Maximum Lifetime Routing in Wireless Ad Hoc networks II - Practice

Riku Jäntti
&
Usman Rafique

University of Vaasa, Finland

Objective

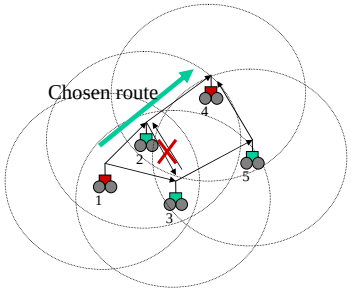
- Develop reactive network layer routing protocol that maximizes the operation time (life-time) of the worst node in the network
- Ensure backwards compatibility IETF AODV protocol
- Implement and test the protocol with IEEE802.11b ad hoc network

Contents

- AODV
- Reactive maximum life-time routing (battery-aware AODV)
- Implementation
- Testing
- Conclusions

Routing

- Routing strategies
  - Proactive:
    - Nodes maintain routing tables.
    - Tables need to be updated when topology changes.
  - Reactive
    - Route is determined based on demand.
    - If there is no traffic on particular route, no routing table entries are stored.



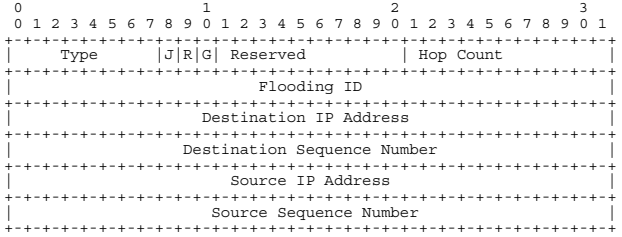
IETF AODV Protocol

Ad Hoc On Demand Distance Vector

- Reactive routing:
  - If there is no route between source and destination RREQ packet is flooded in the network
  - Node that knows route to destination answers by sending RREP packet
  - Signaling packets are normal UDP/IP packets

Route request

RREQ Header:

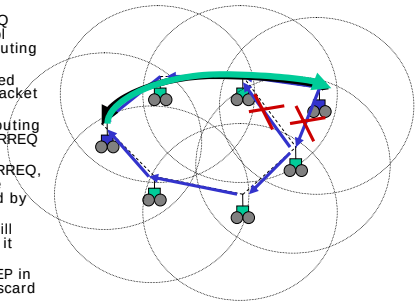


Route request

- Type = 1 "standard" AODV
- J,R reserved for multicast
- G: Gratuitous RREP flag, if set RREP will be generated and unicast to source
- Reserved: Reserved for future use
- Flooding ID: Each RREQ packets is given unique ID number to detect multiple copies
- Two counters for virtual time
  - Orginator sequence number: When a packet generates RREQ it increases the counter by one
  - Destination sequence number: The last known value of the counter at the destination node
- Hop count: Number of hops in a path from source to destination.
  - When RREQ packet is forwarded, the hop count is increased by one

Determining the route

- If there is no route, a source will broadcast RREQ packet
- When a node receives an RREQ packet it will store the protocol header information into the routing table.
- If the node already has received RREQ with the same ID, the packet will be discarded.
- If the ID number is greater, routing table will be updated and the RREQ packet forwarded (broadcast).
- If the destination node hears RREQ, it will issue RREP which will be unicasted using the route used by the RREQ.
- When a node hears RREP, it will store the routing table entries it made earlier.
- If a node does not receive RREP in some time window T, it will discard the routing table entries.
- When source receives the RREP it will start data transmission using the route.

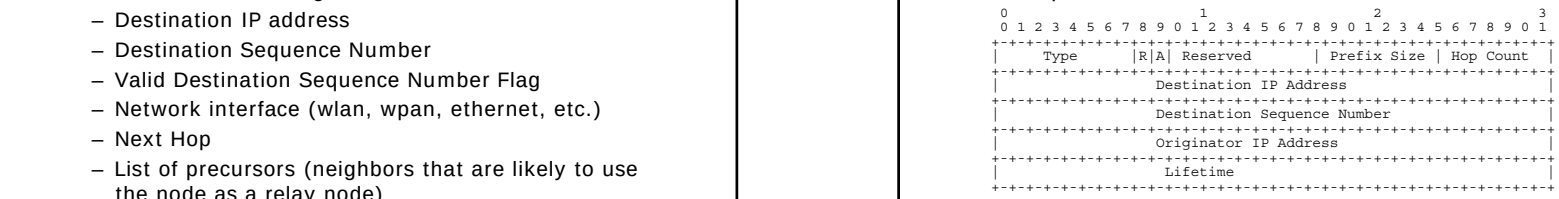


Routing table

- Contents of the routing table:
  - Destination IP address
  - Destination Sequence Number
  - Valid Destination Sequence Number Flag
  - Network interface (wlan, wpan, ethernet, etc.)
  - Next Hop
  - List of precursors (neighbors that are likely to use the node as a relay node)
  - Lifetime

