Monitoring BGP and Route Leaks using OpenBMP and Apache Kafka

Tim Evens (tievens@cisco.com)

NANOG-65

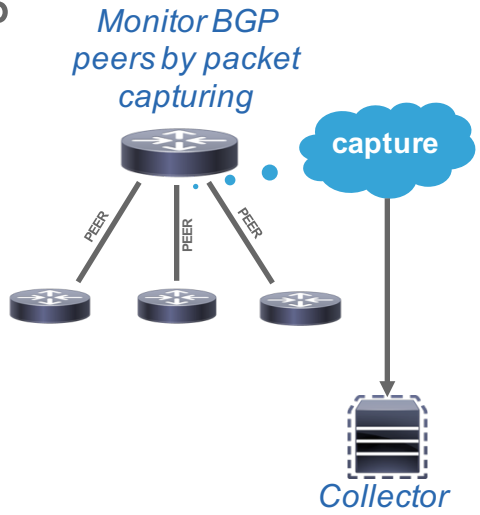
Traditional Method: **VTY** (cli/netconf/xml)

- Data is polled instead of pushed (**not real-time**)
- Large queries can impact router CPU
- VTY access can be slow and cumbersome
- Requires access credentials per router
- Maintainability is complex due to schema/output syntax changes in different software versions
- Maintaining persistent connections to routers may not be feasible

```
Paths: (3 available, best #3)
  Advertised to peers (in unique update groups):
    173.39.225.61   192.168.1.1
  Path #1: Received by speaker 0
  Not advertised to any peer
  6939 1299 3356 13445 1343
    64.71.176.49 from 64.71.176.49 (216.218.252.163)
      Origin IGP, localpref 95, valid, external, group-best
      Received Path ID 0, Local Path ID 0, version 0
      Origin-AS validity: not-found
```

Traditional Method: **Packet Capture**

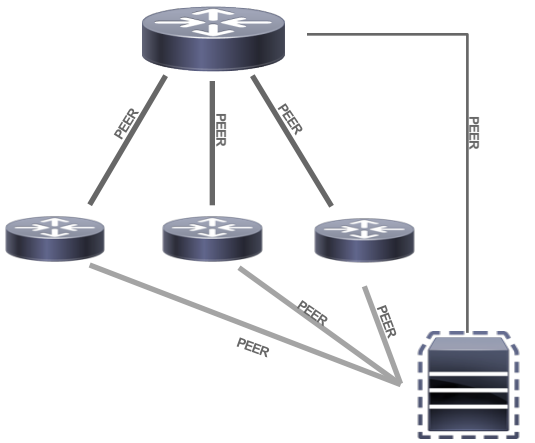
- Requires specific deployment architecture
- Doesn't work well when peering is multi-hop/iBGP considering multiple interfaces could be used for forwarding packets
- Collector has no or limited visibility into router changes that would effect the feed of captured packets
- Can be a security concern



Traditional Method: **BGP Peering**

- Can only see what the router selects (post policy with local-RIB selection) – **no pre-policy**
- ADD-PATH imposes more resources on the router and is still limited to post-policy and local-RIB selection (not pre-policy)
- Collector has no visibility of peers (assumptions have to be made)
- Pre-policy would require peering to all peering routers (**does not scale**)
- Standard peering can be a security concern and can jeopardize the router

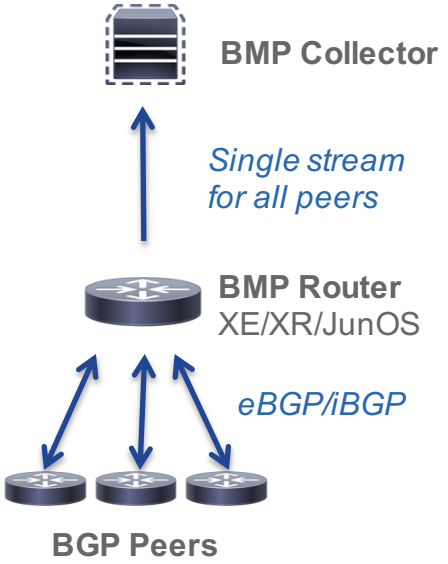
Monitor BGP peers by using standard BGP peering



