Exploring Mitigation Strategies for Common MANET Network Layer Attacks

Cameron Oakley
School of Computing and Design, California State University, Monterey Bay

Undergraduate Research Opportunities Center
UROC
California State University, Monterey Bay

Introduction

Mobile Adhoc Networks (MANETs) are a type of wireless architecture that breaks away from the confines of typical wireless networks, with clients requesting network resources from an access point. In these MANETs, nodes mesh together and act as clients, requesting network resources, and as routers, forwarding traffic to neighboring nodes. This architecture relies on nodes working together to ensure the network is operating efficiently. **The network relies on a level of trust that can be easily exploited by malicious actors**. A malicious node has the ability to negatively impact the reliability of the network by **dropping all packets** that it receives. The focus of this work is to examine these malicious nodes that threaten the reliability of a MANET and develop an efficient mitigation strategy.

Background

Blackhole

- a type of attack where a malicious node drops all packets that it receives.

B.A.T.M.A.N. Advanced (**batman-adv**)

- is an open-source layer 2 routing protocol.

Source node

- node that creates the original request and sends it in the direction of the destination node.

Destination node

- recipient of a message sent in the network.
- can be several or thousands of hops away from the source node.

ACK

- a message used to confirm a successful data transfer over the wireless medium.

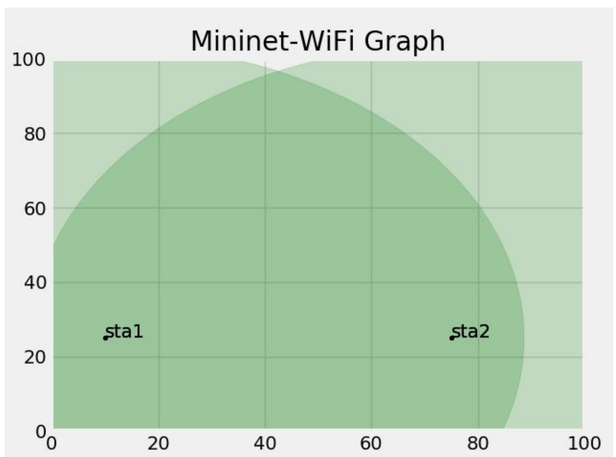

Figure 1a: Simple MANET with only 2 nodes.

```
mininet-wifi> sta1 ping sta2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=25.0 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=8.27 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=3.82 ms
```

Figure 1b: Connectivity between the 2 nodes.

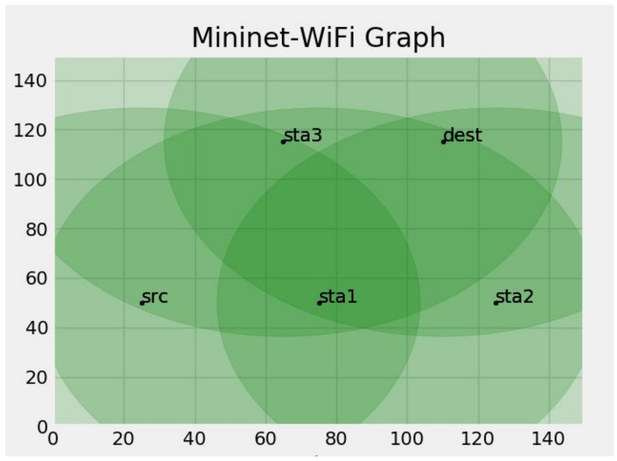

Figure 2a: MANET with 5 nodes. Source and Destination nodes are labeled.

```
mininet-wifi> src ping dest
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
```

Figure 2b: We need a routing algorithm to route information between nodes.

Current Work

- batman-adv is one of the most popular routing algorithms for MANETs.
- batman-adv's routing algorithm determines the two best links and forwards packets based on the throughput over the wireless medium.
- A malicious node could exploit this by lying about the throughput.
- Novel approach discussed later…

```
1  static int batadv_v_neigh_cmp(struct batadv_neigh_node *neigh1,
2                                struct batadv_hard_iface *if_outgoing1,
3                                struct batadv_neigh_node *neigh2,
4                                struct batadv_hard_iface *if_outgoing2)
5  {
6      struct batadv_neigh_ifinfo *ifinfo1, *ifinfo2;
7      int ret = 0;
8
9      ifinfo1 = batadv_neigh_ifinfo_get(neigh1, if_outgoing1);
10     if (!ifinfo1)
11         goto err_ifinfo1;
12
13     ifinfo2 = batadv_neigh_ifinfo_get(neigh2, if_outgoing2);
14     if (!ifinfo2)
15         goto err_ifinfo2;
16
17     ret = ifinfo1->bat_v.throughput - ifinfo2->bat_v.throughput;
18
19     batadv_neigh_ifinfo_put(ifinfo2);
20 err_ifinfo2:
21     batadv_neigh_ifinfo_put(ifinfo1);
22 err_ifinfo1:
23     return ret;
24 }
```

Note that this code was not written by me. See reference [1].

Figure 3: source code of batman-adv. This code is determining the throughput to two nodes over the wireless medium.

Proposed Directions

A Novel Approach

- batman-adv is focused on the speed of getting a packet through the network. To ensure there is reliability in the network, we need to mitigate the chance of blackhole attacks.
- Sacrifice performance for reliability.
- Create several tables in memory that are used to improve the reliability of routing.
- Audit these tables to decrease routing overhead and improve performance.
- Determine if a neighboring nodes can be trusted.

Trust-based Table

- Have a table in memory used to keep a list of currently known neighbor nodes and their MACs.
- A hash table that uses MACs as a key and has an associated integer trust value.
- All tables are audited based on some timer.
- Nodes increase trust values after successful bidirectional communications.
- During routing, when presented with a tie-breaker choose the route with the higher trust value.

Trusting Neighbor Nodes

- Periodically send false packets to test the reliability of neighboring nodes.
- False packets contain a spoofed source address and destination address of the source node.
- If the packet is forwarded to the source node from the neighboring node, then we know that the link is good and that the node is not dropping packets.
- If the packet is not received, this means that the connection is either bad or the node is dropping packets.
- Update trust table increasing or decreasing the trust level of the neighboring node.

Literature Cited

[1] Open-Mesh. (2023). batman-adv. GitHub. https://github.com/open-mesh-mirror/batman-adv/blob/028bce4a802301ba0abf274a01919b9bf07fdef8/net/batman-adv/bat_v.c#L451

Acknowledgments

I would like to thank Dr. Sam Ogden for his work in mentoring me through the research process. Also, I'd like to thank the CSUMB UROC program for the funding which made this project possible.