Route response

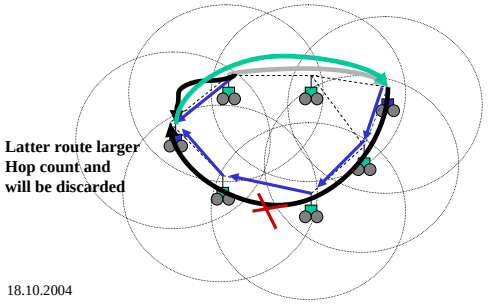
- RREP packet



  - R: Repair flag
  - A: Acknowledgement required
  - Lifetime: How long the route will be maintained in the intermediate nodes.

Route response

- If a node knows the route to the destination, it will issue RREP packet.
- If source gets multiple copies of RREP it will determine which route to use based on the hop count field

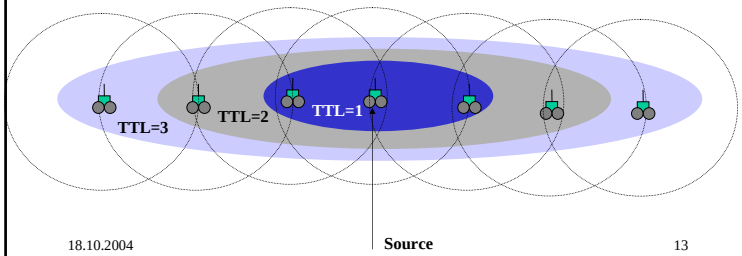


HELLO exchange

- In order to detect topology changes, nodes periodically broadcast HELLO messages, once in HELLO_INTERVAL.
- If a node does not receive HELLO from a neighbor used in some route, it will issue a RERR packet which will be unicasted to the source.
- When a node receives RERR it will remove the corresponding routing table information.
- When a source receives RERR it will issue a new RREQ packet.
- Due to the HELLO exchange, the neighbors of destination already know route to the destination and can issue the RREP packet.
- HELLO packet is simply RREP packet with time-to-live set TTL=1, Hop Count =0 and Sequence Number set to correspond to the node broadcasting the HELLO.

IETF AODV Protocol

- Flooding of RREQ causes lot of overhead.
  - Limit the propagation of the packets to control the time-to-live TTL field of the IP header of the RREQ.
  - When a router relays the packet it will subtract one from the TTL field. Packets with zero TTL are discarded.



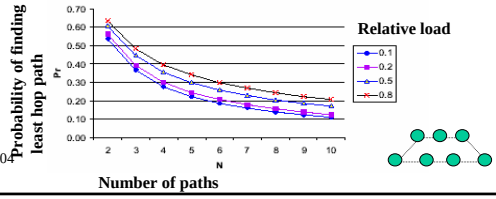
IETF AODV Protocol

- Congestion on the MAC protocol will cause random delays to the RREQ packets

=>The first packet to arrive destination is not necessarily the one corresponding to the shortest route.

  - A (destination) node should not discard a RREQ packet even if it has received RREQ with same ID before.

  => Multiple routes will be established based on which the source can select the best one.



IETF AODV Protocol

- The routing does not take into account the different battery capacities and traffic loads experienced by the nodes. => **Shortest route is not necessarily the best one.**
- The objective of our work is to modify the AODV protocol such that it can find the route maximizing the operation time of the worst node.
- Idea:
  - Each node estimates its lifetime
  - Lifetime of a path is the lifetime of the worst node on the path
  - Set up multiple paths
  - Source selects the path having the greatest lifetime

Expected life-time

- Lifetime of a node is estimated as follows

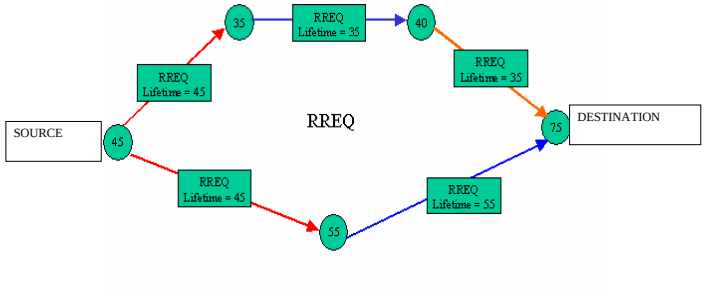
$$Lifetime = \frac{E_{Total}}{\phi_{Tx}P_{Tx} + \phi_{Rx}P_{Rx} + \phi_{Idle}P_{Idle}}$$

where

ϕ_{Tx}	Transmitted Traffic = Transmitted Bytes/ Data Rate / Time Interval
ϕ_{Rx}	Received Traffic = Receives Bytes/ Data Rate / Time Interval
E_{Total}	Total energy of the battery
P_{Tx}	Transmitter power consumption
P_{Rx}	Receiver power consumption
P_{Idle}	Power consumption in idle state $P_{Idle} = (1-\phi_{Tx}-\phi_{Rx})P_{Idle,radio} + P_{CPU}$