Collector

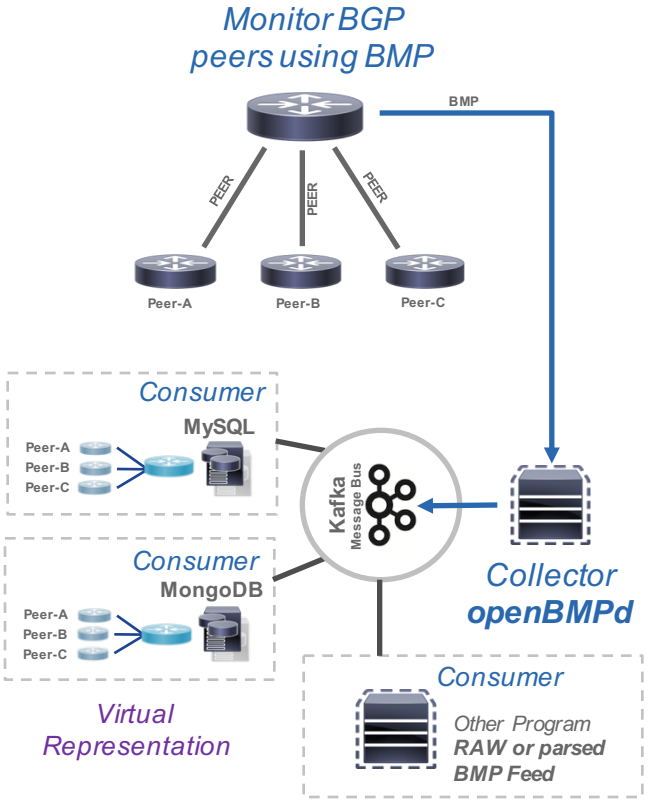
BMP Overview

- BGP Monitoring Protocol (BMP) encapsulates BGP messages from one or more BGP peers into a single TCP stream to one or more collectors
- Efficient, *[near]* real-time, low memory/CPU on router, little to no service impact with peering
- Simplified configuration (one-time setup) with granular controls per peer
- All address families supported
- Controlled RIB dumps
- https://tools.ietf.org/html/draft-ietf-grow-bmp-15

