C-BGP
A new approach to BGP simulation

http://cbgp.info.ucl.ac.be/

(1/2)

Bruno Quoitin
(bqu@info.ucl.ac.be)
Université Catholique de Louvain
Computer Science and Engineering Dept.
Introduction & Internals
Agenda

- Introduction
- How does it work? Internals
  - topology representation
  - simulator's principle
  - BGP details
- How do we use it? Scripting
  - setting up the topology
  - setting up BGP routers
  - policies
- Hands on...
Motivations

- **Main interests: Traffic Engineering**
  - study effects of manipulating BGP attributes
  - investigate changes in BGP protocol

- **Requirements**
  - Policy Routing (no protocol dynamics)
  - Large timescale aspects (10ths of minutes, hours)
  - Scalability & efficiency: topologies with same order of magnitude as Internet
Motivations

- Why a new simulator?

- Traditional simulators do not scale
  - Javasim/SSFNet
  - Packet-level simulation
  - Granularity is too high
Development started in **December 2002**
- under the **ATRIUM** project and now continued under the **TOTEM** project.

**First version (1.0.0)**
- simplified BGP decision process (local-pref, as-path, tie-break)
- AS-level: one router per AS
History

- **Latest version (1.1.17)**
  - complete decision process
  - route reflectors
  - versatile filters
  - intradomain routing
  - various MED behaviours
  - command-line

- **Announced version (1.1.18)**
  - integration within TOTEM toolbox
  - support for XML topologies
New approach

**Assumptions:**
- BGP sessions rely on TCP
- Outcome of BGP decision process does not rely on time

**Consequences:**
- Transmission lines never fail => simulator only preserves BGP messages ordering on each transmission line
- Deterministic BGP decision process
How does it work?

Conceptual view

- **script / CLI**
- **setup**
- **read**
- **update**

- **start/stop**
- **read**

- **topology DB**
  - [nodes, links, ...]

- **Router configs DB**
  - [peerings, ...]

- **routers states DB**
  - [RIBs, sessions...]

- **scheduler**
  - [queue of events]
**Internals (1)**

- **Topology database**
  - Topology represented as a **graph**
    - node = router, identified by a single IP address
    - edge = physical link
  - **Graph implementation**
    - radix-tree contains all nodes: O(1)
    - adjacencies maintained in sorted heaps: O(n)
  - **Edge attributes**
    - delay (informal)
    - metric (for intradomain routing)
  - **Current limitation:**
    - single link between nodes
Example: topological view

- **AS1**
- 10.0.0.1
- 10.0.0.2
- 10.0.0.3
- 10.0.0.4

- Physical link
- iBGP session

```
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
```

© 2004, B. Quoitin
Internals (3)

- **Example:** content of topology database

```
10.0.0.1
adjacencies (sorted heap)
10.0.0.2, 10
10.0.0.3, 5

10.0.0.2
...

10.0.0.3
...

10.0.0.4
...
```

network (radix-tree)
Internals (4)

- **Routers configs database**
  - List of BGP neighbors (sorted heap)
  - Per peer input/output filters

- **Routers states database**
  - Routing table (radix-trees)
    - STATIC routes
    - IGP routes (computed)
    - BGP routes (filled by BGP decision process)
  - BGP sessions
  - Adj-RIBs, Loc-RIBs (radix-trees)
Example: routers configs/states databases network

Routing table (radix-tree)

BGP neighbors: (sorted heap)
- 10.0.0.2 [filters, Adj-RIB-in, Adj-RIB-out]
- 10.0.0.3 [...]
- 10.0.0.4 [...]

