Journal of Computer Science 9 (8): 1069-1078, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1069.1078 Published Online 9 (8) 2013 (http://www.thescipub.com/jcs.toc)

FORMAL VERIFICATION OF SAFETY MESSAGE DISSEMINATION PROTOCOL FOR VANETS

[1]M.A. Berlin and [2]Sheila Anand

[1]Department of Computer Science and Engineering,
R.M.D. Engineering College, Anna University, Chennai, India
[2]Rajalakshmi Engineering College, Anna University, Chennai, India

Received 2013-05-22, Revised 2013-07-03; Accepted 2013-07-09

ABSTRACT

This paper presents a formal verification of a safety message dissemination protocol used in vehicular ad-hoc networks. It is proposed to use Road Side Units to broadcast road hazard information to vehicles travelling on highways. Quick dissemination of road hazard information, like road blocks, slippery roads and other obstacles can help to prevent road accidents and improve passenger safety. Formal verification is a mathematical approach that helps developers to validate the protocol and correct design errors. The well known model checker, SPIN has been used to model the possible behavior of the protocol and provide formal verification of the correctness of the protocol.

Keywords: Safety Message Dissemination Protocol, Road Side Units, Road Hazard, Highways, Formal Verification, SPIN Model Checker

1. INTRODUCTION

Vehicular Ad-hoc NETwork (VANET) is an infrastructureless network which is formed by vehicles travelling on roads. Each vehicle is equipped with wireless communicating devices capable of communicating with each other. There are two types of VANET communications: Vehicle-to-Vehicle (V2V) and Vehicle-to-Road side units (V2R). In V2V, vehicles are used as routers to transmit messages between each other. In V2R, Road Side Units (RSUs) are used to transmit important information to vehicles (Momeni and Mahmood, 2008).

VANET applications can be broadly classified as Safety and Comfort (non-safety) applications. Comfort applications provide additional services like map download, video streaming, web access, entertainment and advertisements (Berlin and Anand, 2011). Safety applications are used to improve the safety of driver and passengers by transmitting safety and road hazard related information among vehicles. Information about road hazards like tree fall, boulder on road and other road condition are sent to vehicles to warn the drivers and enable them to take corrective action. Other safety related information includes immediate collision warning, forward obstacle detection and avoidance, lane changing warning and other emergency message dissemination (Kamini and Kumar, 2010).

The conventional approach of implementing and validating a communication protocol is through network simulation and testing. Simulations are inexpensive and describe the behavior of the system through sequence of steps specified in the algorithm and it can simulate only a limited set of blocks of code in any considerable amount of time. When the design size of the model grows, then it becomes difficult to cover such huge and complex models through simulation (Anamaya *et al*., 2010). Testing can show the presence of errors in the design but it can never conclusively prove the absence of errors. Another approach is to use formal verification to validate safety critical system and establish their correctness (Bowen and Hinchey, 1995). It requires a model of a

Corresponding Author: M.A. Berlin, Department of Computer Science and Engineering, R.M.D. Engineering College, Anna University, Chennai, India

system to be verified, set of states, variables, transitions between processes and set of design rules. It should however be noted that simulation, testing and formal verification are all quality assurance techniques that complement each other (Katoen, 1999).

Formal verification is a mathematical approach used for verifying software and hardware, based on a series of formal proofs specified in the form of mathematical argument to guarantee that a formal specified system has/has not a specific property (Camara, 2009). Since formal verification is a successful tool for performing exhaustive verification, it is used to verify communication protocols which have complex models. This method is better than testing approach because it can provide a complete verified system with absence of errors such as deadlock, live-lock, cycles and forming loops and also provides fully automated system (Islam *et al.*, 2006).

There are three kinds of formal verification techniques: Model checking, Theorem Proving and Equivalence Checking. Model checking, normally, uses an exhaustive search to check the properties specified in the system and in which verification can be done automatically (Holtzman, 1997). During verification, if the given property of the system is satisfied, then it returns “yes”; otherwise produces “no” answer. When the given property of the system fails, then the model checker provides a counterexample to examine the improper behavior (Clarke *et al.*, 1999). Theorem proving technique uses design axioms and rules to verify and prove the correctness of the system. Equivalence checking is mostly used in industries for checking two different implementations of the same system to verify if the behavior is the same (Kern and Greenstreet, 1999).

