Robust Routing Policy Architecture

Job Snijders

NTT Communications

job@ntt.net

LACNIC 30, Rosario, Argentina

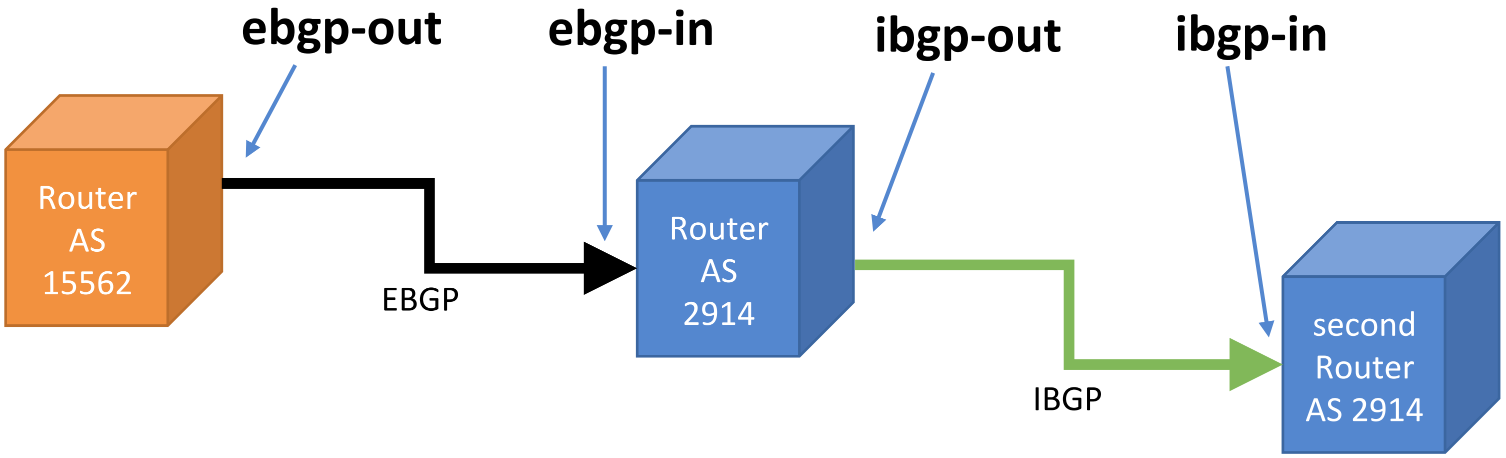
Robust Routing Policy Architecture

- Conceptual model of routing policy
- Routing policy terminology
- Routing policy design patterns
    - Maximum Prefix Limits
    - 2 Phase Pruning
    - Classification & Execution
    - Hints

Conceptual model & Terminology

- Attachment points
- Directionality

“One man’s ebgp-out is another man’s ebgp-in.”
– ancient Dutch proverb



Example

Attachment Point

```
router bgp 15562
  neighbor 192.147.168.1 route-map AS2914-in in
  neighbor 192.147.168.1 route-map AS2914-out out
!
route-map AS2914-in deny 10
  match ip address prefix-list bogons-v4
route-map AS2914-in permit 10
  match community graceful-shutdown
  set local-preference 0
!
```

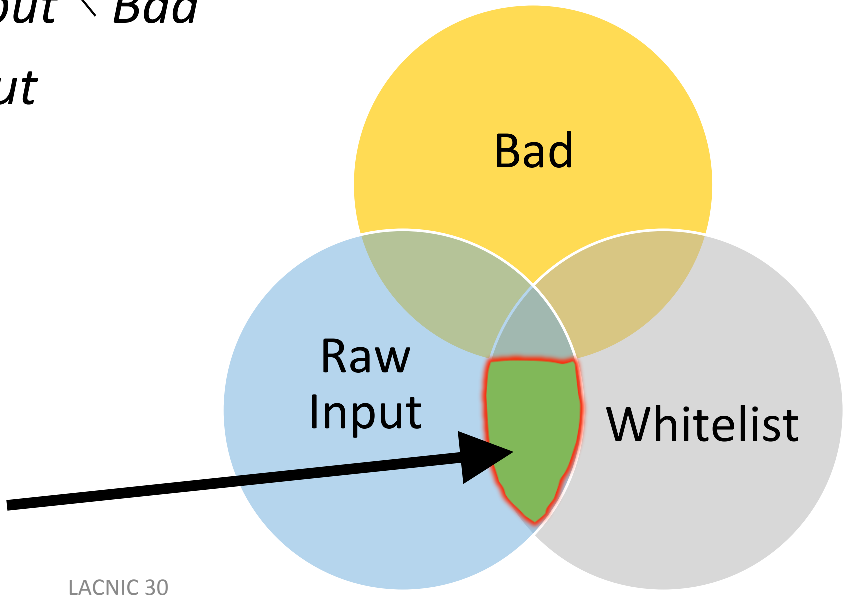
Direction

Policy

Term

ebgp-in Filtering – what to accept?

