Introducing a Distance Vector Routing Protocol for ns-3 Simulator

Janaka Wijekoon, Rajitha Tennekoon, Erwin Harahap, and Hiroaki Nishi.

Hiroaki Nishi Laboratory, Department of Information and Computer Science
Keio University, Yokohama, Japan

A discussion of introducing, designing, implementing and simulating a distance vector routing protocol on ns-3 simulator
What was the real motivation?

- ISP-CP collaboration towards effective content delivery
- Understand the impact of network components and CP for content delivery
ns-3’s routing architecture

- Supports Linux IP stack like IP stack
- Serializes and de-serializes packets through IP stack
- Supports most generic real-world packet structures
- Supports both IPv4 and IPv6 addressing schemes
The ns-3 as a simulator

• ns-3 is a modular based discrete-event network simulator
  ➢ Modules to emulate network devices including client and servers
  ➢ Real-time scheduler
  ➢ Interactive mode
  ➢ Application and protocol emulation with DCE

• Parallelization
  ➢ Run more than one independent simulation instances

• Hybrid emulations
  ➢ Comment simulations with real systems

• ns-3 supports Direct Code Execution (DCE)
Existing ns-3 routing protocols

- Ns-3 simulator is relatively famous among the researchers who are researching on wireless ad-hoc networks.

- **Wireless Protocols**
  - DSR
  - DSDV
  - AODV
  - OLSR

- **Wired Protocols**
  - Global
  - NIX-Vector
  - BRITE
  - CLICK
  - RIPng [RFC 2080](https://www.rfc-editor.org/rfc/rfc2080)

- Apart of RIPng protocol, most of the routing protocols are only connectivity methods.

- DCE module support most of QUAGGA routing protocols (yet in experimentation level).
A Routing protocol

• For realistic evaluations of ns-3 modules,
  - Network behavior,
  - Network utilization,
  - Network traffic patterns,

• It is a hard task to achieve kink state protocols such as OSPF without spending over 2 years to implement.

At least a distance vector routing protocol.

• Thus, as an initial step, a distance vector routing protocol; DVRP, is designed to the ns-3
What DVRP is?

- A distance vector routing protocol based on both RFCs 2453 and 2080.
  - Note that some modifications were done to ensure
    - steady route processing
    - fast and reliable route management.
- Designed as an Interior Gateway Protocol (IGP)
- Route matrix is calculated using the number of hops for a destination prefix.
- The Bellman-ford algorithm was used for minimum path selection
What DVRP is?

- Minimum Distance Path Calculation

\[
\bar{D}_{ij} (t) = \min_{k \neq j} \{d_{ik} (t) + \bar{D}_{kj} (t)\}
\]
ns-3 implementation overview

ns3Ipv4RoutingProtocol

NeighborTable

NeighborTableEntry

DVRPModule

RoutingTable

RoutingTableEntry

Advertisement Module

RoutingHeader

HelloPacket

RouteUpdateMessage (DVRPRUM)
ns-3 implementation overview

<table>
<thead>
<tr>
<th>Neighbor</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>VALID</td>
</tr>
<tr>
<td>k2</td>
<td>VALID</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dest</th>
<th>GateWay</th>
<th>Cost</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>i</td>
<td>1</td>
<td>VALID</td>
</tr>
<tr>
<td>k2</td>
<td>k2</td>
<td>1</td>
<td>VALID</td>
</tr>
<tr>
<td>k1</td>
<td>k2</td>
<td>2</td>
<td>VALID</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Specifications

- Neighbor Management
  - Neighbor discovery
  - Periodic checking
Specifications

• Neighbor Management
  ➢ Neighbor discovery
  ➢ Periodic checking

• Route management
  ➢ Periodic updates

• UDP based protocol.
• It uses UDP port number 272 for route updates
• Send periodic updates every 20s
• Send entire routing table at the periodic update messages
• Poisoned routes are differentiate using “ODD” values for sequence number.
Specifications