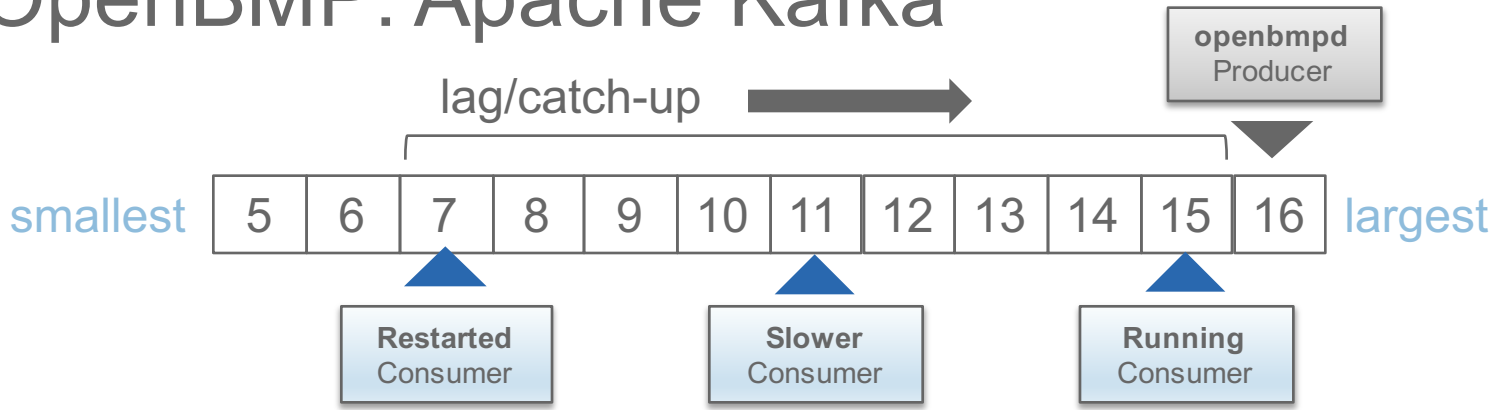


OpenBMP Overview

- OpenBMP is an open-source collector that universally makes available both **parsed** and **RAW** BMP data to any number of applications
- BMP data is forwarded to Apache Kafka in both parsed and BMP raw formats. Any number of consumers can consume this data without requiring multiple BMP feeds from the router
- The MySQL consumer stores parsed data for simplified consumption to enable granular analytics on BGP data without restrictions
- The file consumer logs parsed messages and binary RAW BMP messages in files on disk
- Multiple consumers are available
- www.openbmp.org and github.com/openbmp



