How to maximize the available capacity !

BGP Traffic Engineering

Examples, do's and don'ts, commercial and technical peering aspects and methods.

Emanuel Kleindienst
kleindienst at init7 dot net
AS13030

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

www.init7.net
www.blogg.ch
www.bgp-and-beyond.com

Twitter: @init7

DENOG3: October 20, 2011
Frankfurt, Germany

Init Seven AG / Init7

- Carrier / Internet Service Provider, based in Zurich, Switzerland & Frankfurt, Germany

- privately owned company

- own international fully dual-stacked v4 and v6 backbone (AS13030), 10gig or multiple 10gig enabled

- connected to 20+ internet exchanges, close to 1000 BGP peers / customers

[this space is a placeholder for one or more marketing slides]

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.

BGP 4 Traffic Engineering

Two types of traffic:

- Inbound Traffic (ingress)

- Outbound Traffic (egress)

We use a totally different set of knobs and tools to adjust inbound or outbound traffic. Which one is more relevant for you depends on your network structure (more eyeball customers vs. more content customers).

The Peering Coordinator (PC)

… an „Egg-Laying-Wool-Milk-Pig“?

(… a jack of all trades?!)

The Peering Coordinator #1

… it's a job with a lot of aspects to cover:

COMMERCIAL ASPECT:
– very tight budget given by the CFO!

Two quotes:

- (Init7 customer): „I have 200 Gigabit of traffic and EUR 100k per month …“ (and the consequence: “where do I buy for the best price?)

- (Peering Coordinator of a European incumbent): “There is always more bandwidth (needed) than money (available)…”

The Peering Coordinator #2

TECHNICAL ASPECT:

- BGP4 is the protocol to learn. In theory / lab everything works just fine. But real life out there in the wild is a different story...

We have been working with a CCIE from a consulting company (EUR 250++ per hour), hired by a customer of Init7. This guy managed somehow to pass the CCIE test which is certainly not a piece of cake, but he wasn't able to set a correct BGP4 configuration for peering and transit...

The Peering Coordinator #3

SOCIAL ASPECT:

- the PC must be able and willing to travel to events, talk / negotiate with prospects (potential vendors / peers / customers), always with the best deal for the company in mind... and he/she suffers from
  - jet lag
  - bad food, horrible hotels
  - away from the family
  - too much alcohol
  - not enough sleep
  - unfinished work piling up on the office desk

BUT IT'S FUN TOO!

The Peering Coordinator #4

QUALITY ASPECT:

- the PC is responsible for delivering traffic with best latency and no packet loss or jitter for the various customer requirements:

  - Business customers with Citrix server farms – fast response time
  - Gamers: low latency
  - VoIP users: no jitter
  - Video: massive bandwidth (if Youtube doesn't load fast, it sucks)
  - Live TV: real-time, massive bandwidth, no queueing
  - etc. etc.

The Peering Coordinator #5

TRANSPARENCY ASPECT:

- Every decision a PC takes is visible immediately, as the BGP table hides no secret. Every new BGP relation is noted worldwide.

  - Home users can/do run traceroute
  - Business customers complain about packet loss / bad reachability, even though it's someone else's fault
  - PC often gets involved in politics (i.e. refuse local peering to make the competition's life harder)
  - (Silly) business decisions by the management overrule PC forecasts and plans

The Peering Coordinator #6

FUTURE GROWTH ASPECT:

- Traffic growth is massive. Cisco predicts four times more traffic by 2015. The budget won't grow the same way...

  - raw assumptions about traffic evolves is usually the „glass ball“ of a PC
  - Marketing department often breaks any forecast
    NEW PROMOTION! HOT DEAL!!! All xDSL users get double bandwidth for free starting on July 1!
    … and PC can read this exciting information in the newspaper advertisement. Marketing forgot to ask about available backbone capacity...

The Peering Coordinator #7

DEPENDENCY ASPECT:

- Deployment cycles are way too long. Ordering new 10Gig waves / colocation / routers means a lot of logistics. The upgrade is needed today, but it will hopefully be delivered within 4-5 months.

  - PC should order upgrades before he gets a clue how traffic will evolve, with the risk of ordering the wrong upgrade…
  - „Remote management“. Different time zones, languages, unreliable suppliers, issues with payment…
  - Internal company processes – who can actually decide? If every single cross connect order has to be approved by the management, PC has a hard life…

The Peering Coordinator #8

UNRELIABLILTY ASPECT:

- Things break. For technical reasons, for political reasons, for (silly) business decisions, by the „Act of God“... and PC is to blame:

- fibre cut
- power outage in repeater stations (because the billing department forgot to pay the invoice)
- dirty patch cables
- broken or stolen hardware
- Political unrest (Egypt, Libya, Syria cut themselves off)
- Peering spats between transit providers (i.e. Cogent AS174 has a very long de-peering history)
- etc. etc.

