Naked BGP

What does BGP4 look like on the wire?

Thomas Mangin
Exa Networks Limited

UKNOF 15
Who reads RFCs?

- Desperate network engineers
  - why is that session « stuck in ACTIVE »
  - why are my routers now crashing (ASN4)
- Bleeding edge engineers
  - what is FlowSpec ?
- Curious Engineers
  - what if I changed the HoldTime value to 3 ??
- MAD people
  - wanting to write their own software ← I am here ..
  - mostly SIP developers nowadays
A new application why?

- Announce our service IP (/32)
  - SMTP, MX, POP, IMAP, WEBMAIL, AUTH DNS, ...
- Others exist but ....
  - OpenBGPD – great but no official support on Linux
  - BIRD – good but no package for all our Linux distros
  - Quagga – Cisco configuration format (pain)
  - bgpfeeder, bgpsimple, pybgp – no IPv6
- Wanted ....
  - easy installation (python always installed, nothing else needed)
  - familiar and simple configuration
  - integrate with our code base (suspension, IWF filtering, etc.)
BGP4 – Main RFCs

✔ RFC 4271
  - A Border Gateway Protocol 4 (BGP-4)
  - Obsoletes: 1771

✔ RFC 5492
  - Capabilities Advertisement with BGP-4
  - Obsoletes: 3392, 2842

✗ RFC 2385
  - Protection of BGP Sessions via the TCP MD5 Signature
    I can't implement it, the Python socket module does not export TCP_MD5_AUTH
BGP4 – Common RFCs

✗ RFC 3107
- Carrying Label Information in BGP-4

✓ RFC 4760 (and RFC 2545)
- Multiprotocol Extensions for BGP-4
- Obsoletes: 2858

✗ RFC 4893
- BGP Support for Four-octet AS Number Space
BGP4 – Less common RFCs

✔ RFC 4724
   - Graceful Restart Mechanism for BGP

✗ RFC 4360
   - BGP Extended Communities Attribute

✗ RFC 5575
   - Dissemination of Flow Specification Rules

→ Find all BGP-4 related RFCs
   - http://www.bgp4.as/rfc
Packets

- Very few types
  - OPEN – to negotiate a BGP4 connection
  - NOTIFICATION – to report issues to the peer
  - KEEPALIVE – to not wait for a TCP timeout
  - UPDATE – to exchange routes
- More defined by other RFCs
  - RFC 2918 – ROUTE REFRESH
  - ...
Steps of a BGP session

Opening sequence of packets

Configured but not ready  (IDLE)
Configured and ready  (ACTIVE)
TCP connection  (CONNECT)
→ OPEN  (OPENSENT)
← OPEN
← KEEPALIVE  (OPENCONFIRM)
→ KEEPALIVE  (ESTABLISHED)
Conversation

exchange of routes (if needed)...

→ UPDATE ?
← UPDATE ?

Routes are not re-sent if no change occurs (unless both routers support route refresh)

And start to send each other messages to detect dead peers

→ KEEPALIVE
← KEEPALIVE
Message

- Marker – 16 bytes
  - legacy header from RFC 1105
    - Marker (2 bytes), Length (2 bytes), version (1 byte)
    - Type (1 byte), HoldTime (1 Byte)
  - kept but blanked
    - 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

- Length of content – 1 short

- Type of Message – 1 byte
  - OPEN 0x01
  - UPDATE 0x02
  - NOTIFICATION 0x04
  - KEEPALIVE 0x08
OPEN

- Version – 1 byte
- ASN – 1 short
- HoldTime – 1 short
- BGP Identifier – 4 bytes
- Optional Parameter Length – 1 Byte
- Optional Parameters – LEN bytes
  - initially for optional authentication (deprecated)
  - now for capabilities allowing protocol extension
OPEN

• HOLDTIME
  - heartbeat interval
  - negotiated as the lower holdtime between both OPEN
    • You can cause lots of BGP traffic forcing a low value
  - default depends on vendor
    • Juniper  90
    • Cisco    180
  - can not be lower than 3 (as KeepAlive is HoldTime / 3)
  - connection MAY be rejected based on KEEPALIVE value

• BGP IDENTIFIER
  - unique 32 bit number
    • not really an IP but set to the IP used for the connection (in IPv4)
  - often called Router-ID
  - used to know which connection the router must keep
CAPABILITIES

• Unknown capabilities received are ignored
• A capability can be sent multiple times with different values
  - For example to indicate support for multiple protocols

• Capabilities Format
  • Code 1 byte
  • Length 1 byte
  • Value LEN byte(s)
CAPABILITIES


