How More Specifics increase your transit bill (and ways to avoid it)

Your Speaker Today....

Fredy Künzler

CTO & Founder

kuenzler at init7.net
www.init7.net
www.blogg.ch
www.bgp-and-beyond.com
AS13030

Twitter: @init7

Init Seven AG
Elias-Canetti-Strasse 7
CH-8050 Zürich

AGENDA

A Init7 / AS13030

B Eyeball Traffic Engineering

C Content Networks: Damage

D Content Networks: Solutions

A Init7 / AS13030

Init7 / AS13030

- [marketing slide - intentionally left blank]

DISCLAIMER

These slides show experience examples of the Init7 / AS13030 backbone over various years. They may work or may not work for you. Please use the methods described with care and at your own risk. Init7 or the author cannot be held responsible for any damage occurred by using the methods described here.

B Eyeball Traffic Engineering

We love Peering, don‘t we?

- Many networks try to peer as much as possible and shift traffic away from transit
- Tier-2 networks connected to multiple exchanges usually can do more than 50 (60, 70% ?) of their traffic via peering and see 40, 50% of the global routing table via peering BGP sessions

Content vs. Eyeballs

Content (outbound heavy) networks commonly try to dump the traffic as close to the source as possible and use the cheapest route

Eyeball (inbound heavy) networks tend to fight against (sometimes very) expensive saturated links and use measures to load-balance the traffic as well as possible

These interests are often contradicting!

Eyeball Traffic Engineering #1

- If you can't upgrade and money is not sufficient for more capacity, then some links tend to be more saturated than others
- Refer to my slides “How to maximize the available capacity!” for traffic engineering measures
  (hint: selecting your transit provider wisely...)

http://www.blogg.ch/uploads/BGP-traffic-engineering-considerations.pdf

Eyeball Traffic Engineering #2

What Eyeball Networks usually do:

- Prepending (works sometimes)
- Selective announcements to various suppliers (drawback: less redundancy)
- Community based traffic engineering (instruct the transit to prepend / do not announce certain prefixes)

Refer to http://onesc.net/communities/

Eyeball Traffic Engineering #3

What Eyeball Networks usually do :

- More-Specific propagation → Acceptable, when smartly & decently executed
- Massive de-aggregation → Pollution of the Global BGP table! → more than 40% of the table size is rubbish…

Eyeball Traffic Engineering #4

- We all know the CIDR report, which shows how much smaller the Global BGP table could be if everybody would aggregate neatly:

```
Aggregation Summary

The algorithm used in this report proposes aggregation only when
   there is a precise match using AS path so as to preserve
   traffic transit policies. Aggregation is also proposed across
   non-advertised address space ('holes').

--- 04Feb12 ---
ASnum   NetsNow   NetsAggr NetGain   % Gain    Description
Table   398067    230015     168052     42.2%    All ASes

Source: http://www.cidr-report.org/as2.0/#Gains
```

Eyeball Traffic Engineering #5

Aggregation of prefixes would be good for the whole community / industry:

- Less memory usage
- Faster BGP conversion / less CPU cycles
- Longer life of equipment

Why are networks so selfish and don't aggregate neatly? Even though they are listed in the top #30 of the polluters for years they don't care...

Eyeball Traffic Engineering #6

Reasons other than traffic engineering for de-aggregation seen in the Global BGP table:

- ‘No-export' community not set
- ‘neighbor x.x.x.x send-community' not set
- lack of knowledge
- “Best [worst] practice consulting” out in the wild – who actively promotes it?!

Eyeball Traffic Engineering #7

... evangelize aggregation!

If everybody would convince customers / fellow network engineers / peers to get rid of the de-aggregated prefixes, the whole community would gain!

C Content Networks: Damage

Content Networks: Damage #1

- Face facts: De-aggregation is a reality and won't vanish anytime soon, actually it gets worse every day
- Due to De-Aggregation we will continue to learn More Specifics from transit, which would actually have been covered by Less-Specifics from Peerings

Content Networks: Damage #2

- This makes all our traffic engineering efforts pretty useless because a More-Specific prefix always wins, regardless of local-pref, MED et. al.

Result:
Peering Traffic is shifted towards expensive transit!

