INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 2)
Available online at: www.ijariit.com

Efficient privacy-preserving dual authentication and key agreement scheme for secure V2V communications in an IOV paradigm

Aishwarya .R
aish14102.cs@rmkec.ac.in
RMK Engineering College, Kavaraipettai, Tamil Nadu

Tanuja .A
avul14110.cs@rmkec.a.cin
RMK Engineering College, Kavaraipettai, Tamil Nadu

Vineela .Y
vine14333.cs@rmkec.a.cin
RMK Engineering College, Kavaraipettai, Tamil Nadu

Dhanlakshmi .R
rdl.cs@rmkec.ac.in
RMK Engineering College, Kavaraipettai, Tamil Nadu

ABSTRACT

In this paper, we focus on the security and privacy-preserving by developing a dual authentication scheme for IoV according to its different scenarios. First, the OBU self-generates an anonymous identity and temporary encryption key to open an authentication session. Second, the legitimacy of the vehicle's real and anonymous identity can be verified by trust authority (TA). After that, the vehicle's reputation is evaluated according to its history interactive behavior and the session key for V2V can be finally established. There are three major advantages, including privacy-preserving and security enhancement without a burden of key management in the condition of acceptable time delay range, introducing trust evaluation into authentication protocol, as well as considering the vehicle behavior attributes in the new reputation evaluation method.

Keywords: *Authentication, Wireless Networks, Encryption.*

1. INTRODUCTION

Wireless ad hoc networks (i.e., decentralized networks created on the fly by hosts located in the proximity of one another) are no longer just a research concept. Due to their aptitude to require minimal effort to setup, ad hoc networks are suitable for a wide range of applications, including battlefields communications and disaster recovery operations. In August of 2008, researchers at the National Institute of Standards and Technology (NIST) demonstrated an ad hoc network prototype for first responders in building fires and minescollapse. Unmanned vehicle(aerial, terrestrial, and aquatic) with the autonomic operation of a few hours, already can be sent to regions where human presence is deemed dangerous and they can form networks on the fly to report observations to command and control centers. When the hosts (or nodes) of an ad network are mobile, the network is called a mobile ad hoc network (MANET). Federal Communication Commission (FCC) in the United States has allocated a bandwidth of 75MHz around the 5.9GHz band for the vehicle to vehicles and vehicles to road side infrastructure communications through the Dedicated Short Range Communications (DSRC) services. The emergence of vehicular networks would enable several useful applications, both safety and non-safety related, such as automatic road traffic alerts dissemination, dynamic route planning, service queries (e.g., parking availability), audio and video file sharing between moving vehicles, and context-aware advertisement. To deploy these services, three types of communications involving moving vehicles are considered, including a cellular network, a vehicle to roadside infrastructure and ad hoc vehicle communications.

Brief descriptions of each of these types of communication are provided below. Note that hybrids mean of communication involving combinations of the methods described here.