In this paper, we present formal verification of the proposed safety message dissemination protocol using SPIN model checking tool. In Section 2, the related work is discussed. In section 3, the safety message dissemination protocol is presented and discussed. In section 4, the implementation and verification of protocol using SPIN model checker has been given. Section 5 concludes the paper.

2. RELATED WORK

Formal verification is a method to check various properties such as liveliness of processes, absence of deadlock and design errors. The use of formal verification for the design and development of communication protocols is discussed. SPIN is one of the efficient model checkers that can be used for formal verification of distributed software systems. SPIN uses a high level language, called PROcess MEta LAnguage (PROMELA), as a formal verification language (Holtzman *et al*., 1991).

Obradovic (2002) and Bhargavan *et al*. (2002) used formal verification to identify loop formation error in AODV implementation (Perkins and Royer, 1999). He was able to extend that work to ensure loop free routes to the destination by modifying and verifying the AODV protocol using SPIN model checker. The author has shown the looping scenarios of AODV and was able to correct the design errors and prove that the modified protocol was loop free.

Renesse and Aghvami (2004) have presented a new technique to verify Wireless Adaptive Routing Protocol (WARP) using SPIN model checker. Since the system had more states, the authors used super-trace mode to verify the protocol. The authors were able to achieve 98% of correctness for the 5-node model of WARP.

Wibling *et al*. (2004) have evaluated two model checking tools SPIN and UPPAAL on Lightweight Underlay Network Ad hoc Routing Protocol (LUNAR) to verify the properties specified in the ad hoc protocol. The authors verified for the absence of deadlock, data and control flow message requirements using SPIN model checker and the timing constraints were verified using UPPAAL tool.

Camara *et al*. (2009) have discussed the techniques and tools available for the formal verification of routing protocols in wireless ad hoc networks. They have discussed the three types of formal verification techniques such as model checking, theorem proving and equivalence checking. They have also presented different tools such as Higher Order Logic (HOL), SPIN, Colored Petri Nets (CPN) and UPPAAL that can be used for verifying ad hoc protocols. They have considered Optimized Link State Routing (OLSR) protocol for case study and explained the behavior of such protocol using SPIN model checker.

Javed *et al*. (2010) implemented Distance Vector Routing (DVR) protocol using PROMELA Language and tested using SPIN model checker to verify the behavior of DVR. They have shown the simulated results for finding distance and path between routers by varying time periods. When a shortest path was found between routers, routing table was updated automatically at each router. The authors could find optimized result of the protocol at various time intervals. They also planned to extend their work for reducing storage space requirements, reduce congestion at high density of the network and also to include security mechanism.

3. SAFETY MESSAGE DISSEMINATION PROTOCOL

Numerous works have been carried out by the researchers to transmit safety related information using

V2V communication (Bai *et al*., 2009; Lee *et al*., 2010; Koubek *et al*., 2010). In V2V communication, safety messages are typically broadcast using flooding technique or other vehicles as forwarders. Hence, V2V communication is more suitable for networks where the density of vehicle traffic is high. Moreover, V2V communication have drawbacks of high network overhead, low packet delivery ratio due to packet loss and generates broadcast storm problem.

The focus of this protocol is propagation of hazard information on highways where the vehicle traffic may be sparse. V2V communication solutions may not be feasible and hence it is proposed to use RSUs for safety message dissemination. Current research shows that RSUs are emerging as a viable economic option for message dissemination. RSUs can be efficiently and reliably used to route packets in VANET, especially when users who are far apart want to communicate (Mershad *et al*., 2012). Sou and Tonguz (2011) have analyzed the connectivity and routing performance when RSUs are deployed for broadcast based safety applications.

The primary objective of the proposed protocol is to route road hazard information to the far away vehicles approaching the location of the hazard. This would enable the vehicles to take advance corrective action to avoid the hazard, like re-routing and thereby also prevent road congestion. The protocol proposes the use of RSUs to verify and disseminate safety message information to vehicles on highways where vehicle traffic may be light or sparse.

