5-1-2010

Group-Based Authentication Mechanisms for Vehicular Ad-Hoc Networks

Marshall K. Riley
Southern Illinois University Carbondale, mkriley99@yahoo.com

GROUP-BASED AUTHENTICATION MECHANISMS FOR VEHICULAR AD-HOC NETWORKS

by

Marshall K Riley

B.S., Southern Illinois University, 2008

A Thesis
Submitted in Partial Fulfillment of the Requirements for the
Master of Science Degree

Department of Computer Science
in the Graduate School
Southern Illinois University Carbondale
May 2010

THESIS APPROVAL

GROUP-BASED AUTHENTICATION MECHANISMS

FOR VEHICULAR AD-HOC NETWORKS

By

Marshall K Riley

A Thesis Submitted in Partial

Fulfillment of the Requirements

for the Degree of

Master of Science

in the field of Computer Science

Approved by:

Kemal Akkaya, Chair

Norman F. Carver III

Henry Hexmoor

Graduate School
Southern Illinois University Carbondale
07 April 2010

AN ABSTRACT OF THE THESIS OF

MARSHALL K RILEY, for the Master of Science degree in COMPUTER SCIENCE, presented on 07 APRIL 2010, at Southern Illinois University Carbondale.

TITLE: GROUP-BASED AUTHENTICATION MECHANISMS FOR VEHICULAR AD-HOC NETWORKS

MAJOR PROFESSOR: Dr. Kemal Akkaya

Vehicular ad hoc networks (VANETs) provide opportunities to exchange traffic information among vehicles allowing drivers to not only adjust their routes but also prevent possible collisions. Due to the criticality of exchanged information, message authentication which will not expose the privacy of vehicles is required. The majority of current authentication schemes for VANETs depend primarily on public-key cryptography which brings extra overhead in terms of delay and requires infrastructure support for certificate verification. Symmetric-key based techniques can be more efficient, but they introduce significant key maintenance overheads. Herein, by considering the natural group behavior of vehicle communications, we propose an efficient and lightweight symmetric-key based authentication scheme for VANETs based on group communication. Expanding the protocol's flexibility, we also propose an extension which integrates certain benefits of asymmetric-key techniques. We analyze the security properties of our proposed schemes to show there applicability when there is little to no infrastructure support. In addition, the proposed protocol was implemented and tested with real-world vehicle data. Simulation results confirmed the efficiency in terms of delay with respect to other proposed techniques.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Vehicular ad-hoc networks (VANETs) have started to receive increasing interest recently due to their potential to be used in traffic and safety applications in the upcoming years [6] [7]. In such an ad hoc network, vehicles equipped with a wireless transmission device can send and receive messages at significantly higher speeds compared to traditional mobile ad-hoc networks [8]. Vehicles exchange traffic information as they move through the network, which allows drivers to adjust their routes to avoid congestion, obtain road-condition warnings, and be warned in advance for potential traffic accidents.

While the majority of recent research focused on medium access control and routing protocols with the goal of handling the dynamic behavior of VANETs [7] [9], an important aspect that needs to be considered is the security in transmitting messages. Security of VANETs is critical in preventing collisions and thus minimizing the risk for major accidents. For instance, all safety-related messages sent by a vehicle must be verified by the recipient for its authenticity and integrity in face of adversaries that may inject messages containing bogus information to the network. Yet the privacy of the driver sending those safety-related messages against unauthorized observers must be guaranteed. However, this anonymity service should be made conditional, meaning it can be revoked for law enforcement purposes whenever necessary. Besides those aforementioned requirements, a secure VANET system should also support availability against various common attacks such as denial-of-service (DoS) attacks and replay attacks.

Most of the prior works on VANET security make exclusive use of public-key cryptography (PKC), requiring that every message be digitally signed and attached with public-key certificates. This incurs significant overhead in terms of both computational cost and bandwidth. In addition, signature verification is a much slower process than signature generation in proposed digital signature schemes, such as ECDSA [10], which makes it even more vulnerable to DoS attacks. Furthermore, the use of PKC may require infrastructure support (e.g. certificate revocation list distribution) which may not always be available everywhere even though VANET deployments are planned in the near future. Lack in full infrastructure deployment could be one of several reasons, such as insufficient funding or being offline due to a malfunction. The concerns mentioned above suggest that symmetric-key cryptography, which is overall much more efficient than public-key techniques, should be used so that VANET's real-time requirements can be met. However, it is infeasible to have every two vehicles share a secret session key due to the huge scale of VANETs. To address this problem, one solution is to use group communication which is motivated by the fact that in a VANET vehicles typically move in groups. The use of groups, together with symmetric-key cryptography, can solve the key distribution problem and improve the efficiency of secure VANETs.

There are many VANET applications that require the formation of groups, particularly for vehicles in geographical proximity. The most prominent example is *platooning* [11], which groups vehicles in a way that allows them to accelerate or brake simultaneously, thereby avoiding collisions as well as increasing road capacity. VANET groups may also find numerous applications in infotainment in the future, such as multi-player games and chat rooms for vehicle passengers. There are a number of ways to construct groups in VANET applications. The most useful category of groups in terms of functionality is a *geodynamic group*, where a group leader is elected dynamically, group membership is changed dynamically, and the group boundary also moves dynamically along the road with the vehicles in the group. Geodynamic groups are naturally the best choice for platooning-like applications.

Nonetheless, the design of secure VANETs is further complicated if groups are allowed to form, because security measures must be implemented to ensure only legitimate vehicles can join a group. In particular, the overhead associated with the formation and management of geodynamic groups poses a significant challenge in designing efficient security schemes (i.e. secure group communication). Herein, we propose a lightweight geodynamic group-based authentication protocol for VANETs which can efficiently create geodynamic groups and provide secure communication among the members of the groups via symmetric-key cryptography. We strive to design a geodynamic group protocol that makes a reasonably good balance among all the different security requirements as well as efficiency and that is readily deployable, yet it can be easily extended to take full advantage of PKC while retaining its efficiency when widespread infrastructure support is available for VANETs. Our protocol is lightweight in the sense that it utilizes symmetric-key cryptography by maintaining a group key for each VANET group. Group keys are created and distributed by group leaders providing efficiency in key distribution and maintenance. Groups are geodynamicly formed within the transmission range of the leader and managed via a small set of control messages.

This thesis is organized as follows:

Chapter 2 gives a general overview on what exactly a VANET comprises of. It further discusses and highlights security issues which should be kept in mind when dealing with authentication within VANETs. The current communication standard for VANET communication is also highlighted.

Chapter 3 lists several previous VANET authentication schemes which were reviewed before the design of our protocol. We briefly discuss and analyze each scheme's benefits and drawbacks.

Chapter 4 details our initial work in designing an efficient and lightweight protocol. Additionally, it outlines the base structure on which we further developed our protocol.

Chapter 5 provides the modifications made to our original protocol to improve flexibility, usability, and efficiency. It includes the PKC extension made to our protocol.

Chapter 6 gives a security analysis of both versions of our protocol. The system is looked at both from the threat model detailed in 2.3.1 and the perspective of security goals listed in 2.3.2.

Chapter 7 is dedicated to experimental evaluation.

Finally Chapter 8 concludes our findings with possible future extensions to this thesis.

CHAPTER 2
VEHICULAR AD-HOC NETWORKS (VANETS)

2.1 OVERVIEW OF VANETS

VANETs allow vehicles equipped with communication technology to perform efficient inter-vehicular communications (IVC) and road-vehicle communications (RVC) thereby enabling the Intelligent Transportation System (ITS) without the need for permanent infrastructure. Therefore, VANETs are also called inter-vehicle communication (IVC) or vehicle-to-vehicle (V2V) communications [12]. Employing these networks will allow information about both surrounding traffic and road conditions to be relayed to the driver of the vehicle, thus allowing them to have an increased awareness of their surroundings. Sensing applications could be incorporated to allow constant and real-time monitoring of the environment, surrounding roadways, or even monitoring of road conditions themselves. The recorded data could then be transmitted to the areas respective transportation authority for timely remediation if needed. A sample application depicting an accident and possible information exchange between various vehicles and roadside units (RSUs) is shown in Figure 1. For the rest of this, the terms ”vehicle” and ”node” will be used interchangeably.

The similarities of VANETs with other ad hoc networks such as Mobile Ad Hoc Networks (MANETs) are in their short radio transmission range, self-organization, self-management, and low bandwidth. However, VANETs can be distinguished from other kinds of ad hoc networks as follows [9]: (1) highly dynamic topology due to high speed movement between nodes; (2) frequently disconnected network caused by high speed movement; (3) sufficient energy and storage provided by the vehicle; (4) geographical type of communication; (5) mobility modeling and prediction; (6) hard delay constraints such as in collision avoidance situation (e.g.

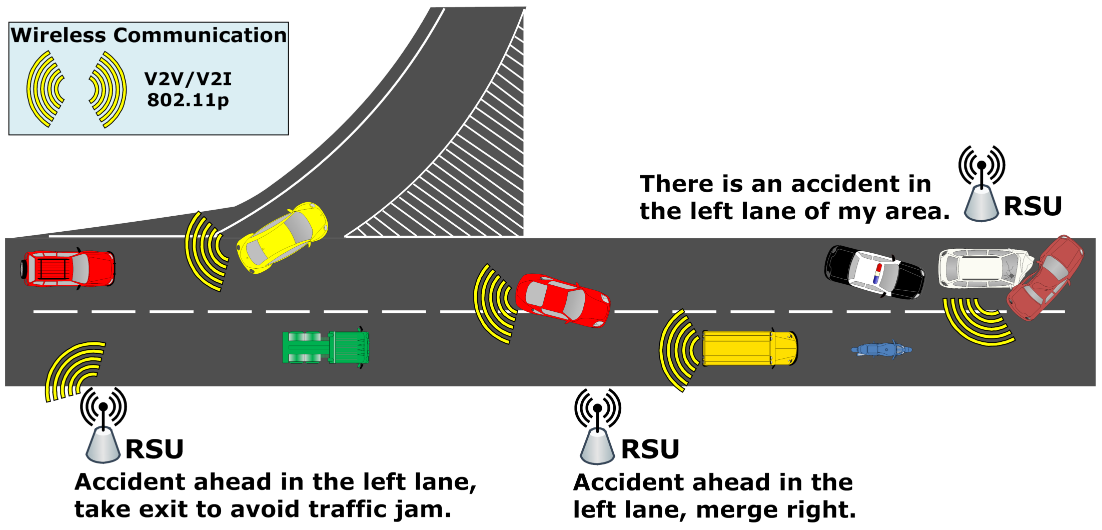

Wireless Communication
V2V/V2I
802.11p
There is an accident in the left lane of my area.
RSU
RSU
Accident ahead in the left lane, take exit to avoid traffic jam.
RSU
Accident ahead in the left lane, merge right.

Figure 1. Example of a safety application within a VANET

the maximum delay of break event information will be very crucial); (7) and interaction with on-board sensors.

