Building and Operating Hybrid-Cloud Networks

BENJAMIN MCALARY | PRINCIPLE NETWORK ENGINEER
BMCALARY@ATLASSIAN.COM

Agenda

Network Requirements

Atlassian’s Network + Network Data

War Stories

Agenda

Network Requirements

Atlassian’s Network + Network Data

War Stories

Agenda

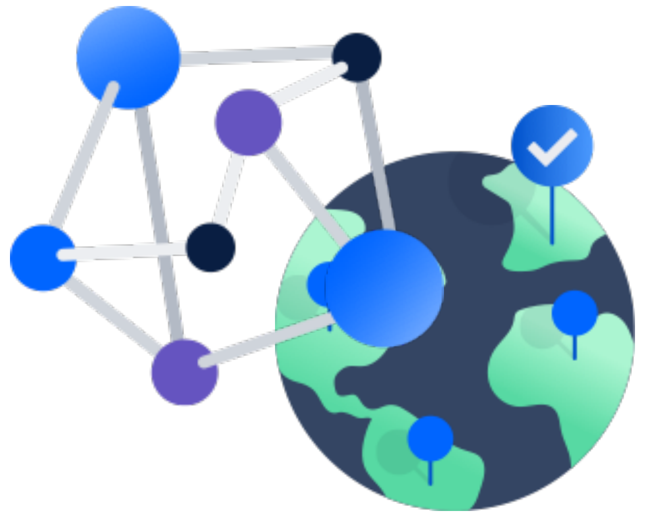
Network Requirements

Atlassian’s Network + Network Data

War Stories

Network Requirements

Features of a ‘good network’



Connectivity to everything

No latency

Infinite bandwidth

Free
(or cheap)

Why do we ~~need~~ want a network?

Security requirements
DDoS, Encryption

Performance
Predictability, visibility

IP space
Limited IPv4, translations are required

Cost
related to 1. and 3.

Why do we ~~need~~ want a network?

Security requirements

DDoS, Encryption

Performance

Predictability, visibility

IP space

Limited IPv4, translations are required

Cost

related to 1. and 3.

Why do we ~~need~~ want a network?

Security requirements

DDoS, Encryption

Performance

Predictability, visibility

IP space

Limited IPv4, translations are required

Cost

related to 1. and 3.

Why do we ~~need~~ want a network?

Security requirements
DDoS, Encryption

Performance
Predictability, visibility

IP space
Limited IPv4, translations are required

Cost
related to 1. and 3.

Atlassian Network Building Blocks

DX

Core

Backhaul

VPC peering

DX

Short for Direct Connect
Redundant pairs of Fibre (4 pairs in some locations)
Physically connects to AWS infrastructure
Can advertise default/summary routes

Atlassian Network Building Blocks

DX

Core

Backhaul

VPC peering

Core

Redundant pair of core network routers
Intersection for all connectivity
8+ 10Gbps interfaces

Atlassian Network Building Blocks

DX

Core

Backhaul

VPC peering

Backhaul

Redundant pairs of private circuits
Going to another regions
Generally 4+
Generally 10Gbps

Atlassian Network Building Blocks

DX

Core

Backhaul

VPC peering

VPC peering

AWS service
Connects 1 VPC to another (prevents the need to hairpin data on our routers)
Some management overhead which we’ve solved

Atlassian’s Network

5

AWS Regions + Physical Points-of-Presence

200+

Privately Routed VPCs

~10

offices

2

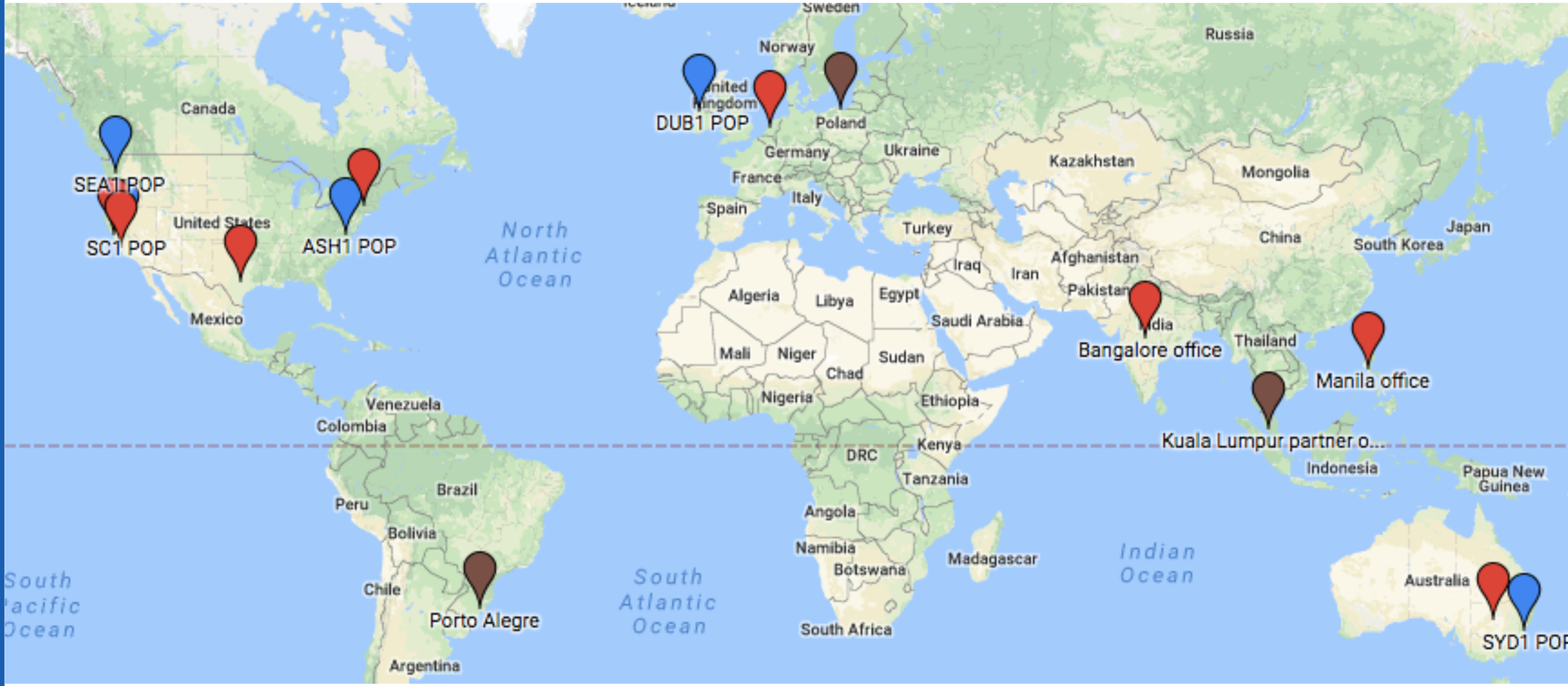
datacenter cages

10-40+

Gbps

Greenland
Iceland
AK
YT
NT
NU
Russia
Sweden
Finland
Norway
Stockholm
Latvia
Denmark
BC
AB
SK
MB
Canada
QC
NL
ON
Ireland
U.K.
Netherlands
Belarus
Poland
Germany
Ukraine
Kazakhstan
Mongolia
Vancouver
France
Switzerland
Austria
Hungary
Romania
MT
ND
WA
OR
ID
SD
WY
NE
NV
United States
Chicago
Toronto
New York
Washington D.C.
Portugal
Madrid
Spain
Italy
Rome
Georgia
Turkey
Greece
Athens
Istanbul
Tashkent
Turkmenistan
Urumqi
Beijing
N. Korea
Seoul
S. Korea
Japan
Tokyo
Osaka
San Francisco
Los Angeles
NM
Tunisia
Casablanca
Morocco
Algeria
Libya
Egypt
Cairo
Syria
Israel
Baghdad
Iraq
Tehran
Iran
Kabul
Afghanistan
Pakistan
China
Chengdu
Shanghai
Houston
Miami
Kuwait
Riyadh
Dubai
U.A.E.
Oman
Saudi Arabia
New Delhi
Nepal
India
Bangladesh
Kolkata
Myanmar
Taipei
Taiwan
Hong Kong
Monterrey
Mexico
Mexico City
Cuba
Puerto Rico
Guatemala
Nicaragua
Panama
Mauritania
Mali
Niger
Chad
Sudan
Eritrea
Yemen
Mumbai
Vietnam
Thailand
Bangkok
Manila
Philippines
Dakar
Senegal
Burkina Faso
Guinea
Liberia
Ghana
Lagos
Nigeria
Cameroon
C.A.R.
South Sudan
Addis Ababa
Ethiopia
Somalia
Bengaluru
Sri Lanka
Caracas
Venezuela
Bogota
Colombia
Ecuador
Peru
Lima
Brazil
Bolivia
Paraguay
Chile
Santiago
Uruguay
Buenos Aires
Argentina
Rio de Janeiro
Sao Paulo
Falk. Isl.
Gabon
Congo
Kinshasa
D.R.C.
Uganda
Kenya
Nairobi
Rwanda
Tanzania
Angola
Zambia
Malawi
Zimbabwe
Mozambique
Namibia
Botswana
Johannesburg
South Africa
Cape Town
Madagascar
Malaysia
Brunei
Singapore
Jakarta
Indonesia
E. Timor
Papua New Guinea
Solomon Islands
Vanatu
Fiji
New Caledonia
N.T.
Qld.
W.A.
Australia
S.A.
N.S.W.
Sydney
A.C.T.
Melbourne
Tas.
Auckland
New Zealand