The main contributions of the proposed protocol, which is an extension of our previous work (Berlin and Anand, 2012) are outlined below:

- The proposed protocol can be deployed in highways where vehicular traffic is sparse. Most of the existing protocols use V2V communication which is more suited for dense traffic
- Use of RSU enables reliable and speedy delivery of the hazard messages
- The proposed protocol focuses on the verification and validation of the correctness of the received hazard message

This paper explains the proposed protocol and focuses on the formal verification of the protocol. Formal verification is a mathematical based technique that can be used by developers to comprehensively simulate and verify communication protocols. This approach helps to find design errors which can be subsequently corrected. The absence of deadlock can be established. This is an automated approach that helps to improve the reliability and correctness of the protocol.

It is assumed that each vehicle has On-Board Unit (OBU) to detect the hazard and also Global Positioning System (GPS) to know its location. When a vehicle comes across a road hazard, it will generate a Hazard Message (HM) and transmit the message to the nearest RSU. RSU will in turn transmit the message to vehicles and neighboring RSUs to enable vehicles to re-route and avoid the hazard.

It is assumed that the RSUs would be deployed on the highways to enable transmission coverage of the entire highway. It is further assumed that the RSUs would be able to communicate with each other to exchange messages to notify the presence and location of hazard. The protocol is explained using a sample scenario. A highway would typically consist of vehicles travelling in both directions as shown in **Fig. 1**.

Each RSU would periodically broadcast hello message giving its ID and location. Vehicles travelling within the transmission range of the RSU would receive the hello message and transmit a hello_reply message to the RSU giving details of its ID, location and speed. It is assumed that each vehicle would have a unique veh_id that would be available in OBU. **Figure 2** depicts the exchange of hello and hello_reply message between RSU and vehicles.

RSU will maintain a Vehicle TaBle (VTB), to store the details of vehicles within its transmission or communication range. VTB would be updated using the hello_reply messages received from the vehicles. RSU would periodically go through the table and remove entries of vehicles from which reply messages are no longer received because they have gone out of range. The creation of VTB has been modeled in Promela as:

```
typedef VTB
{
        byte veh_id;
        bool dir_of_travel;
        byte speed;
        byte veh_loc;
        byte time;
}
```

The dir_of_travel in VTB is updated based on the GPS location received from the vehicle in its hello_reply message. If the vehicle is travelling in the side in which the RSU is located, then dir_of_travel would be taken as forward direction and the value would be stored as 1 and the dir_of_travel of vehicles travelling in the opposite direction would be considered as 0. For the given scenario in **Fig. 1**, the VTB in RSU would have entries for vehicles V and v1 to v14.

