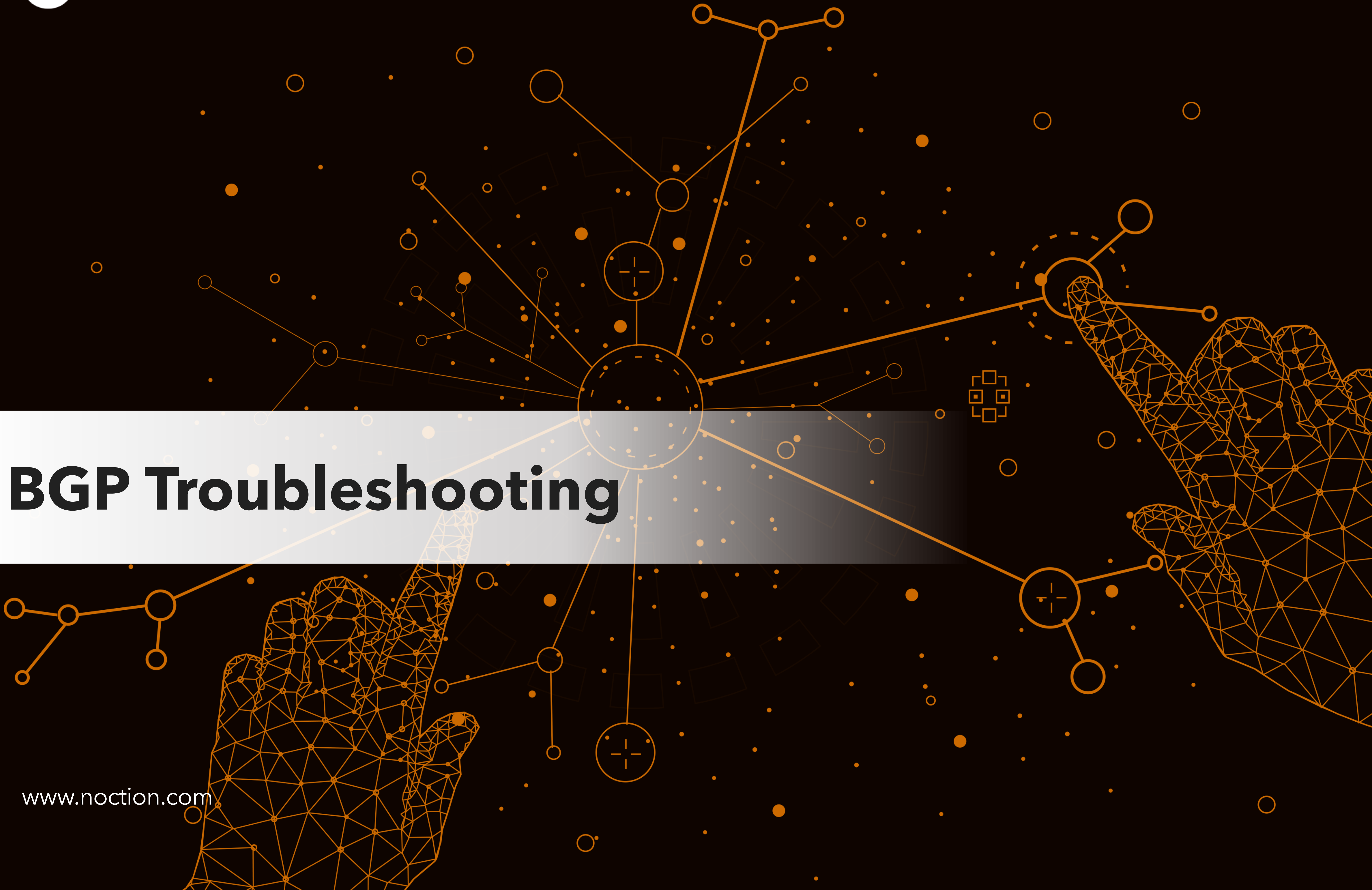
NOCTION
NETWORK INTELLIGENCE



BGP Troubleshooting

www.noction.com

Table of Contents

Introduction

BGP, or Border Gateway Protocol, is a critical and complex protocol for exchanging routing information among various autonomous systems (ASes) on the Internet. Troubleshooting BGP issues requires careful analysis and diagnosis of potential problems across multiple network components.

Effective troubleshooting of BGP issues requires a calm and organized approach. One of the best ways to stay organized is to create a checklist and work through it systematically, starting with layer 1 and progressing to the subsequent layers (layer 2, layer 3, etc.). This approach ensures that each layer is thoroughly investigated and eliminates the possibility of overlooking any potential issues.

The first step in troubleshooting is to identify the root cause of the problem. The root cause could be a misconfiguration or something as simple as a typo or incorrect command. Careful analysis of the issue and understanding of the underlying BGP concepts are necessary to identify the root cause.

When resolving the problem, avoiding randomly executing commands across the network is recommended. This approach can cause further complications and may result in more problems. Instead, the focus should be on one area at a time, dealing with one issue and learning about the problem gradually. This approach is particularly important when tackling unfamiliar issues as it allows for a more thorough understanding of the problem and ensures that the issue is resolved effectively.

1. Verifying BGP Peering Status

BGP peering is the process of establishing a connection between two BGP routers to exchange routing information. The first step in troubleshooting BGP issues is to verify the peering status to ensure that the BGP session is up so the routes can be exchanged between the two routers (Figure 1).