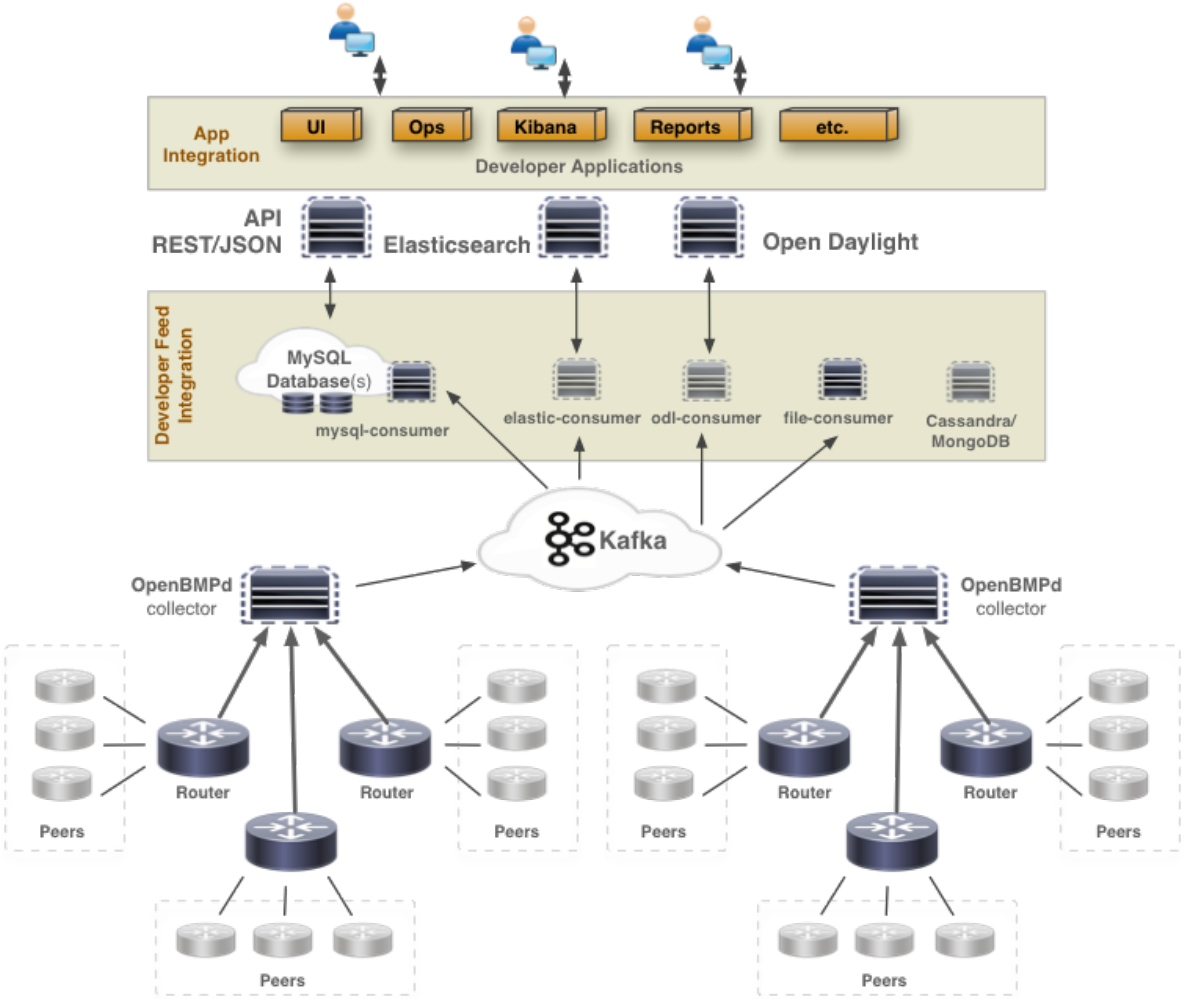
OpenBMP: Apache Kafka



- Consumers track their offset and resume where left off when restarted
- When consumer first starts within group, starting point will either be **smallest** or **largest**
- Kafka retention policy defines how far back the consumer can go. Suggested **log.retention.hours** is **36** hours or less
- Slower consumers do not jeopardize other consumers
- Load balancing consumers
- Track offsets and performance using https://cwiki.apache.org/confluence/display/KAFKA/System+Tools

Architecture

- Common distributed collection of BGP data
- Multiple consumers leveraging existing BGP data feeds, including binary BMP data feeds
- Integration into live data feed is as simple as consuming data from Kafka
- Application integration can be at the Kafka layer or at the app integration layer



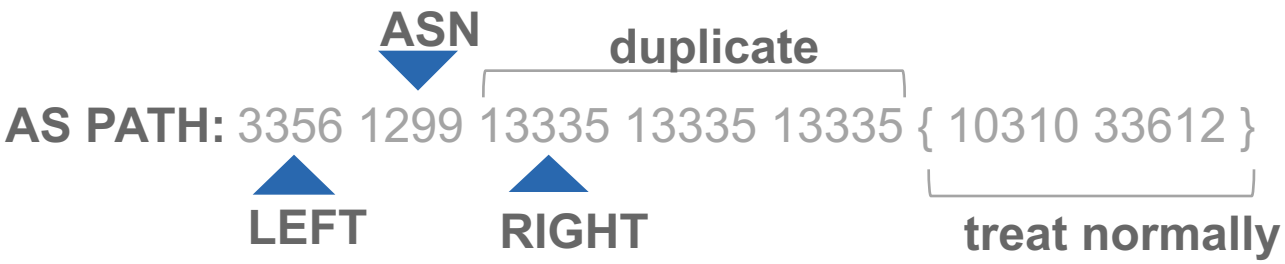
Message Feeds

http://openbmp.org/#!docs/MESSAGE_BUS_API.md

```
bin/kafka-console-consumer.sh  --zookeeper bmp-dev.openbmp.org:2181  \
                --topic openbmp.parsed.unicast_prefix
V: 1
C_HASH_ID: 8d31683940705f39520ccc05bd2d128c
L: 729
R: 2

add     3808297 d82097c34e2e6b2c39c45a97131d62d4 0080e0d7a01dca408c047568804cbed5
        173.39.209.78   a5f605c1a55b7138c2fe57cc14b0a1311f5c87f0b265b5609a2a5df0621faa7b
        x.x.x.x 3356    2015-08-31 20:44:36.427581      105.96.0.0      22      1
        igp      3356 12956 36947       3       36947   x.x.x.b 0       3356:3 3356:22
3356:100 3356:123 3356:575 3356:2006 12956:123 12956:10400 12956:19070 65000:0
        0       1
add     3808298 498a6202fd627d673ec9b0c55aca6d08 0080e0d7a01dca408c047568804cbed5
        173.39.209.78   a5f605c1a55b7138c2fe57cc14b0a1311f5c87f0b265b5609a2a5df0621faa7b
        x.x.x.x 3356    2015-08-31 20:44:36.427581      81.52.162.0     24      1
        igp      3356 12956 36947       3       36947   x.x.x.b 0       3356:3 3356:22
3356:100 3356:123 3356:575 3356:2006 12956:123 12956:10400 12956:19070 65000:0
        0       1
```

