International Journal of Information & Network Security (IJINS)
Vol.2, No.3, June 2013, pp. 239~244
ISSN: 2089-3299

Algorithm to Detect Spurious Communications in Vehicular Ad hoc Networks

Muhammad Aamir*, Shabbir Mukhi**
* MS-Computing, SZABIST
** Faculty of Computer Science, SZABIST

Article Info

Article history:

Received Jan 10th, 2013
Accepted Feb 15th, 2013

Keyword:

Algorithm
VANET
Costing
Function
Analysis

ABSTRACT

Vehicular ad hoc network is a form of mobile ad hoc networks in which end nodes are vehicles communicating to each other without a central control or pre-defined infrastructure. An important security consideration is event based alerts received by VANET nodes. In the presence of attackers, a chance to receive malicious alerts is increased and a response to such messages by the receiver can be dangerous. Therefore, an algorithm is designed to detect such communications in order to protect vehicles in VANET environments. The focus in this paper remains on the evaluation of proposed algorithm in terms of computational requirements. The computational complexity of suggested algorithm is determined to be in $O(n^2)$ after computing the processing costs at individual levels of execution. It is also found that the algorithm is an efficient scheme of detecting spurious alert messages in VANETs as compared to some other suggested methods with equivalent or higher computation cost.

Corresponding Author:

Muhammad Aamir,
MS-Computing,
Shaheed Zulfikar Ali Bhutto Institute of Science & Technology (SZABIST),
90 Clifton, Karachi, Pakistan.
Email: aamir.nbpit@yahoo.com

1. INTRODUCTION

A vehicular ad hoc network (VANET) is a mobile ad hoc network [1] in which nodes (vehicles) communicate to each other with high mobility. As a conventional Manet, they do not have a pre-defined topology and routing path from source to destination. Not only the vehicles are end nodes of the network, but Road Side Units (RSUs) are also present in the overall infrastructure of a VANET. They are fixed entities alongside roads installed by trusted authorities (Certificate Authroity - denoted by *CA*) used to broadcast control and alert messages.

Due to the particular nature of environment in VANETs with high mobility, they require specific treatments such as special routing protocols and security arrangements [2]. An important security consideration is event based alerts received by VANET nodes. In the presence of attackers, a chance to receive malicious alerts is increased and a response to such messages by the receiver can be dangerous. Therefore, an algorithm is proposed in this paper to detect such communications in order to protect vehicles in VANET environments.

Detecting spurious communications of alerts from malicious senders is useful in reducing many adverse effects on the roads in VANETs including severe accidents [3]. Hence, implementation of an algorithm that can effectively detect spurious communications of alerts from malicious senders can be a key to ensure road safety of vehicles in VANETs.

2. RELATED EFFORTS

In past, serious efforts have been made to detect node misbehavior. Some efforts suggest solutions depending upon the reputations of vehicles [4]-[5]. However, if the reputation of malicious nodes is more than the average, a receiver can be tricked by some spurious alert presented as a valid one. Some other schemes are also proposed that emphasize on ceritifcate revocations, digital signatures and neighbor support to detect spurious communications of alerts from malicious senders in VANETs [6]-[8].

In this paper, we present an algorithm to detect spurious messages generated in Vehicular Ad hoc Networks by attackers and misbehaving nodes. It does not depend on some individual parameter as others suggest, but investigates a series of different parameters to reach a decision regarding the validity of a received message. It is therefore an effective tool to differentiate spurious messages from alerts that are based on occurrence of actual events.

3. PROPOSED ALGORITHM

3.1. Assumptions

The following assumptions are made in designing the proposed algorithm:

(1) Vehicles are equipped with GPS device and they follow the IEEE 1609.2 security standard [9]. It enables a receiver to check digital signature of sender through Public Key Infrastructure (PKI).
(2) Vehicles consider alerts generated by RSUs as 100% valid as they are sent by trusted entities.
(3) A reference database of digital signatures of vehicles in VANET is available to the receiver through Certificate Authority. It is assumed that for n^{th} sender, its digital signature can be one of the **1** to **n** entries in **Sign** table of reference database **DB**. Moreover, digital signatures are randomized in Sign table among 1 to n entries by Certificate Authority and hence a signature is to be checked in all 1 to n entries for each search.

