Building traffic matrices to support peering decisions

Paolo Lucente

<paolo at pmacct dot net>

http://www.pmacct.net/

NANOG #49 meeting, San Francisco, CA – Jun 2010

Building traffic matrices to support peering decisions

Agenda

- **Introduction**
- The tool: pmacct
- Setting the pitch

Why speaking of traffic matrices?

- Are traffic matrices useful to a network operator in the first place? Yes …
  - Capacity planning (build capacity where needed)
  - Traffic Engineering (steer traffic where capacity is available)
  - Better understand traffic patterns (what to expect, without a crystal ball)
  - Support peering decisions (traffic insight, traffic engineering at the border, support what if scenarios)

Why speaking of traffic matrices?

- Traffic matrices keep a relatively behind the scenes topic
- Some works approach the topic formally
- Other works say about the goodies of traffic matrices:
  - But where to start building one?
  - What challenges the task presents?
  - What resources do I need?
  - Which choices and options do I have?

Back to square 1

(Building traffic matrices to support peering decisions)

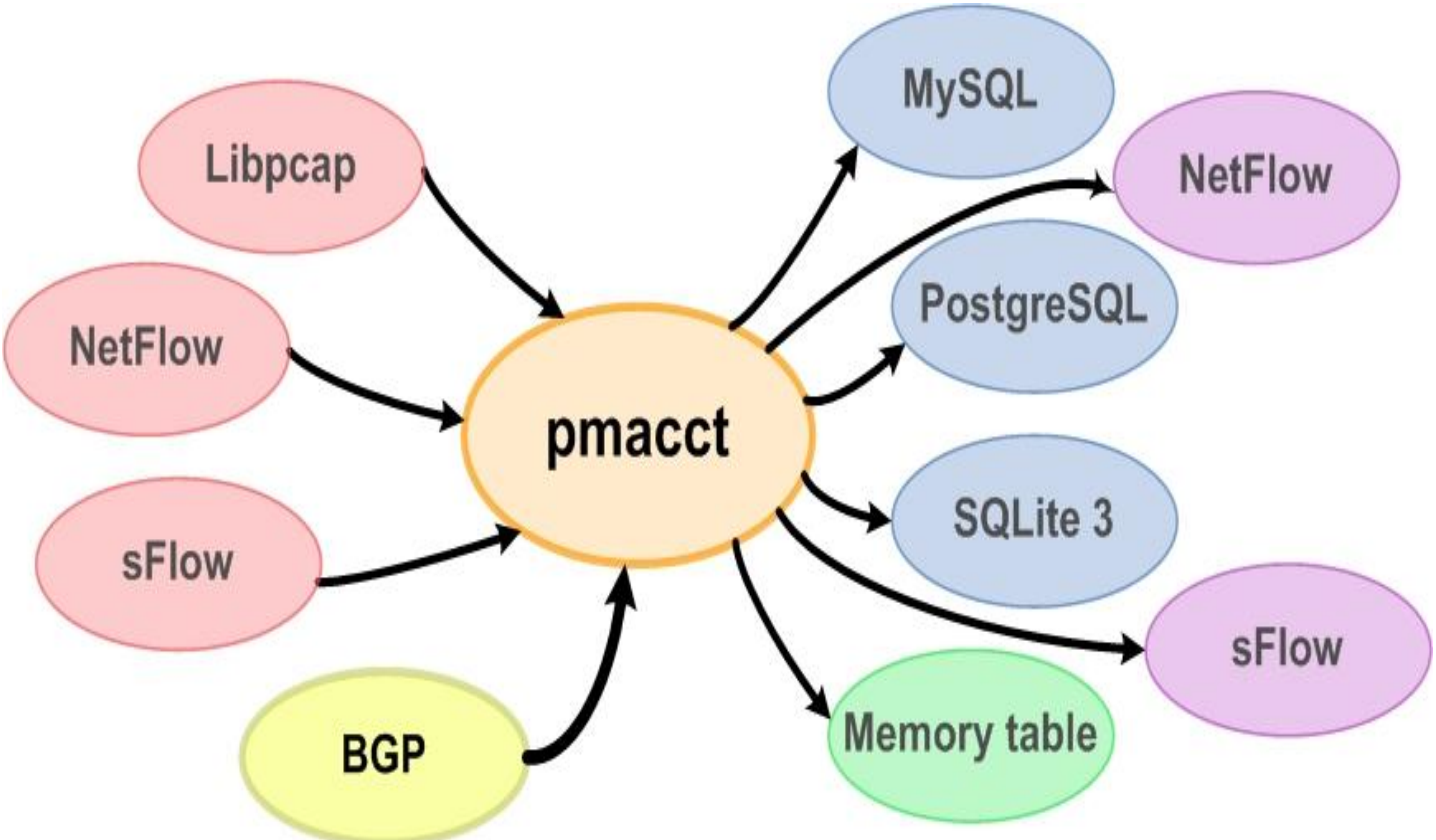
– What is needed:
  - BGP
  - Telemetry data: NetFlow, sFlow
  - Collector infrastructure: tool, system(s)
  - Storage: RDBMS, RRD or home-grown solution
  - Maintenance and post-processing scripts
– Risks:
  - 800 pound gorilla project
  - Switch to "*MySQL for dummies*" from "*How to get a pot of gold from a Leprechaun*" as bedtime reading

