Routing Security Tutorial

Philip Smith

Network Startup Resource Center

SANOG 33

Thimphu, Bhutan

11th January 2019

Last updated 11th January 2018

Acknowledgements

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

- Use of these materials is encouraged as long as the source is fully acknowledged and this notice remains in place

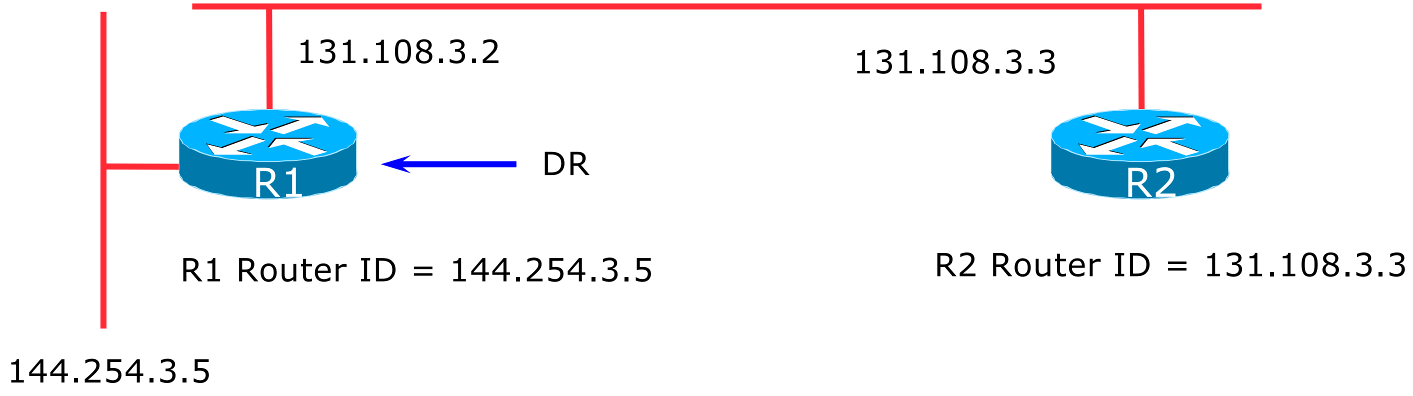
- Bug fixes and improvements are welcomed
  - Please email *workshop (at) bgp4all.com*

Philip Smith

Securing OSPF

Selecting the Designated Router

- Recommended: Configured priority (per interface)
  - Configure high priority on the routers to be the DR/BDR
- Else priority determined by highest router ID
  - Router ID is 32 bit integer
  - Set manually, otherwise derived from the loopback interface IPv4 address, otherwise the highest IPv4 address on the router



Adding interfaces to OSPF

- OSPF interface configuration:
  - When OSPF is configured for a subnet or on an interface, the router will automatically attempt to find neighbours on that subnet or interface

```
router ospf 42
 passive-interface default
```

  - ISP Best Practice is to disable this behaviour:

    And then explicitly enable the interface to allow OSPF to search for neighbours as required:

```
router ospf 42
 no passive-interface Gigabit 4/0
```

OSPF on Cisco IOS

- Enabling OSPF on an interface does two things:
  1. Enables the Hello protocol for forming neighbour relationships and adjacencies with other routers connected to that interface
  2. Announces the interface subnet(s) into OSPF
- Care needed
  - Must avoid enabling the Hello protocol on untrusted networks
    - (e.g. those outside your Autonomous System)

OSPFv2 on Cisco IOS

- Forming neighbour relationships
  - OSPFv2 needs to be activated on the interface the neighbour relationship is desired on:

```
interface Gigabit 4/0
 ip address 192.168.1.1 255.255.255.252
 ip ospf 42 area 0
!
router ospf 42
 passive-interface default
 no passive-interface Gigabit 4/0
!
```

OSPFv3 on Cisco IOS

- Forming neighbour relationships
  - OSPFv3 needs to be activated on the interface the neighbour relationship is desired on:

```
interface Gigabit 4/0
 ipv6 address 2001:DB8:10:FE::4/64
 ipv6 ospf 42 area 0
!
ipv6 router ospf 42
 passive-interface default
 no passive-interface Gigabit 4/0
!
```

OSPF Neighbour Authentication

- Neighbour authentication is highly recommended
  - Prevents unauthorised routers from forming neighbour relationships and potentially compromising the network
- OSPFv2 – Authentication is built-in
  - There are two types:
    - Plain text password
    - MD5 hash
- OSPFv3 – uses standard IP security header
  - There are two types:
    - MD5 hash
    - SHA1

OSPFv2 – Neighbour Authentication

- Configuring authentication for area 0
  - Interfaces still need the authentication key, e.g. POS4/0

```
router ospf 42
 area 0 authentication message-digest
!
interface Gigabit 4/0
 ip ospf message-digest-key <key-no> md5 <passwd>
!
```

- Configuring authentication per interface:

```
interface Gigabit 4/0
 ip ospf authentication message-digest
 ip ospf message-digest-key <key-no> md5 <passwd>
!
```

OSPFv3 – Neighbour Authentication

- Configuring authentication for all interfaces in area 0
  - The key is included in the command turning on authentication for area 0:

```
ipv6 router ospf 42
 area 0 authentication ipsec spi 256 md5 <passwd>
!
```

- Configuring authentication per interface:

```
interface Gigabit 4/0
 ipv6 ospf authentication ipsec spi 256 md5 <passwd>
!
```