The Peering Coordinator

… indeed an „Egg-Laying-Wool-Milk-Pig“!

Managing Outbound Traffic (egress)

Remember!

- route-map „TRANSITin“ is affecting outbound

```
router bgp 4
 neighbor 192.168.20.30 remote-as 1
 neighbor 192.168.20.30 description "TIER-1 UPSTREAM”
 neighbor 192.168.20.30 next-hop-self
 neighbor 192.168.20.30 soft-reconfiguration inbound
 neighbor 192.168.20.30 prefix-list MYSELFv4 out
 neighbor 192.168.20.30 route-map in TRANSITin
 neighbor 192.168.20.30 send-community

route-map TRANSITin permit 10
 set metric +1                    ! MED accepted
 set local-preference 50          ! depreference transit
 set community 65000:1            ! tag the incoming prefixes
```

Config examples are Brocade compatible – use it with care on Cisco or Juniper!

BGP4 Best Path Selection Algorithm

… learned it, but can you really make use of it?

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… be aware: adjusting BGP4 parameters will only affect the outbound (egress) traffic!

Any traffic which is destined towards your network (ASN) cannot be adjusted by these 13 parameters…

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #1:

1. ~~WEIGHT (proprietary by Cisco)~~
– don't use it, this is not a tool, it's a sledgehammer

2. LOCAL_PREF
- yes, optimal for distinguishing between customer/peer/transit routes (commercial aspect)

Example:
customer route = local_pref 300
peering route = local_pref 150
(default = local_pref 100 – don't use it)
transit route = local_pref 50

Use it with care, it can increase latency!

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #2:

~~3. LOCAL ORIGIN~~
- usually not important, needed for for Anycast (multiple origin) prefixes

4. SHORTER AS-PATH
- very important, primary selection parameter

~~5. ORIGIN TYPE (IGP before EGP)~~
- only important for hot potato routing, makes each router in a BGP mesh behave differently

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #3:

6. MED (Multi Exit Discriminator)

- Important! Used for cold-potato routing and fine-tuning of the preferences (i.e. select always one route over another with the same AS-Path length, as long as the path is available)

~~7. eBGP before iBGP~~

- only in use when two similar routes are available – not a good criteria for traffic engineering

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #4:

8. lowest IGP metric to the next-hop

- OSPF cost will help to find the closest exit in a larger backbone infrastructure.

Example: Peering in three locations with the same peer (i.e. AMSIX, DECIX, LINX with the same AS-PATH length / MED) – this selection criteria will choose automatically the closest exit seen from the origin server

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #5:

~~9. multipath enabled? (best path selected?)~~

- usually not relevant for smaller networks

~~10. older eBGP path preferred~~

- not recommended. If a peer resets (i.e. for maintenance), traffic swaps permanently to another path, and the traffic flow becomes unpredictable, possibly unreliable and you don't even know about it. Makes de-bugging very difficult, because you see only the current path selection, but not the history

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #6:

~~11. lower router ID~~
~~12. lower cluster ID~~
~~13. lowest neighbor address~~

- don't let the selection process go that far. These 'last resort' selection criterias are as good as 'random'. Ensure that every single route is selected at criteria #6 (MED) or #8 (lowest IGP metric) the latest.

1. WEIGHT (proprietary by Cisco)
2. LOCAL_PREF
3. LOCAL ORIGIN
4. SHORTER AS-PATH
5. ORIGIN TYPE (IGP before EGP)
6. MED (Multi Exit Discriminator)
7. eBGP before iBGP
8. lowest IGP metric to the next-hop
9. multipath enabled? (best path selected?)
10. older eBGP path preferred
11. lower router ID
12. lower cluster ID
13. lowest neighbor address

BGP4 Best Path Selection Algorithm

… what is relevant #7:

That's all about engineering the outbound (egress) traffic. It doesn't matter much for traffic leaving the eyeball network, but it's good to know that the packets leave your backbone in a predictable way…

BUT:

Managing inbound (ingress) traffic requires an understanding of how content networks distribute their traffic, because it has the biggest influence on how traffic arrives at your backbone edge!

Managing Inbound Traffic (ingress)

Remember!

- route-map „TRANSITout“ is affecting inbound

```
router bgp 4
 neighbor 192.168.20.30 remote-as 1
 neighbor 192.168.20.30 description "TIER-1 UPSTREAM”
 neighbor 192.168.20.30 next-hop-self
 neighbor 192.168.20.30 soft-reconfiguration inbound
 neighbor 192.168.20.30 prefix-list MYSELFv4 out
 neighbor 192.168.20.30 route-map out TRANSITout
 neighbor 192.168.20.30 send-community

route-map TRANSITout permit 10
 set metric 1000                ! send MED
 set community 65000:2775       ! send instructions
```

Config examples are Brocade compatible – use it with care on Cisco or Juniper!