As Path Analysis



- Order is maintained to allow rebuilding the distinct AS Path
- Extract each ASN and their respective LEFT and RIGHT ASNs by iterating through the AS PATH
- Upstream ASN is LEFT ASN, Downstream is RIGHT ASN
- Transit is ASN with non-ZERO RIGHT
- Peering is ASN with zero LEFT
- Originating ASN is ASN with zero RIGHT

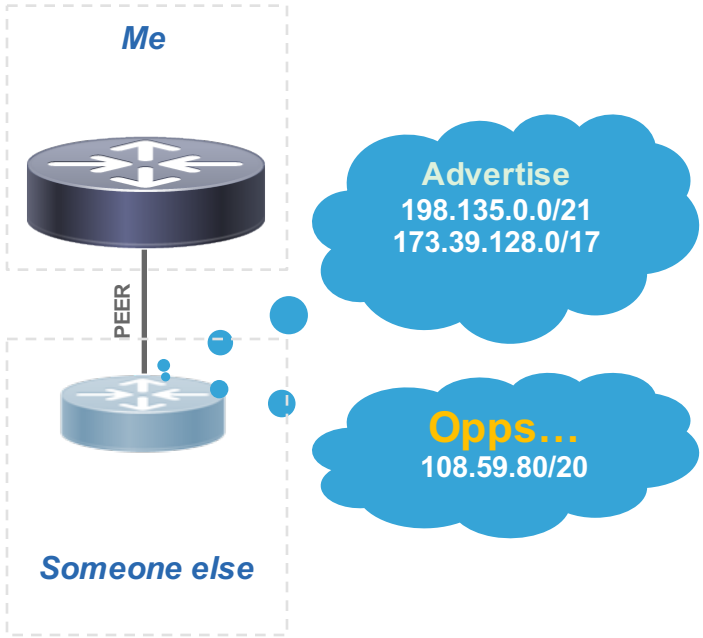
Result: {ASN, LEFT, RIGHT}

```
{3356,  0,     1299}
{1299,  3356,  13335}
{13335, 1299,  10310}
{10310, 13335, 33612}
{33612, 10310, 0}
```

BGP Route-Leaks

A route leak is when routes are advertised to a BGP peer when they shouldn't have been (wrongfully advertised)

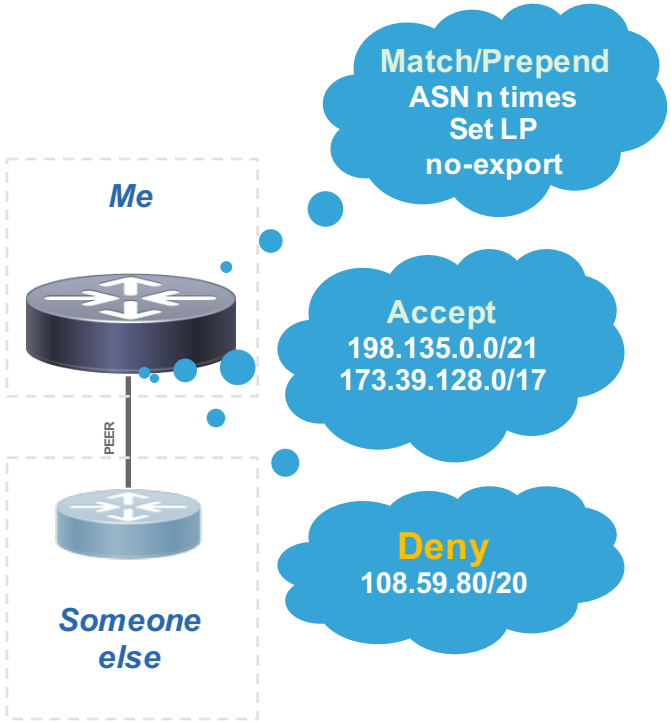
- Is intentional or unintentional
- Can impact few prefixes or many (thousands) prefixes
- Not just Unicast/Internet prefixes, can also be VPNv4/v6 prefixes
- Not limited to originating ASN of prefix
- Agreed policies not being enforced can be considered leaking



Shouldn’t this be moot?