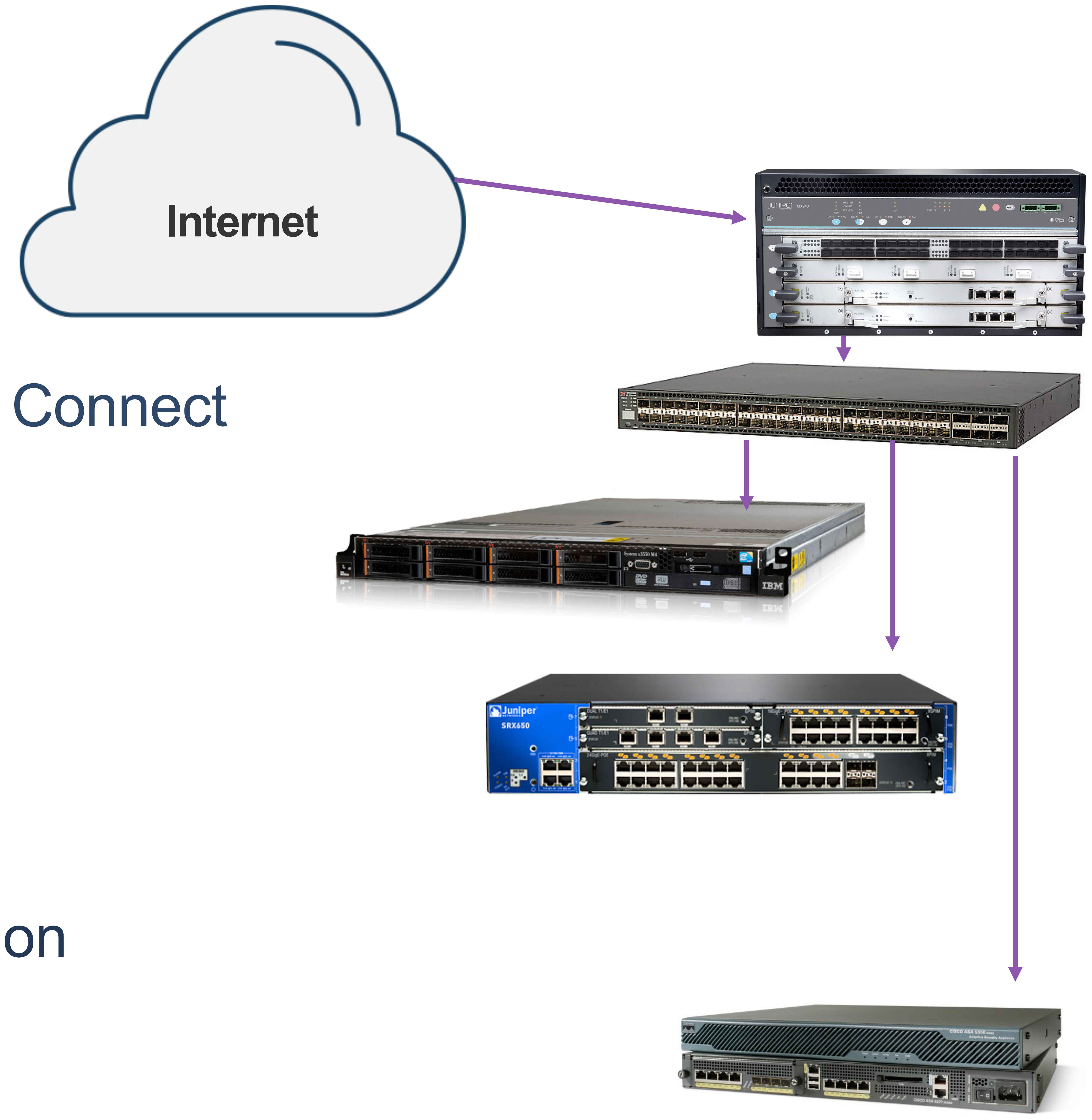
Atlassian Sites // Presence

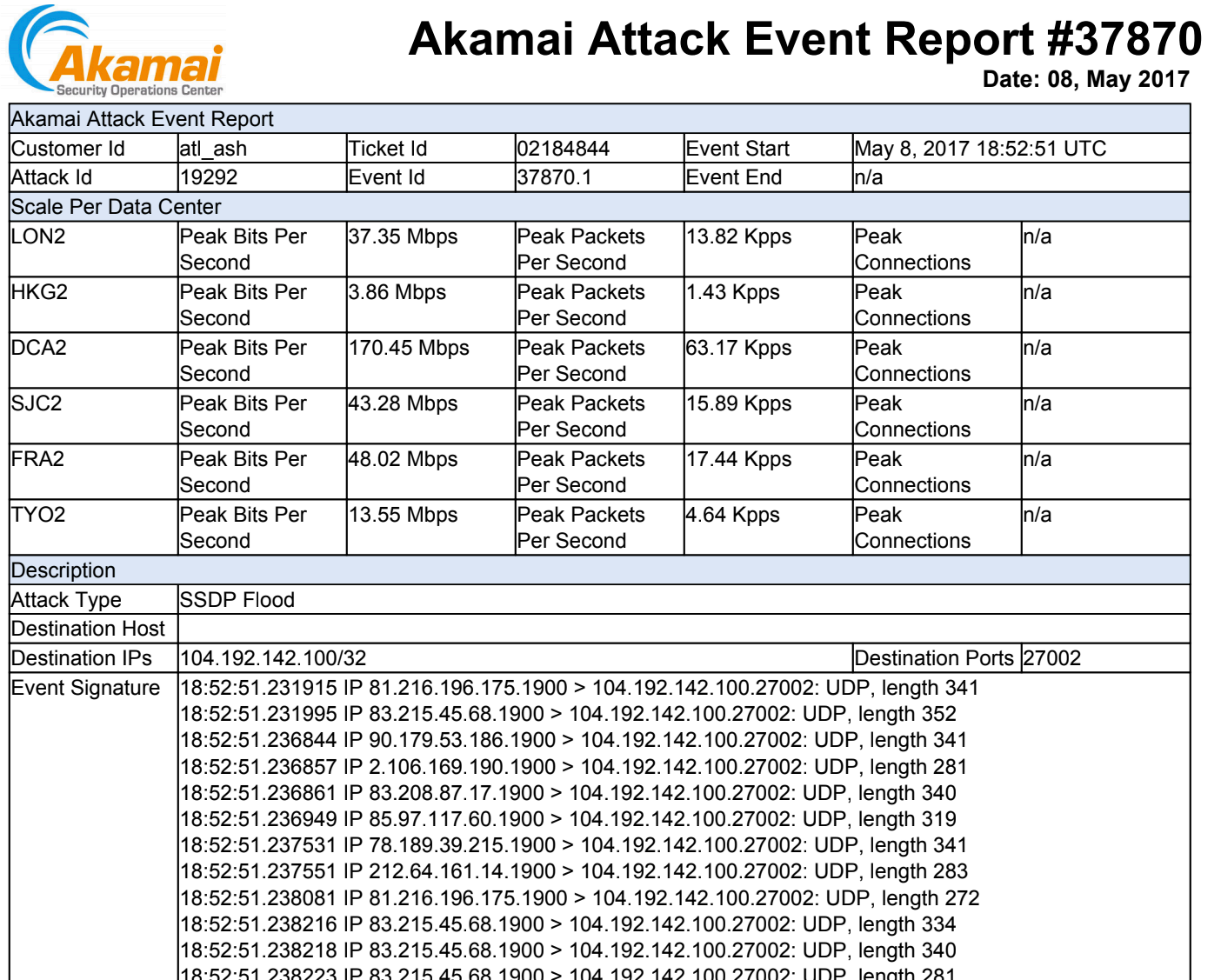


WHAT IS IN A NETWORK POP?

Networking

- Multiple Redundant 10GB+ AWS Direct Connect
- 10GB+ Public Connectivity (Tier 1)
- Links to BU or Components
- Managed via Ansible
- x86 Load Balancers
- Akamai Prolexic or AWS DDOS protection