- RESERVED 0x00
- MULTIPROTOCOL_EXTENSIONS 0x01 [RFC2858]
- ROUTE_REFRESH 0x02 [RFC2918]
- OUTBOUND_ROUTE_FILTERING 0x03 [RFC5291]
- MULTIPLE_ROUTES 0x04 [RFC3107]
- EXTENDED_NEXT_HOP 0x05 [RFC5549]
- Unassigned 0x06 - 0x3F (63)
- GRACEFUL_RESTART 0x40 [RFC4724]
- FOUR_BYTES ASN 0x41 [RFC4893]
- Deprecated 0x42 (66)
- DYNAMIC_CAPABILITY 0x43 [Chen]
- MULTISESSION_BGP 0x44 [Appanna]
- ADD_PATH 0x45 [draft-ietf-idr-add-paths]
- Unassigned 0x46 (70) - 0x7F (127)
- Reserved for Private Use 0x80 (128) – 0xFF (255) [RFC5492]
- CISCO_ROUTE_REFRESH 0x80
  - Can only find reference to this in the router logs
CAPABILITIES

• AFI - Address Family Identifiers
  • http://www.iana.org/assignments/address-family-numbers/
  • IPv4 – 0x01
  • IPv6 – 0x02

• SAFI - Subsequent AFI
  • http://www.iana.org/assignments/safi-namespace
  • SAFI Unicast – 0x01
  • SAFI Multicast – 0x02
  • MPLS-labeled VPN address – 0x80
CAPABILITIES

• Multiprotocol extension
  • OPEN with family (AFI/SAFI) of extra protocols supported
    – one capability per pair supported
  • http://www.iana.org/assignments/address-family-numbers/
  • http://www.iana.org/assignments/safi-namespace

• Graceful Restart
  • let the speaker know
    – if the session is from a restart
    – how long to wait before dropping stale routes
  • AFI/SAFI for which GR is supported
OPEN Message
Marker: 16 bytes
Length: 45 bytes
Type: OPEN Message (1)
Version: 4
My AS: 100
Hold time: 180
BGP identifier: 1.1.1.1
Optional parameters length: 16 bytes
Optional parameters
   Capabilities Advertisement (8 bytes)
      Parameter type: Capabilities (2)
      Parameter length: 6 bytes
      Multiprotocol extensions capability (6 bytes)
         Capability code: Multiprotocol extensions capability (1)
         Capability length: 4 bytes
         Capability value
            Address family identifier: IPv4 (1)
            Reserved: 1 byte
            Subsequent address family identifier: Unicast (1)
   Capabilities Advertisement (4 bytes)
      Parameter type: Capabilities (2)
      Parameter length: 2 bytes
      Route refresh capability (2 bytes)
         Capability code: Route refresh capability (128)
         Capability length: 0 bytes
   Capabilities Advertisement (4 bytes)
      Parameter type: Capabilities (2)
      Parameter length: 2 bytes
      Route refresh capability (2 bytes)
         Capability code: Route refresh capability (2)
         Capability length: 0 bytes
NOTIFICATION

• Format
  • Error code 1 byte
  • Error subcode 1 byte
  • Data variable

• Error codes
  1 – Message header error 4 – Hold timer expired
  2 – OPEN message error 5 – State machine error
  3 – UPDATE message error 6 – Cease

• Error Sub Code
  • too many to list, see RFC 4271 section 4.5

• Data is a human readable string
  • its length is calculated from the length of the message
KEEPALIVE

• No content, just the BGP Header
• Heartbeat message
• If no message is seen during a HoldTime period, the session must be torn down
  • KeepAliveTime = HoldTime / 3
  • « a reasonable maximum time »
  • « no more than once a second »
• KEEPALIVE message should be sent every KeepAlive time if no UPDATE was generated to make sure no Timeout occurs
UPDATE

- Used to update remote RIB
- For IPv4 Nice and simple
  - routes to remove (in NLRI format)
  - characteristics of the new routes
  - new routes (in NLRI format)

Format

- Withdrawn Routes Length 2 bytes
- Withdrawn Routes LEN above bytes
- Total Path Attribute Length 2 bytes
- Path Attributes LEN above bytes
- NLRI(s) what is left

Space efficient

- Maximum message size is 4096
NLRI

• Network Layer Reachability Information
  • Fancy RFC name for a prefix
  • Netmask as a character => /32 byte of value 32
  • Followed by only the necessary bytes of the IP address