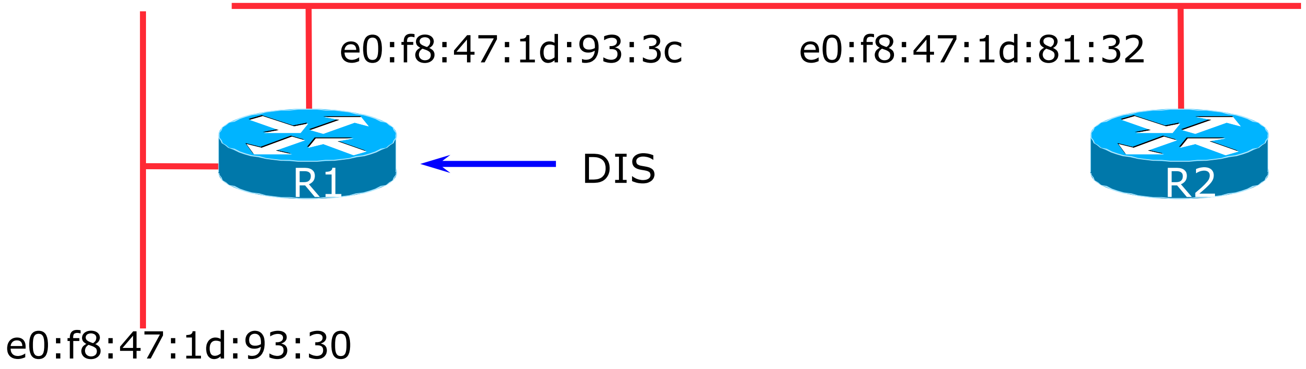
Securing IS-IS

Selecting the Designated Router

- Configured priority (per interface)
  - Configure high priority on the router to be the DIS

```
interface gigabitethernet0/1
 isis priority 127 level-2
```

- Else priority determined by highest MAC address
  - Best practice is to set two routers to be highest priority – then in case of failure of the DIS there is deterministic fall back to the other



Adding interfaces to IS-IS

- To activate IS-IS on an interface:

```
interface Gigabit 4/0
 ip router isis as42
```

  - Puts interface subnet address into the LSDB
  - Enables CLNS on that interface
- To disable IS-IS on an interface:

```
router isis as42
 passive-interface Gigabit 2/0
```

  - Disables CLNS on that interface
  - Puts the interface subnet address into the LSDB
- No IS-IS configuration for an interface
  - No CLNS run on interface, no interface subnet in the LSDB

IS-IS Neighbour Authentication

- Neighbour authentication is highly recommended
  - Prevents unauthorised routers from forming neighbour relationships and potentially compromising the network

- Create a suitable key-chain

```
key chain isis-as42
 key 1
   key-string <password>
!
```

  - There can be up to 255 different keys in each key chain

IS-IS Neighbour Authentication

- Apply key-chain per interface:

```
interface Gigabit 4/0
 isis authentication mode md5 level-2
 isis authentication key-chain isis-as42 level-2
!
```

- Apply key-chain to IS-IS process (all interfaces):

```
router isis as42
 authentication mode md5 level-2
 authentication key-chain isis-as42 level-2
!
```

BGP Best Configuration Practices

Cisco IOS Good Practices

- ISPs should start off with the following BGP commands as a basic template:

```
router bgp 64511
 bgp deterministic-med
 distance bgp 200 200 200
 no synchronization
 no auto-summary
```

Replace with public ASN

Make ebgp and ibgp distance the same

- If supporting more than just IPv4 unicast neighbours

```
 no bgp default ipv4-unicast
```

  - Turns off IOS assumption that all neighbours will exchange IPv4 prefixes

EBGP Configuration Best Practices

- Industry standard is described in RFC8212
  - https://tools.ietf.org/html/rfc8212
  - External BGP (EBGP) Route Propagation Behaviour without Policies

- NB: BGP in Cisco IOS is permissive by default
  - This is contrary to industry standard and RFC8212

- Configuring BGP peering without using filters means:
  - All best paths on the local router are passed to the neighbour
  - All routes announced by the neighbour are received by the local router
  - Can have disastrous consequences (see RFC8212)

EBGP Configuration Best Practices

- Best practice is to ensure that each eBGP neighbour has inbound and outbound filter applied:

```
router bgp 64511
 address-family ipv4
  neighbor 100.64.0.1 remote-as 64510
  neighbor 100.64.0.1 prefix-list as64510-in in
  neighbor 100.64.0.1 prefix-list as64510-out out
  neighbor 100.64.0.1 activate
```

Receiving Prefixes

Receiving Prefixes

- There are three scenarios for receiving prefixes from other ASNs
  - Customer talking BGP
  - Peer talking BGP
  - Upstream/Transit talking BGP
- Each has different filtering requirements and need to be considered separately

Receiving Prefixes: From Customers

- ISPs should only accept prefixes which have been assigned or allocated to their downstream customer
- If ISP has assigned address space to its customer, then the customer IS entitled to announce it back to his ISP
- If the ISP has NOT assigned address space to its customer, then:
  - Check in the five RIR databases to see if this address space really has been assigned to the customer
  - The tool: whois –h jwhois.apnic.net x.x.x.0/24
    - (jwhois is "joint whois" and queries all RIR databases)

Receiving Prefixes: From Customers

- Example use of whois to check if customer is entitled to announce address space:

```
$ whois -h jwhois.apnic.net 202.12.29.0

inetnum:          202.12.29.0 - 202.12.29.255
netname:          APNIC-SERVICES-AU
descr:            Asia Pacific Network Information Centre
descr:            Regional Internet Registry for the Asia-Pacific Region
descr:            6 Cordelia Street
descr:            South Brisbane
geoloc:           27.4731138 153.0141194
country:          AU
admin-c:          AIC1-AP
tech-c:           AIC1-AP
mnt-by:           APNIC-HM
mnt-irt:          IRT-APNIC-IS-AP
status:           ASSIGNED PORTABLE
changed:          hm-changed@apnic.net 20170327
changed:          hm-changed@apnic.net 20170331
source:           APNIC
```

inetnum – means it is an address delegation to an entity

Portable – means its an assignment to the customer, the customer can announce it to you

