INTERNET2

2022 TECHNOLOGY exchange

Route-Policy in the NGI Network:
The Journey and the Result

Jeff Bartig
Interconnection Architect, Internet2

James Harr
Sr NetDevOps Engineer, Internet2

INTERNET2

Overview

- IOS-XR RPL Techniques
- Anatomy of an eBGP Policy
- BGP Maintenance
- Modeling in NSO
- Future work
- Q&A

IOS-XR RPL Techniques

What's in the toolbox

Crash course in RPL - Attaching a policy

```
router bgp 1234
 address-family ipv6 unicast
  network ... 2001:468::/48 route-policy EX1
  redistribute connected route-policy EX2

  neighbor 2001:db8::1
   route-policy EXAMPLE1 in
   route-policy EXAMPLE2 out

vrf FOO
 address-family ipv6 unicast
  import route-policy EXAMPLE1
  export route-policy EXAMPLE2
```

- Single-policy at attachment point
- Attach a policy at:
  - Route origination/redist/
  - BGP neighbor
  - VRF import/export

Crash course in RPL - Conditions and Actions

```
route-policy EXAMPLE1
 if destination or-longer (10.0.0.0/8) then
  set local-preference 200
  pass
 endif
end-policy


route-policy EXAMPLE2
 if community matches-any (11537:100) then
  drop
 endif
 set community (11537:200) additive
end-policy
```

Code structure

- Policies read like code/macro
- If statements can be nested

Do I accept this route?

- default - reject
- **drop** - reject, stop processing
- **done** - accept, stop processing
- **pass** - accept, continue processing
- **set ...** - set attribute
  - accept
  - continue processing

Crash course in RPL - Calling other policies

```
route-policy EXAMPLE1
 if ... then
  pass
 else
  drop
 endif
end-policy

route-policy EXAMPLE2
 apply EXAMPLE1
 set community (11537:100) additive
end-policy

route-policy EXAMPLE3
 if apply EXAMPLE1
  set local-preference 300
 endif
 pass
end-policy
```

Code structure

- Policies read like code
- If statements can be nested

Do I accept this route?

- default - reject
- **drop** - reject, stop processing
- **done** - accept, stop processing
- **pass** - accept, continue processing
- **set ...** - set attribute
  - accept
  - continue processing

Crash course in RPL - Passing Parameters

```
route-policy EXAMPLE1($var)
 set local-preference $var
 set community $var additive
 set community (11537:$var) additive
 if destination in $var ...
 if destination or-longer $var ...
 apply $var
 apply FOO($var)
 if apply $var ...
 if apply FOO($var) ...
end-policy

route-policy EXAMPLE2
 apply EXAMPLE1(123)
 apply EXAMPLE1(MY-PREFIX-SET)
 if apply EXAMPLE1(MY-COMMUNITY-SET)
end-policy
```

Parameters can be passed

- At attachment point
- When calling via **`apply`**

What can be passed

- Basic integers
- Names of sets (prefix, community, asn)
- Names of other policies

Crash course in RPL - Implicit parameters

```
route-policy EXAMPLE1
 set community (11537:peeras) ...
 set community (peeras:11537) ...

 if community matches-any (11537:peeras) ...
 if community matches-any (11537:not-peeras)
    ...

end-policy
```

Implicit parameters available in BGP:

- **peeras**
- **not-peeras**

Limitations:

- 32 bit ASNs with BGP communities
- Only on neighbors

Crash course in RPL - Global Parameters

```
policy-global
 NODE_REGION '100'
end-global


route-policy EXAMPLE1
 set community (11537:$NODE_REGION) ...
end-policy
```

Global parameters are available everywhere.

Useful for

- node maintenance state
- node region community

Crash course in RPL - Global Variables

```
route-policy EXAMPLE1
 var globalVar1 100
end-policy

route-policy EXAMPLE2
 if globalVar1 ge 100
  do something
 endif
end-policy

route-policy EXAMPLE3
 apply EXAMPLE1
 apply EXAMPLE2
end-policy
```

Global is a bit of a misnomer

- "Global" is relative...
- Think a CPU register
- Not shared across attachments
- Not shared across prefixes

Pitfalls

- You only get 5
- They're numbered not named

Useful for:

- Calculations - max(a,b,c)

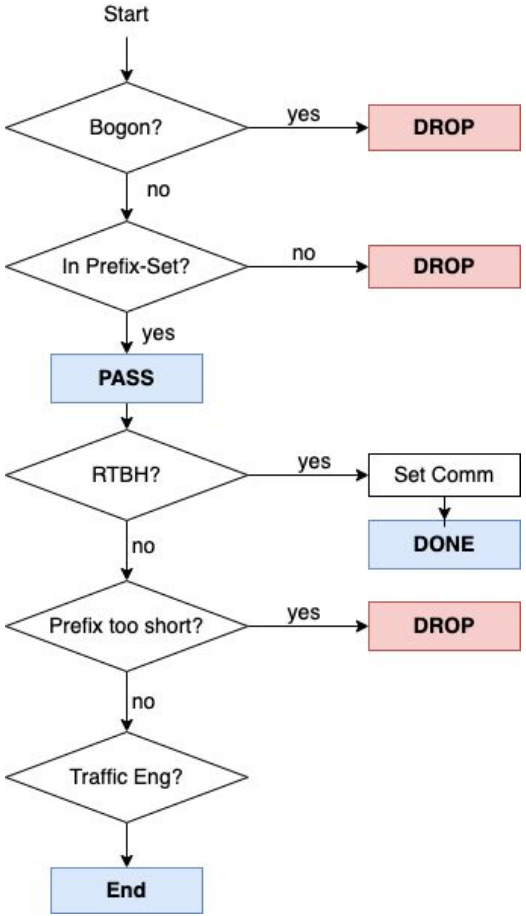
Structuring a policy

Things can get complicated

- Minimum prefix-lengths with RTBH routes
- Private ASNs
- RPKI validation in a community that wants /25 and /49 routes

Things that helped:

- Find examples
- Draw it out a flow-chart
  - Differentiate decisions
  - Highlight actions
- Choose one mentality when writing RPL
  - Explicit drop. AKA permit-unless-dropped



Debugging IOS-XR RPL

```
Expand apply statements, lists into a flat policy
# show rpl route-policy EXAMPLE inline

Display XML structure (pipe through `xmlstarlet format` to view)
# show rpl route-policy EXAMPLE pxl

Look at PCL structure
# debug pcl profile detail
# show pcl protocol bgp speaker-0 neighbor-in-vrf I2PX-IPv4-Uni-10.200.1.1 policy linked
# show pcl protocol bgp speaker-0 neighbor-in-dflt default-VPNv6-Uni-... policy linked

With Profiling information
# show pcl protocol bgp speaker-0 neighbor-in-vrf I2PX-IPv4-Uni-10.200.1.1 policy profile
```

References:

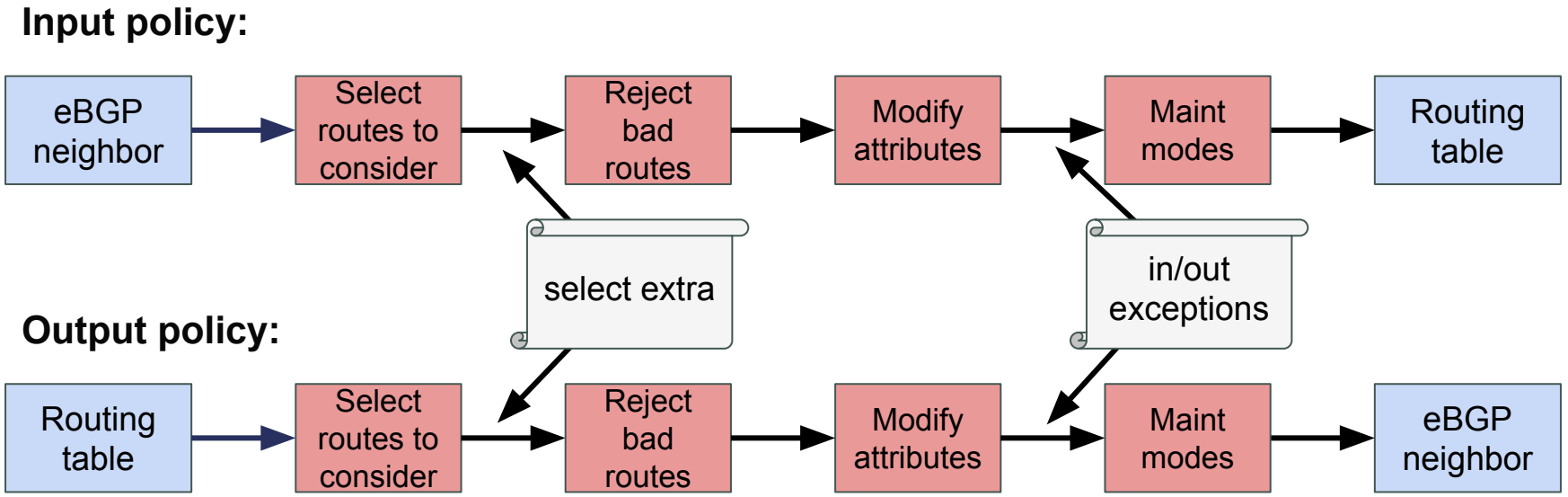
- Troubleshoot Slow BGP Convergence Due to Suboptimal Route Policies on IOS-XR: https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/217634-troubleshoot-slow-bgp-convergence-due-to.html