Building traffic matrices to support peering decisions

Agenda

- Introduction
- **The tool: pmacct**
- Setting the pitch

pmacct is open-source, free, GPL'ed software



Introducing BGP natively into a NetFlow/sFlow collector

- pmacct introduced a Quagga-based BGP daemon
  - Implemented as a parallel thread within the collector
  - Doesn't send UPDATEs and WITHDRAWs whatsoever
  - Behaves as a passive BGP neighbor
  - Maintains per-peer BGP RIBs
  - Supports 32-bit ASNs; IPv4 and IPv6 families
- Why BGP at the collector?
  - Telemetry reports on forwarding-plane
  - Telemetry should not move control-plane information over and over

Building traffic matrices to support peering decisions

Agenda

- Introduction
- The tool: pmacct
- **Setting the pitch**

Getting BGP to the collector

- Let the collector BGP peer with all PE devices: facing peers, transit and customers.
  - No best-path computation at the collector: scalability preferred to memory usage
  - Count some 50MB of memory per full-routing table
  - Simply take 64-bit at the collector into consideration for 75+ BGP peers scenarios (on a single collector)
- Set the collector as iBGP peer at the PE devices:
  - Configure it as a RR client for best results
  - Collector acts as iBGP peer across (sub-)AS boundaries

Getting BGP to the collector (cont.d)

– BGP next-hop has to represent the remote edge of the network model:
  - Typical scenario for MPLS networks
  - But can be followed up a configurable amount of times in order to cover specific scenarios like:
    - BGP confederations
      - Optionally polish the AS-Path up from sub-ASNs
    - hop-by-hop routing
    - default gateway defined due to partial or default-only routing tables