Receiving Prefixes: From Customers

- Example use of whois to check if customer is entitled to announce address space:

```
$ whois -h jwhois.apnic.net 193.128.0.0/16

inetnum:        193.128.0.0 - 193.133.255.255
netname:        UK-PIPEX-193-128-133
country:        GB
org:            ORG-UA24-RIPE
admin-c:        WERT1-RIPE
tech-c:         UPHM1-RIPE
status:         ALLOCATED UNSPECIFIED
remarks:        Please send abuse notification to abuse@uk.uu.net
mnt-by:         RIPE-NCC-HM-MNT
mnt-by:         AS1849-MNT
mnt-routes:     AS1849-MNT
mnt-routes:     WCOM-EMEA-RICE-MNT
mnt-irt:        IRT-MCI-GB
created:        2002-06-25T15:05:40Z
last-modified:  2016-10-31T12:20:01Z
source:         RIPE
```

ALLOCATED – means that this is Provider Aggregatable address space and can only be announced by the ISP holding the allocation (in this case Verizon UK)

Receiving Prefixes from customer: Cisco IOS

- For Example:
  - Downstream has 100.69.0.0/20 block
  - Should only announce this to upstreams
  - Upstreams should only accept this from them
- Configuration on upstream

```
router bgp 100
 address-family ipv4
  neighbor 100.67.10.1 remote-as 101
  neighbor 100.67.10.1 prefix-list customer in
  neighbor 100.67.10.1 prefix-list default out
  neighbor 100.67.10.1 activate
!
ip prefix-list customer permit 100.69.0.0/20
!
ip prefix-list default permit 0.0.0.0/0
```

Receiving Prefixes: From Peers

- A peer is an ISP with whom you agree to exchange prefixes you originate into the Internet routing table
  - Prefixes you accept from a peer are only those they have indicated they will announce
  - Prefixes you announce to your peer are only those you have indicated you will announce

Receiving Prefixes: From Peers

- Agreeing what each will announce to the other:
  - Exchange of e-mail documentation as part of the peering agreement, and then ongoing updates

    OR

  - Use of the Internet Routing Registry and configuration tools such as the IRRToolSet

https://github.com/irrtoolset/irrtoolset

Receiving Prefixes from peer: Cisco IOS

- For Example:
  - Peer has 220.50.0.0/16, 61.237.64.0/18 and 81.250.128.0/17 address blocks
- Configuration on local router

```
router bgp 100
 address-family ipv4
  neighbor 100.67.10.1 remote-as 101
  neighbor 100.67.10.1 prefix-list my-peer in
  neighbor 100.67.10.1 prefix-list my-prefix out
  neighbor 100.67.10.1 activate
!
ip prefix-list my-peer permit 220.50.0.0/16
ip prefix-list my-peer permit 61.237.64.0/18
ip prefix-list my-peer permit 81.250.128.0/17
ip prefix-list my-peer deny 0.0.0.0/0 le 32
!
ip prefix-list my-prefix permit 100.67.16.0/20
```

Receiving Prefixes: From Upstream/Transit Provider

- Upstream/Transit Provider is an ISP who you pay to give you transit to the WHOLE Internet
- Receiving prefixes from them is not desirable unless really necessary
  - Traffic Engineering – see BGP Multihoming presentations
- Ask upstream/transit provider to either:
  - originate a default-route

    OR
  - announce one prefix you can use as default

Receiving Prefixes: From Upstream/Transit Provider

- Downstream Router Configuration

```
router bgp 100
 address-family ipv4
  network 100.66.0.0 mask 255.255.224.0
  neighbor 100.65.7.1 remote-as 101
  neighbor 100.65.7.1 prefix-list infilter in
  neighbor 100.65.7.1 prefix-list outfilter out
  neighbor 100.65.7.1 activate
!
ip prefix-list infilter permit 0.0.0.0/0
!
ip prefix-list outfilter permit 100.66.0.0/19
```

Receiving Prefixes: From Upstream/Transit Provider

- Upstream Router Configuration

```
router bgp 101
 address-family ipv4
  neighbor 100.65.7.2 remote-as 100
  neighbor 100.65.7.2 default-originate
  neighbor 100.65.7.2 prefix-list cust-in in
  neighbor 100.65.7.2 prefix-list cust-out out
  neighbor 100.65.7.2 activate
!
ip prefix-list cust-in permit 100.66.0.0/19
!
ip prefix-list cust-out permit 0.0.0.0/0
```

Receiving Prefixes: From Upstream/Transit Provider

- If necessary to receive prefixes from any provider, care is required.
  - Don’t accept default (unless you need it)
  - Don’t accept your own prefixes
- Special use prefixes for IPv4 and IPv6:
  - http://www.rfc-editor.org/rfc/rfc6890.txt
- For IPv4:
  - Don’t accept prefixes longer than /24 (?)
    - /24 was the historical class C
- For IPv6:
  - Don’t accept prefixes longer than /48 (?)
    - /48 is the design minimum delegated to a site

Receiving Prefixes: From Upstream/Transit Provider

- Check Team Cymru's list of "bogons"
  - http://www.team-cymru.com/bogon-reference.html
- For IPv4 also consult:
  - https://www.rfc-editor.org/rfc/rfc6441.txt (BCP171)
- For IPv6 also consult:
  - http://www.space.net/~gert/RIPE/ipv6-filters.html
- Bogon Route Server:
  - https://www.team-cymru.com/bogon-reference-bgp.html
  - Supplies a BGP feed (IPv4 and/or IPv6) of address blocks which should not appear in the BGP table

Receiving IPv4 Prefixes

```
router bgp 100
 network 101.10.0.0 mask 255.255.224.0
 neighbor 100.65.7.1 remote-as 101
 neighbor 100.65.7.1 prefix-list in-filter in
!
ip prefix-list in-filter deny 0.0.0.0/0                ! Default
ip prefix-list in-filter deny 0.0.0.0/8 le 32          ! RFC1122 local host
ip prefix-list in-filter deny 10.0.0.0/8 le 32         ! RFC1918
ip prefix-list in-filter deny 100.64.0.0/10 le 32      ! RFC6598 shared address
ip prefix-list in-filter deny 101.10.0.0/19 le 32      ! Local prefix
ip prefix-list in-filter deny 127.0.0.0/8 le 32        ! Loopback
ip prefix-list in-filter deny 169.254.0.0/16 le 32     ! Auto-config
ip prefix-list in-filter deny 172.16.0.0/12 le 32      ! RFC1918
ip prefix-list in-filter deny 192.0.0.0/24 le 32       ! RFC6598 IETF protocol
ip prefix-list in-filter deny 192.0.2.0/24 le 32       ! TEST1
ip prefix-list in-filter deny 192.168.0.0/16 le 32     ! RFC1918
ip prefix-list in-filter deny 198.18.0.0/15 le 32      ! Benchmarking
ip prefix-list in-filter deny 198.51.100.0/24 le 32    ! TEST2
ip prefix-list in-filter deny 203.0.113.0/24 le 32     ! TEST3
ip prefix-list in-filter deny 224.0.0.0/3 le 32        ! Multicast & Experimental
ip prefix-list in-filter deny 0.0.0.0/0 ge 25          ! Prefixes >/24
ip prefix-list in-filter permit 0.0.0.0/0 le 32
```

