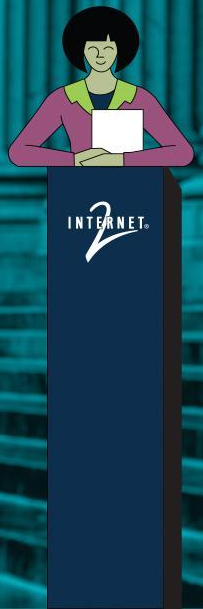
INTERNET2
2023
COMMUNITY
exchange
May 8-11, 2023 Atlanta, GA



Workshop: How to Ensure No Detours

eBGP Route Policy for R&E Networks

May 8, 2023 - Version 0.1 (first public release)

Jeff Bartig
Senior Interconnection Architect, Internet2

Introduction

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

eBGP Routing Policy

Table of Contents/Agenda

Definitions

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

Definitions

- **eBGP:** external BGP. Used for exchanging routes between two different autonomous systems (AS).
-
- **iBGP**: internal BGP. Used for exchanging routes between routers that are part of a single AS.

Definitions

- **Settlement-free peering**: when two networks eBGP peer with each other and exchange routes to their respective customers' networks. No money is exchanged (settlement-free) and each network pays their own costs for the interconnection.

- **Transit provider**: a network that is able to provide a customer network access to the full Internet, generally for a fee. In R&E networking, this is sometimes referred to as a "commodity" Internet provider. Since no one transit network directly connects to every other network that is a member of the Internet, a transit provider must settlement-free peer with other networks and/or purchase transit from other providers to be able to reach networks that are not its direct customers.

- **Customer-Provider BGP relationship:** The customer network sends its routes via BGP to the provider. The provider sends the customer its full routing table, consisting of routes it learns from its customers, settlement-free peers, and any transit providers it may have.

Definitions

- **Tier 1 network:** a transit provider network that has access to the full Internet by settlement-free peering with all other tier 1 and select other networks and does not receive transit from any provider.
-

A Few Important BGP Attributes

INTERNET2
2023
COMMUNITY exchange
May 8-11, 2023 Atlanta, GA

BGP Path Attributes

RFC4271: A Border Gateway Protocol 4 (BGP-4) defines several Path Attributes of BGP update messages. The ones that may be of interest for our topic today include:

- NEXT_HOP
- AS_PATH
- MULTI_EXIT_DISC
- LOCAL_PREF

RFC4271 (January 2006) obsoleted RFC1771 (March 1995), which obsoleted the original BGP-4 RFC1654 (July 1994). Some of the strengths of BGP that led to its long-term success include its simplicity and its ability to be enhanced. An example is the ability to add additional path attributes.

RFC1997: BGP Communities Attribute added a new path attribute:

- COMMUNITIES

BGP Path Attributes – AS_PATH

The AS_PATH is a well-known mandatory attribute. It identifies the autonomous systems which the routing information carried in a BGP UPDATE message has passed.

A BGP speaker doesn't modify the AS_PATH when advertising a route to its iBGP neighbors. When a BGP speaker advertises a route to its eBGP neighbors, it prepends its own AS number to the AS_PATH.

The primary purpose of the AS_PATH is to prevent routing loops. If a BGP speaker finds its own AS number in the AS_PATH of a route learned from an eBGP neighbor, the default behavior is to drop the route to prevent an AS loop.

BGP Path Attributes – AS_PATH

A secondary use of the AS_PATH is to determine route preference. A shorter AS_PATH is preferred. Unfortunately, as we will soon discuss, a shorter AS_PATH does not necessarily indicate a more desirable route.

Recognizing this secondary use, most BGP speakers allow the prepending of additional AS numbers to the AS_PATH. This is frequently the BGP speaker's local AS number, but some router operating systems allow the prepending of other AS numbers.

BGP Path Attributes – MULTI_EXIT_DISC

The multi-exit discriminator (MED) is an optional non-transitive attribute that is intended to be used on external (inter-AS) links to discriminate among multipleexit or entry points to a neighboring AS.

It is a 16-bit value representing a metric or cost. A lower cost is preferred, so a route with a lower MED is a preferred.

MEDs learned from an eBGP neighbor MAY be propagated over iBGP to other BGP speakers within the same AS. MEDs received from an eBGP neighbor MUST NOT be propagated to any other eBGP neighbor.

BGP Path Attributes – LOCAL_PREF

The local preference attribute (LOCAL_PREF) is a well-know attribute that is mandatory in all UPDATE messages a BGP speaker sends to its iBGP neighbors. LOCAL_PREF is internal to an AS and is never sent to eBGP neighbors (except BGP confederations).

The higher the value, the higher the preference indicated for a route.

It is simplest to set LOCAL_PREF when a route is learned from an eBGP neighbor. While LOCAL_PREF can be modified by other routers within an AS, this must be approached with caution, as it can result in routing loops. This is beyond the scope of this presentation.

BGP Path Attributes – COMMUNITIES

A BGP COMMUNITY is an optional transitive attribute.

The COMMUNITIES attribute allows a route to be **tagged** with one or more community numbers. There are a few communities that are defined by IANA as well-known, but most are locally defined by the operators of an AS.

RFC1997 defined the first COMMUNITIES attribute for BGP, which consisted of a string of 32-bit community values. These are typically used as two 16-bit unsigned values separated by a colon when written. The first 16 bits is the AS number of network utilizing the community. The second 16 bits is a locally significant value for the AS. **ASN:value**

RFC4360 introduced Extended Communities, which provides a 64-bit community. The first 8 or 16 bits is a type field, defining the use of the community. The format of the remaining bits is defined by the specific type of community. Extended communities are generally not used in eBGP policy.

BGP Path Attributes – COMMUNITIES

When BGP COMMUNITIES were first introduced by RFC1997 (1996), AS numbers were 16-bit values. By the early 2000s, it was clear 16-bits was insufficient for AS numbers and eventually 32-bit AS number were introduced by RFC4893 (2007). Unfortunately, COMMUNITIES in their typical asn:value format only allowed for 16-bit ASNs.

RFC8092 (2017) introduced Large Communities to address 32-bit AS numbers and the need for more flexibility in the use of communities. Large communities are 96 bits, three 32-bit unsigned values. It consists of a 32-bit administrator value followed by two 32-bit operator defined values. It is typically written as **ASN:value1:value2**.

BGP Path Attributes – COMMUNITIES

BGP COMMUNITIES allow additional information to be added to a route. The operator gets to decide the definition of these values. They may be used only internally by the operator of an AS or an AS operator might choose to accept COMMUNITIES from their eBGP neighbors.

COMMUNITIES can be classified into two categories:

- **Informational:** these tags are set by the operator of an AS to augment the route with additional meta-data. This information might be used within the AS, but may also be sent to eBGP neighbors, so they or other interested parties have access to the encoded information.

- **Action**: these tags are set by eBGP neighbors of an AS to influence the routing policies of an AS. Typically, these are set by a customer AS to influence the routing policies of a provider AS.

BGP Path Attributes – COMMUNITIES

Informational communities are often used for

- Tagging the **source of a route**. Was the route learned from an eBGP neighbor or is it an internal route? Was the eBGP neighbor an upstream transit provider, a peer, or a customer. If you only use communities for one thing in your network, it should be for this purpose.
- **Geographical** tagging. Where was this route learned? Which POP, city, region, country, and continent?

BGP Path Attributes – COMMUNITIES

Action communities are often used for

- Controlling whether or not a prefix should be advertised to
  - A specific neighbor AS
  - A specific region
  - A specific type of neighbor AS
  - etc
- Controlling if any BGP attributes should be modified
  - LOCAL_PREF override
  - AS_PATH extra prepending
  - NEXT_HOP manipulation (for example RTBH signaling)
  - Requesting DDoS mitigation
  - etc

BGP Path Attributes – COMMUNITIES

11164:51240	action	Set local-pref higher than default
11164:51200	action	Set local-pref lower than default
11164:53666	action	RTBH signal - set next hop to discard traffic
11164:7500	info	Route learned from customer
11164:7880	info	Route learned from public peer
11164:7890	info	Route learned from private peer
65000:ASN	action	Do not advertise this route to ASN
650001:ASN	action	Prepend 1 extra hop to this ASN
65002:ASN	action	Prepend 2 extra hops to this ASN

BGP Path Attributes – COMMUNITIES

It is important for an AS operator to control who can set BGP communities that have internal significance for the AS. Typically,

- Informational communities of the form ASN:value are only set by the AS operator. The AS operator must not accept any of its informational communities from other eBGP neighbors.

- Action communities of the form ASN:value and possibly privateASN:value are typically accepted from customers of an AS, but not from peers.

BGP Path Attributes – COMMUNITIES

Many router operating system configurations treat communities as text strings, rather than numeric values. In route policy, BGP communities are often matched using regular expressions that match the community text.

When choosing values to use for BGP communities, it can be very helpful to assign informational and action communities in a way that make it easier to match with a simple regular expression for filtering.

I highly recommend checking out the NANOG40 *BGP Communities* presentation on the references page for some great design suggestions for BGP communities.

