INTERNATIONAL JOURNAL OF ADVANCE RESEARCH AND DEVELOPMENT

IJARnD

(Volume2, Issue5)
Available online at www.ijarnd.com

Prevention of MANETs from Malicious Nodes Performing FNA Using OLSR

Jayasudha .K[1], Mr. Scaria Alex[2]

[1] *Final Year Student of M.Tech CSE, Jawaharlal College of Engineering and Technology, Lakkidi, Kerala.*
[2] *Asst.Professor, Department of CSE, Jawaharlal College of Engineering and technology, Lakkidi, Kerala*

ABSTRACT

In MANETs, it is effective for mobile nodes to retrieve data items using top-k queries, in which data items are ordered according to a particular attribute score, and the query-issuing node acquires the data items with the k highest scores. However, accurate results may not be acquired in environments where malicious nodes are present. In top-k queries, it is important to neutralize the Data Replacement Attack (DRA). The DRA is defined as the malicious nodes attempt to replace necessary data items with unnecessary ones. The proposed method addresses top-k query processing and malicious node identification against data replacement attack in MANETs. In the top-k query processing method, in order to maintain the accuracy of the query result, nodes reply with data items with the k highest scores, along with multiple routes. Moreover, to enable detection of data replacement attacks, reply messages include information on the route along which reply messages are forwarded, and thus the query-issuing node can know the data items that properly belong to the message. In the malicious node identification method, the query-issuing node first narrows down the malicious node candidates, using the received message information, and then requests information on the data items sent by these candidates. In this way, the query-issuing node can identify the malicious node. When multiple malicious nodes are present, the query-issuing node may not be able to identify all malicious nodes at a single query. It is effective for a node to share information about the identified malicious nodes with other nodes. In the proposed method, each node divides all nodes into groups by using the similarity of the information about the identified malicious nodes. Then, it identifies malicious nodes based on the information on the groups. In this case, however, a malicious node may declare fake information that claims normal nodes as the malicious nodes. This type of attack is called as false notification attack (FNA) and the node who is performing FNA only called as Liar Nodes (LN). Identification of the LN is also based on the information on the groups itself. By using Optimized Link State Routing Protocol (OLSR) designing a message authentication method to prevent malicious nodes from performing FNA.

Keywords: MANETs, top-k Query Processing, DRA, FNA, Grouping, and OLSR.

1. INTRODUCTION

A Mobile Ad hoc Network (MANET) is a continuously self-configuring, infrastructure-less network of mobile devices connected wirelessly. Each mobile device can be called as a node which has communication capabilities. Nodes can communicate directly or indirectly with each other. The mobile devices can communicate each other directly if they fall in the radio coverage range of each other otherwise the node can use the concept of multihop. Each node behaves like Station and Router. All the nodes participate in the routing and the performance may depend upon the cooperation between nodes. MANETs are useful where there is no infrastructure is or installation is not possible.

Due to the poor resources (i.e., the communication bandwidth and the battery life of mobile nodes are limited) for each node in the MANET, it is effective to retrieve only the necessary data items using top-k query. On the other hand, in MANETs, if a normal node becomes malicious owing to an attack from outside the network, the malicious node tries to disrupt the operations of the system. Figure 1.2 shows an application example of a top-

k query assuming a rescue effort at a disaster site where the communication infrastructure, e.g., the Internet, has been broken down. Assume that each rescuer (P1, P2,P3,P4,P5) manages the information on his/her responsible sites(A, B,C,D,E,F,G,H……..O), and a rescuer searches for sites with high damage to dispatch rescuers with a limited number of apparatuses. When the rescuer P1 wants to find the two highest damaged sites in the entire area, he issues a top-k query where k is set to 2. If each rescuer transmits the information on the two highest damaged sites in his/her responsible sites, the query issuing rescuer acquires the information on more sites than necessary.

