THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

A Novel Method for Secure Routing in Vanets

T.Valliammai
Assistant Professor, Department of Information Technology
Prathyusha Institute of Technology And Management, Thiruvallur

Abstract: *We address the problem of effective vehicular routing in hostile scenarios where malicious nodes intend to jeopardize the delivery of messages. Compromised vehicles can severely affect the performance of the network by a number of attacks, such as selectively dropping messages, manipulating them on the fly, and the likes. One of the best performing solutions that has been used in static wireless sensor networks to deal with these attacks is based on the concept of watchdog nodes (also known as guard nodes) that collaborate to continue the forwarding of data packets in case a malicious behavior in a neighbor node is detected. In this work we consider the BRAVE routing protocol, which has been previously shown to perform very well in vehicular networks, and analyze whether a similar solution would be feasible for vehicular environments.*
All nodes are secure from attackers by using
S-BRAVE protocol. The overall performance is getting increased compared to watchdog nodes, which is used in BRAVE Protocol.

Keywords: *Security, routing, vehicular adhoc networks.*

1. Introduction

Vehicular Ad Hoc Networks (VANET) have emerged with a great strength gaining a lot of interest by government, traffic authorities, car manufacturers. They open a new market that allows for different services like enhanced safety, traffic congestion detection and avoidance, and Internet connectivity, among others.
VANET require an effective solution to send data messages to vehicles located farther than their radio range. Most VANET routing protocols in the literature [1], [2] like GSR [3], SAR [4], A-STAR [5], GeOpps [6] or BRAVE [7] are based on geographical routing to reach the destination.
BRAVE, is a beaconless routing protocol that uses an opportunistic forwarding mechanism. It ensures that the next selected forwarder is able to receive the data packet successfully. It has shown a great performance in terms of delivery ratio with respect to the other approaches. After receiving a packet from a sender, neighboring nodes propose themselves as candidate forwarders depending on their position. They are not able to deal with certain situations such as sybil attacks, selective forwarding or sinkhole attacks, where malicious nodes try to impair the routing protocol by not forwarding the information to other nodes.
In this paper, we propose S-BRAVE protocol , where messages are signed by taking advantage of the PKI. Nevertheless, we have developed an efficient certificate exchange mechanism where the certificate will be inserted in V2V messages only if the other vehicle has not received it yet. Thus, authenticity and integrity is guaranteed for every message transmitted along the VANET.
Using the aforementioned technique, neighbors watch other selected nodes to be sure that they forward packets to the next hop. If a node is selected to forward the packet and it does not transmit it, then neighboring nodes will select themselves as a forwarder, taking the responsibility of sending the packet to the next hop. The whole process is detailed in later sections.
The remainder of this paper is organized as follows. In Section II describes the BRAVE routing protocol as well as the main threats that it must deal with. In Section III we detail S-BRAVE, our proposal. The evaluation of the performance of S-BRAVE is shown in Section IV. Finally, we summarize the main outcomes of this protocol and conclude the paper in Section V.

2. Background

2.1. Brave

BRAVE is a beaconless routing algorithm which does not consider information gathered from periodic beacons in making routing decisions.
In BRAVE there are four message types: DATA, RESPONSE, SELECT and ACK. Due to the existence of many data sources in the network, every message includes a unique key which is the result of concatenating the identifier of the source node and a sequence number. BRAVE uses an opportunistic scheme when making forwarding decisions. Thus, when a node intends to send data to a destination, it broadcasts the DATA packet (scheduling also a timer in case no neighbors answer the message). After receiving this DATA packet, every neighboring node schedules a response timer before answering. The more progress provided by a neighbor towards the destination, the less such neighbor has to wait to answer with a RESPONSE message.
The sender selects the neighbor whose RESPONSE message arrives first. For that, a SELECT message aimed at the chosen vehicle is broadcasted. Therefore, all neighbors get aware of the vehicle that has been selected to be the next forwarder. Those