Several techniques are used to prevent route-leaking, but prefixes can still slip through the cracks

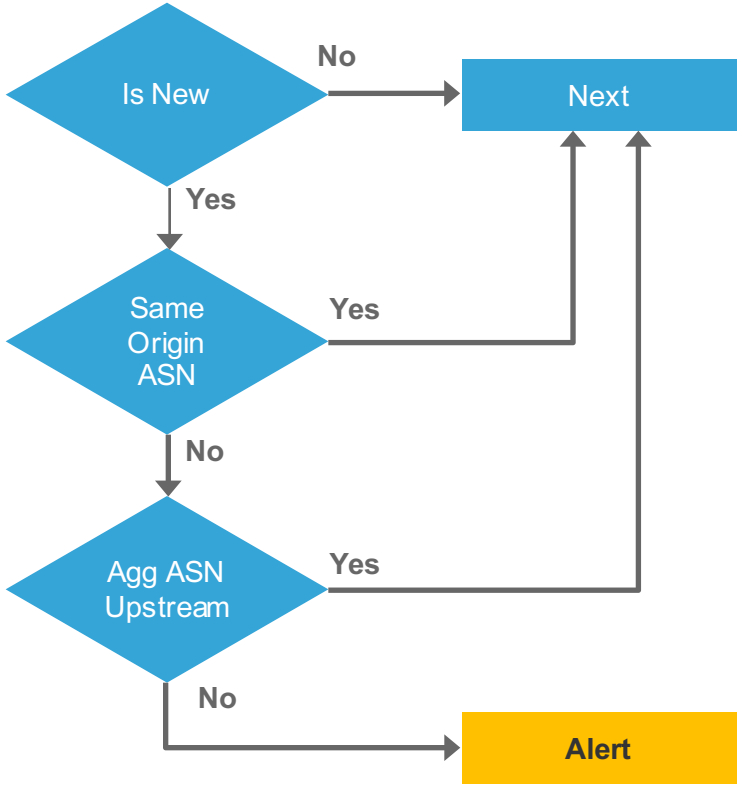
- Not limited to just advertisement or withdrawal of a prefix, it also includes wrongly applied prepending/local-pref/etc.
- In some cases, policies may conflict, such as preferring wholesale over transit, which may use more relaxed filters
- Maximum prefixes doesn’t help when it’s just a few that get advertised inadvertently
- Mistakes can be made with filters, both programmatically and human applied
- RIR, IRR, RPKI, etc. primarily focus on assigned address blocks by originating ASN for registered addresses, leaving VPNv4/v6 out of scope



Route Leak: More Specific Observed

- Check if prefix aggregate ASN matches itself
- Check if aggregate ASN is upstream of prefix ASN
- Alert otherwise
- ~8000 prefixes match with no normalization/baseline