BGP Path Attributes – COMMUNITIES

11164:**5**1240	action	Set local-pref higher than default
11164:**5**1200	action	Set local-pref lower than default
11164:**5**3666	action	RTBH signal - set next hop to discard traffic
11164:**7**500	info	Route learned from customer
11164:**7**880	info	Route learned from public peer
11164:**7**890	info	Route learned from private peer
65000:ASN	action	Do not advertise this route to ASN
650001:ASN	action	Prepend 1 extra hop to this ASN
65002:ASN	action	Prepend 2 extra hops to this ASN

BGP Path Attributes – References

- RFC4271: A Border Gateway Protocol 4 (BGP-4)
  - https://www.rfc-editor.org/rfc/rfc4271.html
- draft-ietf-grow-as-path-prepending
  - https://datatracker.ietf.org/doc/draft-ietf-grow-as-path-prepending/
- APNIC blog: Excessive BGP AS-PATH prepending is a self-inflicted vulnerability
  - https://blog.apnic.net/2019/07/15/excessive-bgp-as-path-prepending-is-a-self-inflicted-vulnerability/

- RFC1997: BGP Communities Attribute
  - https://www.rfc-editor.org/rfc/rfc1997.html
- RFC8092: BGP Large Communities Attribute
  - https://www.rfc-editor.org/rfc/rfc8092.html
- RFC8195: Use of BGP Large Communities
  - https://www.rfc-editor.org/rfc/rfc8195.html

BGP Path Attributes – References, COMMUNITIES

- NANOG40: BGP Communities: A Guide for Service Provider Networks
  - Slides: https://archive.nanog.org/meetings/nanog40/presentations/BGPcommunities.pdf
- Internet2 BGP Community Documentation
  - https://noc.net.internet2.edu/i2network/maps-documentation/documentation/bgp-communities.html
- OneStep BGP Community Guides Archive
  - https://onestep.net/communities/
- NLNOG Looking Glass BGP Community Documentation
  - https://github.com/NLNOG/lg.ring.nlnog.net/tree/main/communities
- IANA BGP Well-known Communites
  - https://www.iana.org/assignments/bgp-well-known-communities/bgp-well-known-communities.xhtml
- A Survey of the Utilization of the BGP Community Attribute
  - https://datatracker.ietf.org/doc/html/draft-quoitin-bgp-comm-survey-00.txt

BGP Best Path Algorithm

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

BGP Best Path Algorithm

While a router's BGP table may contain multiple different routes to a single destination prefix, it must select just one of these routes to be the best path. Typically*, only this best path route is eligible to be sent to a BGP speaker's iBGP and eBGP neighbors.

* RFC7911: *Advertisement of Multiple Paths in BGP (ADD-PATH)* defines a BGP extension that allows additional paths beyond the best path to be sent.

BGP Best Path Algorithm

It's important to understand the algorithm used by your specific router vendor. The goal of the algorithm is to consistently select a single best path based on attributes of the available routes to a prefix.

Each vendors' approach differs in the details, but at a high level, most include the following components:

1. Prefer the route with the highest LOCAL_PREF
2. Prefer the route with the shortest AS_PATH
3. Prefer the route with the lowest MED
4. Prefer the route learned from a local eBGP neighbor over external routes learned via iBGP
5. Prefer the route with the lowest IGP metric

BGP Best Path Algorithm – References

- Cisco BGP Best Path Algorithm
  - https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html
- Juniper BGP Best Path Algorithm
  - https://www.juniper.net/documentation/us/en/software/junos/bgp/topics/topic-map/bgp-overview.html#id-10119586

Route Preferences In R&E Networks

INTERNET2
2023
COMMUNITY exchange

May 8-11, 2023 Atlanta, GA

Route Preference – Typical LOCAL_PREF use

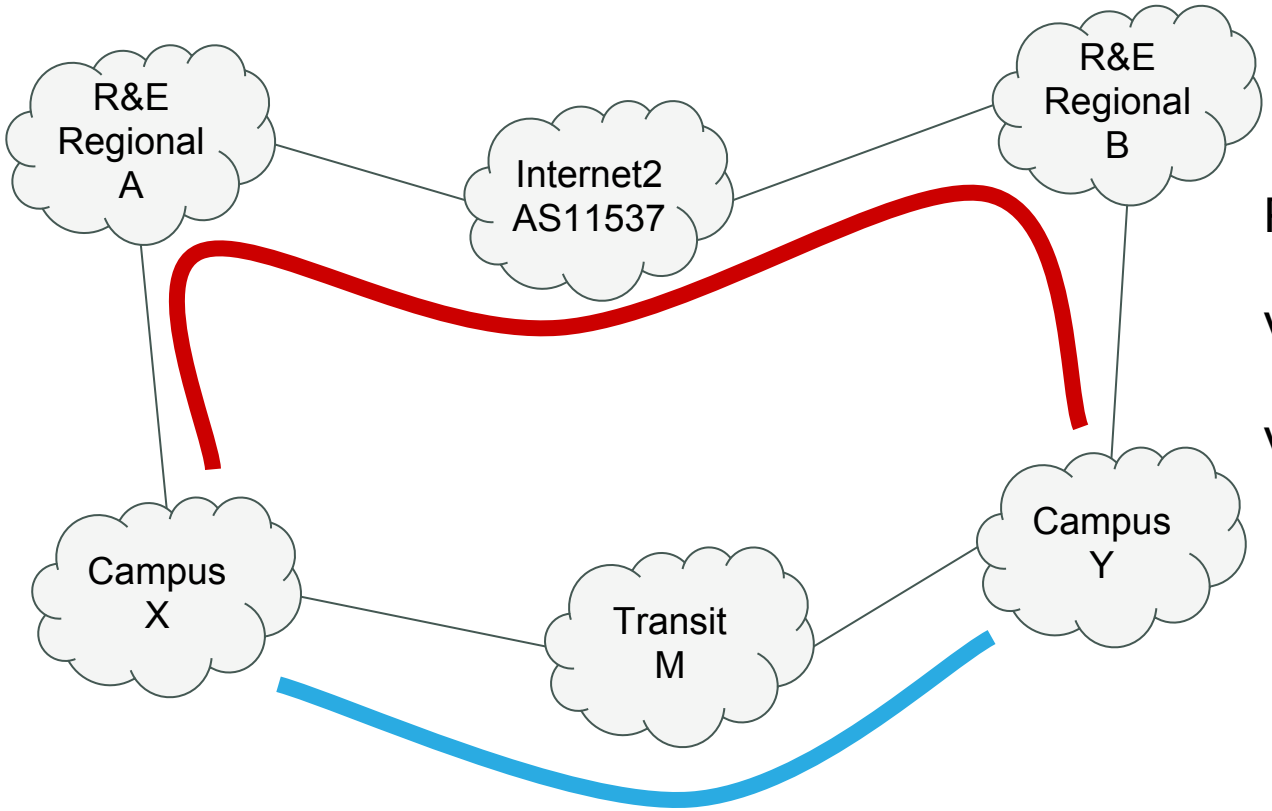
In a typical network, LOCAL_PREF is used to rank route preference. A ranking such as

- Local-pref 200: customer route
- Local-pref 100: peer route
- Local-pref 50: paid transit route

This ranking would implement an operator's policy to prefer route from settlement-free peers over routes from paid transit, which frequently are more costly.

Customer routes should by default be preferred over routes learned from other sources. If customer routes are not preferenced higher, there is the possibility that they might not be selected as best path and then they wouldn't be advertised to BGP neighbors, defeating the purpose of being a customer.

Route Preferences – Shortest AS_PATH isn't always optimal

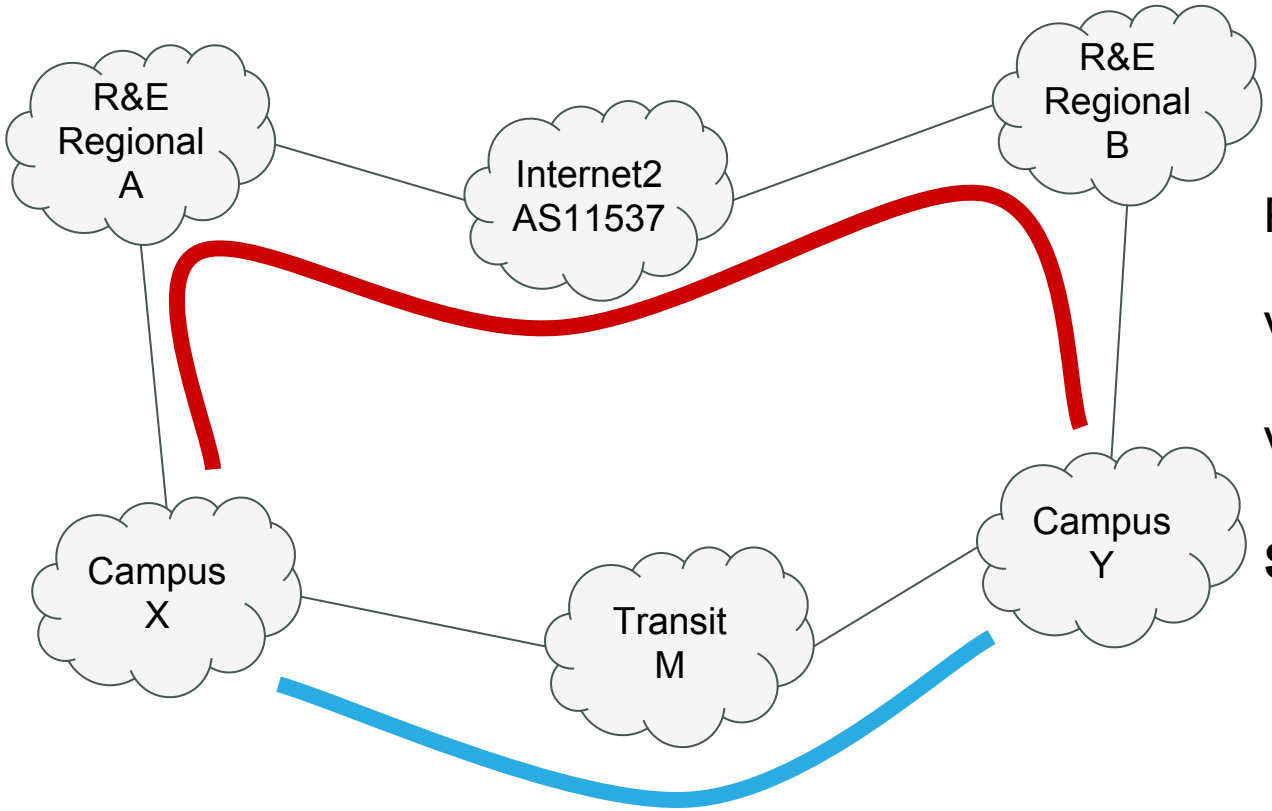


Path between X and Y:

Via Internet2 = 4 hops

Via paid transit = 2 hops

Route Preferences – Shortest AS_PATH isn't always optimal



Path between X and Y:

Via Internet2 = 4 hops

Via paid transit = 2 hops

Solution: LOCAL_PREF

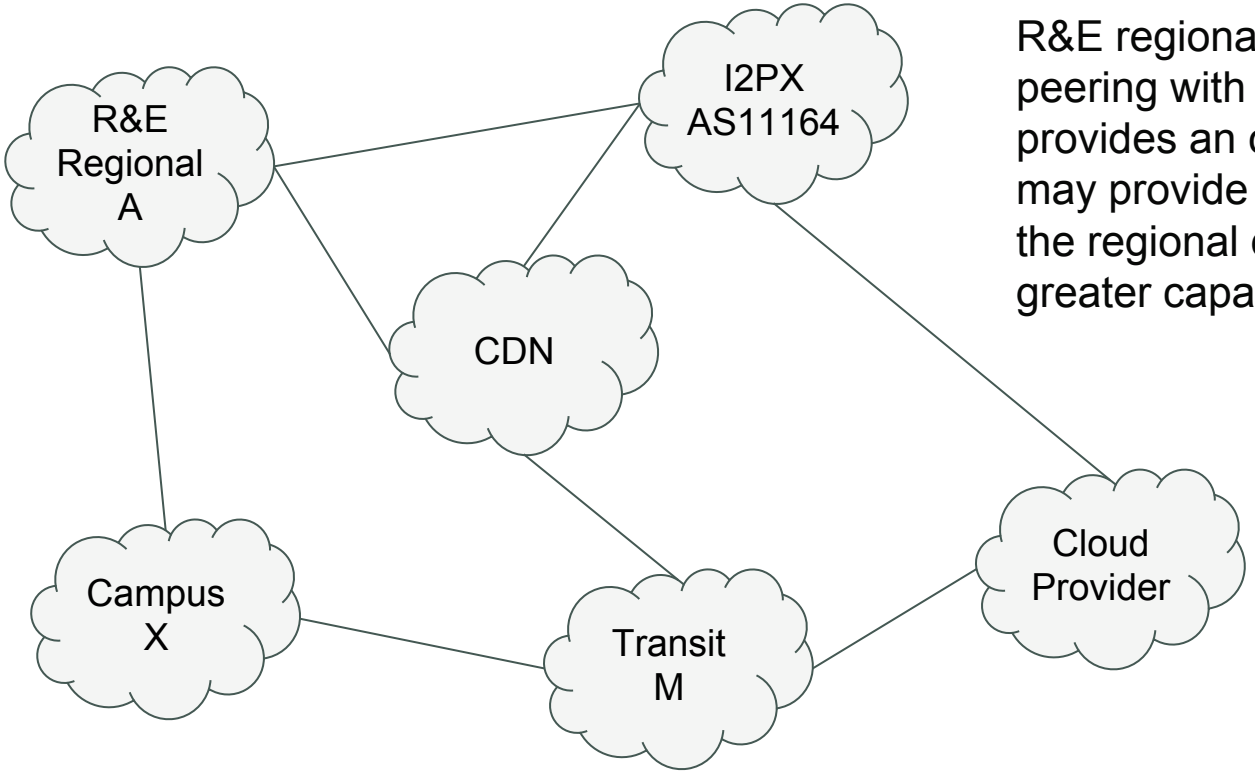
Route Preference – R&E LOCAL_PREF use

A network that receives both R&E and commodity routes should prioritize R&E routes over commodity routes.

- Local-pref 200: customer route
- Local-pref 150: R&E peer route
- Local-pref 100: commodity peer route
- Local-pref 50: paid commodity transit route

This ranking only addresses the route taken to egress the R&E network. A network can really only control where it sends a packet next. But if all R&E networks implement this approach, then traffic in the opposite direction should also prefer R&E network routes.

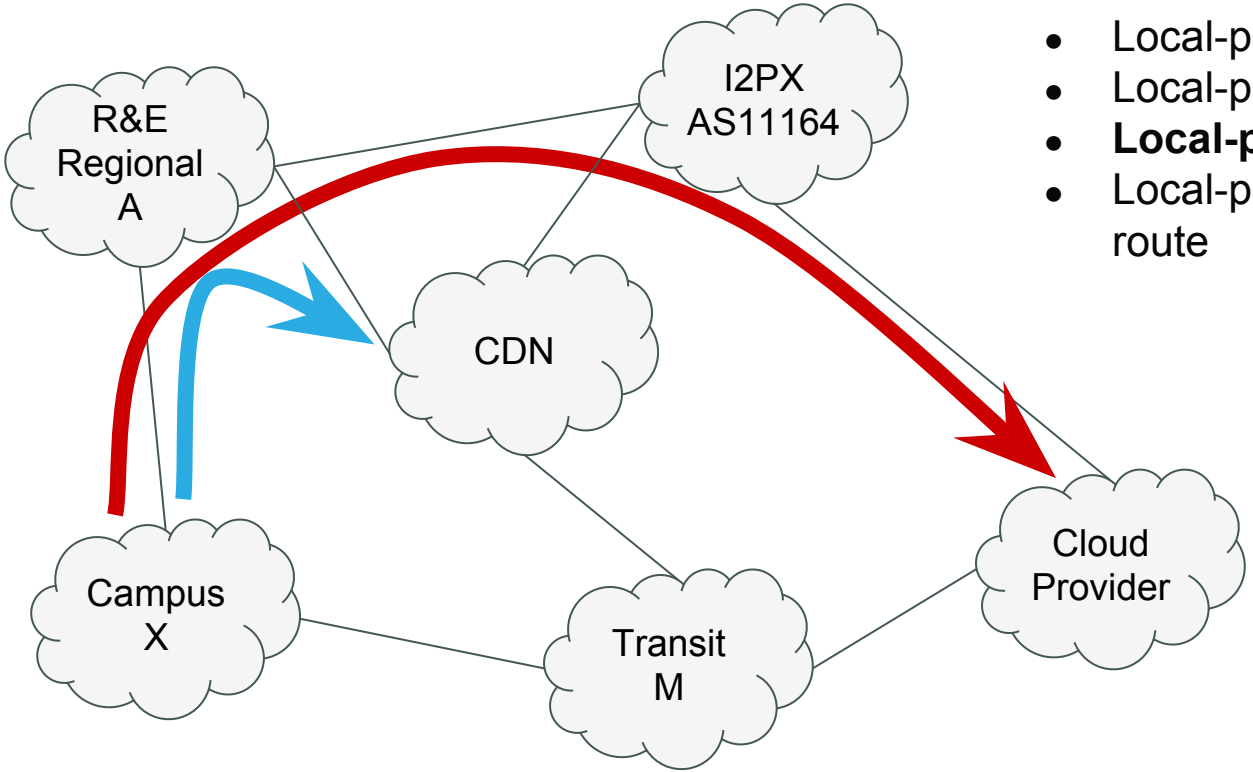
Route Preferences – Optimizing Peering routes – egress?



R&E regional network may have local peering with commercial networks that provides an optimal low-latency path. I2PX may provide routes to commercial networks the regional doesn't peer with or that have greater capacity.