Due to possibility of accidents, the exchange of information among vehicles is critical. This has put a lot of emphasis to security issues within VANETs. Security has started to receive a great deal of attention recently in addition to other issues regarding routing, medium access, connectivity, etc [13]. As a VANET can be thought of as a specialized form of a MANET, several security challenges which are applicable to MANETs, according to [14], apply to VANETs. For example, a VANET, like a MANET, suffers from being a wireless environment, dynamic topology, physical security of the system (i.e. tamper resistance), and system failures to name a few. VANETs go above and beyond MANET security challenges though, as nodes have identities which need to be protected but at the same time need to be identifiable should one misbehave. The increased mobility of nodes and the sheer size of the possible network are also unique security challenges. Additionally, due to the initial intentions of a VANET being utilized to relay safety information between users driving, if the data is maliciously altered this could potentially cause fatalities. A detailed list of possible attacks, challenges, and discussion of proposed solutions can be found in [13]. To address the security issues in VANETs, the IEEE 1609 trial-use standard for Wireless Access in Vehicular Environments (WAVE) [6] has been developed. The IEEE 1609.2 trial-use standard specifies security services for applications and communication management messages which are based on industry standards for public-key cryptography [6, 14] as will be explained in detail in Section 2.2.

2.2 IEEE 1609.2 FOR WAVE

The set of IEEE 1609 standards have been developed to enhance IEEE 802.11 standards for supporting wireless communication both between vehicles (V2V) and between vehicles and the roadside infrastructure (V2I). These are also commonly known as dedicated short range communications (DSRC) schemes. In particular IEEE 1609.2 addresses security within WAVE communications. The standard dictates that confidentiality, authenticity, and integrity must be provided within WAVE communications.

Though it identifies the usage of symmetric (secret-key), asymmetric (public-key), and hash functions as being able to provide these requirements, as far as authentication is concerned, IEEE 1609.2 identifies the utilization of public-key infrastructure (PKI) for establishing authenticity. As such, for authentication purposes, IEEE 1609.2 is based on public-key cryptographic standards, such as elliptic curve cryptography (ECC), as well as public standards for other PKI administration functions thereof, such as certificate revocation. Broadcast messages (e.g. safety vehicle warnings, vehicle safety messages) are defined as only being signed and, in general, not encrypted. As such, asymmetric techniques are again defined to be ideal. However, other transactional messages are left open to be protected by either symmetric or asymmetric means. The standard utilizes the

SHA-1 hash function for creating identifiers for certificates and fragmented messages.

IEEE 1609.2 also defines an additional requirement of anonymity. That is the broadcast transmissions by a private individual should not reveal information which can be utilized in identifying them to unauthorized recipients. It is noted that vehicles, such as public safety vehicles, do not, in general, have an anonymity requirement. Though this extra requirement is identified, IEEE 1609.2, as dictated in [6], does not provide or define a mechanism for providing anonymity.

2.3 AUTHENTICATION IN VANETS

Authentication is a must feature in a VANET as the source of the information should be verified to ensure the legitimacy of the data communicated. We first identify a general attack model which is commonly utilized in VANET security analysis. Additionally, we provide several attack vectors based off of this threat model. Furthermore, we provide a discussion on the security goals and assumptions in designing an authentication scheme. The goals and assumptions presented reflect the majority of present work within VANET authentication research.

2.3.1 Attack Model

Although all possible attacks cannot be identified, several attack vectors that authentication schemes should attempt to protect against have been identified in [13]. Several authentication schemes reviewed herein utilized the following threat model detailed in [13] and summarized here. We utilized this threat model to guide us in decisions on what and how to protect while designing our protocol. The general threat model is defined as follows:

Insider vs Outsider: An insider is an authenticated node within the network and as such contains, at least, one valid key for usage with communicating with other nodes in the VANET. An outsider is a node which is not authenticated and as such is seen as a non-group member or intruder should they attempt to communicate within the group.

Malicious vs Rational: A malicious attacker seeks to decrease network functionality or attack members of the network without the intent of personal gain. A rational attacker seeks personal gain from their attacks.

Active vs Passive: Active attackers are capable of injecting messages into the network. Passive attackers only eavesdrop on communication going across the network.

Local vs Extended: Local attackers control nodes or RSUs which are relatively close in location as a whole, thus limiting their view to a specific, localized area. Extended attackers control nodes or RSUs which are scattered throughout the network, thus giving them a broader, more generalized view of the network.

Following in giving a generalized threat model, we also summarize several attack vectors. Most attacks defined in the reviewed papers directly stem from one or a combination of these attack vectors. As with the threat model, we utilized these attack vectors in guiding our design process for our protocol.

Erroneous information propagation: Attackers attempt to propagate erroneous information throughout the network in an attempt to affect the actions taken by other nodes.

Cheating the sensors: Attackers alter the readings of the sensors within the node (primarily their own sensors).

Identity disclosure: The attacker is able to reveal the identity of the node and track their movements through network communications.

Denial of Service: The attacker jams the channel or floods bogus messages in an attempt to overload the computation capabilities of nodes.

***Masquerading**:* Attackers attempt to assume the identity of another node or RSU.

2.3.2 Security Goals and Assumptions

Goals

In addition to message authentication (which includes integrity), several of the reviewed schemes also identified other desirable goals sought within a VANET network based on their authentication model. For instance, providing authentication may lead to exposing the location, driving pattern, and IDs of vehicles and drivers. Obviously, this is an important privacy concern for the drivers as exposing this information allows the driver to be tracked based solely on their message communication within the network, unbeknownst to them. In addition to privacy, another crucial goal is to meet the real-time constraints which are specific to VANETs. Any delay overhead introduced by an authentication process is not desirable as it may lead to fatalities. Though, as adding in various extra checks (e.g. authentication) will undoubtedly add additional delay, no matter how small, the impact to delay of any protocol should be kept down to a minimum. A non-exhausted list of security goals is given below:

***Privacy (Anonymity)**:* Individual vehicles/drivers should be protected against unauthorized, identifying observations.

***Real-time constraints**:* Due to the high speed nature of a VANET, timely communications and strict time constraints should be enforced.

***Availability**:* The solution should not significantly increase or add in new attacks to deny communication ability (e.g., denial of service (DoS) from half-open connections).

Non-repudiation: When needed, the identity of the sending vehicle should be recoverable from message communications, as well as ensuring the sender can not deny transmitting the message.
Infrastructure Independency: Due to possible unavailability of RSUs, a proposed solution should not require frequent access to an infrastructure when performing authentication.

Tamper Resistance

IEEE 1609.2 dictates that whenever practical, keying material should be protected from exposure and embedded in a tamper-resistant Hardware Security Module (HSM), also known as a tamper-proof device (TPD), on which many VANET authentication schemes also rely. A TPD is a physical device dedicated to secure storage of cryptographic keys and sensitive data as well as accelerating and securing cryptographic operations, with multiple layers of physical security measures that provide a high degree of tamper resistance. The primary goal of these devices are to make it difficult for an individual to access the material or data inside. It is accessible only by authorized personnel and it zeroizes its memory in the event of probing or scanning. We say difficult as most any device, given a sufficiently motivated/funded individual, could be eventually compromised. A TPD:

- Houses the keying material
- Has a secure cryptoprocessor
- Has its own battery
- Has its own clock (for timestamps)
- Has its own pseudorandom number generator for cryptographic operations

- Is accessible only by authorized personnel

- Zeroizes its memory in the event of probing or scanning

Without the assumption of at least a TPD, most VANET authentication schemes will be vulnerable to several attacks not previously noted as the keying material then becomes accessible by anyone. The primary drawback to a TPD, however, is that they can cost on the upwards of several thousand of dollars as of this writing. An alternative is the use of tamper resistance devices which can provide security to a certain extent. These devices are cheaper compared to TPDs but the probability of being compromised is not 0%. Thus, the trade off in protocols for this becomes one of monetary cost versus data/user security.

CHAPTER 3
LITERATURE REVIEW

Despite the IEEE 1609.2 standard, there has been a significant number of newly proposed authentication schemes over the past few years. This is primarily due to lack of scalability, speed of the cryptographic scheme, and cryptographic overhead in IEEE 1609 [15]. Other problems include a high degree of dependence upon network infrastructure and lack of ability to provide other desirable security features.

The most common objectives of recent authentication schemes were to lower communication overhead, preserve node anonymity, isolation of misbehaving nodes, and non-repudiation. These, however, were not the only objectives stated in the following works. One key note, is that most of the works aimed at providing various security properties which would be appropriate for one particular application or privacy level, but is more than required for another. This is where the main differences lie between the proposed schemes for authentication.

3.1 CATEGORIZATION OF AUTHENTICATION SCHEMES

Herein, we classify the existing authentication schemes as follows (see Figure 2):

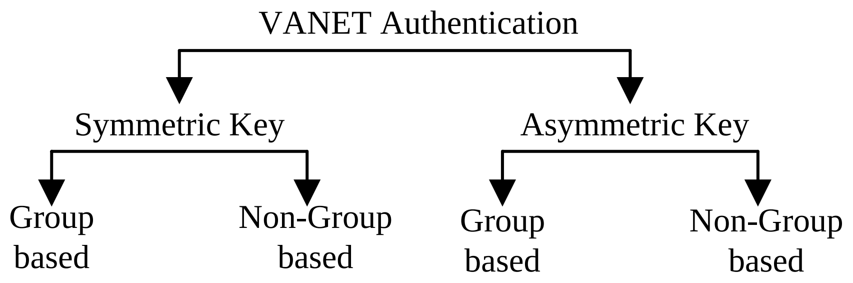


Figure 2. Categorization of Authentication Schemes

1. *Symmetric Authentication Mechanisms:* These schemes use symmetric-key

cryptography for authenticating messages among the vehicles. They are further divided into two sub-categories:

   (a) *Group based Mechanisms:* A shared group key is used in order to exploit group communication.

   (b) *Non-group based Mechanisms:* Each node uses its own key for creating/verifying message authentication codes (MAC).

2. *Asymmetric Authentication Mechanisms:* These schemes use public-key cryptography for authentication. Similarly, they can be further divided into two sub-categories.

   (a) *Group based Mechanisms:* A group public key, which corresponds to multiple private keys (each of which is held by a single group member), is used in order to exploit group communication. A group signature scheme [16] based on such a group public key and corresponding set of private keys allows the group manager to determine which private key signs a particular message.

   (b) *Non-group based Mechanisms:* Each node uses its own public/private keys for data communication and/or for creating/verifying digital signatures.

For each of the schemes in the mentioned categories, we also provide the level of infrastructure requirement, such as the frequency of the need for accessing the network to verify certificates, revoke keys, gather new keys, etc. The proposed authentication schemes in the literature are presented in a manner in which primarily symmetric key schemes, with increasing levels of infrastructure requirement, are presented first and then asymmetric key schemes are presented, again with increasing levels of infrastructure requirement.

3.2 SYMMETRIC AUTHENTICATION SCHEMES

We start by discussing authentication schemes which primarily rely on symmetric-key cryptography to perform authentication. No symmetric-key cryptographic scheme reviewed could work completely without infrastructure, such as RSUs. As such, the section is organized by presenting the schemes which require little infrastructure support first and progress to the more infrastructure demanding schemes.

