DIRECT TRUST-BASED DETECTION AND RECOVERY PROCESS OF JELLYFISH ATTACK IN MANET

Anjugam S *1 and Muthupriya V *2

*1 Department of Computer Science and Engineering, B.S Abdur Rahman University, Vandalur 48, Chennai
*2 Department of Computer Science and Engineering, B.S Abdur Rahman University, Vandalur 48, Chennai

Abstract— A Mobile ad-hoc Network (MANET) has unique characteristics such as dynamic topology, the absence of an administrative aspect and no constant infrastructure. These capabilities make it at risk of many attacks. JellyFish attack is a sort of Denial of service (DoS) attack, where in it produces the delay before the transmission and reception of information. This attack exploits the behavior of closed loop protocols which includes TCP and disturbs the communication procedure without obeying any protocol policies. As a result it is difficult to overcome across this attack, which ends up in reduced in network throughput. In proposed work a light-weight Direct Trust-based Detection (DTD) algorithm and Monitor, Detect, Rehabilitate (MrDR) technique are carried out, where these two algorithms are used to become aware of and remove a JellyFish node from an innovative transmission route. In DTD algorithm, trust value was estimated by way of each node to determine whether its neighbor node is JF-attacker or not by using over a time period. The Monitor, Detect, Rehabilitate (MrDR) technique, is applied as an enhancement to detect and eliminate jellyfish attack. Subsequently the development in the performance after removing the jellyfish attack is analyzed and compared using Network Simulator (NS2).

I. INTRODUCTION

In recent years, the technological advancements in hand-held device and improvements in deployment methods have made mobile ad hoc networks (MANETs) appealing for a variety of applications. Due to their inexpensive and on-the-fly deployment nature, MANETs are useful in rescue and emergency operations carried out during earthquake and flooding, military operations, vehicular ad hoc networks (VANETs), sensor networks, indoor and outdoor conferences, campus networks, robot networks etc. Despite the possible use of MANETs in a variety of applications, its practical implementation is limited due to its inherently insecure communication process and non-centralized architecture. As a result, providing security in MANETs has become a major concern for researchers.

Due to the following features a MANET is prone to many types of attack. Firstly, there is no central authority so detecting misbehaving nodes is difficult. Secondly, dynamic topology hinders the detection of malicious nodes, as they can disturb communication before leaving the network. Thirdly, packet losses are possible, as mobility and interferences between nodes dramatically increases packet loss. Fourthly, power life is limited as mobile nodes expend the energy through transmission and reception of the packets. Fifthly, a DoS attack is severe and prevents authentic users from having access to their legitimate services. Therefore, this type of attack prevents connections between users and degrades the performance of the network considerably. The security goal framework aims to provide the three pillars of security: Confidentiality, Integrity and Availability (CIA). Availability means that the network resources and data must be available at all times to authenticated users and it is one of the security pillars affected by this type of attack.

The main aim of this paper is to study the effects of protocol-compliant DoS attack called JellyFish on TCP-based MANETs and devise a countermeasure. In this paper, we shall refer to a mobile node launching JellyFish attack as ‘JF-node’. Attack being a protocol compliant methodology is harder to detect. A JF-node targets a closed loop protocol such as TCP and exploits its working mechanism to degrade the communication performance. We have analyzed the effects of three JF (Jelly-Fish) attack variants e (1) JF-reorder, (2) JF-delay and (3) JF-drop over TCP-SACK , the most robust TCP as compared to others such as TCP-Reno, TCP-newReno, TCPahoe etc. in terms of handling packet losses and retransmission timeouts. The simulation results are collected for static as well as mobile ad hoc networks by varying number of attackers and their positions on active routes. From attack analysis, it has been observed how a JF-attacker behaves with the change in flow and congestion control mechanisms of TCP and vice-versa. To detect the JF-nodes the change in flow and congestion control mechanisms of TCP and vice-versa. To detect the JF-nodes and prevent them from participating in route discovery processes, a novel detection algorithm that works efficiently for all three variants of the JellyFish attack has proposed. To implement the attacks and performance analysis, we have simulated MANET scenarios using network simulator known as NS2. To the best of knowledge, this is the first work that includes the impact analysis of all the three variant of JellyFish attack and also proposed a common solution to identify and countermeasure the JF-attackers in small as well as large MANET scenarios.
II. RELATED WORK