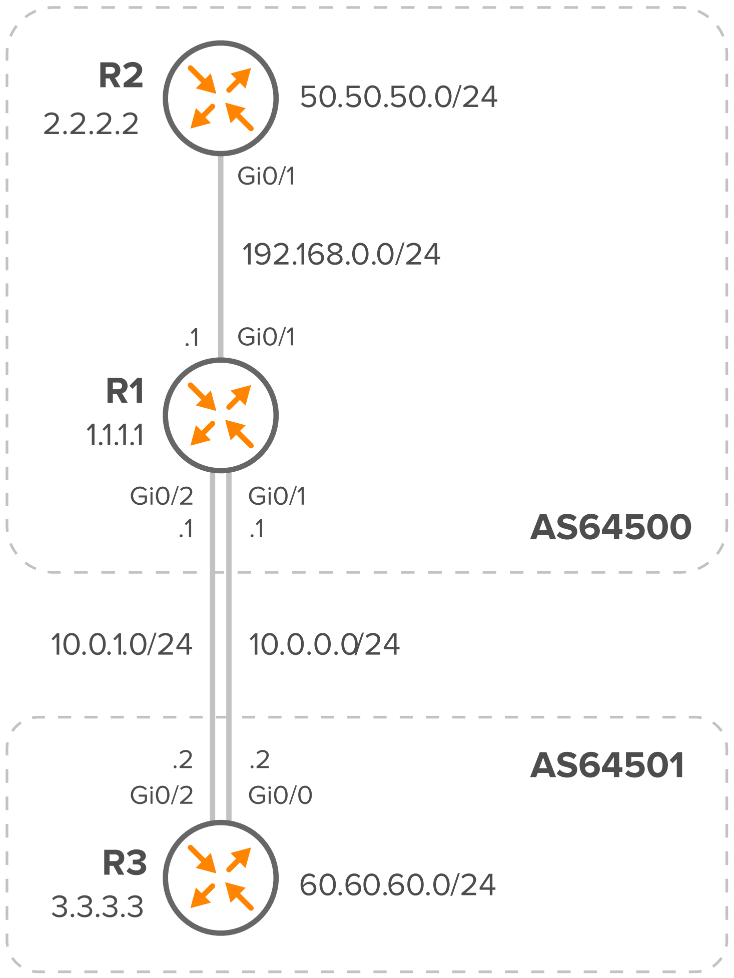
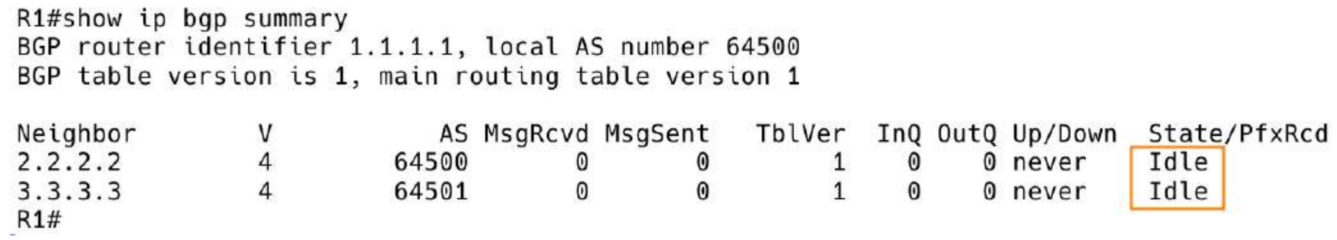


Figure 1 - *Network Topology*

To check the status of all BGP neighbors, use the show ip bgp summary command (Figure 2).



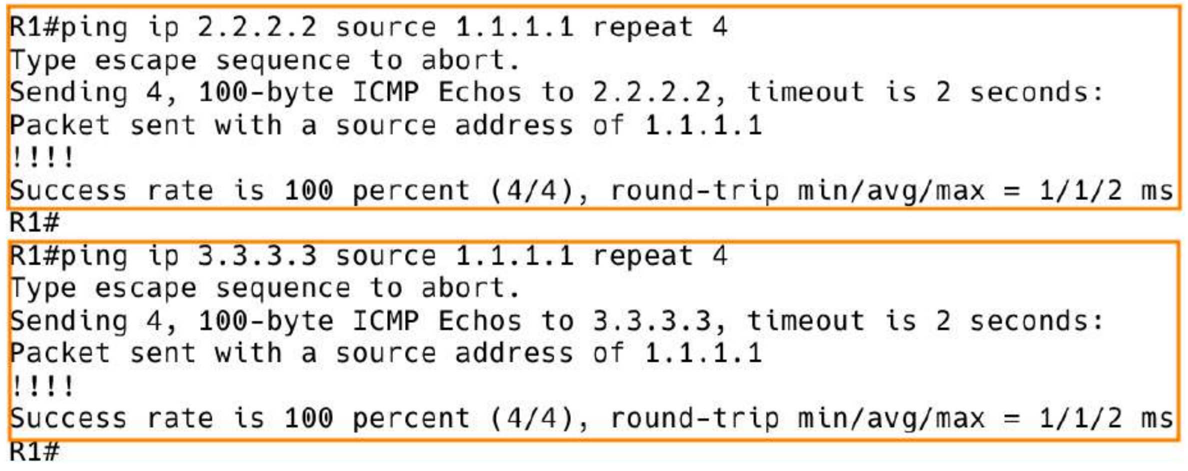
```
R1#show ip bgp summary
BGP router identifier 1.1.1.1, local AS number 64500
BGP table version is 1, main routing table version 1

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2.2.2.2         4        64500       0       0        1    0    0 never    Idle
3.3.3.3         4        64501       0       0        1    0    0 never    Idle
R1#
```

Figure 2 - *Checking BGP Peering Status*

The BGP sessions are currently down, and both peers are in an idle state. When BGP is in the idle state, it is able to detect a start event, initiate a TCP connection to the BGP peer, and listen for a new connection from a peer router.

To check the connectivity between routers when peering is established between loopback interfaces, a loopback-to-loopback ping must be done (Figure 3).



```
R1#ping ip 2.2.2.2 source 1.1.1.1 repeat 4
Type escape sequence to abort.
Sending 4, 100-byte ICMP Echos to 2.2.2.2, timeout is 2 seconds:
Packet sent with a source address of 1.1.1.1
!!!!
Success rate is 100 percent (4/4), round-trip min/avg/max = 1/1/2 ms
R1#
R1#ping ip 3.3.3.3 source 1.1.1.1 repeat 4
Type escape sequence to abort.
Sending 4, 100-byte ICMP Echos to 3.3.3.3, timeout is 2 seconds:
Packet sent with a source address of 1.1.1.1
!!!!
Success rate is 100 percent (4/4), round-trip min/avg/max = 1/1/2 ms
R1#
```