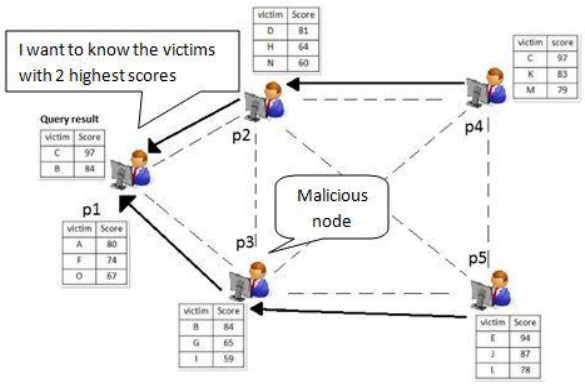

Fig.1.2 Example of Top-k query processing in a MANET

The attacks in the MANETs due to the malicious nodes are basically classified into two types. They are: (1). Data Replacement Attack (DRA) and (2). False Notification Attack (FNA). DRA is a new type of attack in which a malicious node replaces the received data items known as the local top-k result with unnecessary data items, for example, its own low-score data items. Since DRAs are a strong attack, and more difficult to detect than other traditional types of attacks, some specific mechanism for defending against DRAs are required. A malicious node may declare fake information that claims normal nodes as the malicious nodes are called as the FNA. The one who is doing only the attack called FNA is known as the liar nodes where it is represented as LN.

MANETs are currently using in many real-time applications like disaster management, military battlefield, personal area network and Bluetooth, commercial sector, and whether reporting etc. Top k query processing is the popular way to get data from MANET. The main objective is that it very important to keep the top k query results error free when it is utilized by the MANETs. The scope of the proposed system is that it is addressing the issue of the malicious nodes and preventing malicious nodes from performing FNA in MANETs by designing a message authentication method using the OLSR.

2. LITERATURE REVIEW

Exact top-k query processing has gained more and more attention recently because of its wide use in many fields, such as information retrieval, multimedia databases, P2P[2] and sensor networks [3][4],[5],[6] etc. The main reason for such attention is that top-k queries avoid overwhelming the user with a large number of uninteresting answers that are resource-consuming. However, it helps to reduce energy consumption and traffic by enabling nodes to filter unnecessary data items but also these methods are unsuitable for use in MANETs and do not protect against DRA because they are not adapted to node mobility.

In [7], [8], and [9] the top-k query processing methods for MANETs, adapted to the node mobility. Here, the query issuing node floods the query message attached with some score information. Each node then estimates the kth score of the data item and sends back data items whose scores are equal to or greater than the estimated one. This method can, therefore, decrease the number of transmitted data items. However, there is still a drawback that queries are forwarded to all nodes in the network by flooding. Thus, many nodes that do not contribute to the data items with k- highest scores have to send redundant top-k queries and replies. As a result, there is a high level of traffic.

To solve high level of traffic, proposed a routing method for top-k query processing in MANETs [10]. When each mobile node issues a query, it first refers to its own routing table, and then sends the query message that specifies the requiring ranks of data items to the query address for this query by unicast. However, these methods are not designed for environments in which malicious nodes exist, for example, the data items in the top-k result are sent back along a single route, and thus are vulnerable to malicious nodes.

In most current reputation systems [12]–[15], a node collects locally-generated node feedbacks and aggregates them to yield the global reputation values for others based on periodical information exchanges

between neighbours. The node whose reputation is below a predefined threshold is considered as selfish and put into a blacklist, otherwise as trustworthy or malicious. However, the systems suffer from a number of problems. First, they lack efficient mechanisms to collect and propagate reputation information. Second, reputation calculation based on partial local information, which may include false information, may result in insufficiently accurate reputation evaluation to truly reflect node behaviours. Third, solely relying on reputation system is not effective enough to thwart uncooperative behaviours. However, this method does not assume that the malicious nodes send false reputation scores.