In a secure wireless network, the main goal is to maintain secure and successful data transmission between two end points. For the network to perform efficiently, it is imperative to devise a security mechanism that can make the network resilient against various attacks [2]. Over the past few years, attacks that exploit MANETs' vulnerabilities have been proposed in conjunction with possible countermeasures.

Attacks such as black-hole, Sybil, worm-hole etc. disrupt the normal behavior of routing protocol by adding incorrect information, modifying information, dropping partial/complete information in control messages during the route discovery process [11]. Such forms of attacks are easy to detect as the attacker node is not following the protocol. Other attacks such as grey-hole, selfish behavior, jellyfish attack etc. comply with the protocol rules and, yet, disrupt the network communication. Detecting such attacks is a challenging task.

In Aad et al. [5], authors proposed three forms of protocol compliant attacks for closed loop protocols and named them as JellyFish attack. The authors show how these attacks identify the loopholes in TCP flow and congestion control algorithm and exploit them to degrade the network throughput. The performance, however, has been analyzed on a small static MANET scenario consisting of few nodes and only one data flow on a mobile MANETs has not been investigated. The detection mechanism proposed by authors [8] is not feasible to implement as per their own admission. Authors in Wazid et al. [7] proposed e-TCP, an improved version of TCP, to mitigate the effects of delay variance Jelly-Fish attack. However, performance for other attack variants has not been reported. A brief survey on malicious node behavior detection is given in Dasilva MVL et al [10]. A reorder density based detection mechanism for detecting jellyfish reorder attacks has been presented in [4]. Each node calculates the reorder density by recording the reordering frequency of its neighbor nodes. Authors, however, did not provide any countermeasure mechanism and no results, simulations or otherwise, have been presented to show the efficiency of their proposed detection method. An analytical model for detecting packet reordering attack by adding two new transition states in TCP-NewReno is proposed.

In Aad et al. [5], authors presented a denial of service model for blackhole and Jellyfish attacks. To analyze the effects of these attacks, various simulations along with the analytical modeling is performed over a large set of MANET scenarios with varying mobility, system size, node density, and counter-DoS strategies. Although no detection mechanism is provided for these attacks, the study provided useful insight.

III. THE PROPOSED SYSTEM

A. JellyFish attack

JellyFish attack maintains compliance with both the control and data protocols to make its detection and prevention difficult. Due to no functional distinction among mobile nodes in MANETs, an intermediate node can introduce a critical vulnerability for TCP congestion control mechanism. Such a compromised/malicious node alters its forwarding behavior as described in following variants of JF attack.

a. JellyFish reordering attack

As the name suggests, an attacker node reorders some of the packets before forwarding them. As ACKs of some of reordered packets are not received in time, the sender need to retransmit them again. From receiver's perspective, each time a packet is received, an ACK is generated. For out-of-order packets, sender shall receive duplicate ACK messages. TCP initiates its flow control mechanism if these duplicate ACK messages exceed a threshold (3 in this case) In this implementation of JF reordering attack, the JF node creates a reordering buffer of size k in its input queue. The data packets in this buffer are reordered before being forwarded.

b. JellyFish periodic dropping attack

In this attack, a JF node randomly discards some packets over a specified period during communication process. In this way, incorrect route congestion information is conveyed to TCP, which uses dropping of packets as an indication of congestion on the route. The JF-node may either choose to discard a fraction of packets (e.g., 10 packets from every 100 packets) or may discard all the packets received during a slice of time (e.g., discarding data packets for few milliseconds every second near the TCP sender timeout). This forces TCP to enter the retransmission timeout (RTO) and to increase its RTO value. The throughput decreases as the frequency of packets dropped by the attacker node increases.

c. JellyFish delay variance attack

In delay variance attack, JF nodes are selfishly delaying packets. Resultant increase in RTT misleads the sender TCP, which increases its congestion window size and sends traffic in bursts. It will eventually result in more collisions.