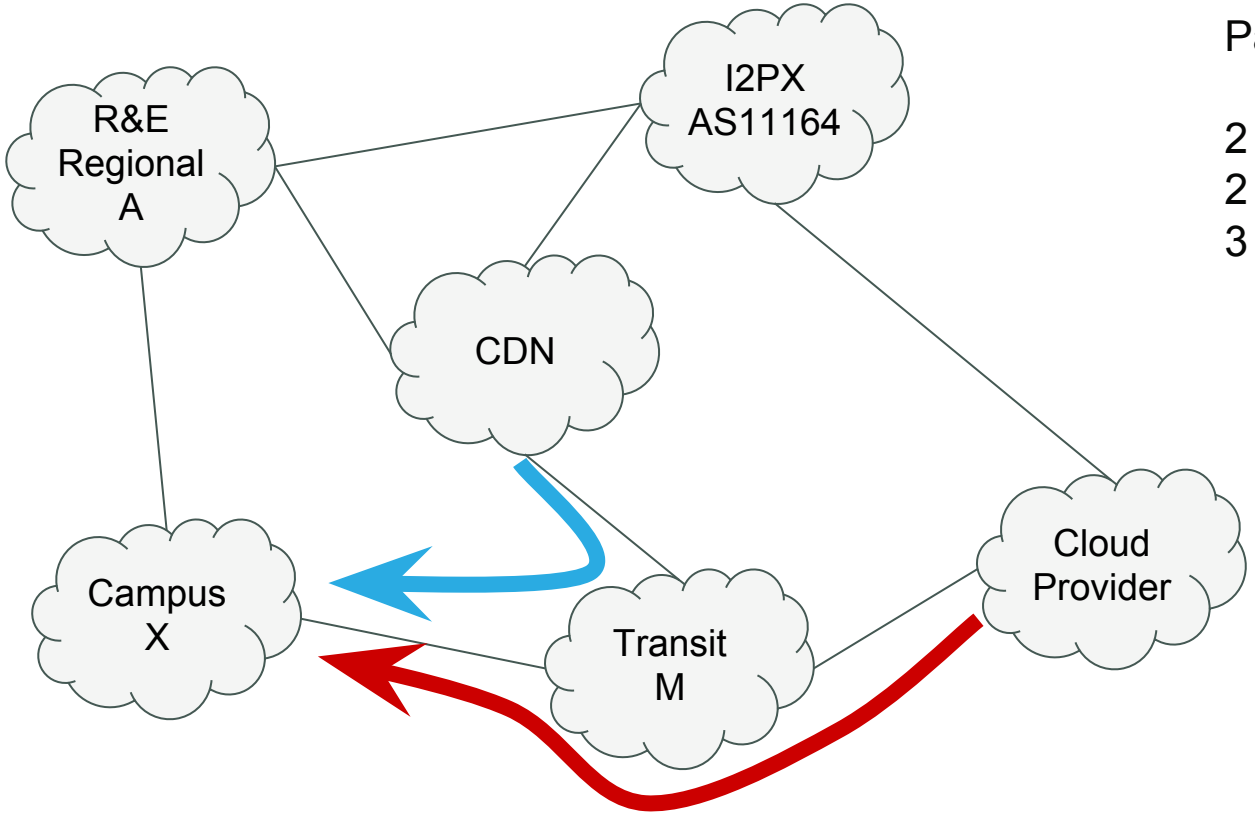
The AS_PATH between the campus and the commercial networks may be shorter via paid transit, but a path via the regional network to direct peers or to I2PX might be preferable operationally.

Route Preferences – Optimizing Peering routes – egress



- Local-pref 200: customer route
- Local-pref 150: R&E peer route
- **Local-pref 100:** commodity peer route
- Local-pref 50: paid commodity transit route

Route Preferences – Optimizing Peering routes – ingress?



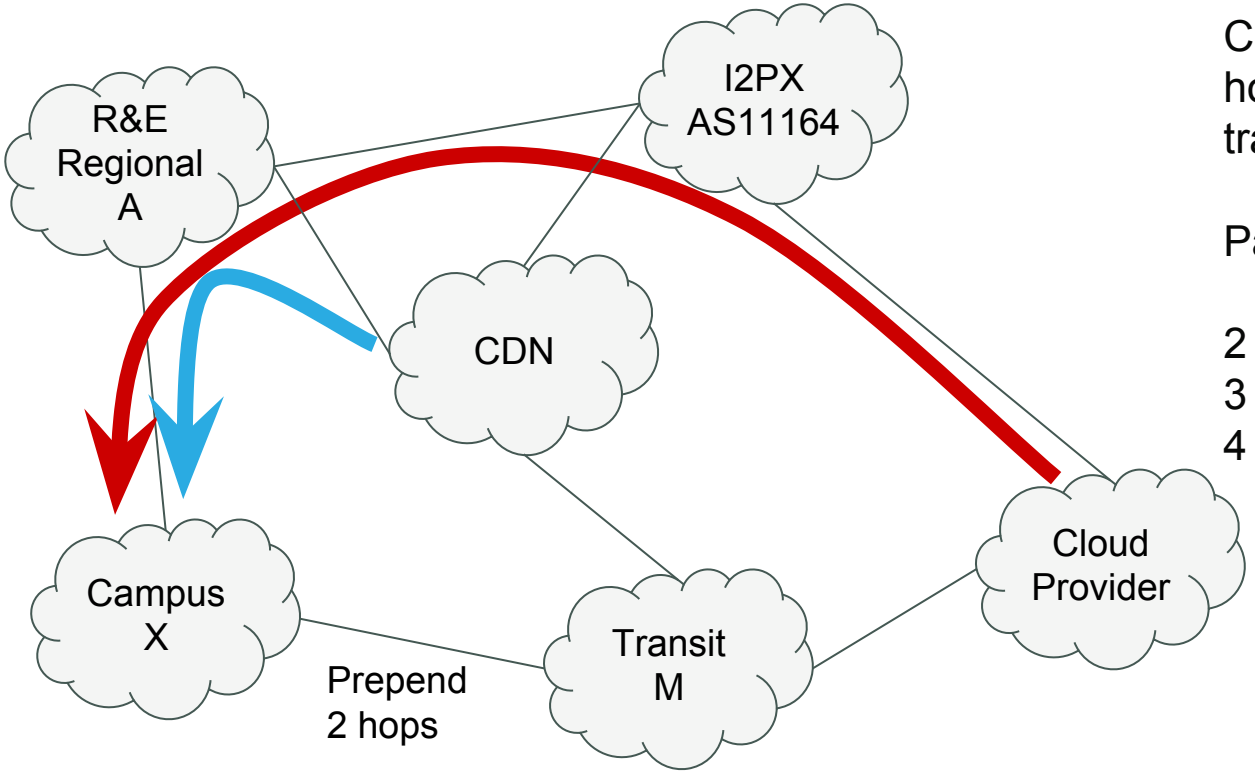
Path from CDN to Campus X:

2 hops via Regional
2 hops via Transit
3 hops via I2PX

Path from Cloud Provider to Campus X:

2 hops via Transit
3 hops via I2PX

Route Preferences – Optimizing Peering routes – ingress



Campus prepends an extra two hops on advertisements to paid transit.

Path from CDN to Campus X:

2 hops via Regional
3 hops via I2PX
4 hops via Transit

Path from Cloud Provider to Campus X:

3 hops via I2PX
4 hops via Transit

Route Policy Components

INTERNET2
2023
COMMUNITY exchange
May 8-11, 2023 Atlanta, GA

Route Policy Components – Ingress

1. Select routes for further consideration. prefix-set filter routes from eBGP customers, possibly also from peers, dropping all others. Allow more specifics for now, if implementing RTBH
2. Drop routes from eBGP customers with ASNs in the AS path not belonging to the customer or their downstream customers
3. Drop bad/unexpected routes
4. Drop route leaks using peerlock/tier 1 AS path filtering
5. Scrub BGP communities
6. Optional: RTBH for eBGP customers
7. Drop long prefixes
8. Drop RPKI ROV invalids
9. Set defaults: communities, local-pref, possibly clear MED
10. Optional: local-pref override communities for eBGP customers
11. Graceful Shutdown processing for received routes
12. Optional: Graceful Shutdown for signalling maintenance of this router

Route Policy Components – Egress

1. Select routes for further consideration using BGP communities set on routes at ingress. Typically, an AS should advertise customer eBGP neighbors routes tagged as learned from transit, peer, customer, and internal sources. For transit and peer eBGP neighbors, an AS should advertise only routes tagged as learned from a customer or internal source. Drop all other routes from further consideration. **Do not select routes using only a prefix-set!**
2. Remember to configure ‘remove-private-as’ for all eBGP neighbors.
3. Drop bad/unexpected routes
4. Scrub BGP communities, only if absolutely necessary
5. Set defaults: communities, clear MED unless required
6. Optional: process action communities for prepending and dropping routes
7. Optional: Graceful Shutdown for signalling maintenance of this router

Route Policy Components – References

- NANOG67: Everyday Practical BGP Filtering...., slides 28-29
  - https://archive.nanog.org/sites/default/files/Snijders_Everyday_Practical_Bgp.pdf#page=28
  - While this presentation focuses on Peer Locking AS_PATH filtering, slides 28-29 have a concise overview of the components of ingress and egress route policies.
- RIPE77: Robust Routing Policy Architecture
  - https://ripe77.ripe.net/wp-content/uploads/presentations/59-RIPE77_Snijders_Routing_Policy_Architecture.pdf
-

Ingress Selecting Routes to Consider – Overview

Before a provider accepts a route from a customer, it should confirm that the route is to a prefix that belongs to the customer or a downstream customer of the customer. The typical approach is to create a prefix-set for each customer containing the vetted routes. Many providers require their customers to provide a written LOA stating that they own the prefix and give the provider permission to advertise it.

Eventually, when RPKI is widely adopted, ROAs could be used to automatically build these prefix-sets.

While some providers use IRR data to build their prefix-sets, this is dangerous, since several widely used IRRs don't authenticate the data they allow into their IRR.