The Peering Coordinator of AS4 has the following challenge:

- AS4 is an eyeball network

- AS3 is a content network

- AS3 sends a massive amount of traffic towards AS4: ~40% of the traffic towards AS4 is originated by AS3

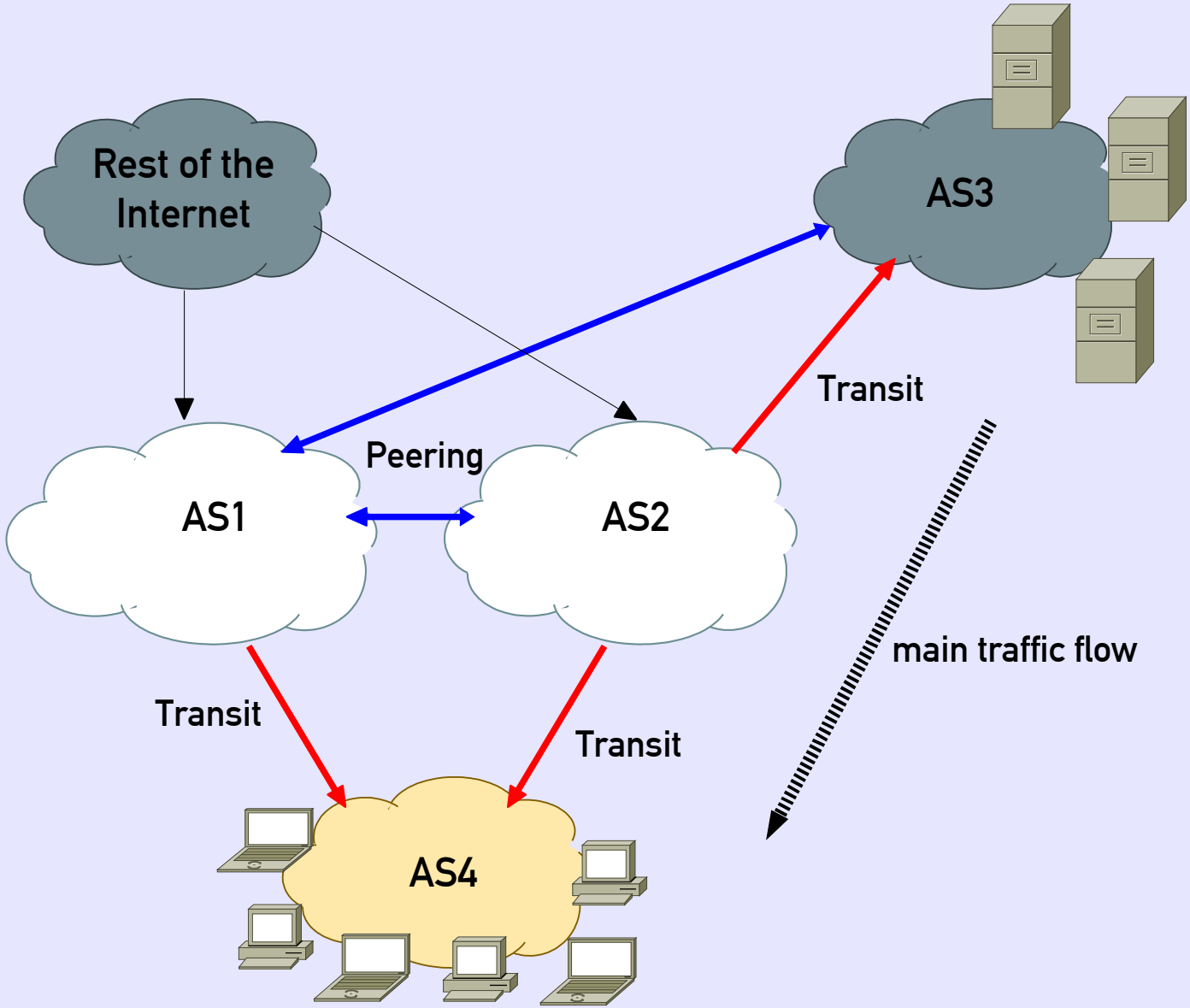
- AS3 buys transit from AS2

- AS3 peers with AS1

- AS4 buys transit from AS1 and AS2

- AS1 and AS2 are peering

<u>Problem:</u>
- link AS1-AS4 is congested
- link AS2-AS4 is empty



How is AS3 sending traffic?