B. JellyFish attack detection using DTD algorithm

In this section the proposed a light-weight direct trust-based detection (DTD) algorithm as a countermeasure for identifying and removing the JellyFish attackers in all three variants (delay-variance, packet reordering and periodic discarding).

a. Assumptions

i) Destination nodes are assumed trustworthy.
ii) Only protocol compliant attacks take place.
iii) Each node can overhear and process the transmissions not designated for it.
iv) JF-node performs only JelleyFish attack and cannot influence Trust computations of its neighbors.

Following notations will be using:

- **N**: Number of nodes
- **S, D, n**: Source, Destination, Intermediate node
- **$\eta_n^k$**: k-hop neighborhood of node n
- **X**: Benign node just before attacker node
- **A**: Attacker JF-node, $A \in \eta_n^k$
- **$R_{S \rightarrow D}^k$$\in \mathbb{N}$**: kth route from S to D
- **$|R_{S \rightarrow D}^k|$$\in \mathbb{N}$**: Length of route R
- **$\mu_{n,t}$**: Control message from node n at instance t
- **$\rho$:** Current packet under consideration
- **$|\rho|$:** Sequence number of packet $\rho$
- **$\tau_p^p, \tau_n^p$:** Primary and Secondary timers at n
- **$\delta_n$:** Transmission time of a packet from n to next_hop
- **$\tau$:** Trust value $0 \leq \tau \leq 1$
- **$\tau_{min}$**: Minimum threshold for $\tau$
- **$Y_n$:** Table for trust value
- **$\alpha, \beta, \gamma$:** Weighting factors
- **T**: Time Interval
- **t, t-1**: Current and previous time instances

### c. MrDR Method

Trust, in its literal meaning, is the strong belief that someone is safe, reliable, honest, and well behaved. Additionally, this person will not harm you in any situation. The proposed method, MrDR, stems from (Monitoring, Detection and Rehabilitation) which are the three stages of the process. This method is based on applying the trust concept between nodes, each of which acts as an agent that can monitor all of its immediate nodes.

**a. Assumptions**

1. There are many ways to measure trust within MANET, such as benevolence, ability, integrity and cooperation with others.
2. Trust is not permanent as the nodes in MANET are dynamic. Thus, trust is a short-lived temporal action that needs to be recalculated at specific time thresholds.
3. Due to dynamic topology and non-fixed infrastructure that are the nature of MANET, trust is intransitive. For instance, if node A trusts node B and node B trusts node C, that does not mean that node A trusts node C. Node A needs to do different checks and make a comparison with its immediate nodes to check the trust value of node C.
4. Trust is a binary variable and has only two values: 1=trusted and 0=untrusted.
5. The default value of any node when it is added to the network is officially ‘untrusted’ until assessment of it is complete.
6. This method uses promiscuous nodes that acknowledge their neighbour’s activities within the network. For example, node A can hear all the communications to and from node B, even if node A does not subscribe to that communication.

### b. Monitoring

The aim in this stage is to monitor the network all the time and calculate Accomplishment Trust Value (ATV) and Reputation Trust Value (RTV).