Loc-RIB (radix-tree)
Internals (6)

- **Scheduler**
  - Single linear global queue
  - Holds messages (with $src$, $dst$ and $next-hop$)
  - Sequential processing of messages by a single thread
  - Messages are propagated hop-by-hop until they reach their final destination
Simulator's FIFO queue

- Source router sends a new message.
- The message is pushed into the FIFO queue.
- The message is popped from the queue and delivered to the next-hop router.
- Two UPDATE messages are shown:
  - UPDATE [192.168.1/24]
    - src=10.0.0.3
    - dst=10.0.0.4
    - next-hop=10.0.0.4
  - UPDATE [192.168.0/24]
    - src=10.0.0.1
    - dst=10.0.0.2
    - next-hop=10.0.0.2
Structure of a router (simplified)
Internals (9)

Decision process (simplified)

0). ignores routes with unreachable next-hop
1). prefer routes with highest LOCAL-PREF
2). prefer routes with shortest AS-PATH

...  

i). prefer eBGP routes over iBGP routes

...  

j). prefer routes with nearest IGP next-hop

...  

n). prefer routes with smallest IP address
Per peer filters

- sequence of rules (predicates, actions)
- predicates: match prefix, community
- actions: local-pref, communities, as-path prepending, ...

Diagram:
- Input filter: (predicates, actions) (predicates, actions)...
- Adj-RIB-in
- Decision process
- Adj-RIB-out
- Loc-RIB
- Output filter: (predicates, actions) (predicates, actions)...

Peer 1

Peer 2
Filtering one route

```python
filter(ROUTE)
    foreach (rule R=(predicate, actions))
        if (R.predicate matches ROUTE)
            for (i = 0; i < size(R.actions); i++)
                if (R.action[i] == ACCEPT)
                    return ACCEPT;
                elsif (R.action[i] == DENY) {
                    return DENY;
                }
            else
                apply(R.action[i], ROUTE);
                return ACCEPT;
```
Example: running simulation
Example: running simulation

Internals (13)
Example: running simulation

Internals (14)
Example: running simulation
Example: running simulation
Example: running simulation

FIFO queue

AS1
Loc-RIB: 192.168/16, local

AS2
2.1
Loc-RIB: 192.168/16, [AS1]

AS3
3.1
Loc-RIB: 192.168/16, [AS1]

AS4
4.1
192.168/16 [AS2 AS1]
src=3.1
dst=4.1
Example: running simulation

**Internals (18)**
• Example: running simulation

**Loc-RIB:** 192.168/16, [AS1]

**Loc-RIB:** 192.168/16, local

**FIFO queue**
Internals (20)

Example: running simulation

- Loc-RIB: 192.168/16, [AS2 AS1]
- Loc-RIB: 192.168/16, [AS1]
- Loc-RIB: 192.168/16, [AS1]
- Loc-RIB: 192.168/16, [AS2 AS1]
- Loc-RIB: 192.168/16, local

FIFO queue
Example: running simulation

FIFO queue
<empty>
How do we setup a C-BGP simulation?

- through scripting
- file / interactive
- CISCO-like syntax

A few main steps:

- create nodes and links
- configure static routes
- configure IGP
- setup BGP routers (peerings, filters, etc)
- run simulation...
Scripting (2)

- CISCO-like syntax
  - sequence of keywords and parameters

Console

```
cbgp> net add node 10.0.0.1
cbgp> bgp add router 1 10.0.0.1
cbgp> bgp router 10.0.0.1
cbgp-router> add network 192.168/16
cbgp-router> add peer 1 10.0.0.2
cbgp-router> peer 10.0.0.2 up
cbgp-router> exit
cbgp> bgp router 10.0.0.1 show rib 192.168/16
```
Scripting (3)

- Classes of commands
  - net ...
    - commands related to topology and IP routing
  - bgp ...
    - commands related to BGP
  - sim ...
    - commands related to the simulator
  - miscellaneous additional commands
Network commands (1)

- `net add node IPaddr`

- `net add link IPaddr1 IPaddr2 Delay`

- `net link IPaddr1 IPaddr2 igp-weight Weight`

- `net node IPaddr route add Prefix Nh Metric`

- `net node IPaddr spf-prefix Prefix`
Network commands (2)

- **net node** IPaddr **show rt** Prefix

- **net node** IPaddr1 **record-route** IPaddr2

- and more...
Example

```
Console
net add node 10.0.0.1
net add node 10.0.0.2
net add link 10.0.0.1 10.0.0.2 10
```
**Network commands (4)**

## Example

**Console**

- net add node 10.0.0.1
- net add node 10.0.0.2
- net add link 10.0.0.1 10.0.0.2 10
- net node 10.0.0.1 spf-prefix 10/8
- net node 10.0.0.2 spf-prefix 10/8