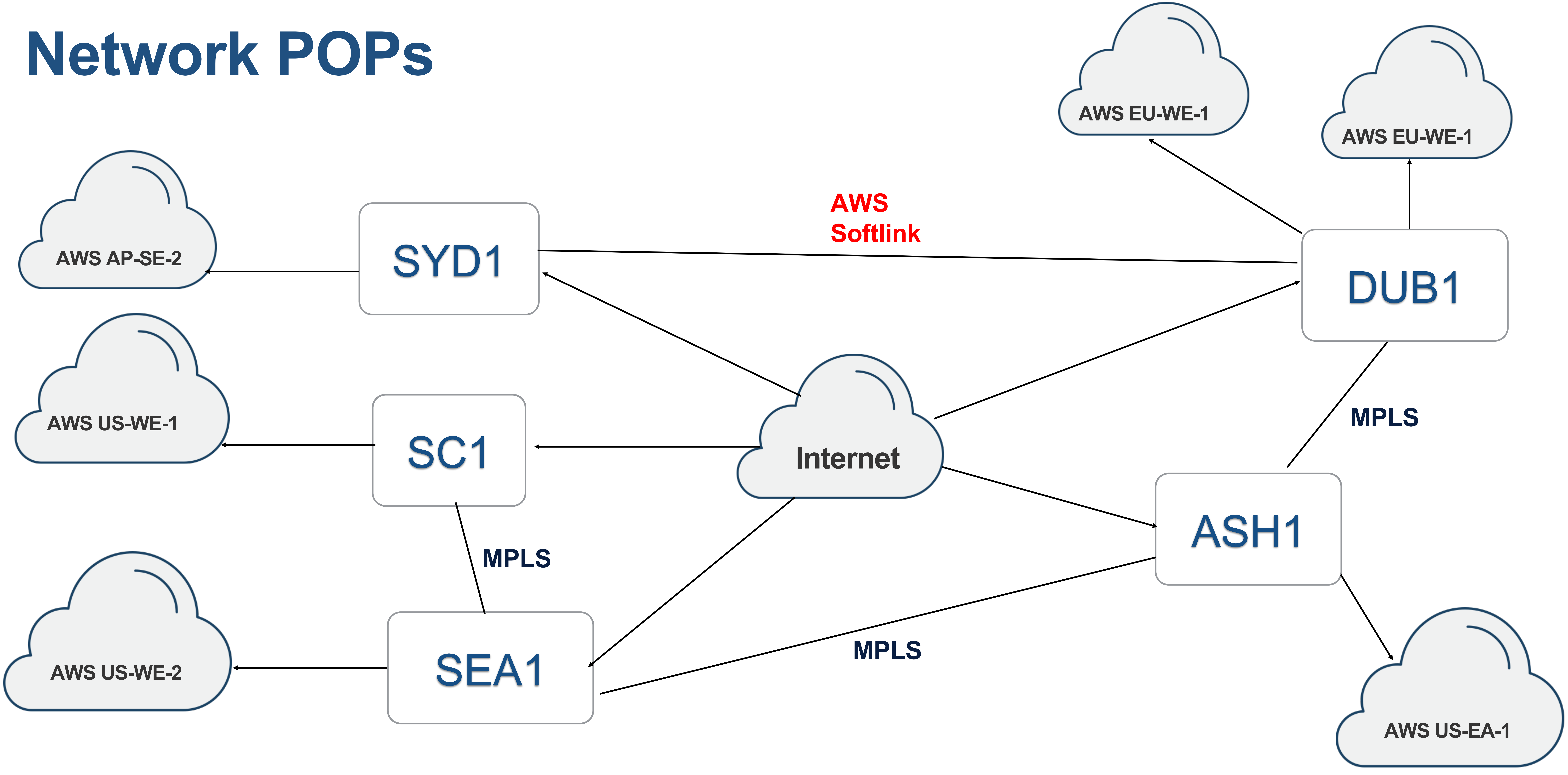


Akamai Attack Event Report #37870

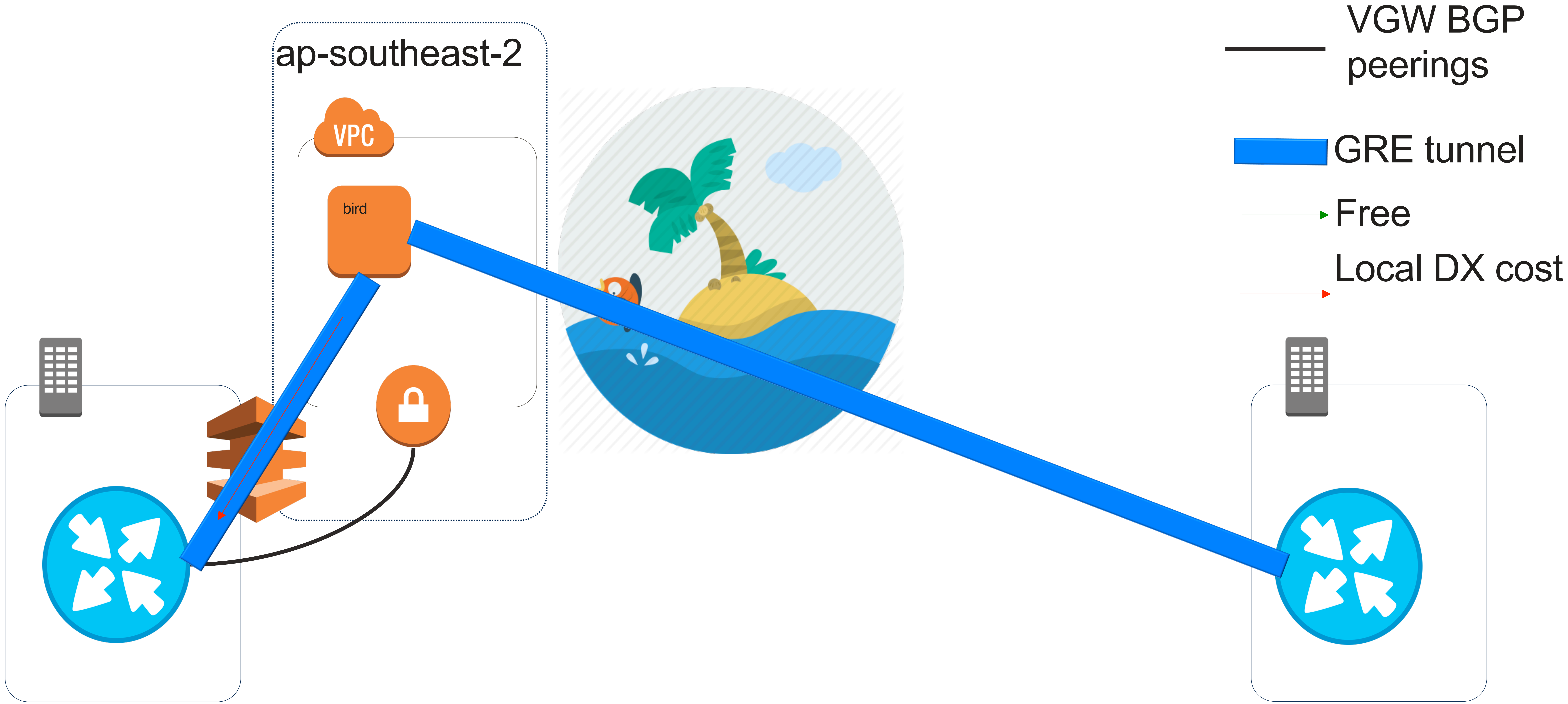
Date: 08, May 2017

Akamai Attack Event Report						
Customer Id	atl_ash	Ticket Id	02184844	Event Start	May 8, 2017 18:52:51 UTC	
Attack Id	19292	Event Id	37870.1	Event End	n/a	
Scale Per Data Center						
LON2	Peak Bits Per Second	37.35 Mbps	Peak Packets Per Second	13.82 Kpps	Peak Connections	n/a
HKG2	Peak Bits Per Second	3.86 Mbps	Peak Packets Per Second	1.43 Kpps	Peak Connections	n/a
DCA2	Peak Bits Per Second	170.45 Mbps	Peak Packets Per Second	63.17 Kpps	Peak Connections	n/a
SJC2	Peak Bits Per Second	43.28 Mbps	Peak Packets Per Second	15.89 Kpps	Peak Connections	n/a
FRA2	Peak Bits Per Second	48.02 Mbps	Peak Packets Per Second	17.44 Kpps	Peak Connections	n/a
TYO2	Peak Bits Per Second	13.55 Mbps	Peak Packets Per Second	4.64 Kpps	Peak Connections	n/a
Description						
Attack Type	SSDP Flood					
Destination Host						
Destination IPs	104.192.142.100/32				Destination Ports	27002
Event Signature	18:52:51.231915 IP 81.216.196.175.1900 > 104.192.142.100.27002: UDP, length 341 18:52:51.231995 IP 83.215.45.68.1900 > 104.192.142.100.27002: UDP, length 352 18:52:51.236844 IP 90.179.53.186.1900 > 104.192.142.100.27002: UDP, length 341 18:52:51.236857 IP 2.106.169.190.1900 > 104.192.142.100.27002: UDP, length 281 18:52:51.236861 IP 83.208.87.17.1900 > 104.192.142.100.27002: UDP, length 340 18:52:51.236949 IP 85.97.117.60.1900 > 104.192.142.100.27002: UDP, length 319 18:52:51.237531 IP 78.189.39.215.1900 > 104.192.142.100.27002: UDP, length 341 18:52:51.237551 IP 212.64.161.14.1900 > 104.192.142.100.27002: UDP, length 283 18:52:51.238081 IP 81.216.196.175.1900 > 104.192.142.100.27002: UDP, length 272 18:52:51.238216 IP 83.215.45.68.1900 > 104.192.142.100.27002: UDP, length 334 18:52:51.238218 IP 83.215.45.68.1900 > 104.192.142.100.27002: UDP, length 340 18:52:51.238223 IP 83.215.45.68.1900 > 104.192.142.100.27002: UDP, length 281						

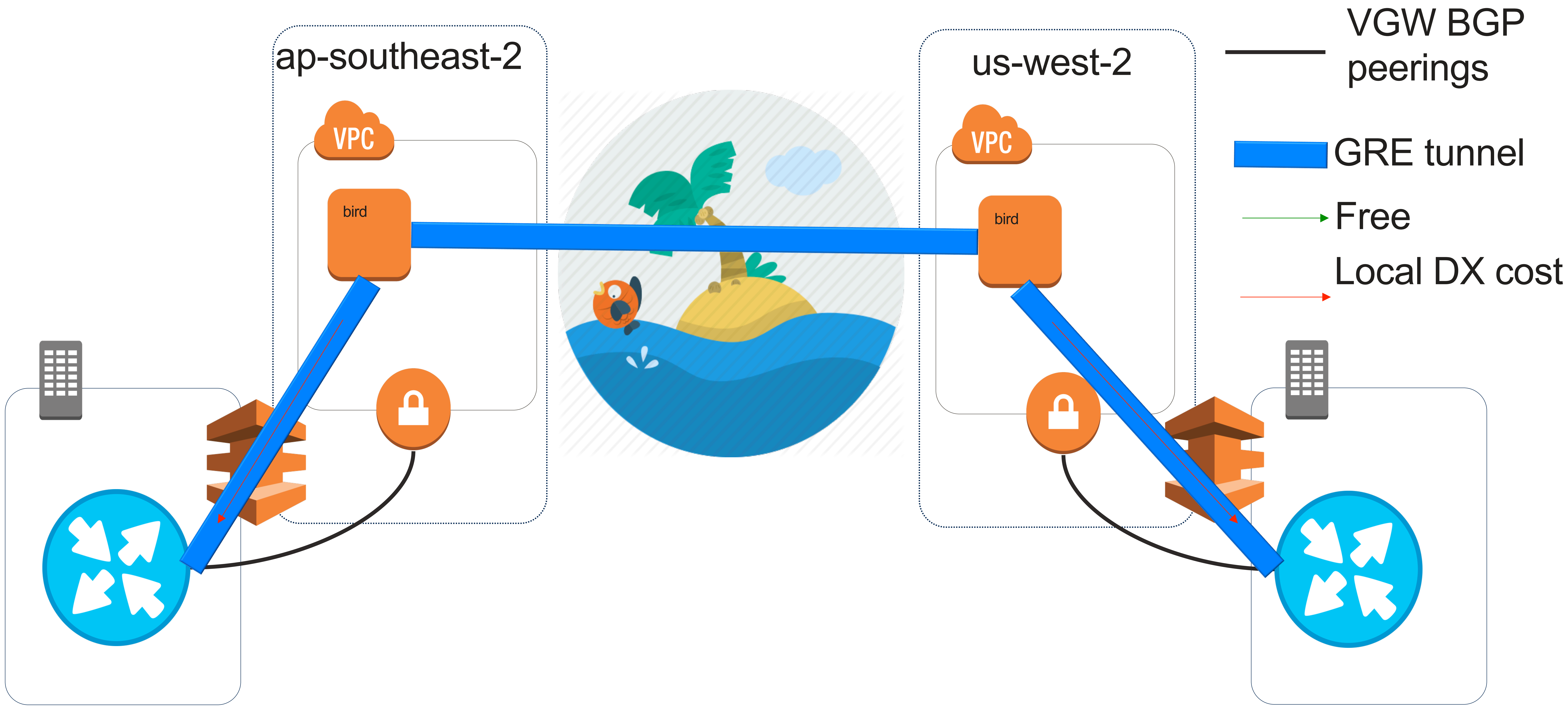
Network POPs



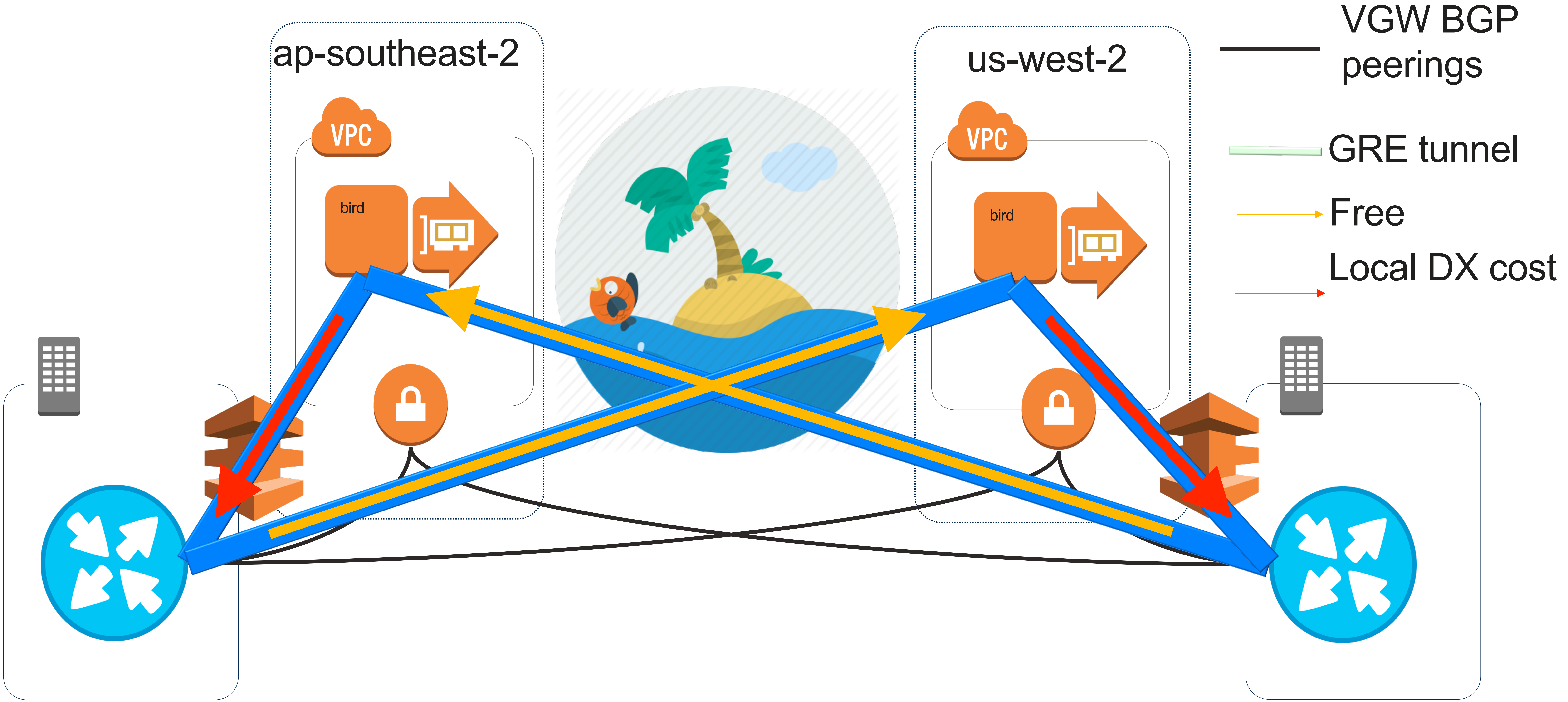
Softlink – Exposed to the internet



Softlink – paying VPC peering or EIP fees



Softlink - low cost backhaul via Cloud Provider



Softlink - low cost backhaul via Cloud Provider

Infra cost (1 month): $211

- 1x r4.large (Sydney): $115.2
- 1x r4.large (Oregon): $95.76

Per GB cost (on-prem to on-prem)

