Black hole attack in mobile Ad Hoc networks

Conference Paper · April 2004

DOI: 10.1145/986537.986560 · Source: DBLP

Mohammad M. Shurman
Jordan University of Science and Technology

Seong-Moo Yoo
University of Alabama in Huntsville

Seungjin Park
Hanyang University

April 2004
Proceedings of the 42nd annual Southeast regional conference ACM-SE 42
Publisher: ACM Press

Black Hole Attack in Mobile Ad Hoc Networks

Mohammad Al-Shurman and Seong-Moo Yoo
Electrical and Computer Engineering Department
The University of Alabama in Huntsville
Huntsville, Alabama 35899
E-mail: {al-shum,yoos}@eng.uah.edu

Seungjin Park
Department of Computer Science
Michigan Technological University
Houghton, Michigan 49930
E-mail: spark@mtu.edu

Abstract

The black hole problem is one of the security attacks that occur in *mobile ad hoc networks* (MANETs). We present two possible solutions. The first is to find more than one route to the destination. The second is to exploit the packet sequence number included in any packet header. Computer simulation shows that compared to the original *ad hoc on-demand distance vector* (AODV) routing scheme, the second solution can verify 75% to 98% of the route to the destination depending on the pause times at a minimum cost of the delay in the networks.

Introduction

In recent years the concern over the security of computer networks has been widely discussed and popularized. The discussion has, however, typically involved only static and wired networking while the mobile or ad-hoc networking issues have not been handled extensively. The emergence of such new networking approaches sets new challenges even for the fundamentals of routing since the *mobile ad-hoc networks* (MANET) are significantly different from the wired networks. Moreover, the traditional routing protocols of the Internet have been designed for routing the traffic between wired hosts connected to a static backbone; thus, they cannot be applied to ad hoc networks because the basic idea of such networks is mobility with dynamic topology [1].

Black hole problem in MANETS [2] is a serious security problem to be solved. In this problem, a malicious node uses the routing protocol to advertise itself as having the shortest path to the node whose packets it wants to intercept. In flooding based protocol, if the malicious reply reaches the requesting node before the reply from the actual node, a forged route has been created. This malicious node then can choose whether to drop the packets to perform a denial-of-service attack or to use its place on the route as the first step in a man-in-the-middle attack.

Proposed Solutions for Black Hole

The first proposed solution here for black hole is to find more than one route to the destination (redundant routes, at least three different routes). Then, the source node unicasts a *ping* packet to the destination using these three routes (we should assign different packet IDs and sequence number, so any node who receive the first packet will not drop the second one if it exists in both paths). The receiver and the malicious in addition to any intermediate node might have a route to the destination will reply to this ping request. The source will check those acknowledgements, and process them in order to figure out which one is not safe and might have the malicious node.

The second proposed solution exploits the packet sequence number included in any packet header. The node in this situation needs to have two extra tables; the first table consists of the sequence numbers of the last packet sent to the every node in the network, and the second table for the sequence number received from every sender. During the RREP phase, the intermediate or the destination node must include the sequence number of last packet received from the source that initiates RREQ. Once the source receives this RREP, it will extract the last sequence number and then compare it with the value saved in its table. If it matches the transmission will take place. If not, this replied node is a malicious node, so an alarm message will be broadcast to warn the network about this node.

First Solution

In this solution, the sender node needs to verify the authenticity of the node that initiates the RREP packet by utilizing the network redundancy. Since any packet can be arrived to the destination through many redundant paths, the idea of this solution is to wait for the RREP packet to arrive from more than two nodes. During this time the sender node will buffer its packets until a safe route is identified. Once a safe route has identified, these buffered packets will be transmitted. When a RREP arrives to the source, it will extract the full paths to the destinations and wait for another RREP.

Two or more of these nodes must have some shared hops (in ad hoc networks, the redundant paths in most of the time have some shared hops or nodes). From these shared hops the source node can recognize the safe route to the destination. If no shared nodes appear to be in these redundant routes, the sender will wait for another RREP until a route with shared nodes identified or routing timer expired.

This solution can guarantee to find a safe route to the destination, but the main drawback is the time delay. Many RREP packets have to be received and processed by the source. In addition, if there are no shared nodes or hops between the routes, the packets will never been sent.

Second Solution

Every packet in MANETs has a unique sequence number. This number is an increasing value, i.e., the next packet must have higher value that the current packet sequence number. The node in regular routing protocols keeps the last packet sequence number that it has received and uses it to check if the received packet was received before from the same originating source or not.

In this solution, every node needs to have two additional small-sized tables; one to keep *last-packet-sequence-numbers* for the last packet sent to every node and the other to keep *last-*

packet-sequence-numbers for the last packet received from every node. These tables are updated when any packet arrived or transmitted. The sender broadcasts the RREQ packet to its neighbors. Once this RREQ reach the destination, it will initiate a RREP to the source, and this RREP will contain the *last-packet-sequence-numbers* received from this source. When an intermediate node has a route to the destination and receives this RREQ, it will reply to the sender with a RREP contains the *last-packet-sequence-numbers* received from the source by this intermediate node.