3.2. Working of Algorithm

In our scheme, the On Board Unit (OBU) in the vehicle is configured to process the information pertaining to a received alert message in order to identify if it is spurious, hence responding only to the valid alerts. If a message is found spurious, it may be filtered out.

(1) **n:** It represents no. of alert messages on an event to be processed by the function at a time. Accordingly, it also represents no. of vehicles involved in sending alert messages on an event to a receiver. During a single run of execution, the algorithm works on n^{th} alert message that corresponds to n^{th} vehicle.
(2) **DS[ ]:** It is an array containing digital signatures of 'n' vehicles sending alert messages to a receiver. It is configured that a digital signature at n^{th} index of DS[ ] is that of n^{th} vehicle.
(3) **RSU[ ]:** It represents an array of binary values (true or false) describing whether the same alert on the same event is being generated by Road Side Unit (RSU). It is configured that a value at n^{th} index of RSU[ ] corresponds to n^{th} vehicle.
(4) **TimeVal[ ]:** It represents an array of binary values (true or false) indicating if the time validity of an alert message holds. It is configured that a value at n^{th} index of TimeVal[ ] corresponds to n^{th} vehicle.
(5) **EventLoc[ ]:** It represents an array of binary values (true or false) indicating if the event location is relevant to the receiver. It is configured that a value at n^{th} index of EventLoc[ ] is communicated by the n^{th} sender.
(6) **Rep[ ]:** It is an array containing reputations of 'n' vehicles sending alert messages to a receiver. A reputation value can be between 0 and 1. The highest reputation corresponds to 1. It is configured that a reputation at n^{th} index of Rep[ ] is that of n^{th} sender.
(7) **v:** It represents no. of total vehicles present in a VANET. It is configured to check that the number of vehicles involved in sending alert messages on an event to a receiver can not be greater than the total number of vehicles in a given VANET.

Proposed Algorithm to detect spurious communication in VANETs

```
1.  Initialize j , count, sign_flag, Trustworthiness_total, Trustworthiness_mean
2.  Do if v >= n
3.    For a ← 1 to n
4.      j=a, sign_flag=false
5.      Do while j>0 AND sign_flag=false
6.        Do if DS[a] ≠ DB.Sign[j]
7.          j=j-1
8.        Else
9.          sign_flag=true
10.     Do if j=0
11.       Return Spurious
12.     Do if sign_flag=true AND RSU[a]=true
13.       Return Valid
14.     Do if RSU[a]=false AND TimeVal[a]=false
15.       Return Spurious
16.     Do if TimeVal[a]=true AND EventLoc[a]=false
17.       Return Spurious
18.     Else
19.       Trustworthiness_total=Trustworthiness_total + Rep[a]
20.       count=count+1
21.   Trustworthiness_mean=Trustworthiness_total / count
22.   Do if Trustworthiness_mean>0.5
23.     Return Valid
24.   Else
25.     Return Spurious
26. Else
27.   Return Invalid Parameters
```

The working of algorithm is described above through a pseudo code implementation. The algorithm first determines whether digital signature of a vehicle in the corresponding message is valid. If no, the message is considered as spurious. If yes, the algorithm determines whether RSUs also broadcast the same alert. If yes, the message is valid. If no, the algorithm further determines whether time validity of the message is passed. If yes, the message is spurious. If no, the algorithm determines whether the event location is relevant to the receiver. If no, the message is spurious. If yes, the algorithm further records the reputation of sender to take a decision on reputation basis. It is obvious that before going to perform the reputation based analysis, the algorithm can rapidly identify a message if it is valid or spurious based on some key parameters of received information. If this part of algorithm is not able to identify received messages as valid or spurious, the "Reputation based" decision logic is applied.