[side-note: all regexp AS paths are with a leading „AS“ for better readability]

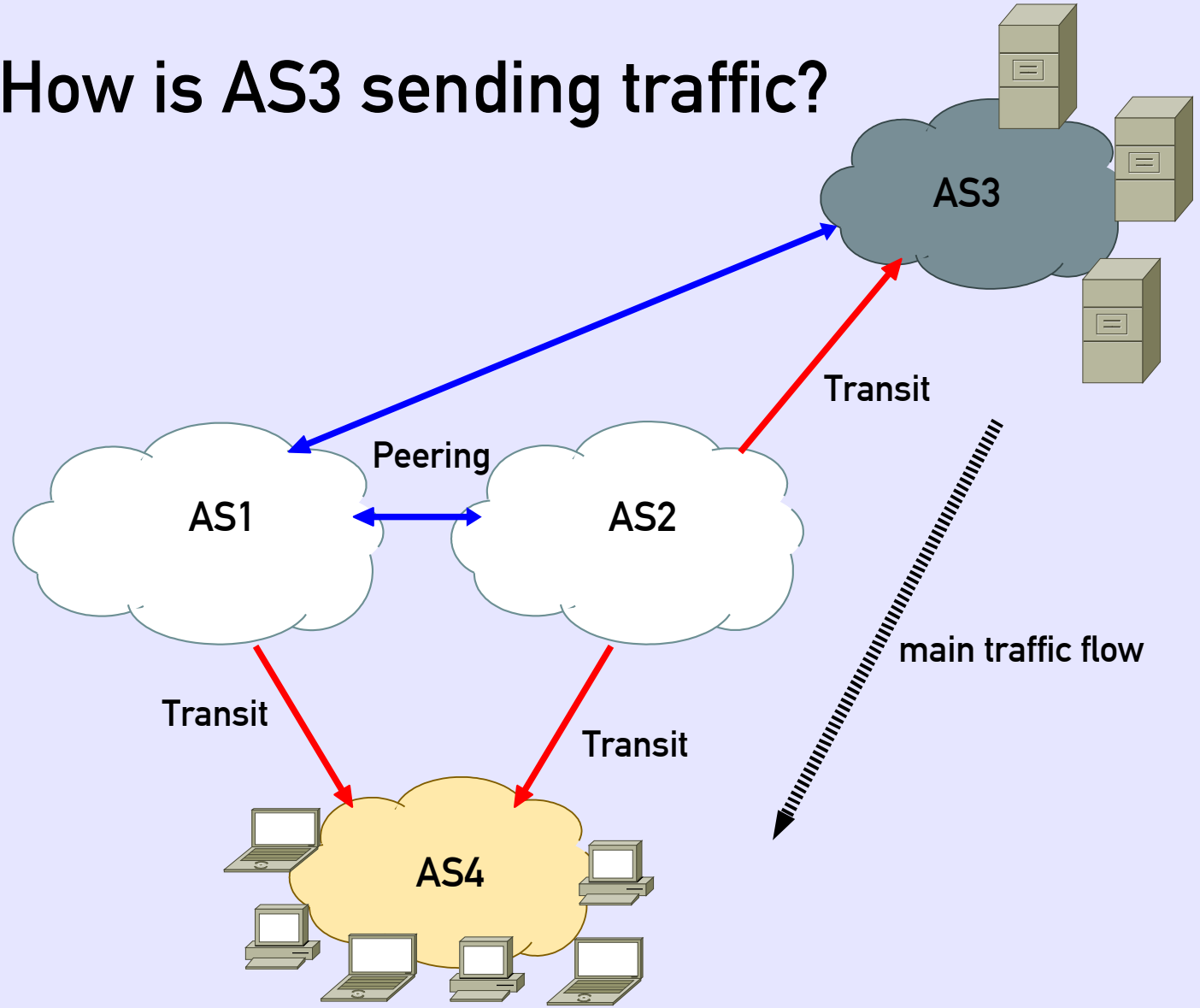
AS3 sees AS4 behind AS1 and AS2:

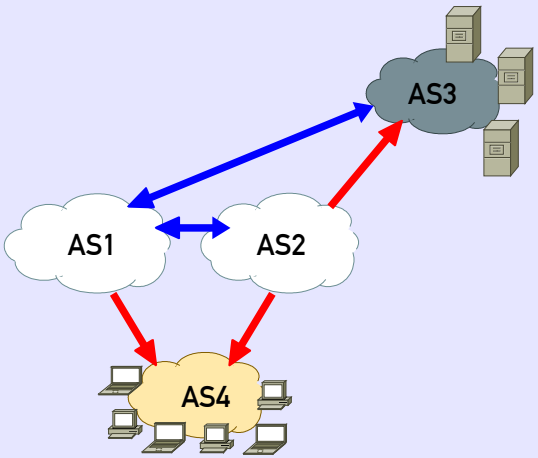
The as-path for the destination prefix(es):

```
^AS1_AS4$
^AS2_AS4$
```

AS3 is peering with AS1, while AS3 buys transit from AS2. Therefore AS3 – in order to avoid transit cost - sends as much traffic as possible via AS1.