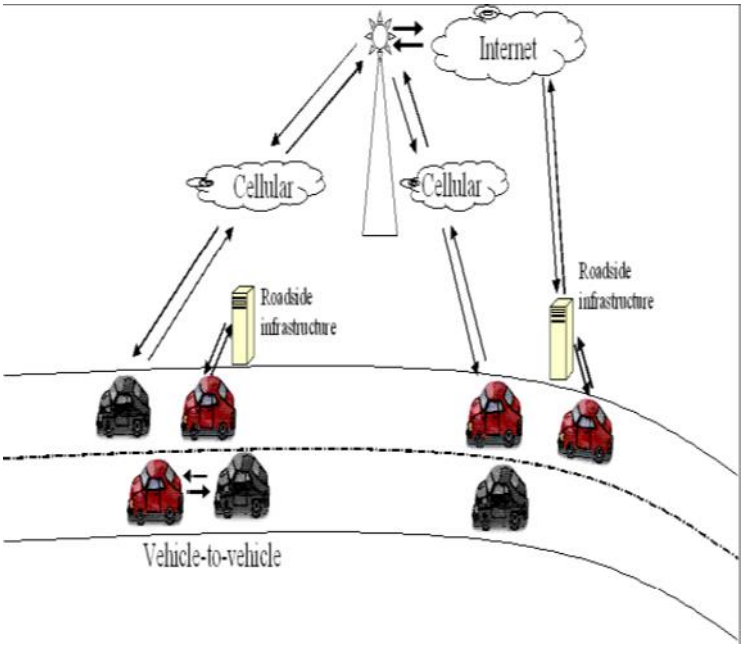


Figure 1.1 Vehicular networks can be formed in three ways: using the cellular network, roadside infrastructure or vehicle-to-vehicle communications.

2. COMMUNICATIONS THROUGH CELLULAR NETWORK

The first method connects vehicles to the Internet through cellular data networks using any of the following technologies: EV-DO, 3G, GPRS, etc. This service is already commercially available from car manufacturers [7] and from other third-parties. In most commercially available solutions, the vehicle is transformed into an IEEE 802.11 (WIFI) hotspot and the Internet connection can be shared by many computers in the car.

Usually, a limit is set on the amount of data transfer (e.g., 1GB or 5GB maximum per month). The main advantage of this method of connection is that the vehicle will have Internet access wherever cellular coverage is available. The main drawbacks are the dependence on the cellular operator coverage network and the limited available data rates (rates vary around 500Kbps-800Kpbs).

3. VEHICLE TO ROADSIDE INFRASTRUCTURE COMMUNICATIONS

The second method uses roadside infrastructure. Here, vehicles connect to other vehicles or to the Internet through roadside access points positioned along the roads. Two main variants can be found in the literature: the access points could be installed specifically for the purpose of providing Internet access to vehicles or the latter could make use of open 802.11 (Wi-Fi) access points encountered opportunistically along city streets The advantage of this method of connection is that vehicles will be able to connect to the Internet using much higher data rates (e.g., 11Mbps) than through the cellular network. The drawbacks include the cost related to installing access points along the roads to obtain reasonable coverage. Additionally, in the case where open access points are used, the access points owners' consent would legally be required before such a service is deployed [18].

4. VEHICLE-TO-VEHICLE (AD HOC) COMMUNICATIONS

Using Internet-based communications to and from vehicles will probably remain the method of choice for communications as long as the ratio of Wi-Fi-enabled vehicles remains low. However, the prevalence of Wi-Fi-ready vehicles will open the way for ad hoc networks of moving vehicles. The advantage here is the addition of a distinct, high bandwidth network to the existing infrastructure network. The main drawback is that these networks could require a new set of protocols as the viability of vehicular networks applications described above is conditioned by whether or not VANET routing protocols are able to satisfy the throughput and delay requirements of these applications.

This dissertation addresses the problem of efficient routing and forwarding in VANET.VANETs were selected for this study because, among the vehicular networks, the ad hoc configuration has the greater potential of widespread use: it is scalable (compared to cellular communication), low-cost, and provides higher bandwidth. Even though VANETs show great promise, their success is dependent on whether VANET routing protocols are able to satisfy the throughput and delay requirements of applications deployed on these networks.

5. ROUTING AND FORWARDING CHALLENGES IN VANETS

To better understand the challenges brought by VANETs, it is important first to understand the characteristics of these networks.

6. CHARACTERISTICS OF VEHICULAR AD HOC NETWORKS

VANETs are characterized by (a) high node mobility, (b) constrained nodes movements (c) obstacles-heavy deployment fields, and (d) a large number of nodes, which all add to the communication challenges. First, vehicles are continually moving along the roads at higher speeds than in a MANET. Thus a VANET will present a continually changing structure, and communication links are

expected to be valid for few minutes or seconds. Next, the movements of vehicles are constrained on roads, hence the existing roadmaps put a limit to the topologies available in VANETs when compared to MANETs. Then, the presence of 6 high-rise buildings and houses between streets impacts the propagation of wireless waves through reflections and refractions. Finally, VANETs have the potential to contain a very large number of nodes as any vehicle can be part of the network. It is assumed that each vehicle is equipped with a Geographical Positioning System (GPS), digital maps or navigation system and an ad hoc wireless communication device.