Anatomy of an eBGP Policy

NGI eBGP Route Policy

- **Challenges**
  - Migrating from Junos policy-statements to IOS-XR route-policy language (RPL)
  - 20+ years of manually maintained router configuration configuration
  - Many neighbor policy customizations, some still needed, some now cruft
  - Hundreds of neighbor ASNs
  - 1000+ eBGP neighbors
- **Goals**
  - Automation
  - Modernize route policy best practices
  - Improve readability of policies
  - Standardize/template config as much as possible
  - Consolidate unique neighbor config into as few places as possible
  - Still allow for customization (exceptions)

NGI eBGP Policy Overview



I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

$nei_maint

- 0 = in-service
- 1 = pre-maintenance
- 2 = maintenance

$ebgp_select_in

NSO dynamically created per-neighbor policy that selects routes to consider for accepting

$ebgp_exception_in

NSO dynamically created per-neighbor policy that applies a list of requested exceptions to the default policy

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

Using a per-neighbor dynamically created route policy, **select** the routes that will be considered in the rest of the policy. For CUST-IN policy, this is based on prefix-sets of the prefixes a participant is allowed to send. An example is below.

```
route-policy EBGP-AUTOGEN-I2PXCUST-POP-PEER-1-SELECT-IN
  if destination or-longer LEGACY-RIF-1234-CUST-V4-IN then
    pass
    done
  endif
  if destination or-longer LEGACY-RIF-1234-CUST-V6-IN then
    pass
    done
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

The select policy allows longer matches to be selected at this point. Later in the policy this is addressed after RTBH processing.

```
route-policy EBGP-AUTOGEN-I2PXCUST-POP-PEER-1-SELECT-IN
  if destination or-longer LEGACY-RIF-1234-CUST-V4-IN then
    pass
    done
  endif
  if destination or-longer LEGACY-RIF-1234-CUST-V6-IN then
    pass
    done
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

Next, any routes for undesirable prefixes and ASNs are dropped. While these shouldn't be in the prefix-sets that were just selected, this adds an extra layer of protection.

- Bogon prefixes and ASNs are dropped
- 0.0.0.0/0 or ::/0 are dropped
- Prefixes for IXPs are dropped
- Internet2 prefixes from the outside are dropped

```
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
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
route-policy EBGP-I2PX-CUST-IN($nei_maint,
    $ebgp_select_in, $ebgp_exception_in)

  if apply $ebgp_select_in then
    pass
  else
    drop
  endif

  apply EBGP-REJECT-BOGON-PREFIXES
  apply EBGP-REJECT-BOGON-ASN
  apply EBGP-REJECT-DEFAULT
  apply EBGP-REJECT-IXP-PREFIXES
  apply EBGP-REJECT-I2PX-ORIGINATED

  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
```

Peerlock lite help protect against accidental transit route leaks. Any routes that have a tier 1 ISP in their AS path are dropped.

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

as-path-set EBGP-TIER1-ASNS
  passes-through '174',
  passes-through '209',
  passes-through '286',
  passes-through '701',
  passes-through '1239',
  passes-through '1299',
  … snip
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

It is helpful have an organization plan for your BGP communities that separate informational and action communities. **Informational** communities are set internally to add metadata about a route, for example where it was learned and the type of peer (transit, settlement-free, customer). **Action** communities are generally set and sent by eBGP neighbors to cause policy actions such as controlling local-pref or AS path prepending.

Example I2PX/AS11164 Action communities:

- 11164:51240 Set local-pref high (240) to indicate a preferred route
- 11164:51200 Set local-pref low (200) to indicate a backup route
- 11164:52001 Prepend once to peers
- 11164:52002 Prepend twice to peers
- 11164:52003 Prepend thrice to peers

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

While Internet2 generally leaves BGP communities on routes learned from eBGP neighbors, a network shouldn't allow external networks to set internal informational communities. These are scrubbed from eBGP learned routes.

```
route-policy EBGP-I2PX-COMMSCRUB-CUST
  # Info communities start with 1, 7, 8, or 9.
  delete 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
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

```
community-set EBGP-I2PX-RTBH
  11164:53666,
  65535:666
end-set

route-policy EBGP-I2PX-RTBH
  if community matches-any EBGP-I2PX-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
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

```
community-set EBGP-I2PX-RTBH
  11164:53666,
  65535:666
end-set

route-policy EBGP-I2PX-RTBH
  if community matches-any EBGP-I2PX-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
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

I2PX allows participants to advertise more specific prefixes up to a v4 /24 and a v6 /48.

```
route-policy EBGP-REJECT-I2PX-CUST-LONGPREFIXES
  if destination in (0.0.0.0/0 ge 25, ::/0 ge 49) then
    drop
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

RPKI ROV is currently commented out of the policy, but coming soon in 1Q2023!!

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

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

Set informational BGP communities and the default participant local-pref of 220.

```
community-set EBGP-I2PX-CUST
  11164:7500
end-set

community-set EBGP-I2PX-REGION
  11164:1170
end-set

route-policy EBGP-I2PX-CUST-DEFAULTS
  set community EBGP-I2PX-CUST additive
  set community EBGP-I2PX-REGION additive
  set local-preference 220
end-policy
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

Process action communities to allow participant to control local-pref. For example, setting 11164:51240 will set local-pref to 240, higher than default 220.

```
route-policy EBGP-I2PX-LPREF-OVERRIDES
  if community matches-any EBGP-I2PX-LPREF-CUST-HIGH then
    set local-preference 240
  elseif community matches-any EBGP-I2PX-LPREF-CUST-LOW then
    set local-preference 200
  elseif community matches-any EBGP-I2PX-LPREF-BELOW-PEER then
    set local-preference 80
  elseif community matches-any EBGP-I2PX-LPREF-BELOW-TRANSIT
then
    set local-preference 50
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

Process NSO dynamically created policy that applies a list of requested exceptions to the default policy. Covered in detail in later slides

I2PX Route Policy - EBGP-I2PX-CUST-IN

```
  apply EBGP-REJECT-PEERLOCKLITE

  apply EBGP-I2PX-COMMSCRUB-CUST

  if apply EBGP-I2PX-RTBH then
    done
  endif

  apply EBGP-REJECT-I2PX-CUST-LONGPREFIXES

  #apply EBGP-REJECT-ROV-INVALID

  apply EBGP-I2PX-CUST-DEFAULTS
  apply EBGP-I2PX-LPREF-OVERRIDES

  apply $ebgp_exception_in

  apply EBGP-MAINT-IN($nei_maint)
end-policy
```

$nei_maint

- 0 = in-service
- 1 = pre-maintenance
- 2 = maintenance

Covered in detail in the next section.

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

OUT policies are much shorter, since most of the hard work is done via the IN policies.

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

During testing, we found that we needed different PEER-OUT policies for 16 and 32 bit peer ASNs because of how we implement per-peer traffic engineering policy.

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

$nei_maint

- 0 = in-service
- 1 = pre-maintenance
- 2 = maintenance

$ebgp_select_out

NSO dynamically created per-neighbor policy that selects additional routes to consider for advertising out beyond the default selected by EBGP-I2PX-PEER-SELECT-OUT. Currently not implemented.

$ebgp_exception_out

NSO dynamically created per-neighbor policy that applies a list of requested exceptions to the default policy

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

Select routes learned from participants and any internal routes to consider advertising to peers.

```
community-set EBGP-I2PX-CUST
  11164:7500
end-set

community-set EBGP-I2PX-STATIC-BB-INT
  11164:9020
end-set

route-policy EBGP-I2PX-PEER-SELECT-OUT
  if community matches-any EBGP-I2PX-CUST or community
matches-any EBGP-I2PX-STATIC-BB-INT then
    pass
    done
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

Set any default attributes for routes advertised to peers. Currently, that is just resetting the MED value to zero, since peers prefer hot potato routing.