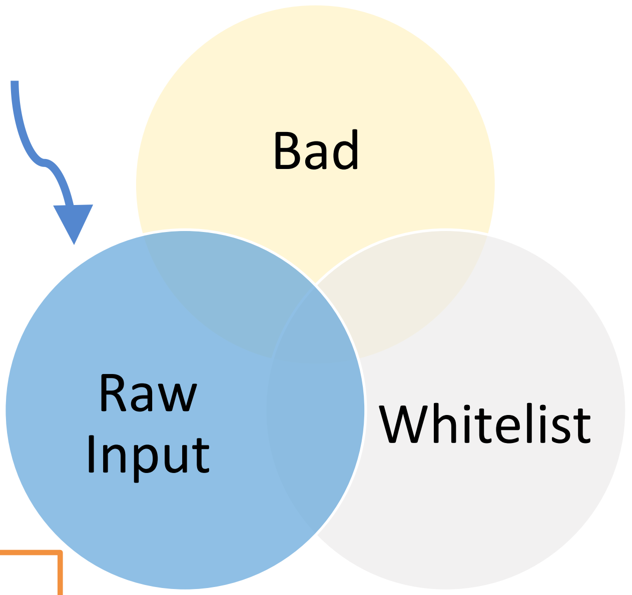
- Phase 1: Pruning: If *Bad* and *Raw Input* are sets, then the **relative complement** of *Bad* in *Raw Input*, is the set of elements in *Raw Input* but not in *Bad: Raw Input* ∖ *Bad*
- Phase 2: *Allowlist* ∩ *Raw Input*



The Good Stuff

Raw Input in context of **ebgp-in**

- *Raw Input* is whatever your EBGP neighbor announces to you
- *Raw Input* can contain anything, in any quantity
- In IETF speak: “**Adj-RIB-In**”
- This is where `maximum-prefix` limits must be applied!



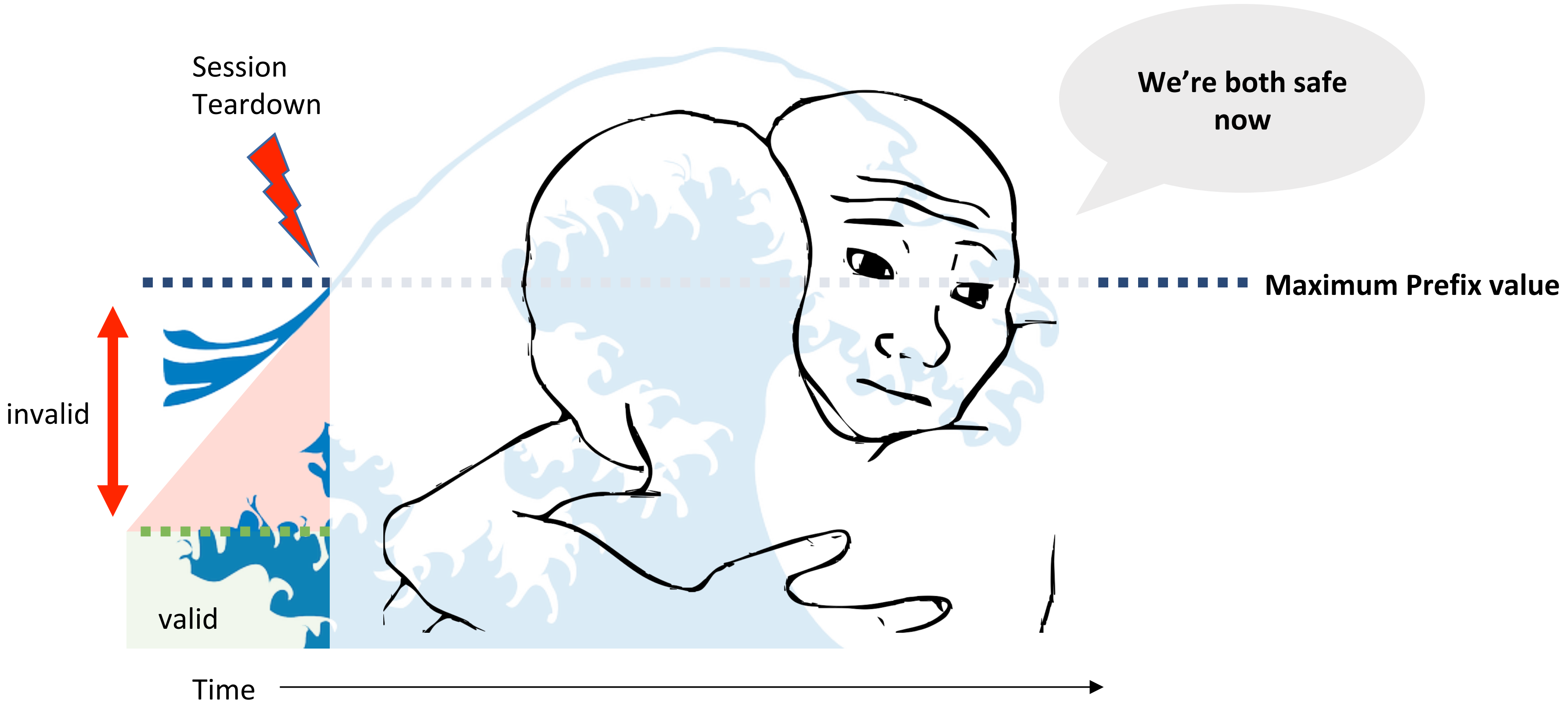
Study resource:
NLNOG Filter Guide: http://bgpfilterguide.nlnog.net/