The design and performance evaluation of a new on-demand secure ad hoc network routing protocol called Ariadne [17] that withstands node compromise and relies only on highly efficient symmetric cryptography. Ariadne prevents attackers or compromised nodes from tampering with uncompromised routes consisting of uncompromised nodes and also prevents a large number of types of Denial-of-Service attacks. Ariadne is more secure, more efficient, or more general. Ariadne can authenticate routing messages using one of three schemes: shared secrets between each pair of nodes shared secrets between communicating nodes combined with broadcast authentication, or digital signatures.

A novel scheme, called Secure Protocol for Reliable dAta Delivery (SPREAD) [18], to statistically enhance data confidentiality in a MANET. The fundamental idea of SPREAD is shown in Figure 2.3. Assume that the secret message is sending through a single path; the enemy can compromise it by compromising any one of the nodes along the path. However, if the secret message is dividing into multiple pieces, and sending these multiple pieces via multiple independent paths, then the enemy has to compromise all the pieces from all the paths to compromise the message. Improved security can be expected by this means. However, these methods neither assume top-k queries nor protect against DRA and thus cannot be directly applied to the problem.

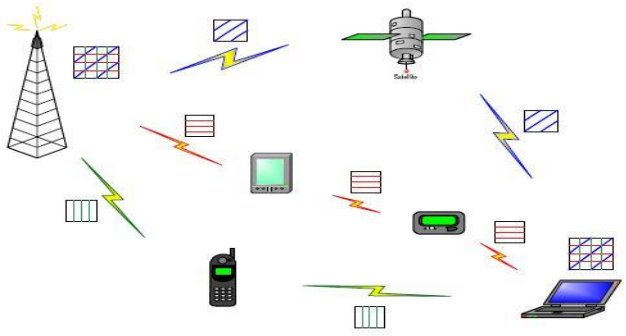

Figure 2.3: Fundamental idea of SPREAD

In the top-k query processing method [19], the query issuing node first floods a query over the entire network, and each node receiving the query stores information on all possible routes to the query-issuing node. Then, each receiving node replies with data items with the k highest scores to two neighbour nodes. In addition, each node includes, in its reply message, information on the reply message forwarding routes which consist of pairs of the sender node and next node IDs. Based on this attached information, the query issuing node can detect an attack occurring along a reply message route. In malicious node identification method, a query issuing node that detects an attack narrows down the malicious node candidates based on information on the reply message routes included in its received reply messages. Then, the query issuing node determines whether a given reply message sent back by a malicious node candidate includes replaced data items or not, by sending inquiries to nodes receiving reply messages from this candidate. In this way, the query issuing node can identify the malicious node. In this method, the existence of only one malicious node in the entire network can only be identified. However, in real environments, there may be multiple numbers of malicious nodes present in the entire network. In this case, since nodes must increase the number of routes along which reply messages are sent, high traffic volume is required to remain high accuracy of the query result.

However, when there are multiple malicious nodes in the network, it is difficult to identify all the malicious nodes in a single query, partly because of nodes, in this method, are more likely to only identify malicious nodes near their own location than those farther away. Therefore, proposing a signature-based top-k query processing method [20] against DRAs in MANETs. The query-issuing node first floods a query over the entire network and then receiving nodes send back reply messages, which include data items with the k highest scores, a list of the reply forwarding routes, and a signature list. After the query issuing node has received all the reply messages, it confirms the received signature lists, and thereby detects any DRAs and identifiers the respective malicious nodes. If the query-issuing node identifies malicious nodes, it shares this information by flooding a notification message over the entire network, including the identifiers of the malicious nodes, and the

signature lists in which these nodes have replaced data items with their own lower-score items. Nodes receiving this notification message for the first time decrypt the signature lists attached to the received message and confirm that the nodes identified as malicious have actually performed DRAs. The traffic required for top-k query processing is large, resulting in low accuracy of the query result especially when *k* is large.

3. MODULE DESCRIPTION

In the proposed system there are mainly two stages:

1) Top-k Query Processing
2) Malicious Node Identification Method

3.1 Top-k Query Processing

The top-k query processing basically consists of Query forwarding, Reply forwarding and Detecting attacks.