```
route-policy EBGP-I2PX-PEER-DEFAULTS-OUT
  set med 0
end-policy
```

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

I2PX allows participants to set traffic engineering BGP communities to control per-peer how each route is advertised to peers.

65000:<ASN> Do not advertise at all to the peer ASN
65001:<ASN> Prepend once toward peer ASN
65002:<ASN> Prepend twice toward peer ASN
65003:<ASN> Prepend thrice toward peer ASN

65009:<ASN> Allow advertisement toward peer ASN if otherwise excluded by more general community control such as 11164:52000

If you want to prepend your route 3 times when I2PX advertises to Google, you could tag with **65003:15169**.

```
route-policy EBGP-I2PX-PEER-16ASN-TE-COMMS
  if community matches-any (65003:peeras) or large-community matches-any (11164:52003:peeras) then
    prepend as-path most-recent 3
  elseif community matches-any (65002:peeras) or large-community matches-any (11164:52002:peeras) then
    prepend as-path most-recent 2
  elseif community matches-any (65001:peeras) or large-community matches-any (11164:52001:peeras) then
    prepend as-path most-recent 1
    # only if no per-peer prepending has been requested, then apply "all" peer prepending
  elseif community matches-any EBGP-I2PX-CUST-PREP3-PEER-ALL or community matches-any EBGP-I2PX-CUST-PREP3-PEER-NA
then
    prepend as-path most-recent 3
  elseif community matches-any EBGP-I2PX-CUST-PREP2-PEER-ALL or community matches-any EBGP-I2PX-CUST-PREP2-PEER-NA
then
    prepend as-path most-recent 2
  elseif community matches-any EBGP-I2PX-CUST-PREP1-PEER-ALL or community matches-any EBGP-I2PX-CUST-PREP1-PEER-NA
then
    prepend as-path most-recent 1
  Endif

  # check to see if we should not be sending this route to this peer
  if not (community matches-any (65009:peeras) or large-community matches-any (11164:52009:peeras)) then
    # only drop the route if the customer hasn't specifically requested that we pass
    if community matches-any (65000:peeras) or large-community matches-any (11164:52000:peeras) then
      drop
    elseif community matches-any EBGP-I2PX-CUST-NOEXP-PEER-ALL or community matches-any EBGP-I2PX-CUST-NOEXP-PEER-NA
then
      drop
    endif
  endif
end-policy
```

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

Apply any per-neighbor exceptions.

I2PX Route Policy - EBGP-I2PX-PEER-OUT

```
route-policy EBGP-I2PX-PEER-16ASN-OUT(
    $nei_maint, $ebgp_select_out,
    $ebgp_exception_out)