which are neither the final destination nor the next forwarder delete the DATA message from their buffer and go back to the initial state. The selected vehicle, on its hand, after receiving the SELECT message becomes the current forwarder and starts over the forwarding process by broadcasting the DATA message. Such message is expected to reach the previous hop, acting as an implicit acknowledgment of reception. Otherwise, the previous hop rebroadcasts the DATA message and the already selected forwarder answers with an explicit ACK message. On the other hand, if the forwarding node observes that it has no neighbors to which forward the message, then it stores the message in its buffer. An explicit ACK message is sent in this case too. Thus,
the vehicle carries the message until it receives a new beacon indicating that there is a neighbor. The reception of this beacon triggers a new event that makes the node check whether there are messages to be delivered. In such case, the whole process starts over.

2.2. Routing Threats

There are different security threats depending on the layer that they are aimed at. Focusing on the network layer, attacks like black hole, selective forwarding, wormhole and the likes are described in the literature. Depending on the messages exchanged in routing protocols, some of them are more vulnerable to these attacks than others. Thus, it is important to analyze the routing protocol to find the threats that affect it the most.
In BRAVE, the first packet to be transmitted contains the data

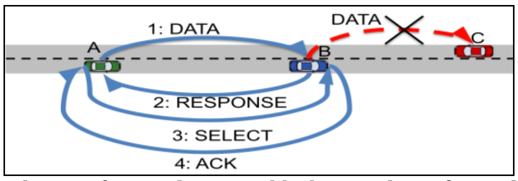


Figure 1: Selective forwarding / sinkhole attack performed by vehicle B.

information, and only nodes that receive this message participate in the next hop selection mechanism. Hence, a black hole attack consisting of a malicious node that silently discards or drops messages without informing the source that the data did not reach its intended recipient will not affect BRAVE at the time of selecting a new neighbor. However, an attacker might participate on the exchange of BRAVE messages and, once it holds the DATA packet and sends back an ACK, it could stop forwarding prematurely. Figure 1 illustrates this case.
BRAVE messages are not authenticated nor integrity protected, enabling other kinds of attacks by a malicious node.
Thus, it can manipulate the information stored in the message,
for instance changing the destination of the packet or altering its content. This issue can be alleviated by employing a PKI, so that vehicles will be able to sign data packets with their private keys. Hence, receivers can validate packets by using the public key contained within the digital certificate of the sending vehicle. In the following section, we describe the mechanism employed to exchange these certificates among nodes.
In a sybil attack, a malicious node presents multiple identities with different locations to other vehicles in the network. This attack is more sophisticated than the previous ones because, in this case, the malicious node announces itself also in other locations, taking advantage of these positions to be selected as the best neighbor to forward a packet. For instance, in Figure 2 vehicle B creates a new identity B! in a more advantageous location. Hence, it is selected as the best forwarder to the destination.
The only way for a malicious node to create more than one entity is to have more than one pair of public/private keys. There are different alternatives for it, like the use of pseudonyms or installing several certificates within the vehicle. We simplify the problem by forcing a single certificate per vehicle, which is generated by a trusted CA. In such case, the sybil attack gets reduced to its minimum exponent. That is, a vehicle could forge its position, but could not create multiple identities.
Finally, a wormhole attack requires the cooperation of at least two malicious nodes. It consists of two vehicles that create a

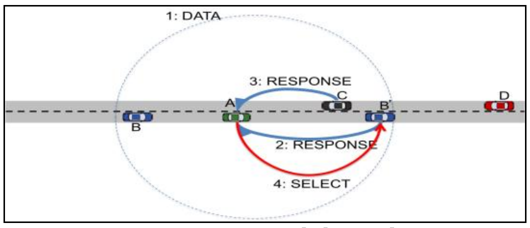


Figure 2: Sybil attack

tunnel between them, so that they can forge their distance to the destination. For instance, if the malicious nodes are far from each other more than one hop, by using the tunnel, for the rest of the neighbors it would be as if there were no distance between them. This attack is harder to perform because of the high variability of links among neighboring nodes due to the high speed of the vehicles.

