TECHNISCHE UNIVERSITÄT BERLIN
Fakultät IV – Elektrotechnik und Informatik
Fachgebiet Intelligente Netze und Management verteilter Systeme
Prof. Anja Feldmann, Ph.D.
Gregor Maier, Jörg Wallerich, Andi Wundsam

9th Work Sheet Praktikum Protokolldesign WS 07/08

Please note: you have two lecture weeks time to solve this assignment. because of the Christmas holidays, the solution is due at January 15., 2008
The next discussion group after the Christmas holidays is at January 7., 2008 and covers the 8th assignment. The next tutorial after the Christmas holidays is at January 15., 2008

Question 1: (100 points) Implementation of a Simplified BGP Router in the P2P Overlay

After the implementation of a simple forwarding table for the 7th assignment, we will now improve the way we generate entries in the forwarding table by using an actual routing protocol. Our routing protocol is inspired by `BGP`[1] that is used in the Internet and was introduced for the 8th assignment.

Assume that each node in the P2P overlay is its own Autonomous System (AS) and that the AS number is represented by the node ID.

The simplified BGP protocol to be used is explained in the following paragraphs:

BGP Messages

Routing protocols like BGP exchange reachability information between routers. In the case of BGP this happens between nodes that are connected by a BGP session. For our P2P network, a node has a BGP session to each of its direct neighbours that is implicitly instantiated during the HELLO handshake.

In order to be able to exchange BGP messages between two neighbour nodes, we introduce a new message type. This messages type represents a generic transport mechanism between two neighbouring nodes. It can also be used to transport information for other applications than BGP. As this transport mechanism is intended for communication between neighbours, its messages contain neither originator and destination sepcification nor a TTL field. As the mechanism is a generic one, the `Application:` parameter of the messages has to contain the application that uses this transport mechanism. The messages look like this:

```
LOCALTRANSPORT P2P/0.2\r\n
Application: <app>\r\n
Content-Length: <body size>\r\n
\r\n
<payload>
```

The handling of LOCALTRANSPORT messages works similar to that of handshake messages: they are never forwarded, contain neither originator or destination nor a TTL field. However, even `LOCALTRANSPORT` messages may only be sent *after* a successful `HELLO` handshake. Else this is a protcol violation that requires approriate error handling.

We will use the `LOCALTRANSPORT` mechanism to exchange BGP messages between neighbours. The actual BGP information is transmitted in the body in `\r\n` terminated lines.

The following BGP information is transmitted using `LOCALTRANSPORT` messages:

[1] Border Gateway Protocol

Reporting of new and changing of existing best routes: The messages to report new or to change existing routes look like this:

```
LOCALTRANSPORT P2P/0.2
Application:  BGP
Content-Length:  <bytes>

ANNOUNCEMENT <destination> <aspath>
```

Withdrawing of routes: Messages to inform a neighbour that a previously reported route is no longer usable:

```
LOCALTRANSPORT P2P/0.2
Application:  BGP
Content-Length:  <bytes>

WITHDRAW <destination>
```

Error messages: Error messages look like this:

```
LOCALTRANSPORT P2P/0.2
Application:  BGP
Content-Length:  <bytes>

NOTIFICATION <error string>
```

If a `NOTIFICATION` is received, the connection must be closed immediately!

The body of a BGP message may consist of several ANNOUNCEMENT and WITHDRAW lines and may also contain both. When computing the `Content-Length`, the line end (`\r\n`) counts as 2 bytes.

BGP routes mainly consist of a destination and path that can be used to reach the destination. The path consists of a sequence of AS numbers (here: node IDs).

There can be at most one route per neighbour and destination, because a node always announces only that route that it is using itself. And that always is the best one it knows of.

If a node has multiple neighbours, it may also have multiple routes to the same destination, each using a different neighbour. This redundancy is necessary in the case that a route disappears for some reason. In such a case a node can switch to one of the other routes.

The best route to a destination is that with the shortest AS path.

The AS path is also used to detect routing loops. As a node knows the entire path to a destination, it can easily avoid that it appears multiple times in the path. If a node learns a new route, and the node finds its own AS number in the path, then the route must be ignored.

