Vol. 1, Issue III, April 2013
ISSN 2320-6802

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS.....

Catching Packet Droppers and Modifiers in Wireless Sensor Networks

[1]Prof. Borkar B.S., [2]Nilesh Madke

Department of Information Technology, Amrutvahini College of Engineering.
Sangamner, Pune University, Maharashtra, India, Pin no.422605.
borkar.bharat@gmail.com[1], 14nileshmadke@gmail.com[2]

Abstract*: Packet dropping and modification are common attacks that can be launched by an adversary to disrupt communication in wireless multi-hop sensor networks. Many schemes have been proposed to mitigate the attacks but none can effectively and efficiently identify the intruders. To address the problem, we propose a simple yet effective scheme, which can identify misbehaving forwarders that drop or modify packets. Extensive analysis and simulations using ns2 simulator have been conducted and verified the effectiveness and efficiency of the scheme.*
Keywords: *Sink Node, node categorization algorithm, Wireless multi-hop Sensor Networks, heuristic ranking algorithms, Node ranking algorithm comma.*

1. INTRODUCTION

In a wireless sensor network, sensor nodes monitor the environment, detect events of interest, produce data and collaborate in forwarding the data towards a sink, which could be a gateway, base station, storage node, or querying user. A sensor network is often deployed in an unattended and hostile environment to perform the monitoring and data collection tasks. When it is deployed in such an environment, it lacks physical protection and is subject to node compromise. After compromising one or multiple sensor nodes, an adversary may launch various attacks to disrupt the in-network communication. Among these attacks, two common ones are *dropping packets* and *modifying packets*, i.e., compromised nodes drop or modify the packets that they are supposed to forward. To deal with packet droppers, a widely adopted countermeasure is multi-path forwarding in which each packet is forwarded along multiple redundant paths and hence packet dropping in some but not all of these paths can be tolerated. This scheme introduces high extra communication overhead. Another category of countermeasures is to monitor the behavior of forwarding nodes. However, these schemes are subject to high energy cost incurred by the promiscuous operating mode of wireless interface. To deal with packet modifiers, most of existing countermeasures are to filter modified messages within a certain number of hops. However, without identifying packet droppers and modifiers, these countermeasures cannot fully solve the packet modification problems because the compromised nodes can continue attacking the network without being caught. To identify packet modifiers, Ye et al. recently proposed a probabilistic nested marking (PNM) scheme to identify packet modifiers with a certain probability. However, the PNM scheme cannot be used together with the false packet filtering schemes, because the filtering schemes will drop the modified packets which should be used by the PNM scheme as evidences to infer packet modifiers. This degrades the efficiency of deploying the PNM scheme. In this paper, we propose a simple yet effective scheme to catch both packet droppers and modifiers. According to the scheme, a dynamic routing tree rooted at the sink is first established. When sensor data is transmitted along the tree structure towards the sink, each packet sender or forwarder adds a small number of extra bits, which is called packet marks, to the packet. The format of the small packet marks is deliberately designed such that the sink can obtain very useful information from the marks. Specifically, based on the packet marks, the sink can figure out the dropping rate associated with every sensor node, and then run our proposed *node categorization algorithm* to identify nodes that are droppers modifiers for sure or are suspicious droppers/modifiers. As the tree structure dynamically changes every certain time

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

interval, behaviors of sensor nodes can be observed in a large variety of scenarios. As the information of node behaviors has been accumulated, the sink periodically run our proposed *heuristic ranking algorithms* to identify most likely bad nodes from suspiciously bad nodes. This way, most of the bad nodes can be gradually identified with small false positive.

2. EXISTING SYSTEM

Existing cooperative caching schemes for the Web environment can be classified as message-based, directory-based, hash based, or router-based [1].
The cache array routing protocol is the most notable hash based cooperative caching protocol. The rationale behind CARP constitutes load distribution by hash routing among Web proxy cache arrays.
A mobile node doesn't know whether the data source or some other nodes serve its request. If multiple data sources exist, or if the mobile node doesn't know where the data source is, Hybrid Cache might not be a good option.
In addition, caching nodes outside the path between the requesting node and the data source might not be able to share cache information with the requesting node.

LIMITATIONS OF EXISTING SYSTEM

Existing cooperative caching schemes for the Web environment can be classified as message-based, directory-based, hash based, or router-based [4].
The cache array routing protocol is the most notable hash based cooperative caching protocol. The rationale behind CARP constitutes load distribution by hash routing among Web proxy cache arrays.

3. PROPOSED SYSTEM

We simulate our caching algorithms in different ad hoc network scenarios and compare them with other caching schemes, showing that our solution succeeds in creating the desired content diversity, thus leading to a resource-efficient information access [10].
Data caching strategy for ad hoc networks whose nodes exchange information items in a peer-to-peer fashion. These decisions are made depending on the perceived "presence" of the content in the nodes proximity, whose estimation does not cause any additional overhead to the information sharing system [3].
However, the solution that was proposed is based on the formation of an overlay network composed of "mediator" nodes, and it is only fitted to static connected networks with stable links among nodes.
We proposed to mitigate or tolerate such attacks, but very few can effectively and efficiently identify the intruders. To address this problem, we propose a simple yet effective scheme, which can identify misbehaving forwarders that drop or modify packets. Extensive analysis and simulations have been conducted to verify the effectiveness and efficiency of the scheme [2].

A. List of modules:

1. Node Configuration
a. Link Configuration
2. Sender Node
a. Packet Splitting
b. Send Packets to Intermediate
3. Intermediate Node
a. Send Packets to Sink
b. Modify or Drop
4. Sink
a. Verify
b. Merge Packets
c. Categorization And Ranking

B. Module Description:

I. Node Configuration

a. Link Configuration

In this module Nodes are configured based on number of nodes in group. We create the network group by connecting nodes to sink. Link configuration means connecting the nodes and intermediate nodes to the sink.