3.2.1 Group-based Schemes

PPGCV

The scheme proposed in [17] is the only group-based scheme under symmetric-key cryptography so far which proposes the privacy preserving group communication scheme for VANETs (PPGCV) to satisfy forward and backward secrecy, authentication, protection against collusion, and privacy.
Additionally, [17] provides what is termed conditional full statelessness property. That is, a node can calculate the new group key as well as update its compromised key list (i.e., keys which are utilized by misbehaving nodes) even if the node misses the group rekeying process. PPGCV has two phases:
Key Bootstrapping Phase: In this phase, each node v in the network randomly selects a set of keys R_v, from a key server which has a key pool, for use as Key Encrypting Keys (KEKs). Each node is additionally loaded with an initial group key k_g for group communication.
The Group Rekeying Phase: This occurs whenever the key server needs to revoke a node u's membership. The server searches its database for the ID, M, of a non-compromised key k_M which is shared by the majority of the non-compromised nodes. It generates an intermediate key $k_{im} = f_{k_M}(k_g)$ and a new group key $k_g^{'} = f_{k_{im}}(0)$, where f_k is a family of pseudorandom functions. It sends a node

revocation message which contains M, the revoked node's ID, and a checksum of the new group key $k_g^{'}$. Any node which has k_M can immediately generate k_{im} by itself and from that $k_g^{'}$, which is verified using the received checksum. The nodes without k_M, on the other hand, will randomly select a subset of keys from its key set and broadcast the key IDs. Any neighboring node with one of those key IDs, and the key k_M, will encrypt k_{im} with the common key and send it to the node. After determining the new group key, all nodes v will update every key k_i of their key sets R_v as follows: $k_i^{'} = f_{k_i}(0) \;\; \forall k_i \in R_v \backslash R_u$ and $k_i^{'} = f_{k_{im}}(k_i) \;\; \forall k_i \in R_u$, where u is the revoked node.

As the nodes delete the intermediate key and their original keys, they save a set of values known as the shadow. This shadow can be used in a (t, n) threshold scheme, where n is the total number of participants and t is the number of participants that can collude to reveal the shared secret, to regenerate the intermediate keys. This is known as intermediate key regeneration for a vehicle which missed the rekeying process.

Authentication occurs in two different aspects of this scheme. It is first achieved in that only a valid group member can encrypt communications with the proper key for other group members, as the group key changes when members are revoked. Secondly, during the intermediate key regeneration step the requesting node sends a subset of key IDs of its own key set, and any neighboring node receiving such a request will perform a check on these key IDs to make sure that they have not been revoked.

From this approach, revocation list distribution does not need to occur for revoking node group membership. Additionally, should some node miss the rekeying phase, it will still be able to get the new group key by querying other valid group members for the shadow. On the other hand, each participant must securely store a set of keys for utilization within the scheme. This approach makes the scheme only minimally reliant on infrastructure support. Additionally this scheme assumes that every node has knowledge of the revoked node's key set in advance and retains the information[1]. This increases the storage requirement of each node as well as the bandwidth consumed by communications between each node and the key server.

The experiments in the paper do not provide any results regarding the speed/efficiency of the proposed scheme, when dealing with a large volume of vehicles, for key regeneration, storage, and searching. Thus, the scalability of this authentication mechanism is questionable.

3.2.2 Non-group-based Schemes

VAST

Authentication mechanisms are identified as needing to have non-repudiation, Denial of Service (DoS) resilience, and support for multi-hop communication in [2]. The proposed scheme, namely VAST, addresses many of these issues which are identified as lacking in IEEE 1609.2. Additionally, [2] allows for tunable parameters to achieve various different properties.

VAST utilizes a combination of elliptic curve digital signature algorithm (ECDSA) signatures and TESLA++, which is a modified version of TESLA [18]. TESLA [18] is a symmetric-key based broadcast authentication scheme which relies on time to create the asymmetric knowledge between the sender and the recipient. In TESLA, the sender precomputes a long hash chain of keys $\{K_i : 1 \leq i \leq n\}$, whose root is given in a certificate signed by an authority. Each key is used for only a short period of time to generate the MAC of messages. To authenticate a message, a sender computes its MAC using the key K_i of the given

[1]This assumption, though not explicitly stated in [17], is made as otherwise it is trivial for a node to circumvent the security measures of the protocol and avoid having group membership revoked.

time interval, and sends the message-MAC pair to the receiver for storage. At the end of the given time interval, the sender reveals K_i so that the receiver can verify the stored MAC. Studer et al. [2] observed that TESLA is subject to memory-based DoS attacks against the recipient due to the storage of previously received messages. Therefore, they proposed TESLA++, a modified version of TESLA, which requires the recipient to store a self-generated MAC of the message's MAC received. The message itself as well as the key will not be revealed until the key expires. The sender first sends the message's MAC, MAC_S, along with the key index i and when the key expires it sends the corresponding message M, key K_i, and i. The receiver first computes the MAC, MAC_R, of the message's MAC, MAC_S, received using a local secret key K_{Recv}. This value is stored and later used for comparison against the value $MAC'_R = MAC_{K_{Recv}}(MAC_{K_i}(M))$ when K_i and M are revealed. The message is accepted if MAC'_R is identical to MAC_R. A comparison of TESLA and TESLA++ is shown in Figure 3.

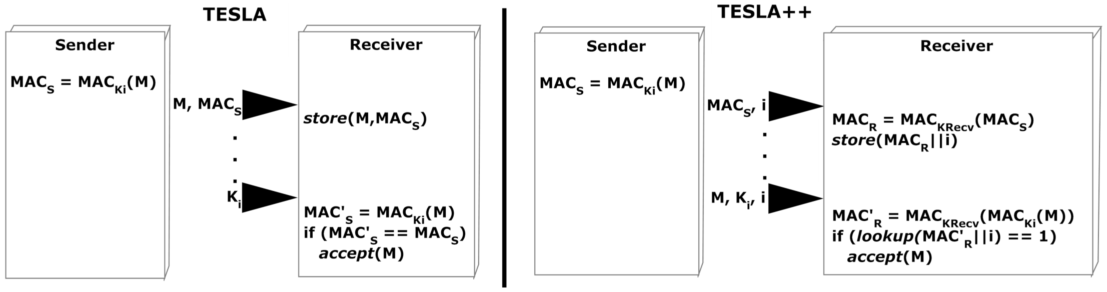


Figure 3. Comparison of TESLA to TESLA++, redrawn from [2]

While TESLA++ provided DoS resilience (i.e. it has relatively low memory storage requirements as only a MAC is needed to be stored for each message not yet received), it lacks non-repudiation and multi-hop authentication. For such purposes, ECDSA signatures are needed. These signatures are, however, only performed when either the application calls for non-repudiation or TESLA++ authentication fails and both CPU utilization and the message queue are below a predetermined threshold.

VAST has a minimal reliance on infrastructure support as it would only primarily be required for the case when ECDSA signatures are utilized and the public key would need to be verified. The minimal reliance on infrastructure makes it very well suited to be utilized in rural areas or during the early phases of VANET deployment when RSUs are not available everywhere. A significant drawback of this approach, however, is the minimum delay, about 100ms, required to authenticate messages, which is the delay set between heartbeat messages (i.e. the time between receiving the MAC and receiving the corresponding key and message).

An Efficient Message Authentication Scheme for Vehicular Communications

In creating a new authentication mechanism, the work in [15] identified the following security objectives to meet: 1) message integrity and source authentication, 2) low communication overhead and fast verification, 3) conditional privacy preservation, and 4) prevention of internal attacks. To address these issues two schemes were proposed, RSU-aided message authentication scheme (RAISE) and cooperative message authentication (COMET).

RAISE. In the RAISE scheme, when an RSU is detected a vehicle attempts to associate with it. The RSU assigns a unique shared symmetric secret key and a pseudo ID which can be released to other vehicles. To ensure anonymity this pseudo ID is associated with k vehicles. Utilizing the symmetric key and pseudo ID, the vehicle can generate a symmetric MAC code for any message that it sends to other vehicles (together with the RSU). Upon receiving a message the receiver must buffer the message until the RSU verifies the message's MAC and notifies its authenticity through the periodic broadcasts of an aggregate of the hashes of authenticated messages.

COMET. The COMET scheme was proposed, as a supplementary to RAISE, to address the issues of lack of deployment/coverage of RSUs, whether due to physical damage or economic constraints, and the message loss-ratio issue of RAISE. When a vehicle receives a message with a PKI-based signature, it has a probability p to verify the signature of the message. If the vehicle chooses to verify the signature, it will stay silent if the message is valid and will inform neighbors that the message is invalid otherwise. This invalid message will be broadcast to 1-hop neighbors. If the vehicle chooses not to verify the message, it will wait for Δt milliseconds for other neighbors' reports. If an invalid broadcast is received, the vehicle will ensure invalidity by verifying the message itself. If no reports are received within the time threshold, it treats the message as valid by default. These one-hop communications effectively reduce the message loss-ratio as the communications from an RSU to a vehicle are not needed.

The two schemes proposed, RAISE and COMET, together provide a low loss ratio under high vehicle density as well as a high degree of anonymity. Both properties are rather desirable in a VANET. RAISE by itself would only occur when infrastructure support, such as RSUs, are present, thus is rather heavily dependent on infrastructure. However, as [15] also proposed, in the supplementary scheme, COMET, the overall approach is not heavily dependent on infrastructure coverage due to other nodes handling the verification of messages instead of RSUs. In short, the overall approach can function with minimal infrastructure support. The scheme, however, does not provide a mechanism to revoke node keys. Additionally, in dealing with the COMET scheme as described in [15], there is a $1 - p$ probability that a vehicle will not self-verify a signature. Should an adversary jam the area around the node, or there are no other neighbors, there is a chance that the node would arbitrarily accept a possible invalid message as valid since no node broadcasts that the message is invalid.

3.3 ASYMMETRIC AUTHENTICATION SCHEMES

This section discusses the authentication schemes utilizing asymmetric-key cryptography. The section is further broken down into group based and non-group based schemes. Similar to Section 3.3, this section is organized by presenting the schemes which require little infrastructure support first and progresses to the more infrastructure demanding schemes.

3.3.1 Group-based Schemes

PPAA

The peer-to-peer anonymous authentication (PPAA) scheme, proposed in [19], views a VANET as a form of a P2P system. PPAA is a credential system which attempts to balance user privacy and accountability in terms of both client and server. [19] identified the four key requirements for security within their system as mis-authentication resistance, peer accountability, peer privacy, and framing resistance. PPAA was presented in two parts, the skeleton structure of the scheme and then the steps to secure the skeleton scheme.

