Maniac Forwarding Strategy

Team
Arunkumar Jayakeerthy and Taha Ben Brahim
Dept of CSSE,
Auburn University.

Solution Summary

Motivation

The main intention behind our solution is to maximize our chances of getting packets destined for our nodes. At the same time we try to minimize the number of packets that we forward. While this is true, we need to take care of maintaining good reputation among neighbors.

Restrictions

- The algorithm/criteria which is used by MANIAC to evaluate the reputation of a given node is unknown. To be more specific the upper threshold on the percentage of packets that a node can drop instead of forwarding and still maintain good reputation is not known. However once the node is tagged "free rider" there is a possibility that MANIAC API might inform the neighbors about it instructing them not ti forward any packets to the tagged node (possibly by modifying the "one-hop-list" of the neighboring nodes.

  The strategy must not assume anything about the reputation mechanism, but instead it must work in a probabilistic manner.

- The node does not know the strategy of the neighboring nodes.

Strategy

Based on the above restrictions, we propose the following strategy. We keep in mind that whenever a packet has to travel across multiple hops, given the heterogeneity of the nodes and their forwarding decisions, the probability of the packet successfully reaching its destination is lower than the packet that has to traverse fewer hops.

Looking at it from our perspective,

- **In case of a packet destined to a distant node** - when we forward a packet which is farther away from our node, we have a certain percentage of confidence that it will not reach its destination i.e. with the same probability we can say that the node to which the packet is destined is unlikely to get 10 points out of that packet. So it is not very useful for us to loose reputation over the packet.

- **In case of a packet destined to a nearby node** - when we forward a packet to a node which is near by our node (say 1 or 2 hops), the probability that the packet will reach its destination depends greatly on our decision i.e. if we do not drop the packet, there is a good chance that the packet will make it to the destination node, thus allowing it to gain 10 points. In other words the packet is worth losing our reputation for i.e. it may be dropped.

Thus our strategy is based on the principle – **"act more on the packets which have more significance to self than those on the packets which are insignificant"**

Implementation

We make use of the current node's routing table i.e. the route_info structure to obtain the number-of-hops to every destination in the MANET and use the hop count to obtain a binary exponential probability with which we need to drop the packet.

For example: For a node that is one hop away we drop with a probability of 1/2, for a node which is 2 hops away, we drop with a probability of ¼ and so on (with a maximum of 4 hops).

Notice that the node behaves differently to different destinations. Thus by avoiding static behavior we decrease the possibility of losing reputation while we are dropping packets

Issues:

Since there is a high probability of dropping packets to immediate neighbor, in cases where we have an inflow of only packets destined to our neighbors, then we end up dropping almost all the packets, there by risking lose of reputation.

Solution: We believe such a possibility occurs when our node is positioned in a location where it forms a bridge between the RN and the rest (most of the remaining part) of the network. This can be simple avoided by changing the current position of the node.

Open questions

1. Is the reputation rating criteria public?
2. How will MANIAC API inform nodes about rogue nodes?
3. The challenge says there are 3 kinds of traffic flows to each team. Other than realtime and non-realtime traffic, how is this information relevant to the forwarding strategy. I mean to ask that at this level TCP or UDP does not matter. Am I missing some information here?