- Sydney to Oregon: $0.02/GB
- Oregon to Sydney: $0.042/GB

Assumptions:

- **Direct Connects are a sunk cost** ($3240/month for 2x10Gbps)
- Control-plane traffic and protocol overhead are negligible
- EBS costs are negligible

Performance - Sydney to Oregon:

```
root@ip-10-117-12-6:/home/ubuntu# iperf3 -c 10.104.9.36 -P 20 -M 
-t 30
(...)
- - - - - - - - - - - - - - - - - - - - - - - - -
[  4]  24.00-25.00  sec  7.50 MBytes  62.9 Mbits/sec   0   1.32
MBytes
[  6]  24.00-25.00  sec  8.75 MBytes  73.4 Mbits/sec   0   2.12
MBytes
[  8]  24.00-25.00  sec  10.0 MBytes  83.9 Mbits/sec   0   2.13
MBytes
(...)
[SUM]  24.00-25.00  sec   171 MBytes  1.44 Gbits/sec
-- - - - - - - - - - - - - - - - - - - - - - - - -
```

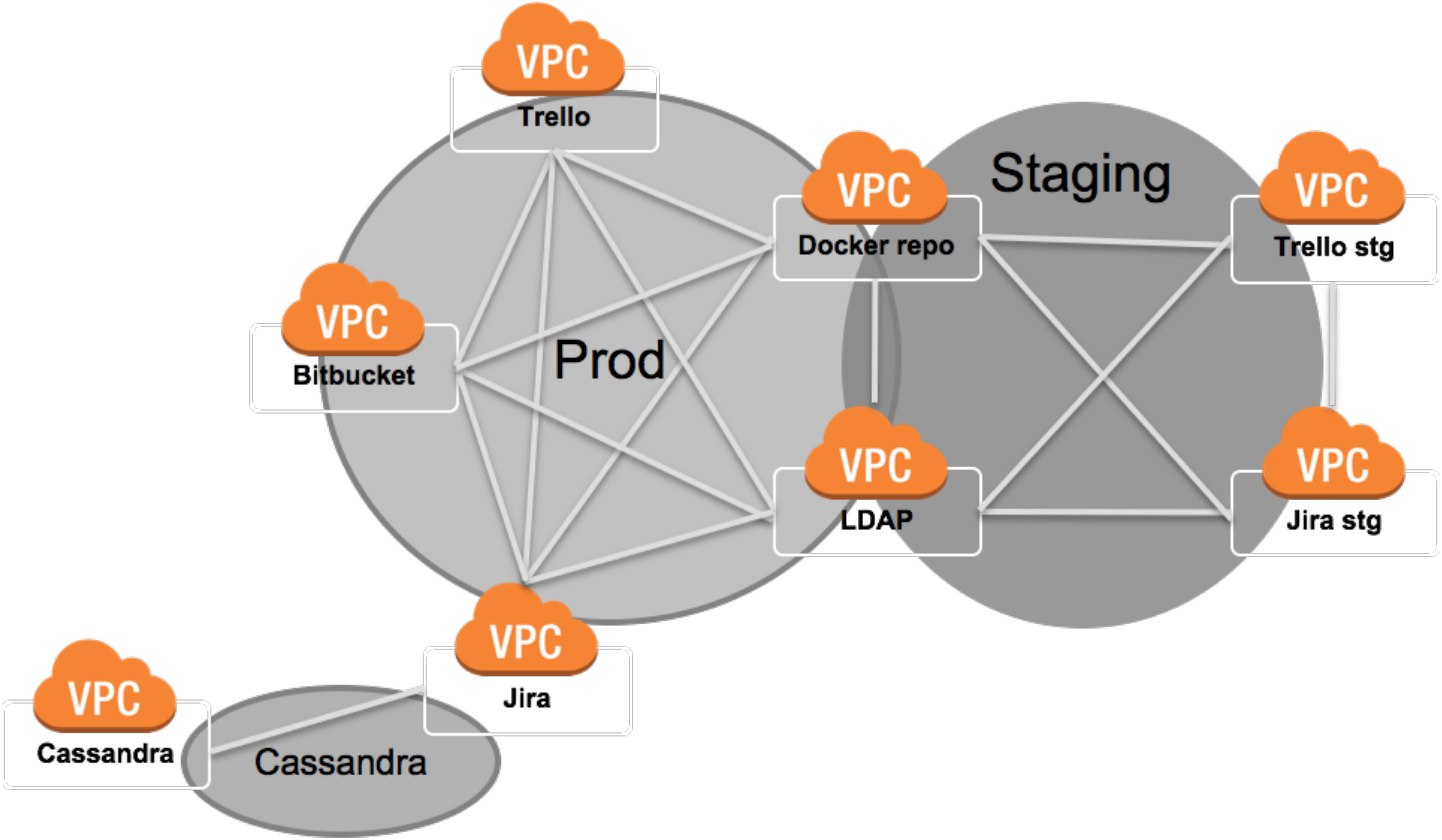
Performance - Oregon to Sydney:

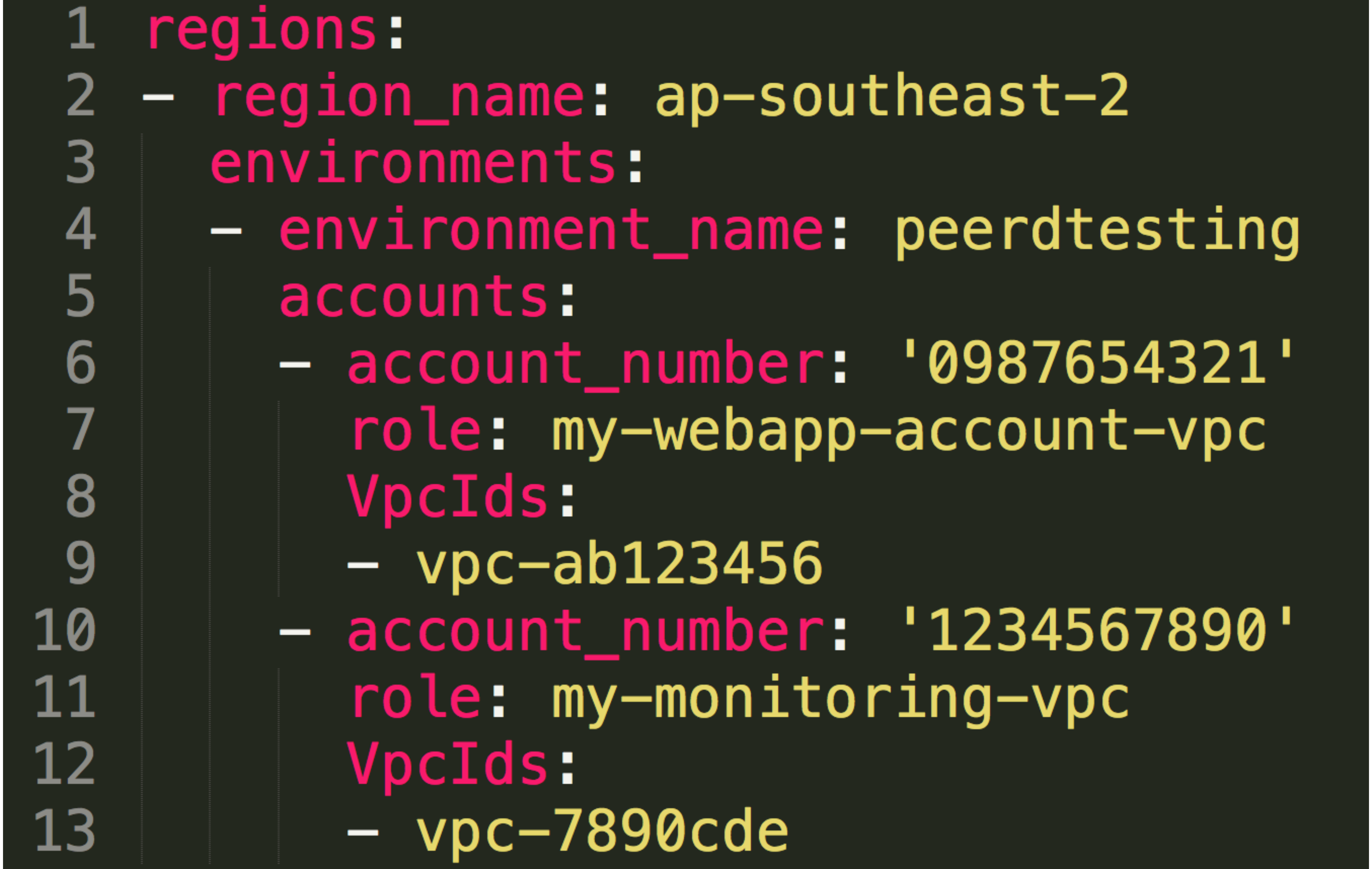
```
root@ip-10-104-9-36:/home/ubuntu# iperf3 -c  10.117.12.6 -M 1440 -
(...)
- - - - - - - - - - - - - - - - - - - - - - - - -
[  4]   4.00-5.00   sec  8.75 MBytes  73.4 Mbits/sec   0   1.52
MBytes
[  6]   4.00-5.00   sec  8.75 MBytes  73.4 Mbits/sec   0   3.04
MBytes
(...)
[SUM]   4.00-5.00   sec   178 MBytes  1.49 Gbits/sec
-------------------------
```

peerd

Patent pending… 15/788,229 ;)

Will be open sourced on Bitbucket





```yaml
regions:
- region_name: ap-southeast-2
  environments:
  - environment_name: peerdtesting
    accounts:
    - account_number: '0987654321'
      role: my-webapp-account-vpc
      VpcIds:
      - vpc-ab123456
    - account_number: '1234567890'
      role: my-monitoring-vpc
      VpcIds:
      - vpc-7890cde
```

peerd - Workflow

1. Describe the state of the mesh using Cloud Provider API
2. Compute the desired state and difference from current state
3. Work across accounts and regions to implement desired state

peerd - Benefit

1. No hair-pinning on your own private network
2. Routing in the cloud, connectivity to everything
3. Reduced management overhead

Network Data

Before (pretty good)

Gigs of logs going into Splunk
Auto-discovery of links and devices
Nagios/SNMP triggering PagerDuty
Loads of Custom Checks
sFlow and NetFlow
MTR and hping
Grafana graphs (so many!)
Pingdom

.. but missing actual end-to-end distributed visibility

Before (pretty good)

Gigs of logs going into Splunk
Auto-discovery of links and devices
Nagios/SNMP triggering PagerDuty
Loads of Custom Checks
sFlow and NetFlow
MTR and hping
Grafana graphs (so many!)
Pingdom

.. but missing actual end-to-end distributed visibility

After (better)

40+ probes
Loss or latency triggers PagerDuty
We open Datadog
Correlate and (most cases) identify immediate problem
Fix the issue or notify the vendor

Vendors and Providers still an issue...

https://github.com/prometheus/blackbox_exporter
https://github.com/fbsamples/OpenNetNorad
Thousandeyes (paid solution)

Monitoring all-the-things!