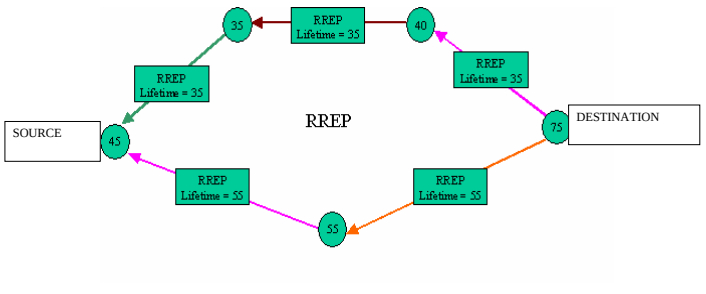
Determining the lifetime of a path

- Timeline:



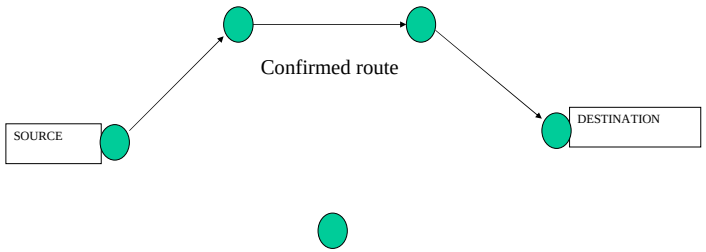
Determining the lifetime of a path

- Timeline:



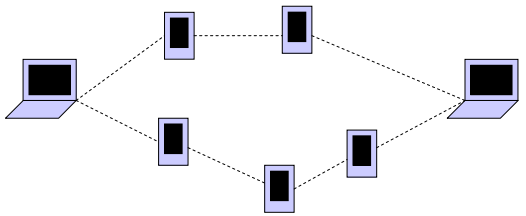
Determining the lifetime of a path

- Timeline:



Implementation

- Hardware
  - Laptops: 2 x Toshiba Satellite Model S1800 - 804
  - PDAs: 5 x iPAQ Pocket pc Model 3630
  - WLAN cards: 7 x Compaq Wireless LAN Version 4.0



Implementation

- Software
  - Linux operation system
    - Laptops:
      - RedHat 8.0
      - Linux Kernel Version:2.4.18-14.9
    - PDAs:
      - Familiar flavour of Linux for handhelds Version 0.7.1
      - Linux Kernel Version: 2.4.18-14.19-rmk6-pxa1-hh13
    - Driver for WLAN: wavlan_cs
  - Reference AODV implementation: Erik Nordström, Uppsala University AODV-UU v0.7.2

Life Time = Energy left / Power consumption rate

Power consumption rate = Transmission Power * Transmission time +
Reception power * Reception time +
Idle CPU power * Idle time

Following are derived from specifications given by Compaq wlan

Transmission power = 1.4W
Reception power =0.9W
Idle CPU power =0.05W

Transmission and Reception Time is calculated as follows

*Transmission time = (Bytes_Transmitted * 8) / (Time_interval * Bit rate)*

*Reception time = (Bytes_Received * 8) / (Time_interval * Bit rate)*

Idle time is calculated as follows

Idle time = Time_interval – ((Bytes_Transmitted + Bytes_Received) / Bit rate)

Energy left is calculated as follows

Energy left = % of Battery remaining * Total Battery Energy

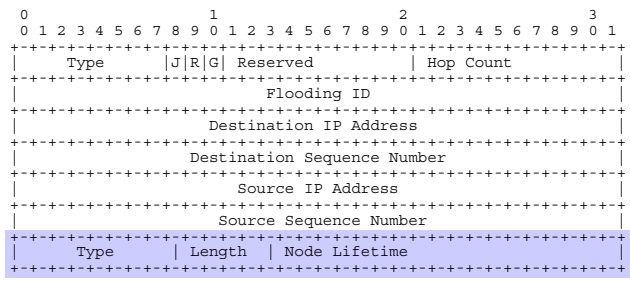
% of Battery left is read directly from /proc/hal/battery while total battery energy determined from specifications is

Total battery energy = 155520 Joules (Toshiba Laptop Specifications)

Bytes transmitted and received are read from /proc/net/dev.