Ingress Selecting Routes to Consider – Overview

Routes received from peers can also be filtered with a prefix-set. The only method to easily automate this task is to use IRR data. While not perfect due to the authentication issues for the data, there is no better solution today. Tools like bgpq4 can be used to build a prefix-set for each peer based on their IRR AS-SET.

Ingress Selecting Routes to Consider – Config Examples

Every customer of a provider will need its own custom route-policy and prefix-set(s) to select their routes.

```
# parent policy calls using an 'if apply'

if apply EBGP-CUST-XYZ-SELECT-IN then
    pass
  else
    drop
  endif
```

IOS-XR

```
route-policy EBGP-CUST-XYZ-SELECT-IN
  if destination or-longer EBGP-CUST-XYZ-V4-IN then
    pass
    done
  endif
  if destination or-longer EBGP-CUST-XYZ-V6-IN then
    pass
    done
  endif
end-policy
```

Egress Selecting Routes to Consider – Config Examples

If all routes are tagged at ingress, selecting routes at egress is a simple matter of looking at the community tags.

```
route-policy EBGP-CUST-SELECT-OUT
  if community matches-any EBGP-CUST or community
matches-any EBGP-INTERNAL or community matches-any
EBGP-PEER or community matches-any EBGP-TRANSIT then
    pass
  else
    drop
  endif
end-policy

route-policy EBGP-PEER-SELECT-OUT
  if community matches-any EBGP-CUST or community
matches-any EBGP-INTERNAL then
    pass
  else
    drop
  endif
end-policy

route-policy EBGP-TRANSIT-SELECT-OUT
  if community matches-any EBGP-CUST or community
matches-any EBGP-INTERNAL then
    pass
  else
    drop
  endif
end-policy
```

Ingress/Egress Selecting Routes– References

- MANRS for Network Operators
  - https://www.manrs.org/netops/network-operator-actions/
- RIPE77: Robust Routing Policy Architecture, slides 16-23
  - https://ripe77.ripe.net/wp-content/uploads/presentations/59-RIPE77_Snijders_Routing_Policy_Architecture.pdf#page=16
- bgpq4 utility
  - https://github.com/bgp/bgpq4

Bad Prefixes and Routes – Overview

There are prefixes and route attributes that should never appear in the global BGP routing table. There are also routes that an AS may never want to accept from other AS. These should generally be dropped if received from an eBGP neighbor.

- **Bogon Prefixes**: These are prefixes that should never appear in the global routing table. Many of these are reserved for special purposes defined in RFCs such as RFC1918 (private), RFC5735 (special use prefixes), RFC6598 (shared space/CG-NAT), and netblocks that have not been allocated to a regional Internet registry (RIR).

- **Bogon ASN**: Similarly, bogon ASNs are ASNs that are unallocated or private/reserved. Routes should not appear in the global routing table that contain bogon ASNs anywhere in their paths. Private ASNs may be used internally within a network, but if they are passed in an AS path to an eBGP neighbor, then they are considered bogons by the receiving network. An exception might be made for customer eBGP neighbors using a private ASN, as long as the AS strips the private ASN when advertising the route to any other eBGP neighbor.

Bad Prefixes and Routes – Overview

- **Prefixes Originated by AS**: An AS should drop any prefixes that it originates that are received from eBGP neighbors, including more specific prefixes. If a more specific route to a network's prefixes is received from an eBGP neighbor, that can result in internal traffic being hijacked by another network. An exception to this is when a provider delegates subnets to downstream customers. The AS needs to accept the prefix from the customer's eBGP neighbors. If the customer is multihomed, then the customer's prefix also needs to be accepted from transit and peer eBGP neighbors.
- **Default Routes**: The only instance where an AS should accept a default route is when it has been requested from an upstream transit provider. Default routes should be dropped if received from any other eBGP neighbors.
- **Public Peering Exchange Prefixes**: If an AS participates in public peering exchanges, a rogue more specific route can cause traffic that has a next-hop of an IXP eBGP neighbor to be hijacked to an unexpected next-hop. Never accept IXP prefixes from eBGP neighbors.

Bad Prefixes – Bogon Prefixes – Config Examples

Make sure to also drop any more specific routes to bogon prefixes. IOS-XR matches more specifics when the ‘or-longer’ parameter is added.

Junos implements this capability in both the policy-statement ‘prefix-list-filter … orlonger’ and ‘route-filter’ match conditions.

```
route-policy EBGP-REJECT-BOGON-PREFIXES
  if destination or-longer EBGP-BOGONS-V4 or
destination or-longer EBGP-BOGONS-V6 then
    drop
  endif
end-policy
```

IOS-XR

```
prefix-set EBGP-BOGONS-V4
  # 'this' network [RFC1122]
  0.0.0.0/8,
  # private space [RFC1918]
  10.0.0.0/8,
  172.16.0.0/12,
  192.168.0.0/16,
  # CGN Shared [RFC6598]
  100.64.0.0/10,
  # localhost [RFC1122]
  127.0.0.0/8,
  # link local [RFC3927]
  169.254.0.0/16,
  # IANA Special-Purpose [RFC6890]
  192.0.0.0/24,
  # documentation TEST-NET-1 [RFC5737]
  192.0.2.0/24,
  # benchmarking [RFC2544]
  198.18.0.0/15,
  # documentation TEST-NET-2 [RFC5737]
  198.51.100.0/24,
  # documentation TEST-NET-3 [RFC5737]
  203.0.113.0/24,
  # multicast
  224.0.0.0/4,
  # reserved for future use
  240.0.0.0/4
end-set
```

Bad Prefixes – Bogon Prefixes – Config Examples

Make sure to also drop any more specific routes to bogon prefixes. IOS-XR matches more specifics when the ‘or-longer’ parameter is added.

Junos implements this capability in both the policy-statement ‘prefix-list-filter … orlonger’ and ‘route-filter’ match conditions.

```
route-policy EBGP-REJECT-BOGON-PREFIXES
  if destination or-longer EBGP-BOGONS-V4 or
destination or-longer EBGP-BOGONS-V6 then
    drop
  endif
end-policy
```

IOS-XR

```
prefix-set EBGP-BOGONS-V6
  # IPv4-compatible, loopback, et al
[RFC4291]
  ::/8,
  # RTBH [RFC6666]
  0100::/64,
  # BMWG [RFC5180]
  2001:2::/48,
  # ORCHID [RFC4843]
  2001:10::/28,
  # documentation [RFC3849]
  2001:db8::/32,
  # 6to4 anycast relay [RFC7526]
  2002::/16,
  # old 6bone [RFC3701]
  3ffe::/16,
  # unique local unicast [RFC4193]
  fc00::/7,
  # link local unicast [RFC4291]
  fe80::/10,
  # old site local unicast [RFC3879]
  fec0::/10,
  # multicast [RFC4291]
  ff00::/8
end-set
```

Bad Prefixes – Bogon ASN – Config Examples

Remember that you may need an exception for customers that use a private ASN for their eBGP neighbor. eBGP neighbor config should strip private ASNs when advertising routes. This can be configured using the IOS-XR ‘remove-private-as’ and Junos ‘remote-private’ BGP neighbor options.

```
route-policy EBGP-REJECT-BOGON-ASN
   # Check for bogon ASN (reserved, documentation)
  if as-path in EBGP-BOGON-ASNS then
    drop
  endif
end-policy
```

IOS-XR

```
as-path-set EBGP-BOGON-ASNS
  # RFC7607 - AS0 Processing,
  ios-regex '_0_',
  # 2 to 4 byte ASN migrations,
  passes-through '23456',
  # RFC5398 - Documentation ASNs,
  passes-through '[64496..64511]',
  passes-through '[65536..65551]',
  # RFC7300 - Highest ASNs Reserved,
  passes-through '65535',
  passes-through '4294967295',
  # IANA reserved,
  passes-through '[65552..131071]'
end-set
```

Bad Prefixes – Originated Prefixes – Config Examples

```
route-policy EBGP-REJECT-ORIGINATED
  # RFC7454 6.1.4 - recommends filtering prefixes originated by
  # network to prevent external hijacking of local traffic
  if destination or-longer EBGP-I2PX-ORIGINATED-V4 or destination
or-longer EBGP-I2PX-ORIGINATED-V6 then
    drop
  endif
end-policy
```

IOS-XR

```
prefix-set EBGP-ORIGINATED-V4
  192.0.2.0/24,
  198.51.100.0/24,
  203.0.113.0/24
end-set

prefix-set EBGP-ORIGINATED-V6
  2001:db8::/32
end-set
```

Bad Prefixes – Default Routes – Config Examples

```
route-policy EBGP-REJECT-DEFAULT
  # RFC7454 6.1.6 - recommends filtering default from unexpected
sources
  if destination in (0.0.0.0/0, ::/0) then
    drop
  endif
end-policy
```

IOS-XR

Bad Prefixes – IXP Prefixes – Config Examples

Drop IXP prefixes and more specific routes from all eBGP neighbors. The prefix-sets should contain at least the IXPs the AS participates at, but could be extended to include any or all IXP prefixes, since none should appear in the global routing table.

```
route-policy EBGP-REJECT-IXP-PREFIXES
  # RFC7454 6.1.5.1 - recommends filtering public IXP prefixes
  # prefix-sets should contain all public exchanges used by AS
  if destination or-longer EBGP-IXP-V4 or destination or-longer
EBGP-IXP-V6 then
    drop
  endif
end-policy
```