7. ROUTING CHALLENGES

Analyzes of traditional routing protocols for MANETs demonstrated that their performance is poor in VANETs. The main problem with these protocols (e.g., in VANETs environments is their route instability, which leads to packets drops, increased overhead from route repairs, low delivery ratios and high transmission delays An alternative routing approach is offered by geographical routing protocols (e.g., GPSR), which decouple forwarding from the nodes identity; they do not establish routes, but use the position of the destination and the position of the neighbor nodes to forward data. Any node ensuring progress toward the destination can be used for forwarding. Yet, it runs the risk of packets being dropped at dead end streets because no consideration is given to the roads layouts. The question then is whether integrating VANETs features (road topology, real-time road traffic flow, the presence of the building, etc.) in the design of routing protocols would lead to better performance.

8. FORWARDING CHALLENGES

The characteristics of VANETs also impact the forwarding of packets. Three main forwarding challenges were identified: next hop selection, queuing disciplines, and paths durations. Protocols such as DSR or GPSR maintain lists of neighbors, which are used to determine the next hop. If the lists are not accurate, the best next hop could be missed, or even worse, a vehicle node which is already out of the transmission range could be chosen. Maintaining up-to-date lists requires frequent “hello“ packet broadcasting. Yet, too much broadcasting will result in a large communication overhead. Thus, the question is how to use accurate node positions in the selection of the next hop without incurring too much overhead Vehicular ad hoc networks often experience congestion faster than well-designed wired networks, leading to high end-to-end delays and jitter even for moderate traffic. This particularly impacts delay sensitive but loss tolerant applications such as traffic or accident monitoring. The choice of queuing discipline had been shown to impact the performance of data transfers in wired IP networks, where TCP was proved to perform better under congestion when routers use FIFO with Front drop instead of FIFO with a Tail drop or RED. The question then is whether ad hoc networks can achieve better end-to-end delay and jitter with a different queuing discipline. The final forwarding challenge considered deals with exploiting the knowledge of routing paths duration to improve the performance of RBVT. Often, a node in a vehicular ad hoc network will try to establish a communication path when the destination is unreachable. Other times, the path will be established only to have it break a few seconds later due to the movements of nodes.

9. NS2 STRUCTURE

NS2 is an object-oriented simulator, written in C++, with a Tcl interpreter as a front-end. The simulator supports a class hierarchy in C++ (also called the compiled hierarchy), and a similar class hierarchy within the Tcl interpreter (also called the interpreted hierarchy).

The two hierarchies are closely related to each other; from the user’s perspective, there is a one-to-one correspondence between a class in the interpreted hierarchy and one in the compiled hierarchy.

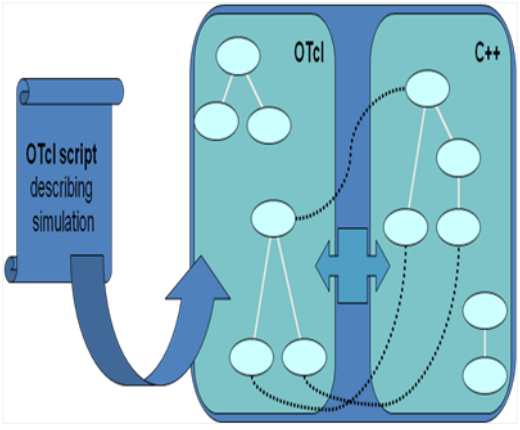


Figure: NS2 internal schematic diagram

NS2 uses two languages because it has two different kinds of things it needs to do: Detailed simulations of protocols require a systems programming language which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets. For these tasks run-time is important and turn-around time (run simulation, find a bug, fix the bug, recompile, re-run) is less important. C++ is fast to run but slower to change, making it suitable for detailed protocol implementation.