Skeleton Scheme. In the skeleton scheme, each user is given a distinct credential by the group manager (GM). The GM is a centralized authority and is only needed for the setup and registration phases. The credential, *cred*, of a node is utilized with another node's credential and an event identifier, *eid*, as the input of a function to generate what is known as a *tag*. After a four way handshake, a *tag* is generated for both nodes and has the following properties. Both parties learn nothing besides the common *tag* from running the scheme and the tags are completely indistinguishable from another *tag* generated for a different peer or a different event. These properties rely on the intractability of the decisional Diffie-Hellman (DDH) problem in a cyclic group of prime order.

Securing the Skeleton Scheme. The PPAA skeleton is not completely secure. To do so, [19] utilizes a group signature scheme and as such the GM needs to compute a group public/private key. The credential generation process is altered in that the GM first sends a random challenge to a node. The node responds with a duple which contains a commitment value and a signature proof of knowledge (SPK) on the commitment concatenated with the random challenge. The GM validates this message and then, if successful, sends the node a tuple utilized for generating the node's *cred* value. One of the requirements of this proposed addition is that no two instances of this registration scheme can run concurrently. The skeleton scheme portion is also modified such that each step is accompanied by a SPK scheme.

Similar to [2], [19] is not heavily reliant on infrastructure as it is only required for the setup and registration phase. Additionally, a high level of anonymity is achieved. Since the scheme is modeled after P2P sessions in general, the proposed scheme requires a direct communication session to be established between every node with which a node wants to communicate though. As this is the case and no experimentation data is given, it is unknown precisely how efficient this type of system would be in a wireless environment with numerous vehicles and conversations occurring.

Efficient and Robust Pseudonymous Authentication in VANET

Multiple concepts are combined in [20] to produce what they termed as the *Hybrid* scheme. Pseudonymous authentication along with group signatures are utilized to achieve this scheme. The objective of this approach is to design a usable system based on pseudonyms (i.e. the pseudo IDs for vehicles) which provides message authentication, integrity, non-repudiation, anonymity, and difficulty in linking multiple messages to a single user.

In this approach, each node v registered with the CA has a secret group signing key gsk_v and a group public key gpk_{CA} that can be used to verify any signature generated by the group signing key of any registered node. Each node generates its own set of pseudonyms (realized as public keys) and corresponding private keys. It places each pseudonym in a certificate and utilizes gsk_v to generate a group signature on each pseudonym certificate. In other words, pseudonyms are generated and certified on-the-fly. To send a message, a node signs the message with the private key of one of its pseudonyms and attaches the message with that pseudonym's certificate. Upon receiving that message, the group signature on the pseudonym certificate is validated using gpk_{CA} and a revocation list distributed by the infrastructure, which can be used to identify any group signature generated by revoked nodes. Once the legitimacy of the pseudonym is established, the pseudonym is used to verify the message's own signature.

In addition to this scheme, [20] identified three optimizations to reduce overhead and increase robustness. The first optimization entails the sender only signing each pseudonym once, as its signature remains unchanged throughout the pseudonym's lifetime. Similarly a verifier only validates a pseudonym's signature upon the initial reception and will then store it for reference later. The second optimization is to only append the pseudonym's certificate once every α messages, where α is denoted as the certificate period. All messages sent would carry a four-byte keyID field indicating which pseudonym should be used to verify the message. This conversely decreases robustness as a new node may have to wait for α messages for the next pseudonym certificate transmission. To solve this, the third optimization was proposed. The pseudonym's certificate is repeated for p consecutive messages when the pseudonym is first issued.

The *Hybrid* scheme allows for a high degree of anonymity of the vehicles without reliance on other systems for on-the-fly pseudonym generation and certification. [20] managed to produce a relatively low communication overhead between nodes through the various optimizations noted. Though the general overhead is low, the system still does rely on revocation lists which cause increased overhead when the lists need to be distributed and increased storage requirements. Additionally, [20] created a custom simulator for testing their scheme and as such making it difficult to make general comparisons in performance with other schemes.

GSIS: A Secure and Privacy-Preserving Scheme for Vehicular Communications

Combining concepts discussed in both [21] and [22], [3] proposed Group Signature and Identity-based Signatures (GSIS) scheme which attempts to address the issue of security assurance and conditional privacy preservation in a VANET. In particular, [3] identified a well developed security scheme as having data origin authentication and integrity, anonymous user authentication, vehicle anonymity, RSU ID exposure, prevention of RSU replication, vehicle ID traceability, and efficiency. GSIS is divided into two categories: security and privacy preservation between vehicles and between RSUs and vehicles.

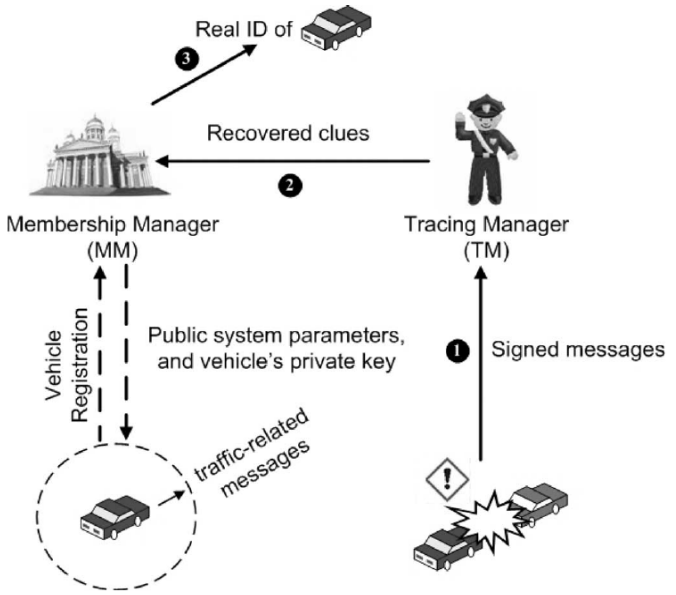


Figure 4. GSIS OBU secure communication overview [3]

In dealing with communication between OBUs of vehicles, [3] identified the traditional public key encryption scheme as not being suitable for signing safety messages as the ID information is included in the public key certificate. A group signature scheme was preferred for communication between OBUs primarily as it provides anonymity of the signer while also allowing a group manager to reveal the unique ID of the signature's originator. For RSU to OBU communication, the privacy requirement is not as strict. As such, an ID-based signature scheme, which uses the identity string of each RSU as the public key for signing messages from the RSU, can be used.

V2V Authentication. The security portion of the scheme for OBU to OBU communication contains five phases, as depicted in Figure 4. The first phase is membership registration in which a membership manager (MM) generates a tuple for each vehicle which serves as the vehicle's private key. The association information is stored by the MM for later use. The next phase consists of signing a message. For any given message, a vehicle signs the message as a function of both the group public key and the vehicle's private key. Verification of a received message is the third phase. A timestamp is first verified followed by, if it passes, signature verification using the group public key and certain system parameters. Should the message fail any of these verification steps, the message is simply dropped. Membership traceability, the fourth phase, occurs whenever there is a dispute and the real ID of the message signer is needed. A tracing manager (TM) first checks the validity of the signature and then generates a part of the tuple, generated during the first phase, for use by the MM in looking up the ID of the vehicle. The final phase is known as the membership revocation phase. [3] examined two different approaches, updating the group public key and private keys of all unrevoked vehicles or utilizing CRL-based verifier-local revocation (VLR), but found them both lacking in different aspects. Therefore, a hybrid approach which combined both approaches was utilized. While a predefined threshold in number of revoked vehicles is not met, the VLR scheme is used. After the threshold is reached, the group public and private keys are updated for all valid vehicles.

V2I Authentication. When dealing with the security portion between RSU and OBU communication, there are only three phases. The first phase deals with private key generation. A unique identifier string is generated for each RSU, by the transportation regulation center (TRC), consisting of a unique serial number, the physical location of the RSU, and the third field being associated with the attribute of the message. This identifier string, serving as the RSU's public key, is then utilized to generate the private key for the RSU and sent to it through a secure channel. The next phase is the signing phase, which occurs for every safety message sent by a RSU. The message is then sent together with a type ID, a timestamp, its signature, and the RSU's identifier string. The third and final phase is the verification phase, done by the receivers. First a check is performed to ensure the physical location of the RSU matches what is located in the RSU's identifier string. Next the message's type ID field is checked to ensure it matches the properties of the message stated in the RSU's identifier string (i.e. a road construction type message should have information about road construction). Finally the timestamp and signature on the message are verified. Should any of these verification steps fail, the message is dropped.

GSIS provides a node with a high degree of anonymity while still allowing for revocable privacy. It also does not require a fully extensive and robust infrastructure system, though some key features require infrastructure support. From simulation results, the message delay is relatively low under all conditions tested. Additionally from the simulation results, the message loss ratio drastically increases as the number of vehicles within communication range increases which is a drawback.

TACKing Together Efficient Authentication, Revocation, and Privacy in VANETs

In addition to authentication, Temporary Anonymous Certified Keys (TACKs) [22] aimed to provide privacy, short-term linkability, traceability and revocation, and efficiency, given the motivation that no other schemes provide all of these properties at the same time. TACK is based on a combination of standard techniques to meet said requirements. The design is heavily dependent on infrastructure as it divides the roadways into geographic regions with Regional Authorities (RAs) acting as the certification authorities for their respective regions and the federal transportation authority as the root of the key hierarchy.

TACK utilizes the group signature scheme described in [16]. Each member of a group has a group user key (i.e. a long-term private key) issued by a trusted group manager, such as the Department of Motor Vehicles (DMV). The group user key is utilized for producing a group signature in the process of obtaining a certificate for a short-lived TACK from an RA. The short-lived TACK is used for signing messages. The certificate is periodically broadcast by a vehicle for use in authenticating the signed messages it sends.

The TACK is updated whenever it expires as well as when the vehicle enters a new geographic region. To update the TACK, the vehicle's OBU generates its own new TACK public/private key pair randomly, utilizes its group user key to sign the TACK public key, and sends the signed public key to the RA for the region for certification. The RA will verify both the group signature on the received TACK public key using the group public key as well as the validity of the requester's OBU against the revocation lists. If both tests pass, the RA signs a certificate for the requester's TACK public key, stores the associated values locally for traceability, and sends the certificate back after τ seconds. The delay is added for decreasing linkability.

Like several other approaches, infrastructure dependency is again a major concern with this proposed scheme as there are several concepts/assumptions which rely on infrastructure being available for security. Though this is an issue, the scheme does limit communication overhead by having the RAs handle all revocation and thus the nodes primarily deal with only data communications. Another drawback, which was identified in [22], is the fact that a node which has been identified and revoked, is still able to communicate, possibly maliciously, with a valid certificate as the revocation only denies the node from getting a new certificate.

Probabilistic Adaptive Anonymous Authentication in Vehicular Networks

In dealing with anonymous authentication, the work in [4] addresses concerns for a higher need of anonymity than provided in a typical authentication scenario in VANETs, as depicted in Figure 5. [4] also regards scalability and timeliness as key concerns. RSUs are delegated authentication tasks, utilizing public-key cryptography, to prevent central authentication server overload. Additionally, [4] views two typical types of servers within vehicle infrastructure integration: 1) public servers controlled by government agencies, such as the Department of Transportation; and 2) private servers controlled by private service providers. When dealing with this server scheme, a user will most likely have varying degrees of trust, which are full-trust, partial-trust, and zero-trust. Zero-trust is the main focus of [4], meaning that no server stores any authentication information, such as user identities.