- Offices
- On x86 in our cages / PoPs
- VPCs in AWS (one per Availability Zone)
- Using puppet, ansible or containers - depending on the environment

- IPsec tunnels
- MPLS Backbone
- Devices
- Offices
- Internet Links + Load Balancers
- VPCs and AWS indirectly

neteng.te.loss

Show 2 d May 7, 7:11PM – May 10, 1:08AM

120
115
110
105
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
agent:micros-prd-probes-useast1-a, test:te.agent-to-server.vip-01-ne-vtm-syd-canary
80.2
21:00
Tue 8
03:00
06:00
09:00
12:00
15:00
18:00
21:00
Wed 9
03:00
05/09 05:20
09:00
12:00
15:00
18:00
21:00
Thu 10

Search events to overlay...

Show 105 m Nov 16, 2:10PM – Nov 16, 3:55PM

UTC

[Outbound] Max of NetEng loss

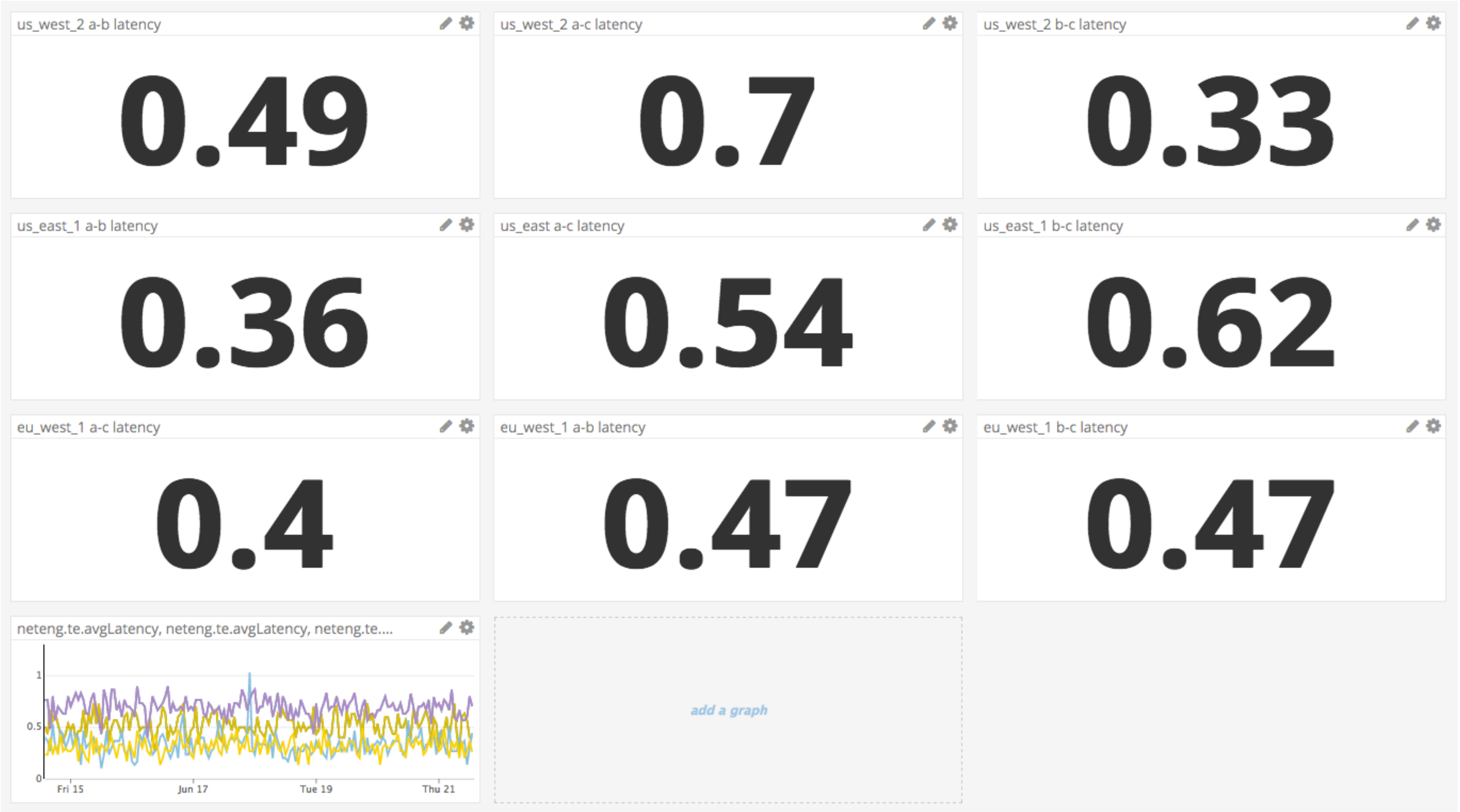
120
100
80
60
40
20
0
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:50
14:55
15:00
15:05
15:10
15:15
15:20
15:25
15:30
15:35
15:40
15:45
15:50
15:5
100
agent:micros-prd-probes-useast1-b

🟦	28	Avg: 13.21	neteng.te.loss	{agent:micros-prd-probes-useast1-a}
🟪	13	Avg: 39.36	neteng.te.loss	{agent:micros-prd-probes-useast1-b}
🟨	3	Avg: 3.15	neteng.te.loss	{agent:micros-prd-probes-useast1-c}

[Outbound] Avg of NetEng loss by destination

120
100
80
60
40
20
0
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:50
14:55
15:00
15:05
15:10
15:15
15:20
15:25
15:30
15:35
15:40
15:45
15:50
15:5

🟦	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.be-probes-useast1}
🟪	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.micros-prd-probes-apse2-a}
🟨	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.micros-prd-probes-useast1-b}
🟦	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.micros-prd-probes-useast1-c}
🟪	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.micros-prd-probes-uswest1-a}
🟨	0	Avg: 0	neteng.te.loss	{agent:micros-prd-probes-useast1-a,test:te.agent-to-agent.micros-prd-probes-uswest1-b}



us_west_2 a-b latency

0.49

us_west_2 a-c latency

0.7

us_west_2 b-c latency

0.33

us_east_1 a-b latency

0.36

us_east a-c latency

0.54

us_east_1 b-c latency

0.62

eu_west_1 a-c latency

0.4

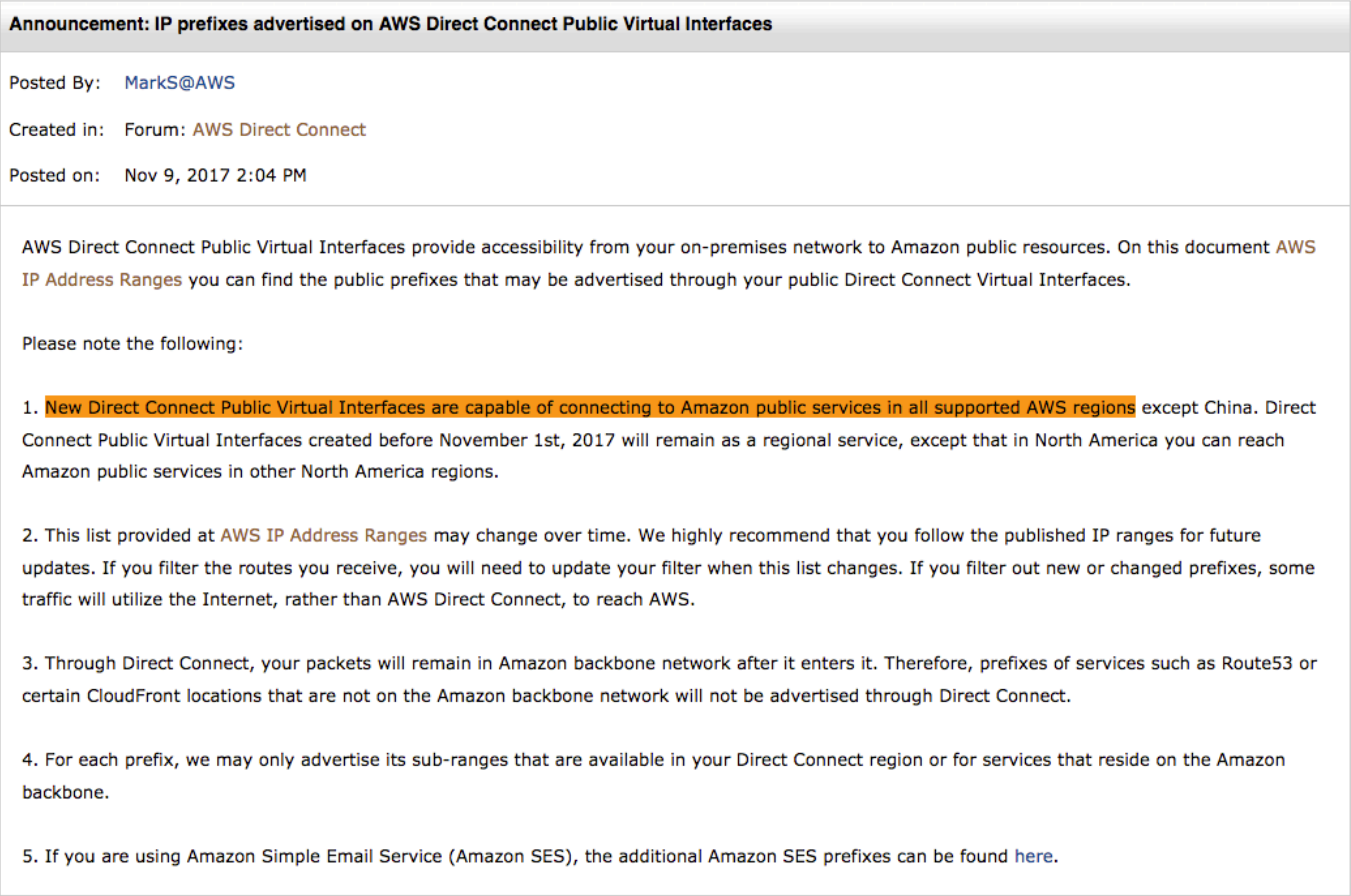
eu_west_1 a-b latency

0.47

eu_west_1 b-c latency

0.47

neteng.te.avgLatency, neteng.te.avgLatency, neteng.te....



Announcement: IP prefixes advertised on AWS Direct Connect Public Virtual Interfaces

Posted By: MarkS@AWS

Created in: Forum: AWS Direct Connect

Posted on: Nov 9, 2017 2:04 PM

AWS Direct Connect Public Virtual Interfaces provide accessibility from your on-premises network to Amazon public resources. On this document AWS IP Address Ranges you can find the public prefixes that may be advertised through your public Direct Connect Virtual Interfaces.

Please note the following:

1. New Direct Connect Public Virtual Interfaces are capable of connecting to Amazon public services in all supported AWS regions except China. Direct Connect Public Virtual Interfaces created before November 1st, 2017 will remain as a regional service, except that in North America you can reach Amazon public services in other North America regions.