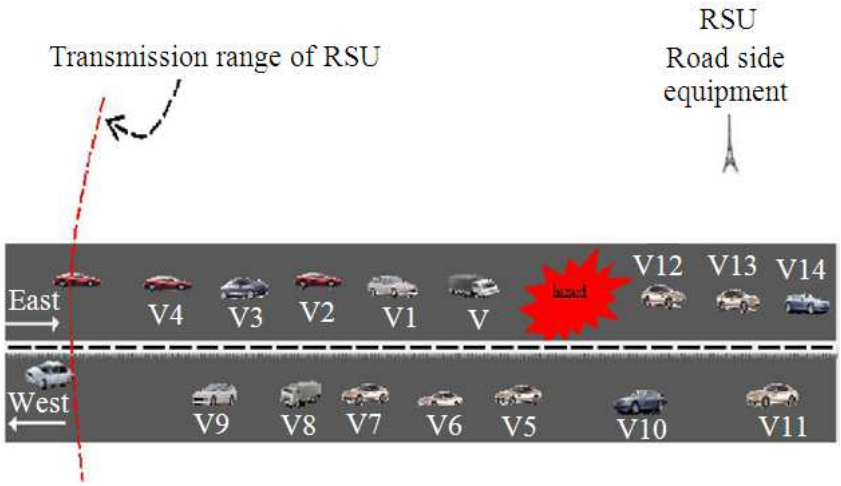

Fig. 1. Highway scenario with hazard

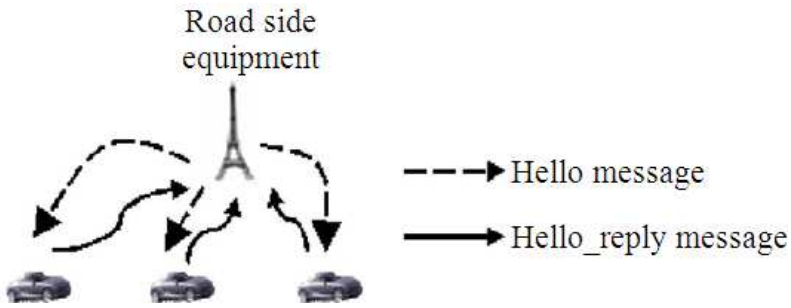

Fig. 2. Exchange of hello and hello_reply message

Vehicle V would be the first vehicle to encounter the road hazard along its travelling path. It would generate a HM giving details of its ID, location of vehicle, speed and hazard location and broadcast the same to RSU. RSU would have to first verify the validity of the HM received from V before sending the Hazard Message (HM_RSU) to the other vehicles.

If the hazard information is correct, then RSU should also receive HM from other vehicles travelling in the same direction as V. From VTB, it would be able to determine which vehicles within its transmission range are likely to encounter the hazard and send a hazard message. Hence it waits for 't' seconds, during which it receives HMs from other vehicles. 't' is determined based on the transmission range of the RSU, so that it is able to receive the message from the farthest vehicle in its range.

RSU would maintain a Hazard TaBle (HTB) to store the HMs received from vehicles during the timer period. HTB is modeled in PROMELA as:

```
typedef HTB
{
    byte veh_id;
    byte speed;
    byte veh_loc;
    byte haz_loc;
    byte time;
    bool haz_dir;
}
```

HTB would be updated only the first time a message is received from a specific vehicle. If a particular vehicle sends more than one message, then duplicate messages from the vehicle would be ignored and not updated in the table. In other words, HTB would contain only one entry for each vehicle from which it has received the hazard message.

In PROMELA, the verification of HM at RSU could be modeled as follows. Let htb[] be the array used for storing hazard information received from vehicle V and another vehicles.

```
inline
verification_at_RSU(vtb_ctr,htb_count)
{
byte  lkey_ctr = 0, htb_ctr = 0;
byte ac = 0,TH;

/* vtb_ctr is the count of vehicles in
VTB */
do
  ::ac < vtb_ctr ->
   if
    :: vtb[ac].dir_of_travel== haz_dir->
      lkey_ctr ++;ac++;
    ::else ->skip;
   fi;
  ::else -> break;
od;

/* threshold  (TH)  set at 50% */

TH =  lkey_ctr / 2;
ac = 0;

/*  haz_loc is the hazard location */
/*  htb[ac].haz_loc is the hazard
location stored at HTB table */

do
  ::ac <= htb_count ->
   if
   :: htb[ac].haz_loc == haz_loc ->
      htb_ctr ++;ac++;
   ::else ->skip;
   fi;
  ::else-> break;
od;

if
  :: htb_ctr > TH ->
     Vflag == True; /* Accept */
  :: else ->
      Vflag == False; /* Reject */

fi;
}
```

Fig. 3. HM verification modeled in PROMELA

Counter *htb_count* is used to compute the count of entries updated in HTB. *Haz_loc* is used to store the location of hazard given in the first HM. When timer 't' exceeds, RSU would read through HTB and increment the received count *htb_ctr* for every message it has received.

To verify the validity of HM, RSU would compare the actual messages received with the messages it is likely to receive. The messages, it is likely to receive *lkey_ctr* is computed from VTB based on the direction of travel of the vehicles and the location of the hazard. If the difference exceeds a preset Threshold (TH), then the message is taken as authentic. It is not possible to expect that all vehicles would send HM and it is also possible that all HM may not be received due to transmission loss, hence a preset threshold is used. In PROMELA, HM verification is modeled as given in **Fig. 3**.