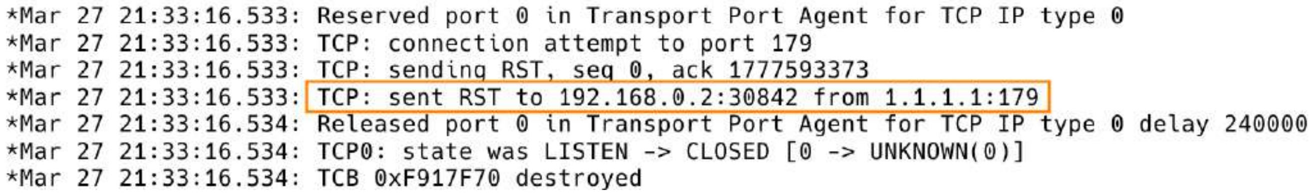
Figure 3 - *Checking Inter-Loopback Connectivity*

If the ping is successful, we must verify locally that the BGP configuration is correct before we hold the peers responsible. It is important to check whether the local AS and remote AS have been assigned correctly. Note that 2.2.2.2 is an iBGP peer, while 3.3.3.3 is an eBGP peer.

```
R1# show running-config | section bgp
router bgp 64500
 bgp log-neighbor-changes
 neighbor 2.2.2.2 remote-as 64500
 neighbor 3.3.3.3 remote-as 64501
 neighbor 3.3.3.3 update-source Loopback0
 !
 address-family ipv4
  neighbor 2.2.2.2 activate
  neighbor 3.3.3.3 activate
 exit-address-family
```

Running the debug ip tcp transactions command reveals that the peer is attempting to initiate a TCP session from the IP address 192.168.0.2 to R1 (as shown in Figure 4). Please note that the source address is that of the outgoing interface towards the destination, but since the peering, in this case, is using loopback interfaces, iBGP fails to establish.

```
R1# debug ip tcp transactions
```



```
*Mar 27 21:33:16.533: Reserved port 0 in Transport Port Agent for TCP IP type 0
*Mar 27 21:33:16.533: TCP: connection attempt to port 179
*Mar 27 21:33:16.533: TCP: sending RST, seq 0, ack 1777593373
*Mar 27 21:33:16.533: TCP: sent RST to 192.168.0.2:30842 from 1.1.1.1:179
*Mar 27 21:33:16.534: Released port 0 in Transport Port Agent for TCP IP type 0 delay 240000
*Mar 27 21:33:16.534: TCP0: state was LISTEN -> CLOSED [0 -> UNKNOWN(0)]
*Mar 27 21:33:16.534: TCB 0xF917F70 destroyed
```

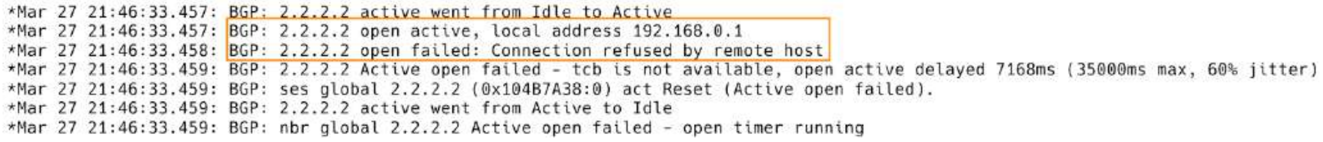
Figure 4 - *Sending RST Packet to Peer 192.168.0.2*

To specify the loopback interface as the outgoing interface, we should add the update-source interface loopback0 statement to the neighbor configuration on R2.

```
R2# show running-config | s bgp
router bgp 64500
 neighbor 1.1.1.1 remote-as 64500
 neighbor 1.1.1.1 update-source Loopback0
```

However, when we run the debug ip bgp command, we can see that R1 is also attempting to establish a TCP session with R2 from the Gi0/1 interface with IP address 192.168.0.1 (as shown in Figure 5).

```
R1# debug ip bgp
```



```
*Mar 27 21:46:33.457: BGP: 2.2.2.2 active went from Idle to Active
*Mar 27 21:46:33.457: BGP: 2.2.2.2 open active, local address 192.168.0.1
*Mar 27 21:46:33.458: BGP: 2.2.2.2 open failed: Connection refused by remote host
*Mar 27 21:46:33.459: BGP: 2.2.2.2 Active open failed - tcb is not available, open active delayed 7168ms (35000ms max, 60% jitter)
*Mar 27 21:46:33.459: BGP: ses global 2.2.2.2 (0x104B7A38:0) act Reset (Active open failed).
*Mar 27 21:46:33.459: BGP: 2.2.2.2 active went from Active to Idle
*Mar 27 21:46:33.459: BGP: nbr global 2.2.2.2 Active open failed - open timer running
```

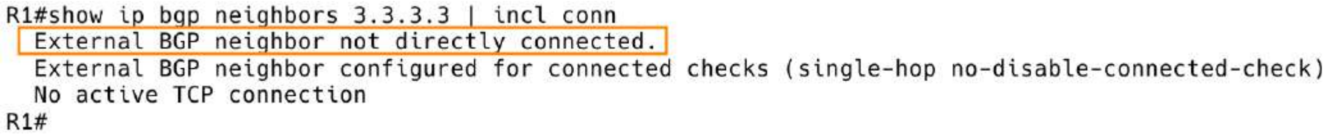
Figure 5 - *TCP Session to R2 from R1 is Utilizing the IP address of Gi0/1 Interface*

The BGP configuration on R1 is missing the "update-source" command for the R2 peer. To resolve this issue, we need to add a statement that specifies the loopback 0 interface to be used for TCP connections.

```
R1# show running-config | s bgp
router bgp 64500
 neighbor 2.2.2.2 remote-as 64500
 neighbor 2.2.2.2 update-source Loopback0
```

Currently, the eBGP session is still not up. This is because eBGP peers are typically directly connected, so the IP address of the outgoing interface is used. However, in our dual-homed design, we have two links to R3 that are of equal cost, and we're using loopback interfaces to achieve load sharing between these links.