• Neighbor Management
  - Neighbor discovery
  - Periodic checking

• Route management
  - Periodic updates
  - Triggered updates

• Fast route update messages.
• Consider only changed routes.
• Uses a “Holddown” timer as a provision.
• Maximum 5s waiting time between two triggered update messages.
• Also considers poisoned routes.
Specifications

• Neighbor Management
  - Neighbor discovery
  - Periodic checking

• Route management
  - Periodic updates
  - Triggered updates
  - Split Horizon
  - Poisoned Routes
  - Sequenced update messages

• These three techniques were used for stable route messaging.
• Poisoned routes denote using “ODD” sequence numbers.
• Sequence numbers are used to avoid old routes updates.
Specifications

• Neighbor Management
  - Neighbor discovery
  - Periodic checking

• Route management
  - Periodic updates
  - Triggered updates
  - Poisoned Routes
  - Split Horizon
  - Sequenced update messages
  - PUSH and PULL message techniques to fasten the route propagation

• Route request messages
  - At the moment, DVRP supports route requests are for entire routing table.
  - In order to respond the request, requests should be made from a valid neighbor

• Route response messages
  - Response messages:
    • Periodic update messages
    • Triggered update messages
    • Response message for a request
# Default Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeepAliveInterval</td>
<td>The time between two Keep Alive Messages.</td>
<td>30s</td>
</tr>
<tr>
<td>NeighborTimeoutDelay</td>
<td>The delay to mark a neighbor as unresponsive.</td>
<td>60s</td>
</tr>
<tr>
<td>GarbageCollection</td>
<td>The delay to remove unresponsive records from routing and neighbor tables.</td>
<td>10s</td>
</tr>
<tr>
<td>PeriodicUpdateInterval</td>
<td>Time between two periodic updates.</td>
<td>20s</td>
</tr>
<tr>
<td>RouteTimeoutDelay</td>
<td>The delay to mark a route is invalidate.</td>
<td>180s</td>
</tr>
<tr>
<td>MinTriggeredCooldown</td>
<td>Minimum time gap between two triggered updates.</td>
<td>1s</td>
</tr>
<tr>
<td>MaxTriggeredCooldown</td>
<td>Maximum time gap between two triggered updates.</td>
<td>5s</td>
</tr>
<tr>
<td>SplitHorizon</td>
<td>Split Horizon strategy.</td>
<td>SPLIT_HORIZON_W_POISON_REVERSE</td>
</tr>
</tbody>
</table>
Header Structures

• Protocol Routing Header

```c
/**
 * \ingroup DVRP
 * \brief DVRP Routing Header
 */
/**
 * --------------------------------------------------DVRP header--------------------------------------------------
 * | 0 | 1 | 2 | 3 |
 * 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
 * +---------------------------------------------------------------
 * | Command | NA |
 * +---------------------------------------------------------------
 * | RUM /KAM~ |
 * +---------------------------------------------------------------
 */
```
Header Structures

• Route Update Message Header

/**
 * \ingroup DVRP
 * \brief Distance Vector Routing Protocol's Route Update Message (RUM)
 */

/**---------------------------------------------------------------------------RUM---------------------------------------------------------------------------
 * | 0 | 1 | 2 | 3 |
 * 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
 * +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 * | Seq# | Metric |
 * +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 * | Network Address / Host Address |
 * +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 * | Network Mask |
 * +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Header Structures

• Hello / Keep-alive Message Header

/**
 * \ingroup DVRP
 * \brief DVRP Hello and Keep Alive Message (KAM) header
 */
/**
 * +-------------------------------------------Keep Alive Message Header-------------------------------------------+
 * | 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 |
 * +-------------------------------------------+-------------------------------------------+
 * | Command                                     |
 * +-------------------------------------------+
 * | Gateway                                    |
 * +-------------------------------------------+
 * | Mask                                       |
 * +-------------------------------------------+
 */
Process Route Messages