The group-based anonymous authentication scheme proposed in [4] is, essentially, an adaptive k-anonymity concept with k as a tunable parameter. That is, the level of anonymity can be dynamically adjusted by changing the value of k.

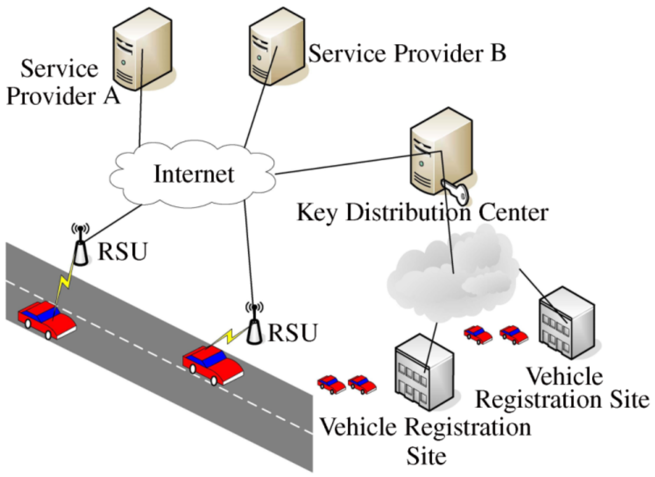


Figure 5. VANET generic authentication scenario [4]

It is achieved through the utilization of the cryptographic construct known as verifiable common secret encoding. This construct is based on public-key cryptography in that there is a group of n users each with a public/private key pair, (Pub_i, Pri_i). To request service from a service provider server through a RSU, a node first sends the ID of its group to the RSU. The RSU chooses a challenge, say x, randomly, constructs the verifiable common secret of x defined as $(Pub_1(x), Pub_2(x), Pub_3(x), ..., Pub_n(x))$, and sends it to the node. The receiving node i can decrypt its corresponding encoded message $Pub_i(x)$ with Pri_i to produce x'. Assuming knowledge of the public keys of all other group members, the node can then encrypt x' with the other $n-1$ public keys to produce its own version of the encrypted secrets, $(Pub_1(x'), Pub_2(x'), Pub_3(x'), ..., Pub_n(x'))$, and verify that these secrets match with those received. Finally, the node can send back the challenge x as well as its service request to the RSU. To reduce computation cost, this procedure can be adaptively addressed, probabilistically, by computing only m encrypted secrets randomly chosen out of the $n-1$ secrets and comparing those with the m corresponding secrets received. The computation cost can thus be adjusted simply by tuning m.

The degree k of anonymity can be tuned by having a node send the ID of a subgroup to which it belongs (rather than the ID of the entire group) to the RSU, and having the RSU generate the verifiable common secret out of the public keys of the subgroup members only. It can be achieved as follows. The group information is managed by a central server which, for each group member, creates a public/private key pair, assigns an index to the pair, and maintains group versions. Members of the groups are kept as dummy members before being assigned to new vehicles. Key assignment, in general, is handled by second level key distribution servers. After initialization, a complete binary tree is created consisting of all the keys. Each subtree root is assigned an ID which is utilized by members of the subtree during authentication. The vehicles store all IDs of all subtrees to which they belong. The anonymity level can therefore be adjusted as a node varies the subgroup ID presented to the RSU based on subgroup size. For dynamic management, when keys are revoked, the central server replaces the revoked key pair with a new key pair and updates the group version. As new members join, it puts the new member into the group with the most dummy members.

The primary advantage of this authentication scheme is the adjustable levels of privacy, provided by the adaptive anonymity mechanism, which could facilitate a greater degree of applications. A major drawback to this scheme is its relatively heavy dependence on infrastructure. Another drawback is that [4] views group construction as relatively static since group membership updating is noted to be infrequent. This is assumed to accommodate their group membership information updates, but such an assumption may not be realistic in practice.

3.3.2 Non-group-based Schemes

SRAAC

The work in [5] sought to improve the following security problems in WAVE: 1) Utilizing weak encryption to protect message unlinkability; 2) Reliance on a single certificate authority (CA) for all anonymous certificates, which could easily allow for linkage between certificates and vehicles; and 3) OBU memory and bandwidth usage from WAVE's reliance on certificate revocation lists (CRLs). Secure Revocable Anonymous Authenticated Inter-Vehicle Communication (SRAAC) is designed to improve WAVE and thus is similar to WAVE in many ways. A general overview of SRAAC can be seen in Figure 6.

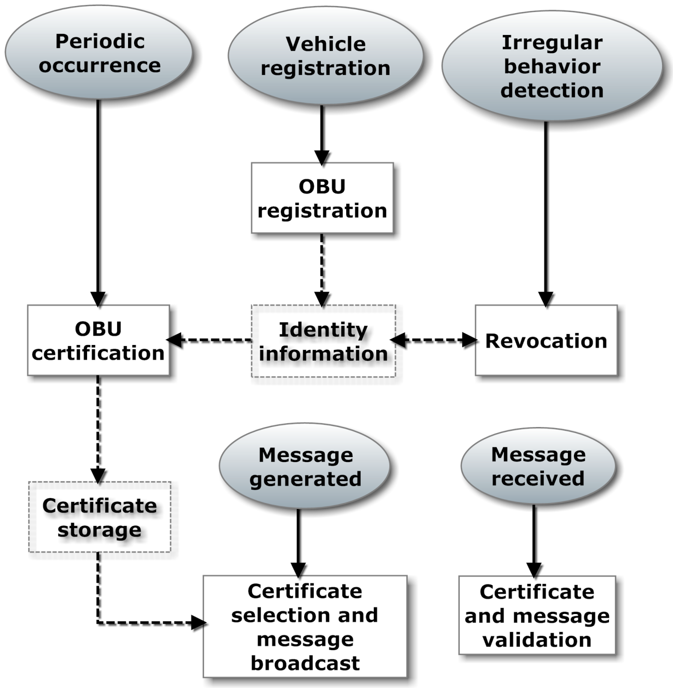


Figure 6. SRAAC scheme overview, redrawn from [5]

Similar to WAVE, each vehicle's OBU is issued a large set of inter-vehicle communication (IVC) certificates. This allows the OBU to change certificates frequently enough to be untraceable even if positional data is included. The certificates are periodically issued to the OBU and have a relatively short lifespan. These factors help in preventing the linkability of messages.

Instead of a single CA, IVC certification is carried out by a quorum of n intermediate IVC certification servers (ICS). These servers, certified by the CA, issue certificates collaboratively using a cryptographic construct known as magic-ink signatures with shared secrets (MI-DSS). A vehicle's OBU periodically creates a new IVC certificate and authenticates itself to an Identity Authority which authorizes the OBU to request the ICSs to sign the certificate, which is then blindly signed by the ICSs using MI-DSS. The use of blind signatures prevent traceability of a vehicle.

This process of blindly signing a IVC certificate also produces a tag, which is saved together with the vehicle's identity in the separate Identity Authority. Later, if a certificate attached to a faulty message is detected, its associated tag can be determined if a revocation key x_t is known, which is protected by a (t_s, n) shared secret scheme in that t_s out of the n ICSs have to agree on the revocation in order to recover x_t. Once the tag is found, the vehicle's identity can be obtained from the Identity Authority and so the vehicle's OBU can be isolated from the network. This isolation will then occur within a certain amount of time as IVC certificates are only valid for a certain time period. Once isolated, the OBU is identified and prevented from receiving new certificates. These factors eliminate the need for CRLs and thus reduce overall memory and bandwidth consumption.

The primary benefit of SRAAC comes from an increase in anonymity and unlinkability due to any IVC certificate issued carrying no linkage information to the vehicle's identity or to another IVC certificate as a result of its blind signature. Another benefit of this scheme is a decrease in memory and bandwidth consumption. This is primarily due to the elimination of the need for CRLs. No experiments were given in [5], so the margin of improvement cannot be stated. One significant drawback to this scheme is the fact that a malicious node is not immediately, or relatively quickly, isolated. When a node is identified as misbehaving they are simply not issued new IVC certificates, but the IVC certificates they currently have could still be valid. As a result, other nodes would still accept the data transmitted by the node as valid.

An Identity-Based Security Framework For VANETs

[21] identified the need for flexibility in setting desired levels of privacy and efficiency in terms of bandwidth utilization for VANET security solutions. It noted the currently proposed solutions fall short in these two areas due to the heavy reliance on public-key cryptography and rigid pseudonym assignments. Delay-sensitive V2V communication was additionally noted as an issue to take into consideration, from which [21] identified a symmetric-key cryptography solution which fell short. To achieve the aforementioned properties, [21] proposed to use identity-based cryptography (IBC) as described by [23] within VANETs. [21] claims to achieve authentication, confidentiality, message integrity, non-repudiation, and pseudonymity.

[21] provides a unique identifier to each vehicle and base station (e.g. RSU), which also denotes its type. These identifiers are periodically certified by a Trusted Authority (TA) and revocation lists are sent to the base stations for storage. During the setup phase the TA computes the relevant system parameters and key information. Every base station I receives the system parameters, master secret key s, and a random secret key K_I. The vehicles receive the system parameters, its identifier, and another public/private key pair. When vehicles need to get a pseudonym, they query a base station and, if their credentials have not been revoked, the base station sends the new pseudonym to the vehicle. This operation utilizes a series of IBC operations, public-key cryptography, and symmetric-key cryptography.

The pseudonyms generated include a timestamp which indicates the point in time in which the vehicle's credentials were validated. This allows a variable level of trust to be assigned as a user could adjust a threshold for how old of pseudonyms to trust. After this, a validation of the identity-based signature on a message allows a receiving vehicle to verify that the sending vehicle, using a pseudonym, has the private key corresponding to the pseudonym. These steps eliminate the need for exchanging certificates between vehicles and the downloading of CRLs.

To accommodate non-repudiation, [21] assumes a black-box is utilized to log messages created and received from a vehicle. A message M in question is turned over to the TA which can verify the message was indeed sent by vehicle pseudonym u and destined for vehicle pseudonym v. Then, through a series of IBC operations and symmetric-key cryptography, the TA can find the true identity of both vehicles.

As with many of the public-key cryptography schemes, [21] is heavily reliant on infrastructure support as well. Though this is true, most of the computation and storage requirements are offloaded to the infrastructure devices as base stations handle all revocations and revocation checking. Unfortunately, no implementation experimentation was performed and thus no performance analysis can be examined.

3.4 SUMMARY AND DISCUSSIONS

A summary of the reviewed authentication schemes along with their categories and other security goals is listed in Table 2. The following notation in Table 1 is used to simplify things on Table 2.