The eBGP peer R3 is not directly connected, as shown in Figure 6. It's important to note that if the hop count is not specified, the default TTL value for iBGP sessions is 255, while the default TTL value for eBGP sessions is 1.



```
R1#show ip bgp neighbors 3.3.3.3 | incl conn
  External BGP neighbor not directly connected.
  External BGP neighbor configured for connected checks (single-hop no-disable-connected-check)
  No active TCP connection
R1#
```

Figure 6 - *eBGP Peers R1 and R2 Peers not Directly Connected*

To establish BGP neighborship with an external BGP (eBGP) peer that is not on the same network address, we must configure the ebgp-multihop command. In our scenario, the eBGP neighbor 3.3.3.3 is up to 2 hops away.

```
R1# show running-config | s bgp
router bgp 64500
 neighbor 3.3.3.3 remote-as 64501
 neighbor 3.3.3.3 ebgp-multihop 2
 neighbor 3.3.3.3 update-source Loopback0
```

2. Verifying Missing Routes

Verifying missing BGP routes typically involves checking the BGP routing table on a router to ensure that it has the expected routes.

BGP monitoring tools, such as **NFA**, BGPmon or ExaBGP, can provide real-time monitoring of BGP sessions and route advertisements. These tools can help identify issues with BGP sessions or route advertisements before they cause routing instability. Use BGP monitoring tools to alert you to potential problems or anomalies in BGP behavior.

So, if a session is established between BGP routers, they start to exchange UPDATE messages to advertise changes in routing information, including new routes or updates to existing routes.

2.1 Route Origination

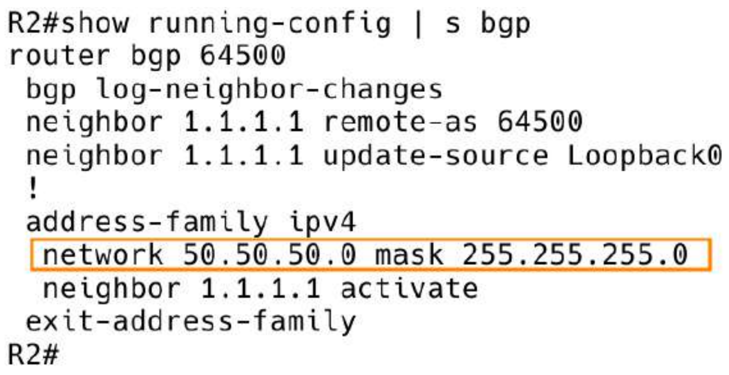
BGP routers only advertise their locally known routes to their neighboring routers, and the best path (the most preferred route based on various attributes such as shortest AS path length, lowest cost, etc.) is selected and advertised to other neighboring routers.

To advertise a prefix in BGP, the prefix must exist in another routing process as well, typically in one of the following ways:

- A static route pointing to a customer's network (for advertising customer routes into your iBGP)
- A static route pointing to Null0 (for advertising aggregates into your eBGP that you don't actually want to route to, as it acts as a blackhole)

2.1.1 Troubleshooting Route Origination

The network statement tells the BGP process to advertise the network 50.50.50.50.0/24 to its BGP neighbors (Figure 7).

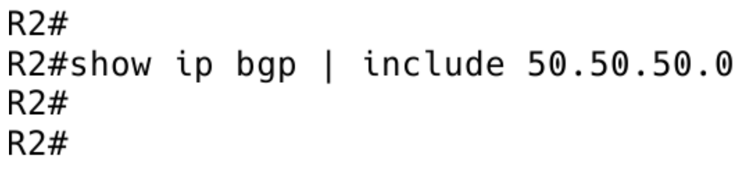


```
R2#show running-config | s bgp
router bgp 64500
 bgp log-neighbor-changes
 neighbor 1.1.1.1 remote-as 64500
 neighbor 1.1.1.1 update-source Loopback0
 !
 address-family ipv4
  network 50.50.50.0 mask 255.255.255.0
  neighbor 1.1.1.1 activate
 exit-address-family
R2#
```

Figure 7 - *Network Statement in BGP Configuration*

However, it seems that BGP is not originating the route 50.50.50.0/24 as shown in Figure 8.

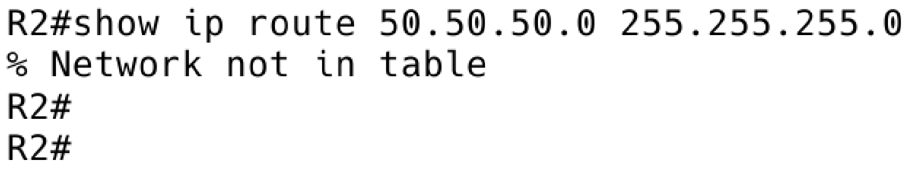
```
R2# show ip bgp | include 50.50.50.0
```



```
R2#
R2#show ip bgp | include 50.50.50.0
R2#
R2#
```

Figure 8 - *R2 is not Advertising Network 50.50.50.0/24 via BGP*

The route 50.50.50.0/24 is missing in the Routing Information Base (RIB) of R2 that is via BGP and is not advertised to the peer R1 via iBGP.



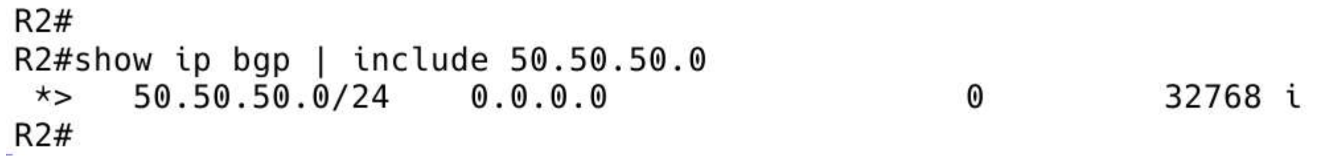
```
R2#show ip route 50.50.50.0 255.255.255.0
% Network not in table
R2#
R2#
```

Figure 9 - *Network 50.50.50.0/24 not in the Routing Table of R2*

To enable R2 to advertise the network via BGP, we must add a static route to its routing information base (RIB), as shown in Figure 10.

```
R2(config)# ip route 50.50.50.0 255.255.255.0 null 0

R2# show ip bgp | include 50.50.50.0
```



```
R2#
R2#show ip bgp | include 50.50.50.0
 *>   50.50.50.0/24    0.0.0.0                  0         32768 i
R2#
```

Figure 10 - *R2 is Originating Route 50.50.50.0/24*

2.2 Update Exchange

It's essential to keep in mind the following:

- When a BGP router receives the best path from an eBGP peer, it will advertise that best path to all its peers, both iBGP and eBGP peers.
- When a BGP router receives the best path from an iBGP peer, it will advertise that best path only to its eBGP peers. This is because BGP routers do not advertise iBGP learned routes to other iBGP peers by default.
- All iBGP routers must be fully meshed, meaning that every iBGP router must have a direct iBGP session with every other iBGP router. This is necessary to ensure that all iBGP learned routes are propagated throughout the iBGP domain.
- If route reflectors are used in the iBGP domain, then the iBGP mesh can be reduced, and iBGP-learned routes can be reflected by the route reflectors to other iBGP peers. This reduces the number of required iBGP sessions and simplifies the configuration.