3. Securing The Brave Protocol

In this section we develop S-BRAVE, an extension of the BRAVE routing protocol targeted at addressing the security threats. We will provide authentication and integrity by exploiting a PKI. Thus, the source vehicle signs data packets with its private key and the receiver uses the public key of the sender to check the validity of the packet. Since the receiver node requires the sender certificate, it is necessary a previous exchange (introducing extra overhead).

3.1. Certificate Exchange

Since a VANET is a distributed environment, vehicles must trust each other somehow. In S-BRAVE we assume a unique CA which is the same for all the vehicles in the VANET.

Each vehicle owns a unique identifier and a pair of keys (public and private) as well as a certificate issued by the CA. The first problem to deal with is how to exchange certificates among vehicles. Every time a node receives a message it must have the certificate of the sender node in order to authenticate it and to check the integrity of the message .

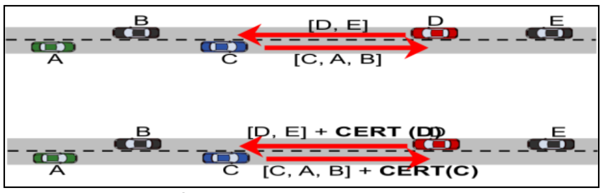


Figure 3: Certificate Exchange Via Periodic Beacons.

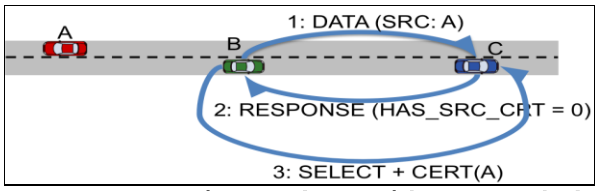


Figure. 4: Certificate exchange of the source vehicle.

We propose a reactive certificate exchange method which minimizes the number of certificate exchanges. Every beacon sent will include a cache of known neighbors, being a known neighbor one whose certificate is stored within the vehicle. When a vehicle receives this beacon, just by looking for its own identifier in the neighbor list, it will be able to determine if its certificate is present in the cache of the neighbor. Such cache is updated with a less recently used (LRU) scheme. If the certificate identifier is not present, then the vehicle will include its own certificate in the next beacon round. Using this strategy only the first beacon will include the certificate, the following messages between those vehicles will not need to include certificates for validation. Besides, other nodes that receives a beacon with the certificate can take advantage of this exchange method to store the certificate for possible use in the future. Figure 3 shows this exchange of messages.

Given that certificates are exchanged in advance, it is possible to authenticate routing messages (RESPONSE, SELECT and ACK) by just using digital signatures. However, in order to check the validity of a DATA message, a vehicle located farther than one hop of the sender needs a mechanism to get the certificate of the source. The reason is that DATA messages are signed by the source, but not by intermediate relays.

Our proposal to solve this problem is based on modifying RESPONSE and SELECT messages. A bit included in the RESPONSE message will indicate if the responding vehicle needs the certificate of the source. After receiving this RESPONSE, the SELECT message will be extended with the certificate of the source node depending on this bit (see Figure 4). This protocol modification entails an overhead decrease mainly when the path between source and destination is stable.

3.2. S-Brave Operation

The certificate exchange scheme described before is a basic building block of our solution. However, BRAVE is still weak against a selective forwarding attack that can be accomplished in two ways. In the first one, a malicious node does not continue the forwarding of the DATA message, nevertheless it answer its previous hop with an ACK message making it believe that it have forwarded it. In the second one, the malicious node does not send the SELECT message. For instance, if a node starts the exchange of messages but it does not send the SELECT message, there will not be a forwarding node and therefore the message will not be forwarded. In both cases, the previous hop may think that the forwarding was completed.