Query Forwarding

First, the query issuing node generates an authentication message and floods it to all the nodes the network. Then the query issuing node floods a query over the entire network. The query consists of the node identifier of the query issuing node (Query-issuing nodeID), the Query identifier of the query (Query ID), the number of requested data items (*k*), the query condition, a list of the node identifiers of nodes on the path to which the query message is to be transmitted (Query path) and an authentication message. The following algorithm is for query forwarding.

```
1:  /* Receive a query message */
2:   if Mq matches authentication message then
3:      if Mq receives a query for the first time then
4:         Store Query path and hop counts as its Parent Query path
5:         Store the node ID at the end of Query path as its parent
6:         Set RD for replying data items
7:         /* Send the query message to neighbour nodes */
8:         Add M'qs node ID to the end of Query path
9:         Send the query to neighbour nodes
10:     else
11:        Store Query path and hop count as its neighbour Query path
12:        Store the node ID at the end of Query path as its neighbour
13:     end if
14:  end if
```

Algorithm 1: Forwarding a Query

Reply Forwarding

When RD has passed, each node sends back a reply message, which includes its own node identifier (Sender node ID), the identifier of the next node along the reply route (Dest node ID), a list of the data items (including their scores) and the node identifiers of the nodes possessing them (Data list), and a list summarizing the reply message routes, i.e., a list of the pairs of sender and next node identifiers (Forwarding Route). When a radio link disconnection to the parent node or next node occurs, the node sends the reply message to another neighbour node among those whose routes to the query-issuing node include the least overlap between Query path in the replay message and their own Query path. The following algorithm is for reply message and this one includes the sufficient steps to overcome when a link disconnection occurred.

```
1:  /* Sends a reply message after RD has elapsed */
2:  /*Select a node to send reply message*/
3:  for each Neighbor do
4:     if Neighbor's hopCount is the minimum then
5:        Insert Neighbor into DestNode
6:     end if
7:  end for
8:  if |DestNode| > 1 then
9:     Select a Neighbor whose Neighbor Query path least overlaps with the parent Query
       path as a DestNode
10: end if
11: Add (Mr, Parent node) to received REP.FR and send REP to parent node
12: for i=0 to 1 do
13:    if i=0 then
```



```
14:        Add(M_r, parent node) to received REP.FR and send REP to parent node
15:    else if i=1 then
16:        Add(M_r, DestNode) to received REP.FR and REP to DestNode
17:    end if
18: end for
19: /*Receive a reply message*/
20: Send ACK to the sender node of REP
21: if before RD then
22:    Store REP
23: else if after RD and M_r receives a data item with a higher score than with the kth highest
    score than with the kth highest score among data items already sent then
24:    Send REP including new local top-k result to parent node and DestNode
25: end if
26: /*Resend the reply message*/
27: if M_r does not receive ACK from DestNode by waiting time for retransmission and the number of
    retransmission<R then
28:    Resend REP to DestNode
29: else if Mr does not receive ACK from DestNode by waiting time for retransmission and the number of
    retransmissions < R then
30:    Resend REP to DestNode
31: else if the number of retransmissions > R then
32:    /* Mr detects the disconnection of radio link */
33:    if M_r has sent REP to all Neighbor then
34:        Discard REP
35:    else if M_r knows a Neighbor whose Neighbor Query path includes DestNode then
36:        Send REP to the Neighbor
37: else
38:        Select randomly a Neighbor among Neighbors which have not been selected yet
39:        Send REP to the Neighbor
40:    end if
41: end if
```

Algorithm 2: Sending a Reply Message

Detecting Attacks

After the query-issuing node, *Mp*, receives all the reply messages, it detects a DRA according to Algorithm 3. The SendRoute is the output of the algorithm. SendRoute denotes the set of node identifiers along the route from the node possessing a given data item to the query- issuing node.

```
1:  /* After the query-issuing node receives all reply messages */
2:  INPUT: Top-k Result, REPs
3:  OUTPUT: SendRoute
4:  SendRoute ←Ø;
5:  for each REP do
6:     for each Top-k Result do
7:        if REP.FR includes the node ID of a node processing a data item in Top-k Result
          and REP.Data does not include the data item then
8:           Insert a route from the node with the missing data item to the query issuing
             node into SendRoute
9:        end if
10:    end for
11: end for
12: if SendRoute ≠ Ø then
13:    Detect Attack
14: end if
```

Algorithm 2: Detecting Attack

3.2 Malicious Node Identification Method

The malicious node identification method includes 2 sub modules such as local identification and global identification.