Troubleshooting commands:

Display routes that we sent to the peer R1. The attribute values shown are taken from the BGP table. As a result, any changes made to the attributes by outbound route-maps will not be visible.

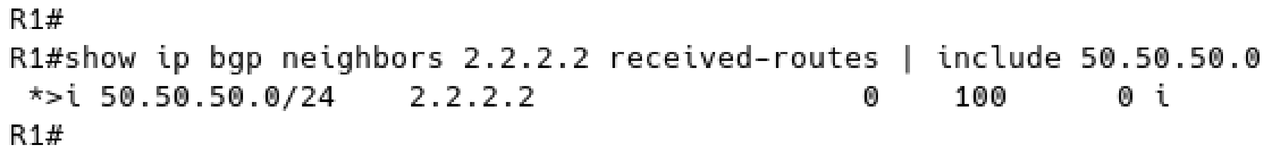
```
R2# show ip bgp neighbor 1.1.1.1 advertised-routes
```

The command show ip bgp neighbor 2.2.2.2 routes on R1 will display the routes sent to us from 2.2.2.2 that have successfully passed through our inbound filters.

```
R1# show ip bgp neighbor 2.2.2.2 routes
```

The command show ip bgp neighbor 2.2.2.2 received will display all routes received from the peer, including those that were denied. Please note that in order to use this command, soft-reconfig inbound needs to be configured.

```
R1# show ip bgp neighbor  2.2.2.2 received-routes
```



```
R1#
R1#show ip bgp neighbors 2.2.2.2 received-routes | include 50.50.50.0
 *>i 50.50.50.0/24    2.2.2.2                  0    100      0 i
R1#
```

Figure 11 - *Received Routers on R1 that Originate from R2*

2.3 BGP Route Filtering

BGP route filtering can be used to limit the set of routes that a router will accept from its BGP peers, as well as limit the set of routes that a router will advertise to its BGP peers.

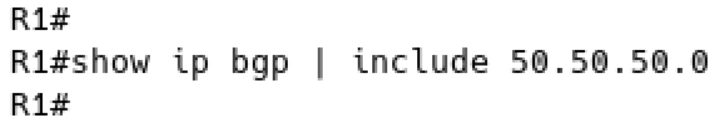
Some of the common types of filters that can be used in BGP route filtering:

- **Prefix filters:** A prefix filter matches and filters routes based on the IP prefix or network address. It can be used to permit or deny specific prefixes or ranges of prefixes.
- **AS_PATH filters:** An AS_PATH filter matches and filters routes based on the AS path attribute of the BGP route. It can be used to permit or deny routes from specific AS numbers or ranges of AS numbers.
- **Community filters:** A community filter matches and filters routes based on the BGP community attribute. Communities are tags that can be attached to BGP routes to mark them for specific treatment by routers. Community filters can be used to permit or deny routes with specific community tags.
- **Route-maps:** A route-map is a more advanced and flexible type of filter that can be used to match and filter routes based on multiple criteria, including prefix, AS path, community, and other attributes. A route-map can also be used to modify or set BGP attributes of routes, as well as control the redistribution of routes between routing protocols.

Use the show ip bgp neighbors x.x.x.x received-routes command to verify the routes being received from the BGP neighbor

2.3.1 Troubleshooting Prefix Filtering

Suppose that R1 is unable to receive the 50.50.50.0/24 route from the R2 peer.

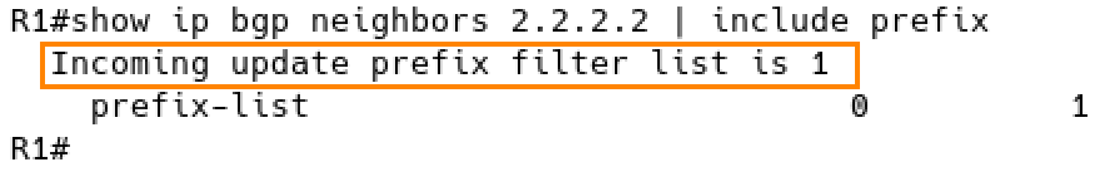


```
R1#
R1#show ip bgp | include 50.50.50.0
R1#
```

Figure 12 - *Missing Prefix 50.50.50.0/24 in BGP Table of R1*

Before we proceed, let's verify whether the prefix-list filter is applied to R2. As you can observe, the prefix list 1 is currently active for R2.

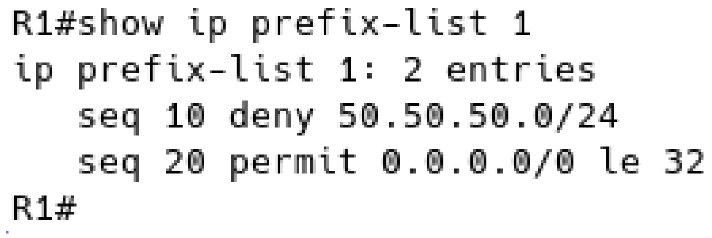
```
R1# show ip bgp neighbors 2.2.2.2 | include prefix
```



```
R1#show ip bgp neighbors 2.2.2.2 | include prefix
  Incoming update prefix filter list is 1
    prefix-list                                    0           1
R1#
```

