Volume 2, Issue 4, April 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Survey Report on Stealthy Packet Dropping

E.S.Phalguna Krishna*, M.Ganesh Karthik,
I.D.Krishna Chandra
Assistant Professor*
Department of CSE,
Sree Vidyanikethan Engg College,
Tirupati,India
phalgunakrishna@gmail.com

Abstract: ***Stealthy Attacks are routing attacks which "Minimize the cost to and visibility of the attacker, but which are about as harmful as Brute force attacks. These allow a skilled but not very powerful attacker to target communication networks in a way that makes it unlikely that he gets traced and caught. Stealthy packet dropping is a suite of four attacks misrouting, power control, identity delegation, and colluding collision that can be easily launched against multihop wireless ad hoc networks. Stealthy packet dropping disrupts the packet from reaching the destination through malicious behavior at an intermediate node. However, the malicious node gives the impression to its neighbors that it performs the legitimate forwarding action. Moreover, a legitimate node comes under suspicion. A popular method for detecting attacks in wireless networks is behavior-based detection performed by normal network nodes through overhearing the communication in their neighborhood. This leverages the open broadcast nature of wireless communication. An instantiation of this technology is local monitoring. We show that local monitoring, and the wider class of overhearing-based detection, cannot detect stealthy packet dropping attacks. Additionally, it mistakenly detects and isolates a legitimate node.***

Keywords: Stealthy Attack, Colluding Collision, Identity Delegation, Packet Misrouting, Multihop wireless adhoc networks

I. INTRODUCTION:

WIRELESS Ad hoc and Sensor Networks (WASN) are becoming an important platform in several domains, including military warfare and command and control of civilian critical infrastructure. They are especially attractive in scenarios where it is infeasible or expensive to deploy significant networking infrastructure. Examples in the military domain include monitoring of friendly and enemy forces, equipment and ammunition monitoring, targeting, and nuclear, biological, and chemical attack detection. Examples in the civilian domain include habitat monitoring, animal tracking, forest fire detection, disaster relief and rescue, oil industry management, and traffic control and monitoring.

Stealthy Attack:

Stealthy Attacks are routing attacks which "Minimize the cost to and visibility of the attacker, but which are about as harmful as Brute force attacks. These allow a skilled but not very powerful attacker to target communication networks in a way that makes it unlikely that he gets traced and caught.

There are two principle types of stealthy attack.

In first type, the adversary wishes to disconnect the network, whether this means a general partition of the network or the isolation of particular nodes.

In second type, the adversary modifies the routing information in order to hi-jack traffic from and to selected victim nodes.

Stealthy packet dropping is a suite of four attacks: Misrouting, Power control, Identity delegation, and Colluding collision that can be easily launched against multihop wireless ad hoc networks. Stealthy packet dropping disrupts the packet from reaching the destination through malicious behavior at an intermediate node. However, the malicious node gives the impression to its neighbors that it performs the legitimate forwarding action. Moreover, a legitimate node comes under suspicion. A popular method for detecting attacks in wireless networks is behavior-based detection performed by normal network nodes through overhearing the communication in their neighborhood. This leverages the open broadcast nature of wireless communication. An instantiation of this technology is local monitoring.

Local Monitoring:

A widely used instantiation of behavior-based detection is Local Monitoring. In local monitoring, nodes oversee part of the traffic going in and out of their neighbors. This leverages the open broadcast nature of wireless communication. Different types of checks are

done locally on the observed traffic to make a determination of malicious behavior.

For example, a node may check that its neighbor is forwarding a packet to the correct next-hop node, within acceptable delay bounds. For systems where arriving at a common view is important, the detecting node initiates a distributed protocol to disseminate the alarm. We call the existing approaches which follow this template **Baseline Local Monitoring (BLM)**. Many protocols have been built on top of BLM for intrusion detection building trust and reputation among nodes protecting against control and data traffic attacks and in building secure routing protocols.