To achieve this behaviour, AS3 sets a higher local-pref for the path ^AS1_AS4$.





How can AS4 avoid congestion? #1

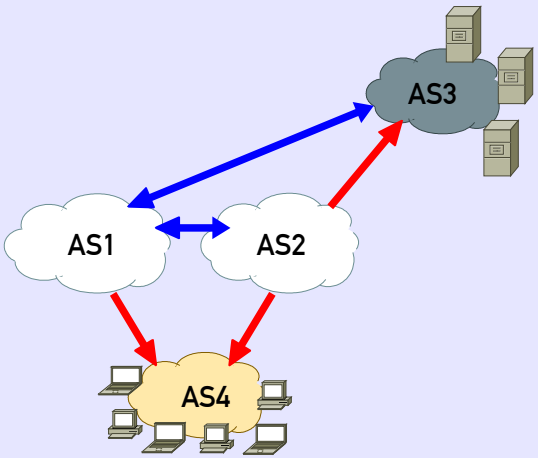
First guess: Peer with AS3!

```
Dear Peering Coordinator of AS3,

We are AS4 and see quite some traffic from you
arriving on our backbone and therefore we would like
to set up a peering...
```

Of course this is a smart and hassle-free solution. Usually content networks as described by AS3 are happy to set up peering, as long as costs are minimal / paying off quickly. Some requirements to consider:
- common location for a private network interconnect (PNI)
- common internet exchange

but let's assume that AS3 cannot peer with AS4 (someone would have to pay long haul transport capacity, for example)...



How can AS4 avoid congestion? #2

Second guess: Prepending!

Prepending once towards AS1 will result in a longer as-path seen by AS3:

```
^AS1_AS4_AS4$
```

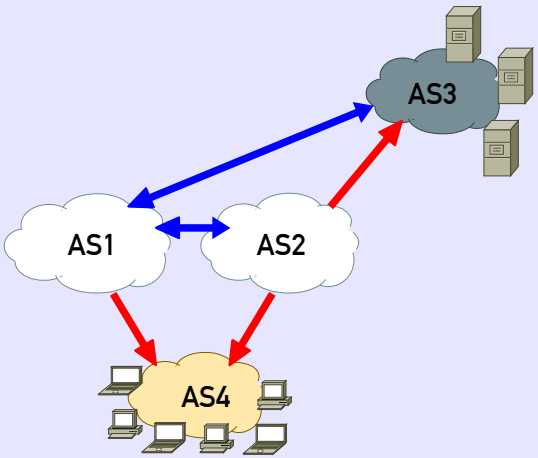
while the path via AS2 is shorter:

```
^AS2_AS4$
```

Result: Little effect. The link AS1-AS4 remains congested, it has no effect on traffic sourced by AS3. It probably will reroute some "rest of the world" traffic from AS1 towards AS2.

Why? AS1 is a peer of AS3, and is preferred by the local-pref setting (let's assume 150 for peering and 50 for transit), regardless of the longer AS path.

AS path is criteria #4, while local-pref is #2 of the BGP path selection algorithm...



How can AS4 avoid congestion? #3

Second guess, reloaded: More Prepending!

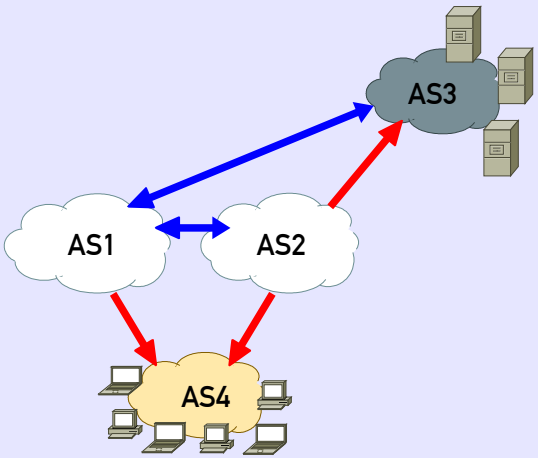
Prepending three times towards AS1 will result in a very long as-path seen by AS3:

```
^AS1_AS4_AS4_AS4_AS4$
```

while the path via AS2 is still very short:

```
^AS2_AS4$
```

Result: Still the same. The link AS1-AS4 remains congested, it has still no effect on traffic sourced by AS3. Likely more “rest of the world” traffic will be rerouted from AS1 towards AS2.



How can AS4 avoid congestion? #4

Ok, AS3. You still choose AS1 over AS2. Im going to punish you now with Massive More Prepending (the full blast)!