In order to try to avoid this type of situation, S-BRAVE employs the concept of watchdog nodes or guard nodes in the following way. Every neighboring vehicle that provides advance to the destination will act as a guard node. Those vehicles not selected as the next forwarder will try to ensure that the whole DATA forwarding process is completed. They keep on listening to the next forwarder, checking whether it retransmit the DATA message. If a guard node does not receive this message, it will take the role of the next forwarder by taking the responsibility of sending the DATA message to the next hop. They also include the detected malicious vehicle in a black list, to avoid that it gets selected as the next forwarder in the future.

First of all, we have modified the ACK message. A new bit has been added, which indicates the reason why this ACK has been sent. Thus, a vehicle can send this message by two reasons: DATA message has already been forwarded previously, or it has been buffered by the vehicle because it did not have any neighbors which provided advance towards the destination.

In addition, we have also defined a black list where neighbors which do not forward messages are registered into. This mechanism is used to avoid a malicious node to continuously impair the protocol performance by being selected by the same node one time after another. Thus, guard nodes will ignore the messages coming from a node of the black list. For instance, after a node sends an answer with a RESPONSE message, the neighbors that have this node into their black list, will also send their RESPONSE message instead of canceling their timers. Besides, the sender of the DATA message will also ignore the RESPONSE of a node if its identifier is stored in the black list.

Finally, neighboring vehicles that receive a RESPONSE or SELECT message do not go back to the initial state. Instead, they will keep the DATA message just received, watching for the right exchange of messages and the subsequent DATA message

forwarding by the selected node. They also schedule a timer that waits for this exchange to be succeeded within a period of time, otherwise the guard nodes will come to the conclusion that a malicious node is attacking by preventing the packet from being delivered. In such case, they collaborate to forward the DATA packet.
In the following, we detail S-BRAVE and provide some pieces of pseudo code of the most relevant operations that must be performed.
The sender vehicle, after issuing a DATA packet, schedules a timer waiting for responses from neighboring nodes (*awaiting RESPONSE*).This packet is the one that triggers the next hop selection. Procedures 1, 2, 3 and 4 deal with main message exchanges of S-BRAVE. In addition, Procedure 5 defines what vehicles do after their timers expire.

Procedure 1 process DATA (m:message, src:address, dst:address)

```
1: if (noActiveTimers) then {Node receives DATA in initial state}
2: if (dst == ownAddress) then {Node is the destination of DATA}
3: send(RESPONSE);
4: scheduleTimer(awaitingToSELECT);
5: else if (nodeProvidesAdvanceToDest(dst)) then
6: schedule Timer(awaitingToAnswer);
7: end if
8: else if ((src == selectedNode) && awaitingACK) then
9: exit; {Next hop, i.e. selectedNode, retransmit the packet}
10: else if (awaitingForwardedMsg) then {guard nodes}
11: cancelTimer(awaitingForwardedMsg);
12: if (nodeProvidesAdvanceToDest(dst)) then
13: scheduleTimer(awaitingToAnswer);
14: end if
15: end if
```

In Procedure 1, a vehicle that has received a DATA message can be in two states. The first one is the *idle* state, where the vehicle is at the beginning of processing the DATA message. If the node is the final destination of the packet it will immediately answer with a RESPONSE message, also scheduling a timer to receive the SELECT message. The receiver can also being the *awaiting ACK* state, meaning that it has nearly finished the exchange of messages but it is expecting the ACK message. After receiving the ACK, the node would go back to the *idle* state. Finally, if the node is a guard node and receives this DATA message it will cancel its timer of watching the packet, scheduling anew timer that depends on the progress provided with respect to the destination.
When a vehicle receives a RESPONSE message (Procedure2), it will send back to the most promising forwarder a SELECT message, also scheduling a new timer. On the other hand, if the vehicle is not the best forwarder, it will schedule a new timer to watch the messages exchange to act as a guard node.
Procedure 3 describes what happens when a vehicle receives a SELECT message. If it has already sent a RESPONSE message, it will be selected as the next forwarder. Thus, it will cancel its waiting timer (awaiting Select). In case the vehicle is the final destination, it will send an ACK message back to the previous hop. Otherwise, it will broadcast the DATA message unless it will not have any neighbors around it. In this latter case, it will store the message in a buffer, answering with an ACK which specifies this. Guard nodes will cancel their timers and will schedule new ones because the messages exchange is being performed correctly.

