Remotely Triggered Black Hole Filtering

ISP Workshops

Last updated 21st July 2022

Acknowledgements

- This material originated from the Cisco ISP/IXP Workshop Programme developed by Philip Smith & Barry Greene

- Bug fixes and improvements are welcomed
  - Please email *workshop (at) bgp4all.com*

Philip Smith

Remotely Triggered Black Hole Filtering

- A simple technique whereby the Network Operator can use their entire backbone to block mischievous traffic to a specific address within their network or their customers' network
- Powerful tool to help with mitigating Distributed Denial of Service Attacks

Remotely Triggered Black Hole Filtering

- Well documented around the Internet, including:
  - Informational RFC from the IETF in 2009:
    - https://tools.ietf.org/html/rfc5635
  - Cisco whitepaper from 2005:
    - https://www.cisco.com/c/dam/en_us/about/security/intelligence/blackhole.pdf
  - Chris Morrow's presentation at NANOG 30 in 2004 describing the technique:
    - https://www.nanog.org/meetings/nanog30/presentations/morrow.pdf

Defending against DDoS

- Link bandwidths from ISPs to their customers are usually quite small
- Link bandwidths from ISPs to their upstreams are usually quite large
- DDoS attacks in this day and age are usually multi-Gbps
  - Significant burden for transit providers to handle
  - Completely swamps the end user link

Defending against DDoS

- Packet filters at the customer side are no good
  - The packets have already traversed the link
  - The link is already swamped
- Packet filters at the ISP side could help
  - Requires human intervention
  - Requires serious CPU power on the ISP access router doing the filtering
  - ISP access router effectively the target now
  - Doesn’t scale!

Defending against DDoS

- Wouldn’t it be better to have all the ISP’s routers dealing with the DDoS ?
- Manual solution:
  - Customer phones ISP and asks them to null route all traffic to the address under attack
  - Which means the ISP has to change router configurations across the backbone; in the middle of the day / outside maintenance
- Automatic solution:
  - Remotely Triggered Black Hole Filtering (RTBH)

RTBH: Two Options

1. Network Operator implements the RTBH function at the customer's request
   - Appropriate for statically connected customers

2. Customer triggers the RTBH activity via their BGP session with their ISP
   - Using a specific RTBH BGP Community (RFC7999)
   - Appropriate for BGP customers of the ISP

RTBH: Option 1

ISP Deploys RTBH Filtering and Trigger Router within their backbone

RTBH – How it works

- Network Operator deploys:
  - RTBH support across their entire backbone
    - Simply a null route for a specific next-hop address
    - (Router Null interfaces simply discard packets sent to them – negligible overhead in modern hardware)
  - A trigger router (usually in the NOC)
    - Talks iBGP with the rest of the backbone (typically as a client to route-reflectors in the core)
    - Used to trigger a blackhole route activity for any address under attack, as requested by a customer

RTBHv4 – Backbone Configuration

- Network Operator sets up a null route for the 192.0.2.1 address on all the backbone routers which participate in BGP

```
ip route 192.0.2.1 255.255.255.255 null0 254
```

- 192.0.2.1 is part of 192.0.2.0/24, the TEST-NET, one of the reserved IPv4 address blocks
  - http://www.iana.org/assignments/iana-ipv4-special-registry
  - It is not used or routed on the public Internet

RTBHv6 – Backbone Configuration

- Network Operator sets up a null route for the 100::1 address on all the backbone routers which participate in BGP

```
ipv6 route 100::1/128 null0 254
```

- 100::1 is part of 100::/64, the Discard Prefix, one of the reserved IPv6 address blocks listed in the IANA registry
  - http://www.iana.org/assignments/iana-ipv6-special-registry
  - It is not used or routed on the public Internet

RTBH – Trigger Router (1)

- Create a route-map to catch routes which need to be blackholed
  - Static routes can be tagged in Cisco IOS – we will tag routes to be blackholed with the value of 666
  - Set origin to be iBGP
  - Set local-preference to be 1000
    - Higher than any other local-preference set in the backbone
  - Set community to be *no-export* and RTBH community (65535:666)
    - Don’t want prefix to leak outside the AS
  - Set next-hop to 192.0.2.1 (IPv4) or 100::1 (IPv6)

RTBHv4 – Trigger Router (2)

- The whole route-map:

```
route-map v4blackhole-trigger permit 10
 description Look for Route 666
 match tag 666
 set local-preference 1000
 set origin igp
 set community no-export 65535:666
 set ip next-hop 192.0.2.1
!
route-map v4blackhole-trigger deny 20
 description Nothing else gets through
```