1. ATV consists of two values: ATV1 and ATV2.
2. When the node transmits the required packets to another node or intended destination ATV1=0.5, otherwise ATV1=0.
3. A node should send back a confirmation message saying that it has already received the packet, so then ATV2=0.5, otherwise ATV2=0. Therefore to calculate ATV:

\[
\text{ATV} = \text{ATV1} + \text{ATV2}
\]  

(1)

4. If ATV overall=1 then the node is normal. However, if the ATV=0.5, then the node is selfish, whereas when the ATV=0, the node is malicious.
v) If the node drops the packet for the first time, the RTV=0.5. When the dropping is repeated the RTV=0.25. However, when it repeats for a third time the RTV=0. The reason that the punishment for dropping packets is not directly=0 is because of the nature of MANET. For example, the punishment for modification is always due to malicious nodes, but dropping is not the same, as it could occur due to a power failure, considering MANET has limited energy.

c. Detection
The purpose of this stage is to detect misbehaving nodes, whether selfish or malicious, and calculate Honesty Trust Value (HTV). This value indicates the quality value of trusted nodes within the network. If node A sends information about the total trust value to node B and this information does not match the information that node B got from its immediate nodes, then the HTV for node A= 0, otherwise HTV=1.

i) The Total Trust Value (TTV) is calculated from stage 1 and 2 using the following equation:

\[
TTSV = \begin{cases} 
0, & ATV < 1, RTV < 1, HTV = 0 \\
1, & ATV = 1, RTV = 1, HTV = 1 
\end{cases}
\]  

ii) TTSV has only two values: trusted=1 and untrusted=0.

d. Rehabilitation
In this stage, misbehaving nodes are rehabilitated for reuse in future communication. MANET, with its dynamic topology nodes, cannot maintain one status all the time. Therefore, stage 1 and 2 would be repeated every (n) seconds, such as 360 seconds, depending on the misbehaving rate. For instance, if node X is malicious three times in succession the rehabilitation time will be longer, such as every 500 seconds. Thus, it will save the power stock as the node is definitely malicious.

i) CTV (Check Trust Value) specifies the amount of time taken between checks of the trust values. If the RTV= 0 three times in succession, the CTV process would be repeated every 500 seconds until the RTV value is changed.

Algorithm 1: Direct trust-based detection (DTD) algorithm
Require: N: Number of nodes
\[ Y_n \in \{1..N\}, 1 \leq n \leq N \]: Trust table of all nodes
S,X,D = Source, Intermediate and Destination nodes
\[ N_J^F \]: Set of JF-nodes identified by node X
\[ \tau_{min} \]: Minimum threshold for trust
Ensure: 
\[ N_J^F = U_{i=1}^{x} N_i^F \]: Set of JF-nodes.

1. \[ \tau_{max} = 1 \]
2. for i=1 to N do
3. \[ N_i^F = 0 \]
4. for j= 1 to N do
5. \[ Y_{ij} \] = \[ \tau_{max} \]
6. end for
7. end for
8. \[ N_J^F = 0 \]
9. for Each packet \( \rho \) forwarded by a X do
10. Set timers \( \xi_{x}^0 \) and \( \xi_{x}^1 \) for \( \rho \)
11. Forward the \( \rho \) to \( A \in \eta_{x}^1 \)
12. Sense the communication of neighbors in promiscuous Mode
13. Flag=FLASE
14. if \( \xi_{x}^0 \) expired then
15. if packet \( \rho \) not overheard by X then
16. \( \xi_{x}^0 = \frac{\tau_{x}^0}{x} \)
17. if packet \( \rho \) overheard by X then
18. if \( \xi_{x}^1 \) not expired then
19. A is suspected a JF-delay attacker
20. Flag = TRUE
21. else
22. A is suspected a JF-drop attacker
23. Flag = TRUE
24. end if
25. end if
26. end if
27. else
28. if Packet \( \rho \) overheard by X then
29. if \( \rho \) is more recent than \( \rho \) then
30. A is suspected a JF-reorder attacker
31. Decrement \( Y_x^{[A]} \)
32. Flag = TRUE
33. end if
34. end if
35. end if
36. if Flag \( \equiv \) True then
37. Decrement \( Y_x^{[A]} \)
38. if \( Y_x^{[A]} \) < \( \tau_{min} \) then
39. \( N_J^F = N_J^F U \{ A \} \)
40. \( N_x^F = N_x^F U \{ A \} \)
41. X initiates a RERR message
42. S initiates a re-routing process
43. end if
44. end if
45. end for
IV. SIMULATION AND RESULT ANALYSIS

