BGP Update Compression
• Problem Overview
• Solution Overview
Problem Overview

• **Problem:** BGP is becoming an ALIDDB (any layer information distributed database)
  - BGP update volume is going up steadily (v6, add-path, EVPN, NFV, more and more attributes and AFs)
  - Certain scenarios like virtualized environments make BGP I/O costly
    - Multiple context switches overhead
    - Many-hops TCP paths
  - vRR, VPE, vRS are becoming more and more common

• **Opportunity:**
  - We have now more idle cores to run control plane on controller cards and CPU cores in virtualized environments are plentiful
  - Idle CPU can be used to extend “I/O envelope” available to BGP by “compression”
Problem Overview Cont’d

• Further observations
  • BGP encoding is very “chatty” and may stress TCP I/O with tons of “small things to say”
  • I/O bottleneck is always limited by the “weakest” link which is any of
    • Kernel/Process Context switching/Hypervisor switching
    • Slowest element in the TCP processing chain
    • Packet engine processing
    • Controller to packet engine communication

• Compression is asymmetric
  • Decoding is far, far cheaper than compression and hence easy and inexpensive to implement inline for low end “clients”

• Compression is classical case of a “channel filter” like “de-noiser” or “encryption”
  • Huffman encoding is very well understood and one of the most stable, portable open source libraries
  • Compression could be “stacked” with other filters in the future to provide desired “channel characteristics”. “Channel filters” are a well understood systems software pattern
  • Compression is agnostic to data carried, e.g. AFs and with that future proof

• Yes, BGP could peer over compressed tunnels with its own set of problems
Solution Overview: Compression

• We optionally compress BGP updates
  • New optional capability to advertise “can decompress”
    • On reception of such capability the receiver MAY compress
  • Compressed and uncompressed updates can be mixed at sender’s discretion
  • Asymmetric
    • Compression is independent of decompression implementation (system can choose to support any combination of both)
One Picture

Updates in Time

Compression Filter

Overflow

Normal Update
Compressed Upd
Keep Alive
R-Refr Request
Solution Overview: Subtle Details

- Multiple compressors can be run on stream at same time
- Sender can reset compressor at any time and signal receiver (forcing it to reset decompressor)
- Sender indicates on compressed message
  - Buffer size needed to decompress the message
  - Possible overflow (compressed message followed immediately by another fragment delivering total up to 8K compressed data)
  - Number of the compressor (up to 8)
  - Possible reset