In **BLM**, a group of nodes, called guard nodes perform local monitoring with the objective of detecting security attacks. The guard nodes are normal nodes in the network and perform their basic functionality in addition to monitoring. Monitoring implies verification that the packets are being faithfully forwarded without modification of the immutable parts of the packet, within acceptable delay bounds and to the appropriate next hop. If the volume of traffic is high (for data traffic in a loaded network), a guard verifies only a fraction of the packets.

II. SURVEY OF DIFFERENT STEALTHY ATTACKS:

Packet Misrouting:

Local monitoring has been demonstrated as a powerful technique for mitigating security attacks in multi-hop ad-hoc networks. In local monitoring, nodes overhear partial neighborhood communication to detect misbehavior such as packet drop or delay. However, local monitoring as presented in the literature is vulnerable to an attack called ***misrouting attack [1]***.

Packet misrouting disrupts the packet from reaching the destination by malicious behavior at an intermediate node. However, the malicious node gives the impression to its neighbors that it performed the legitimate forwarding action. Moreover, a legitimate node comes under suspicion. Through the misrouting attack, the attacker achieves the objective of disrupting the packet from reaching the destination by maliciously forwarding it to the wrong next-hop. The malicious node gives the impression to its neighbors capable of overseeing the packet that it has performed the required action (i.e., relaying the packet to the correct next-hop *en route* to the destination). This attack is applicable to unacknowledged packets. Due to the resource constraints of bandwidth and energy, much traffic in multi-hop ad hoc wireless networks (e.g., sensor networks) is unacknowledged. This is particularly true for the more common data traffic or broadcast control traffic [2] than for rare unicast control traffic.

Packet Misrouting Attack Description:

In packet misrouting, a malicious node relays the packet to the wrong next-hop, which results in a packet drop. Note that, in basic LM a node that receives a packet to relay without being in the route to the destination either drops the packet or sends a one-hop broadcast that it has no route to the destination. They argue that the latter case would be more expensive and dangerous since it gives malicious nodes valid excuses to drop packets. Therefore, they go with the first choice, even though it may result in some false accusations. In this attack, a malicious intermediate node achieves the same objective as if it were dropping a packet. However, none of the guard nodes using basic LM become any wiser due to the action. In addition, some legitimate node is accused of packet dropping.

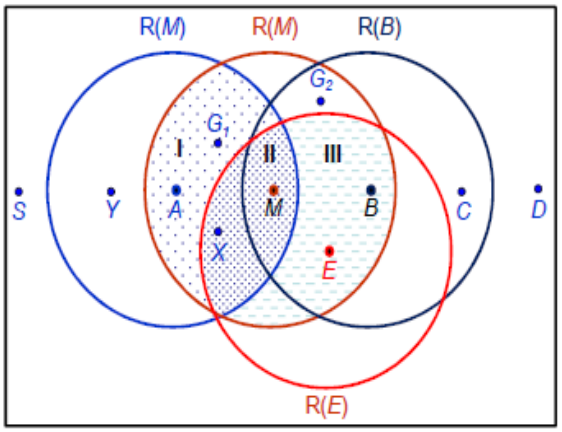

Fig: 1 Packet Misrouting Scenario

Consider the example scenario in Figure 1. Node *A* sends a packet to the malicious node *M* to be relayed to node *B*. Node *M* simply relays the packet to node *E* which is not in the route to the final destination of the packet. Node *E* drops the packet. The result is twofold: (i) node *M* successfully drops the packet without being detected since all the guards of *M* over *A-M* (regions I & II) have been satisfied by the transmission of *M->E*, and (ii) legitimate node *E* will be wrongly accused by its guards over *M->E* (regions II & III) as maliciously dropping the packet.