For the highway scenario given in **Fig. 1**, RSU would receive HM from V. It would be able to determinethat it should also receive the HM from vehicles v1 to v4. Vehicles v5 to v9 are travelling in the opposite direction and would not detect the hazard. Hence RSU should receive four HM messages from v1 to v4. If the Threshold (TH) value is set to 50%; then for the sample scenario, the HM would be considered valid by RSU if at least two HM are received.

If the HM is verified to be valid, then RSU would broadcast a HM_RSU to all vehicles giving the location of the hazard. The protocol also proposes that RSU receiving the HM would also transmit the HM to neighboring RSUs preceding the hazard. RSUs would broadcast the HM to all vehicles in their transmission range to enable them to take advance corrective action to re-route and avoid the hazard.

Communication between vehicles and RSUs takes place through message channels. Communication channels for RSU are modeled in PROMELA as follows:

```
chan fromRSU[2] = [N] of {mtype, byte};
chan toRSU[2] = [N] of {mtype, byte};
```

Two channels are specified for each RSU: From RSU the send channel and to RSU is the receive channel. The send channel is used to transmit messages to other nodes and receive channel is used for receiving messages from other nodes. N indicates the maximum messages that can be sent or received through these channels.

Communication channels have been modeled in a similar manner for vehicles. Broadcasting from RSU is modeled as channel-to-channel unicast message transmission to vehicles.

4. PROTOCOL VERIFICATION USING SPIN

The quality and correctness of the safety dissemination protocol is examined through SPIN model checker. SPIN is a powerful model checker and one of the commonly used tools for verifying communication protocols. The protocol behavior is specified using a formal validation model.