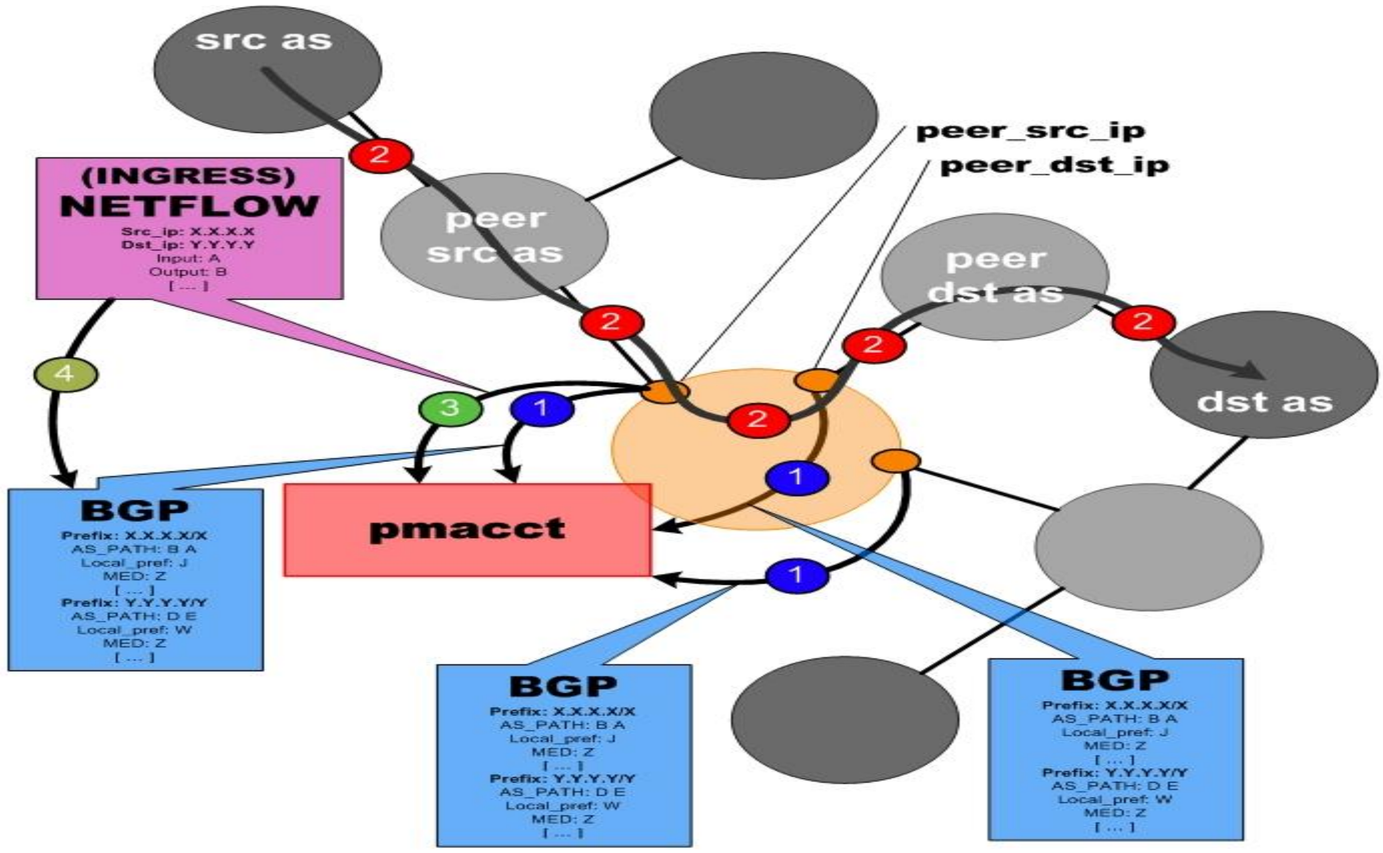
Getting telemetry to the collector

- Export ingress-only measurements at all PE devices: facing peers, transit and customers.
  - Traffic is routed to destination, so plenty of information on where it's going to
  - It's crucial instead to get as much as possible about where traffic is coming from
- Leverage data reduction techniques at the PE:
  - Sampling
  - Aggregation (but be sure to carry IP prefixes!)

Getting telemetry to the collector (cont.d)

- The collector toolbox can include several (say, N) tools. Multiple export models possible:
  - Single tier, unicast: PEs perform N exports
  - Single tier, multicast: PEs perform M exports (M < N)
  - Multiple tiers: PEs perform export to transparent replicators in active/standby fashion; these in turn stream content to the collectors.
- It's crucial collectors can tag, manipulate, filter, discriminate, aggregate, etc. telemetry data.
  - … might be not all data is for everybody

Telemetry data/BGP correlation



1. Edge routers send full BGP tables to pmacct
2. Traffic flows
3. NetFlow records are sent to pmacct
4. pmacct looks up BGP information: NF src addr == BGP src addr

Storing data persistently

- Data need to be aggregated both in spatial and temporal dimensions before being written down:
  - Optimal usage of system resources
  - Avoids expensive consolidation of micro-flows
  - Suitable for project-driven data-sets
- Open-source RDBMS appear a natural choice
  - Able to handle large data-sets
  - Flexible and standardized query language
  - Solid and evolving storage and indexing engines
  - Scalable: clustering, spatial and temporal partitioning

Storing data persisently (cont.d)

```
create table acct_bgp (
```

Tag

```
    agent_id INT(4) UNSIGNED NOT NULL,
```

BGP Fields

```
    as_src INT(4) UNSIGNED NOT NULL,
    as_dst INT(4) UNSIGNED NOT NULL,
    peer_as_src INT(4) UNSIGNED NOT NULL,
    peer_as_dst INT(4) UNSIGNED NOT NULL,
    peer_ip_src CHAR(15) NOT NULL,
    peer_ip_dst CHAR(15) NOT NULL,
    comms CHAR(24) NOT NULL,
    as_path CHAR(21) NOT NULL,
    local_pref INT(4) UNSIGNED NOT NULL,
    med INT(4) UNSIGNED NOT NULL,
```

Counters

```
    packets INT UNSIGNED NOT NULL,
    bytes BIGINT UNSIGNED NOT NULL,
```

Time

```
    stamp_inserted DATETIME NOT NULL,
    stamp_updated DATETIME,
    PRIMARY KEY (…)
);
```

```
shell> cat pretag.map
id=100  peer_src_as=<customer>
id=80   peer_src_as=<peer>
id=50   peer_src_as=<IP transit>
[ … ]
```

```
shell> cat peers.map
id=65534 ip=X in=A
id=65533 ip=Y in=B src_mac=J
id=65532 ip=Z in=C bgp_nexthop=W
[ … ]
```

– In any schema (a subset of) BGP primitives can be freely mixed with (a subset of) L1-L7 primitives