Mitigating Packet Misrouting:

To detect packet misrouting, the local monitoring mechanism has to incorporate additional functionality and information. The basic idea is to extend the knowledge at each guard to include the identity of the next-hop of the packet being relayed. This additional knowledge can be collected during route establishment. Many multi-hop wireless routing protocols provide this knowledge without any modification while some changes are necessary in others. The first class includes both reactive routing protocols such as Dynamic Source Routing (DSR)[3] and its variants and proactive routing protocols such as TinyOS beacon routing and Destination Sequenced Distance Vector routing (DSDV). In all source routing protocols, the packet header carries the identity of all the nodes in the route from the source to the destination. Therefore, no additional traffic is required to be generated for the guard nodes to be able to detect this kind of attack. Moreover, no additional information is required to be maintained at the guards since each packet carries the required information in its header. In TinyOS beacon routing, the base station periodically broadcasts a beacon to establish a breadth first search tree rooted at the base station. Each node within the transmission range of the base station overhears the beacon, sets its parent to be the

base station, sets the hop count to the base station to be one, and rebroadcasts the beacon. Each beacon carries the identity of the broadcasting node, the identity of its parent, and the hop count to the base station. Each guard overhearing the beacon broadcasting saves parent node identity for each neighbor. Later, when a node, says *B*, is sent a packet to relay, the guard of *B* can detect any misrouting by *B* since it knows the next-hop *en route* to the base station.

The second class of routing protocols requires modification to the protocol to build the next-hop information at the guards. Examples of these protocols are the reactive routing protocols that use control packet flooding of route requests (*REQ*) and route replies (*REP*) to establish the route between the source and the destination (e.g., LSR[1] and AODV[2]). In these protocols, when a source node desires to send a message to some destination node and does not already have a valid route to that destination, it initiates a route discovery process to locate the other node. It broadcasts a route request (*REQ*) packet to its neighbors, which then forward the request to their neighbors, and so on, until either the destination or an intermediate node with a "fresh enough" route to the destination is located. During the process of forwarding the *REQ*, intermediate nodes record in their route tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path. Once the *REQ* reaches the destination, the destination node responds by unicasting a route reply (*REP*) packet back to the neighbor from which it first received the *REQ*. As the *REP* traverses along the reverse path, nodes along this path set up forward route entries in their route tables which point to the node from which the *REP* came. Next, I show the required changes to the basic version of AODV [2] to enable the guards to build the necessary knowledge for detecting the misrouting attack. The idea behind the solution is that during route establishment, when the relation about which node to forward a packet between a given source-destination pair is determined, this information is broadcast by a neighbor to the guards which will be responsible for monitoring the node. To collect the next-hop identity information, the forwarder of the *REQ[3]* attaches the previous two hops to the *REQ* packet header. Let the previous hop of *M* be *A* for a route from source *S* to destination *D*, and the next hop from *M* be *B* (Figure 1). When *M* broadcasts the *REQ* received from *A*, it includes the identity of *A* and its own identity (*M*) in the *REQ* header <*S, D, REQ_id, A, M*>. When *B* and the other neighbors of *M* get the *REQ* from *M*, they keep in a *Verification Table* (*VT*) <*S, D, REQ_id, A, M, ->* (last field is currently blank). When *B* broadcasts the *REQ*, the common neighbors of *M* and *B* update their *VT* to include *B* <*S, D, REQ_id, A, M, B*>. When *B* receives a *REP* to be relayed to *M*, it includes in that *REP* the identity of the node that *M* needs to relay the *REP* to, which is *A* in this example. Therefore, all the guards of *M* now know that *M* not only needs to forward the *REP* but also that it should forward it to A and not any other neighbor.

Colluding Collision:

Stealthy packet dropping in multihop wireless sensor networks can be realized by the *colluding collision attack*. Colluding collision attack disrupts a packet from reaching its destination by malicious collusion at intermediate nodes. Moreover, the malicious nodes give the impression to their neighbors that they performed the legitimate forwarding action. Therefore, a legitimate node comes under suspicion. The MCC mitigation technique takes two steps. First, it extends the number of guards from only the common neighbors of the relaying node and the next hop to include all the neighbors of the relaying node.

Second, it creates a counter at each node for each neighbor which is responsible for counting the number of forwards by that neighbor. The latter technique makes use of the fact that under the colluding collision attack, the attacker tries to divide the neighbors into two sets having differing views in terms of the amount of forwarding traffic generated by the Attacker.