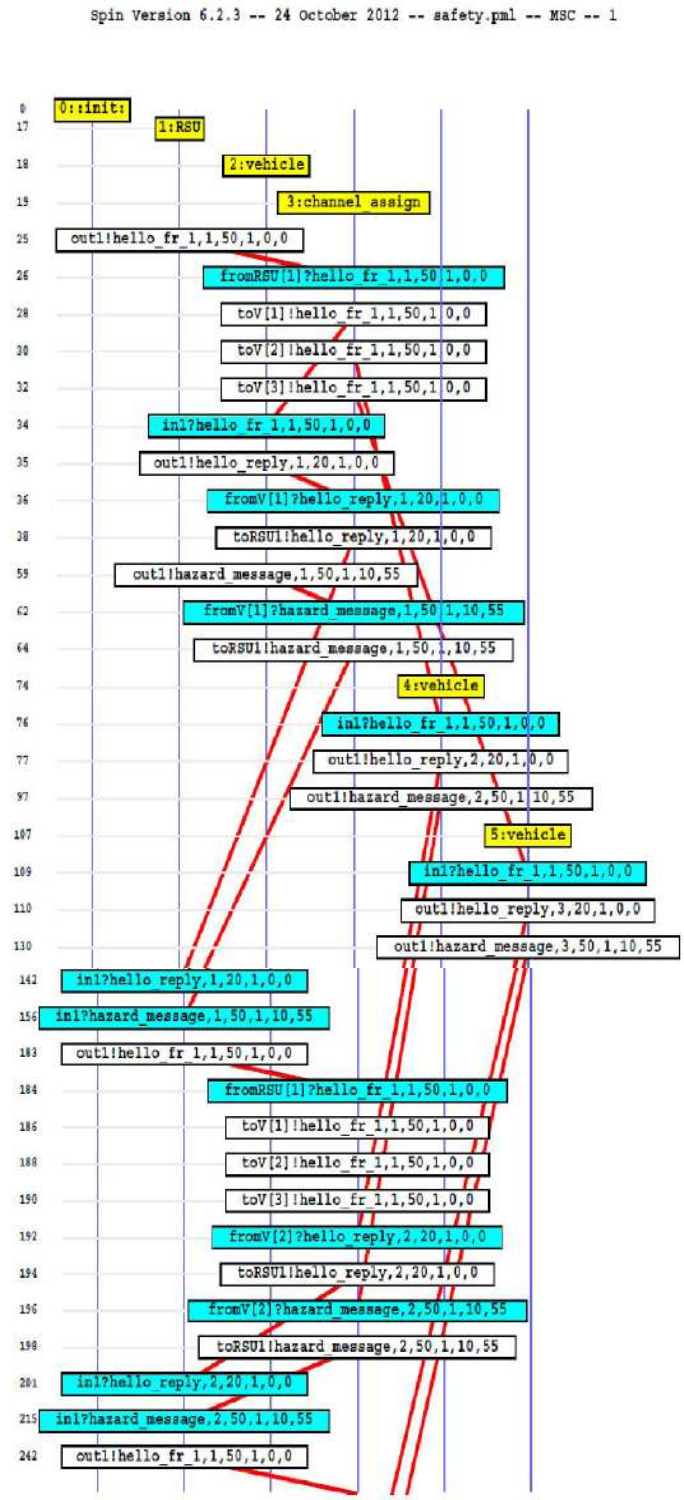

Fig. 4. Message sequence chart

```
<Spin Version6.2.3 – 24 October 2012>
        + Partial Order Reduction
Full statespace search for:
        never claim             + <p0>
        assertion violations    + <if within scope of claim>
        cycle checks            - <disabled by -DSAFETY>
        invalid end state       - <disabled by never claim>

State-vector 1272 byte, depth reached 785, errors:  0
        161368 states, stored
        134326 states, matched
        295694 transitions < = stored + matched>
            4  atomic steps
hash conflicts:          264 <resolved>
```

Fig. 5. Verification result from SPIN tool

Exhaustive verification using SPIN is used to establish whether the model behavior is error-free.

System components of SPIN are specified as a set of processes. The processes interact with each other by passing messages through channels, global variables or via sharing memory. The coding is done using PROMELA and iSpin version 6.2.3 graphical environment has been used to simulate and validate the behavior of the developed protocol. During simulation and execution, SPIN generates C code for the developed PROMELA code to verify the system properties.

4.1. Simulation

The functionality of the vehicles and RSUs has been defined using processes. Channels have been defined for the vehicles and RSUs to interact with each other. Messages are passed through these channels. The network has been modeled for the proposed protocol using ten vehicles (nodes). Five of these vehicles are assumed to be moving in forward direction and the remaining in the opposite direction. The protocol is verified to check whether communication takes place in the intended manner between RSUs and the vehicles. A portion of the Message Sequence Chart (MSC) obtained by simulation of the proposed protocol using SPIN model checker is shown in **Fig. 4**.

The red bars show the transmission of messages from one process to other process. The messages sent and received are shown graphically.

The modeling process is further explained. Initially three vehicles are assumed to be in the range of RSU. RSU broadcasts a hello message to indicate its location to vehicles in its range. The broadcast message is modeled as messages sent to each of the three vehicles. The vehicles receive the *hello* message and respond with a *hello_reply* message giving their current location, speed and time. RSU would receive these replies and update the vehicle details in VTB as explained in Section 3.

The hazard is assumed to be at location 55. Vehicle 1 would first encounter the hazard and send a hazard message to RSU. RSU would receive the hazard message and update its HTB table. As there are two other vehicles within its range and range of the hazard, it will wait till it receives the hazard message from the other two vehicles. MSC shows that vehicles 2 and 3 also send hazard messages to RSU. RSU would take the hazard message as correct only if it receives these two other hazard messages.

4.2. Verification

The verification output from the SPIN tool of the proposed protocol is shown in **Fig. 5**. SPIN model checker executes the PROMELA code and also verifies the absence of deadlocks or live-locks during the execution of the protocol. A deadlock occurs when further execution is not possible as the protocol is waiting for a condition that cannot be met. Live-locks

occurs when execution sequences are repeated indefinitely without termination.

The verification result shows that there are no errors during the execution of the protocol. SPIN does not construct the complete state space; instead it is done dynamically during processing. The number of depth reached during exhaustive search and the number of states and transitions explored are shown in the result.

Invalid end-state, assertions and never claims were also checked during the verification process. This is explained as follows.

4.2.1. Invalid End-State

PROMELA executes each process of the protocol to examine whether it reaches a valid end-state. Valid end states are reached when the process reaches the end of the defined process and the channels are all empty. In the proposed protocol, process RSU would not reach valid end state due to two reasons. The first reason, it would have to transmit the *hello* message periodically to vehicles within its transmission range. The second reason, it would have to be in ready state to receive the hazard messages from the vehicles.