When decision is to be taken on the basis of reputation, the algorithm analyzes trustworthiness factor by calculating the mean value of reputations of all message sending nodes. If the mean value is equal to or greater than 0.5, the corresponding alerts are treated as valid. Otherwise, they are treated as false or spurious.

4. COMPUTATIONAL COST ANALYSIS

For an analysis of the processing costs at individual levels of execution, we let:

C_1 = Initialization; Initialization + Assignment
C_2 = Comparison (Greater than or equal to; Less than or equal to; Equal to; Not equal to)
C_3 = Increment; Decrement
C_4 = Assignment
C_5 = Addition; Addition + Assignment
C_6 = Division; Division + Assignment
C_7 = Return

We let $\mathbf{t_a}$ denote the number of times **while** loop is executed for an instantaneous value of **a** within **for** loop. Since two statements are compared, we take twice of the comparison cost. In a **while** loop, test is executed one time more than the loop body. So we let $\mathbf{t_a-1}$ denote number of times **If** statement is checked within **while** loop. If **j** becomes **zero** after complete execution of **while** loop cycles, it indicates that **digital signature** of sender is not found in reference database (DB) and hence it will return "spurious" after executing full **while** loop cycles. Therefore in order to compute maximum possible running cost of algorithm, we assume that the last entry to be compared in reference database is the digital signature of the sender under consideration so that the algorithm can execute further instructions. This search is made for each message from the beginning in order to make the algorithm capable of finding the digital signature of a node in refresh mode i.e. without remembering any past knowledge of its own. It makes the algorithm less reliable on computer's memory and therefore it can also be useful in storage constrained environments. Since the **If** statement within **while** loop executes $\mathbf{t_a-1}$ times but **j** is decremented one time less, we let $\mathbf{t_a-2}$ denote the number of times **j** is decremented within **while** loop. Therefore, ***sign_flag*** becomes **true** one time within **while** loop and it is totally done **n** times with reference to **for** loop.

The running time of an algorithm is the sum of running times for each statement executed. A statement that takes '**x**' steps to execute and its execution is '**y**' times, it contributes '**xy**' to the total running time. So we have the total running time T(n) of our designed algorithm as:

$$\mathbf{T(n)} = 5C_1 + C_2 + C_1 + (n+1)\,C_2 + n\,C_3 + n\,2C_4 + 2C_2\sum_{a=1}^{n} t_a + C_2\sum_{a=1}^{n}(t_a-1) + C_3\sum_{a=1}^{n}(t_a-2) + n\,C_4 + n\,C_2 + n\,2C_2 + n\,2C_2 + n\,2C_2 + n\,C_5 + n\,C_3 + C_6 + C_2 + C_7$$

We must compare each digital signature DS[a] with each signature in the entire sub-section of reference database DB.Sign [a, a-1,…,1], so $\mathbf{t_a=a}$ for a=1,2, …,n.

$$\mathbf{T(n)} = 5C_1 + C_2 + C_1 + (n+1)\,C_2 + n\,C_3 + n\,2C_4 + 2C_2\sum_{a=1}^{n} a + C_2\sum_{a=1}^{n}(a-1) + C_3\sum_{a=1}^{n}(a-2) + n\,C_4 + n\,C_2 + n\,2C_2 + n\,2C_2 + n\,2C_2 + n\,C_5 + n\,C_3 + C_6 + C_2 + C_7$$