Colluding Collision Attack Description:

In many wireless sensor network deployment scenarios, the 802.11 MAC protocol RTS-CTS[4] mechanism that reduces frame collisions due to the hidden and exposed terminal problems are disabled for the sake of energy saving. This is also explained by the fact that packets in sensor networks are often quite small and fall below the threshold for packet length for which RTS/CTS is turned on. The attacker may exploit the absence of the RTS/CTS frames to launch a stealthy packet dropping attack through collision induced by a colluding node. The colluding node creates a collision in the vicinity of the next hop node at an opportune time. Consider the scenario shown in Figure 2. The malicious node *M1* receives a packet from *S* to be relayed to *T*. Node *M1* coordinates its transmission with a transmission generated by its colluding partner *M2* to *T*. This simultaneous transmission creates a collision at *T*, which prevents it from correctly receiving the packet relayed by *M1*. The damage caused by this attack is threefold: (i) *M1* successfully drops the packet due to a collision at *T*, (ii) node *M1* evades detection, and (iii) node *T* is accused of dropping the packet by some of its guards over the link *M1*→*T* (the guards that are out of the range of *M2*, region I). Note that for *M1* to be able to send data to *T*, it is has to be a legitimate neighbor (compromised by attacker), otherwise, the attack would be considered a physical layer jamming [5], which is assumed to be detectable through techniques complementary. However, *M2* could be an external malicious node.

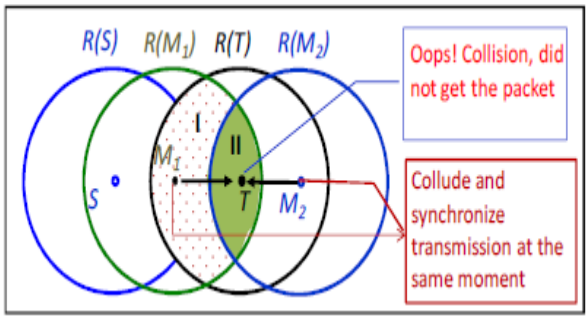

Fig: 2 Colluding Collision scenarios

The farther *S* is from *M2*, the better it is for the attacker since less number of guards of *M1* over *S→M1* is affected by the collision and therefore, the stealthier the attack. For this attack to succeed, the attacker must know the location of each neighbor and the detection confidence index γ. typically, security is not achieved through obfuscation and therefore, protocol parameters such as γ are taken to be known to all and location determination is routinely run upon deployment of nodes. When the number of guards of *M1* over the link *S→M1* that are affected by the colluding collision is greater or equal to γ, an intelligent attacker refrains from launching the attack since it will be detected and isolated by all its neighbors either directly or indirectly.

Mitigating Colluding Collision Attack:

The key observation behind the stealthy packet dropping using the colluding collision attack is that the attacker tries to defeat BLM by reducing the number of guards that can detect the malicious packet drop to less than γ. In the colluding collision attack shown in Figure 2, the attacker narrows the BLM guards of *M1* over the link *S→M1* that are capable to detect the packet drop into those that lie on the intersection between *R(S)* and *R(M2)* while the remaining majority of the guards (on the intersection of *R(S)* & *R(M1))* are satisfied. The countermeasure we propose against this attack is based on the observation that an adversary evades detection of dropping packets by allowing only a subset of guards to overhear the message being forwarded. Therefore, we expand the set of nodes that can guard from only the common neighbors of the node being monitored and its previous-hop node to include all the neighbors of the node being monitored. Since all neighbors are included in verifying the node, by definition, more neighbors will have the chance to see evidence of packet drop. The detection technique makes use of the fact that, under the colluding collision attack, neighbors have differing views of a node in terms of the volume of traffic it has forwarded and all the neighbors cannot be convinced by a single broadcast. To achieve this goal we need to introduce additional tasks for the nodes in the network. (i) Each node, say *X*, keeps a count of the number of messages each of its neighbors, say *Y*, had forwarded (*FCount(X,Y)*) over a predetermined time interval (T_{win}) and (ii) each node has to announce the number of packets it has forwarded over T_{win}. The adversary evades detection of packet dropping by allowing only a subset of guards to overhear the packet being forwarded. Thus, the subset of guards that had overheard the packet would have a higher count than the nodes that did not. By forcing a node to announce the number of messages it has forwarded over T_{win}, a malicious node would have the problem of satisfying two sets of neighbors that expect to hear different counts through a single broadcast.