II. Sender Node

a. Packet Splitting

In this module, Sender selects the file which is to be sent. And then it split into the number of packets based on the size for adding some bits in it.

b. Send Packets to Intermediate

And then it encrypts all the splitted packets. And then sender adds some bits to each encrypted packets before sending that. Bit Addition for each packet is identification for sender. After adding of bits to each packet, it sends the packets to the nearest node or intermediate node.

III. Intermediate Node

a. Send Packets to Sink

In this module, the intermediate node receives Packets from the sender. After receiving all packets

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS.....

from sender, it encrypts all packets again for authentication. Before sending to sink, intermediate add some bits to each packet for node identification. After adding some bits from intermediate, it sends all packets to the sink [7].

b. Modify or Drop

Before sending all packets to sink, packets dropping or packets modifying may be occur in intermediate.

IV. Sink

a. Verify

In this module, Sink receives all packets from the sender node, and it verifies all packets which are dropped or not. And it also verifies the packets which are modified or not and it can identify the modifiers in the process based on the bit identification.

b. Merge Packets

After receiving all packets in sink, it decrypts all packets. After the decryption if there is no modified or dropped packets, it merge all packets. After merging, Sink can receive the original file.

c. Categorization And Ranking

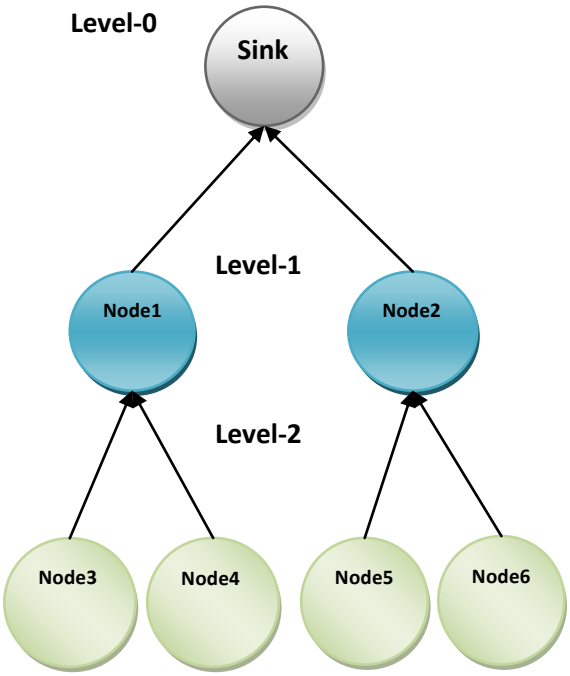
In this module Categorization and Ranking will be performed based on the node behavior. If there is any modification or drop of packets in node it assumes negative value for modifier or dropper. Sink performs Ranking for each node based on the Category of nodes. Sink gives ranking like Good, Temporarily Good, Suspiciously Bad, Bad based on the node behavior in the process [5].

ADVANTAGES OF PROPOSED SYSTEM

Data caching can significantly improve the efficiency of information access in a wireless ad hoc network by reducing the access latency and bandwidth usage.

Data for information retrieval and caching in mobile ad hoc networks, and it is built on an underlying routing protocol and requires the manual setting of a network wide "Caching zone".

STRUCTURE OF NODE CONFIGURATION



4. CONCLUSION

To address the problem of packet dropping and modification, we proposed a simple yet effective scheme to identify misbehaving forwarders that drop or modify packets. Extensive analysis and simulations have been conducted and verified the effectiveness of the proposed scheme in various scenarios [6][9][11][12].

5. REFERENCES

[1] R. Roman, J. Zhou, and J. Lopez, "Applying intrusion detection systems to wireless sensor networks," Third IEEE Annual ConsumerCommunications and Networking Conference (CCNC), pp. 640–644,Jan. 2006.

[2] S. Lee and Y. Choi, "A resilient packet-forwarding scheme against maliciously packet-dropping nodes in sensor networks," Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks(SASN), pp. 59–70, 2006.

[3] Z. Yu and Y. Guan, "A Dynamic En-route Scheme for Filtering False Data in Wireless Sensor Networks," IEEE Infocom 2006, April 2006.

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS.....

[4] F. Ye, H. Yang, and Z. Liu, "Catching Moles in Sensor Networks," IEEE International Conference on Distributed Computing Systems (ICDCS), June 2007.
[5] H.Chan and A. Perrig, "Security and Privacy in Sensor Networks," *IEEE Computer*, October 2003.
[6] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures," *the First IEEE International Workshop on Sensor Network Protocols and Applications*, pp. 113–127, May 2003.
[7] V. Bhuse, A. Gupta, and L. Lilien, "Dpdsn: Detection of packet-dropping attacks for wireless sensor networks," *In the Trusted Internet Workshop, International Conference on High Performance Computing*, December 2005.
[8] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *ACM MobiCom*, August 2000.
[9] R. Roman, J. Zhou, and J. Lopez, "Applying intrusion detection systems to wireless sensor networks," *Third IEEE conference , jan.2006*
[10] S. Lee and Y. Choi, "A resilient packet-forwarding scheme against maliciously packet-dropping nodes in sensor networks," *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks (SASN)*, pp. 59–70, 2006.
[11] Z. Yu and Y. Guan, "A Dynamic En-route Scheme for Filtering False Data in Wireless Sensor Networks," *IEEE Infocom 2006*, April 2006.
[12] F. Ye, H. Yang, and Z. Liu, "Catching Moles in Sensor Networks," *IEEE International Conference on Distributed Computing Systems (ICDCS)*, June 2007.