RTBHv6 – Trigger Router (2)

- The whole route-map:

```
route-map v6blackhole-trigger permit 10
 description Look for Route 666
 match tag 666
 set local-preference 1000
 set origin igp
 set community no-export 65535:666
 set ipv6 next-hop 100::1
!
route-map v6blackhole-trigger deny 20
 description Nothing else gets through
```

RTBHv4 – Trigger Router (3)

- Then introduce the route-map into the BGP configuration
  - NB: the iBGP on the trigger router cannot use "next-hop-self" – Cisco IOS over writes the route-map originated next-hop with "next-hop-self"

```
router bgp 100
 address-family ipv4
  redistribute static route-map v4blackhole-trigger
  neighbor 100.65.0.2 remote-as 100
  neighbor 100.65.0.2 description iBGP with RR1
  neighbor 100.65.0.2 update-source Loopback 0
  neighbor 100.65.0.2 send-community
  neighbor 100.65.0.3 remote-as 100
  neighbor 100.65.0.3 description iBGP with RR2
  neighbor 100.65.0.3 update-source Loopback 0
  neighbor 100.65.0.3 send-community
!
```

RTBHv6 – Trigger Router (3)

- Then introduce the route-map into the BGP configuration
  - NB: the iBGP on the trigger router cannot use "next-hop-self" – Cisco IOS over writes the route-map originated next-hop with "next-hop-self"

```
router bgp 100
 address-family ipv6
  redistribute static route-map v6blackhole-trigger
  neighbor 2001:DB8::2 remote-as 100
  neighbor 2001:DB8::2 description iBGP with RR1
  neighbor 2001:DB8::2 update-source Loopback 0
  neighbor 2001:DB8::2 send-community
  neighbor 2001:DB8::3 remote-as 100
  neighbor 2001:DB8::3 description iBGP with RR2
  neighbor 2001:DB8::3 update-source Loopback 0
  neighbor 2001:DB8::3 send-community
!
```

RTBHv4 – Trigger Router (4)

- To implement the trigger, simply null route whatever address or address block needs to be blackholed
  - With Tag 666

```
ip route 50.62.124.1 255.255.255.255 null0 tag 666
```

  - And this ensures that (for example) 50.62.124.1/32 is announced to the entire backbone with next-hop 192.0.2.1 set

RTBHv6 – Trigger Router (4)

- To implement the trigger, simply null route whatever address or address block needs to be blackholed
  - With Tag 666

```
ipv6 route 2001:DB8:F::E0/128 null0 tag 666
```

  - And this ensures that (for example) 2001:DB8:F::E0/128 is announced to the entire backbone with next-hop 100::1 set

RTBHv4 – End Result

- Prefixes which need to be null routed will come from the trigger router and look like this in the BGP table:

```
 *>i 50.62.124.1/32  192.0.2.1   0    1000    0 i
```

- Routing entry for 50.62.124.1 is this:

```
cr1>sh ip route 50.62.124.1
Routing entry for 50.62.124.1/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 1.2.0.1 7w0d ago
  Routing Descriptor Blocks:
  * 192.0.2.1, from 1.2.0.1, 7w0d ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
      MPLS label: none
```

RTBHv4 – End Result

- Routing entry for 192.0.2.1 is this:

```
cr1>sh ip route 192.0.2.1
Routing entry for 192.0.2.1/32
  Known via "static", distance 1, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Null0
      Route metric is 0, traffic share count is 1
```

- Traffic to 50.62.124.1 is sent to null interface

RTBHv6 – End Result

- Prefixes which need to be null routed will come from the trigger router and look like this in the BGP table:

```
 *>i 2001:DB8:F::E0/128   100::1   0    1000       0 i
```

- Routing entry for 2001:DB8:F::E0 is this:

```
cr1>sh ipv6 route 2001:DB8:F::E0
Routing entry for 2001:DB8:F::E0/128
  Known via "bgp 100", distance 200, metric 0, type internal
  Route count is 1/1, share count 0
  Routing paths:
    100::1
      MPLS label: nolabel
      Last updated 00:00:03 ago
```

RTBHv6 – End Result

- Routing entry for 100::1 is this:

```
cr1>sh ipv6 route 100::1
Routing entry for 100::1/128
  Known via "static", distance 1, metric 0
  Route count is 1/1, share count 0
  Routing paths:
    directly connected via Null0
      Last updated 00:05:21 ago
```