Local Identification

After detecting a DRA, the query-issuing node tries to identify the malicious nodes. In Algorithms 4 and 5, the query-issuing node narrows down the candidates for malicious nodes, and identify the malicious nodes by making respective inquiries.

```
INPUT: SendRoute
OUTPUT: Candidate
/* The query-issuing node confirms whether the reply message included replaced data items */
for each node ID in SendRoutes do
    if node ID is included in SendRoute then
        Insert node ID into Candidate
    end if
end for
if |Candidate| = 1 then
    return Candidate as a Malicious Node
else if |Candidate| > 1 then
    /* Inquire to the candidates of a malicious node */
    Perform the procedure of Algorithm 5
end if
```

Algorithm 4: Narrowing Down the Malicious Node Candidates

In the proposed method, according to Algorithm 4, the query-issuing node narrows down the malicious node candidates by using SendRoute, obtained in Algorithm 3. In Algorithm 4, Candidate denotes the set of node identifiers of malicious node candidates, ordered by ascending hop count from the query-issuing node, and missing Top-k result denotes the replaced data items. The nodes included in SendRoute, whose data items are corrupted, are all possible attackers. Therefore, the query-issuing node recognizes these nodes as malicious node candidates.

```
INPUT: Candidate
OUTPUT: MaliciousNode
/* Mp starts to inquire */
for each i in Candidate.size do
    if InqRoute include other candidates in Candidate then
        /* End procedure without inquiring */
        break
    else if hop count to Candidate > 1 then
        /* Send an inquire message */
        Send MNI-INQ to Mdesti to ask data items that Candidate[i] sent
    end if
    /* Mv receives an inquire message */
    if Mv receives MNI-INQ then
        Send MNI-INQ to the next node of Mv in InqRoute
    end if
    /* A malicious node candidate receives an inquire message */
    if Mdesti receives MNI-INQ then
        Send MNI-IREP including scores of data items sent by Candidate[i] to Mp
    end if
    /* Mu receives a reply message for the inquiry */
    if Mu receives MNI-IREP then
        Send MNI-IREP to sender MNI-INQ
    end if
    /* Mp receives a reply message for the inquiry */
    if Mp receives MNI-IREP then
        /* the query issuing node identifies the malicious node */
        if scores includes the score of the missing data items in global Top-k result then
            return Candidate[i -1]
        end if
```



```
30:       end if
31:   end for
```

Algorithm 5: Identification of a Malicious Node

Algorithm 5 shows the procedures for inquiring about information on data items sent from malicious node candidates.

Global Identification

Each node individually identifies malicious nodes using the shared information by the two steps: node grouping and malicious node identification.