A large part of network research involves slightly varying parameters or configurations or quickly exploring a number of scenarios. In these cases, iteration time (change the model and re-run) is more important. Since configuration runs once (at the beginning of the simulation), the run-time of this part of the task is less important. Tcl runs slower than C++ but can be changed very quickly (and interactively), making it ideal for simulation configuration.

Users create new simulator objects through the Tcl interpreter. These objects are instantiated within the interpreter and are closely mirrored by a corresponding object in the compiled hierarchy.

Class TclObject is the base class for most of the other classes in the interpreted and compiled hierarchies. Every object in the class TclObject is created by the user from within the interpreter. An equivalent shadow object is created in the compiled hierarchy. The two objects are closely associated with each other.

The interpreted class hierarchy is automatically established through methods defined in the class TclClass. User instantiated objects are mirrored through methods defined in the class TclObject.

Tcl / C++ variable binding:

Class InstVar defines the methods and mechanisms to bind a C++ member variable in the compiled shadow object to a specified Tcl instance variable in the equivalent interpreted object. The binding is set up such that the value of the variable can be set or accessed either from within the interpreter or from within the compiled code at all times.

Whenever the variable is read through the interpreter, the trap routine is invoked just prior to the occurrence of the read. The routine invokes the appropriate get function that returns the current value of the variable. This value is then used to set the value of the interpreted variable that is then read by the interpreter. Likewise, whenever the variable is set through the interpreter, the trap routine is invoked just after to the write is completed.

The routine gets the current value set by the interpreter and invokes the appropriate set function that sets the value of the compiled member to the current value set within the interpreter.

The basic primitive for creating a node is:

set ns [new Simulator] $ns node

The instance procedure node constructs a node out of simpler classifier objects (to be discussed later). The Node itself is a standalone class in Tcl. However, most of the components of the node are themselves TclObjects.

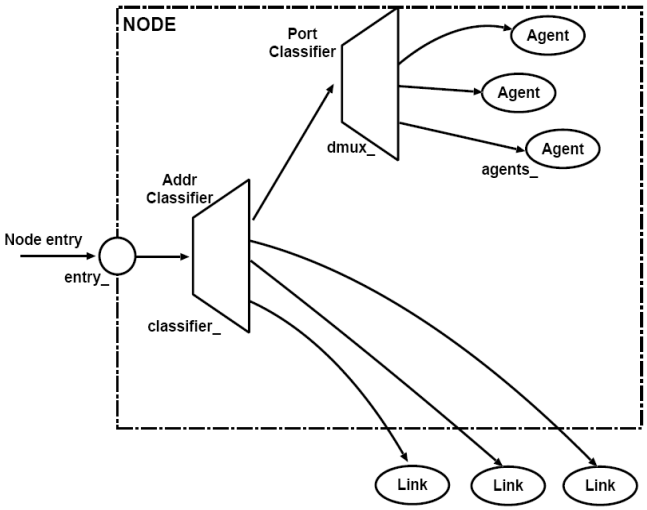


Figure: Node structure

This simple structure consists of two TclObjects: an address classifier (classifier_) and a port classifier (dmux_). The function of these classifiers is to distribute incoming packets to the correct agent or to correct outgoing link.

10. TRACE AND MONITORING SUPPORT

There are a number of ways of collecting output or trace data on a simulation. Generally, trace data is either displayed directly during execution of the simulation, or (more commonly) stored in a file to be post-processed and analyzed. There are two primary but distinct types of monitoring capabilities currently supported by the simulator. The first, called *traces*, record each individual packet as it arrives, departs, or is dropped at a link or queue. Trace objects are configured into a simulation as nodes in the network topology, usually with a Tcl "Channel" object hooked to them, representing the destination of collected data (typically a trace file in the current directory). The other types of objects, called *monitors*, record counts of various interesting quantities such as packet and byte arrivals, departures, etc.

