Routy: Seminar Report

Weherage, Pradeep Peiris

March 4, 2016

1 Introduction

Routy is a link-state routing protocol implemented in Erlang. This report explains the implementation background of Routy together with how the concepts of link-state protocol and its algoritim are applied.

Link state protocols are built with Dijkstra shortest path algorithm in Graph Theory. Compare to other dynamic routing protocol like Distance Vector Routers, Link state protocols are quite accurate as each router maintains the same complete model of the network.

2 Main problems and solutions

Router processes are designed to run on a specific device (Router) to determine and allow communication between networks. There is no shared memory or public services that a router process could access and determine the best route for its receviced messages (Packets). Its routers own process task is to maintain the network picture and decide the best route for its received messages.

The Link State protocol handles above problem with the following main modules.

1. Routing Interface
2. Link State message
3. Network Topology
4. Routing table

2.1 Routing Interface

Each router process needs communication interfaces for its adjacent routers. In Routy, this is simply achieved with Erlang message passing. The interface is defined with a symbolic name, process reference and a process identifier.

Therefor each router processes can send messages to its adjacent router processes via pid.

```
broadcast(Message, Intf)
    -> lists:foreach(fun({_, _, Pid}) -> Pid ! Message end, Intf).
```

2.2 Link State message

Once a link is established the router process publish a Link State message to all its neighbors. This refers as Link State advertisement (LSA) flooding, as in turn each router received a LSA, is forwarded to all its neighbors.

The main implementation challenge here is to avoid cyclic messages to each router processes. Routy sends Link State message with an increasing number of tag value. And let it maintains the history of messages receive from other routers. Using the tag value now, it can avoid the old messages.

```
update(Node, N, History) ->
   case lists:keyfind(Node, 1, History) of
      {Node, H} ->
         case N =< H of
            true -> old;
            false -> {new, lists:append([{Node, N}],
                           lists:keydelete(Node, 1, History))}
         end;
      false -> {new , lists:append([new(Node)], History)}
   end.
```

2.3 Network Topology

With link state message propagation, all routers can generate it's own copy of a network model. It is quite straight forward to implement this with Erlang's lists library.

2.4 Routing table

It is Dijkstra algorithm the challenging part of Routy implementation. Dijkstra uses the network model and routers adjacent gateways as the input for the algorithm. It iterates over its adjacent nodes and calculate the shortest path to other reachable nodes in the network model.

```
iterate([], _, Table) -> Table;
iterate([{_, inf, unknown}|_], _, Table) -> Table;
iterate([{NodeTo, N, NodeFrom}|T], Map, Table) ->
   ReachableNodes = map:reachable(NodeTo, Map),
   SortedUptd = lists:foldl(fun(Node, SortedList) ->
```

```
    update(Node, N + 1, NodeFrom, SortedList) end, T, ReachableNodes),
  UpdatedTable = lists:append([{NodeTo, NodeFrom}], Table),
  iterate(SortedUptd, Map, UpdatedTable).
```

3 Evaluation

A evaluation is done building a simple network of routers. And test that it creates the best routing table upon add and removal of routres from the network.

```
lund --> stockholm --> malmo
lund --> uppsala --> gotebory --> malmo
```

Upon manual broadcast of link-state message, all routers get the following network map.

```
[{gotebory,[malmo]},{uppsala,[gotebory]},
  {lund,[stockholm,uppsala]},{stockholm,[malmo]}]
```

The following is the routing table for 'lund' node.

```
[{gotebory,uppsala},{malmo,stockholm},{uppsala,uppsala},{stockholm,stockholm}]
```

Once 'stockholm' node is dropped shortest path to 'malmo' is updated via 'uppsala'.

```
[{malmo,uppsala},{gotebory,uppsala},{uppsala,uppsala}]
```

4 Conclusions

Routy is a basic implmentation of link-state protocol. The implementation considered only main modules of the protocol; Routing Interface, Link State message propergation, Network topoloty and the Routing table.

Routy can be further improved with proper model of its interface handling. The link-state message broadcasting is now handled manually. It should be automated on network router changes. Also, the update of the Routing table should be executed on arrival of Link State message or scheduled it periodically.