Post-processing and reporting

– Traffic delivered to a BGP peer, per location:

```
mysql> SELECT  peer_as_dst, peer_ip_dst, SUM(bytes), stamp_inserted \
       FROM acct_bgp \
       WHERE peer_as_dst = <peer | customer | IP transit> AND
             stamp_inserted = < today | last hour | last 5 mins > \
       GROUP BY peer_as_dst, peer_ip_dst
```

– Aggregate AS PATHs to the second hop:

```
mysql> SELECT SUBSTRING_INDEX(as_path, '.', 2) AS as_path, bytes \
       FROM acct_bgp \
       WHERE local_pref = < IP transit pref> AND
             stamp_inserted = < today | yesterday | last week > \
       GROUP BY SUBSTRING_INDEX(as_path, '.', 2)
       ORDER BY SUM(bytes)
```

– Focus peak hour (say, 8pm) data:

```
mysql> SELECT … FROM … WHERE … \
        stamp_inserted LIKE '2010-02-% 20:00:00' \
       …
```

Post-processing and reporting (cont.d)

– Traffic breakdown, ie. top N grouping BGP peers of the same kind (ie. peers, customers, transit):

```
mysql> SELECT … FROM … WHERE … \
       local_pref = <<peer | customer | IP transit> pref> \
       …
```

– Traffic matrix (or a subset of it):

```
mysql> SELECT peer_ip_src, peer_ip_dst, bytes, stamp_inserted \
       FROM acct_bgp \
       WHERE [ peer_ip_src = <location A> AND \
             peer_ip_dst = <location Z> AND \ ]
             stamp_inserted = < today | last hour | last 5 mins > \
       GROUP BY peer_ip_src, peer_ip_dst
```

Cariden application notes: regressed measurements

- Use interface stats as gold standard:
  - Traffic management policies based on interface stats
    - ops alarm if 5-min average utilization goes >90%
    - traffic engineering considered if any link util approach 80%
    - cap planning guideline is to not have link util above 90% under any single failure
    - etc.
- Mold NetFlow ... to match interface stats
  - Builds on Traffic Matrix estimation methods
    - Tutorial: Best Practices for Determining the Traffic Matrix in IP Networks, NANOG 43
    - http://www.nanog.org/meetings/nanog43/abstracts.php?pt=MjUmbmFub2c0Mw==&nm=nanog43
  - Adds information from NetFlow to linear system to solve
  - Solve system such that there is strict conformance with link stat values, with other measurements matched as best possible.

Cariden application notes: regressed measurements deployment

- Interface counters remain the most reliable and relevant statistics
- Collect NetFlow as convenient:
  - Can afford partial coverage (ie. a few big POPs)
  - More sparse sampling (ie. 1:10000 instead of 1:1000)
  - Less frequent measurements (ie. hourly instead of 5 mins)
- Use regression (ie. Cariden Demand Deduction™ or similar method) to find the traffic matrix conforming primarily to interface stats but is guided by NetFlow stats

Briefly on scalability

- A single collector might not fit it all:
  - Memory: can’t store all BGP full routing tables
  - CPU: can’t cope with the pace of telemetry export
- Divide-et-impera approach is valid:
  - Assign PEs (both telemetry and BGP) to collectors
  - Assign collectors to RDBMSs; or cluster the RDBMS.
- Tricky scenario is BGP next-hop follow-ups:
  - Gateways or RRs peer with all collectors or
  - All eggs in one basket approach or
  - BGP peer mapping
    - Optionally introduce a route-server layer in the middle

Briefly on scalability (cont.d)

- Intuitively, the matrix can become big:
  - Can be reduced by excluding entities negligible to the specific scenario:
    - Keep smaller routers out of the equation
    - Filter out specific (class of) customers on dense routers
    - Strip down to the essential specific traffic directions (ie. downstream if CDN, upstream if ISP)
    - Sample or put thresholds on traffic relevance
- Project-driven data set:
  - If we were to use this for <billing, security, ...> …
  - … we would aggregate differently in the first place

Further information

- http://www.pmacct.net/lucente_pmacct_uknof14.pdf
  - AS-PATH radius, Communities filter, asymmetric routing
  - Entities on the provider IP address space
  - Auto-discovery and automation
- http://wiki.pmacct.net/OfficialExamples
  - Quick-start guide to setup a NetFlow/sFlow+BGP collector instance
- http://wiki.pmacct.net/ImplementationNotes
  - Implementation notes (RDBMS, maintenance, etc.)

Building traffic matrices to support peering decisions

Thanks for your attention!

Questions?

Paolo Lucente

<paolo at pmacct dot net>

http://www.pmacct.net/

NANOG #49 meeting, San Francisco, CA – Jun 2010