• Examples
  • 10.0.0.0/8  0x08  0x10
  • 192.0.2.0/24  0x18  0xC0  0x00  0x02
  • 192.0.2.1/29  0x1D  0xC0  0x00  0x02  0x01
  • 0.0.0.0/0  0x00
Path Attributes

- Store routes meta-data
  - Transitive: Router must relay the Attribute
    - Unknown Transitive SHOULD be accepted
    - Unknown non-transitive MUST be ignored
  - Optional: Understanding of this attribute is optional
  - Mandatory: Must be present (or Discretionary)
    - Well known MUST be transitive
    - MUST be supported by every implementation
  - Partial: Do we know this attribute
    - Once set as unknown the value stays set
  - Every route in the path can add some optional transitive attribute

- Well Known Attributes (minimum implementation)
  - Mandatory ORIGIN, AS_PATH, NEXT_HOP
  - Discretionary LOCAL_PREF, ATOMIC_AGGREGATE
Path Attributes

- **Best known attributes**
  - **CODE** | **NAME** | **FLAGS** | **Number** | **Other**
  - 0x01 | ORIGIN | Mandatory, Transitive | Unique
  - 0x02 | AS-PATH | Mandatory, Transitive | Unique
  - 0x03 | NEXT_HOP | Mandatory, Transitive | Unique
  - 0x04 | MED | Optional | Unique | EBGP only
  - 0x05 | LOCALPREF | Discretionary, Transitive | Unique | IBGP only
  - 0x06 | ATOMIC_AGGREGATE | Discretionary, Transitive |
  - 0x07 | AGGREGATOR | Optional | Unique |
  - 0x08 | COMMUNITIES | Optional, Transitive | Unique |
  - 0x09 | ORIGINATOR_ID, 0x0A CLUSTER_LIST |
  - 0x0E | MP REACH NLRI | Optional, Transitive | Multiple |
  - 0x0F | MP UNREACH NLRI | Optional, Transitive | Multiple |

- **Selection Algorithm order**
  - highest LOCAL_PREF – shorter AS_PATH – lower ORIGIN – lowest MED – EBGP over IBGP
Path Attributes

- **Attribute Flag** 1 byte
  - Flags description
    - 0x10 EXTENDED_LENGTH: The length is two bytes and not one
    - 0x20 PARTIAL: do we understand what is relaid
    - 0x40 TRANSITIVE: order to pass the attribute even if non known
    - 0x80 OPTIONAL: zero for Well Known Attributes
  - Sum of all the flags (some would say binary OR)

- **Attribute Code** 1 byte
- **Length** 1 byte or 1 short
- **Attribute Value** LEN Above
  - content of the Attribute dependant on the attribute code
ORIGIN

- Attribute Value
  - 1 byte with the origin
    - 0x00  IGP
      Network Layer Reachability Information is interior to the originating AS
    - 0x01  EGP
      Network Layer Reachability Information learned via the EGP protocol [RFC904]
    - 0x02  INCOMPLETE
      Network Layer Reachability Information learned by some other means
AS_PATH

• Attribute Value
  • Sequence of one or multiple path segments
    – path segment type 1 byte
      • 0x01 AS_SET
        – unordered set of ASes
        – Included when performing an aggregation
      • 0x02 AS_SEQUENCE
        – ordered set of ASes
        – Used path the path vector algorithm
    – path segment length 1 byte
    – length, path segment value ABOVE LEN * 2 byte(s)
      • list of short integer
NEXT_HOP

• Attribute Value
  – IP 4 bytes
    • inet_aton representation of the IPv4

• Well Known Attribute
  – in RFC 4271

• Does not always need to be present
  – in RFC 4760
LOCAL_PREF, MED, ...