2. This list provided at AWS IP Address Ranges may change over time. We highly recommend that you follow the published IP ranges for future updates. If you filter the routes you receive, you will need to update your filter when this list changes. If you filter out new or changed prefixes, some traffic will utilize the Internet, rather than AWS Direct Connect, to reach AWS.

3. Through Direct Connect, your packets will remain in Amazon backbone network after it enters it. Therefore, prefixes of services such as Route53 or certain CloudFront locations that are not on the Amazon backbone network will not be advertised through Direct Connect.

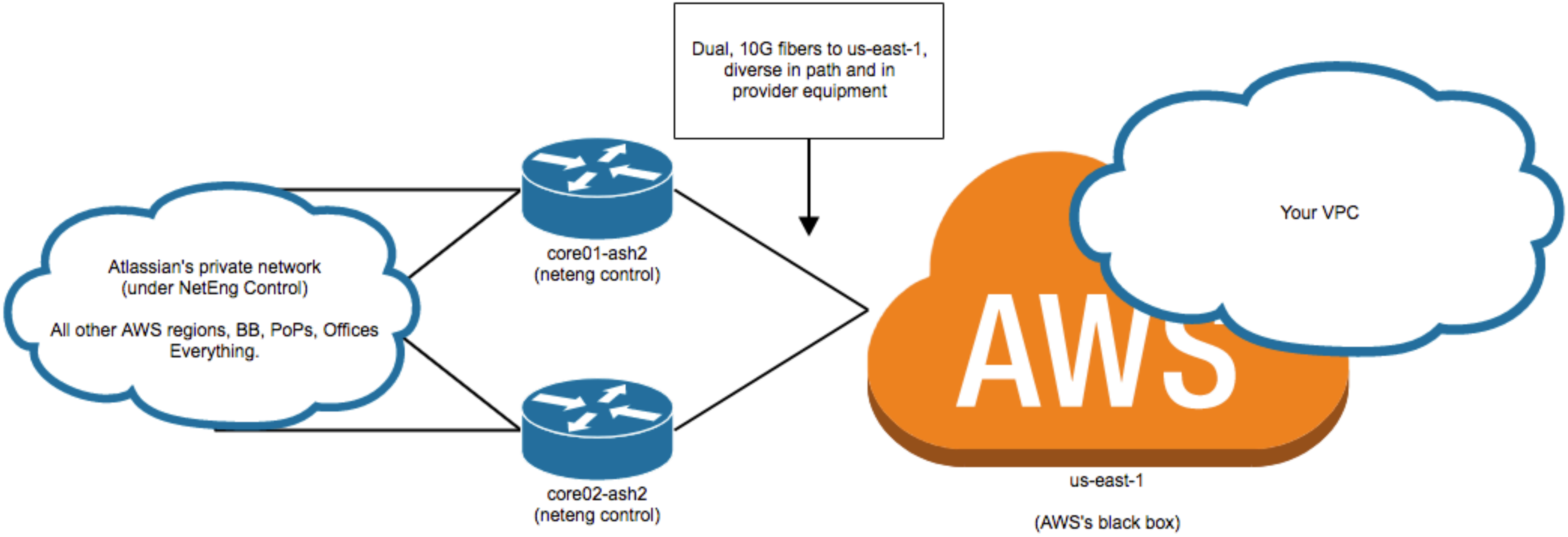
4. For each prefix, we may only advertise its sub-ranges that are available in your Direct Connect region or for services that reside on the Amazon backbone.

5. If you are using Amazon Simple Email Service (Amazon SES), the additional Amazon SES prefixes can be found here.

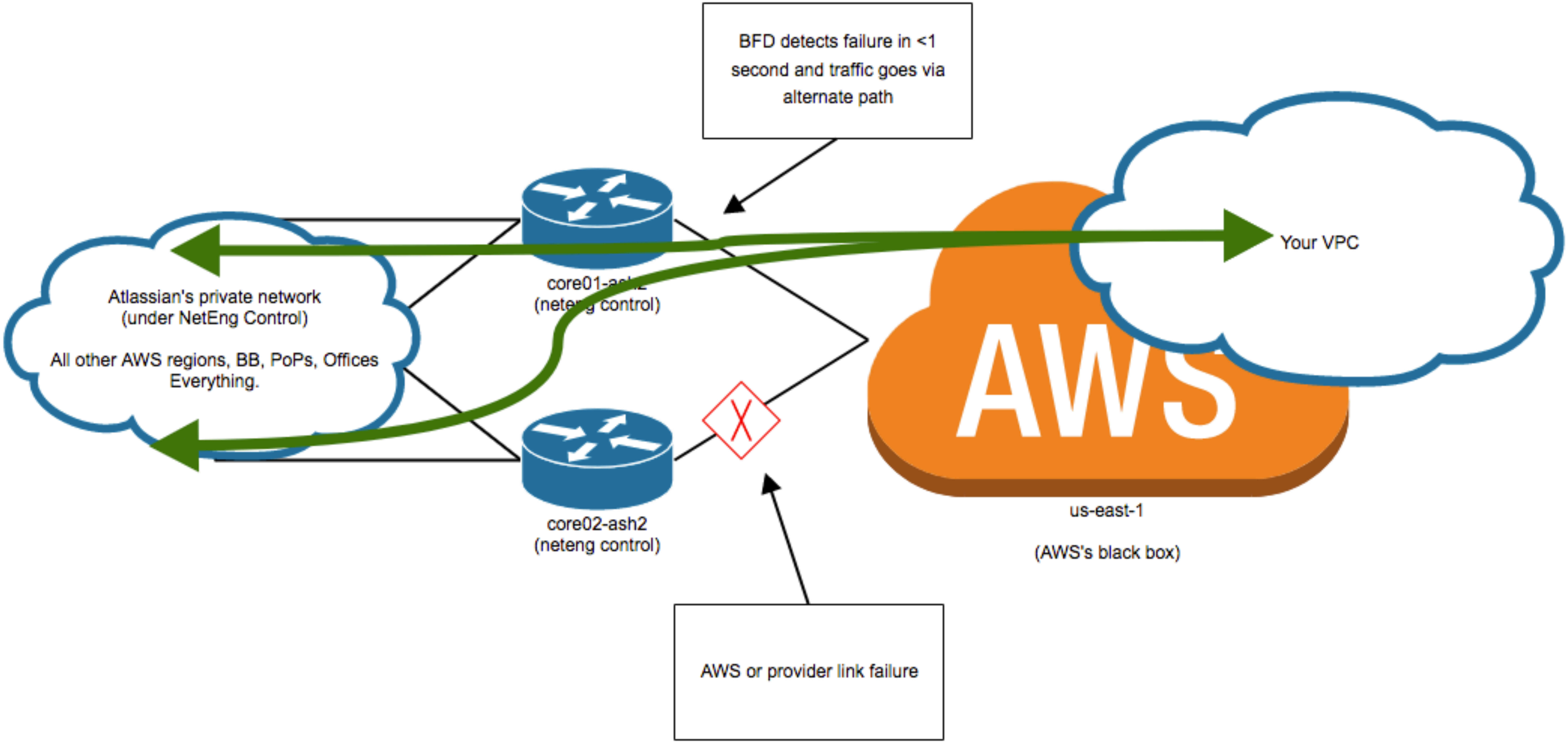
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
18
Wed 3
Fri 5
Jan 7
Tue 9
Thu 11
Sat 13
Mon 15
Wed 17
Fri 19
Jan 21
Tue 23
Thu 25
Sat 27
Mon 29
Wed 31
February
Sat 3
Mon 5
Wed 7
Fri 9

War Stories

The setup


Dual, 10G fibers to us-east-1, diverse in path and in provider equipment
Atlassian's private network (under NetEng Control)
All other AWS regions, BB, PoPs, Offices Everything.
core01-ash2 (neteng control)
core02-ash2 (neteng control)
AWS
Your VPC
us-east-1
(AWS's black box)

The expected behaviour



BFD detects failure in <1 second and traffic goes via alternate path

Atlassian's private network (under NetEng Control)

All other AWS regions, BB, PoPs, Offices Everything.

core01-ash2 (neteng control)

core02-ash2 (neteng control)

Your VPC

us-east-1

(AWS's black box)

AWS or provider link failure

Metric: Availability

Agent: All agents

24h
7d
14d
Average Availability
100%
49%
03:00
03:30
04:00
04:30
05:00
05:30
06:00
Aug 14
Aug 17
Aug 20
Aug 23

Path Visualization

Showing: **1 of 9 Agents** (Show All) □ Node labels

Grouping: Interfaces by **IP Address**

Highlighting: **Link Delay > 100 ms** (0 links) **Forwarding Loss >10%** (2 nodes)

Selecting: **1 node** **Quick Selection** (1 info) Deselect All

Highlight nodes that match all / any

Search on Network, Country, Address, Prefix, or Title...

Loss Frequency Low

Avg. Response 2 ms

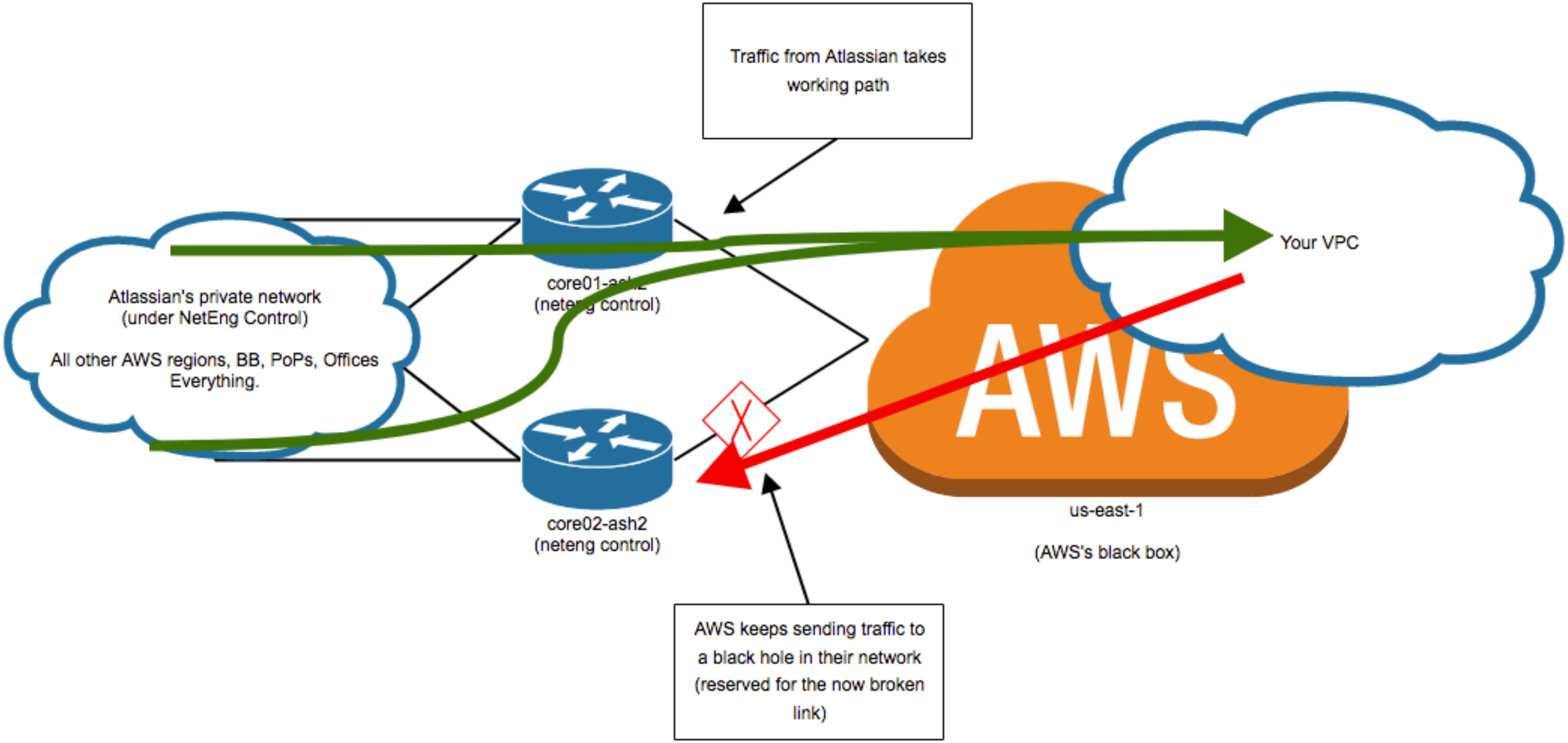
Tests using this interface across all ThousandEyes customers