Table 1
Notation utilized in Table 2

Value	**Meaning**
✔	The scheme has or utilizes the property containing the ✔
g	Group key based cryptography is utilized
0, 1, 2, 3	Judged level of infrastructure support

The values of 0,1,2,3 in Table 2 signify the level of infrastructure support outside of initial setup of any of the protocols. That is a zero value means there is no need for RSUs what so ever and an infrequent revisiting of the assets required during initial setup of the protocol. A one value means the protocol could still perform with sparse, possibly even no, RSU type infrastructure support. A two value signifies the protocol requires RSU type infrastructure, but can still partially function whenever an RSU is not immediately available. Finally, a three value, similar to two, requires RSU type infrastructure, but also signifies the protocol will not work when an RSU is not present. That is to say, vehicles do not get the data needed for authenticating other vehicles when they are outside of an RSU's range.

Table 2
Summary of Authentication Schemes

Protocol	Keying Metric		Infrastructure	No TPD	Security Measures Provided			Additional Issues Explicitly
	Asymmetric	Symmetric	Requirement	Required	Anonymity	Non-Repudiation	Revocable Credentials	Identified and Addressed
PPGVC [17]		g/✔	1		✔		✔	Forward/Backward secrecy, Collusion protection
VAST [2]	✔	✔	1			✔	✔	DoS resilience, Multi-hop communication
RAISE [15]		✔	3		✔	✔		Message integrity, Communication efficiency, Internal attack prevention
COMET [15]		✔	1					
PPAA [19]	g/✔		0		✔	✔	✔	Peer accountability, Framing resistance
Hybrid [20]	g/✔		1		✔	✔	✔	Message integrity
GSIS [3]	g/✔		2	✔	✔	✔	✔	Message integrity, RSU ID exposure, Prevention of RSU replication, Computation and communication efficiency
TACK [22]	g/✔		3		✔	✔	✔	Computation efficiency
Xi et al [4]	g/✔		3		✔		✔	Scalability and timeliness, Varying trust/privacy levels
SRAAC [5]	✔		2		✔	✔	✔	Separation of authority, Computation and communication efficiency
Kamat et al [21]	✔	✔	3		✔	✔	✔	Bandwidth efficiency, Confidentiality, Message integrity, Pseudonymity

Several protocols reviewed utilized some form of a symmetric key algorithm. In general, symmetric key algorithms are less computationally intensive than asymmetric algorithms. Though the OBUs are considered powerful enough to make this a moot point in some cases, any protocol which would be intended for utilization in time sensitive safety applications should aim to keep the speed of their protocol as fast as possible as even a few milliseconds can make a big difference in cases such as braking to avoid an accident. Symmetric key algorithms also provide the same level of security with smaller key sizes, thus less storage requirement. However, key exchange becomes an issue when dealing with symmetric key algorithms as a node has to have/establish a common key between any other node with whom they wish to communicate. Whereas asymmetric key algorithms do not. Additionally, in asymmetric key algorithms, once a private key is compromised, only the owner of the compromised key is adversely affected. With symmetric algorithms, anyone using the symmetric key is compromised.

These keying issues are some of the driving factors behind some reviewed protocols utilizing group based communication. When dealing with key management issues (e.g. key exchange, key revocation) groups alleviate some of the overhead and complications. Additionally, since vehicles typically move along the same routes, at around the same speeds, groups seem like a natural choice. However, utilizing groups creates more issues such as establishing the group, joining the group, leaving the group, and revoking a node's membership to the group. All of these, and more, could add more overhead than they save in some cases.

Node anonymity, message authentication, non-repudiation, and computational and communication efficiency are some of the very common core concepts identified in the reviewed papers. Other concerns such as those dealing with RSUs in [3], are introduced due to the specifics of the proposed authentication scheme.

Although a few schemes (e.g. [2,3,19]) directly identified attack vectors within their security concerns in VANETs, all works identified and addressed a possible attack vector towards each security issue (e.g. message authentication with an attack vector of node impersonation).

All of the reviewed schemes attempt to address the respectively identified shortcomings of IEEE 1609.2 and/or various other similar works. While a few, such as [5,24] would still rely on the same proposed cryptographic system (i.e. elliptic curve cryptography) and proposed architecture as in IEEE 1609.2, other schemes take completely different approaches to achieve their goals, such as [15,17,19]. Some other schemes, such as [2,21], tried to make improvements by creating a hybrid scheme which only periodically utilizes public-key cryptography.

Nearly every work included an analysis section for their new authentication scheme. Most works, however, did not provide comparisons against other possible solutions or even against IEEE 1609.2 which is a significant drawback. While some did utilize similar test scenarios (e.g. node densities, communication ranges, map layout) there was no relative consensus on the testbed setup. Additionally, the metrics utilized in assessing the proposed schemes were, in general, not similar across the works. As a result, any direct comparison of efficiency, even though addressed as a concern in several papers, would only allow for very rough estimates.

Several of the reviewed authentication schemes require infrastructure in the form of RSUs and as such will not be complete solutions in areas with little to no infrastructure coverage. A few in fact, such as [4,5,22], are so heavily dependent on RSU type infrastructure that they will not work without vast coverage. However, a few schemes, such as [2,15,17,19], need very little infrastructure to function and thus provide more flexibility in deployment options.

CHAPTER 4
GROUP-BASED LIGHTWEIGHT AUTHENTICATION PROTOCOL

The Group-based Lightweight Authentication Protocol (GLAP) described herein makes exclusive use of symmetric-key cryptographic techniques to authenticate messages sent among nodes of a VANET group. As mentioned earlier, there are two major motivations for such an authentication protocol. First, by the nature of VANETs most of the time vehicles move in groups where an action can affect all the group members simultaneously. Therefore, rather than dealing with pairwise communication issues among the vehicles, it is more efficient and faster to communicate in groups. Second, we follow a symmetric-key based protocol. The use of symmetric-key in place of public-key cryptography significantly reduces the computational overhead and thus the delay associated with authentication. Furthermore, it takes into account the lack of infrastructure support for vehicles. Besides message authentication, GLAP guarantees user privacy in face of passive adversaries, yet provides non-repudiation for law enforcement purposes.

4.1 SYSTEM MODEL AND ASSUMPTIONS

We assume that the on-board unit (OBU) of each vehicle is a *tamper-proof device* (TPD) [25] which securely stores all cryptographic credentials and performs all cryptographic computations. Given that the underlying cryptographic algorithms are secure, no active adversaries are able to forge a message with a valid MAC by cryptanalytic means. Adversaries can obviously inject falsified and malicious data by other means, for example, by cheating the sensors. This threat can be mitigated through the use of data crosschecking and scoring techniques [26].

Each vehicle has a unique identifier V, such as an electronic license plate (ELP), that relates to other information about the vehicle and its registered owner. Furthermore, each vehicle has a unique pseudonym P_V whose mapping with the vehicle identifier V is kept solely by the transportation authority. Both V and P_V are preloaded into the OBU by the transportation authority at the time of vehicle registration.

Furthermore, all OBUs manufactured share two symmetric keys: a *join key* K_J used by vehicles joining a VANET group, as well as a *law enforcement key* K_L, which is also possessed by the law enforcement agency (not the transportation authority). Since these two keys are universal, their lengths should be sufficiently long, for example, at least 128 bits.

Finally, time synchronization among vehicles is assumed. This can be easily achieved if each OBU is equipped with a GPS receiver.

4.2 PROTOCOL OVERVIEW

In GLAP, the groups are created based on a distributed algorithm among the vehicles within a neighborhood. Each node is in one of the following three roles at any moment: non-member, group leader, or group member. A *non-member* node is not associated with any group. A *group leader* is associated with a single group only of which it is the creator and in charge of. A *group member* belongs to one or more groups yet cannot be the leader of a group. The idea of GLAP is based on identifying group leaders so non-members can join one of the groups nearby. Specifically, the nodes within the transmission range of a group leader form a group. In designing the group construction and maintenance algorithm, we aimed at minimizing overhead in the sense of both packet size and number of messages required in order to find/establish a group and start group communication. The algorithm has been constructed in such a way that no query/response messages are required, thus eliminating the possibility of half-opened connections.

Initially, all nodes will start out as non-member nodes. After changing this status, a node will only revert back to group construction procedures, that is become a non-member node, if it was a group member and no other groups are within range. Note that all the nodes of a group share a symmetric group key, which is used to construct the MAC of every safety message sent by any node to others within the same group. However, there is no need to change the group key when a vehicle leaves the group.

A pseudocode of the group construction and maintenance algorithm is depicted in Algorithm 1. Next, we will describe group construction and maintenance under different subsections by referring to Algorithm 1. In each instance, we further divide the sections into the actions performed by both group leaders and non-group leaders under each scenario.

4.3 GROUP CONSTRUCTION

4.3.1 Non-Member Nodes

Nodes which are not currently associated with any groups will construct a *search* message S which consists of nothing more than a randomly generated number x and they will periodically broadcast this message as seen in line 2 of Algorithm 1. The actual range of the number is irrelevant so long as there is at least one higher number than another. Although in essence this number could be represented by a single bit, we allocated a few bytes in our experiments to allow greater flexibility should it be needed.

Upon receipt of an S message from a non-member node, a contention will be performed at the receiving node, say V, as detailed in Algorithm 2. That is, the random number y will be extracted from the S message and compared versus V's random number x. If V's random number is greater than the sender's, V will promote itself to group leader status and begin broadcasting GL messages, constructed using the join key K_J as seen in lines 8-11 of Algorithm 2. If V's

Algorithm 1 GLAP Group Construction & Maintenance

```
if group leader == false && member == false then
    if S received then
        node contention
    else if GL received then
        join group associated with GL message
    else
        create and broadcast S
    end if
else if group leader == true && member == false then
    if GL received then
        node contention
    else
        create and broadcast GL
    end if
else if group leader == false && member == true then
    if GL received then
        join group associated with GL message
    end if
    recentTS ← query $L$ for first $T$
    if recentTS > threshold then
        revoke all group membership
        member ← false
    else
        unicast a message-MAC pair to all group leaders
    end if
end if
```

number is smaller, it will simply continue the procedure of broadcasting S messages and doing contentions. Any node which is already a member of a group or a group leader will simply discard any S message received.

Alternatively, we may instead initialize all nodes as group leaders, and this would eliminate the need for an additional, unsecured, message S. As a result, nodes would simply follow the procedures as described in 4.3.2 instead and thus nodes would only alternate between group members and group leaders. This, however, increases the usage of the join key K_J significantly. This was undesirable and thus the overhead incurred for sending our new additional message is seen as a fair trade-off.

Algorithm 2 Node Contention

```
Require: (group leader == true && $GL$ received) || (group leader == false && member == false && $S$ received)
1: $x \leftarrow$ own number to compare against
2: $y \leftarrow$ extracted number from message received
3: if $x < y$ then
4:     if $GL$ received then
5:         group leader $\leftarrow$ false
6:         join group associated with $GL$ received
7:     end if
8: else
9:     if group leader == false then
10:        group leader $\leftarrow$ true
11:        generate group key $K_G$ randomly
12:        begin broadcasting $GL$ messages
13:    end if
14: end if
```

4.3.2 Group Leaders and Group Members

Once a node assumes the role of a group leader, G, it will begin periodically broadcasting a GL message (lines 5-6 in Algorithm 1) which is constructed by encrypting a triple, consisting of the timestamp T, the group key K_G and the *metadata field* m, using the join key K_J:

$$G \rightarrow * : \{T, K_G, m\}_{K_J}$$

Any vehicle receiving such a GL message is aware of the existence of a group leader, and can thereby join the group by decrypting that message with the join key K_J and verifying the timestamp T, finally obtaining the group key K_G.