**Routing table:**

- 10.0.0.2 IGP
- 10.0.0.1 IGP
Network commands (5)

Example

```
net add node 10.0.0.1
net add node 10.0.0.2
net add node 20.0.0.1
net add link 10.0.0.1 10.0.0.2 10
net add link 10.0.0.2 20.0.0.1 5
net node 10.0.0.1 spf-prefix 10/8
net node 10.0.0.2 spf-prefix 10/8
net node 10.0.0.2 route add 20.0.0.1/32 20.0.0.1 5
net node 20.0.0.1 route add 10.0.0.2/32 10.0.0.2 5
```

Routing table:
10.0.0.2/32 STATIC

Routing table:
10.0.0.2/32 IGP

Routing table:
10.0.0.1/32 IGP
20.0.0.1/32 STATIC
Network commands (6)

Example

```
Console

net options igp-inter on
net add node 10.0.0.1
net add node 10.0.0.2
net add node 20.0.0.1
net add link 10.0.0.1 10.0.0.2 10
net add link 10.0.0.2 20.0.0.1 5
net node 10.0.0.1 spf-prefix 10/8
net node 10.0.0.2 spf-prefix 10/8
```
BGP commands (1)

- `bgp add router ASnum IPaddr`
- `bgp router IPaddr add network Network`
- `bgp router IPaddr1 add peer ASnum IPaddr2`
- `bgp router IPaddr1 peer IPaddr2 up`
- `bgp router IPaddr1 peer IPaddr2 next-hop-self`
BGP commands (2)

- `bgp router IPaddr1 show rib IPaddr2|Prefix/*`

- `bgp router IPaddr1 show rib-in Peer|* IPaddr2|Prefix|*`

- `bgp router IPaddr record-route Prefix`

- `bgp router IPaddr1 peer IPaddr2`
  - next-hop-self
  - rr-client
  - virtual
  - `recv Message`
BGP commands (3)

Example (iBGP)

<table>
<thead>
<tr>
<th>Console</th>
</tr>
</thead>
<tbody>
<tr>
<td>bgp add router 1 10.0.0.1</td>
</tr>
<tr>
<td>bgp router 10.0.0.1</td>
</tr>
<tr>
<td>add peer 1 10.0.0.2</td>
</tr>
<tr>
<td>peer 10.0.0.2 up</td>
</tr>
<tr>
<td>bgp add router 1 10.0.0.2</td>
</tr>
<tr>
<td>bgp router 10.0.0.2</td>
</tr>
<tr>
<td>add peer 1 10.0.0.1</td>
</tr>
<tr>
<td>peer 10.0.0.1 up</td>
</tr>
</tbody>
</table>

sim run

Routing table:
- 10.0.0.2 IGP

Routing table:
- 10.0.0.1 IGP

AS 1
iBGP session

10.0.0.1

10.0.0.2

10
BGP commands (4)

Example (eBGP)

Console

bgp router 10.0.0.2
   add network 10.0.0/24
   add peer 2 20.0.0.1
   peer 20.0.0.1 up

bgp router 20.0.0.1
   add network 20.0.0/24
   add peer 1 10.0.0.2
   peer 10.0.0.2 up

sim run

Routing table:
10.0.0.2/32 STATIC
20.0.0/24 BGP

Routing table:
10.0.0.1/32 IGP
20.0.0.1/32 STATIC
20.0.0/24 BGP

! missing route!
(20.0.0/24)
**BGP commands (5)**

- **Next-hop-self**

**Console**

```
bgp router 10.0.0.2
   add network 10.0.0/24
   add peer 2 20.0.0.1
peer 20.0.0.1 next-hop-self
peer 20.0.0.1 up

bgp router 20.0.0.1
   add network 20.0.0/24
   add peer 1 10.0.0.2
peer 10.0.0.2 up

sim run
```

Routing table:
10.0.0.2/32 STATIC
20.0.0/24 BGP

Routing table:
10.0.0.1/32 IGP
20.0.0/24 BGP

Use **peer 20.0.0.1 next-hop-self**
BGP commands (6)