Figure 13 - *Checking Prefix Filters for R2 on R1*

The first line of prefix-list 1 denies 50.50.0./24/24, and the second line permits 0.0.0.0/0 (all networks).



```
R1#show ip prefix-list 1
ip prefix-list 1: 2 entries
   seq 10 deny 50.50.50.0/24
   seq 20 permit 0.0.0.0/0 le 32
R1#
```

Figure 14 - *Prefix Filter 1*

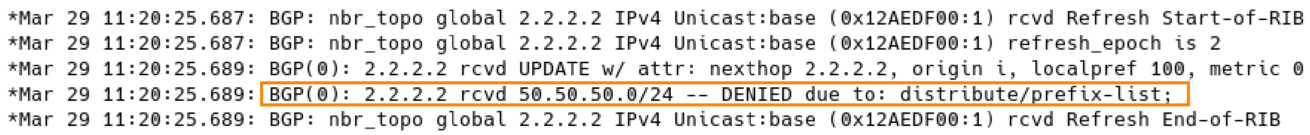
R1 configuration also proves that the prefix 50.5.0.50.0/24 received from BGP peer R2 is blocked by the prefix filter 1 installed on R1.

```
R1# show run | include neighbor 2.2.2.2
 neighbor 2.2.2.2 remote-as 64500
 neighbor 2.2.2.2 update-source Loopback0
  neighbor 2.2.2.2 activate
  neighbor 2.2.2.2 prefix-list 1 in
```

Running a debug command can also be helpful. Begin by creating a standard access list (ACL) that matches the missing prefix on R1:

```
R1(config)# access-list 99 permit 50.50.50.0 0.0.0.255

R1# debug ip bgp updates 99
```

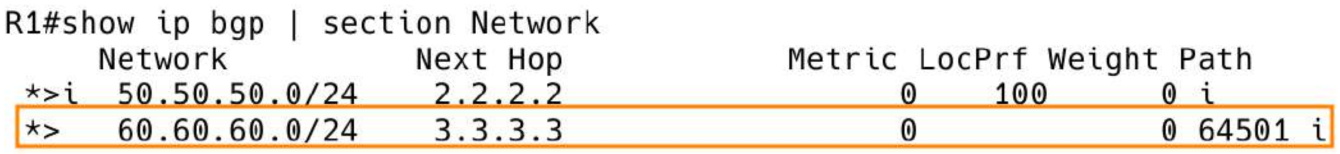


```
*Mar 29 11:20:25.687: BGP: nbr_topo global 2.2.2.2 IPv4 Unicast:base (0x12AEDF00:1) rcvd Refresh Start-of-RIB
*Mar 29 11:20:25.687: BGP: nbr_topo global 2.2.2.2 IPv4 Unicast:base (0x12AEDF00:1) refresh_epoch is 2
*Mar 29 11:20:25.689: BGP(0): 2.2.2.2 rcvd UPDATE w/ attr: nexthop 2.2.2.2, origin i, localpref 100, metric 0
*Mar 29 11:20:25.689: BGP(0): 2.2.2.2 rcvd 50.50.50.0/24 -- DENIED due to: distribute/prefix-list;
*Mar 29 11:20:25.689: BGP: nbr_topo global 2.2.2.2 IPv4 Unicast:base (0x12AEDF00:1) rcvd Refresh End-of-RIB
```

Figure 15 - *Debugging BGP Updates on R1 from R2*

2.4 iBGP Next-Hop

R1 is an edge router of AS 64500, and it is propagating the external BGP prefix 60.60.60.0/24 to the internal BGP peer R2, as illustrated in Figure 16. R1 reports the prefix learned from AS65501 as valid * and the best >.

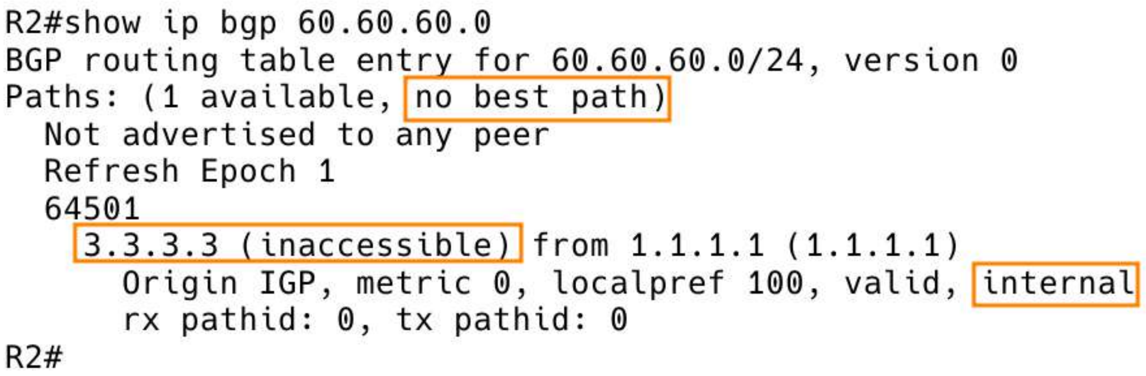


```
R1#show ip bgp | section Network
     Network          Next Hop            Metric LocPrf Weight Path
 *>i  50.50.50.0/24    2.2.2.2                  0    100      0 i
 *>   60.60.60.0/24    3.3.3.3                  0             0 64501 i
```

Figure 16 - *BGP table of R1 with Prefix Received from R3*

R1 follows a loop avoidance rule in which the iBGP does not modify BGP next hop and leaves the Autonomous System Path (AS_PATH) unchanged.

R2 reports the prefix 60.60.60.0//24 learned from R3 as valid (*) and internal (i), but it is not the best path (Figure 17). The route is not inserted into the RIB of router R2 because the next hop 3.3.3.3 is not accessible. R1 preserves the next hop attribute 3.3.3.3 learned from eBGP peer R3.