A neighbor of a node *N* that collects the number of forwarded packets by *N* and compares the result with the count announced by *N* is called a *comparator* of *N*, denoted by *C (N)*. For any node *N* all nodes in radio range *R(N)* act as comparators of *N*. Recall that a guard of a node *B* over the link *Y→B*, has been defined in the BLM as any node that lies within the transmission range of both *Y* and *B*. Therefore, each guard of *N* over a certain link is a comparator of *N*; however, not every comparator of *N* is a guard of *N*. The function of a comparator is to count the total number of packets forwarded from the node within a time period. During certain time periods, node *N* may be required to announce the number of messages it has forwarded in that period. If a comparator's count is not within an acceptable range of the announced forward count, the comparator accuses the announcing node of dropping the packet. However, note that a suspicion would not be raised by a discrepancy of one due to natural losses. Detection is triggered only when the discrepancy crosses a predetermined threshold (FC_{th}). In order to reduce radio traffic, we do not require all nodes to announce their forward count in every period. Instead, a node must announce whenever it receives a request to do so.

For simplicity of exposition, we will consider that a discrepancy of a single packet is sufficient for detection. In Figure2, all the neighbors of *M1* except those that lie on area II would have one more count for the number of packets forwarded by *M1* as compared to the counters in the rest of *M1*'s comparators that lie on area II. The best the attacker can do is to satisfy the larger set, however, the nodes of the other set would detect the discrepancy and propagate the detection knowledge to the nodes of the other set. All the nodes of the smaller set would then directly isolate the malicious node. The nodes of the larger set indirectly isolate the malicious node if the number of nodes in the smaller set is not less than γ.

Identity Delegation:

Identity delegation is an act whereby an entity delegates his or her authority to use identity information to another entity. It has most often been implemented in enterprise environments, but previous studies have focused little on the dynamic data and access management model [6] as well as the design from a practical viewpoint. An identity delegation framework is described for using access tokens across security domains. The framework enables fine-grained access control with limited overhead cost for access management and permission assignment for delegated access. Bob and Alice are a married couple who are working for different companies. They are interested in financial planning in order to reduce their income taxes, repay their loans, and prepare for their future. Alice considers signing up for a financial planning service publicized by the Financial Planning Center (FPC) on the Internet. Such a service

would require her to display her and Bob's current financial status. Bob stores his pension records on his account managed by the Internet Service Provider (ISP) to view the information privately. The financial planning service Alice is considering requires both Bob's and Alice's pension records and information about their loans or mortgage in order to offer an ideal plan for their future lives.

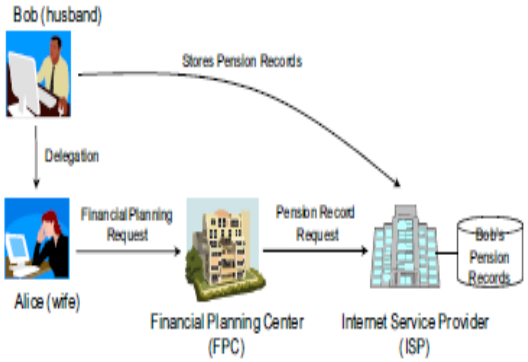

Fig: 3 Motivating Scenario

In this case, Bob's pension records at the ISP need to be provided to the FPC in a secure and privacy-preserving manner because the FPC can only provide Alice with the service at its site. If we assume that the FPC requests access to Bob's pension records managed by the ISP, the access request needs to be identified by the ISP in order to determine which entity, and what kind of data, is attempting to access the ISP in order to ensure an appropriate access control for the request. In addition, because Alice is the recipient of the financial service using Bob's pension records—in other words, one person is accessing and using another person's data—Alice's access needs to be authorized by Bob, even though they are married, before the ISP will accept her request for his data via the FPC. In order to do so, Alice needs to be provided with an appropriate privilege to access Bob's pension records and receive the service using the information at the FPC.