Receiving IPv6 Prefixes

```
router bgp 100
 network 2020:3030::/32
 neighbor 2020:3030::1 remote-as 101
 neighbor 2020:3030::1 prefix-list v6in-filter in
!
ipv6 prefix-list v6in-filter permit 64:ff9b::/96          ! RFC6052 v4v6trans
ipv6 prefix-list v6in-filter deny 2001::/23 le 128        ! RFC2928 IETF prot
ipv6 prefix-list v6in-filter deny 2001:2::/48 le 128      ! Benchmarking
ipv6 prefix-list v6in-filter deny 2001:10::/28 le 128     ! ORCHID
ipv6 prefix-list v6in-filter deny 2001:db8::/32 le 128    ! Documentation
ipv6 prefix-list v6in-filter deny 2002::/16 le 128        ! Deny all 6to4
ipv6 prefix-list v6in-filter deny 2020:3030::/32 le 128   ! Local Prefix
ipv6 prefix-list v6in-filter deny 3ffe::/16 le 128        ! Formerly 6bone
ipv6 prefix-list v6in-filter permit 2000::/3 le 48        ! Global Unicast
ipv6 prefix-list v6in-filter deny ::/0 le 128
```

Note: These filters block Teredo (serious security risk) and 6to4 (deprecated by RFC7526)

Receiving Prefixes

- Paying attention to prefixes received from customers, peers and transit providers assists with:
  - The integrity of the local network
  - The integrity of the Internet
- Responsibility of all ISPs to be good Internet citizens

Unicast Reverse Path Forwarding

Anti-spoofing filtering

Background

- uRPF is known as the anti-spoofing filter
- Most DDoS attacks today are caused by hijacked systems sending floods of packets with a fake source address
  - This fake source address is the IP address of the DDoS target
- There are hundreds of thousands of compromised systems around the world:
  - Websites hijacked through PHP or other vulnerabilities
  - Home routers/modems/simple IP devices hijacked through backdoors or misconfiguration
  - Computing devices with malware installed unknowingly by the user (via clickable links, email attachments etc)

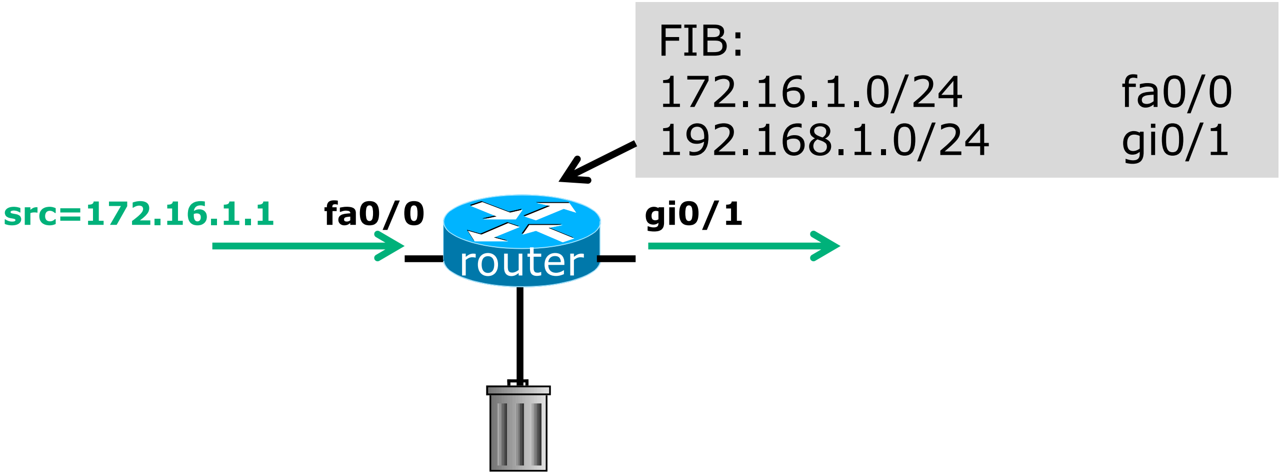
Background

- The hijacked systems are part of a command and control network
  - The malware waits for commands from their controllers
- When a DDoS is instigated, the DDoS “controller” informs the hijacked systems to launch the attack
  - Spoofed source address
  - Amplifier target (eg one packet in produces hundreds of packets response)
- And the attack begins
- How can we stop this??

Solution

- Implement uRPF on all access networks:
  - End-user access network: uRPF on every routed interface/VLAN
  - Service Provider: uRPF on every single-homed customer access interface
- uRPF was first implemented by most major equipment vendors back in 1996
  - More efficient than packet filtering
  - Implemented in hardware in the line cards of routers
- How does it work?

What is uRPF?