A. PERFORMANCE METRICS

In this paper, we have used following performance metrics for evaluating effects of JellyFish attack and effectiveness of our detection algorithm:

TCP Throughput ($\Gamma$): It is the ratio of the total number of bits transmitted ($B_{tx}$) to the time required for this transmission, i.e. the difference of data transmission end time and start time ($t_{end} - t_{start}$). This metric depicts how the congestion control mechanism at the source node is affected by the packet losses caused by JF-nodes. A decrease in throughput is an outcome of any JF attack.

$$\Gamma = \frac{B_{tx}}{t_{end} - t_{start}}$$ (1)

End-to-end delay ($\Delta$): It is average transmission delay of packets transmitted from source to destination. D is computed as the ratio of the sum of individual delay of each received data packet to the total number of data packets received. This metric is used to evaluate impact of a JF-attack on delay-sensitive applications of TCP-based MANETs. By intentionally discarding, delaying or reordering packets, a JF-node can increase the value of this metric; increase being caused by re-transmissions of such packets due to timeout at TCP source.

$$\Delta = \sum_{i=1}^{N_{rxd}} \frac{\Delta}{N_{rxd}} \cdot N_{rxd} = \text{no.of received packets}$$ (2)

TCP overhead (L): It is the ratio of the sum of overhead packets transmitted by TCP to the total number of packets (including data and control) transmitted. In our case, overhead consists of the number of retransmitted (including fast retransmits) data packets and duplicate ACKs.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>NS2 Simulator</td>
</tr>
<tr>
<td>Simulation time</td>
<td>1000s</td>
</tr>
<tr>
<td>Scenario dimension</td>
<td>600*600 sq.meter</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>60</td>
</tr>
<tr>
<td>Transport protocol</td>
<td>TCP</td>
</tr>
<tr>
<td>Routing protocols</td>
<td>AODV</td>
</tr>
<tr>
<td>Mobility model</td>
<td>Random way point</td>
</tr>
<tr>
<td>Node speed</td>
<td>1 to 5 m/s</td>
</tr>
<tr>
<td>MAC specification</td>
<td>IEEE 802.11</td>
</tr>
<tr>
<td>Radio type</td>
<td>802.11a/g</td>
</tr>
<tr>
<td>Data rate</td>
<td>24Mbps</td>
</tr>
<tr>
<td>Application layer protocol</td>
<td>FTP Generic</td>
</tr>
<tr>
<td>Frame size</td>
<td>512 bytes</td>
</tr>
<tr>
<td>Number of frames per second</td>
<td>38 to 100</td>
</tr>
</tbody>
</table>

B. Simulation results

In this section, the effectiveness of proposed DTD (direct trust-based detection) algorithm for identifying JF-nodes has evaluated through simulation process. Improvement in TCP throughput once a JF-node is detected and blacklisted by the proposed countermeasure method is a measure of effectiveness of the proposal.

The Figure 2 shows the network includes 60 nodes, including one source node and one destination. The proposed method will check the network every two minutes, with the total period of each experiment equaling approximately six minutes. By using the DTD method the network will be updated as anomalies such as DoS attacks occur.

Fig. 2. Network Environment

The Figure 3 shows when found the malicious nexthop select the alternate path and switch over that path for data transmission using MrDR Method. The packet delivery ratio and network throughput increase. In addition, the packet delay is decreased, compared with the packet delivery ratio and the network throughput.

Fig 3. Alternative path selection
The Figure 4 shows the comparative graph of End-to-end Delay between DTD algorithm and MrDR method. In this X axis - Number of intervals, Y axis - Delay(s). This shows MRDR achieves reduced delay when compared to DTD.