Content Networks: Damage #3

- How can we avoid the damage, if your network is outbound heavy?
- Lets do some analysis of the routing table. What do we actually see?

Content Networks: Damage #4

Bad example #1 (anonymized)

```
   Network          Next Hop Metric LocPrf Weight Path
*>i10.64.0.0/15     10.0.0.1   1    140      0    2222 i
*>i10.64.0.0/18     10.0.0.2   1     50      0    4444 2222 i
*>i10.64.0.0/19     10.0.0.1   1    140      0    2222 i
*>i10.64.32.0/19    10.0.0.1   1    140      0    2222 i
*>i10.64.64.0/18    10.0.0.2   1     50      0    4444 2222 i
*>i10.64.64.0/19    10.0.0.1   1    140      0    2222 i
*>i10.64.96.0/19    10.0.0.1   1    140      0    2222 i
*>i10.64.128.0/18   10.0.0.2   1     50      0    4444 2222 i
*>i10.64.128.0/19   10.0.0.1   1    140      0    2222 i
*>i10.64.160.0/19   10.0.0.1   1    140      0    2222 i
*>i10.64.192.0/18   10.0.0.2   1     50      0    4444 2222 i
*>i10.64.192.0/19   10.0.0.1   1    140      0    2222 i
*>i10.64.224.0/19   10.0.0.1   1    140      0    2222 i
*>i10.65.0.0/18     10.0.0.2   1     50      0    4444 2222 2222 2222 i
*>i10.65.0.0/19     10.0.0.1   1    140      0    2222 I
[etc. - more similar prefixes]
```


Content Networks: Damage #5

Bad example #2 (anonymized)

```
   Network          Next Hop Metric LocPrf Weight Path
*>i10.104.0.0/13    10.0.0.1   1    150      0 7777 9999 5555 2222 ?
*>i10.104.0.0/15    10.0.0.1   1    150      0 7777 9999 5555 2222 ?
*>i10.106.0.0/15    10.0.0.2   1     50      0 4444 6666 2222 ?
*>i10.108.0.0/15    10.0.0.2   1     50      0 4444 6666 2222 ?
*>i10.109.128.0/17  10.0.0.1   1    150      0 8888 6666 2222 ?
*>i10.110.0.0/15    10.0.0.1   1    150      0 7777 9999 5555 2222 ?
*>i10.110.0.0/16    10.0.0.1   1    150      0 8888 6666 2222 ?
```

D Content Networks: Solutions

Content Networks: Solutions #1

The Challenge

- As there are too many More-Specific prefixes advertised by transit, manual filtering is not an option
- Once a prefix filter is applied on the transit link, the More-Specific prefix has vanished and is no longer visible in "our" BGP table. This makes automated checking for black holes difficult

Content Networks: Solutions #2

The Challenge

- We don't know how routers behave when applying a prefix list with several thousand entries. It actually worked with Brocade XMR gear, but results in a very time consuming BGP conversion (> 10 minutes...)

- The risk of black-holing is severe. Some networks with a lot of de-aggregation may remove covering less specifics without being noticed...

Content Networks: Solution #3

- Init7 did some research and scripted an automated filtering system, based on the Piranha BGP daemon / route collector: http://www.spale.com/download/piranha/

- ~~Status of the filter script at this moment is very alpha and not yet to be published, but Init7 plans to publish its findings under a 'GPL alike' license once the results are more mature~~