FIB:

172.16.1.0/24	fa0/0
192.168.1.0/24	gi0/1

src=172.16.1.1

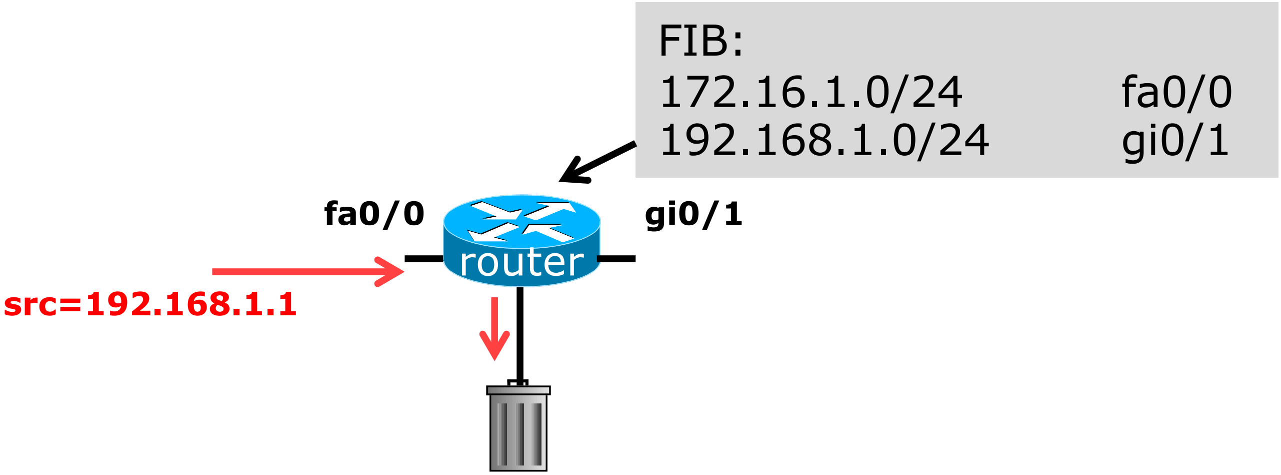
fa0/0

router

gi0/1

- Router compares source address of incoming packet with FIB entry
  - If FIB entry interface matches incoming interface, the packet is forwarded
  - If FIB entry interface does not match incoming interface, the packet is dropped

What is uRPF?



- Router compares source address of incoming packet with FIB entry
  - If FIB entry interface matches incoming interface, the packet is forwarded
  - If FIB entry interface does not match incoming interface, the packet is dropped

Recommendation

- Every operator of an access network must implement uRPF on their access LANs
- This goes towards helping deal with the DDoS threat facing all Internet content hosting

Remotely Triggered Blackhole Filtering

Dealing with DDoS attacks

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:
    - http://www.cisco.com/c/dam/en/us/products/collateral/security/ios-network-foundation-protection-nfp/prod_white_paper0900aecd80313fac.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 null 0  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 null 0  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 66
  - Set origin to be iBGP
  - Set local-preference to be 200
    - 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 66
 match tag 66
 set local-preference 200
 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 66
 match tag 66
 set local-preference 200
 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 1.2.0.2 remote-as 100
  neighbor 1.2.0.2 description iBGP with RR1
  neighbor 1.2.0.2 update-source Loopback 0
  neighbor 1.2.0.2 send-community
  neighbor 1.2.0.3 remote-as 100
  neighbor 1.2.0.3 description iBGP with RR2
  neighbor 1.2.0.3 update-source Loopback 0
  neighbor 1.2.0.3 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:dbd::2 remote-as 100
  neighbor 2001:dbd::2 description iBGP with RR1
  neighbor 2001:dbd::2 update-source Loopback 0
  neighbor 2001:dbd::2 send-community
  neighbor 2001:dbd::3 remote-as 100
  neighbor 2001:dbd::3 description iBGP with RR2
  neighbor 2001:dbd::3 update-source Loopback 0
  neighbor 2001:dbd::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 66

```
ip route 50.62.124.1 255.255.255.255 null0 tag 66
```

  - 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 66

```
ipv6 route 2001:db8:f::e0/128 null0 tag 66
```

  - 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     150     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 1.2.0.1 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    200       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 66 will 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 66
 match tag 66
 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 “66” 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 1.1.1.1 remote-as 100
  neighbor 1.1.1.1 description Transit ISP
  neighbor 1.1.1.1 prefix-list upstream-in in
  neighbor 1.1.1.1 prefix-list my-prefixes out
  neighbor 1.1.1.1 send-community
!
```

RTBHv6 – Customer Configuration (2)

- Then introduce the route-map into the BGP configuration
  - We will tag static routes with “66” 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 66

```
ip route 50.62.124.1 255.255.255.255 null0 tag 66
```

  - 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 66

    ```
    ipv6 route 2001:db8:f::e0/128 null0 tag 66
    ```

  - 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 expanded RTBH permit 100:666
!
route-map ibgp-policy permit 10
 description Look for Blackhole Routes
 match community RTBH
 set local-preference 150
 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 1.2.0.1
!
```

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 expanded RTBH permit 100:666
!
route-map ibgpv6-policy permit 10
 description Look for Blackhole Routes
 match community RTBH
 set local-preference 150
 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 1.2.0.2 remote-as 100
  neighbor 1.2.0.2 description iBGP with RR1
  neighbor 1.2.0.2 update-source Loopback 0
  neighbor 1.2.0.2 send-community
  neighbor 1.2.0.2 route-map ibgp-policy out
  neighbor 1.2.0.3 remote-as 100
  neighbor 1.2.0.3 description iBGP with RR2
  neighbor 1.2.0.3 update-source Loopback 0
  neighbor 1.2.0.3 send-community
  neighbor 1.2.0.3 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 null 0  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 null 0  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    150    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

Route Origin Authorisation

Steps to securing the Routing System

What is RPKI?