Maximum prefix limits in **ebgp-in**

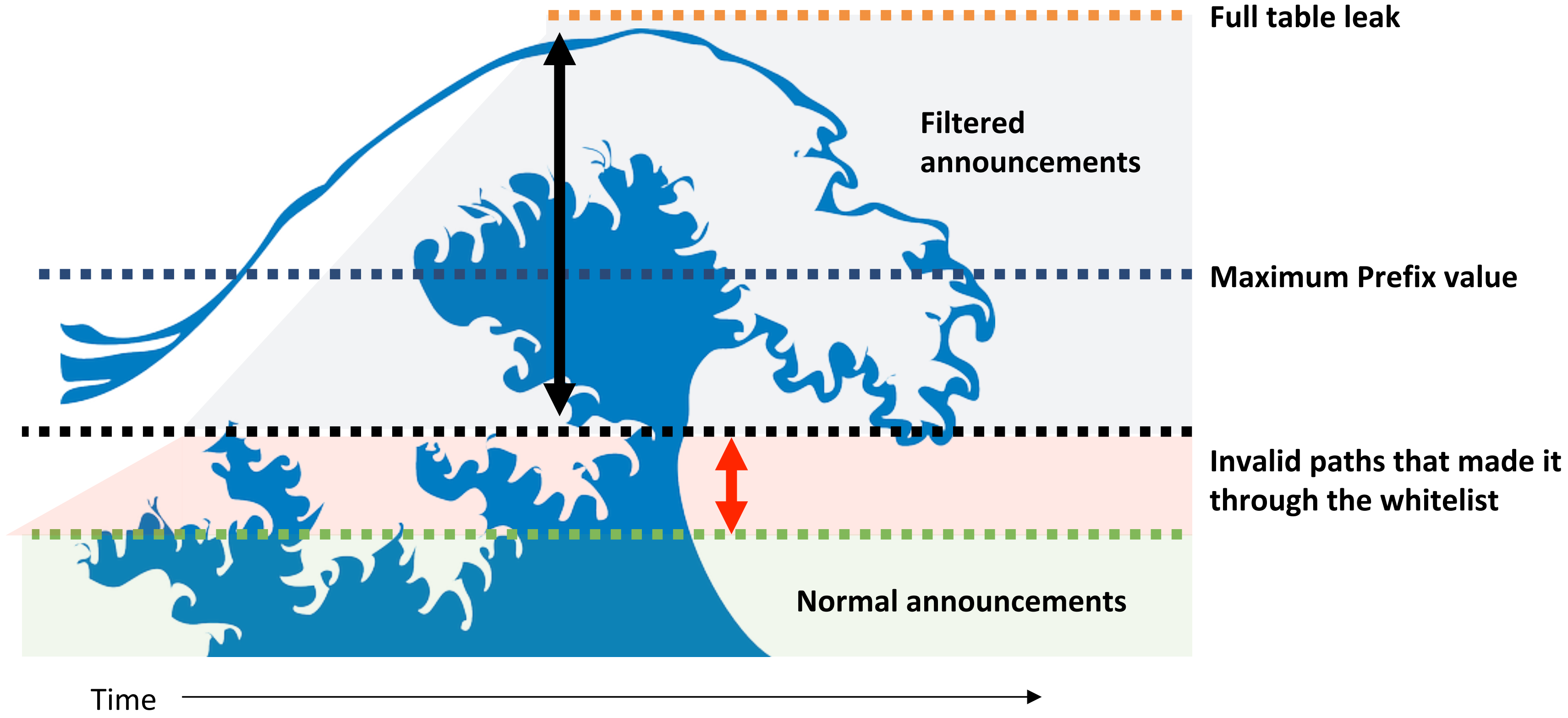
- These limits are a design feature to ensure the network inherently responds in a way that will cause no or minimal harm to the network or the global Internet.