Procedure 2 processRESPONSE (m:message, src:address, dst:address)

```
1: if (awaitingRESPONSE && (dst == ownAddress)) then
2: cancelTimer(awaitingRESPONSE);
3: send(SELECT, src);
4: selectedNode! src;
5: scheduleTimer(awaitingACK);
6: else if (awaitingToAnswer) then
7: cancelTimer(awaitingToAnswer);
8: scheduleTimer(awaitingNextForwarderSelected);
9: end if
```

Procedure 3 processSELECT (m:message, src:address,dst:address)

```
1: if (awaitingSELECT) then
2: cancelTimer(awaitingSELECT);
3: if (finalDest == ownAddress) then
4: send(ACK,src);
5: scheduleTimer(awaitingPostProc);
6: else if (dst == ownAddress) then
7: if (noNeighbors) then
8: send(ACK); {It buffers the DATA}
9: else
10: send(DATA);
11: scheduleTimer(awaitingRESPONSE);
12: end if
13: else
14: scheduleTimer(awaitingForwardedMsg);
15: end if
16: else if (awaitingToAnswer) then
17: cancelTimer(awaitingToAnswer);
18: scheduleTimer(awaitingForwardedMsg);
19: else if (awaitingNextForwarderSelected) then
20: cancelTimer(awaitingNextForwarderSelected);
21: scheduleTimer(awaitingForwardedMsg);
22: end if
```

Procedure 4 describes the ACK reception process. If the vehicle that receives the ACK is the sender, it will cancel its timer assuming the whole messages exchange is completed. On the other hand, guard nodes will analyze the reason of sending this ACK. In case the message indicates a forwarding not heard by them, they will take the role of forwarders by broadcasting the DATA packet.

Procedure 4 processACK (m:message, src:address, dst:address)

```
1: if (awaitingACK) then
2: cancelTimer(awaitingACK);
3: exit;{Node goes back to initial state}
4: else if awaitingForwardedMsg then
5: if (m.reason == Forwarded) then {reason is an attribute
   of the message m}
6: cancelTimer(awaitingForwardedMsg);
7: send(DATA);
8: scheduleTimer(awaitingRESPONSE);
9: else {m.reason == Buffered}
10: if (noPromisingNeighbors) then
11: buffer(DATA);
12: else
13: cancelTimer(awaitingForwardedMsg);
14: send(DATA);
15: scheduleTimer(awaitingRESPONSE);
16: end if
17: end if
18: end if
```

In Procedure 5, if the vehicle state is *awaitingToAnswer*, it will send a RESPONSE message. This is the case where the vehicle has received the DATA packet and has scheduled a timer to answer to it. On the other hand, guard nodes (the last two cases) will take the role of new forwarders by broadcasting the DATA message. In the remainder of this section, we analyze possible attacks and how S-BRAVE behaves against them.

Procedure 5 timer Expires(timer)

```
1: if (timer == awaitingToAnswer) then
2: send(RESPONSE);
3: scheduleTime r (awaitingSELECT);
4: else if (timer == awaitingPostProc) then
5: exit;{Node goes back to initial state}
6: else if (timer == awaitingNextForwarderSelected) then
7: send(DATA);
8: scheduleTimer(awaitingRESPONSE);
9: else if (timer == awaitingForwardedMsg) then
10: send(DATA);
11: scheduleTimer(awaitingRESPONSE);
12: end if
```