This solution provides a fast and reliable way to identify the suspicious reply. No overhead will be added to the channel because the sequence number itself is included in every packet in the base protocol.

Computer Simulation

We used the network simulator (ns-2) [3, 4]. A hypothetical network was constructed for the simulation purpose and then monitored for a number of parameters. We simulate our model for 50 nodes. Pause time is varied from 0 to 900 sec. Each mobile node in the MANET is assigned an initial position within the simulation dimensions (1000×1000) meters and joins the network at a random time. The packets are generated using CBR with rate of 4 packets per sec. The simulation takes place for 900 seconds every run. Nodes are normally distributed when initialized, and the initial position for the node is specified in a movement scenario file created for the simulation using a feature within ns-2. The nodes move randomly among the simulation area.

We simulate for both Solution 1 and Solution 2 with relative to the base protocol AODV. For both solutions, since we did not inject any attacker node (no security model is implemented in ns-2), we tried to verify the route to the destination and drop the request if we could not verify the route. We can see that Solution 1, in the best case, could verify 80% only of the routing packets, while Solution 2 could verify up to 99%.

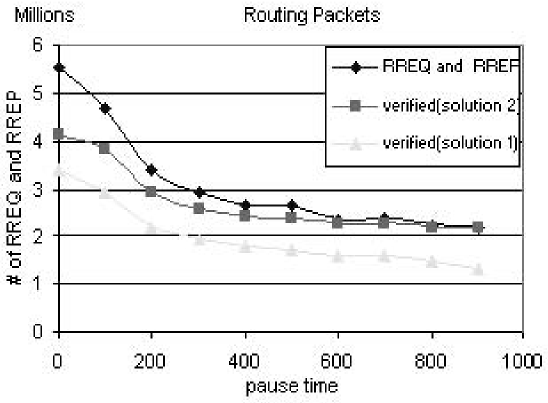

Fig 1: Routing packets in plain AODV and with the solutions

For Solution 1, we tried to find more than one path and process those paths for shared nodes. If at least one shared node is found, the path is verified. If no shared node is found, we will wait a specified amount of time (1 sec) to receive more RREP. If no other RREP arrived, we conclude that the destination cannot be verified. In Solution 2, the nodes need to exchange some packets during setting up the network to fill their *last-packet-sequence-numbers* tables. We used the sequence number to verify the identity of the nodes.

Refer to Figure 1. With zero pause time, Solution 1 could verify only 60% of the routes, while Solution 2 could verify 75%. For both solutions, this percentage is increased as the pause time is increased. For 900 sec pause time, Solution 1 could verify 80% of the routes and Solution 2 could verify 98% of the requests.

In Figure 2, we will study the delay in the network. The delay in Solution 1 is very large compared to the base AODV. Solution 2 shows a very close results to the base AODV, and the difference is negligible.

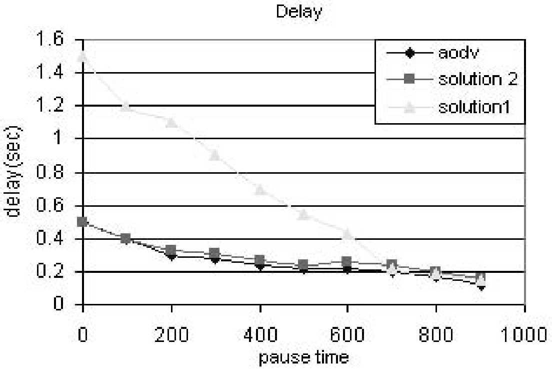

Fig 2: Routing delay in plain AODV and with the solutions

From the above results we can see that we can avoid the black hole problem. Solution 1 has a longer delay and lower number of verified routes than Solution 2, but Solution 1 appears to be more secure than Solution 2 in the sense that, in Solution 2, the attacker node can listen to the channel and update the tables for the last packet sequence number. We are trying to merge both solutions to find a better solution for the black hole problem.

Future Work

In this paper we have proposed two solutions for the black hole problem. Here, we have studied only one node attack to be in the route (not a group of attackers). The group attack for this problem should be studied.

References

1. Janne Lundberg, Helsinki University of technology, "Routing Security in Ad Hoc Networks" **http://citeseer.nj.nec.com/400961.html**.
2. H. Deng, W. Li, and Dharma P. Agrawal, "Routing Security in Ad Hoc Networks,"IEEE Communications Magazine, Special Topics on Security in Telecommunication Networks, Vol. 40, No. 10, October 2002, pp. 70-75.
3. University of California and Lawrence Berkeley Laboratory, AODV source code for network simulator, 1997.
4. Network Simulator Official Site for Package Distribution, web reference, **http://www.isi.edu/nsnam/ns**.