Study resource:
Fail-safe in engineering: https://en.wikipedia.org/wiki/Fail-safe
Control Theory: https://en.wikipedia.org/wiki/Control_theory

What happens when limits are applied in pre-policy during a full table leak:



What happens when limits are applied post-policy



Pre vs Post policy prefix limits in **ebgp-in**

Pre policy limits:

- Protect against memory exhaustion
  - Keep in mind: a pre-policy limit only works if the router remembers the list of rejected routes
- Protect against route leaks

Post policy limits:

- Protect against RIB+FIB exhaustion
- To enforce contractual agreements

Maximum prefix limits in context of **ebgp-in**

Vendor	Pre-Policy (the most effective place)	Post-Policy
Cisco IOS XR	**Not available**	`“maximum-prefix”`
Cisco IOS XE	**Not available**	`“maximum-prefix”`

Juniper Junos	`”prefix-limit”`	`“accepted-prefix-limit”`
		or
		`“prefix-limit” + “keep none”`

Nokia SR-OS	`“prefix-limit”`	Not available

NIC.CZ’s BIRD	`“import keep filtered”`
	+
	`“receive limit”`
		`“import limit”`
		or
		`“receive limit”`

OpenBSD’s OpenBGPD	`“max-prefix”`	Not available

Outbound maximum limits?

This was raised before on nanog@nanog.org – we should work to get *outbound* maximum prefix limits to to use in **ebgp-out**

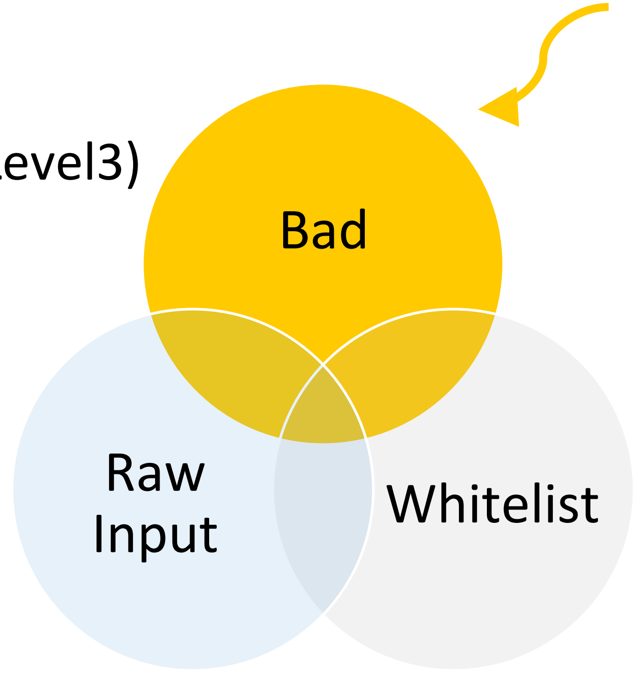
A "self-destruct the session" control action, in case you end up announcing far more than plausible.

Only BIRD supports this today. We'll need to standardize this in IETF.