IOS-XR

```
prefix-set EBGP-IXP-V4
  198.32.160.0/23,
  198.32.176.0/24,
  206.126.115.0/24,
  206.126.236.0/22,
  206.223.116.0/23,
  206.223.118.0/23,
  206.223.123.0/24,
  206.223.143.0/24,
  206.72.210.0/23,
  206.81.80.0/23,
  206.81.82.0/23,
  206.82.104.0/22,
  208.115.136.0/23
end-set
!
prefix-set EBGP-IXP-V6
  2001:504:0:1::/64,
  2001:504:0:2::/64,
  2001:504:0:3::/64,
  2001:504:0:4::/64,
  2001:504:0:5::/64,
  2001:504:13::/64,
  2001:504:16:1::/64,
  2001:504:16::/64,
  2001:504:17:115::/64,
  2001:504:1::/64,
  2001:504:36::/64,
  2001:504:d::/64
end-set
```

Bad Prefixes and Routes – References

- NLNOG BGP Filter Guides
  - https://bgpfilterguide.nlnog.net
  - Configuration examples for several router operating systems for filtering bogon prefixes, bogon ASNs, excessively long AS paths, and other bad routes.
- MANRS: Routing Security Terms: Bogons, Vogons, and Martians
  - https://www.manrs.org/2021/01/routing-security-terms-bogons-vogons-and-martians/
- NTT: Bogon ASN Filter Policy Examples
  - http://as2914.net/bogon_asns/configuration_examples.txt
- RFC7454: Section 6.1.4. Filtering Prefixes Belonging to the Local AS and Downstreams
  - https://tools.ietf.org/html/rfc7454#section-6.1.4
- RFC7454: Section 6.1.6. Filtering Default Route
  - https://tools.ietf.org/html/rfc7454#section-6.1.6
- RFC7454: Section 6.1.5.1. Filtering IXP Prefixes
  - https://tools.ietf.org/html/rfc7454#section-6.1.5.1

Bad Prefixes and Routes – References

- Team Cymru Bogon Reference
  - https://team-cymru.com/community-services/bogon-reference/
  - https://www.team-cymru.com/bogon-reference-http
  - Team Cyrmu publishes both a standard Bogon prefix list and a Fullbogons list. The standard list includes any prefix ranges that have not been allocated to RIRs. The Fullbogons list is much larger and includes prefixes allocated to RIRs that have not yet been assigned to end-user or service provider networks. The standard bogon list is fairly static, while Fullbogons changes frequently as RIRs make assignments of address space. Use of Fullbogons requires an automated update solution to make sure newly assigned prefixes are not dropped. Their bogon lists can be downloaded or received via BGP.
-

Route Leak Protection Filtering – Overview

A route leak happens when an AS advertises routes it learns from a transit provider or peer eBGP neighbor to another transit or peer eBGP neighbor. Routes learned from transit and peer eBGP neighbors should only be advertised to customers.

The best route leak prevention is for all AS to make use of BGP communities to tag route source types and then to use those for determining which routes to advertise to each type of eBGP neighbor.

Route Leak Protection Filtering – Overview

Job Snijders, while working for NTT, proposed the PeerLock method for networks to drop route leaks. It requires a network to contact each of their eBGP peers and learning which transit providers, if any, the peer uses. Utilizing this data, an AS Path filter could be generated for all of NTT's peer eBGP neighbors, which would allow or drop routes with AS paths containing the AS of a peer based on PeerLock data the peer provided. This requires a lot of work and likely very trusted relationships with each peer to gather the required information. Very useful, though, for large networks willing to put in the effort.

For the rest of us, a popular approach that has been used for a long time is to drop routes from customers and peers when the AS path contains the ASN of a tier 1 transit provider. A tier 1 network does not receive transit from any other network, so a tier 1 will never be a customer of a peer network or a downstream customer of a customer.

Route Leak Protection Filtering – Config Examples

```
route-policy EBGP-REJECT-PEERLOCKLITE
  if as-path in EBGP-TIER1-ASNS then
    drop
  endif
end-policy
```

IOS-XR

```
as-path-set EBGP-TIER1-ASNS
  # https://en.wikipedia.org/wiki/Tier_1_network,
  passes-through '174',
  passes-through '209',
  passes-through '286',
  passes-through '701',
  passes-through '1239',
  passes-through '1299',
  passes-through '2828',
  passes-through '2914',
  passes-through '3257',
  passes-through '3320',
  passes-through '3356',
  passes-through '3491',
  passes-through '3549',
  passes-through '5511',
  passes-through '6453',
  passes-through '6461',
  passes-through '6762',
  passes-through '7018',
  passes-through '12956'
end-set
```

Route Leak Protection Filtering – References

- NANOG67: Practical everyday BGP filtering with AS_PATH filters: Peer Locking
  - https://archive.nanog.org/sites/default/files/Snijders_Everyday_Practical_Bgp.pdf
- NTT Peer Locking Documentation
  - http://instituut.net/~job/peerlock_manual.pdf
- Wikipedia Tier 1 Network List
  - https://en.wikipedia.org/wiki/Tier_1_network#List_of_Tier_1_networks
  - Many peering relationships, especially with larger networks, are hidden in the secrecy of non-disclosure agreements. While there is no perfect source of information about the peering relationships between networks, the list of tier 1 networks in the Wikipedia seems to be well maintained.

BGP Community Scrubbing – Config Examples

On ingress, clear any informational BGP community tags. It is very helpful for scrubbing, if a network's communities are well organized into informational and action groups using methods easy to match in a regular expression, such as grouping by the first character of the value or possibly by the length of the value string.

Do you use MPLS L3VPNs? What would happen if someone outside your network was to send a route target extended community? Does your router protect from this? Best to just delete extended communities.

```
route-policy EBGP-COMMSCRUB-CUST
  # customers should never set informational communities,
  # which always have a value starting with 1, 7, 8, or 9.
  delete community in (ios-regex '^11164:[1,7-9]')
  delete large-community in (ios-regex '^11164:[1,7-9]:*')

  # customers should never send any extended communities
  delete extcommunity color all
  delete extcommunity rt all
  delete extcommunity soo all
end-policy
```

IOS-XR

BGP Community Scrubbing – Config Examples

On ingress, clear any informational BGP community tags. It is very helpful for scrubbing, if a network's communities are well organized into informational and action groups using methods easy to match in a regular expression, such as grouping by the first character of the value or possibly by the length of the value string.

Do you use MPLS L3VPNs? What would happen if someone outside your network was to send a route target extended community? Does your router protect from this? Best to just delete extended communities.

```
route-policy EBGP-I2PX-PEER-COMMSCRUB
  # peers should never send any I2PX specific communities
  delete community in (11164:*)
  delete large-community in (11164:*:*)

  # peers should never send any extended communities
  delete extcommunity color all
  delete extcommunity rt all
  delete extcommunity soo all
end-policy
```

IOS-XR

BGP Community Scrubbing – References

- NANOG40: BGP Communities: A Guide for Service Provider Networks
  - https://archive.nanog.org/meetings/nanog40/presentations/BGPcommunities.pdf
  - So good, I have to mention it again.
- RFC7454: BGP Operations and Security, Section 11: BGP Community Scrubbing
  - https://tools.ietf.org/html/rfc7454#section-11
  - AS operators SHOULD scrub their own communities, but SHOULD NOT scrub other communities, in particular the well-known no-export community.
- RFC8642: Policy Behavior for Well-Known BGP Communities
  - https://tools.ietf.org/html/rfc8642
  - Exercise caution when deleting communities, for example don't delete '*:*'. Be aware when setting communities that some routers will delete existing communities unless an 'additive' type option is specified.

Remote Triggered Black Holing (RTBH) – Overview

Remote Triggered Black Holing (RTBH) provides a method for a network customer to tag a route with a BGP community that causes any traffic to the destination to be discarded. Frequently this is applied to a more-specific host route, rather than to an entire network prefix. RTBH allows for undesired traffic, such as a DDoS against a host, to be discarded, protecting the rest of the hosts on the network from congestion.

Remote Triggered Black Holing (RTBH) – Config Examples

```
community-set EBGP-RTBH
  11164:53666,
  65535:666
end-set

route-policy EBGP-RTBH
  if community matches-any EBGP-RTBH then
    set local-preference 260
    set community (no-export) additive

    # set next-hop to discard prefix
    if destination or-longer (0.0.0.0/0) then
      set next-hop 192.0.2.1
    elseif destination or-longer (::/0) then
      set next-hop 100::1
    endif
    pass   # return TRUE if RTBH is requested
  endif
  done   # return FALSE if not RTBH
end-policy
```

```
# parent policy calls RTBH using an 'if apply'

  if apply EBGP-I2PX-RTBH then
    done
  endif
```

IOS-XR

Remote Triggered Black Holing (RTBH) – References

- RFC5635: Remote Triggered Black Hole Filtering
  - https://tools.ietf.org/html/rfc5635
- RFC6666: A Discard Prefix for IPv6
  - https://tools.ietf.org/html/rfc6666
- RFC7999: BLACKHOLE Community
  - https://tools.ietf.org/html/rfc7999

Long Prefixes – Overview and Config Examples

When prefixes were selected at ingress for consideration from eBGP customer neighbors, more specific prefixes were temporarily allowed for RTBH purposes. Transit and peer (if no prefix filter is done) eBGP neighbors may also have sent excessively long prefixes. Time to drop those.