  if apply EBGP-I2PX-PEER-SELECT-OUT then
    pass
  else
    drop
  endif

  apply EBGP-I2PX-PEER-DEFAULTS-OUT

  apply EBGP-I2PX-PEER-16ASN-TE-COMMS

  apply $ebgp_exception_out

  apply EBGP-MAINT-OUT($nei_maint)
end-policy
```

$nei_maint

- 0 = in-service
- 1 = pre-maintenance
- 2 = maintenance

Covered in detail in the next section.

BGP Maintenance

Making use of BGP graceful-shutdown

Risks of hard route withdrawal

Consider the following scenario:

- Peer link needs to be shutdown to perform maintenance
- eBGP session is shutdown -or- a drop-all policy is applied to an existing session

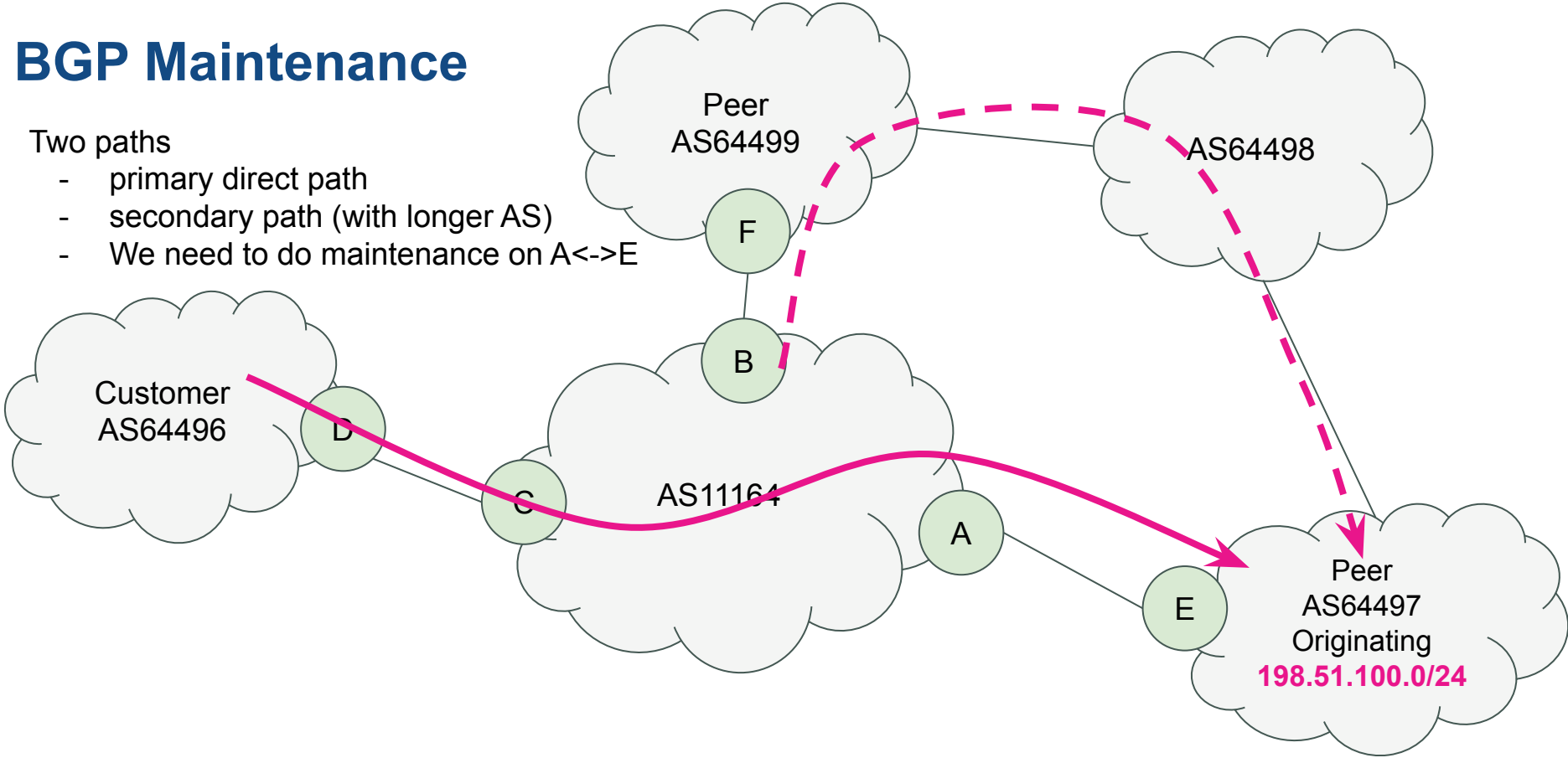
What happens:

- Immediate BGP neighbor drops route
- During convergence, there may not be a route to a given prefix while convergence happens

Let's look at an illustration

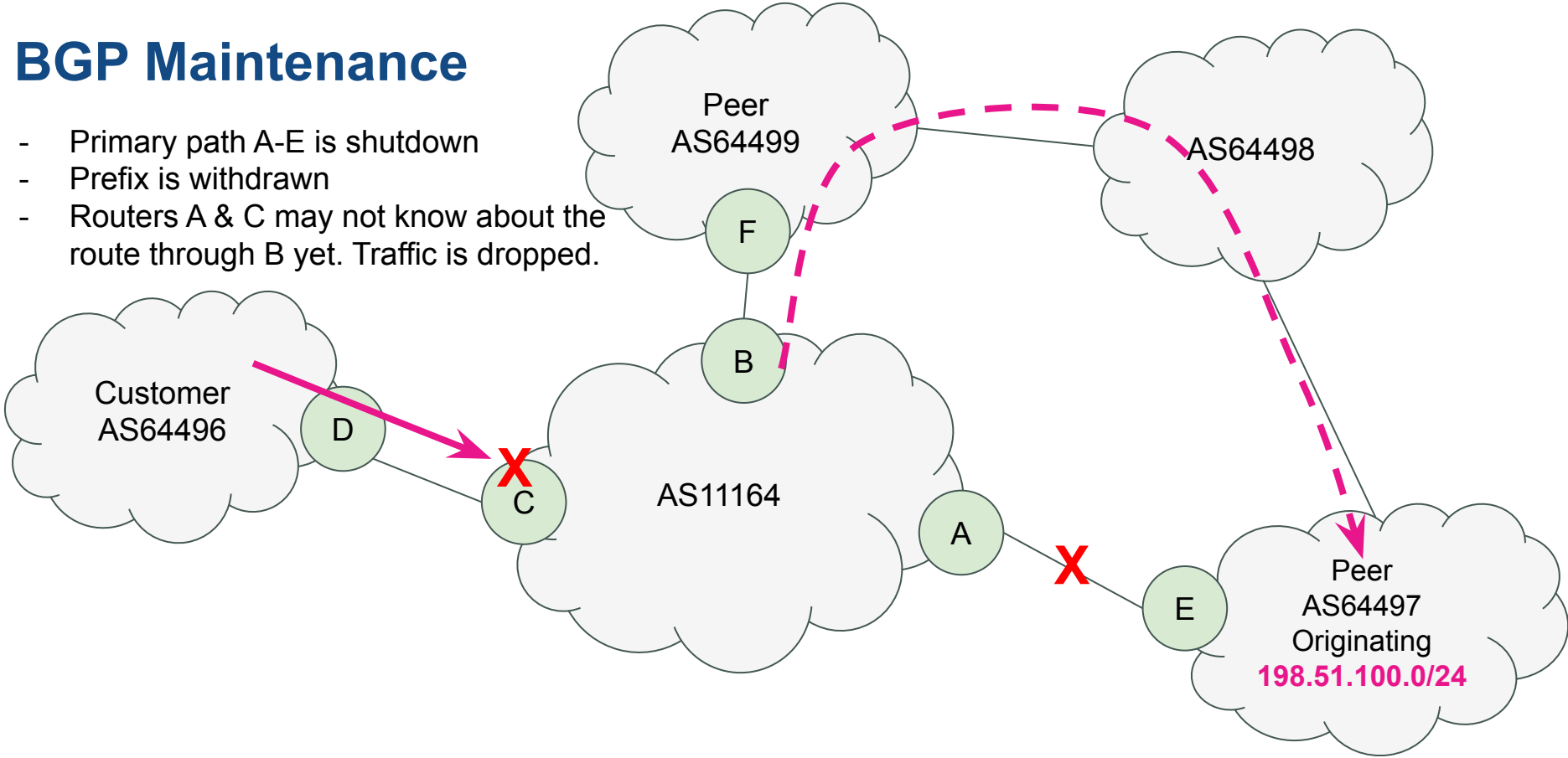
BGP Maintenance

Two paths
- primary direct path