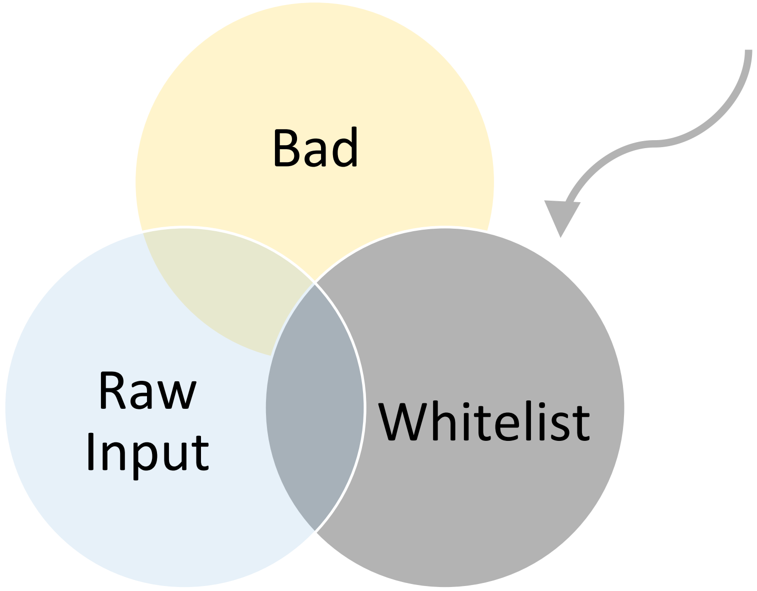
Rejecting *Bad* – defense in depth in **ebgp-in**

- Bogon or Private ASNs
- Bogon or Private Prefixes
- Leaks (example: NTT seeing Comcast via Level3)
- IXP more-specifics
- RPKI Invalid announcements
- Your own space and more-specifics

Study resource:
NLNOG BGP Filter Guide
http://bgpfilterguide.nlnog.net/



Creating a whitelist for **ebgp-in**



Study resource:
ARIN-WHOIS:
https://www.youtube.com/watch?v=L2Zo9AqQqww

Overview of IRR and RPKI Sources:
https://ripe76.ripe.net/archives/video/22/

Now what..?

“When in doubt,
always use BGP communities.”

- traditional Belgian saying

What is a BGP community?

“A community is a group of destinations which share some common property.”

- RFC 1997

Study resource:
RFC 1997: https://tools.ietf.org/html/rfc1997
RFC 1998: https://tools.ietf.org/html/rfc1998

How to use BGP communities?

- ***Classification*** on the **ebgp-in** attachment point
  - "`set community XXX additive`"
- ***Execution*** on the **ibgp-in** and **ebgp-out** attachment point
  - "`match community YYY`"

Common Classifiers

- "learned from transit customer"
- "route via peering partner"
- "learned from upstream provider"
- "route learned in Europe"
- "route learned in Denver, CO"

Common Execution Outcomes

- Announce to this EBGP neighbor
- Do not announce
- Prepend AS_PATH once

Study resource:
RFC 8195
https://tools.ietf.org/html/rfc8195

Day in the life of a BGP announcement

1. AS 15562 announces `192.147.168.0/24` to AS 2914
2. The routing policy at the **ebgp-in** attachment point in 2914 doesn’t reject the announcement: it was not a bogon, and part of the whitelist
3. Still inside **ebgp-in**, AS 2914’s policy classifies the route as “from customer” and “learned in Europe” using BGP communities
4. Still inside **ebgp-in**, features such as LOCAL_PREF modification, blackholing are executed
5. The route announcement propagates to other 2914 routers

Day in the life of a BGP announcement (cont.)

6. Announcement passes through **ibgp-in**, this is an attachment point that offers opportunity for advanced features such as selective blackholing, traffic engineering for anycasters, etc.
7. Announcement enters **ebgp-out**, at this attachment point the classifiers decide whether the route will be announced, and final features are applied such as prepends

Example Classifier / Execution matrix

Classifier (attached in ebgp-in)	ebgp-out to customer	ebgp-out to peer	ebgp-out to upstream
Learned from customer	**accept**	**accept**	**accept**
Learned from peer	**accept**	**reject**	**reject**
Learned from upstream	**accept**	**reject**	**reject**
NO CLASSIFIER	**reject**	**reject**	**reject**

Without a classifier, reject at **ebgp-out**?!