The Figure 5 shows the graphical representation of Throughput. Green colour represents Throughput increased with MrDR method and Red colour represents Throughput with DTD algorithm. Which shows MRDR achieves increased Throughput when compared to DTD.

The Figure 6 shows the graphical representation of packet delivery ratio. This graph shows MRDR achieves increased packet delivery ratio when compared to DTD.

V. CONCLUSION AND FUTURE WORK

In this proposed scheme, a detailed performance evaluation of JellyFish attack over TCP based MANETs is presented. The protocol compliant nature of JF-attack makes its detection process a difficult task. Based on the insight obtained through JF-attack results, a novel trust-based JF-attack countermeasure mechanism is proposed and evaluated which identifies JF-nodes dynamically during the communication process. Proposed detection method smoothly extends for detection of the JellyFish attack. In enhancement Monitor, Detect, Rehabilitate (MrDR) method, which is applied to detect jellyfish attacks, is an effective mechanism for dealing with them. In this method using an in-direct and direct monitoring method in which all the neighbours of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility. After Monitoring and detection the rehabilitation is applied to discard the attacker and find the alternate path for future data transmission.

In future, will extend the detection mechanism accuracy by using an in-direct monitoring method in which all the neighbors of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility.

ACKNOWLEDGMENT

The authors are grateful to the Faculty of B.S Abdur Rahman University for their valuable support in deploying the system and in providing experimental data and technical documentation concerning the “Direct Trust-based Detection and recovery process of JellyFish attack in MANET”.

V. CONCLUSION AND FUTURE WORK

In this proposed scheme, a detailed performance evaluation of JellyFish attack over TCP based MANETs is presented. The protocol compliant nature of JF-attack makes its detection process a difficult task. Based on the insight obtained through JF-attack results, a novel trust-based JF-attack countermeasure mechanism is proposed and evaluated which identifies JF-nodes dynamically during the communication process. Proposed detection method smoothly extends for detection of the JellyFish attack. In enhancement Monitor, Detect, Rehabilitate (MrDR) method, which is applied to detect jellyfish attacks, is an effective mechanism for dealing with them. In this method using an in-direct and direct monitoring method in which all the neighbours of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility. After Monitoring and detection the rehabilitation is applied to discard the attacker and find the alternate path for future data transmission.

In future, will extend the detection mechanism accuracy by using an in-direct monitoring method in which all the neighbors of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility.

ACKNOWLEDGMENT

The authors are grateful to the Faculty of B.S Abdur Rahman University for their valuable support in deploying the system and in providing experimental data and technical documentation concerning the “Direct Trust-based Detection and recovery process of JellyFish attack in MANET”.

V. CONCLUSION AND FUTURE WORK

In this proposed scheme, a detailed performance evaluation of JellyFish attack over TCP based MANETs is presented. The protocol compliant nature of JF-attack makes its detection process a difficult task. Based on the insight obtained through JF-attack results, a novel trust-based JF-attack countermeasure mechanism is proposed and evaluated which identifies JF-nodes dynamically during the communication process. Proposed detection method smoothly extends for detection of the JellyFish attack. In enhancement Monitor, Detect, Rehabilitate (MrDR) method, which is applied to detect jellyfish attacks, is an effective mechanism for dealing with them. In this method using an in-direct and direct monitoring method in which all the neighbours of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility. After Monitoring and detection the rehabilitation is applied to discard the attacker and find the alternate path for future data transmission.

In future, will extend the detection mechanism accuracy by using an in-direct monitoring method in which all the neighbors of a node will monitor and share their observations with each other. This approach will decrease the false JF-attacker detections caused by improper overhearing of data packets during promiscuous mode due to interference and node mobility.
REFERENCES


ANJUGAM S received B.Tech degree in Information Technology from P.B College of Engineering in 2014. Pursuing M.Tech degree in Computer Science & Engineering from B.S Abdur Rahman University.

MUTHUPRIYA V is holder of B.E and M.E in Computer Science Engineering, Working Assistant professor in Computer Science Engineering.