Content Networks: Solution #4

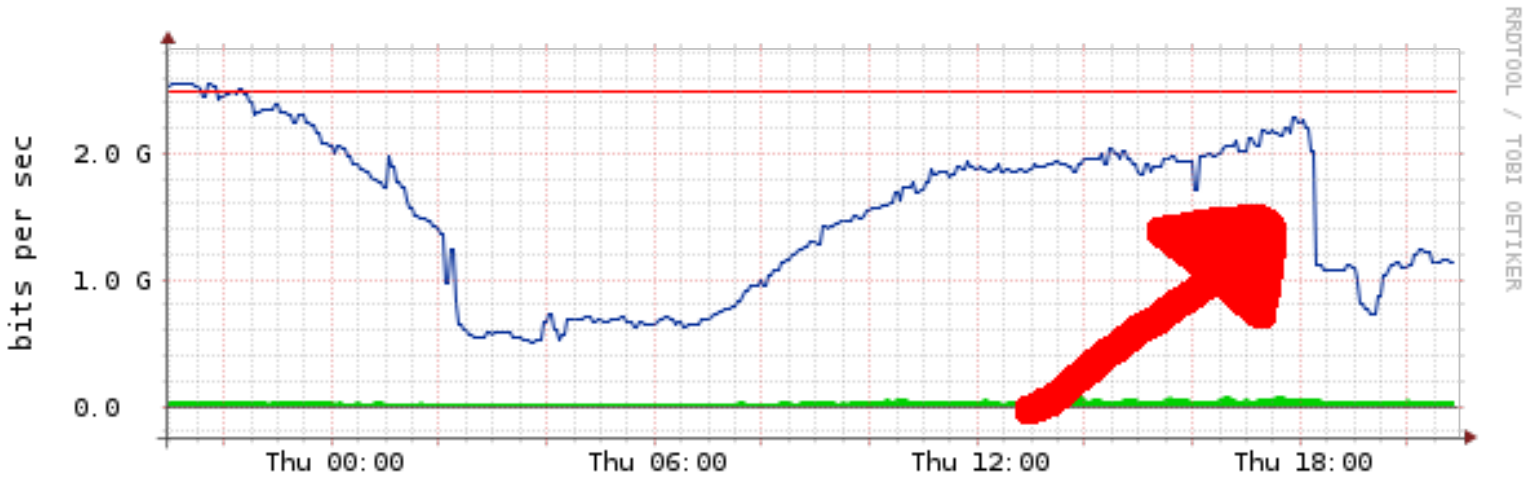
- As mentioned, for the processing of the prefix-list, an unfiltered dump of the transit BGP feed is required
- We achieve this with a shell login and parsing the output of the router command

```
sh ip bgp neighbor x.x.x.x received-routes
```

- …and dump it into the Piranha BGP daemon. This is not the foreseen way of feeding Piranha, but it eliminates the need of the transit supplier to configure a 2nd transit session (with eBGP multihop to our route collector)

Content Networks: Solution #5

- After applying a prefix list of several hundred lines step-by-step, more than 20k More-Specific prefixes got filtered… and transit traffic got reduced a lot – several Gigs!
- This graph shows traffic reduction towards [mayor TIER-1] (based on sflow data), which got shifted to peering



Content Networks: Solution #6

- We started to refine the script, and the prefix-list became more sophisticated, however it really became huge!
- Result: Router (in our case Brocade XMR) could no longer save the running configuration ('wr mem' command)
- On top: when applying a new prefix-list, BGP convergence took hours, not minutes
- Consequence: filtering more-specifics by prefix-list works only for a few hundred or a few thousand prefixes, not on a large scale

Content Networks: Solution #7

- So let's think: how can we handle this problem without overloading router configuration and CPU cycles?

- We need a new BGP feature:

```
router bgp
   neighbor x.x.x.x remote-as yyyyy
   neighbor x.x.x.x suppress-more-specifics
```

Content Networks: Solution #8

- New feature behaviour:

  `neighbor x.x.x.x suppress-more-specifics`

- to be enabled/disabled per neighbor
- not enabled by default
- dumps any received route from RIB/FIB if covered by a less-specific route learned by the same or another neighbor
- active monitoring – if a covering prefix vanishes, the previously dumped prefix should become active

Content Networks: Solution #9

- Other possible solutions #1:
- Quagga | OpenBGPd box which injects a BGP feed with rewritten next-hops
- advantage: if this box crashes, no outage as BGP just works the normal way
- disadvantage: not tested, quite some programming work expected

Content Networks: Solution #10

- Other possible solutions #2:
- large scale prefix filtering along the RIR boundaries - see
  http://www.swinog.ch/meetings/swinog7/BGP_filtering-swinog.ppt
- advantage: BGP table reduces significantly – less memory
- disadvantage: requires default route (many black-holes without!)
- RIR boundaries are meanwhile too often /24

If you have any questions, please contact me...

Fredy Künzler

CTO & Founder

kuenzler at init7.net
www.init7.net
www.blogg.ch
www.bgp-and-beyond.com
AS13030

Twitter: @init7

Init Seven AG
Elias-Canetti-Strasse 7
CH-8050 Zürich