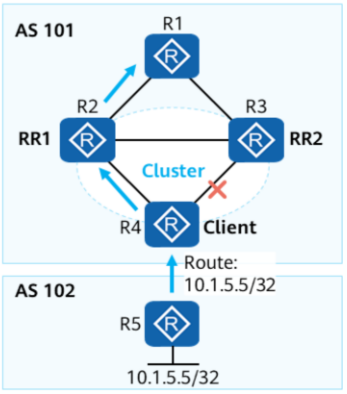


- In practice, hierarchical RR deployment is used more widely.
- As shown in the figure, AS 101 is divided into three clusters:
  - The four devices in cluster 1 are core routers and work in master/backup mode to ensure high reliability. Two Level-1 RRs are deployed in cluster 1, and the other two routers function as clients of the RRs.
  - A Level-2 RR is deployed in each of cluster 2 and cluster 3. The Level-2 RRs are also clients of the Level-1 RRs. An IBGP connection does not need to be established between the Level-2 RRs.

- The route advertisement rules for hierarchical RR networking are the same as those for single-cluster RR networking.
- The following factors need to be considered for hierarchical RR design:
  - Size of the top-layer full-mesh topology: If the number of full-mesh IBGP connections has exceeded the management capacity, hierarchical RR networking can be deployed.
  - Number of alternate paths: This factor affects load balancing and resource consumption. More layers reduce the number of links for load balancing but require fewer router resources.

Single-Cluster Problem

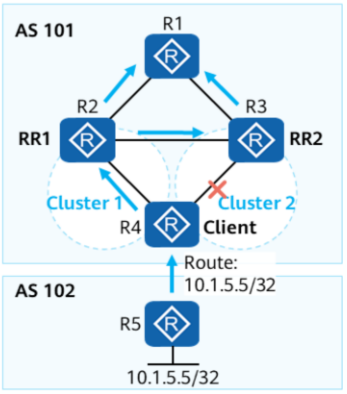
To provide the desired redundancy in an RR-based architecture, it is important to properly divide an AS into clusters.



- Scenario description:
  - As shown in the figure, AS 101 uses the backup RR networking. RR1 and RR2 use the same cluster ID, and provide a redundant link for traffic from R1 to access 10.1.5.5/32.
  - After R4 advertises the route 10.1.5.5/32, the two RRs advertise the route to R1 and to each other. Because RR1 and RR2 have the same cluster ID, the Update messages exchanged between the RRs are discarded.
- An IBGP session failure causes the redundancy failure:
  - Assume that the IBGP session between R3 and R4 fails (for example, due to incorrect configurations). As R3 ignores the route 10.1.5.5/32 advertised by R2, no redundant link is available for traffic from R1 to access 10.1.5.5/32.

Multi-Cluster Design

Multi-cluster design not only provides physical redundancy in response to link failures, but also provides logical redundancy in response to IBGP session failures between clients and RRs.



- As shown in the figure, R2 and R4 are added to cluster 1, and R3 and R4 are added to cluster 2.
- If the IBGP session between R3 and R4 fails, R3 can still forward traffic because R3 learns the route 10.1.5.5/32 advertised by R2.

Quiz

1. (Single) Which of the following AS_Path value is matched against the **ip as-path-filter 1 permit ^(100|200)$** command? ( )
   A. AS_Path 100
   B. AS_Path 200
   C. AS_Path 100 200
   D. AS_Path 100 or AS_Path 200
2. (TorF) BGP GTSM can prevent the establishment of rogue BGP peer relationships. ( )
   A. True
   B. False
3. (TorF) In the backup RR networking of BGP, the master and backup RRs discard the routes reflected by each other to prevent routing loops. ( )
   A. True
   B. False

1. D
2. B
3. A

Summary

- The AS_Path and community filters are dedicated route matching tools of BGP. You can use them to match BGP routes based on the AS_Path and community attributes, respectively. After routing policies are applied to the matched routes, BGP route control can be implemented.
- BGP also supports various advanced features, including ORF to achieve on-demand route advertisement, peer group to simplify configurations, and BGP security features to prevent the establishment of rogue peer relationships as well as attacks from bogus BGP messages.
- BGP uses RRs to break the split horizon rule of IBGP, avoid the need for full-mesh IBGP connections, and reduce the burden on the network and CPU. Common RR networking modes include backup RR networking and multi-cluster RR networking.


Thank You
www.huawei.com