- secondary path (with longer AS)
- We need to do maintenance on A<->E



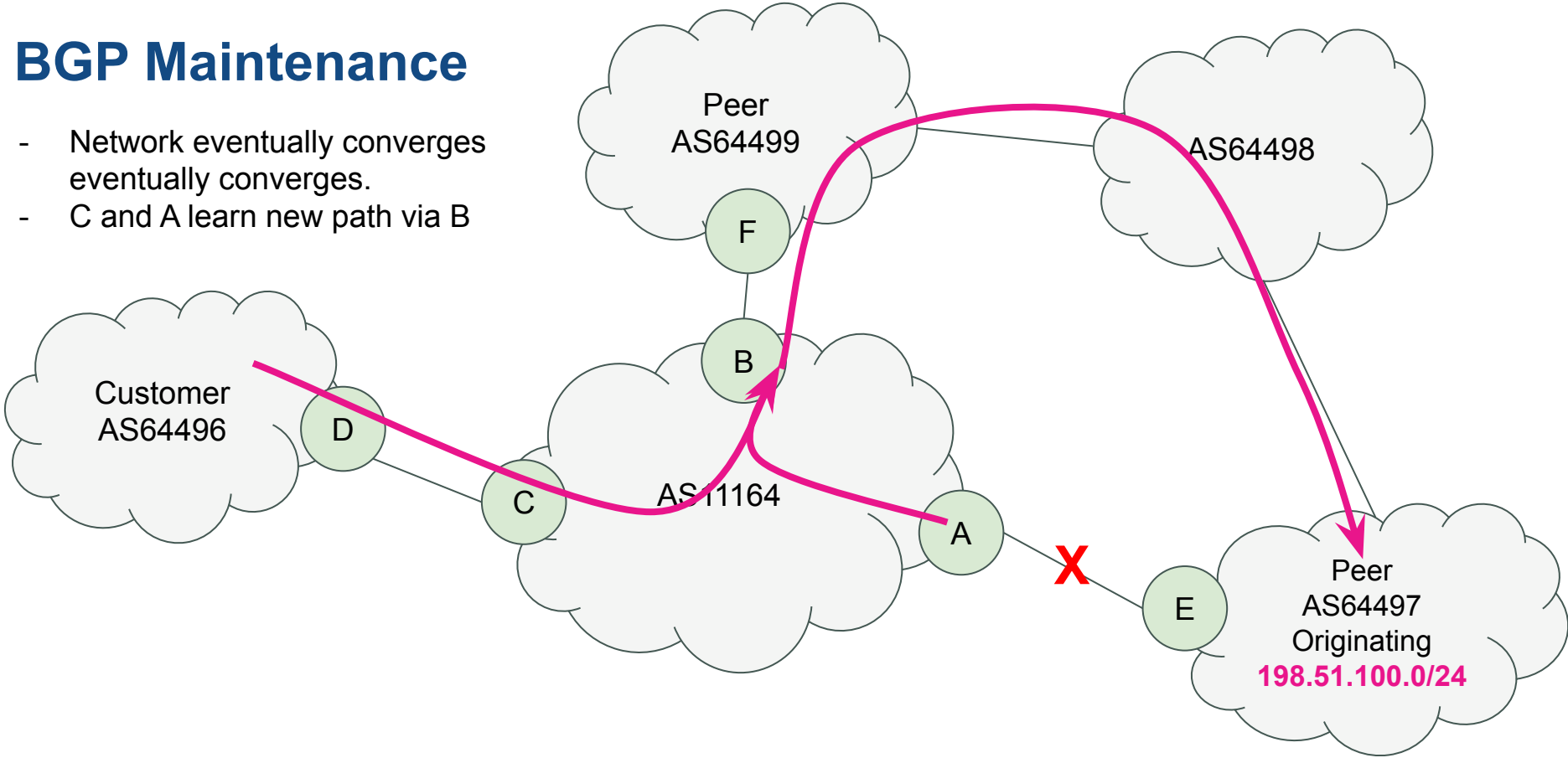
BGP Maintenance

- Primary path A-E is shutdown
- Prefix is withdrawn
- Routers A & C may not know about the route through B yet. Traffic is dropped.



BGP Maintenance

- Network eventually converges eventually converges.
- C and A learn new path via B



Maintenance with Graceful Shutdown

Consider the following scenario:

- Peer link needs to be shutdown to perform maintenance
- Prior to shutdown, route is spoiled

What happens:

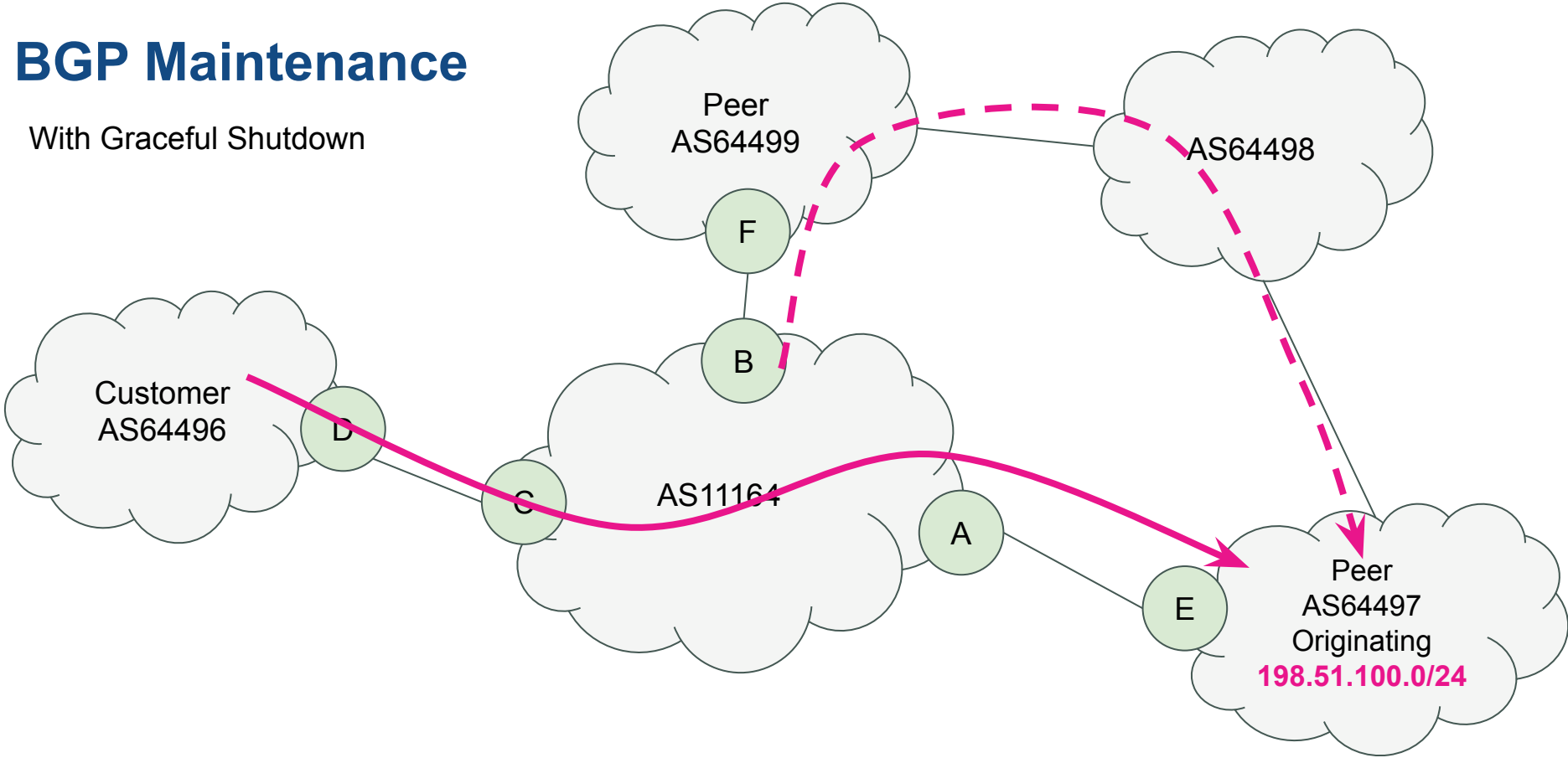
- BGP best-path picks up alternative route prior to
- Traffic is rerouted

Let's look at an illustration

- Spoil primary path first
- Reconvergence happens without dropping traffic

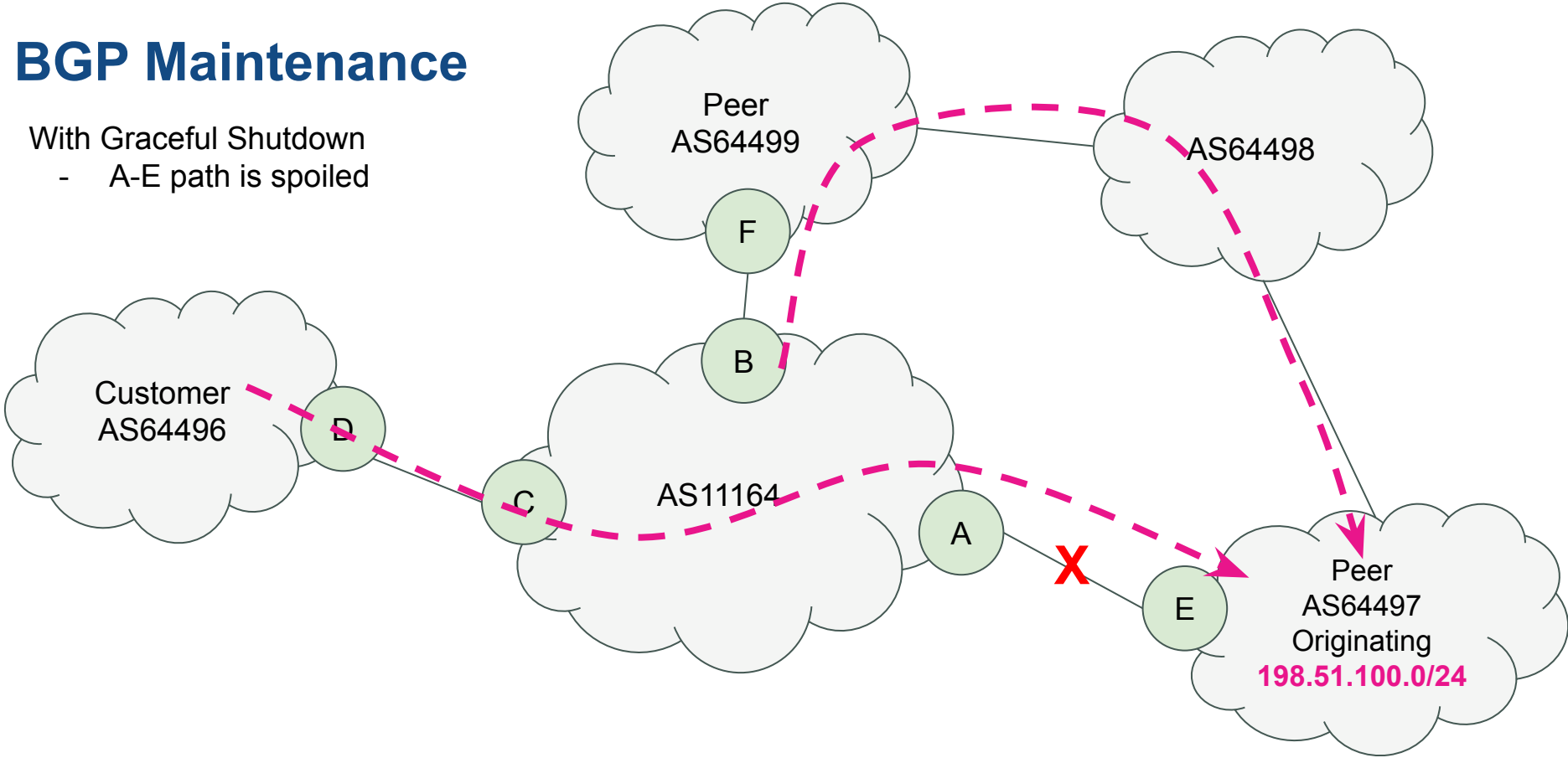
BGP Maintenance

With Graceful Shutdown



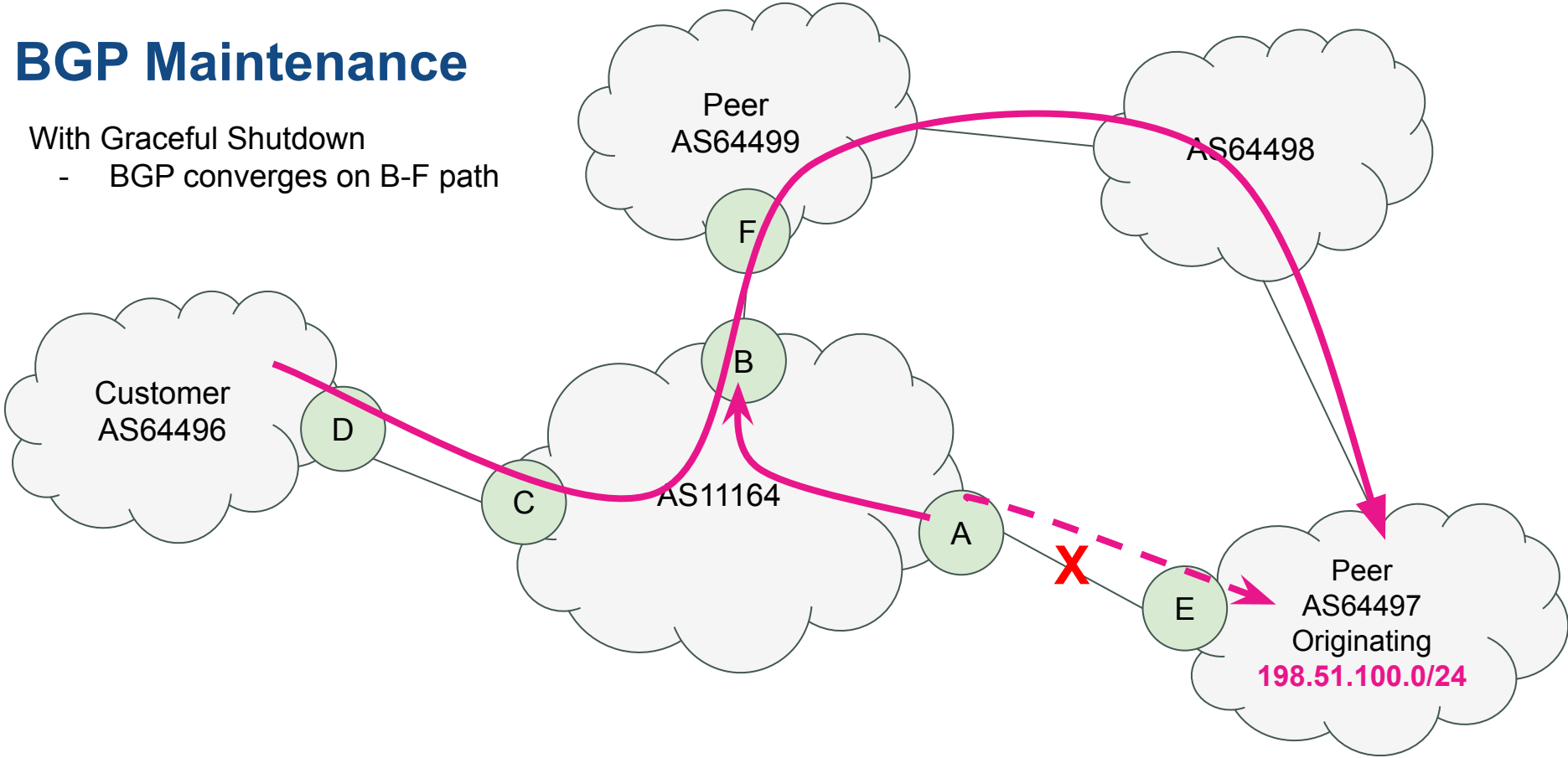
BGP Maintenance