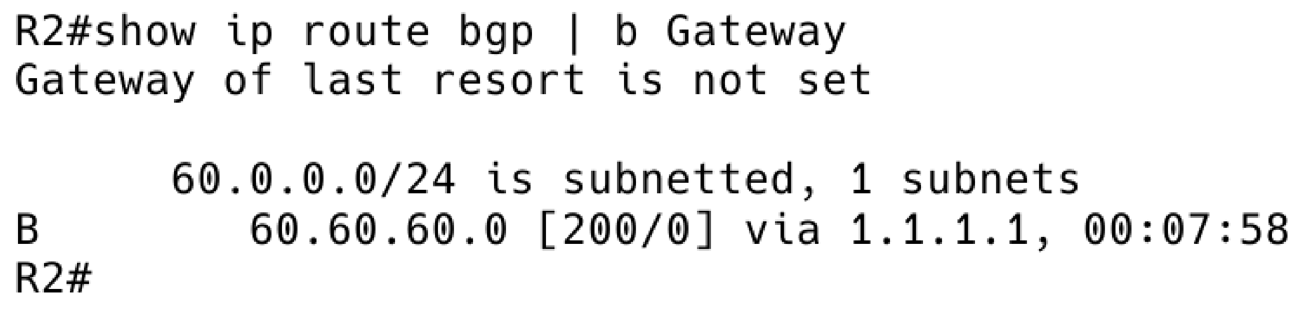
```
R2#show ip bgp 60.60.60.0
BGP routing table entry for 60.60.60.0/24, version 0
Paths: (1 available, no best path)
  Not advertised to any peer
  Refresh Epoch 1
  64501
    3.3.3.3 (inaccessible) from 1.1.1.1 (1.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, internal
      rx pathid: 0, tx pathid: 0
R2#
```

Figure 17 - *BGP table of R2 with Prefix Received from R1*

In order to make sure we can reach the eBGP next hop 3.3.3.3, include the network that the next hop belongs to in the IGP or issue the next-hop-self neighbor command on R1 to force the router to advertise itself, rather than the external peer, as the next hop.

With the missing "next-hop-self" configuration now added to R1, the route 60.60.60.0/24 is successfully installed into the Routing Information Base (RIB) of R2 (Figure 18).

```
R1# show running-config | s bgp
router bgp 64500
 bgp log-neighbor-changes
 neighbor 2.2.2.2 remote-as 64500
 neighbor 2.2.2.2 update-source Loopback0
 neighbor 3.3.3.3 remote-as 64501
 neighbor 3.3.3.3 ebgp-multihop 2
 neighbor 3.3.3.3 update-source Loopback0
 !
 address-family ipv4
  neighbor 2.2.2.2 activate
  neighbor 2.2.2.2 next-hop-self
  neighbor 3.3.3.3 activate
```



```
R2#show ip route bgp | b Gateway
Gateway of last resort is not set

      60.0.0.0/24 is subnetted, 1 subnets
B        60.60.60.0 [200/0] via 1.1.1.1, 00:07:58
R2#
```

Figure 18 - *R2 Routing Table*

NOTE: It is never a good idea to allow external events (such as link flaps in the access network) to affect the stability of your main IGP. Therefore, using next-hop-self on AS edge routers (and changing external next hops to the loopback address of the edge router) is almost always the preferred design.

3. Internet Reachability Issues

3.1 Troubleshooting BGP Configuration for Inbound Traffic Manipulation

To understand why inbound traffic is being routed in a certain way to a local AS, you need to investigate several BGP configurations, such as:

- **AS_PATH prepending**
- **BGP Multi-EXIT Discriminator (MED)**
- **Communities with local preference**

AS_PATH prepending involves adding multiple instances of an AS number to a BGP route's AS_PATH attribute, which can affect the path's attractiveness to BGP peers, thus, the routing of inbound traffic.

Important: Most operators set the AS-PATH limit to 20 or thereabouts, as this is a reasonable value that takes into account the typical longest AS-PATH lengths and ensures that routes with excessively long AS-PATHs are not used.

MED is an attribute that influences routing decisions by setting a lower value for a particular route that is being sent to BGP peers. Lower MEDs are preferred over the higher MEDs. Therefore, the exit or entry point with the lower metric gets favored.

Communities with local preference are another way to affect routing decisions by assigning communities to specific routes and instructing BGP peers to give those routes a higher local preference value If the path is preferred. The default MED on Cisco IOS is 0.

In Cisco IOS, BGP communities (both standard and extended) are not sent by default unless the send-community command is explicitly configured. This is because the send-community command tells the router to include BGP communities in the updates that it sends to its BGP peers.

In contrast, in Cisco IOS-XR, BGP communities (both standard and extended) are sent by default on iBGP sessions but not on eBGP sessions. This means that if you want to propagate BGP communities to external BGP peers, you need to enable the send-community command on the eBGP sessions as well.

It's important to be aware of these default behaviors when configuring BGP communities, as they can impact how the communities are propagated across the network.

3.2 Troubleshooting BGP Configuration for Outbound Traffic Manipulation

When analyzing why a certain path is preferred over another for outbound traffic, it is crucial to examine the LOCAL_PREF configuration associated with inbound announcements. This value indicates the preference of a route among other routes advertised by the same AS. A higher LOCAL_PREF value indicates a higher preference for a specific route, making it more likely for outbound traffic to be sent using that path.

3.3 Troubleshooting Network Connectivity

R1 has announced the 50.50.50.0/24 prefix to the AS2, but AS3 is unable to see the network. As mentioned in the introduction, we should follow the checklist to resolve this issue.

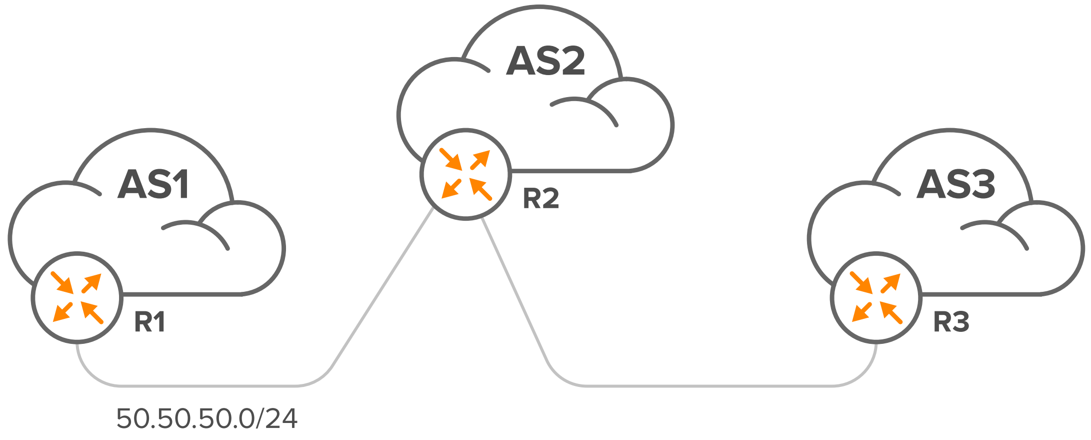


Figure 19 - *Troubleshooting Connectivity*

Firstly, we need to check the eBGP filters on R1 and R2. However, we will require assistance and cooperation from our peer to access R2.