- Traffic to 2001:DB8:F::E0 is sent to null interface

RTBH: Option 2

ISP Deploys RTBH Filtering across their backbone, and supplies BGP community for their customer

RTBH – How it works

- Customer announces the address being attacked by BGP to their upstream provider
  - Prefix is tagged with a special community
- Upstream provider sees the special community from their customer
  - This flags their BGP speaking routers to set the next-hop to the Null interface
  - All traffic to the customer address is discarded

RTBH – Customer Configuration (1)

- Create a route-map to tag routes which need to be blackholed by upstream
  - Routes tagged with 666 need to be blackholed
  - Set origin to be iBGP
  - Set community to the well-known RTBH community (RFC7999)

```
route-map blackhole-trigger permit 10
 description Look for Route 666
 match tag 666
 set origin igp
 set community 65535:666
!
route-map blackhole-trigger deny 20
```

RTBHv4 – Customer Configuration (2)

- Then introduce the route-map into the BGP configuration
  - We will tag static routes with "666" to indicate they are blackhole routes
- And use it on the eBGP with the upstream:

```
router bgp 200
 address-family ipv4
  redistribute static route-map blackhole-trigger
  neighbor 100.65.1.1 remote-as 100
  neighbor 100.65.1.1 description Transit ISP
  neighbor 100.65.1.1 prefix-list upstream-in in
  neighbor 100.65.1.1 prefix-list my-prefixes out
  neighbor 100.65.1.1 send-community
!
```

RTBHv6 – Customer Configuration (2)

- Then introduce the route-map into the BGP configuration
  - We will tag static routes with "666" to indicate they are blackhole routes
- And use it on the eBGP with the upstream:

```
router bgp 200
 address-family ipv6
  redistribute static route-map blackhole-trigger
  neighbor 2001:db8:1::1 remote-as 100
  neighbor 2001:db8:1::1 description Transit ISP
  neighbor 2001:db8:1::1 prefix-list upstreamv6-in in
  neighbor 2001:db8:1::1 prefix-list my-v6prefixes out
  neighbor 2001:db8:1::1 send-community
!
```

RTBHv4 – Customer Configuration (3)

- To implement the trigger, simply null route whatever address or address block needs to be blackholed
  - With Tag 666

```
ip route 50.62.124.1 255.255.255.255 null0 tag 666
```

  - And this ensures that (for example) 50.62.124.1/32 is announced to the upstream provider with community 65535:666 set

RTBHv6 – Customer Configuration (3)

- To implement the trigger, simply null route whatever address or address block needs to be blackholed
  - With Tag 666

```
ipv6 route 2001:DB8:F::E0/128 null0 tag 666
```

  - And this ensures that (for example) 2001:DB8:F::E0/128 is announced to the upstream provider with community 65535:666 set

RTBHv4 – Upstream Configuration (1)

- Upstream provider sets up route-map to look for trigger community from their BGP customers
  - Need to set next hop for non-blackhole routes to be loopback of local router

```
ip community-list standard RTBH permit 65535:666
!
route-map ibgp-policy permit 10
 description Look for Blackhole Routes
 match community RTBH
 set local-preference 1000
 set ip next-hop 192.0.2.1
 set community no-export
!
route-map ibgp-policy permit 20
 description Let everything else through
 set ip next-hop 100.65.0.1
!
```

RTBHv6 – Upstream Configuration (1)

- Upstream provider sets up route-map to look for trigger community from their BGP customers
  - Need to set next hop for non-blackhole routes to be loopback of local router

```
ip community-list standard RTBH permit 65535:666
!
route-map ibgpv6-policy permit 10
 description Look for Blackhole Routes
 match community RTBH
 set local-preference 1000
 set ipv6 next-hop 100::1
 set community no-export
!
route-map ibgpv6-policy permit 20
 description Let everything else through
 set ipv6 next-hop 2001:DB8::1
!
```

RTBHv4 – Upstream Configuration (2)

- The route-map is now applied to the iBGP neighbours of this edge router
  - Note the absence of "next-hop-self" – this is now done in the route-map

```
router bgp 100
 address-family ipv4
  neighbor 100.65.0.2 remote-as 100
  neighbor 100.65.0.2 description iBGP with RR1
  neighbor 100.65.0.2 update-source Loopback 0
  neighbor 100.65.0.2 send-community
  neighbor 100.65.0.2 route-map ibgp-policy out
  neighbor 100.65.0.3 remote-as 100
  neighbor 100.65.0.3 description iBGP with RR2
  neighbor 100.65.0.3 update-source Loopback 0
  neighbor 100.65.0.3 send-community
  neighbor 100.65.0.3 route-map ibgp-policy out
!
```