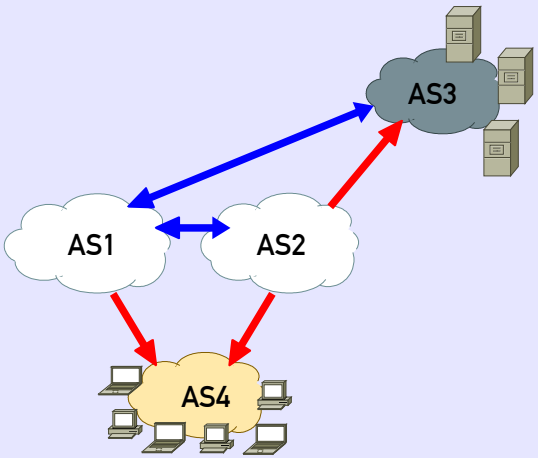
Prepending massive towards AS1 will result in a very long as-path seen by AS3:

```
^AS1_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4
_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4_AS4$
```

Of course, the result is zero.

There are plenty of such stupid long AS pathes out in the wild. It qualifies only the knowledge of the Peering Coordinator of the respective AS.

Rule of the thumb: if three prepends don't help, more prepends won't either. Please avoid it...



How can AS4 avoid congestion? #5

Let's try something else. Assume that AS4 has two prefixes, one /13 and one /16. We are going to propagate the /13 only towards AS2.

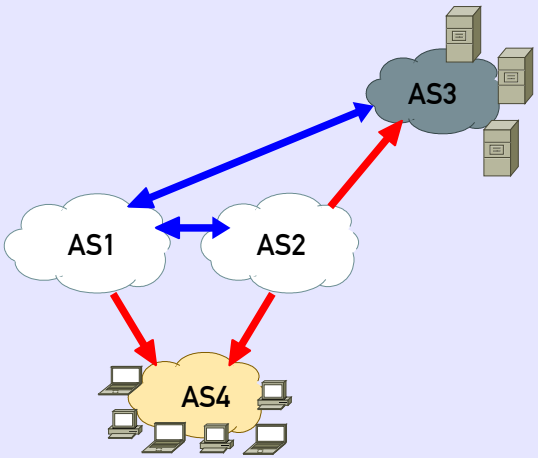
AS3 sees the following paths:

```
10.0.0.0/13       ^AS2_AS4$
10.5.0.0/16       ^AS1_AS4$
```

Result: the congestion swaps to the other transit link, and AS3 is unhappy, as they have to pay more transit capacity...

Traffic engineering by selective prefix advertising has several drawbacks:

1. less redundancy (see next slide)
2. no consistent routing makes de-bugging difficult
3. not too scalable, esp. if prefixes are not equally sized.



How can AS4 avoid congestion? #6

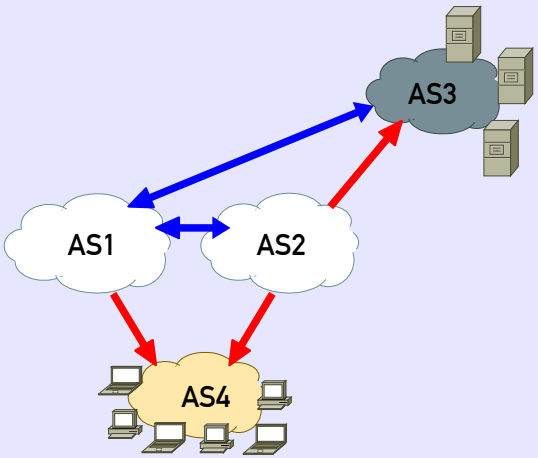
Ok, selective advertisement has drawbacks, but there is a workaround. The traffic from AS3 must be distributed over both transit links. I'm going to de-aggregate the /13 and the /16 into ten /17 and advertise five to each AS1 and AS2.

AS3 sees the following paths:

```
10.0.0.0/17       ^AS2_AS4$
10.0.128/17       ^AS1_AS4$
10.1.0.0/17       ^AS2_AS4$
10.1.128.0/17     ^AS1_AS4$
...
```

Result: the traffic is balanced, and AS3 sends traffic about 50:50 towards both AS1 and AS2. Mission completed? No!

What if AS1 breaks? Half of my networks will be invisible for AS3! Selective prefix advertising certainly breaks redundancy.



How can AS4 avoid congestion? #7

We have to fix the redundancy. This can be achieved by the additional advertisement of the less-specific /13 and /16 networks towards AS1 and AS2, additionally to the ten /17 more-specifics, which are advertised 50:50 to either AS1 and AS2.