We know that:

$$\sum_{a=1}^{n} a = [\frac{n(n+1)}{2}]$$

$$\sum_{a=1}^{n}(a-1) = [\frac{n(n-1)}{2}]$$

$$\sum_{a=1}^{n}(a-2) = [\frac{n(n-3)}{2}]$$

Therefore we get:

$$\mathbf{T(n)} = 5C_1 + C_2 + C_1 + (n+1)\,C_2 + n\,C_3 + n\,2C_4 + 2C_2[\frac{n(n+1)}{2}] + C_2[\frac{n(n-1)}{2}] + C_3[\frac{n(n-3)}{2}] + n\,C_4 + n\,C_2 + n\,2C_2 + n\,2C_2 + n\,2C_2 + n\,C_5 + n\,C_3 + C_6 + C_2 + C_7$$

$$\mathbf{T(n)} = 5C_1 + C_2 + C_1 + n\,C_2 + C_2 + n\,C_3 + n\,2C_4 + n^2\,C_2 + n\,C_2 + n^2\,(C_2/2) - n\,(C_2/2) + n^2\,(C_3/2) - n\,(3C_3/2) + n\,C_4 + n\,C_2 + n\,2C_2 + n\,2C_2 + n\,2C_2 + n\,C_5 + n\,C_3 + C_6 + C_2 + C_7$$

$$\mathbf{T(n)} = (\frac{3}{2}C_2 + \frac{1}{2}C_3)n^2 + (\frac{17}{2}C_2 + \frac{1}{2}C_3 + 3C_4 + C_5)n + (6C_1 + 3C_2 + C_6 + C_7)$$

The above running time can be expressed as:

$$T(n) = an^2 + bn + c$$

The constants *a*, *b* and *c* depend on the machine costs, it is thus a *quadratic function* of *n*. Hence it can be expressed in *Big-O* notation as:

$T(n) = O(n^2)$

5. ANALYSIS FOR DIFFERENT CASES

5.1. Best Case Scenario

The best case scenario can be mentioned as an incident when n is greater than v. In this situation, the algorithm will not check the validity of received alerts and it will directly return the message "Invalid parameters". Only initialization of five variables, one comparison and a return will contribute towards total cost which is **$T(n) = 5C_1 + C_2 + C_7$**.

However, for the sake of an effective algorithm analysis, we consider the best case scenario when n is less than or equal to v. In this situation, the best case is observed when the first received alert has no match of its digital signature with reference database entry. Hence the algorithm will treat it as a spurious message. Initialization of five variables, one comparison, execution of **for** and **while** loops, two assignments before **while** loop, comparison within **while** loop, comparison after **while** loop and a return will contribute towards total cost. Therefore, we have:

$$T(n) = 6C_1 + 7C_2 + C_3 + 2C_4 + C_7 \quad (1)$$

5.2. Average Case Scenario

The average case scenario appears when half of the total received alerts reach the reputation based calculation in algorithm. Moreover, the next message will be decided spurious at the stage before reaching reputation based calculation. Therefore, in this case we have:

$$T(n) = (\frac{3}{8}C_2 + \frac{1}{8}C_3)n^2 + (\frac{17}{4}C_2 + \frac{1}{4}C_3 + \frac{3}{2}C_4 + \frac{1}{2}C_5)n + (6C_1 + 2C_2 + C_7) \quad (2)$$

5.3. Worst Case Scenario

The worst case scenario exists when all 'n' alert reports reach the reputation based calculation in algorithm. Hence, the mean value is finally calculated and compared with threshold value which is 0.5. Therefore, we have:

$$T(n) = (\frac{3}{2}C_2 + \frac{1}{2}C_3)n^2 + (\frac{17}{2}C_2 + \frac{1}{2}C_3 + 3C_4 + C_5)n + (6C_1 + 3C_2 + C_6 + C_7) \quad (3)$$

From equations (1), (2), and (3), we analyze that the best case scenario running time of our designed algorithm is a machine-dependent constant value whereas average and worst case scenario running times are found in **$O(n^2)$**.

6. COMPARISON WITH OTHER METHODS

It is a known fact that identifying spurious communications in VANETs in accurate manner is a difficult task. Therefore, many efforts have been made to suggest methods of this problem. Some of them lack in overall security, whereas some are inefficient due to high computational complexity. We compare the computational complexity of our designed algorithm with a couple of other methods discussed in year 2011.