RTBHv6 – Upstream Configuration (2)

- The route-map is now applied to the iBGP neighbours of this edge router
  - Note the absence of "next-hop-self" – this is now done in the route-map

```
router bgp 100
 address-family ipv6
  neighbor 2001:DB8::2 remote-as 100
  neighbor 2001:DB8::2 description iBGP with RR1
  neighbor 2001:DB8::2 update-source Loopback 0
  neighbor 2001:DB8::2 send-community
  neighbor 2001:DB8::2 route-map ibgpv6-policy out
  neighbor 2001:DB8::3 remote-as 100
  neighbor 2001:DB8::3 description iBGP with RR2
  neighbor 2001:DB8::3 update-source Loopback 0
  neighbor 2001:DB8::3 send-community
  neighbor 2001:DB8::3 route-map ibgpv6-policy out
!
```

RTBHv4 – Upstream Configuration (3)

- Upstream provider then sets up a null route for the 192.0.2.1 address on all the backbone routers which participate in BGP

```
ip route 192.0.2.1 255.255.255.255 null0 254
```

- Note: It is NOT possible in Cisco IOS to change the next-hop of the blackhole route as it arrives on the IPv4 eBGP session
  - Which is why the policy to change the next-hop to 192.0.2.1 is applied on the iBGP sessions

RTBHv6 – Upstream Configuration (3)

- Upstream provider then sets up a null route for the 100::1 address on all the backbone routers which participate in BGP

```
ipv6 route 100::1/128 null0 254
```

- Note: It is NOT possible in Cisco IOS to change the next-hop of the blackhole route as it arrives on the IPv6 eBGP session
  - Which is why the policy to change the next-hop to 100::1 is applied on the iBGP sessions

RTBH – End Result

- Prefixes which need to be null routed coming from the customer will look like this in the BGP table:

```
 *>i 50.62.124.1/32  192.0.2.1   0    1000    0 200 i
```

- Routing entry for 50.62.124.1 is this:

```
cr1>sh ip route 50.62.124.1
Routing entry for 50.62.124.1/32
  Known via "bgp 100", distance 200, metric 0, type internal
  Last update from 1.2.0.4 7w0d ago
  Routing Descriptor Blocks:
  * 192.0.2.1, from 1.2.0.4, 7w0d ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
      MPLS label: none
```

RTBH – End Result

- Routing entry for 192.0.2.1 is this:

```
cr1>sh ip route 192.0.2.1
Routing entry for 192.0.2.1/32
  Known via "static", distance 1, metric 0 (connected)
  Routing Descriptor Blocks:
  * directly connected, via Null0
      Route metric is 0, traffic share count is 1
```

- Traffic to 50.62.124.1 is sent to null interface

RTBH – Conclusion

- Very effective method of dealing with DDoS attacks
  - Enlisting the support of upstream ISP
  - Lightweight on resources
    - Null interface is a discard interface, takes negligible CPU on line card, negligible CPU on control plane
  - Uses a BGP Community for signalling between customer and transit provider
- Recommendation 1: Only take Internet transit from an operator who supports RTBH filtering
- Recommendation 2: Provide the RTBH filtering feature to all your customers

RTBH – Other hints

- Single host addresses are usually what are announced by RTBH Trigger routers
  - /32 for IPv4 and /128 for IPv6
- Websites are usually the most frequent targets
  - Good idea to keep the DNS TTL very low for websites (few minutes)
  - If under DDoS attack:
    - Announce the host address from Trigger router
    - Change the IP address of the website on the host and in the DNS
    - End-users can access web-site again
  - If attacker changes the target address to the new IP address, then repeat the above

RTBH – Other hints

- What can be used as the Trigger Router?
  - Anything that runs modern BGP
  - Trigger router simply announces prefixes into the operator's IBGP mesh
    - Minimal CPU requirements
    - No packeting forwarding requirements
    - Minimal RAM requirements
  - Trigger router could be:
    - FRrouting running on a VM or container
    - Old/retired routing hardware
    - Tiny hardware running BGP daemon

Remotely Triggered Black Hole Filtering

ISP Workshops