Extended Optional Parameters

John Scudder jgs@juniper.net
IDR, July 30, 2009
Problem Statement

- BGP Capabilities are widely, and increasingly, used
- Capabilities are carried within the Optional Parameters field of the OPEN message
- Optional Parameters has a one-byte length field so is limited to 255 bytes of payload
- Could become too limiting as more Capabilities are introduced
Present-Day Worst Case

• Address family based capabilities have length multiplied
• Address families:
  – AFI: IPv4, IPv6, NSAP, L2VPN
  – SAFI: unicast, multicast, label, tunnel, MDT, VPN
  – (AFI, SAFI) combinations: \( N = AFI \times SAFI \)
• Address family based capabilities:
  – Multiprotocol: \( 6 \times N \)
  – Graceful restart: \( 2 + 6 \times N \)
  – ORF: \( 2 + 7 \times N \)
  – Add-path: \( 2 + 4 \times N \)
  – Extended nexthop encoding: \( 2 + 6 \times N \)
• Thus, worst-case total bytes = 704
Proposed Solution

• If (and only if) Optional Parameters value exceeds 255, set the (legacy) length field and the following byte to 255
  – This indicates that the subsequent two bytes contain the Extended Optional Parameters Length

• Details next
Current Encoding

• Normal Optional Parameter encoding is `<length, value>` where length is a one-byte field
  – Value contains individual Optional Parameters, typically the Capabilities parameter (type 2).

• Example, Optional Parameters with one MP-BGP capability listing IPv4 Unicast:
  – Opt Parms Length = 8
    • Opt Parm Type = 2, Opt Parm Length = 6
      – Capability Type = 1, Capability Length = 4, Capability value = 0x00010001
Proposed Encoding

• If (and only if) Optional Parameters value exceeds 255 bytes, change encoding to be <255, 255, extended length, value>
  – Extended length is two bytes
  – Also change length field of individual optional parameters to be two bytes instead of one

• Example, Optional Parameters with 1000 bytes worth of Capabilities
  – Legacy Opt Parms Length = 255
  – Subsequent ("cookie") byte = 255
  – Extended Opt Parms length = 1003
    • Opt Parm Type = 2, Extended Opt Parm Length = 1000
      – (followed by 1000 bytes worth of Capabilities)
Backward Compatibility

• Encoding doesn’t change as long as payload is less than or equal to 255 bytes
  – If exactly 255, new speaker recognizes old encoding because subsequent byte is *not* 255, but instead is an Optional Parameter Type code
  – Document reserves Optional Parameter Type code of 255, thus it will never legitimately appear

• If payload exceeds 255 bytes, peering wouldn’t have come up anyway!
Conclusion

• Problem is real, though not imminent
• Fix is completely backward compatible
  – Encodings don’t even change until they must
• Fix easy to roll out if we start now
  – Painful if we wait until it’s an emergency
• Similar to four-byte AS situation
Next Steps

• Current draft is draft-chen-bgp-ext-opt-param-01
• Propose we make this an IDR working group document