Typically in the Internet, most providers and peers will only accept up to a IPv4 /24 and an IPv6 /48 prefix length. R&E networks may accept and advertise longer prefixes.

```
route-policy EBGP-REJECT-CUST-LONGPREFIXES
  if destination in (0.0.0.0/0 ge 25, ::/0 ge 49) then
    drop
  endif
end-policy

route-policy EBGP-REJECT-LONGPREFIXES
  if destination in (0.0.0.0/0 ge 25, ::/0 ge 49) then
    drop
  endif
end-policy
```

IOS-XR

RPKI Route Origin Validation (ROV) – Overview

- Using RPKI ROV for filtering routes requires setting up or having access to a validation server and configuring each of your edge routers to retrieve validation data. Those details are outside the scope of this route policy presentation.
- When validation is enabled, routes are validated and assigned one of three states:
  - **Unknown**: this simply means that a ROA has not been created by the owner of this prefix
  - **Valid**: a ROA exists for this prefix that has a matching prefix length and origin AS
  - **Invalid**: a ROA exists for this prefix, but the prefix length and/or the origin AS do not match
- RPKI invalid routes should be dropped from all eBGP neighbors: transit, peers, and customers.
- An exception is if you are using private ASNs with customer BGP neighbors. In this special case, ROV can not be enabled, since a prefix with a private AS origin should be invalid. An option would be to explore adding the prefixes and private AS origins to an internal exception file in your validation server.

RPKI Route Origin Validation (ROV) – Config Examples

IOS-XR marking of routes with their validation state is handled in the BGP configuration. Junos handles this in the route policy. In both examples, any routes that are invalid are dropped.

```
route-policy EBGP-REJECT-ROV-INVALID
  if validation-state is invalid then
    drop
  endif
end-policy
```

IOS-XR

```
policy-statement EBGP-REJECT-ROV-INVALID {
  term reject_invalid {
    from {
      protocol bgp;
        validation-database invalid;
    }
    then {
      validation-state invalid;
      reject;
    }
  }
  term mark_valid {
    from {
      protocol bgp;
      validation-database valid;
    }
    then {
      validation-state valid;
      next policy;
    }
  }
  then {
    validation-state unknown;
    next policy;
  }
}
```

Junos

RPKI Route Origin Validation (ROV) – References

- NLNOG BGP Filter Guide: ROV
  - https://bgpfilterguide.nlnog.net/guides/reject_invalids/
- RPKI Documentation @ ReadTheDocs
  - https://rpki.readthedocs.io
- NIST SPECIAL PUBLICATION 1800-14B Protecting the Integrity of Internet Routing
  - https://www.nccoe.nist.gov/sites/default/files/legacy-files/sidr-nist-sp1800-14b-final.pdf
- RFC9324: Policy Based on the RPKI without Route Refresh
  - https://tools.ietf.org/html/rfc9324
  - If implementing ROV, make sure your router is configured to keep the full Adj-RIB-In table (all pre-policy routes received from a BGP neighbor). Juniper does this by default. Cisco requires configuration of 'soft reconfiguration inbound'. If this isn't done, neighboring routers can be impacted by frequent route refresh requests after every RPKI data refresh.
- ~~RFC8097: BGP Prefix Origin Validation State Extended Community~~
  - ~~https://tools.ietf.org/html/rfc8097~~
  - Many older config examples for ROV recommended RFC8097, setting extended BGP communities on routes to indicate their validation status (valid, invalid, unknown). This is now strongly recommended against. It is better to just drop invalids and not set any communities.

Default Local-Pref, Communities, MED – Overview

At **ingress**, Depending upon the type of eBGP neighbor (customer, R&E peer, commodity peer, paid transit), different informational communities will be added and local-pref will be set to implement route preferences. For example,

- Local-pref 200: customer route
- Local-pref 150: R&E peer route
- Local-pref 100: commodity peer route
- Local-pref 50: paid commodity transit route

Many providers allow customers to set MEDs to allow a customer multihomed to the provider to influence where the provider send traffic to the customer. Most commercial peers refuse to accept MEDs from their peers, instead preferring closest exit routing (hot potato).

At **egress**, if MEDs are not required/desired, don't send them, especially if they are tied to IGP metrics. This will just generate unneeded BGP route UPDATES when the IGP topology changes.

Default Local-Pref, Communities, MED – Config Examples

The local-preference and communities will vary depending on the operator of the AS. Using community examples from the I2PX network.

```
community-set EBGP-CUST
  11164:7500
end-set
community-set EBGP-REPEER
  11164:7600
end-set
community-set EBGP-PEER
  11164:7890
end-set
community-set EBGP-TRANSIT
  11164:7900
end-set

community-set EBGP-REGION
# set for each router based on its geo location
  11164:1170
end-set
```

IOS-XR

```
route-policy EBGP-CUST-DEFAULTS-IN
  set community EBGP-I2PX-CUST additive
  set community EBGP-I2PX-REGION additive
  set local-preference 200
  # allow CUST to send MEDs
end-policy
!
route-policy EBGP-REPEER-DEFAULTS-IN
  set community EBGP-REPEER additive
  set community EBGP-I2PX-REGION additive
  set local-preference 150
  # allow REPEER to send MEDs
end-policy

route-policy EBGP-PEER-DEFAULTS-IN
  set community EBGP-PEER additive
  set community EBGP-I2PX-REGION additive
  set local-preference 100
  set med 0
end-policy
!
route-policy EBGP-TRANSIT-DEFAULTS-IN
  set community EBGP-TRANSIT additive
  set community EBGP-REGION additive
  set local-preference 50
  set med 0
end-policy
```

Default Local-Pref, Communities, MED – Config Examples

On egress, we default to clearing MEDs for all eBGP neighbor types. A single policy could have been used for all types, but I kept them separate for future flexibility.

If an eBGP neighbor needs to receive MEDs, they can be re-allowed later in the policy chain.

```
route-policy EBGP-CUST-DEFAULTS-OUT
  set med 0
end-policy

route-policy EBGP-REPEER-DEFAULTS-OUT
  set med 0
end-policy

route-policy EBGP-PEER-DEFAULTS-OUT
  set med 0
end-policy

route-policy EBGP-TRANSIT-DEFAULTS-OUT
  set med 0
end-policy
```

Ingress Local-pref overrides – Overview

Local-preference is the primary method for BGP routes to be ranked internally within a network. A typical service provider network will make use of local-preference to prefer customer routes over peer routes over paid transit routes. Within these general categories, providers might have additional levels to allow for primary and backup paths.

The BGP local-preference attribute is not transitive, it doesn't get passed across AS boundaries. In order for a customer multi-homed to a provider to rank primary and backup paths, some providers have BGP action community tags that allow customers limited control over the local-preference settings of the routes they advertise.

Ingress Local-pref overrides – Config Examples

```
route-policy EBGP-LPREF-OVERRIDES
  if community matches-any EBGP-LPREF-CUST-HIGH then
    set local-preference 220
  elseif community matches-any EBGP-LPREF-CUST-LOW then
    set local-preference 180
  elseif community matches-any EBGP-LPREF-BELOW-PEER then
    set local-preference 80
  elseif community matches-any EBGP-LPREF-BELOW-TRANSIT then
    set local-preference 40
  endif
end-policy
```

Ingress Local-pref overrides – References

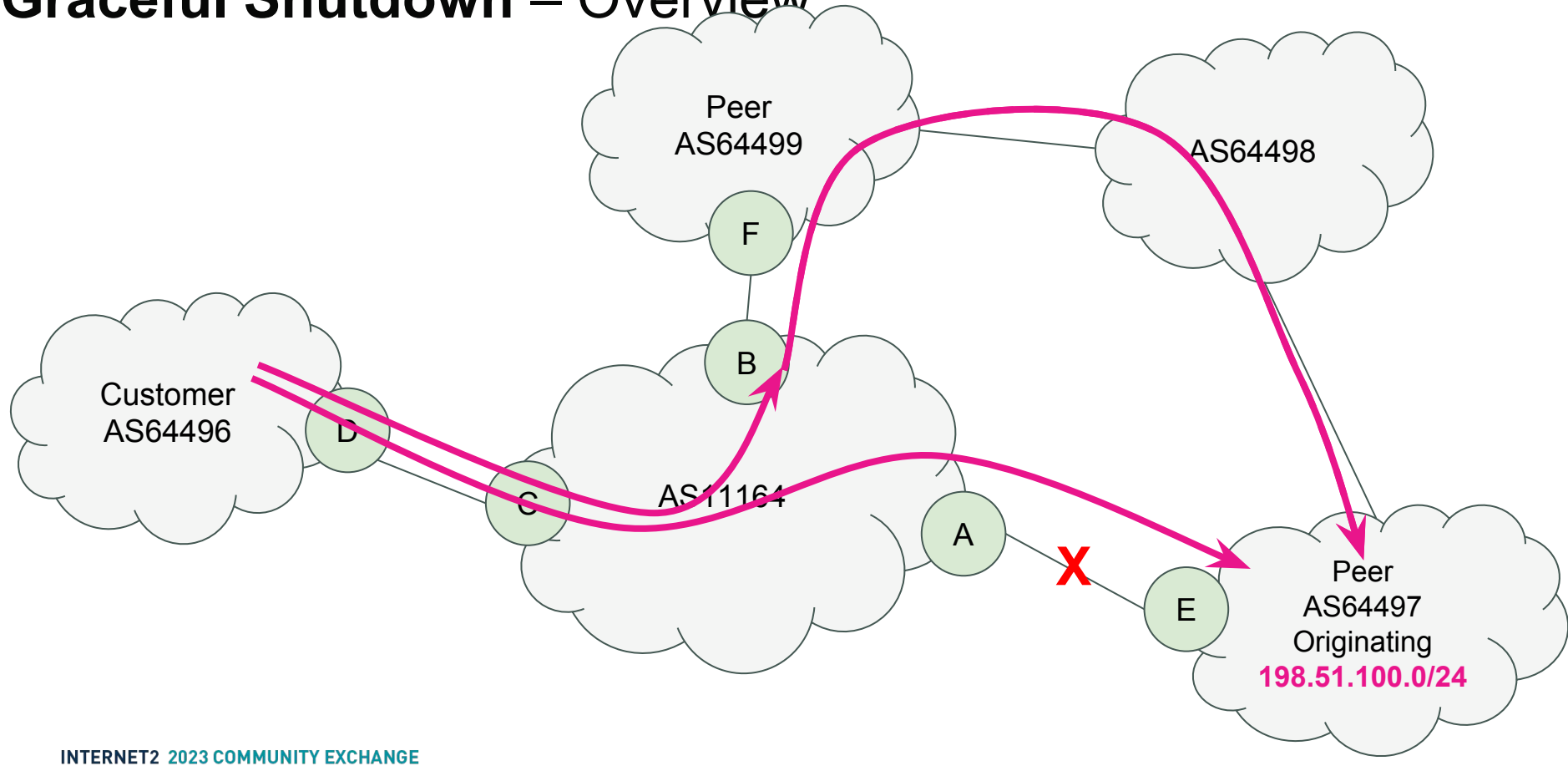
- RFC1998: An Application of the BGP Community Attribute in Multi-home Routing
  - https://tools.ietf.org/html/rfc1998