- Resource Public Key Infrastructure (RPKI)
  - A security framework for verifying the association between resource holder and their Internet resources
  - Created to address the issues discussed in RFC 4593 “Generic Threats to Routing Protocols” (Oct 2006)
- Helps to secure Internet routing by validating routes
  - Proof that prefix announcements are coming from the legitimate holder of the resource
  - RFC 6480 – An Infrastructure to Support Secure Internet Routing (Feb 2012)

Benefits of RPKI - Routing

- Prevents route hijacking
  - A prefix originated by an AS without authorisation
  - Reason: malicious intent
- Prevents mis-origination
  - A prefix that is mistakenly originated by an AS which does not own it
  - Also route leakage
  - Reason: configuration mistake / fat finger

BGP Security (BGPsec)

- Extension to BGP that provides improved security for BGP routing
- Being worked on by the SIDR Working Group at IETF
- Implemented via a new optional non-transitive BGP attribute that contains a digital signature
- Two components:
  - BGP Prefix Origin Validation (using RPKI)
  - BGP Path Validation

Route Origin Authorisation (ROA)

- A digital object that contains a list of address prefixes and one AS number
- It is an authority created by a prefix holder to authorise an AS Number to originate one or more specific route advertisements
- Publish a ROA using your RIR member portal

Router Origin Validation

- Router must support RPKI
- Checks an RP cache / validator
- Validation returns 3 states:
  - Valid = when authorization is found for prefix X
  - Invalid = when authorization is found for prefix X but not from ASN Y
  - Unknown = when no authorization data is found

Router Origin Validation

- Vendor support:
  - Cisco IOS – available in release 15.2
  - Cisco IOS/XR – available in release 4.3.2
  - Juniper – available in release 12.2
  - Nokia – available in release R12.0R4
  - Huawei – available in release V800R009C10
  - Brocade – available in release TBA

RPKI Validator Caches

- MANRS:
  - https://www.manrs.org/isps/guide/filtering/
- NLnet Labs Routinator
  - https://www.nlnetlabs.nl/projects/rpki/routinator/
  - https://github.com/NLnetLabs/routinator
- Dragon Research validator
  - https://rpki.net
  - https://github.com/dragonresearch/rpki.net/
- RIPE NCC validator
  - https://github.com/RIPE-NCC/rpki-validator-3/wiki

Configure Router to Use Cache

- Point router to the local RPKI cache
  - Server listens on port 43779
  - Cisco IOS example:

```
router bgp 64512
 bgp rpki server tcp 10.0.0.3 port 43779 refresh 60
```

BGP Table (IPv4)

```
RPKI validation codes: V valid, I invalid, N Not found

Network          Metric LocPrf Path
N*>  1.0.4.0/24       0        37100 6939 4637 1221 38803 56203 i
N*>  1.0.5.0/24       0        37100 6939 4637 1221 38803 56203 i
...
V*>  1.9.0.0/16       0        37100 4788 i
N*>  1.10.8.0/24      0        37100 10026 18046 17408 58730 i
N*>  1.10.64.0/24     0        37100 6453 3491 133741 i
...
V*>  1.37.0.0/16      0        37100 4766 4775 i
N*>  1.38.0.0/23      0        37100 6453 1273 55410 38266 i
N*>  1.38.0.0/17      0        37100 6453 1273 55410 38266 {38266} i
...
I*   5.8.240.0/23     0        37100 44217 3178 i
I*   5.8.241.0/24     0        37100 44217 3178 i
I*   5.8.242.0/23     0        37100 44217 3178 i
I*   5.8.244.0/23     0        37100 44217 3178 i
...
```

Courtesy of SEACOM: http://as37100.net

BGP Table (IPv6)

```
RPKI validation codes: V valid, I invalid, N Not found

Network              Metric LocPrf Path
N*>  2001::/32            0        37100 6939 i
N*   2001:4:112::/48      0        37100 112 i
...
V*>  2001:240::/32        0         37100 2497 i
N*>  2001:250::/48        0         37100 6939 23911 45
N*>  2001:250::/32        0         37100 6939 23911 23910 i
...
V*>  2001:348::/32        0         37100 2497 7679 i
N*>  2001:350::/32        0         37100 2497 7671 i
N*>  2001:358::/32        0         37100 2497 4680 i
...
I*   2001:1218:101::/48   0         37100 6453 8151 278 i
I*   2001:1218:104::/48   0         37100 6453 8151 278 i
N*   2001:1221::/48       0         37100 2914 8151 28496 i
N*>  2001:1228::/32       0         37100 174 18592 i
...
```

Courtesy of SEACOM: http://as37100.net

RPKI BGP State: Valid

```
BGP routing table entry for 2001:240::/32, version 109576927
Paths: (2 available, best #2, table default)
  Not advertised to any peer
  Refresh Epoch 1
  37100 2497
    2C0F:FEB0:11:2::1 (FE80::2A8A:1C00:1560:5BC0) from
                                 2C0F:FEB0:11:2::1 (105.16.0.131)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 37100:2 37100:22000 37100:22004 37100:22060
      path 0828B828 RPKI State valid
      rx pathid: 0, tx pathid: 0x0
```

Courtesy of SEACOM: http://as37100.net

RPKI BGP State: Invalid

```
BGP routing table entry for 2001:1218:101::/48, version 149538323
Paths: (2 available, no best path)
  Not advertised to any peer
  Refresh Epoch 1
  37100 6453 8151 278
    2C0F:FEB0:B:3::1 (FE80::86B5:9C00:15F5:7C00) from
                                  2C0F:FEB0:B:3::1 (105.16.0.162)
      Origin IGP, metric 0, localpref 100, valid, external
      Community: 37100:1 37100:12
      path 0DA7D4FC RPKI State invalid
      rx pathid: 0, tx pathid: 0
```

Courtesy of SEACOM: http://as37100.net

RPKI BGP State: Not Found

```
BGP routing table entry for 2001:200::/32, version 124240929
Paths: (2 available, best #2, table default)
  Not advertised to any peer
  Refresh Epoch 1
  37100 2914 2500
    2C0F:FEB0:11:2::1 (FE80::2A8A:1C00:1560:5BC0) from
                                2C0F:FEB0:11:2::1 (105.16.0.131)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 37100:1 37100:13
      path 19D90E68 RPKI State not found
      rx pathid: 0, tx pathid: 0x0
```