In [10], different methods of RSU allocation algorithms are discussed for certificate update in VANET environments. Thus it deals with the certificate revocation schemes that are effectively handled by Certificate Authority and may be employed further in false alert identifications. It is mentioned that the most driving routes first method has computational complexity of $O(n^2)$. Here, 'n' represents the set of intersections in a given graph that represents a city map. Moreover, another method called the most satisfied intersection pairs first method has overall computational complexity of $O(n^4)$. Again, 'n' represents the set of intersections in a given graph that represents a city map. Hence, the algorithm proposed in this paper is an efficient scheme of detecting spurious alert messages in VANETs and comparable to some other suggested methods of equivalent or higher computation costs.

Table 1. Running Time Costs of different Security Algorithms in VANETs

Security Algorithm	Running Time Cost
The most driving routes first method	$O(n^2)$
The most satisfied intersection pairs first method	$O(n^4)$
Detection of spurious alerts (proposed algorithm)	$O(n^2)$

7. CONCLUSION

In this paper, we designed a new algorithm with improved security to detect spurious communications in VANETs. The algorithm takes all necessary elements of information that relate to different alert reports on an event as input parameters and produces a decision on the validity of messages (valid or spurious). The computational complexity of suggested algorithm is determined to be in $O(n^2)$ after computing the processing costs at individual levels of execution. However, the best case scenario running time of designed algorithm is found to be a machine-dependent constant value. It is also found that the proposed algorithm is an efficient scheme of detecting messages from malicious nodes in VANETs as compared to some other suggested methods that have equivalent or higher computation costs.

REFERENCES

[1] H. Hartenstein and K.P. Laberteaux, "A tutorial survey on Vehicular Ad hoc Networks," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 164-171, June 2008.
[2] X. Lin, *et al.*, "Security in Vehicular Ad hoc Networks," *IEEE Communications Magazine*, vol. 46, no. 4, pp. 88-95, Apr 2008.
[3] F. Sabahi, "The Security of Vehicular Adhoc Networks," *2011 3rd IEEE International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN)*, pp. 338-342, 2011.
[4] M. Raya, P. Papadimitratos and J.P. Hubaux "Securing Vehicular Communications," *IEEE Wireless Communications Magazine*, vol. 13, no. 5, pp. 8-15, Oct 2006.
[5] P. Papadimitratos, *et al.*, "Architecture for Secure and Private Vehicular Communications," *2007 7th IEEE International Conference on ITS Telecommunications (ITST '07)*, pp. 1-6, 2007.
[6] M. Raya, *et al.*, "Eviction of Misbehaving and Faulty Nodes in Vehicular Networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, pp. 1557-1568, Oct 2007.
[7] G. Calandriello, *et al.*, "Efficient and Robust Pseudonymous Authentication in Vanets," *2007 4th ACM International Workshop on Vehicular Ad hoc Networks (VANET '07)*, pp. 19-28, 2007.
[8] M. Ghosh, *et al.*, "Distributed Misbehavior Detection in VANETs," *2009 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1-6, 2009.
[9] 1609.2-2006 – "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments" – Security Services for Applications and Management Messages. *IEEE Standards*, 2006.
[10] S. Singh and R. Vijayan, "Enhanced Security for Information Flow in VANET using Signcryption and Trust level", *International Journal of Computer Applications*, vol. 16, no. 5, pp. 13–18, Feb 2011.

BIOGRAPHY OF AUTHORS

Muhammad Aamir is MS-Computing from SZABIST, Karachi, Pakistan. He is working in NBP as Technology Support Officer in various domains of Applications & Networking. His research interests include Computer Networks & Security, Communication Systems and I.T. based Industrial Automation. He is student member of IEEE

Shabbir Mukhi is attached with Faculty of Computer Science SZABIST, Karachi, Pakistan. He has a vast teaching & research experience in multiple computer science domains such as Computer Networks, Network Security, Algorithms and Cryptography.