Additionally, we need to verify if AS2 is able to view the 50.50.50.0/24 prefix over the entire network. Misconfigurations in iBGP, lack of full mesh, or Route Reflector problems could cause this issue.

Next, we must investigate the eBGP configuration on R2. There might be a configuration error with as-path filters, prefix lists, or communities that is causing only local prefixes to be seen.

Lastly, we should check the eBGP configuration on R3 to determine if AS3 is viewing all of AS2's originated prefixes. It's possible that AS3 does not know to expect prefixes from AS1 in the peering with AS2 or has similar issues with as-path, prefix, or community filters.

NOTE: Troubleshooting across the Internet can be challenging due to the complexity and distributed nature of the network, but there are tools available to assist with this process. Looking Glasses, which offer traceroute, ping, and BGP status information, are available at many locations around the world and can help identify connectivity issues.

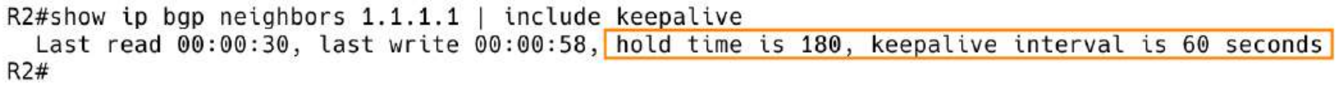
In general, most connectivity problems tend to be found at the network's edge rather than in the transit core. This is because the edge of the network is where customer networks connect to service provider networks, and issues such as misconfigured routers or firewall rules can cause connectivity problems. Problems with the transit core are typically less common and tend to be intermittent and short-term in nature. However, when issues do occur in the transit core, they can have a significant impact on network performance and require immediate attention.

4. Verifying BGP Timers

BGP timers are used to manage the BGP session and ensure that the routers stay synchronized. Check the BGP timers to ensure that they are set correctly. Verify that the keepalive and hold-down timers are set to reasonable values. Misconfigured BGP timers can cause BGP sessions to be dropped, leading to routing instability.

The default value for the hold time suggested in the BGP specification (RFC 4271) is 90 seconds, and keepalives should be sent at intervals of one-third the hold time (30 seconds). However, Cisco uses defaults of 180 and 60 seconds, as shown in Figure 20.

So when two Cisco routers have established a BGP session and exchanged prefixes, 60 seconds later they'll each send a KEEPALIVE message. Upon reception of the keepalive by the other router, that router's hold time for the session will have counted down from 180 to 120, but it now gets reinitialized to 180. This continues every 60 seconds. However, should router R1 lose power, then router R2 won't see any keepalives. So after 180 seconds, router R1 decides that router R2 is dead, sends a NOTIFICATION message and tears down the session.



```
R2#show ip bgp neighbors 1.1.1.1 | include keepalive
  Last read 00:00:30, last write 00:00:58, hold time is 180, keepalive interval is 60 seconds
R2#
```

Figure 20 - *Checking R2 Hold Time and Keepalive Interval Values on R1*

5. BGP Debug Logs

Analyze the BGP debug logs to identify any potential issues with the BGP session or route exchange. Use the debug ip bgp x.x.x.x command to enable BGP debug logging. BGP debug logs can be overwhelming, so focus on specific events or messages that indicate a problem.

Please refer to parts 1 and 2 of the document for more information on examples of BGP debugging scenarios.

6. BGP Troubleshooting and the Intelligent Routing Platform

Troubleshooting BGP can be a complex task, as there are many factors that can impact the network performance. The Noction Intelligent Routing Platform (IRP) can help simplify BGP troubleshooting. It delivers real-time BGP monitoring that allows visually tracking network performance, generating triggers, and sending various alerts and notifications when specific problems occur. The platform is a perfect tool for network planning, delivering Quality of Services, and ensuring the network is performing on its premises.

IRP provides a set of comprehensive reports and graphs reflecting the current state of the network as well as overall statistics on the system's performance. These offer real-time and historical data on optimized destination prefixes, problematic ASs, improved traffic volumes, etc. Based on analytics, organizations can generate a detailed audit trail of ISP performance and the route control process. The reports resulting from real-time BGP monitoring can be used to objectively measure ISP performance and, over time, help in determining the right mix of ISP vendors.

IRP’s troubleshooting tools, such as traceroute, looking glass, whois, and prefix probing, provide instant visibility into the middle mile segment of the Internet and offer quick information on specific remote networks. With on-demand manual probing and visually explicit traceroutes, you can easily detect inter-domain routing anomalies and react accordingly.

Conclusion

Network administrators can quickly and efficiently troubleshoot BGP network issues by following the best practices covered in this eBook, ensuring smooth and uninterrupted network connectivity. However, it's worth noting that when it comes to troubleshooting, practice makes perfect. By gaining hands-on experience and familiarizing themselves with common BGP issues and their resolutions, network administrators can become experts in BGP troubleshooting.

BGP is a complex and evolving protocol, and best practices can change over time. Stay up-to-date with the latest BGP best practices and recommendations to ensure that your network is secure and resilient. Follow industry standards and guidelines, and participate in BGP communities to learn from other network operators.

By adhering to the best practices covered in this tutorial, staying up-to-date with the latest developments in BGP, and actively engaging with the BGP community, network administrators can ensure that their networks are reliable, secure, and resilient.

NOCTION
NETWORK INTELLIGENCE

This ebook was brought to you by Noction

Other names and brands may be claimed as the property of others. Information regarding third party products is provided solely for educational purposes.

Noction Inc. is not responsible for the performance or support of third party products and does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices or products.