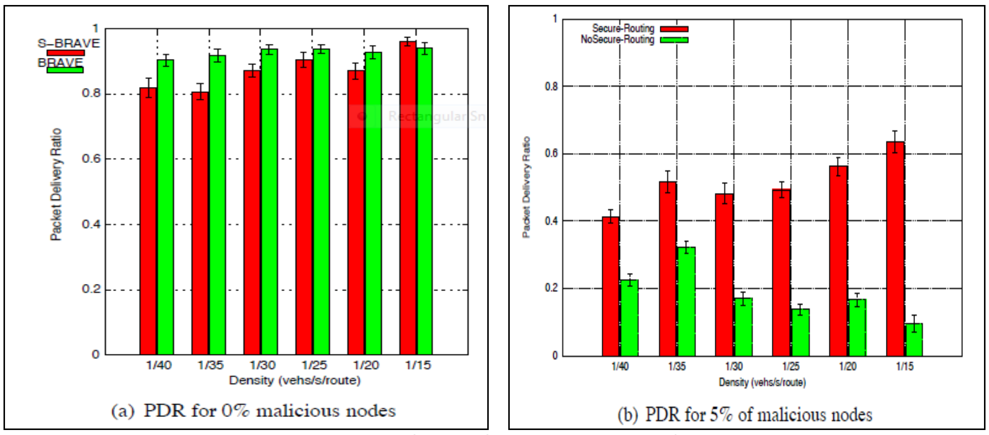
4. Performance Evaluation

4.1. Brave Vs S-Brave

We have compared both protocols for a varying percentage (0%, 5% of the total number of vehicles) of malicious nodes that randomly apply one of the two ways of the selective forwarding attack.

Figure 6(a) shows the performance of both approaches in terms of the packet delivery ratio (PDR). The x-axis represents the different simulated densities, while the y-axis shows the PDR obtained in a scenario where there is not any malicious node. In this scenario, both approaches obtain great results with more than 80% of the packets being delivered to the destination. Analyzing the figure in more detail, we can see that the performance of S-BRAVE is lower than BRAVE for sparse scenarios. This is caused by the false positives occurred during the simulation and their corresponding overhead. During the simulation, guard nodes watching the packets to be forwarded do not receive the forwarded message, making the decision of being themselves the new forwarders. However, the denser the scenario the better performance is obtained from S-BRAVE, reaching the same results as BRAVE (and even outperforming it).

As the percentage of malicious nodes is increased, the performance of both protocols is deteriorated. Taking a look at Figure 6(b), where 5% of the vehicles are malicious, S-BRAVE managed to deliver from 40% to 70% of the packets to their destination. However, BRAVE is only able to deliver from 10%to 30%. S-BRAVE outperforms BRAVE in at least 20%. When density is low, even though S-BRAVE manages to deal with attackers, it may happen that the attacker is the only forwarding alternative. When density is higher it is easier to find guard nodes that can help avoid attacks.



(a) PDR for 0% malicious nodes

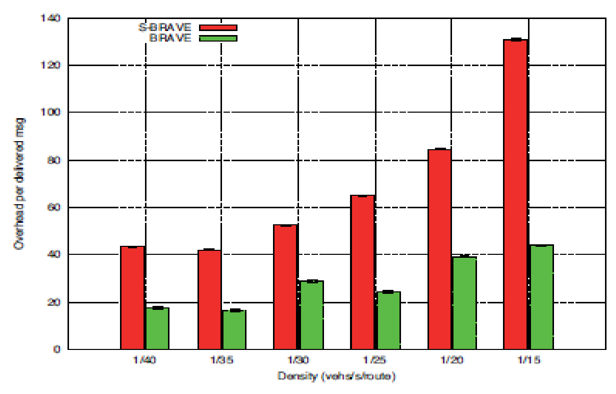
(b) PDR for 5% of malicious nodes

Figure 6: Percentage of PDR for 0% and 5% of malicious nodes.