Node Grouping: Each node divides nodes in the network into some groups based on the information in the notification messages received by the nodes, according to Algorithm 6. The following algorithm follows the node grouping.

```
1:    /* After receiving queries NumQuery times */
2:    INPUT: Ri (i =1,2,……, n)
3:    OUTPUT: Group
4:    /* Calculate the similarity between nodes */
5:    for each a € n do
6:        for each € 2 n do
7:            if detected node in a and b are equal then
8:                Sim(a, b)=1
9:            else
10:               Sim(a, b)=0
11:           end if
12:       end for
13:   end for
14:   /* Node grouping*/
15:   for each a €  n do
16:       for each b € n do
17:           if Mal_can ≠ Ø; and sim(a,b) θ then
18:               Insert M_a,M_b into G_can
19:           else if Mal_can ≠Ø ; and { ∀ x € Mal_can,sim(x,b)≥θ} then
20:               Insert M_b into G_can
21:           end if
22:       end for
23:       if G_can ∉ Group then
24:           Group ←G_can
25:       end if
26:       clear G_can
27:   end for
28:   /*Cleaning in each group*/
29:   for each group do
30:       for each M_g,e in Group_g do
31:           if M_g,e identified in a node include in Groupg then
32:               eliminate M_g,e in from Groupg
33:           end if
34:           for each BL_g,f in BLg do
35:               if Count_B,f ≤ ρ and Mg,e identifies BL_g,f then
36:                   eliminate Mg,e from Groupg
37:               end if
38:           end for
39:       end for
40:   end for
```

Malicious Node Identification: After the node grouping, each node conclusively determines malicious nodes based on the information about malicious nodes identified by nodes in each group. The following algorithm shows the identification of malicious nodes from the group.

```
1:    After grouping nodes in the network */
```



```
2:    INPUT: Group
3:    OUTPUT: Malicious
4:    /* Calculate the malicious nodes identified by all nodes in each group */
5:    While All CountMx<Ø do
6:        All CountMx and Mal clear
7:        for each Group do
8:            if Mx is identified by all nodes in Group then
9:                Mal ←Mx
10:               CountMx ++
11:           end if
12:       end for
13:       /* Identified the malicious nodes */
14:       for each My in Mal do
15:          if CountMy ≥Ø then
16:              Malicious ←My
17:          end if
18:       end for
19:       for each Malicious do
20:           for each Group do
21:               if Maliciousm€Groupg then
22:                   Delete Malicious from Group
23:               end if
24:           end for
25:       end for
26:   end while
```

Algorithm 7: Global Identification

Identification of FNA:

```
1:    For each group do
2:       Count=0
3:       For each malware in group do
4:          If group.malware not equal to detected malware then
5:             Count++
6:          End if
7:    end for
8:    If Count==number of detected malware in the group then
9:       FNA<= group.members
10:   end if
```

Algorithm 8: Identification of FNA

The algorithm 8 is for the identification of FNA. To the little possibility of a liar, nodes try to identify them in the section called identification of FNA. For that checking that which nodes are not at all identified any one of the malicious nodes from the group which is made in the section node grouping. If there exists any node that does not identify as malicious one from that groups then the owner node is collecting as liar nodes in the variable FNA.

4. CONCLUSION

The query issuing node generates an authentication message and floods it over the entire network before issuing a query. While issuing the query it may also consist of the authentication message for verification. Authentication of nodes is a security function of OLSR for addressing the vulnerabilities in the MANET. So this mechanism is adopted for addressing the malicious nodes from performing FNAs. If the verification is succeeded nodes reply with k data items with the highest score along multiple routes. After detecting attacks, the query issuing node narrows down the malicious node candidates and then try to identify the malicious nodes through message exchanges with other nodes. When multiple malicious nodes are present, the query issuing node may not be able to identify all malicious nodes at a single query. It is effective for a node to share the information about the identified malicious nodes with other nodes. In proposed method, each node divides all nodes into some groups by using the similarity of the information about the identified malicious nodes. Then, it identifies malicious nodes

based on the information on the groups. Some nodes may not share the information on the identified the malicious nodes in the group. So that node is considered as the liar nodes.

The path of the reply message may not consist the malicious node. Since in such cases, the proposed system detects no malicious nodes. As a part of future work, plan to address this issue.

5. REFERENCES

[1] **Takuji Tsuda, Yuka Komai, Takahiro Hara, Shojiro Nishio** (2016), "Top-k Query Processing and Malicious Node Identification Based on Node Grouping in MANETs" *IEEE Trans. Mobile Computing*, no. 7, pp. 2541864.

[2] **W.T. Balke, W. Nejdl, W. Siberski, and U. Thaden** (2005), "Progressive distributed top-*k* retrieval in peer-to-peer networks," in *Proc. ICDE*, pp. 174_185.