With Graceful Shutdown
- A-E path is spoiled



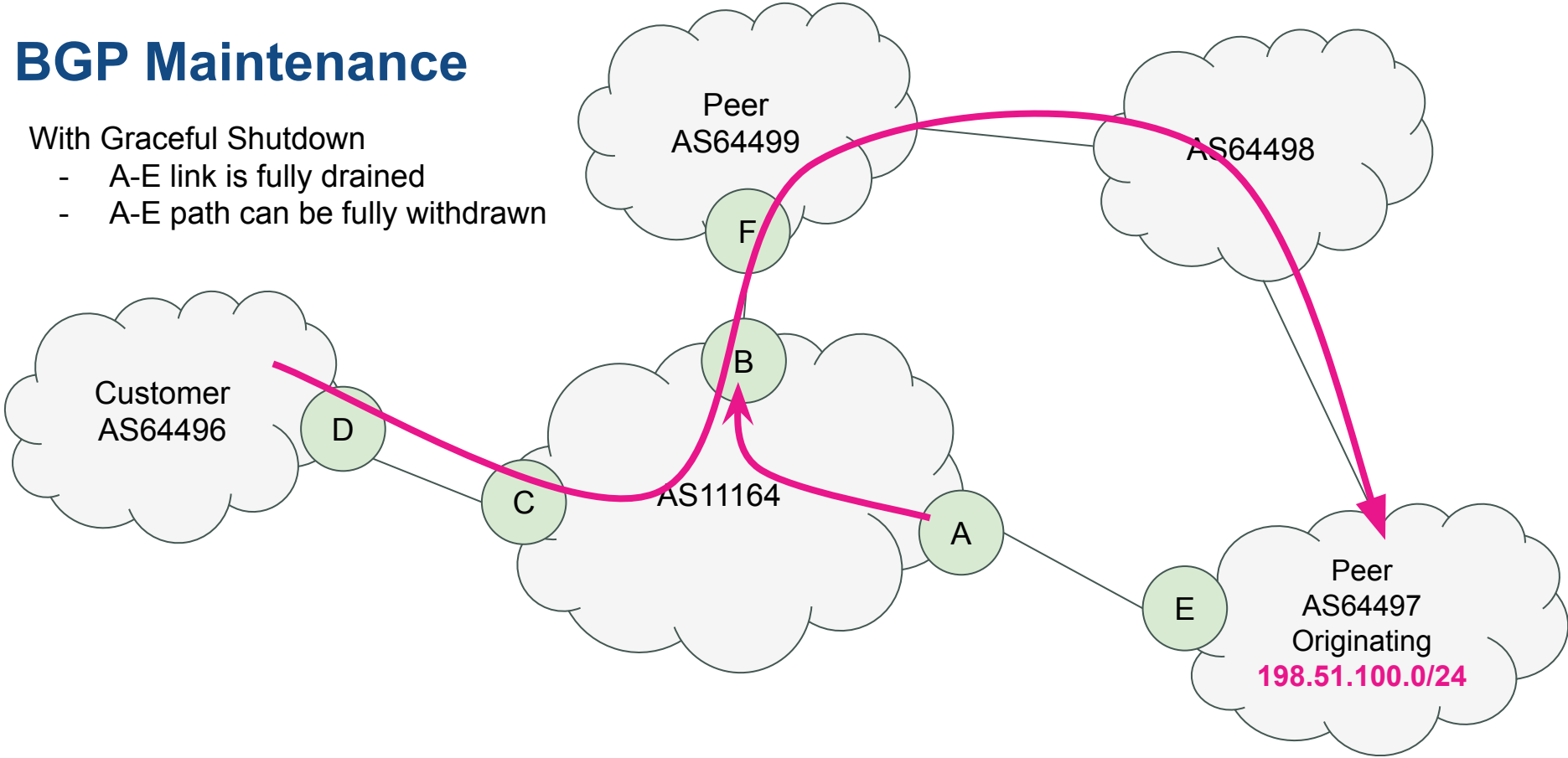
BGP Maintenance

With Graceful Shutdown
- BGP converges on B-F path



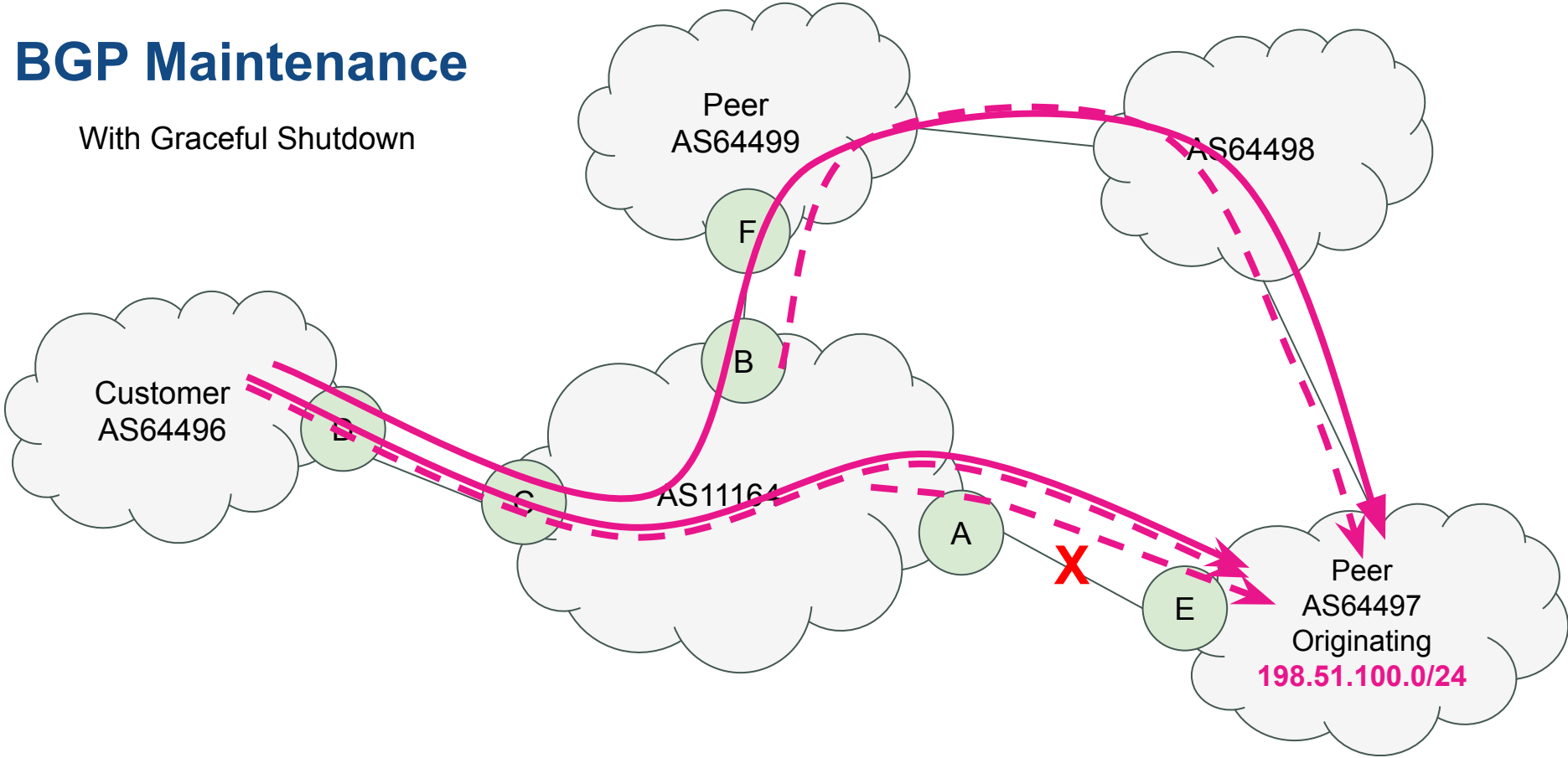
BGP Maintenance

With Graceful Shutdown
- A-E link is fully drained
- A-E path can be fully withdrawn



BGP Maintenance

With Graceful Shutdown



Maintenance with Graceful Shutdown

Graceful-Shutdown Community (RFC 8326)

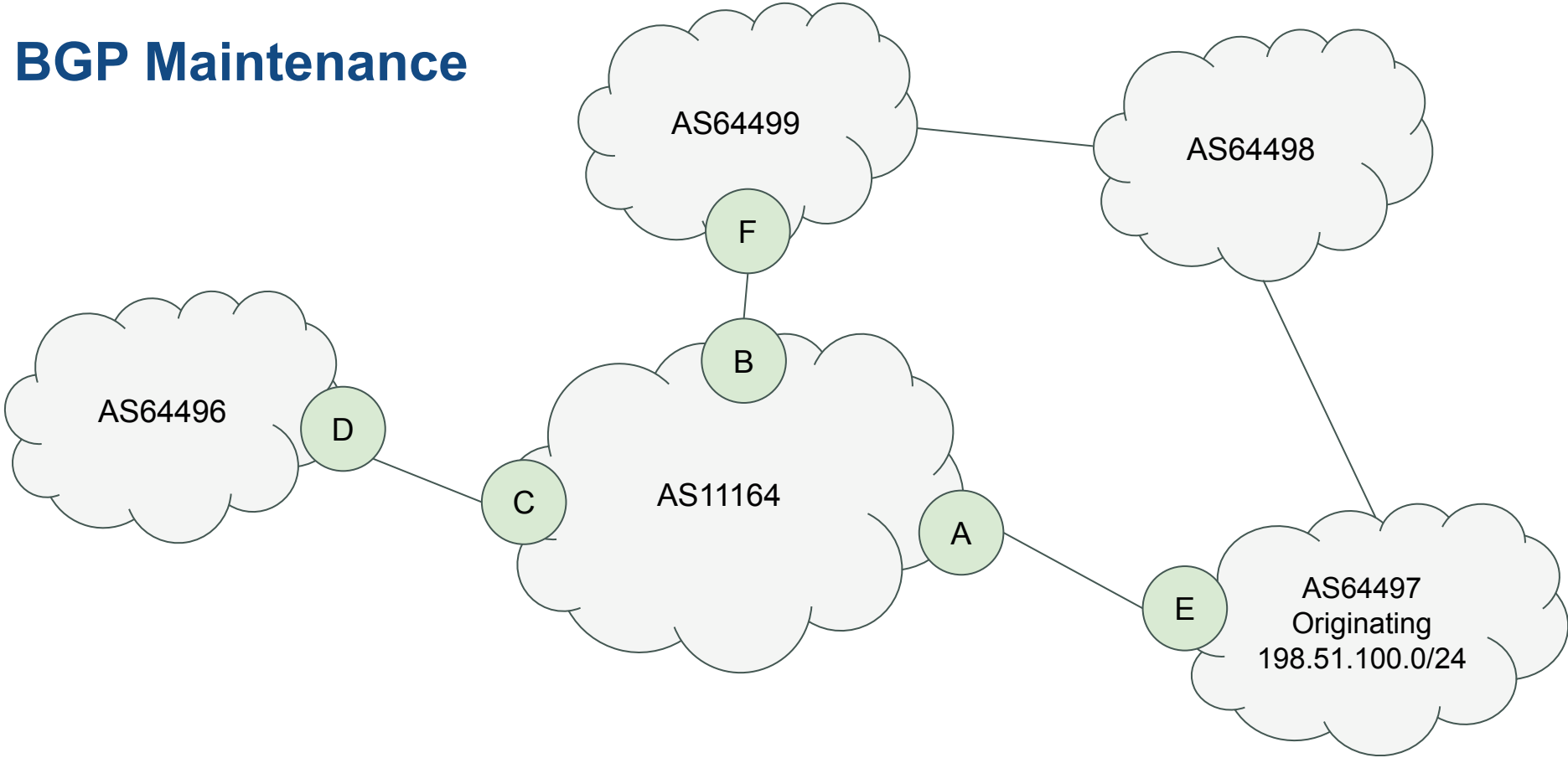
- Well known BGP community string
- Some BGP implementations can enable via CLI
- NOT integrated into Best-Path Algorithms (that we could find)
- Can't rely on community alone

Use a multi-prong approach:

- Set local-preference to 0
- Set med max-reachable
- Prepend ASN
- Still signal via BGP community

Hope one of the above work spoils the path that's going to undergo maintenance

BGP Maintenance



NGI Route Policy - EBGP-MAINT-OUT