🟥 100% affected (2 total, 1 in this account group)

🟦 0% not affected

Show only agents using this node

What was actually happening



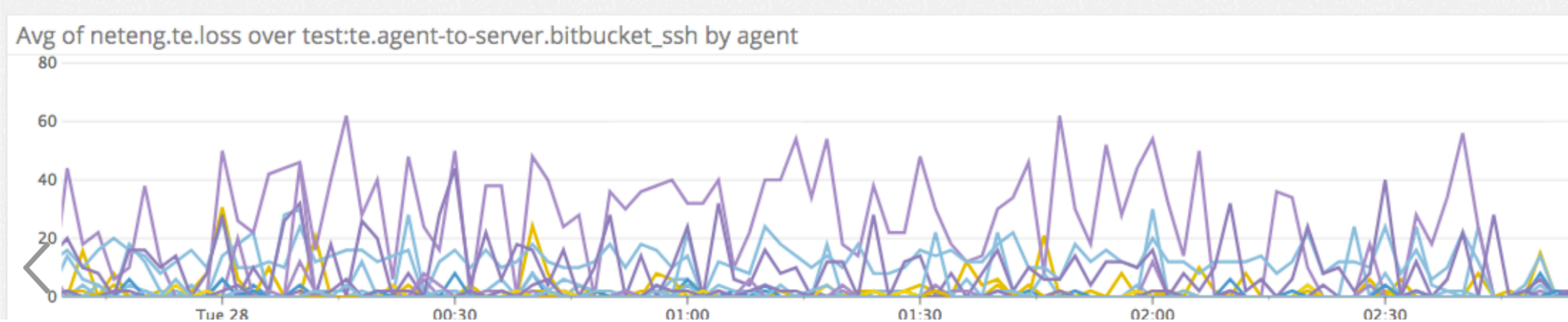
Traffic from Atlassian takes working path

AWS keeps sending traffic to a black hole in their network (reserved for the now broken link)

TCP timeouts from Bitbucket

```
SSH proxy tried to connect to [git@bitbucket.org/
104.192.143.1:22]: and failed due to the following error:
[Authenticating remote session failed]: Connection reset by
peer] fatal: Could not read from remote repository.
```

TCP timeouts from Bitbucket

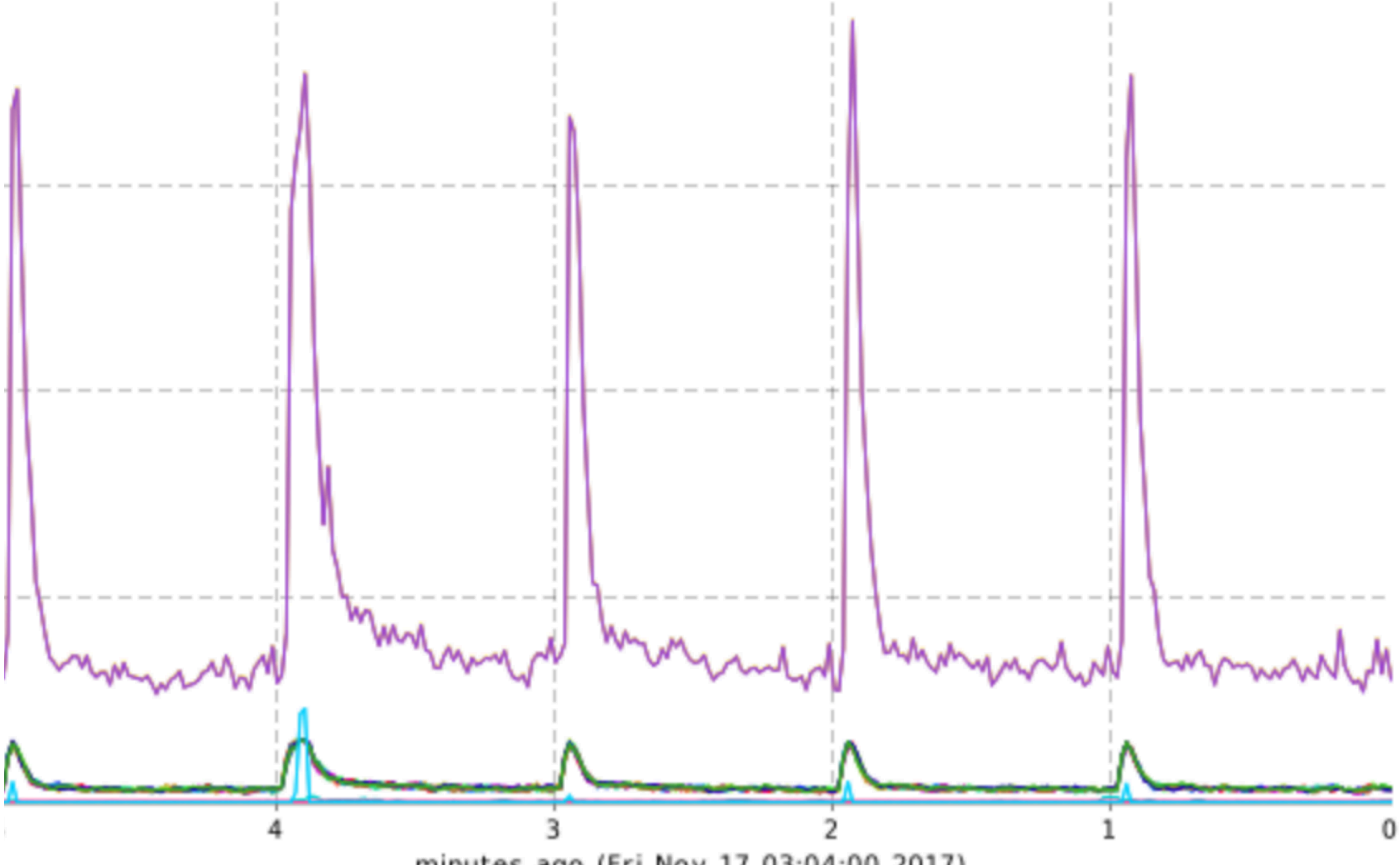


TCP timeouts from Bitbucket

```
Session ID: 65193,
Policy name: internet_shell/60,
State: Active, Timeout: 8, Valid
In: /37884 --> 104.192.143.1/22;tcp, Conn Tag: 0x0, If:, Pkts: 1, Bytes: 56,
Out: 104.192.143.1/22 --> 104.192.139.x/57887;tcp, Conn Tag: 0x0, If:, Pkts: 0,
Bytes: 0,
```

04-Dec-2017 15:00:04 UTC
30-Nov-2017 16:00:05 UTC
29-Nov-2017 16:00:04 UTC
29-Nov-2017 14:00:04 UTC

TCP timeouts from Bitbucket



TCP timeouts from Bitbucket

```
while true; do netstat -ant|grep SYN_RECV|wc -l; done
netstat –statistics

watch -n 5 "nstat | grep Listen"
TcpExtListenOverflows           6662               0.0
TcpExtListenDrops               11580

ss -lti '( sport = :22 )'
LISTEN     129    128         104.192.143.1:ssh                      :
          rto:1000 mss:536 cwnd:10 unacked:128
LISTEN     117    128         104.192.143.1:ssh                      :
          rto:1000 mss:536 cwnd:10 unacked:117
LISTEN     129    128         104.192.143.1:ssh                      :
          rto:1000 mss:536 cwnd:10 unacked:128
```

“the TCP implementation will simply drop the SYN packet” - section 14.5, *listen Backlog Queue* in W. Richard Stevens’ textbook *TCP/IP Illustrated, Volume 3*.

TCP timeouts from Bitbucket

```
net.ipv4.tcp_max_syn_backlog=65535 ( depends on next setting)
net.core.somaxconn=65535
net.core.rmem_max=16777216
net.core.wmem_max=16777216
net.ipv4.tcp_rmem=4096 87380 16777216
net.ipv4.tcp_wmem=4096 65535 16777216
net.ipv4.tcp_slow_start_after_idle=0
net.ipv4.tcp_fastopen=3
net.core.default_qdisc=fq
net.ipv4.tcp_congestion_control=bbr
```

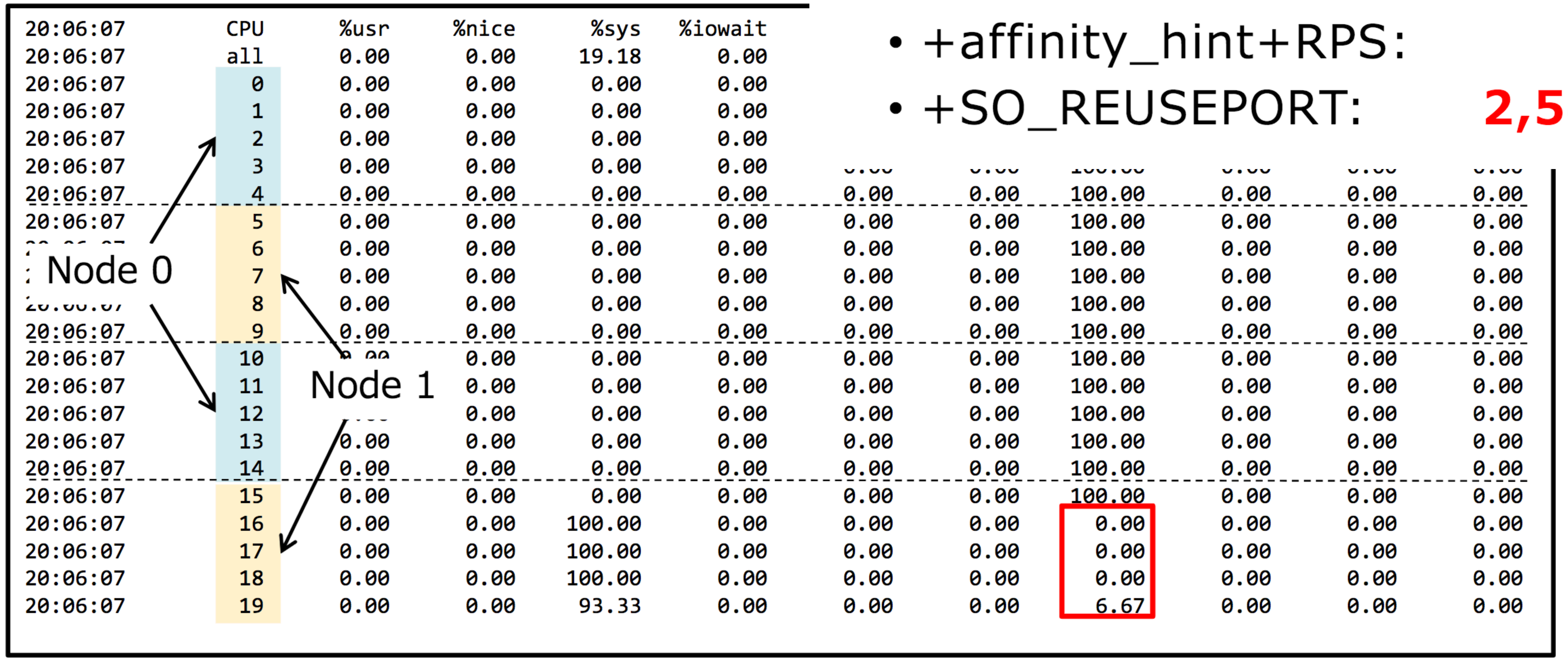
https://www.techrepublic.com/article/how-to-enable-tcp-bbr-to-improve-network-speed-on-linux/
https://help.hostunmetered.net/tutorials/tuning-your-os-for-10-gbps-network
https://blog.apnic.net/2017/05/09/bbr-new-kid-tcp-block/
https://wiki.mikejung.biz/Sysctl_tweaks#net.ipv4.tcp_slow_start_after_idle
http://www.lognormal.com/blog/2012/09/27/linux-tcpip-tuning/
https://bradleyf.id.au/nix/shaving-your-rtt-wth-tfo/