[3] **B. Chen, W. Liang, R. Zhou, and J. X. Yu** (2010), "Energy-efficient top-*k* query processing in wireless sensor networks," in *Proc. CIKM*, pp. 329_338.

[4] **X. Liu, J. Xu, and W. C. Lee** (2010), "Across pruning framework for top-*k* data collection in wireless sensor networks," in *Proc. MDM*, pp. 157_166.

[5] **B. Malhotra, M. A. Nascimento, and I. Nikolaidis**(2011), "Exact top-*k* queries in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 10, pp. 1513_1525.

[6] **M. Wu, J. Xu, X. Tang, and W. C. Lee** (2007), "Top-k monitoring in wireless sensor networks," IEEE Trans. Knowl. Data Eng., vol. 19, no. 7, pp. 962_976.

[7] **Hagihara, M. Shinohara, T. Hara, and S. Nishio** (2009), "A message processing method for a top-k query for traffic reduction in ad hoc networks," *Proc. Int. Conf. on Mobile Data Management*, pp. 11–20.

[8] **Y. Sasaki, R. Hagihara, T. Hara, M. Shinohara, and S. Nishio**(2010), "A top-k query method by estimating score distribution in mobile ad hoc networks," *Int. Workshop on Data Management for Wireless and Pervasive Communications (DMWPC 2010)*, pp. 944–949.

[9] **Y. Sasaki, T. Hara, and S. Nishio** (2013), "Two-phase top-k query processing in mobile ad hoc networks,"' in *Proc. NBiS*, pp.42_49.

[10] **D. Amagata, Y. Sasaki, T. Hara, and S. Nishio** (2013), "A routing method for top-k query processing in mobile ad hoc networks," in *Proc. Int. Conf. on Advanced Information Networking and Applications*.

[11] **D. Amagata, Y. Sasaki, T. Hara, and S. Nishio** (2013), "A robust routing method for top- k queries in mobile ad hoc networks," in *Proc. MDM*, pp. 251_256.

[12] **Ashish Kumar, Vidya Kadam, Subodh Kumar and Shital Pawar**(2011), "An Acknowledgement- Based Approach for the Detection of Routing Misbehavior in MANETS" in *Proc IJAES* Volume 1, Issue 1, pp-04-06.

[13] **Z. Li and H. Shen** (2011), "A hierarchical account-aided reputation management system for large-scale MANETs," in *Proc. INFOCOM*, pp. 909_917.

[14] **S. Marti, T. J. Giuli, K. Lai, and M. Baker** (2000), ``Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. MobiCom*, pp 255_265.

[15] **P. Michiardi and R. Molva** (2002), "Core: A collaborative reputation mechanism to enforce node cooperation in manets". In *Proc. of CMS*.

[16] **T. Anantvalee and J. Wu** (2007), **"**Reputation-based system for encouraging the co-operation of nodes in mobile ad hoc networks". In *Proc. of ICC*.

[17] **Y.C. Hu, A. Perrig, and D. B. Johnson** (2002), "Ariadne: A Secure on-demand routing protocol for ad hoc networks," in *Proc. MobiCom*, pp. 12_23.

[18] **W. Lou, W. Liu, and Y. Fang** (2002), "SPREAD: Enhancing data confidentiality in mobile ad hoc networks,"' in *Proc. INFOCOM*, vol. 4., pp. 2404_2413.

[19] **T. Tsuda, Y. Komai, Y. Sasaki, T. Hara, and S. Nishio** (2014), ``Top-*k* query processing and malicious node identification against data replacement attack in MANETs," in *Proc.MDM*, pp. 279_288.

[20] **Takuji Tsuda, Yuka Komai, Takahiro Hara and Shojiro Nishio (2015)**, "Signature- Based Top-k Query Processing against Data Replacement Attacks in MANETs", *IEEE Tras. Reliable Distributed Systems.*