Graceful Shutdown – Overview

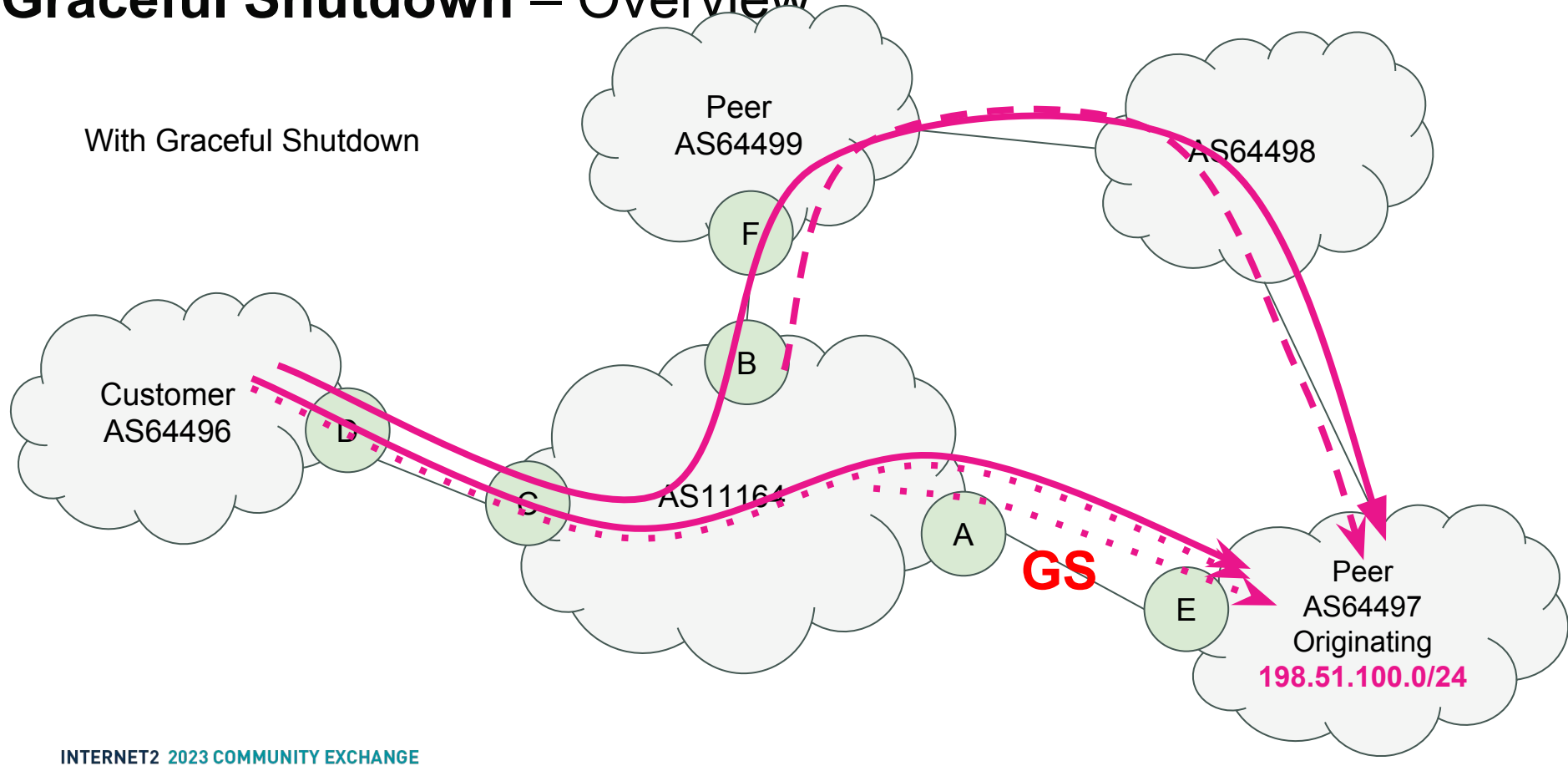
Network maintenance can be highly disruptive to BGP routing, resulting in packets being blackholed until BGP reconverges. It is a best practice to take action to get BGP to select alternative routes before the maintenance starts. Even shutting down BGP neighbors in advance of maintenance can result in a period of time when a router doesn't have any path to a destination, due to the BGP best path selection algorithm resulting in lower preference paths not being visible until a better path is fully withdrawn.

RFC6198 documented this issue and RFC8326 proposes a solution for networks to signal their eBGP neighbors of upcoming maintenance. RFC8326 defines a new well-known BGP community tag, GRACEFUL_SHUTDOWN. When a network receives this community attached to a route, it should take action to de-preference the route by setting local-preference to 0.

Graceful Shutdown – Overview



Graceful Shutdown – Overview



Graceful Shutdown – Config Examples

If an eBGP neighbor sends a route tagged with the well-known graceful-shutdown community to indicate that maintenance will soon be happening, a network should keep the route, but make it less preferable than any other routes that may exist to the prefix. This allows the other routers within the AS to gracefully select an alternate route before this route disappears.

```
route-policy EBGP-MAINT-IN
  if community matches-any (graceful-shutdown) then
    set local-preference 0
  endif
end-policy
```

IOS-XR

Graceful Shutdown – References

- RFC8326: Graceful BGP Session Shutdown
  - https://tools.ietf.org/html/rfc8326
- RFC6198: Requirements for the Graceful Shutdown of BGP Sessions
  - https://tools.ietf.org/html/rfc6198
- NLNOG BGP Filter Guide for Graceful Shutdown
  - https://bgpfilterguide.nlnog.net/guides/graceful_shutdown/
- Avoiding disruptions during maintenance operations on BGP sessions
  - https://inl.info.ucl.ac.be/system/files/ucl-ft-bgp-shutdown-inl.pdf

Assembling The Components Into Complete Policies

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

Assembling The Components – Config Example

An example of a customer eBGP ingress policy. Every customer needs a unique policy. In this case, the policy is for customer XYZ.

```
route-policy EBGP-CUST-XYZ-IN

  if apply EBGP-CUST-XYZ-SELECT-IN then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-ORIGINATED
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES

  apply EBGP-REJECT-PEERLOCKLITE

# continued to the right --->
```

IOS-XR

```
  apply EBGP-COMMSCRUB-CUST

  if apply EBGP-RTBH then
    done
  endif

  apply EBGP-REJECT-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-CUST-DEFAULTS-IN
  apply EBGP-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN

end-policy
```

Useful Router Commands

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

Useful Tools For Troubleshooting

INTERNET2
2023
COMMUNITY exchange

May 8-11, 2023 Atlanta, GA

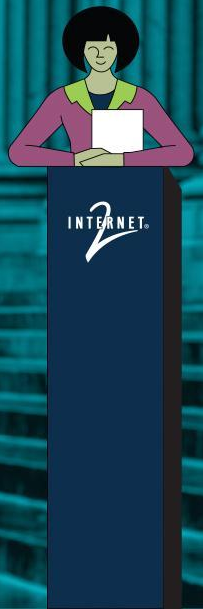
Routing Policy Problems

Mistakes are a great way to learn

INTERNET2
2023
COMMUNITY
exchange

May 8-11, 2023 Atlanta, GA

INTERNET2
2023
COMMUNITY
exchange
May 8-11, 2023 Atlanta, GA



Questions?

Jeff Bartig
Senior Interconnection Architect, Internet2
jbartig@internet2.edu