11. SIMULATOR

The simulator is an event-driven simulator. The scheduler runs by selecting the next earliest event, executing it to completion, and returning to execute the next event. Unit of time used by the scheduler is seconds. Presently, the simulator is single-threaded and only one event in execution at any given time.

If more than one event is scheduled to execute at the same time, their execution is performed on the FIFO manner (first scheduled – first dispatched). No partial execution of events or pre-emption is supported.

An event generally comprises an event time, event id and a handler function. Two types of objects are derived from the base class Event - packets events and "at-events". Packets events will be discussed later in detail.

An "at-event" is a Tcl procedure execution scheduled to occur at a particular time. This is frequently used in simulation scripts. A simple example of how it is used is as follows:
set ns [new Simulator]
$ns use-scheduler Heap
$ns at 300.5 "finish"

This Tcl code first creates a simulation object, then changes the default scheduler implementation to be heap-based, and finally schedules the function "finish" to be executed at time 300.5 (in seconds).

In communication and computer network research, network simulation is a technique where a program models the behavior of a network either by calculating the interaction between the different network entities (hosts/routers, data links, packets, etc.) using mathematical formulas, or actually capturing and playing back observations from a production network. The behavior of the network and the various applications and services it supports can then be observed in a test lab; various attributes of the environment can also be modified in a controlled manner to assess how the network would behave under different conditions. When a simulation program is used in conjunction with live applications and services in order to observe end-to-end performance to the user desktop, this technique is also referred to as network emulation.

12. NETWORK SIMULATOR

A network simulator is a software program that imitates the working of a computer network. In simulators, the computer network is typically modeled with devices, traffic etc. and the performance is analyzed. Typically, users can then customize the simulator to fulfill their specific analysis needs. Simulators typically come with support for the most popular protocols in use today, such as WLAN, Wi-Max, UDP, and TCP.
Why Network simulation?

- Protocol validation
- controlled experimental conditions
- Low cost in $, time, collaboration, complexity

Why NS?

Provides:

- Protocols: TCP, UDP, HTTP, etc.
- Traffic Models: Web Traffic, CBR,
- Topology Generation tools
- Visualization tools
- Large validation package (people believe it works)

NS Structure

- C++ event scheduler protocols (most)
- TCL scripts protocols (mostly extensions to C++ core)
- TCL objects expose an interface to C++ objects (shadow objects) system configuration (defaults, etc.)

Advantage

- Flexible and state of the art tool
- contains wide classes of internet protocols including

Multicasting, SRM, RTP, ATM and wireless networks

- Widely used => respectful results + easy to compare

Disadvantages

- "alpha" quality
- Minimal docs
- Incomplete API

Simulations

Most of the commercial simulators are GUI driven, while some network simulators require input scripts or commands (network parameters). The network parameters describe the state of the network (node placement, existing links) and the events (data transmissions, link failures, etc.,). Important outputs of simulations are the trace files.

Trace files can document every event that occurred in the simulation and is used for analysis. Certain simulators have added functionality of capturing this type of data directly from a functioning production environment, at various times of the day, week, or month, in order to reflect the average, worst-case, and best-case conditions.

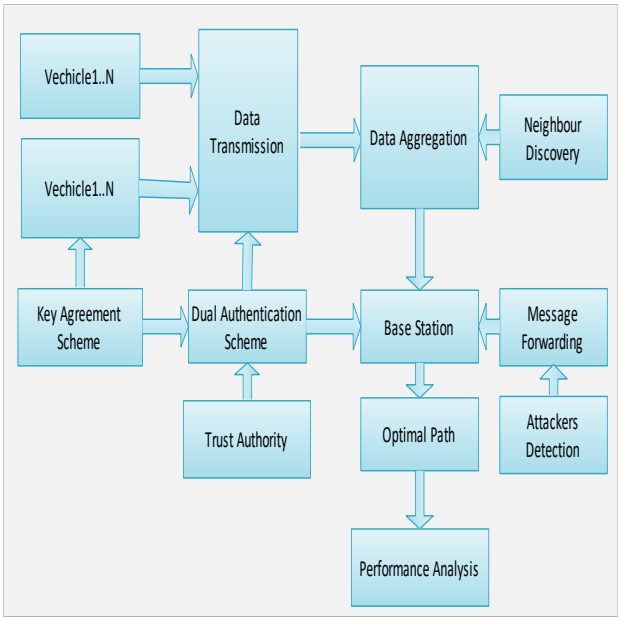
Network simulators can also provide other tools to facilitate visual analysis of trends and potential trouble spots.