- without Next-hop-self

**Console**

- bgp router 10.0.0.2
  - add network 10.0.0/24
  - add peer 2 20.0.0.1
  - peer 20.0.0.1 up

- bgp router 20.0.0.1
  - add network 20.0.0/24
  - add peer 1 10.0.0.2
  - peer 10.0.0.2 up

- sim run

Routing table:
- 10.0.0.2/32 IGP
- 10.0.0.1/32 IGP
- 20.0.0/24 BGP

Routing table:
- 10.0.0.1/32 IGP
- 20.0.0.1/32 IGP
- 20.0.0/24 BGP

Use net options igp-inter on
Miscellaneous commands

- sim run

- include Script
  - example: include /workspace/scripts/toto.cli

- print Message
  - example: print “Hello World\n”

- # This is a comment
Policies (1)

- `bgp router IPaddr1 peer IPaddr2 filter [in|out]
  - add-rule
  - insert-rule `Index`
  - remove-rule `Index`

where a rule is composed of
- **match** *logical combination of predicates*
- **actions** *comma separated list of actions*
Policies (2)

- Supported **match** predicates
  - **any**
    - match all routes
  - **community is Community**
    - match routes that contain given community
      (specified as X or X:Y)
  - **prefix is Prefix**
    - match routes for exact given prefix
  - **prefix in Prefix**
    - match routes contained in given prefix
Policies (3)

- Not yet implemented
  - regular expressions on AS-Path
  - less specific prefixes
Supported actions

- **accept**
  accept the route

- **deny**
  deny the route

- **as-path prepend** *Amount*
  prepend the route's as-path *Amount* times

- **community add** *Community*
  add the given community to the route

- **community strip** *Community*
  remove all the route's communities
Policies (5)

- Supported actions (cont')
  - **local-pref** $Pref$
    change the route's local-pref
  - **metric** $Med/\text{internal}$
    change the route's multi-exit-discriminator (absolute value or based on IGP)
  - **red-community add prepend** $Amount \ Target$
    add a redistribution community that requests prepending $Amount$ times towards $Target$
  - **red-community add ignore** $Target$
    add a redistribution community that requests ignoring $Target$
Policies (6)

Example

Prefer routes received from neighbor 2.0.0.1

```
bgp router 1.0.0.1
  add peer 2 2.0.0.1
peer 2.0.0.1
  filter in
    add-rule
      match any
      action "local-pref 120"
```
Example

Deny routes towards martian prefix 192.168/16

```plaintext
bgp router 1.0.0.1
  add peer 2 2.0.0.1
peer 2.0.0.1
  filter out
  add-rule
    match "prefix is 192.168/16"
action deny
```
C-BGP results (1)

- **Different means**
  - displaying routing table of one router
  - displaying Adj-RIB-in and Loc-RIB of one BGP router
  - tracing route from one router to another
  - tracing AS-path from one router to a prefix
  - recording BGP messages exchanged during the simulation
C-BGP results (2)

Routing table of one node

```
cbgp> net node 1.0.0.1 show rt *
4.0.0.0/8   4.0.0.1 0   BGP
4.0.0.1/32  4.0.0.1 5   STATIC
5.0.0.0/8   5.0.0.2 0   BGP
5.0.0.2/32  5.0.0.2 5   STATIC

cbgp> net node 1.0.0.1 show rt 5/8
5.0.0.0/8   5.0.0.2 0   BGP

cbgp> net node 1.0.0.1 show rt 5.0.0.2
5.0.0.2/32  5.0.0.2 5   STATIC
```
C-BGP results (3)