4.2.2. Assertion Violations

Assertion violation is a valid PROMELA expression which can be expressed as Boolean condition. This expression is an executable statement which must be satisfied each time the process reaches the state where assertion condition is given. When the evaluation of the specified expression produces false answer or a zero result, then an assertion violation error would be reported to the user.

In the proposed safety message dissemination protocol, two assertion conditions were evaluated. The first condition, asserts that only vehicles which are travelling in the direction where hazard is present, would transmit the hazard message. This condition is coded in PROMELA as:

```
assert (haz.haz_dir==veh.dir_of_travel)
```

Second assertion was used in verification of hazard message process. RSU will start a timer after receiving the first hazard message. Hazard messages from other vehicles are received during the timer process. The verification process must start only after the timer ends. This was modeled as:

```
assert (timer == EndTimer)
```

The modeled code of the proposed protocol did not encounter any assertion violation error as seen from the results obtained.

4.2.3. Never Claims

SPIN also supports an important correctness property called “never claims”. Never claims are used to verify safety properties of the system being developed. The properties are generally expressed in the form of LTL (Linear Temporal Logic). The following properties were set in our proposed system.

4.2.3.1. *Property LTL p0*

This LTL property was used to validate the *HM* verification phase at RSU. RSU would not send hazard message to vehicles unless verification is successful. The equivalent LTL claim was modeled as:

```
[] (!p U q)
```

[] symbol denotes always. ‘p’ is defined as message type; hazard message sent by RSU. ‘q’ is defined as a Boolean variable which is set to True only when verification is successful. Claim violation error was not reported during the verification of the proposed protocol.

4.2.3.2. Property LTL p1

This LTL property is defined for valid update of HTB table. HTB table must not be updated when duplicate messages are received from the vehicles. The equivalent LTL claim was modeled as:

```
[] ( p ) -> []r
```

‘p’ is defined to check whether the vehicle ID is not duplicated and is defined as (veh_id !=veh_id.HTB). The vehicle ID of the received vehicle must not be already present in HTB. ‘r’ indicates that HTB table is updated only when ‘p’ is true. Claim violation error was not reported during the verification of the proposed protocol.

5. CONCLUSION

SPIN was found to be very useful and efficient for protocol modeling, testing and verification. The behavior of the proposed safety dissemination protocol was simulated and verified using SPIN. The communication between RSU and vehicles was simulated and safety properties like absence of deadlock, invalid end-states, assertions and never claims were verified. As future work, it is proposed to extend the protocol to model RSUs with

greater interspacing. Portions of the highway between the RSUs would then be outside the range of the RSUs and vehicles travelling on the highway may have to be used as forwarders of hazard messages to the RSUs.

6. REFERENCES

Anamaya, K., B.A. Sullerey and A. Jain, 2010. Formal verification of an ASIC ethernet switch block. Proceedings of the Formal Methods in Computer-Aided Design, Oct, 20-23, IEEE Xplore Press, Lugano, pp: 13-20.

Bai, S., Z. Huang, D. Kwak, S. Lee and J. Jung *et al*., 2009. Vehicular multi-hop broadcasting protocol for safety message dissemination in VANETs. Proceedings of the 70th Vehicular Technology Conference Fall IEEE, Sep. 20-23, IEEE Xplore Press, Anchorage, AK., pp: 1-5. DOI: 10.1109/VETECF.2009.5378794

Berlin, M.A. and S. Anand, 2011. Review of safety data dissemination and safety applications in VANET. Global J. Comput. Appli. Technol., 1: 254-269.

Berlin, M.A. and S. Anand, 2012. Driving hazards message propagation using road side infrastructure in VANETs. Adv. Comput. Sci. Inform. Technol., 86: 1-8. DOI: 10.1007/978-3-642-27317-9_1

Bhargavan, K., D. Obradovic and C.A. Gunter, 2002. Formal verification of standards for distance vector routing protocols. J. ACM, 49: 538-576. DOI: 10.1145/581771.581775

Bowen, J.P. and M.G. Hinchey, 1995. Seven more myths of formal methods. IEEE Soft., 12: 34-41. DOI: 10.1109/52.391826

Camara, D., 2009. Formal verification of communication protocols for wireless networks. Federal University of Minas Gerais.