Modified RREQ

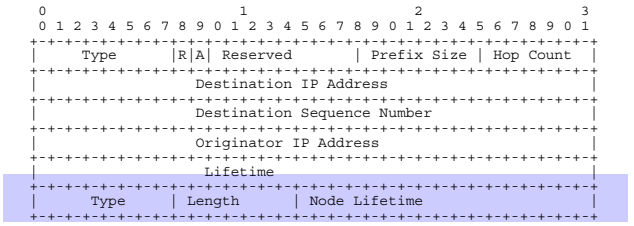
Modified RREQ



Extension field

Type: RREQ_EXT, which is defined as 2 in source code.
Length: Length of RREQ_EXT
Node Lifetime: Life time of the node

RREP_EXT



Modified RREQ

RREQ:

- Lifetime of a source has no significance on the routing. Consequently, source will set 0xFFFF as its Node_Lifetime
- After a packet is detected as a routing request packet, it is forwarded to the function *rreq_process()*.
- The packet header is parsed in function *read_rreq()* that is called from *rreq_process()*.
- Lifetime is changed in *read_rreq()* function.

Modified RREP and HELLO

RREP:

- When RREQ reach destination, RREP will be generated.
- The maximum of all the Node_Lifetime values of the path will be added to the header in *rrep_process()* function.

HELLO:

- HELLO packets are RREQ packets and Node_Lifetime is added to them as to RREP.

Routing table

- Routing table is updated by
  - *Hello_process()*
  - *Neighbout_add()*
  - *Rrep_process()*
  - *Rrep_process()*
- Life-time of the node needs to be considered in all of them

Modified Routing Table

```
struct rt_table {
    list_t l;
    u_int32_t dest_addr;     /* IP address of the destination */
    u_int32_t dest_seqno;    /* Destination sequence number */
    u_int32_t life_um;       /*Minimum lifetime of a route*/
    unsigned int ifindex;    /* Network interface index... */
    u_int32_t next_hop;      /* IP address of the next hop to the dest */
    u_int8_t hcnt;           /* Distance (in hops) to the destination */
    u_int16_t flags;         /* Routing flags */
    u_int8_t state;          /* The state of this entry */
    struct timer rt_timer;   /* The timer associated with this entry */
    struct timer ack_timer;  /* RREP_ack timer for this destination */
    struct timer hello_timer; /* HELLO_ timer for this destination */
    struct timeval last_hello_time;
    u_int8_t hello_cnt;
    hash_value hash;
    int nprec;               /* Number of precursors */
    list_t precursors;       /* List of neighbors using the route */};
```

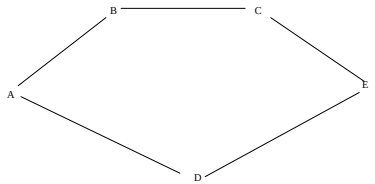
Minimum life-time of a route for a Source – Destination pair

Routing decision

- In original AODV implementation, routing decision is made when a RREP message has arrived at the source node.
- Upon arrival of first RREP packet, a new entry in routing table is made and there is no question about route selection.
- Upon subsequent arrival of RREP packets that are resulted from different routes, route selection is needed.
- If new route reply packet has larger hop count , it is neglected otherwise route is updated with this new RREP. *Rt_table_insert()* function.
- In order to select the maximum life-time route, *Rt_table_insert()* function has to be modified.

Testing

- Network topology



- Outside: Nodes were placed around a pentagonal shaped building

Testing

- Raippaluoto, Vaasa

Testing

- Insidie
  - EMC room
  - Topology is obtained by preventing certain roots to be formed by using iptables:
  - For example following set of commands must be used at node A if mac address of C and E are 00:02:A5:2E.BD:80 and 00:02:A5:2E.BD:81respectively.
  - *iptables –t filter –A INPUT –m mac –mac-source 00:02:A5:2E.BD:80 –j DROP*
  - *iptables –t filter –A INPUT –m mac –mac-source 00:02:A5:2E.BD:81 –j DROP*

Testing

- Test result in laboratory (EMC room)

Node	Remaining Energy
A	200
B	200
C	200
D	10
E	200

First route is broken in original AODV after 10 minutes.
In the modified, Battery, protocol, the first route breaks after 190 minutes.

Conlusions

- Benefits of our proposed Battery-aware AODV protocol
  - Long lasting routes
    - Less rerouting: Reduced signalling overhead and energy consumption
    - Long operation time of nodes
  - Hidden congestion control: Life-time metric takes the traffic volumes into account.
- Drawbacks
  - Longer routes can be utilized
    - Aggregate power consumption increases
    - Packet delay increases
    - If nodes are mobile, long routes last shorter time
  - Additional signaling overhead (32 bits per message)

Conclusions

- The proposed protocol is suitable for small scale networks where the nodes are relatively stable (slow mobility)
- In order to take mobility into account, life-time estimates should include link stability
- Current research:
  - Joint transmit power control (topology control) and routing
  - Implementations with sensor networks (nodes are embedded devices)