The group key should be chosen randomly by a node when it first becomes a group leader. The metadata field m can be used to store any control data for group management (e.g. message count, neighbor count, or random number), which will be detailed in the next subsection.

4.4 GROUP MAINTENANCE

4.4.1 Group Members

To support multiple group memberships, each member node maintains a *group-info list* L that stores the information of each group of which it maintains membership. Whenever a node receiving a GL message joins a group, a list element will be constructed consisting of the timestamp T, the group key K_G, and the group leader's address extracted from the GL message just received. This element will then be inserted at the top of L. As such, L becomes sorted by freshness of the timestamps. As shown in Algorithm 1 beginning on line 19, a member node will periodically perform a check on the most recent group information on the top of L. Should that group's timestamp T be past a certain threshold (which also implies that the timestamps of all other groups in L are past the threshold as well), the node will revoke its membership of all groups by emptying L, thereby assuming the role of a non-member, and begin broadcasting search messages S again as described in 4.3.1. In fact, it is often feasible to even scan L periodically and check each of its elements for timestamp freshness. Any time a list element is encountered with a timestamp that is beyond the threshold limit for timestamp freshness, that element and all elements after it may be removed from L.

Group information from received GL messages are not blindly added to the list L however. Whenever a GL message is received and verified, an iterative search should first be performed on L to check if the node is already a member of the group associated with that GL message. If so, the information for the group is simply updated by removing the old entry from L and inserting the new information at the top of L. Since the maximum allowable number of group memberships (i.e. the capacity of L) is typically small and fixed, any iterative search on L can be performed in constant time.

4.4.2 Group Joining

It was identified early on that should a node encounter a group, no matter for how brief of a time, the node would join the group and start sending data to it. This, in essence, was not immediately identified as undesirable as it would allow data to flow more freely throughout the network. However, given our restriction the capacity of the group list L be kept small, say two for example, then data flow is negatively impacted. This occurs as a node will constantly cycle through membership of the groups which are located around the node. As such, more time is spent performing the join process than on actually sending data to those groups. To remedy this issue we propose several solutions.

1. The most basic solution would be to only allow a node to join another group should there be room within the node's group listing L. As expired group information is already periodically checked for and removed from L, it is simply a matter of ensuring L is not full when attempting to insert the new information into L. For sake of simplicity we utilized this approaching in modifying GLAP.

2. Should a node receive a GL message, and the L has reached a soft threshold, say $lsize_{soft}$, of fullness, instead of immediately performing grouping measures it will start a timer. Should the node receive a threshold, say $count_{GL}$, amount of GL messages over a certain period of time, than it is the case that the new group has a high probability of being within communication range for an extended period of time driving (i.e. the group is traveling along the same road, a similarly directed maintenance road, a

parallel street, etc). After $count_{GL}$ has been reached the node can begin group joining procedures. This can continue to occur until a hard threshold, say $lsize_{hard}$, is met at which time the node will not join any new groups unless old groups are left due to group membership information expiration. This approach offers the overall best compromise out of the proposed approaches.

3. As L is ordered and old information is already identified and removed, a node could simply be allowed to join as many groups as needed. This approach allows for the highest degree of connectivity, however it has several drawbacks. First, the time taken to iterate through L is less predictable as the size of L will unpredictably fluctuate. Secondly, additional checks should be made to ensure malicious nodes are not arbitrarily spoofing grouping information in an attempt to overwhelm some node (e.g. a check to see if x new grouping information has been seen over y seconds, etc.). However, as grouping information expiration time is set on the node itself and not contained within the group joining message, this would have a limited impact on the node should only one malicious node be attempting this. After the expiration time the expired information would be removed and thus the list would fluctuate between a relatively small range of sizes, as a malicious node can only produce so many messages without jamming the communication channel by flooding it.

4.4.3 Group Leaders

Once a node becomes a group leader, it will only lower its status when another group leader is encountered, or the system becomes inactive (e.g. the vehicle is turned off). Upon receiving a rival group leader's GL message, the receiving node will decrypt and verify the message as all other nodes do. Once validity is assured, the node will perform a node contention as outlined in Algorithm 2 lines 1-6, in which the node's control value x (such as message count) stored in the metadata field m of its own GL messages is compared against the control value y extracted from the GL message received. Should the receiver have a larger control value it will continue to serve as the group leader normally would and thus returns to its normal duties. Otherwise it will demote itself to a member of the sending group leader's group. It does not have to wait for another GL message as it has already verified the validity of the current GL message and may join the group with it.

4.5 DATA COMMUNICATION AND AUTHENTICATION

Every node should regularly send updated safety messages to other nodes within the same group, for each group recorded in the group-info list L. The data D of a safety message M would typically contain information such as the current speed, location and trajectory of the node. A node V can send a safety message to a particular group as follows:

1. Construct the *law enforcement access field* (LEAF) by concatenating V's pseudonym P_V with the group key K_G obtained from L, padding the result with random bitstrings, and encrypting it with the law enforcement key K_L:

   $$LEAF = \{r_1||P_V||K_G||r_2\}_{K_L}$$

   where $r_1, r_2 \in_R \{0,1\}^{32}$.

2. Construct the safety message M, consisting of the data D, the LEAF, and the timestamp T:

   $$M = \langle D, LEAF, T\rangle$$

3. Compute the HMAC [27] of the message M using the group key K_G, and send the message-MAC pair via unicast to the group leader, G (unless V is the group leader itself). For this unicast, V should use a fake source address (i.e. at the link layer) generated randomly using a pseudo-random number generator:

$$V \rightarrow G : M, HMAC_{K_G}(M)$$

4. G would broadcast that message-MAC pair received from V to the entire group. Note that the group leader's message can reach to every node within the group:

$$G \rightarrow * : M, HMAC_{K_G}(M)$$

Any node within the same group receiving such a message-MAC pair would authenticate the data D by verifying both the timestamp T and the MAC (recomputing it using the group key K_G). The LEAF would be discarded.

CHAPTER 5
PUBLIC-KEY GROUP-BASED LIGHTWEIGHT AUTHENTICATION PROTOCOL

Though we had achieved an efficient authentication protocol with GLAP, we assumed the availability of TPDs which are currently very expensive. In case, similar devices which are not 100% tamper proof are used because of cheaper costs, it may be possible that they can be compromised. In such cases, the universal keys, K_J and K_L will also be compromised. Having unfeathered access to these two keys would allow an individual to impersonate any node and/or any group within the network. This in turn results in a compromise of the entire system solely from compromising one TPD. As such we added the capability to utilize public-key infrastructure (PKI)/public-key cryptography (PKC) in conjunction with symmetric-key cryptography in an attempt to prevent this single point of failure. We denote this PKI extension of GLAP as PGLAP throughout the rest of the document.

5.1 ENTITY KEY AGREEMENT

One of the issues faced when utilizing PKC is generating a common key between two entities to allow secure communication to begin. As the WAVE standard suggests the utilization of elliptic curve (EC) cryptography (ECC) to secure communications within a VANET. PGLAP utilizes the elliptic curve Diffie-Hellman (ECDH) key agreement protocol in dealing with communication between two nodes when a certificate is involved (i.e. $K_{G'}$ and LAW_{PUB}). This allows two entities, who each have a public/private key pair, to securely establish a shared secret through a possible insecure medium.

In short, suppose we have two nodes U and V which have previously agreed on a set of EC domain parameters. Each node has an EC public/private key pair, say (U_d, U_Q) for U and (V_d, V_Q) for V where d is the private key and Q is the public key. The private key d is a randomly selected integer over a specific interval and $Q = d * G$, where G is a generator point. U finds a shared secret by taking V's public-key and finding the product. Formally, the two results are equivalent as shown:

$$K_{G'} = V_d * U_Q = V_d * U_d * G = U_d * V_d * G = U_d * V_Q = K_{G'}$$

5.2 PROTOCOL OVERVIEW

PGLAP behaves much in the same way as GLAP as it is merely an extension which facilitates the utilization of PKI/PKC within the network. As in GLAP, PGLAP utilizes a distributed algorithm among vehicles within a neighborhood to create groups. The node categories and their status changes are also as in GLAP. Essentially, what is different in PGLAP is the elimination of the two universal keys, K_J and K_L.

Specifically, any node wishing to join a VANET group would broadcast its public key certificate. A group leader would verify the certificate and encrypt the group membership information payload with the nodes public key retrieved from the certificate. In this sense, the join key, K_J, would be eliminated. Group communication would still be based on symmetric-key cryptography to meet the real-time constraints of VANETs.

Similarly, instead of a universal K_L, a session key encrypted using PKC would be used. Specifically, the $LEAF$ will be encrypted with a symmetric session key as in the case of K_L, yet such a session key is randomly generated whenever a node wishes to join a group. This session key is encrypted with the law enforcement public key, whose public key certificate is preloaded on every nodes TPD. The encrypted session key is included in every S message broadcast from the owning node. Law enforcement can obtain this encrypted session key and thereby decrypt a LEAF either by sniffing S messages or by having any group leader receiving the S message store the encrypted session key and transmit it to law enforcement upon contact with a RSU later.

The group construction and maintenance algorithm pseudocode is depicted in Algorithm 3. We next describe group construction and maintenance under different subsections by referring to Algorithm 3. In each instance, we further divide the sections into the actions performed by both group leaders and non-group leaders under each scenario.

5.3 GROUP CONSTRUCTION

5.3.1 Non-Member Nodes

Nodes which are unassociated with a group periodically construct a *search* message S consisting of a randomly generated number x, an encrypted version of the law key K_L, and the node's public certificate K_{PUB}.

$$S =< N_R, \{K_L\}_{LAW_{PUB}}, K_{PUB} >$$

This will be periodically broadcast as seen in line 7 of Algorithm 3. Similar to GLAP the range of the number is irrelevant so long as there exists a number which is higher than another.

When the receiving node, say V, is a non-member and receives an S message, a contention will be performed as detailed in Algorithm 4. That is, the random number encoded in S will be extracted and compared versus V's random number. If V's random number is greater than the sender's, V promotes itself to group leader status, generates a group key K_G, and then unicasts the grouping

Algorithm 3 PGLAP Group Construction & Maintenance

```
1: if group leader == false && member == false then
2:    if $S$ received then
3:       node contention
4:    else if $GL_u$ received then
5:       join group associated with $GL_u$ received
6:    else
7:       create and broadcast $S$
8:    end if
9: else if group leader == true && member == false then
10:   broadcast $GL_b$
11:   if $S$ received then
12:      unicast $GL_u$ to sender
13:   else if $GL_b$ received then
14:      unicast $S$ to sender
15:   else if $GL_u$ received then
16:      node contention
17:   end if
18: else if group leader == false && member == true then
19:   if $GL_b$ received then
20:      if already member of group then
21:         rejoin group associated with $GL_b$
22:      else
23:         create and unicast $S$ to group leader
24:      end if
25:   else if $GL_u$ received then
26:      join group associated with $GL_u$
27:   end if
28:   recentTS ← query $L$ for first $T$
29:   if recentTS > threshold then
30:      revoke all group membership
31:      member ← false
32:   else
33:      unicast a message-MAC pair to all group leaders
34:   end if
35: end if
```

information via a GL_u, the contents of which are detailed in section 5.3.2, to the other node as seen in lines 10-12 of Algorithm 4. Should V's number be smaller, it will simply continue broadcasting S messages and doing contentions.