Get IPv4 interface information

Get SocketIPTTL information

Get SocketAddress information

Filter out the piggybacked packets

Get the bounded socket information

If no socket information found ignore the update

Pass header, socket, and interface info. to KAM handling method

Get the neighbor information

If no neighbor information found, ignore

Otherwise, get the bounded socket information

Pass header, socket, and interface info. to Request handling method

Pass header, socket, and interface info. to Response handling method
Process RUM

For each RUM

- If “j” is a local network, go to next RUM
- If “j” is about a broken destination, invalidate routes. Cancel “triggeredevent” and send an immediate triggered update

For each legitimate RUM

- If no route is found for “j”, add it to table and set settlingevent
- Calculates metric

If sequence number is greater

- If Cost is better and gateway is different
  - Replace the route
- If Cost is same and gateway is same
  - Reset the expiration timer
- If Cost is same and gateway is different
  - If the existing route is about to expire (2/3)
    - Replace the route
  - If the cost is lesser than 16
    - update the route
- ELSE
  - mark as poisoned and send triggered update

For next RUM
Neighbor table

// Network topology
//
// SRC
// |<= source network
// A----B
// \ / |  \
// \ / |  \
// C /  |
// | /  |
// |/  |
// D
// |<= target network
// DST
//
// A, B, C and D are DVRP Enabled routers.
// A and D are configured with static addresses.
// SRC and DST will exchange packets.
Routing table

// Network topology
//
// SRC
// |<=== source network
// A---- B
// \ /  |
// \ /  |
// C /  |
// | /  |
// |/  |
// D   |
// |<=== target network
// DST
//
//
// A, B, C and D are DVRP Enabled routers.
// A and D are configured with static addresses.
// SRC and DST will exchange packets.

Node: 3 Time: 270s Ipv4ListRouting table
Priority: 0 Protocol: ns3::DVRPRoutingProtocol

Node: 3 Time: 270s DVRP Routing Table

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>If</th>
<th>Seq#</th>
<th>Metric</th>
<th>Validity</th>
<th>Changed</th>
<th>Expire in (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.16.0/30</td>
<td>15.16.16.1</td>
<td>1</td>
<td>32</td>
<td>1</td>
<td>VALID</td>
<td>0</td>
<td>159.222</td>
</tr>
<tr>
<td>10.10.10.0/24</td>
<td>11.118.126.2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>VALID</td>
<td>0</td>
<td>169.404</td>
</tr>
<tr>
<td>172.16.1.0/30</td>
<td>11.118.126.2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>VALID</td>
<td>0</td>
<td>172.584</td>
</tr>
<tr>
<td>203.15.19.0/24</td>
<td>15.16.16.1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>VALID</td>
<td>0</td>
<td>158.491</td>
</tr>
<tr>
<td>11.118.126.0/24</td>
<td>0.0.0.0</td>
<td>3</td>
<td>36</td>
<td>0</td>
<td>VALID</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>201.13.15.0/24</td>
<td>0.0.0.0</td>
<td>2</td>
<td>36</td>
<td>0</td>
<td>VALID</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15.16.16.0/24</td>
<td>0.0.0.0</td>
<td>1</td>
<td>36</td>
<td>0</td>
<td>VALID</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>127.0.0.1/32</td>
<td>0.0.0.0</td>
<td>0</td>
<td>36</td>
<td>0</td>
<td>Loc. Host</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Result and Discussion

Route Decision Making Time

Packet Delivery Ratio
Result and Discussion

RTT measurement at a client

Usage of Network Links
Conclusion

• Using on RFCs of RIP and RIPng (with added modifications) the DVRP is implemented for the ns-3 simulator.

• The protocol is still do not support,
  ➢ multicast updates,
  ➢ VLSM (routing table compression)
  ➢ Pull when a link breaks

• Such features are being implementing now.

• Can find the protocol @
  https://github.com/westlab/ns3dvrp
Thank you

Kindly forward your questing and concerns to Janaka Wijekoon by contacting janaka@west.sd.keio.ac.jp or janaka.wijekoon@gmail.com