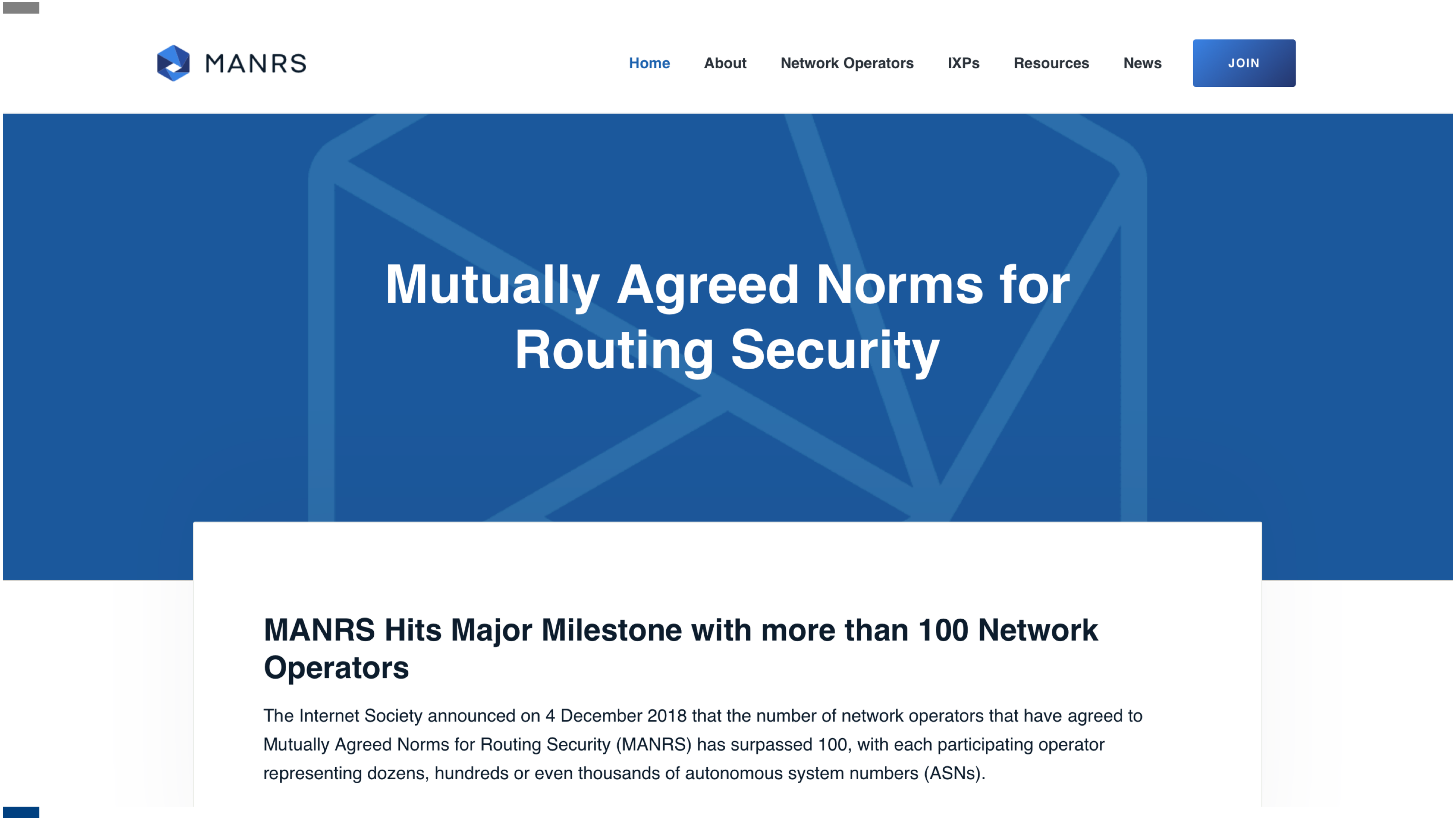
Courtesy of SEACOM: http://as37100.net

Using RPKI

- Network operators can make decisions based on RPKI state:
  - Invalid – discard the prefix
  - Not found – let it through (maybe low local preference)
  - Valid – let it through (high local preference)
- Some operators even considering making “not found” a discard event
  - But then Internet IPv4 BGP table would shrink to about 55000 prefixes and the IPv6 BGP table would shrink to about 9600 prefixes!

RPKI Summary

- All AS operators must consider deploying
  - Which means: **Signing ROAs & dropping invalids**
- An important step to securing the routing system
  - Origin validation
- Doesn’t secure the path, but that’s the next hurdle to cross
- With origin validation, the opportunities for malicious or accidental mis-origination disappear
- FAQ:
  - https://nlnetlabs.nl/projects/rpki/faq/



Mutually Agreed Norms for Routing Security

MANRS Hits Major Milestone with more than 100 Network Operators

The Internet Society announced on 4 December 2018 that the number of network operators that have agreed to Mutually Agreed Norms for Routing Security (MANRS) has surpassed 100, with each participating operator representing dozens, hundreds or even thousands of autonomous system numbers (ASNs).

How can MANRS help?

MANRS outlines four simple but concrete actions that network operators should take:

- **Filtering** – Ensure the correctness of your own announcements and of announcements from your customers to adjacent networks with prefix and AS-path granularity
- **Anti-spoofing** – Enable source address validation for at least single-homed stub customer networks, your own end-users, and infrastructure
- **Coordination** – Maintain globally accessible up-to-date contact information
- **Global Validation** – Publish your data, so others can validate routing information on a global scale

A separate set of Actions applies explicitly to Internet Exchange Points.

