BGP Routing for Large Scale Data Centers

Goal: Informational RFC
Agenda

Design Requirements
Network Design
Why BGP over IGP
Details and design choices
Feature Standardization?
Design Requirements
Online Service DC Specifics

Server Perspective

100’s thousands of servers
10G NICs

Distributed Applications

Aware of the network
Explicit parallelism
Example: Web Index computation

“Network as a computer” concept
Two types of traffic flows
  Query
  Background

Query
  Latency Sensitive
  Partition/Aggregate

Background
  East/West
  Compute & Synchronize
Design Requirements

REQ1: Build upon a topology providing horizontal bandwidth scalability

REQ2: Minimize feature/protocol set

REQ3: Select simplest most common protocols

REQ4: Protocol must support “some” traffic engineering
Network Design
Topology choice: Clos

Multiple definitions exist...

Has $N$ stages ($N=3,5,7,..$)

- Folded on diagram

Full bisection bandwidth if $M \geq N$

Natural link load-balancing

- ECMP Based – implements Valiant Load Balancing
Scaling Clos Topology

Think multiple parallel Clos topologies
Lower port density on switches

Horizontal capacity scaling at every layer above ToR
Routing Design for Parallel Clos

BGP all the way down to the ToR (eBGP)
Separate BGP ASN per ToR
Design Specifics: Default Routing

Don’t use “default route only” model

Don’t hide specific prefixes

Otherwise: Route Black-Holing on link failure!
Design Specifics: Route Summarization
Don’t summarize server subnets!

Summarizing P2P links is OK

Otherwise: Route Black-Holing on link failure!
Why BGP over IGP
BGP Simplicity

Simpler protocol design concepts compared to IGPs

Better vendor interoperability
Less state-machines, data-structures etc

BGP allows for per-hop traffic engineering

This way we can inject prefixes at any layer
Used for unequal-cost Anycast load-balancing solution
BGP Simplicity

Troubleshooting BGP is simpler

- BGP RIB structure is simpler compared to link-state LSDB
- Clear picture of what sent where (RIBIn, RIBOut)

Event propagation is more constrained in BGP

- E.g. link failures have limited propagation scope
- More stability due to reduced event “flooding” domains
- Hard to achieve the same using areas in IGPs
Common arguments against BGP

What about configuration complexity – BGP neighbors, etc?

   Not a problem with automated configuration generation

What about convergence properties?

   Is not our primary goal anyways, few seconds are OK
   Practical convergence in less than a second (Fast Fallover)
Details and Design Choices
BGP Specific: Features

Requires “BGP AS_PATH Multipath Relax”

We rely on ECMP for routing
Needed for Anycast prefixes

We use 16-bit Private BGP ASN’s ONLY

Simplifies path hiding at WAN edge (remove private AS)
Simplifies route-filtering at WAN edge (single regexp)

But we only have 1022 Private ASN’s…
BGP Specifics: Allow AS In

Reuse Private ASNs on the ToRs

Use of *Allow AS in* on ToR eBGP peerings

Effectively, ToR numbering is local to the container

*Requires vendor support…*
Features that would benefit standardizing

There isn’t that many requirements...

- ECMP programming
- AS_PATH Multipath Relax
- Allow AS In
- Fast eBGP Fall-over
- Remove Private AS
- Unequal-cost load-balancing
- 32-bit Private ASNs
Questions?