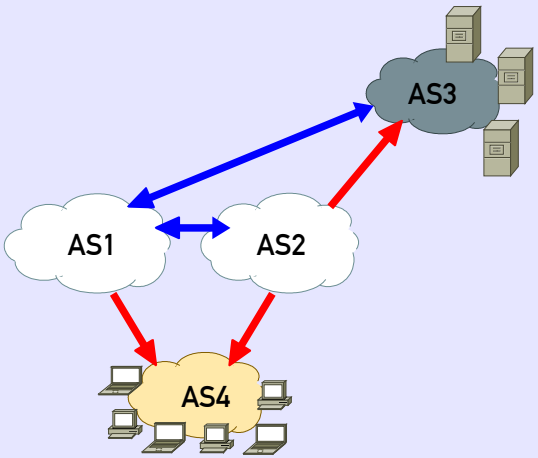
AS3 sees (and prefers by local-pref) the following paths:

```
10.0.0.0/13       ^AS1_AS4$     150
10.5.0.0/16       ^AS1_AS4$     150

10.0.0.0/17       ^AS2_AS4$      50
10.0.128/17       ^AS1_AS4$     150
10.1.0.0/17       ^AS2_AS4$      50
10.1.128.0/17     ^AS1_AS4$     150
...
```

Result: the traffic is balanced, and AS3 sends traffic about 50:50 towards both AS1 and AS2. The /13 and the /16 won't be used as long as there is a more-specific /17 in the table.

Mission completed? Well, half way.



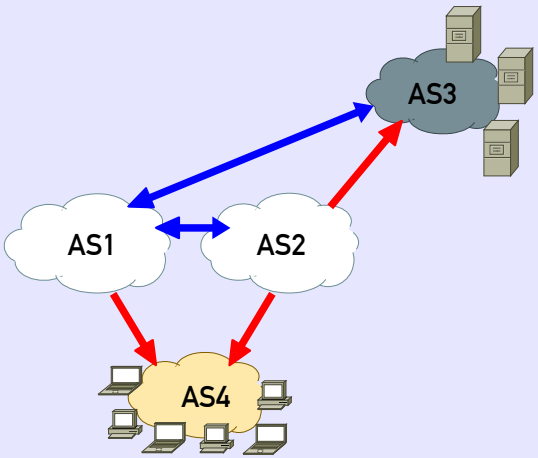
How can AS4 avoid congestion? #8

De-aggregation is generally not recommended, and the global peering community strongly advises not to use de-aggregation. It increases the BGP routing table unnecessarily. Which means: more BGP updates, more CPU cycles, higher memory requirements, higher hardware costs...

The example to de-aggregate one /13 and one /16 into ten /17 is pretty stupid as the same effect could be reached by de-aggregating into two /14 and two /17, advertising only 4 extra prefixes.

The de-aggregation of a /13 into two /14 might just be acceptable, but there are so many networks out in the wild which de-aggregate a /19 into 32 (thirty-two!) /24 etc. for no reason, probably because they don't know better, who knows.

If every prefix were neatly aggregated, the global BGP table would just be about half of the size of today. And with the IPv4 exhaustion, it's going to get worse...



How can AS4 avoid congestion? #9

So we finally have been able to load-balance the traffic between the two transit links AS1 and AS2. We had to pollute the global BGP table with additional 10 (or 4) prefixes, but, so what? No one would blame us... especially not our suppliers AS1 and AS2, because we pay them $$$ … lean back, mission completed.

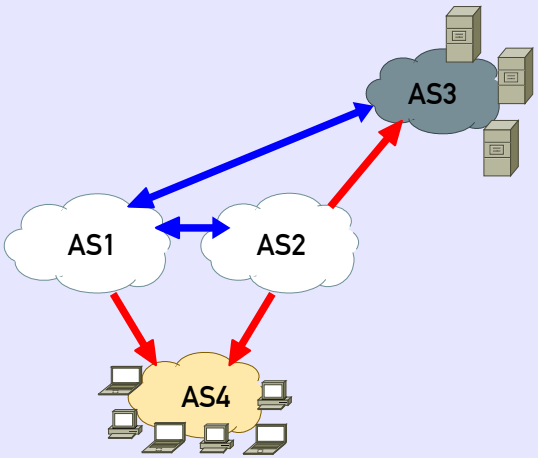
But then... the peering coordinator of AS3 wonders why he all of a sudden has a lot more traffic towards his transit AS2. After some investigation he figures that AS4 started to send more-specifics.

If the Peering Coordinator of AS3 is smart, he will configure a filter, removing again the ten extra /17 prefixes on his border routers, accepting only the /13 and the /16:

```
10.0.0.0/13       ^AS1_AS4$     150
10.5.0.0/16       ^AS1_AS4$     150
```

and traffic switches back to the congested transit link. You again start from scratch.

Content wins 1:0 vs. Eyeballs.