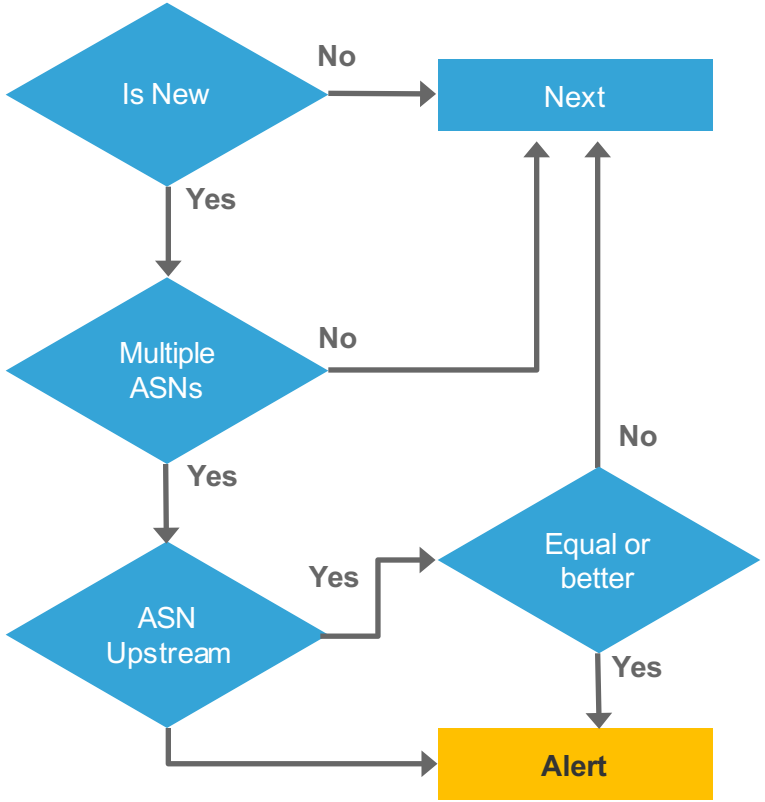
```
SELECT r.prefix as RibPrefix,r.prefix_len as RibLen,r.origin_as as RibOriginAs,
       agg.prefix as AggPrefix,agg.prefix_len as AggPrefixLen,agg.origin_as as AggOriginAs
       FROM rib r
              JOIN rib agg ON (r.prefix_bin <= agg.prefix_bcast_bin and
                  r.prefix_bin >= agg.prefix_bin and r.prefix_len > agg.prefix_len)
              LEFT JOIN as_path_analysis a ON (a.asn = r.origin_as and a.asn_left = agg.origin_as)
       WHERE
           r.isWithdrawn = False AND agg.isWithdrawn = False
           AND r.origin_as != agg.origin_as
           AND a.asn is null
       GROUP BY agg.prefix_bin,agg.prefix_len
```



Route Leak: Originating ASN Inconsistent

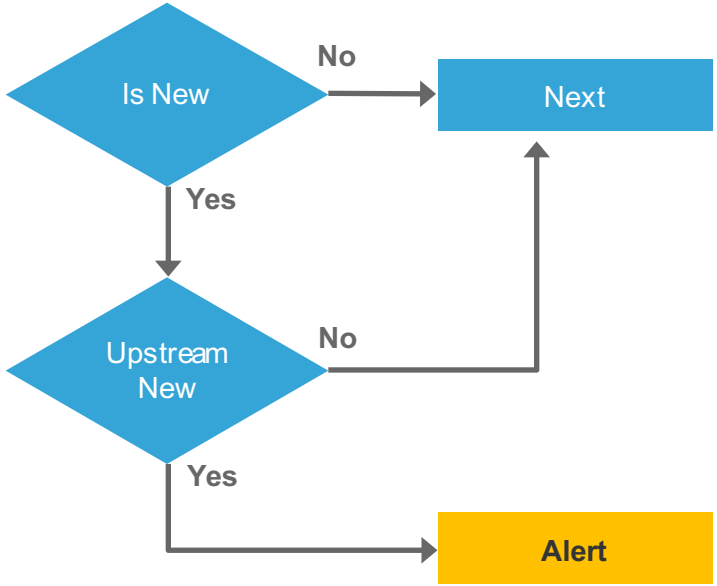
- Check if prefix is actively advertised by other peers with a different originating ASN
- If history includes multiple ASNs but only one active, prefix has likely relocated instead of possible hijack
- ~1000 prefixes match with no normalization/baseline

```
SELECT r.prefix as RibPrefix,r.prefix_len as RibLen,r.origin_as as RibOriginAs,
            count(distinct r.origin_as) as count,
            sum(if(isWithdrawn = False, 1, 0)) as sum_count
    FROM rib r
    WHERE origin_as != 23456
    GROUP BY r.prefix_bin,r.prefix_len
    HAVING count > 1;
```