```
policy-global
  NODE_MAINTENANCE '0'
end-global


route-policy EBGP-MAINT-OUT($nei_maint)
  var globalVar1 $NODE_MAINTENANCE
  if globalVar1 le $nei_maint then
    var globalVar1 $nei_maint
  endif
  if globalVar1 ge 2 then
    drop
  elseif globalVar1 eq 1 then
    prepend as-path own-as 3
    set med max-reachable
    set community (graceful-shutdown) additive
  endif
end-policy
```

NSO can set a maintenance state either for a specific neighbor or for all services on a specific router. Per-neighbor state is set by setting the $nei_maint value. Per-router state is set by modifying the NODE_MAINT global variable (really a constant).

Use the greater of the two values.

- 0 = in-service
- 1 = pre-maintenance
- 2 = maintenance

NGI Route Policy - EBGP-MAINT-OUT

```
policy-global
  NODE_MAINTENANCE '0'
end-global


route-policy EBGP-MAINT-OUT($nei_maint)
  var globalVar1 $NODE_MAINTENANCE
  if globalVar1 le $nei_maint then
    var globalVar1 $nei_maint
  endif
  if globalVar1 ge 2 then
    drop
  elseif globalVar1 eq 1 then
    prepend as-path own-as 3
    set med max-reachable
    set community (graceful-shutdown) additive
  endif
end-policy
```

- 0 = in-service

If the neighbor is in-service, then this policy is a no-op.

NGI Route Policy - EBGP-MAINT-OUT

```
policy-global
  NODE_MAINTENANCE '0'
end-global


route-policy EBGP-MAINT-OUT($nei_maint)
  var globalVar1 $NODE_MAINTENANCE
  if globalVar1 le $nei_maint then
    var globalVar1 $nei_maint
  endif
  if globalVar1 ge 2 then
    drop
  elseif globalVar1 eq 1 then
    prepend as-path own-as 3
    set med max-reachable
    set community (graceful-shutdown) additive
  endif
end-policy
```

- 1 = pre-maintenance

Still advertises routes, but encourage a different path to be selected.

In a pre-maintenance state, routes are still advertised to the eBGP neighbor, but I2PX sets the well-known GRACEFUL_SHUTDOWN (RFC8326) community. Since many networks haven't yet implemented RFC8326 recommendations, I2PX also makes the route less preferable by prepending and setting a high MED.

NGI Route Policy - EBGP-MAINT-OUT

```
policy-global
  NODE_MAINTENANCE '0'
end-global


route-policy EBGP-MAINT-OUT($nei_maint)
  var globalVar1 $NODE_MAINTENANCE
  if globalVar1 le $nei_maint then
    var globalVar1 $nei_maint
  endif
  if globalVar1 ge 2 then
    drop
  elseif globalVar1 eq 1 then
    prepend as-path own-as 3
    set med max-reachable
    set community (graceful-shutdown) additive
  endif
end-policy
```

- 2 = maintenance

Drop all routes.

In a maintenance state, no routes are sent to an eBGP neighbor.

NGI Route Policy - EBGP-MAINT-IN