Most network simulators use discrete event simulation, in which a list of pending "events" is stored, and those events are processed in order, with some events triggering future events -- such as the event of the arrival of a packet at one node triggering the event of the arrival of that packet at a downstream node.

Some network simulation problems, notably those relying on queuing theory, are well suited to Markov chain simulations, in which no list of future events is maintained and the simulation consists of transiting between a different system "states" in a memory less fashion. Markov chain simulation is typically faster but less accurate and flexible than detailed discrete event simulation. Some simulations are cyclic based simulations and these are faster as compared to event-based simulations.

Simulation of networks can be a difficult task. For example, if congestion is high, then estimation of the average occupancy is challenging because of high variance. To estimate the likelihood of a buffer overflow in a network, the time required for an accurate answer can be extremely large. Specialized techniques such as "control variates " and "importance sampling" have been developed to speed simulation.

13. ARCHITECTURE DIAGRAM



14. CONCLUSION

We address the security problem by focusing on the scenario where the TA classifies the users into primary, secondary, and unauthorized users. In this paper, first, we present a dual authentication scheme to provide a high level of security on the vehicle side to effectively prevent the unauthorized vehicles entering into the VANET. Second, we proceed by identifying the frequent individual data in the Server and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the server. Frequent data Identification and dual group key management scheme to efficiently distribute a group key to a group of users and to update such group keys during the users' join and leave operations. The major advantage of the proposed dual key management is that adding/revoking users in the VANET group can be performed in a computationally efficient manner by updating a small amount of information.

15. REFERENCES

[1] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of
the H.264/AVC video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, 2003.
[2] J. Vella and S. Zammit, "A survey of multicasting over wireless access networks," *Communications Surveys Tutorials, IEEE*, vol. 15, no. 2, pp.718–753, 2013.
[3] M. Asefi, J. W. Mark, and X. Shen, "A mobility-aware and quality drove
retransmission limit adaptation scheme for video streaming over VANETs," *Wireless Communications, IEEE Transactions on*, vol. 11,no. 5, pp. 1817–1827, 2012.
[4] M. Xing and L. Cai, "Adaptive video streaming with inter-vehicle relay
for highway VANET scenario," in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 5168–5172.
[5] O. Oyman and S. Singh, "Quality of experience for HTTP adaptive streaming services," *Communications Magazine, IEEE*, vol. 50, no. 4,pp. 20–27, April 2012.

[6] W.-H. Kuo, W. Liao, and T. Liu, "Adaptive resource allocation for layerencoded iptv multicasting in ieee 802.16 wimax wireless networks,"
Multimedia, IEEE Transactions on, vol. 13, no. 1, pp. 116–124, Feb 2011.
[7] F. Soldo, C. Casetti, C. Chiasserini, and P. Chaparro, "Video streaming
distribution in VANETs," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 7, pp. 1085–1091, 2011.
[8] C. Rezende, H. Ramos, R. Pazzi, A. Boukerche, A. Frery, and A. A. F.
Loureiro, "VIRTUS: A resilient location-aware video unicast scheme for vehicular networks," in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 698–702.
[9] C. Rezende, A. Mammeri, A. Boukerche, and A. A. F. Loureiro, "A receiver-based video dissemination solution for vehicular networks with content transmissions decoupled from relay node selection," *Ad Hoc Netw.*, vol. 17, pp. 1–17, Jun. 2014.
[10] R. Wang, C. Rezende, H. Ramos, R. Pazzi, A. Boukerche, and A. A. F.