- Attribute Value
  - long integer  4 bytes

- The other Attributes are waiting for you in RFC 4271
UPDATE Message (I removed a MED attribute and removed a route to fit the slide so the sizes are off)
Marker: 16 bytes
Length: 52 bytes
Type: UPDATE Message (2)
Unfeasible routes length: 0 bytes
Total path attribute length: 25 bytes
Path attributes
  ORIGIN: IGP (4 bytes)
    Flags: 0x40 (Well-known, Transitive, Complete)
      0... .... = Well-known
      .1.. .... = Transitive
      ..0. .... = Complete
      ...0 .... = Regular length
    Type code: ORIGIN (1)
    Length: 1 byte
    Origin: IGP (0)
  AS_PATH: 100 (7 bytes)
    Flags: 0x40 (Well-known, Transitive, Complete)
      0... .... = Well-known
      .1.. .... = Transitive
      ..0. .... = Complete
      ...0 .... = Regular length
    Type code: AS_PATH (2)
    Length: 4 bytes
    AS path: 100
      AS path segment: 100
        Path segment type: AS_SEQUENCE (2)
        Path segment length: 1 AS
        Path segment value: 100
  NEXT_HOP: 10.0.0.1 (7 bytes)
    Flags: 0x40 (Well-known, Transitive, Complete)
      0... .... = Well-known
      .1.. .... = Transitive
      ..0. .... = Complete
      ...0 .... = Regular length
    Type code: NEXT_HOP (3)
    Length: 4 bytes
    Next hop: 10.0.0.1 (10.0.0.1)
Network layer reachability information: 4 bytes
  50.0.0.0/24
    NLRI prefix length: 24
    NLRI prefix: 50.0.0.0 (50.0.0.0)
Path Attribute and IPv6

- Announcing an IPv6 route
  - The AFI/SAFI family pair must have been received in the OPEN CAPABILITY
- Special case of MultiProcol BGP
  - Create a UPDATE
    - with no withdrawal
    - with no NLRI
    - with an ORIGIN and AS_PATH (NEXT_HOP ignored)
    - If any, one MP UNREACH NLRI with all the routes to remove
    - If any, one MP REACH NRI with all the routes to add
  - Only takes a few bytes more to use MP BGP for IPv4
- MP BGP is an elegant solution to avoid BGP5
MP_UNREACH_NLRI

• Format
  - AFI 2 bytes
  - SAFI 1 byte
  - Withdrawn NLRIs remaining data

• To send IPv6 routes
  • The AFI/SAFI family pair must have been received in the OPEN CAPABILITY

• Could be used to send IPv4 routes as well
  • Most routers do not announce IPv4 Unicast/Multicast in their OPEN
MP_REACH_NLRI

- **Format**
  - AFI: 2 bytes
  - SAFI: 1 byte
  - Length of Next HOP: 1 byte
  - Next HOP: ABOVE LEN
  - Reserved (must be zero): 1 byte
  - List of NLRIs: remaining data

- **IPv6 has 3 unicast address scope**
  - Global: well suited for routing
  - Site-local: BGP has no concept of site and can not use it
  - Link-local: only relevant for both BPG speakers

- **IPv6 Next HOP**
  - next-hop size can be 16 or 32 (one or two IPs)
  - global IP is required
  - Link-local
    - may be included
    - may be remove by the receiving router
Graceful Restart

• A Change to the forwarding
  – keep routes in RIB
    • when BGP connection is lost
    • If an new OPEN negotiation start even if nothing wrong detected

• End-of-RIB Marker
  – is a valid UPDATE for the AFI/SAFI family
    • with no reachable NLRI
    • with empty withdrawn NLRI
    • with no Path Attribute
  – inform that all the routes have been (re)transmitted

• Often implemented even if hardware can not retain routes on reboot for faster route selection
Graceful Restart

- Capability
  - Restart Flag
    - indicate we are recovering from a failure
    - prevent deadlock caused by waiting for the EOR marker when multiple BGP speakers peering with each other restart
  - Restart Time
    - estimated time to re-establish the connection
    - prevent waiting for a dead peer
  - The AFI/SAFI for which GR is supported
  - Address Family Flag
    - let the router know if forwarding was well maintained during reboot
Graceful Restart

- Allow hitless switch of BGP process
  - switch master to backup RE and back
  - the router must still route during the BGP restart

- Peer announced Graceful Restart
  - connection is detected as failed
    - no end notification was sent
  - the router does NOT remove the BGP routes
    - mark them as stale but keep using them
    - wait for the time specified in the OPEN capability
    - if no changes, remove the route
RFC 5575 / Flow Spec

- **What is RFC 5575 ?**
  - previously known as « flow spec » before August 2009
  - supported by Juniper (no idea about Cisco)
  - drafts by Juniper, Abor and NTT
    - 2 of the 4 Juniper engineers have Cisco emails in the RFC :)

- **What is a « flow » ?**
  - new NLRI (like IPv6, MPLS, VPLS, …)
    - but not a « route » more a firewall match condition
    - AFI 1, SAFI 133 for internet traffic
    - AFI 1, SAFI 134 for MPLS traffic
    - validated against corresponding unicast routing table
  - build with « components »

- **Why use it ?**
  - handle DDOS with ASIC accelerated routers
  - throttle protocols
  - redirect selected type of traffic
RFC 5575 / Flow Spec