When a node reports a route to a neighbour, it extendeds the AS path by adding its own node ID, because for the neighbour, the node is a part of the path. The prepending is also necessary for the loop detection mechanism.
When a path is extended to be sent in an ANNOUNCEMENT message, this always happens at the beginning (left hand side) of the path. This is called 'path prepending'.

Format of BGP routes As we don't want to implement all faetures of the real BGP protocol, we will work with the following simplified form of routes:

```
dest    ::= node-id
as-path ::= node-id |
            node-id "," as-path
route   ::= dest _ as-path
```

The node ID at the end of the path is the node that originally reported the route, so it should be identical to the destination. Moreover, the first node at the beginning of the path should be our direct neighbour. And it should be the neighbour we learned that route from.

As described above, the path of the route has to be extended by one's own node ID before sending it to a neighbour.

BGP Table

The BGP table is there to hold all routes we learn over time. Only one of the routes to the same destination is the best one and so the one we use to forward messages.

Entries in the BGP Table An entry in the BGP table has the following format:

```
DESTINATION NEXTHOP ASPATH BEST
```

DESTINATION is the ID of the destination node for this route. NEXTHOP is the neighbour we learned the route from and also the neighbour over which we would forward messages to the destination of the route, if the route was the best. ASPATH contains the AS path to the destination. BEST is a flag that marks a route as the currently best one to its destination.

Initialization of the BGP Table Upon starting the node software, the table is empty. This also holds for the forwarding table.

New Neighbour Connections As soon as a new neighbour connection is set up (after the HELLO handshake) a new route to the new neighbour is entered into the BGP table. The destination of that route is the new neighbour and the AS path consists of the newneighbour. Now a route calculation is started (see below) with the new neighbour as destination.

Afterwards, all our best routes are sent to the new neighbour using ANNOUNCEMENT messages. The AS paths of all routes sent to the new neighbour must be extended by prepending our own node ID to the AS path! This makes the path one hop longer!

The entry in our BGP table is not changed, we only perpend our node ID to paths we send.

Receiving an ANNOUNCEMENT Message When we receive an ANNOUNCEMENT message, we first perform the loop detection check. This is done by searching for our own node ID in the AS path of the announced route. If we find our own node ID in the path, the message must be ignored!

Otherwise, we enter the new route into our BGP table. If there is already a route to the same destination over the same neighbour, we replace the entry in table with the new ones, as this means that our neighbour changed its own route. This is called 'implicit withdraw' as it has the same effect as first withdrawing a route and then announcing another one.

Then, a route calculation for the destination of the route must be performed (see below).

Receiving a WITHDRAW Message When a WITHDRAW message is received, this means that the sending neighbour doen't know how to reach the affected destination anymore. As we cannot use the route to this destination using this neighbour as next hop anymore, the reaction to such a message consists in removing the corresponding entry from the BGP Table

Afterwards we have to perform a route calculation for the affected destination (see below).

Loss of a Neighbour Connection When losing a neighbour connection, we have to delete all routes from the BGP table that use the lost neighbour as their next hop. Afterwards, we have to perform a route calculation for the affected destination (see below). The whole procedure can be handled like a sequence of WITHDRAW messages for all affected routes.

Route Calculation

The route calculation is done one a per-destination basis. If the calculation results in a change of the best route for the destination (this includes if we cannot find a route anymore), all neighbours must be informed about it. Moreover, the forwarding table has to be updated.

A short description of the calculation algorithm looks like this:

- for a given destination:
  - Look for the best route for the current destination (the route with the shortest AS path).
  - if the new best route is different from the old one (different next hop or AS path)
    - Update the forwarding table to use the next hop node of the new best route
    - Broadcast an ANNOUNCEMENT message with the new best route (do path prepending!)
  - if we cannot find a route but there was one before, i.e., we deleted our last route
    - Delete corresponding entry in forwarding table
    - Broadcast a WITHDRAW message for the current destination
  - if the new and the old best routes are the same
    OR
    we didn't have route before and do not have one now
    - done.

Deliverables:

- Your program/script (both source code and binary if written in C, C++ or Java)
- A short description of how to use the software, and what things are not quite working yet.

For details concerning the submission of your solutions see FAQ
(`http://www.net.t-labs.tu-berlin.de/teaching/ws0708/PD_labcourse/`).
Due: Tuesday, Januaery 15., 2008, 11:59h s.t.
MERRY CHRISTMAS AND A HAPPY NEW YEAR!