- *"Reject routes without communities in **ebgp-out**"* coincidentally is an incredible safety device, consider:
  - What if you connect a BGP speaker to your network and don't configure policies?
  - What if you accidentally remove the routing policy at the **ebgp-in** attachment point on a session with one of your upstreams?
- If the route does not contain BGP communities that provide explicit guidance on what to do – the route should not be propagated
- The *worst* way of configuring **ebgp-out** policies is doing *only* a match on a prefix-list and calling it a day.
- Bonus: as your network grows, using BGP communities is the least amount of work!

Without a classifier, reject at **ebgp-out**?!

- ”*Reject routes without communities in **ebgp-out**”* is an incredible safety device.

- We call this “*Robust Termination of the routing policy*”

- By applying the *Fail Closed* principle we prioritize security. The network “outage” that results from a failure to correctly set BGP communities on the route is just a delay in the provisioning process. This is far less costly than leaking.

Avoid regular expressions where possible.

- Trying to be clever can result in public embarrassment
- your coworkers will thank you if `grep` just works

Curse or policy? `^\(6(451[2-9]|4[6-9]..|5...)(_6(451[2-9]|4[6-9]..|5...))*\)_.*\(`

" Always code as if the guy who ends up maintaining your routing policy will be a violent psychopath who knows where you live. Write routing policy for readability."

- Adaption of John F. Wood's motto, 1991

Write <u>separate</u> policies and prefix-lists for IPv4 and IPv6

- What is the meaning of an IPv4 prefix-list match on an IPv6 route? Undefined?
- Don’t run IPv4 over IPv6 or vice versa on EBGP: each AFI their own session

Some things simply don’t mix very well… ☺

How many policies to generate?

- One separate policy per ASN per **ebgp-in** attachment point
  - You need per-ASN unique prefix-list filters
- Policies for **ebgp-out** often can be shared across customers
- Peering/Upstreams may share an **ebgp-out**, if you can do conditional matching inside the policy for per-peer specific outbound traffic engineering (otherwise generate **ebgp-out** per-peer)
- **ibgp-out** is often the same across the whole network
- **ibgp-in** is often generated per-device (for selective blackholing & friends)

Overview: so, how many policies are we talking?

Attachment point	When / where to create	Count	Order of magnitude in NTT
ebgp-in	Per EBGP neighbor, per device, per AFI	*N* EBGP neighbors * 2	Tens of thousands
ebgp-out	Per group (customers, peers, etc), per AFI	*N* groups * 2	High hundreds
ibgp-in	Per device, per AFI	*N* devices * 2	Low hundreds
ibgp-out	Network wide, one per AFI	2	1*

Avoid "`set community X`" to delete communities

- Some BGP implementations **delete all** communities and add X
- Some BGP implementations **delete some** communities and add X
- Some BGP implementations add X, and **don't delete anything**
- Instead: use "`delete community Y`", "`set community X additive`"
    - Be precise and concise, delete as little as possible.

NTT went from tens of thousands of instances of "`set community`" to just a few hundred after implementing support for `GRACEFUL_SHUTDOWN`.

Study resource:
Well-known Communities behavior: https://tools.ietf.org/html/draft-ymbk-grow-wkc-behavior

What to communities to delete?

- Network administrators SHOULD scrub inbound communities with their number in the high-order bits, and allow only those communities that customers/peers can use as a signaling mechanism.
- Networks administrators SHOULD NOT remove other communities applied on received routes.
- This may be the *one* place where regular expressions are acceptable

Study resources:
RFC 7454: https://tools.ietf.org/html/rfc7454#section-11

What to communities to send?

- Send at least your geolocation BGP communities to EBGP
- Just like we ask people to be considerate in what they delete, we now ask to be conservative on how many communities you send to others.
- Rule of thumb: don’t send more than 4 BGP communities per route
- <u>Publicly</u> document what your communities mean, on your own website

RFC 8212 – Default Deny on EBGP

What happens when no routing policy is defined at the EBGP attachment points? There now is a RFC that defines what should happen: safety first, don’t exchange routes!

- Cisco IOS XR, BIRD 2.0.2, and OpenBGPD 6.4 support RFC 8212 natively 🎉
- On Arista this can be enabled under “`router bgp …`”:

```
  bgp missing-policy direction in action deny
  bgp missing-policy direction out action deny
```

- On Juniper Junos this can be done with a SLAX script (no native support yet): https://github.com/packetsource/rfc8212-junos
- On Nokia support is coming in 2019-2020.
- Ask your vendors!

Questions, Comments – job@ntt.net