• Possible components making the flow
  • Prefix (source and destination)
  • IP Protocol (list of <action, value>)
    – end of list, AND, LEN, less than, more than, equal
    – allow to express a port range, ie  > 6880 and < 6890
  • Port (source, destination, either)
  • ICMP (type, code)
  • TCP flag (list of <action, value>)
    – end of list, AND, LEN, NOT, match (set or unset)
  • Packet Len
  • DSCP
  • Fragment
    – Don't Fragment, Is Fragment, First Fragment, Last Fragment

• Format
  • the RFC includes some example packets
  • and how to decode them in the RFC :D
RFC 5575 / Flow Spec

- Filtering actions

- Use communities (your network, your choice)
  - Normal or extended
  - No convention but a small set of extended communities
    - See RFC 4360 …
      - 0x8006 traffic-rate 2-byte as#, 4-byte float
      - 0x8007 traffic-action bitmask
        - 0x47 Terminal Filtering Action
        - 0x46 Sample and Log for this NLRI
      - 0x45-0x00 Reserved / Undefined
      - 0x8008 redirect 6-byte Route Target
      - 0x8009 traffic-marking DSCP value
Variation between vendors

- Pretty clear and well followed RFC
  - make reading SIP RFC painful
  - no major variation noted

- Malformed Packets
  - Quagga and Cisco accept wrong Attribute Flag for Well Known Attributes (like with wrong Transitivity)
  - Juniper refuse and send you some obscure NOTIFICATION (my fault in the first instance)

- Not many differences
  - CISCO_ROUTE_REFRESH and ROUTE_REFRESH
  - Cisco extra KEEPALIVE as EOR
Extra KEEPALIVE

• Sequence of messages
  → OPEN
  ← OPEN
  ← KEEPALIVE
  → KEEPALIVE (end of OPEN sequence)
  ← KEEPALIVE (as no update / EOR ?)
  ← KEEPALIVE (used as EOR / Normal KA ?)

  Normal usage of KEEPALIVE

• Not in any RFC
BGP route injector

• Usage
  • initially for ASN 112 announcement
  • now to announce all customer facing IPs (/32)
    – for both IPv4 and IPv6
  • Replaced some LVS and Wackamole

• Graceful Restart allows for
  – for service on one machine only
    • restart the daemon without flap on config change
    • reboot machine without causing any routing change

• A low hold-time allows to:
  – rapid fail-over to a active backup machine
BGP route injector

- Juniper do not like gratuitous ARP
  - disabling it is a security risk
  - behaviour may only be changed per interface, not VLAN
  - causes issues with most failover systems client side
  - not able to announce /32 or /128 using ARP broadcast

- Exa Networks' BGP route injector
  - http://bgp.exa.org.uk/
  - Juniper like syntax
Example – ASN 112

neighbor 192.0.2.254 {
    description "a core bgp router";
    router-id 192.175.48.254;
    local-address 10.0.0.254;
    local-as 112;
    peer-as 64511;
    hold-time 30;
    graceful-restart 300;

    static {
        route 192.175.48.0/25 {
            next-hop 192.0.2.1;
            med 100;
            community [ 64511:30740 64511:0 ];
        }
        route 192.175.48.128/25 next-hop 192.0.2.2 community 0x101;
    }
}
The program itself

- No dependencies
- No need to run as root (does not bind)
- Single threaded with co-routine
- Recommend the issue of daemon tools for supervision
- In production in our network for a few months
QUESTIONS ??

Answers :

• Why is a router stuck in active ?
  • it could not establish a connection to its peer
  • it is not trying anymore (configuration, algo choice, ...)
  • your peer is not trying neither
  • forcing the peer to return to IDLE state will force a new attempt to connect

• Why is my router crashing
  • The answer is at http://www.andyd.net/media/talks/asn4_breaks_network.pdf

• What is flow spec
  • Now you know !

• What if I use a Holdtime of 3
  • Lots of KEEPALIVE packets being exchanged
  • The fastest possible detection of peer failure without BFD
Mandriva fun ...

# urpmi bird
To satisfy dependencies, the following packages are going to be installed:

<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
<th>Release</th>
<th>Arch</th>
</tr>
</thead>
<tbody>
<tr>
<td>libquagga0</td>
<td>0.99.7</td>
<td>2mdv2008.0</td>
<td>i586</td>
</tr>
<tr>
<td>quagga</td>
<td>0.99.7</td>
<td>2mdv2008.0</td>
<td>i586</td>
</tr>
</tbody>
</table>

3.7MB of additional disk space will be used.
Proceed with the installation of the 2 packages? (Y/n) n