```
route-policy EBGP-MAINT-IN($nei_maint)
  var globalVar1 $NODE_MAINTENANCE
  if globalVar1 le $nei_maint then
    var globalVar1 $nei_maint
  endif
  if globalVar1 ge 2 then
    drop
  elseif globalVar1 eq 1 then
    prepend as-path own-as 3
    set local-preference 0
    set community (graceful-shutdown) additive
  elseif community matches-any
              (graceful-shutdown) then
    prepend as-path most-recent 3
    set local-preference 0
  endif
end-policy
```

The EBG-MAINT-IN policy is quite similar to the -OUT policy, but also checks if the eBGP neighbor is sending the well-known GRACEFUL_SHUTDOWN community tag.

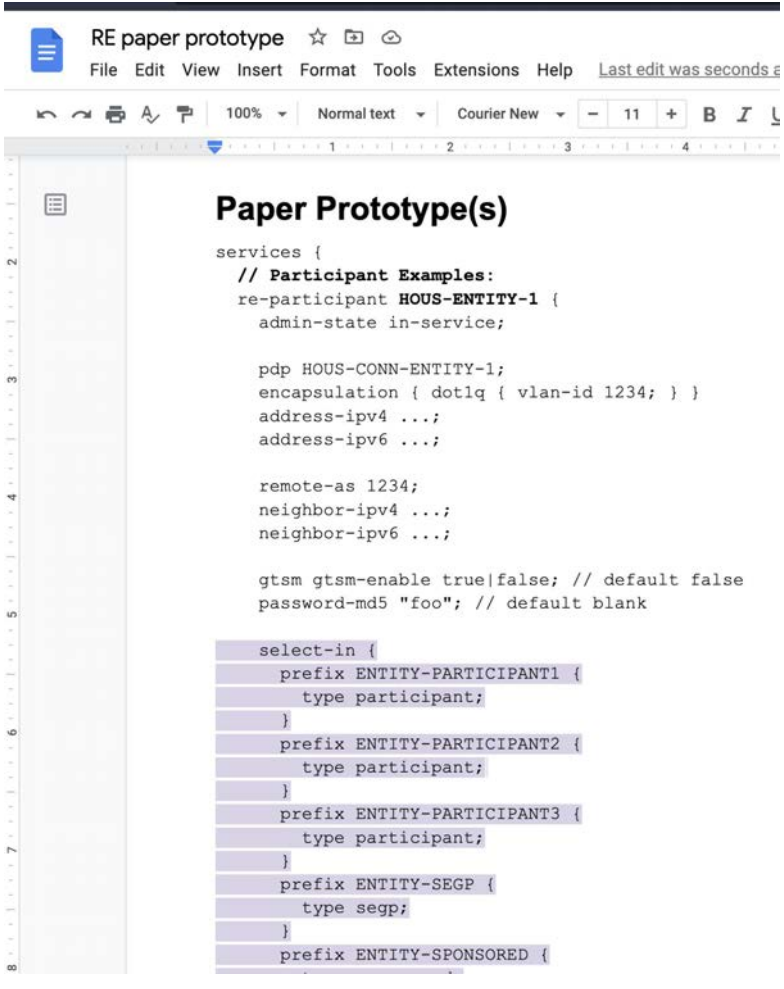
If the router or neighbor is in a pre-maintenance state or the GRACEFUL_SHUTDOWN community is received, then we follow the RFC8326 recommendation of setting local-pref to 0 to encourage an alternate path to be found.

Modeling in NSO

Make the easy way the right way

Modeling in the NSO

- Internal eBGP policy whitepaper
- All config for a service is in one place
  - Encapsulation, Address, Neighbor, Policy
- Design decisions
  - Common case = Simple way = Short way
  - Exceptions become very obvious
- Design mockups
  - Google Docs - easy to edit/comment/etc
- Start with use-cases
  - Let the patterns emerge, then refactor
  - Tests make changes easier (~219 test cases)
- Put it into practice



Modeling in the NSO

```
i2px-cust POP-PEER-1 {
    admin-state in-service;
    service-id  1234;
    pdp         POP-CONN-PORT-2;
    encapsulation {
        dot1q { vlan-id 123; }
    }
    address-ipv4 192.0.2.1/30;
    address-ipv6 2001:db8::1/126;
    remote-as    65001;
    neighbor     192.0.2.2;
    neighbor     2001:db8::2;
    password-md5 luggagecombo1234;

    ... BGP policy config?
}
```

What do we know already?

- This is an i2px-cust service

What do we need to fill in?

- Route selection
  - Incoming: Based on explicit prefix
  - ~~Outbound~~ standardized
- Exceptions
  - Modify attributes
  - Explicitly drop routes

Modeling in the NSO - Selecting routes

```
i2px-cust POP-PEER-1 {
    ... port + admin-state + metadata
    ... L2 encapsulation + L3 addressing
    ... BGP session config

    select-in {
        prefix LEGACY-RIF-1234-CUST-V4-IN;
        prefix LEGACY-RIF-1234-CUST-V6-IN;
    }

    select-out { ... }
}
```

What do we need to fill in?

- Route selection
  - Incoming: Based on explicit prefix
  - Outbound: has a set sent to everyone
- Exceptions
  - Modify attributes
  - Explicitly drop routes

Modeling in the NSO - Selecting Routes

```
# Generated by NSO for i2px-cust POP-PEER-1
route-policy EBGP-AUTOGEN-I2PXCUST-POP-PEER-1-SELECT-IN
  if destination or-longer LEGACY-RIF-1234-CUST-V4-IN then
    pass
    done
  endif
  if destination or-longer LEGACY-RIF-1234-CUST-V6-IN then
    pass
    done
  endif
end-policy

# Applied to neighbor by NSO
neighbor ...
  route-policy in EBGP-I2PX-CUST-IN(0, EBGP-AUTOGEN-I2PXCUST-POP-PEER-1-SELECT-IN, EBGP-EXCEPTION-NONE)

# Standard route-policy
route-policy EBGP-I2PX-CUST-IN($nei_maint, $ebgp_select_in, $ebgp_exception_in)
  if apply $ebgp_select_in then
    pass
```

Modeling in the NSO - Exceptions

```
i2px-cust POP-PEER-1 {
    ... port + admin-state + metadata
    ... L2 encapsulation + L3 addressing
    ... BGP session config

    select-in {
        prefix LEGACY-RIF-1234-CUST-V4-IN;
        prefix LEGACY-RIF-1234-CUST-V6-IN;
    }
    exception-in {
        drop {
            originates-from 65002;
            originates-from 65003;
        }
        prepend 65001 3 {
            passes-through 65004;
            passes-through 65005;
        }
    }
}
```

Exceptions:

- **Drop** or **modify** routes
- Cannot **accept** routes that the standard policy can't accept
- Stick out

Standardized on an **action + condition**:

- Often want to do the same thing to many conditions
- Same set of options for all BGP sessions

Modeling in the NSO - Exception

```
# Generated by NSO
route-policy EBGP-AUTOGEN-I2PX-CUST-POP-PEER-1-EXCEPTION-OUT {
  if as-path originates-from '65002'  then
    drop
  endif
  if as-path originates-from '65003'  then
    drop
  endif
  if as-path passes-through '65004'  then
    prepend as-path 65001 3
  endif
  if as-path passes-through '65005'  then
    prepend as-path 65001 3
  endif
end-policy

# Standard policy
route-policy EBGP-I2PX-CUST-IN($nei_maint, $ebgp_select_in, $ebgp_exception_in)
  ...
  apply $ebgp_exception_in
```

Modeling in the NSO - Exceptions

```
i2px-cust POP-PEER-1 {
  ... port + admin-state + metadata
  ... L2 encapsulation + L3 addressing
  ... BGP session config
  exception-in {
    drop {
      xr-condition "destination or-longer (::/0)";

      xr-condition "not (as-path neighbor-is
         '65001' or as-path neighbor-is '65002'
         or as-path neighbor-is '65003' or
         as-path neighbor-is '65004' or
         as-path neighbor-is '65005' )";
    }
  }
}
```

How do we do really hard things?

- Punt to the engineer and IOS-XR
- Platform-specific hack

What we use this for?

- Deal with a broken AFI
- Implement "I only want routes from these 5 ASNs"
- Compound conditionals

2nd level escape hatch

Future Work

Future work - James

- "The escape hatches"
  - exception-in/out
  - xr-condition...
    - "peer only wants to receive 5 ASNs"
- Future Work - NSO
  - neighbor-group
    - max-prefix (frequently updated)
    - route-policy
      - Peer only wants 4-5 ASNs for all their peers
      -
  - AFI-specific maintenance: xr-condition "destination or-longer (::/0)"

Future work - Jeff

- Flat policy per-neighbor
  - Still use `apply` statements to implement standard policy
  - Enable things like peerlock-lite - Filtering Tier1 ASNs from non-transit peers
    - Have to leave Zayo out of tier1-asn list...
- Automated policy testing
  - Allows us to codify our policies in tests and can spot unintended side-effects
  - The tests would be resource intensive, involve working with multiple systems, etc
  - Another place for inter-team collaboration

Q&A