- Loc-RIB of one router

```
cbgp> bgp router 1.0.0.1 show rib *
i> 1.0.0.0/8  1.0.0.1  0  0  null  i
*> 4.0.0.0/8  4.0.0.1  0  4294967295  4  i
*> 5.0.0.0/8  5.0.0.2  0  4294967295  5  i

cbgp> bgp router 1.0.0.1 show rib 5/8
*> 5.0.0.0/8  5.0.0.2  0  4294967295  5  i

cbgp> bgp router 1.0.0.1 show rib 5.0.0.2
*> 5.0.0.0/8  5.0.0.2  0  4294967295  5  i
```
## Adj-RIB-in of one router

```plaintext
cbgp> bgp router 1.0.0.1 show rib-in 4.0.0.1 *
* 2.0.0.0/8  4.0.0.1 0  4294967295  4 2 i
*> 4.0.0.0/8  4.0.0.1 0  4294967295  4  i
* 6.0.0.0/8  4.0.0.1 0  4294967295  4 2 6 i
```
C-BGP results (5)

- Traceing route

```
cbgp> net node 1.0.0.1 record-route 5.0.0.2
1.0.0.1 5.0.0.2 SUCCESS 1.0.0.1 5.0.0.2

cbgp> net node 1.0.0.1 record-route 6.0.0.2
1.0.0.1 6.0.0.2 SUCCESS 1.0.0.1 2.0.0.1 6.0.0.1 6.0.0.2
```
Traceing AS-path

cbgp> bgp router 1.0.0.1 record-route 5/8
1.0.0.1 5.0.0.0/8     SUCCESS 1 5

cbgp> bgp router 1.0.0.1 record-route 6/8
1.0.0.1 6.0.0.0/8     SUCCESS 1 2 6
C-BGP results (7)

- Recording BGP messages
  - use "bgp options msg-monitor File"

```plaintext
...
6.0.0.2|BGP4|114.00|W|6.0.0.1|6|5.0.0.0/8
5.0.0.2|BGP4|127.00|A|5.0.0.1|5|6.0.0.0/8|3 6|IGP|5.0.0.1|0||
1.0.0.1|BGP4|147.00|A|5.0.0.2|5|6.0.0.0/8|5 3 6|IGP|5.0.0.2|0||
5.0.0.1|BGP4|147.00|W|5.0.0.2|5|6.0.0.0/8
...
```
Route-reflectors (1)

**Goal**
- Decrease number of iBGP sessions
- Decrease information known by BGP routers
- Affect routing choices

**How to setup RRs?**
- Only change sessions with RR-clients
Route-reflectors (2)

Example

```
Console

bgp router 1.0.0.1
add peer 1 1.0.0.2
add peer 1 1.0.0.3
peer 1.0.0.2 rr-client
peer 1.0.0.3 rr-client
peer 1.0.0.2 up
peer 1.0.0.3 up
```

Only change sessions with RR-clients
Route-reflectors (3)

- **Multiple RRs**

  ```
  Console
  bgp router 1.0.0.1
  add peer 1 1.0.0.2
  add peer 1 1.0.0.3
  add peer 1 1.0.0.4
  peer 1.0.0.2 rr-client
  peer 1.0.0.3 rr-client
  peer 1.0.0.2 up
  peer 1.0.0.3 up
  peer 1.0.0.4 up
  ```

- Full-mesh of iBGP sessions between RRs
- Default cluster-ID based on IP address (change with `set cluster-id`)
Hands on... (1)

**Installation**

- download from [http://cbgp.info.ucl.ac.be/](http://cbgp.info.ucl.ac.be/)
  - libgds 1.1.5
  - cbgp 1.1.17

```
tar xvzf libgds-1.1.5.tar.gz
cd libgds-1.1.5
./configure; make; sudo make install
cd..
tar xvzf cbgp-1.1.17.tar.gz
cd cbgp-1.1.17
./configure; make
```
Hands on... (2)

Exercises

- **Ex1**: Write a C-BGP script that models *base-topo-1* (next slide)

- **Ex2**: Based on the topo of Ex1, show the inbound and outbound paths passing on each access link of AS6

- **Ex3**: Try to balance the inbound/outbound paths on each access link of AS6 using local-pref and as-path prepending
base-topo-1

AS4  4.0.0.1
      ↓
    AS2  2.0.0.1
        ↓
      AS6  6.0.0.1
    ↓  ↓  ↓
AS1  1.0.0.1  AS3  3.0.0.1
      ↑  ↑  ↑
    AS5  5.0.0.1

Legend:
- eBGP session (peer-peer)
- eBGP session (customer-provider)
- iBGP session