Route Leak: Unknown Upstream Observed

- Check if adjacent ASN has previously been seen for the originating ASN




```
SELECT distinct asn_left from as_path_analysis
    WHERE asn = 109 AND asn_left != 0
    ORDER BY asn_left;
```

Route Leak: Selection Attribute Changed

Attributes change often, monitoring for such events should take this into account by alerting when multiple prefixes are affected or by snapshotting before/after change

Snapshotting

- Snapshot topology **before** and **after** a change. Report showing prefixes changed by local preference, as path length (including “this” ASN prepends added or removed), well-known community added/removed, MED, …

Scheduled Report

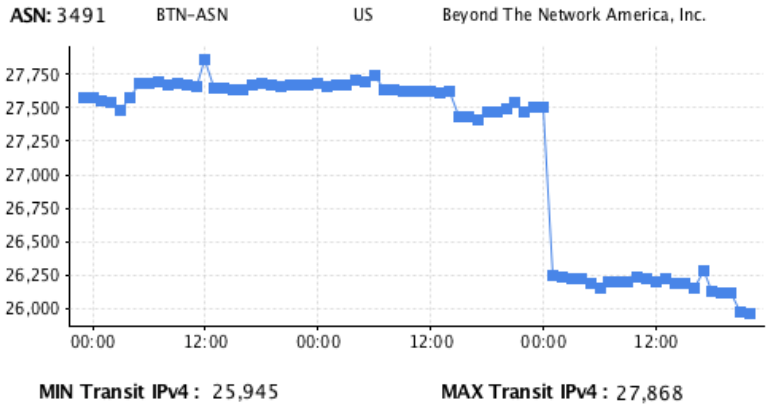
- Uses snapshotting technique but run on a schedule instead of before/after a change

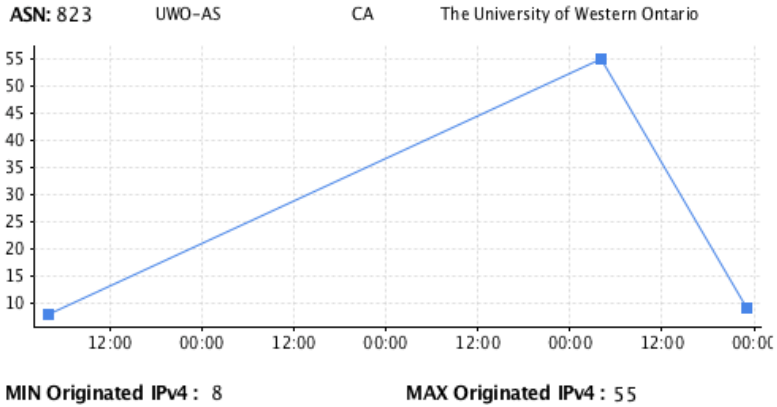
Watch List

- Alert on specific communities or community pattern/convention being added/removed
- Alert on specific prefixes
- Alert on specific originating or transit ASNs
- Watch any attribute and alert if change from previous, this includes the lack of or presence of an attribute

Route Leak: Transit and Origination

Traffic shifting by peer, transit/originating ASN, or by community results in significant prefixes changing in attributes (e.g. next hop, communities, …). A significant change in prefixes seen by transit and/or originating ASNs could be a route leak





Analytics

There are many types of analytics but at a high level, most would fall under one of the below

- Real-Time
  - Alert if a specific condition presents itself, such as prefixes change to traverse a sub-optimal AS
- On-demand
  - Transit AS increases significantly in number of prefixes that would be preferred, trigger an analysis to correlate impacted prefixes
- Discovery & Investigation
  - Report on the number of distinct prefixes that traverse a specific transit AS over time
- Topologies
  - Management application queries for topology data to correlate impact events by source and destination IP addresses

OpenBMP and Apache Kafka can be utilized to collect and analyze historical and real-time BGP data with custom and third-party applications

Install OpenBMP today using docker hub **openbmp/aio**

For more details see:

http://openbmp.org/#!docs/INSTALL.md

Questions?

Thank You

www.openbmp.org

github.com/OpenBMP