How can AS4 avoid congestion? #10

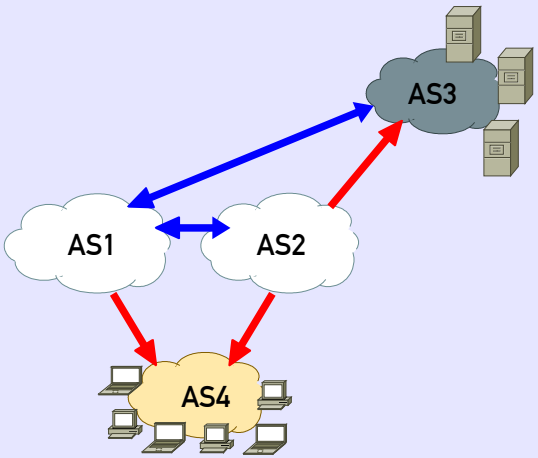
Content wins 1:0 vs. Eyeballs? No! We want revenge!

- Prepending? No.
- selective advertisement? Only with redundancy degradation.
- more specifics? Pollution of the BGP table, risk of filtering

Any other ideas?

BGP Community support by the upstream!

We need to instruct the upstream provider how to treat our routes. Most transit providers do support communities for inbound traffic engineering. An incomprehensive and slightly outdated collection of BGP community support information of various transit providers can be found at http://onesc.net/communities/ - ask your current supplier for the latest information.



How can AS4 avoid congestion? #11

Most larger networks allow settings for each individual prefix like

- do not announce to peer X
- prepend once / two / three times to peer Y

(there are other settings available which are out of the scope of this presentation)

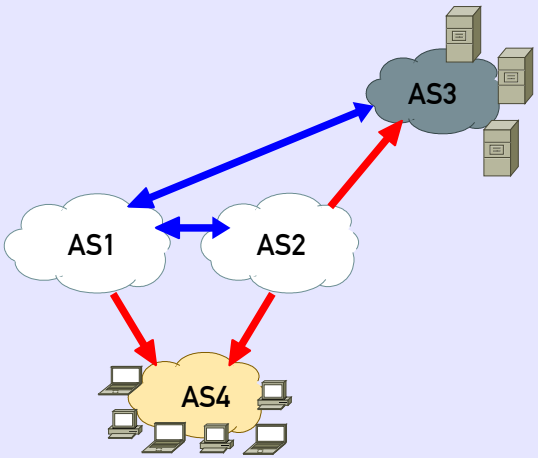
If we instruct AS1 not to announce 10.0.0.0/13 towards AS3, the inbound traffic will flow via AS1 and AS2 for each prefix separate.

AS3 will then see

```
10.0.0.0/13      ^AS2_AS4$        50
10.5.0.0/16      ^AS1_AS4$        150
```

and in the failure case (link between AS2 and AS4 lost):

```
10.0.0.0/13      ^AS2_AS1_AS4$    50
10.5.0.0/16      ^AS1_AS4$        150
```

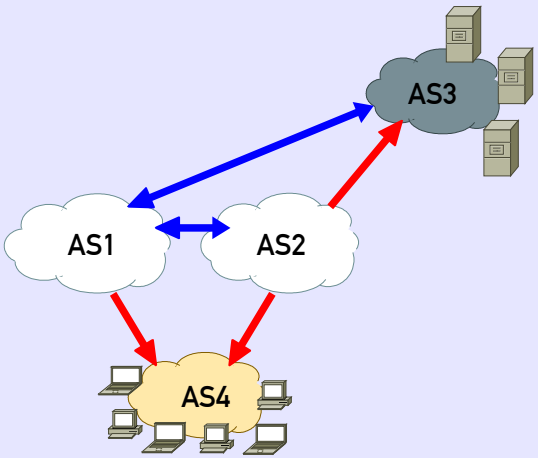


How can AS4 avoid congestion? #12

Conclusion: Inbound traffic engineering is not an easy task. A fine-tuned combination of all the described methods is required:

- (selective) pre-pending
- (selective) advertisement in combination with more-specifics
- (selective) use of BGP community support of the upstream

Before the any of these methods, a forecast is evident. We need to know exactly what the effect is before adjusting a knob... otherwise things break!



How can AS4 avoid congestion? #13

Last but not least another idea how to avoid congestion (without buying more capacity):

Large traffic sources (AKAMAI, Google etc.) do support traffic caching servers. It's possible to shift servers „physically“ from AS3 to AS4. The result: better latency, less traffic on the upstream connection.

The servers of course are not shifted physically, but a new serverfarm is installed within AS4, which takes the load coming from AS3.

AKAMAI calls this program „AANP“ (Akamai Accelerated Network Program) – search for „AKAMAI AANP“ presentations.

Google has a similar program, and other vendors too.

Questions?

Emanuel Kleindienst, AS13030
kleindienst at init7 dot net

www.init7.net
www.blogg.ch
www.bgp-and-beyond.com

Twitter: @init7