TCP timeouts from Bitbucket – Final Word

Boost Network Transaction Performance - Toshiaki Makita (great talk!)

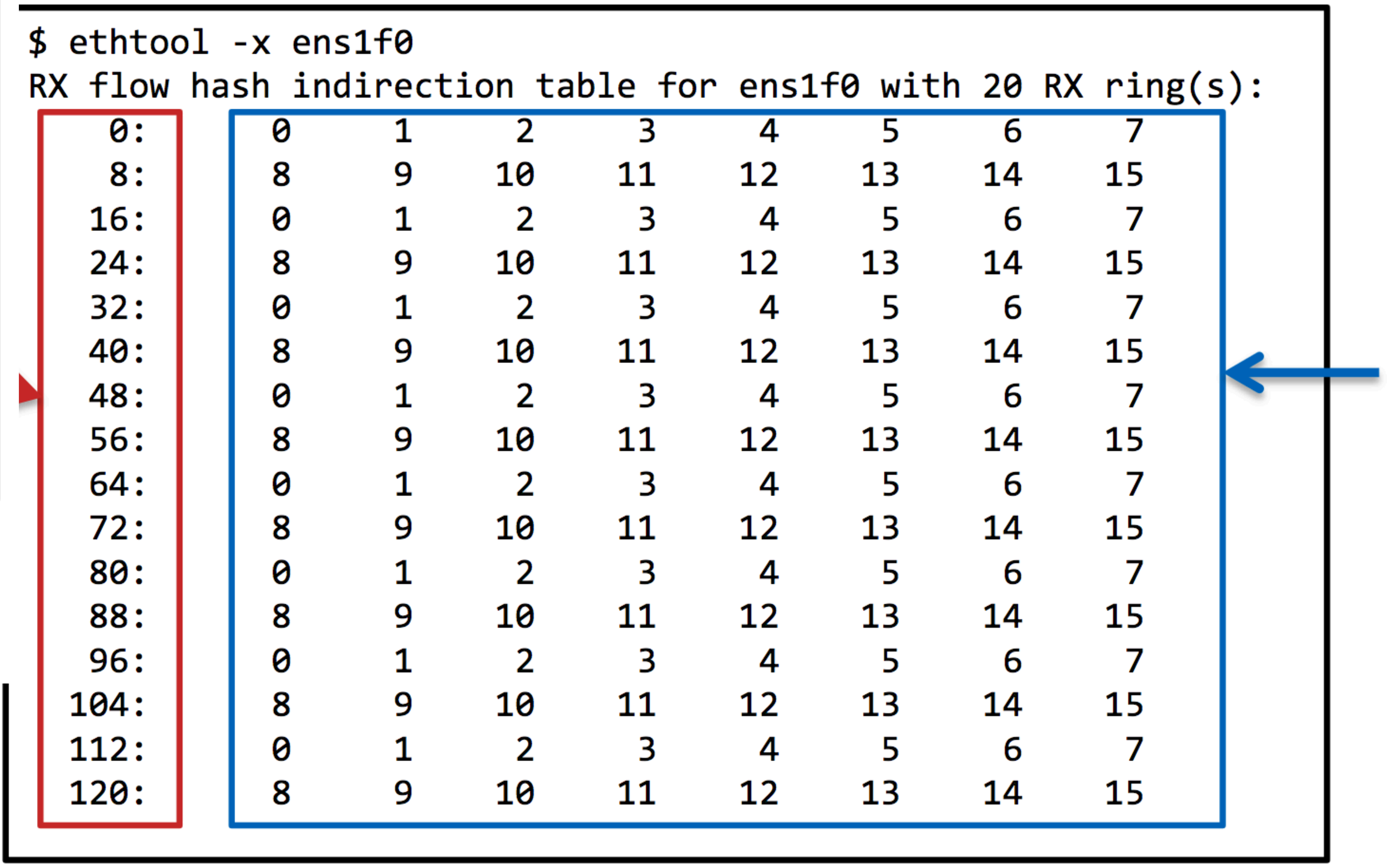
- **Perfomance change**
  - RSS: 270,000 tps (approx. 360Mbps)
  - +affinity_hint+RPS: 17,000 tps (approx. 23Mbps)
  - +SO_REUSEPORT: **2,540,000** tps (approx. **3370Mbps**)

- **sar -u ALL -P ALL 1**



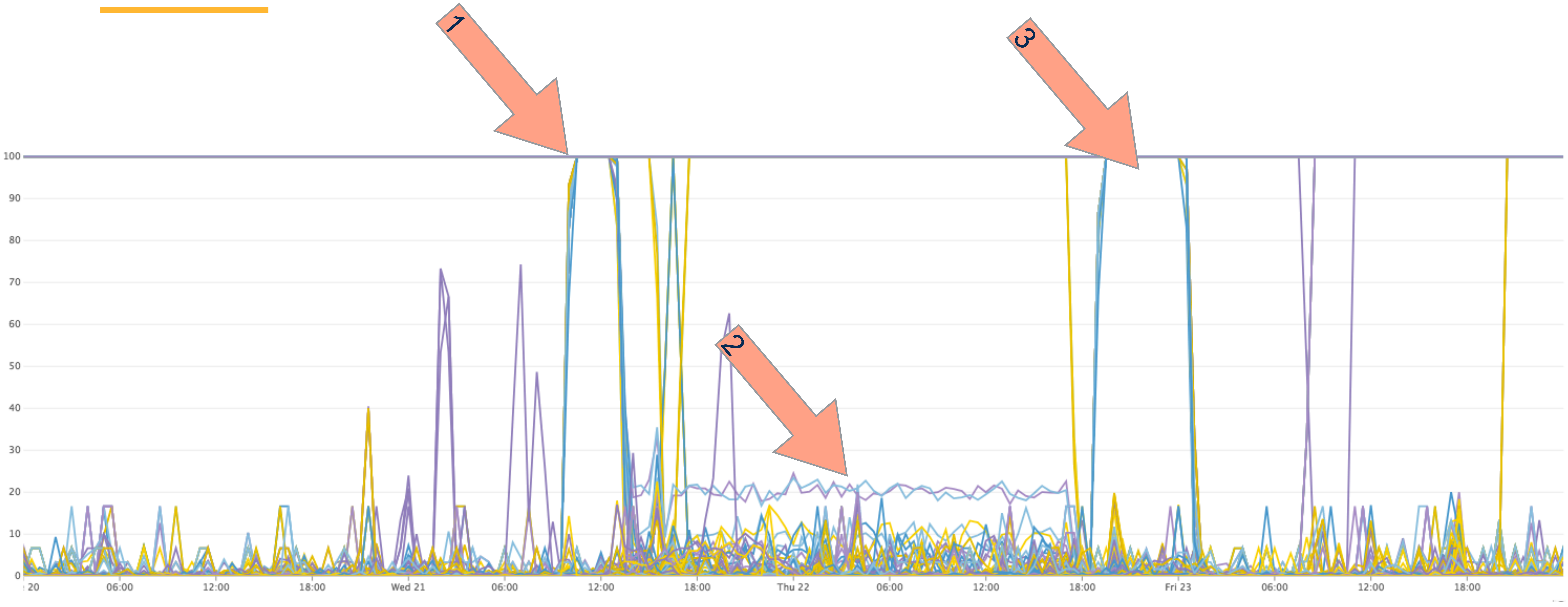
```
20:06:07      CPU    %usr    %nice    %sys   %iowait
20:06:07      all    0.00    0.00    19.18    0.00
20:06:07        0    0.00    0.00     0.00    0.00
20:06:07        1    0.00    0.00     0.00    0.00
20:06:07        2    0.00    0.00     0.00    0.00
20:06:07        3    0.00    0.00     0.00    0.00
20:06:07        4    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07        5    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07        6    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07        7    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07        8    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07        9    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       10    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       11            0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       12    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       13    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       14    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       15    0.00    0.00     0.00    0.00    0.00    0.00   100.00    0.00    0.00    0.00
20:06:07       16    0.00    0.00   100.00    0.00    0.00    0.00     0.00    0.00    0.00    0.00
20:06:07       17    0.00    0.00   100.00    0.00    0.00    0.00     0.00    0.00    0.00    0.00
20:06:07       18    0.00    0.00   100.00    0.00    0.00    0.00     0.00    0.00    0.00    0.00
20:06:07       19    0.00    0.00    93.33    0.00    0.00    0.00     6.67    0.00    0.00    0.00
```

- **Though irqs looks distributed evenly, core 16-19 are not used for softirq…**



```
$ ethtool -x ens1f0
RX flow hash indirection table for ens1f0 with 20 RX ring(s):
    0:      0      1      2      3      4      5      6      7
    8:      8      9     10     11     12     13     14     15
   16:      0      1      2      3      4      5      6      7
   24:      8      9     10     11     12     13     14     15
   32:      0      1      2      3      4      5      6      7
   40:      8      9     10     11     12     13     14     15
   48:      0      1      2      3      4      5      6      7
   56:      8      9     10     11     12     13     14     15
   64:      0      1      2      3      4      5      6      7
   72:      8      9     10     11     12     13     14     15
   80:      0      1      2      3      4      5      6      7
   88:      8      9     10     11     12     13     14     15
   96:      0      1      2      3      4      5      6      7
  104:      8      9     10     11     12     13     14     15
  112:      0      1      2      3      4      5      6      7
  120:      8      9     10     11     12     13     14     15
```

AWS trying out new things in eu-central-1?



AWS trying out new things in eu-central-1?

1
3
100
90
80
70
60
50
40
30
20
10
0
20
06:00
12:00
18
06:00
12:00
18:00
There is no cloud
it's just someone else's computer

Thank you!

BENJAMIN MCALARY | PRINCIPLE NETWORK ENGINEER
BMCALARY@ATLASSIAN.COM