Configuration Tips

Other tricks to help improving routing security

Limiting AS Path Length

- Some BGP implementations have problems with long AS_PATHS
  - Memory corruption
  - Memory fragmentation
- Even using AS_PATH prepends, it is not normal to see more than 20 ASes in a typical AS_PATH in the Internet today
  - The Internet is around 5 ASes deep on average
  - Largest AS_PATH is usually 16-20 ASNs

```
neighbor x.x.x.x maxas-limit 20
```

Limiting AS Path Length

- Some announcements have ridiculous lengths of AS-paths
  - This example is an error in one IPv6 implementation

```
*> 3FFE:1600::/24       22 11537 145 12199 10318 10566 13193 1930 2200 3425 293 5609 5430
13285 6939 14277 1849 33 15589 25336 6830 8002 2042 7610 i
```

  - This example shows 100 prepends (for no obvious reason)

```
*>i193.105.15.0         2516 3257 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404 50404
50404 i
```

- If your implementation supports it, consider limiting the maximum AS-path length you will accept

BGP Maximum Prefix Tracking

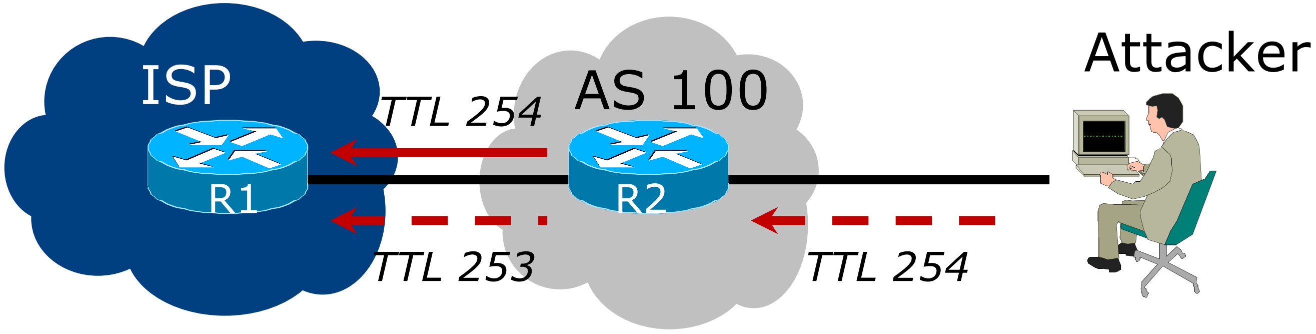
- Allow configuration of the maximum number of prefixes a BGP router will receive from a peer
- Two level control:
  - Warning threshold: log warning message
  - Maximum: tear down the BGP peering, manual intervention required to restart

    ```
    neighbor <x.x.x.x> maximum-prefix <max> [restart N] [<threshold>] [warning-only]
    ```

- *restart* is an optional keyword which will restart the BGP session N minutes after being torn down
- *threshold* is an optional parameter between 1 to 100
  - Specify the percentage of <max> that will cause a warning message to be generated. Default is 75%.
- *warning-only* is an optional keyword which allows log messages to be generated but peering session will not be torn down

BGP TTL "hack"

- Implement RFC5082 on BGP peerings
  - (Generalised TTL Security Mechanism)
  - Neighbour sets TTL to 255
  - Local router expects TTL of incoming BGP packets to be 254
  - No one apart from directly attached devices can send BGP packets which arrive with TTL of 254, so any possible attack by a remote miscreant is dropped due to TTL mismatch



BGP TTL “hack”

- TTL Hack:
  - Both neighbours must agree to use the feature
  - TTL check is much easier to perform than MD5
  - (Called BTSH – BGP TTL Security Hack)
- Provides “security” for BGP sessions
  - In addition to packet filters of course
  - MD5 should still be used for messages which slip through the TTL hack
  - See https://www.nanog.org/meetings/nanog27/presentations/meyer.pdf for more details

BGP TTL ‘hack’

- Configuration example:

```
neighbor 100.121.0.2 ttl-security hops 1
```

- BGP neighbour status:

```
Router# sh ip bgp neigh 100.121.0.2
...
Mininum incoming TTL 254, Outgoing TTL 255
Local host: 100.121.0.1, Local port: 41103
Foreign host: 100.121.0.2, Foreign port: 179
```

- The neighbour must set the same configuration
  - If they don’t, the BGP session will not come up

Mutually Agreed Norms for Routing Security

Industry Best Practices to ensure Security of the Routing System

MANRS

Routing Security

- Implement the recommendations in https://www.manrs.org/manrs
  1. Prevent propagation of incorrect routing information
     - Filter BGP peers, in & out!
  2. Prevent traffic with spoofed source addresses
     - BCP38 – Unicast Reverse Path Forwarding
  3. Facilitate communication between network operators
     - NOC to NOC Communication
  4. Facilitate validation of routing information
     - Route Origin Authorisation using RPKI

MANRS

MANRS 1)

- Filtering prefixes inbound and outbound
  - RFC8212 requires all EBGP implementations to reject prefixes received and announced in the absence of any policy

- Advice: ***Never*** set up an EBGP session without inbound and outbound prefix filters
  - If full table required, block at least the bogons (see earlier)

MANRS 2)

- Implementing BCP 38
  - Unicast Reverse Path Forwarding
  - (Deny outbound traffic from customers which has spoofed source addresses)

- Advice: implement uRPF on ***all*** single-homed customer facing interfaces
  - Cheaper (CPU & RAM) than implementing packet filters

MANRS 3)

- Facilitate NOC to NOC communication
  - Know the **direct** NOC contacts for your customer Network Operators, your peer Network Operators, and your upstream Network Operators
  - This is not calling their “customer support line”
  - Make sure NOC contact info is part of any service contract
- Advice: NOC contact info for all connected Autonomous Networks is known to your NOC

MANRS 4)

- Facilitate validation of Routing Information
  - RPKI and Route Origin Authorisation (ROA)
  - All routes originated need to be signed to indicate that your AS is authorised to originate these routes
    - Helps secure the global routing system

- Advice: Sign ROAs for all originated routes using RPKI
  - And make sure all customer originated routes are also signed
  - Validate received routes from all peers
    - High priority to validated routes
    - Discard invalid routes
    - Low priority for unsigned routes

MANRS summary

- If your organisation supports and implements all 4 techniques in your network
  - Then join MANRS
  - https://www.manrs.org/join/
  - MANRS for Operators
  - MANRS for IXPs

Summary

- Use configuration templates
- Standardise the configuration
- Be aware of standard "tricks" to avoid compromise of the BGP session
- Anything to make your life easier, network less prone to errors, network more likely to scale
- Implement the four fundamentals of MANRS
- It's all about scaling – if your network won't scale, then it won't be successful

Routing Security Tutorial