System Architecture:

The system architecture consisting of three entities (a user, an IDP, and an SP), and including component functions in each entity.

The IDP has five functions: the *access management (AM)*, the *authentication*, the *access control (AC)*, the *identity management (IdM)*, and the DM functions.

The AM receives access requests from other entities and invokes other functions such as the authentication, the IdM, and the DM for performing their corresponding operations according to the content of the requests.

The authentication function authenticates a user by means of a particular authentication method by referring to the information managed at the personal attribute DB.

The AC function controls access from other entities in compliance with access control policies that are stored in the *access control policy DB*.

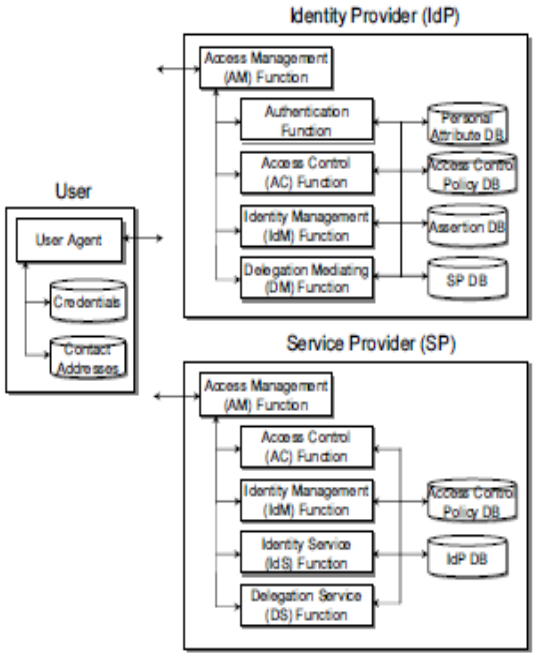

Fig: 4 System architecture

The IdM function provides a set of identity management sub-functions, such as identity federation with SPs and assertion issuance to SPs. The assertions produced by this function are stored in the assertion DB.

The DM function performs authorization for delegation by supporting the delegation request protocol and its operation. This function generates the basic information about a DAT and a DAA and provides it to the IdM function through which they are issued to the user or the SP. The SP has five functions, the *access management (AM)*, the *access control (AC)*, the *identity service (IdS)*, the *delegation service (DS)*, and the IdM functions.

The AM receives access requests from other entities for the services at the SP publicized on the Internet and invokes other functions for executing corresponding operations based on the content of the requests. This is similar to, but independent of the AM at the IDP.

The AC is a function that controls access from other entities in compliance with access control policies that are stored in the *access control policy DB*. The IdM is a function for identity management, such as identity federation and personal attribute retrieval. This function manages the IDP information with which the SP has a trust relationship. The information includes the identifier of the IDP and the addresses (in the *IdP DB*) at which the SP exchanges specific messages.

The IdS function provides a delegator with a particular service using his or her personal attributes obtained from an IDP.

III. CONCLUSION:

This paper describes the detailed survey of different Stealthy attacks in wireless ad hoc networks. It also gives the detailed description of detection and mitigation of each and every stealthy attacks.

REFERENCES

[1] A. A. Pirzada and C. McDonald, "Establishing Trust In Pure Ad-hoc Networks," in Proceedings of the 27th Australasian Computer Science Conference (ACSC 04), 26(1), pp. 47-54.

[2] C. E. Perkins and E. M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," in Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), 1999, pp. 90-100.

[3] D. Johnson, D. Maltz, and J. Broch, "The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks," in Ad Hoc Networking, Ed. Addison-Wesley, 2001.

[4] I. Khalil, S. Bagchi, and C. Nina-Rotaru, "DICAS: Detection, Diagnosis and Isolation of Control Attacks in Sensor Networks," in IEEE/CreateNet SecureComm, p.p. 89-100, September, 2005.

[5] R. Muraleedharan and L. A. Osadciw, "Jamming attack detection and countermeasures in wireless sensor network using ant system" in Wireless Sensing and Processing, proceedings of the SPIE, volume 6248, pp.62480G, 2006.

[6] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Rolebased access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.