Camara, D., A.A.F. Loureiro and F. Filali, 2009. Formal Verification of Routing Protocols for Wireless Ad Hoc Networks. In: Guide to Wireless Ad Hoc Networks, Woungang, I. and S.C. Misra (Eds.), Springer, London, ISBN-10: 1848003279, pp: 189-210.

Clarke, M.E., O. Grumberg and D.A. Peled, 1999. Model Checking. 1st Edn., MIT Press, Cambridge, ISBN-10: 0262032708, pp: 314.

Holtzman, G., J. Englewood and N.J. Cliffs, 1991. Design and Validation of Computer Protocols. 1st Edn., Prentice Hall, Englewood Cliffs, ISBN-10: 0135399254, pp: 500.

Holtzman, G.J., 1997. The model checker SPIN. IEEE Trans. Soft. Eng. 23: 279-295. DOI: 10.1109/32.588521

Islam, S.M.S., M.H. Sqalli and S. Khan, 2006. Modeling and formal verification of DHCP using SPIN. Int. J. Comput. Sci. Appli., 3: 145-157.

Javed, K., A. Kashif and E. Troubitsyna, 2010. Implementation of SPIN model checker for formal verification of distance vector routing protocol. Int. J. Comput. Sci. Inform. Secu., 8: 1-6.

Kamini and R. Kumar, 2010. VANET parameters and applications: A Review. Global J. Comput. Sci. Technol., 10: 72-77.

Katoen, J., 1999. Concepts, Algorithms and Tools for Model Checking. Friedrich-Alexander Universität Erlangen Nürnberg.

Kern, C. and M.R. Greenstreet, 1999. Formal verification in hardware design: A survey. ACM Trans. Design Automation Eelectr. Syst., 4: 123-193. DOI: 10.1145/307988.307989

Koubek, M., S. Rea and D. Pesch, 2010 Event suppression for safety message dissemination in VANETs. Proceedings of 71st Vehicular Technology Conference, May 16-19, IEEE Xplore Press, Taipei, Taiwan, pp: 1-5. DOI: 10.1109/VETECS.2010.5494204

Lee, D., S. Bai, T. Kim and J. Jung, 2010. Enhanced selective forwarding scheme for alert message propagation in VANETs. Proceedings of the International Conference on Information Science andApplications, Apr. 21-23, IEEE Xplore Press, Seoul, pp: 1-9. DOI: 10.1109/ICISA.2010.5480516

Mershad, K., H. Artail and M. Gerla, 2012. We can deliver messages to far vehicles. IEEE Trans. Intell. Trans. Syst., 13: 1099-1115. DOI: 10.1109/TITS.2012.2183124

Momeni, S. and F. Mahmood, 2008. VANET's communication. Proceedings of the 10th International Symposium on Spread Spectrum Techniques and Applications, Aug. 25-28. IEEE Xplore Press, Bologna, Italy, pp: 608-612. DOI: 10.1109/ISSSTA.2008.115

Obradovic, D., 2002. Formal analysis of routing protocols. University of Pennsylvania.

Perkins, C.E. and E.M. Royer, 1999. Ad-hoc on-demand distance vector routing. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, Feb. 25-26, IEEE Xplore Press, New Orleans, LA., pp: 90-100. DOI: 10.1109/MCSA.1999.749281

Renesse, R.D. and A.H. Aghvami, 2004. Formal verification of Ad-Hoc Routing protocol using SPIN model checker. Proceedins of the IEEE MELECON, Croatia, May 12-15.

Sou, S.I. and O.K. Tonguz, 2011. Enhancing VANET connectivity through roadside units on highways. IEEE Trans. Vehicular Technol., 60: 3586-3602. DOI: 10.1109/TVT.2011.2165739

Wibling, O., J. Parrow and A. Pears, 2004. Automatized verification of ad hoc routing protocols. Lecture Notes Comput. Sci., 3235: 343-358. DOI: 10.1007/978-3-540-30232-2_22