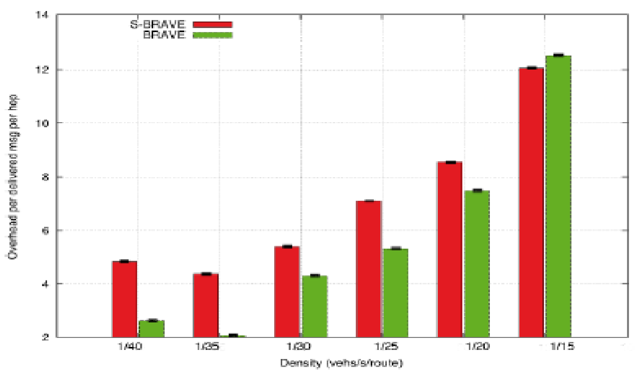
As expected, S-BRAVE has more overhead per successful delivery than BRAVE (Figure 7(a)). In fact, S-BRAVE sends certificates when needed while BRAVE does not use it. However, if we consider that overhead per successful delivery per hop we can see (Figure 7(b)) that S-BRAVE only adds little overhead compared to BRAVE, despite the need of certificates. The reason for the higher overhead in Figure 7(a) is that S-BRAVE manages to deliver packets to destination which are located far from the source (# of hops), while BRAVE just cannot do it.

5. Conclusion

We analyze the problem of secure routing in VANET. In particular, we focus on the BRAVE routing protocol, which is one of the best performing proposals so far.



(a) Overhead of both protocols for 5% malicious nodes per delivered msg.



(b) Overhead of both protocols for 5% malicious nodes per delivered msg per hop.

Figure 7: Overhead (number of BRAVE messages) per hop (two first graphs) and number of delivered packets (last graph).

For this purpose we have introduced a certificate exchange mechanism guaranteeing the authenticity and integrity of the messages as they traverse intermediate nodes until they reach their destination. Besides, we have also developed a way of securing BRAVE against selective forwarding attacks using neighboring nodes as guard nodes. They watch for the message to be sent by the next forwarder and, in case this vehicle does not forward the message, they take the responsibility of sending the message to the next hop.
In order to compare both protocols we have implemented them in NS-2. In light of the results of the previous section. S-BRAVE outperforms BRAVE in terms of PDR. On the other hand, in high dense scenarios, its performance gap compared with BRAVE is up to a 50% of the PDR.

6. References

1. M. Mauve, A. Widmer, and H. Hartenstein, “A survey on position-based routing in mobile ad hoc networks,” IEEE network, vol. 15, no. 6, pp.30–39, 2001.
2. F. Li and Y. Wang, “Routing in vehicular ad hoc networks: A survey, ”IEEE Vehicular Technology Magazine, vol. 2, no. 2, pp. 12–22, 2007.
3. C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve, “A routing strategy for vehicular ad hoc networks in city environments,” in IEEE Intelligent Vehicles Symposium, 2003. Proceedings,2003, pp. 156–161.
4. J. Tian, L. Han, K. Rothermel, and C. Cseh, “Spatially aware packet routing for mobile ad hoc inter-vehicle radio networks,” in Proc. of the IEEE6th Intl. Conf. on Intelligent Transportation Systems (ITSC, vol. 2, 2003,pp. 1546–1551.
5. B. Seet, G. Liu, B. Lee, C. Foh, K.Wong, and K. Lee, “A-STAR: A mobile ad hoc routing strategy for metropolis vehicular communications,” NETWORKING2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications, pp. 989–999, 2004.
6. I. Leontiadis and C. Mascolo, “Geopps: Geographical opportunistic routing for vehicular networks,” in IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007,2007, pp. 1–6.
7. P. Ruiz, V. Cabrera, J. Martinez, and F. Ros, “BRAVE: Beacon-less Routing Algorithm for Vehicular Environments,” IEEE 7th International Conference on Mobile Adhoc and Sensor Systems, MASS 2010, pp. 709–714,2010.