Algorithm 4 Node Contention

```
Require: (group leader == true && $GL_u$ received) || (group leader == false &&
    member == false && $S$ received)
 1: $x \leftarrow$ own number to compare against
 2: $y \leftarrow$ extracted number from message received
 3: if $x < y$ then
 4:     if group leader == true then
 5:         group leader ← false
 6:         join group associated with $GL_u$ received
 7:     end if
 8: else
 9:     if group leader == false then
10:         group leader ← true
11:         generate group key $K_G$ randomly
12:         unicast $GL_u$ message to other contending node
13:     end if
14: end if
```

5.3.2 Group Leaders

When a node assumes group leader status it begins periodically broadcasting GL_b messages as shown in line 10 of Algorithm 3. This is different from GLAP as GLAP's GL message functionality has been split into two separate messages in PGLAP. The broadcasted GL_b message is utilized for group information updates and the unicasted GL_u is utilized for allowing nodes to join the group.

The GL_b message is the most similar to GLAP's GL message save for the symmetric key utilized to encrypt the message. GL_b consists of a triple, the timestamp T, the group key K_G, and a *metadata field* m. Instead of utilizing a universal join key (e.g. K_J in GLAP) to encrypt GL_b, it is encrypted with the group key K_G:

$$G \rightarrow * : \{T, K_G, m\}_{K_G}$$

Since the GL_b message's primary function is to freshen group information on nodes within the group, utilizing K_G works as all group members already have this key. The group key K_G should be chosen randomly by a node when it first becomes a group leader. The metadata field m can be used to store any control data for group management as in GLAP. The GL_b message could additionally be utilized to periodically change the group key K_G in an attempt to help further increase node anonymity.

Upon receipt of an S message a group leader, say G, will extract the sender's, say U, certificate information and utilize this to construct the shared secret $K_{G'}$. G then creates what is known as a GL_u message. That is, G encrypts the group information with $K_{G'}$, appends it's certificate K_{PUB}, and then unicasts, as shown in line 12 of Algorithm 3, it to the node who sent the original S message:

$$G \rightarrow U :< \{T, K_G, m\}_{K_{G'}}, K_{PUB} >$$

Upon receipt of a GL_u message, a non-group member node will extract G's certificate, and then use it to find the $K_{G'}$ utilized to encrypt the group information. Once V has $K_{G'}$ it can decrypt the remainder of the GL_u message and then extract the group key K_G. Similar to GLAP, before the group information is added to V's group member list the timestamp T in the group information is checked to ensure freshness. Only if everything succeeds does the node join the group.

5.3.3 Group Members

A node who currently has group member status, say V, will join another group when a GL_b message, for a group it is not currently a member of, is overheard. As depicted in line 23 of Algorithm 3, when this situation occurs V will construct an S message as if it was a non-member and unicast it to the group leader who sent the GL_b message. As the group leader does not differentiate between actual non-member nodes and member nodes when dealing with an S message, the processing is handled the same. V also handles the GL_u sent in reply to the S message in the same manner as if it had non-memeber status.

5.4 GROUP MAINTENANCE

5.4.1 Group Members

Upon receipt of a GL_b message for a group which the node is currently a member of, the data in L is updated in a similar manner as in GLAP when dealing with a GL message. Additionally, PGLAP, like GLAP, supports multiple group memberships as eluded to in subsection 5.3.3 of this chapter. It maintains a *group-info list* L for storing group membership information of the groups the node is currently a member of. Whenever a node joins a group, a list element will be constructed consisting of the timestamp T, the group key K_G, and the group leader's address, all of which is extracted from the GL_u message. The grouping information is then inserted to the top of L and as such, L is sorted by timestamp freshness. Line 28 of Algorithm 3 shows that a node periodically performs a check on the most recent group information, which is at the top of the list L. Should the timestamp T be past a threshold than the node revokes all remaining group membership by emptying L. As such, the node changes it's status to be that of a non-member and begins the process of broadcasting S messages again.

Grouping information, as in GLAP, is still checked to ensure the group information of K_G and the group leader is not already present in L. If the information is already present, the old information is removed from L and the new information is inserted at the top of L. The capacity of L is additionally kept small in PGLAP as in GLAP to ensure any iterative search of L can be performed in constant time.

5.4.2 Group Leaders

Upon becoming a group leader, a node only drops this status when another group leader is encountered or the systems becomes inactive (e.g. the vehicle is turned off). Upon receipt of the rival group leader's GL_b message, the receiving group leader, say V, will respond with an S message to initiate the process of getting the group information from the rival group leader, say U, as depicted in line 14 in Algorithm 3. A GL_u message will be sent to V from U. V extracts the control data encoded in the metadata field m from the U's GL_u message and performs a contention as detailed in lines 3-6 of Algorithm 4. V's own metadata field information is utilized as the x value and U's data is utilized as the y. Should V have a larger control value than it will continue to function as a group leader. However, should the value be lower, than V will demote itself to a group member of U's group. As this process is taken, V does not have to issue any additional messages to retrieve the group key K_G from U. It can immediately join the group and begin data communication within the group.

Seeing as the process of contending with other group leaders contributes to more control packet overhead, one modification which could be made to this scheme deals with when a node V first joins a new group. After key exchange information has occurred and the node is fully integrated in the group, thus ready to send data messages, V could send an additional control type packet. This packet would contain all group keying information contained in V's group information list L which is still valid (i.e. the timestamp T is not past the expiry threshold). This would allow two group leaders to have a high probability to be able to instantly start the node contention process of Algorithm 4 instead of having to first find a shared secret $K_{G'}$.

5.5 DATA COMMUNICATION AND AUTHENTICATION

In dealing with data communication the only change to occur between GLAP and PGLAP is the construction of the law enforcement access field $LEAF$. The $LEAF$ field is now padded with timestamp T information instead of a random bitstring padding. That is, the timestamp T is split into two portions T_1 and T_2. The reason for this switch in padding is further explained in 6.3.2. These values encase the node's pseudonym P_V and the group key K_G for which the message is destined:

$$LEAF = \{T_1, P_V, K_G, T_2\}_{K_L}$$

All further data communication and authentication for PGLAP occurs as in GLAP. Group members still regularly send updated safety messages to other nodes within each group for which the group member has valid information. That is, information contained within the group information list L which is not expired. The $LEAF$ is constructed as previously stated. This is then added, along with the message data D and timestamp T, to a safety message M as:

$$M = \langle D, LEAF, T \rangle$$

The HMAC [27] of the message M is computed using the group key K_G and then sent as a message-MAC pair via unicast to the group leader, G (unless V is the group leader itself):

$$V \rightarrow G : M, HMAC_{K_G}(M)$$

Finally, G broadcasts the message-MAC pair received from V to the entire group:

$$G \rightarrow * : M, HMAC_{K_G}(M)$$

Any node within the same group receiving such a message-MAC pair would authenticate the data D by verifying both the timestamp T and the MAC (recomputing it using the group key K_G). The LEAF would be discarded.

CHAPTER 6
PROTOCOLS SECURITY ANALYSIS

Provided that a secure underlying hash function is used and the key length is sufficiently long, the HMAC algorithm, of both GLAP and PGLAP, provides message authentication, message integrity, and is provably secure against forgery attacks. Unlike the join key K_J and the law enforcement key K_L in GLAP, since the group key K_G, for both GLAP and PGLAP, used for MAC construction is not universal, its length can be relatively shorter, for example, 80 bits. The following parts discuss and analyze other security services provided by GLAP and PGLAP, as well as other implementation issues of our concern.

6.1 PRIVACY

6.1.1 GLAP

GLAP preserves privacy, as no vehicle identifiers are included in any GLAP messages. Furthermore, since the law enforcement key K_L used for encrypting the $LEAF$ is possessed by tamper-proof OBUs and the law enforcement agency only, no eavesdroppers are able to decrypt the $LEAF$ to obtain the vehicle pseudonym, which could be used as an alternative label for identifying a particular vehicle. Of course, if the $LEAF$ were a function of solely the vehicle pseudonym P_V and the group key K_G, both of which typically remain unchanged for a certain period of time, then the same vehicle would generate multiple identical $LEAF$s, which again could serve as an alternative label for that vehicle. To eliminate this problem, random pads (64 bits in total) are appended to P_V and K_G before encryption in the $LEAF$ construction process. In other words, the $LEAF$ encryption function is practically made probabilistic, resulting in different $LEAF$ values given the same P_V and K_G. Finally, the degree of message anonymity is further increased with use of fake source addresses when sending safety messages to the group leader.

6.1.2 PGLAP

Similarly, PGLAP contains the same properties as GLAP with regards to the $LEAF$. Since K_L is randomly generated before a node joins a group and is different for each the node joins, this further decreases the chance of cryptanalysis. Furthermore, PGLAP does not utilize random pads, but instead utilizes a timestamp T in place of the pads. To help maintain unlinkability between any two $LEAF$s from the same node, T is divided into two parts T_1 and T_2. Each part is utilized in a similar fashion as the random padding of GLAP, that is, T_1 is prepended to P_V and K_G while T_2 is appended.

PGLAP additionally does not utilize only a single public/private key pair for the asymmetric cryptographic purposes. Each node is loaded with a relatively small pool of certificates for all their personal key pairs over a specific time interval, which can be updated whenever they are within an urban area (i.e. have access to an RSU). The certificates are only valid over specified time intervals, so as such it must be periodically updated for a node to be able to participate in group construction. These steps will aid with anonymity by decreasing the odds of a node being tracked based on where the associated certificate is seen. Additionally, as the public/private key information is only utilized during group creation, no information need be retained when flushing used certificates.

6.2 EFFICIENCY

6.2.1 GLAP

In GLAP, encryptions are performed in the construction of GL messages and the $LEAF$. Assuming that AES [28] is used for encryption which has a block size of 128 bits, both a GL message and the $LEAF$ can typically fit within two AES blocks, i.e., 256 bits. In the case of constructing the $LEAF$, after allocating 64 bits for the random pads r_1 and r_2 and 80 bits for the group key K_G, there are still 112 bits remaining for the vehicle pseudonym P_V, and that is more than enough to represent the pseudonyms of all possible license plate numbers in any jurisdiction (e.g. the number of possible license plate numbers in the U.S. is not more than 2^{42}). Since only two AES blocks are involved in any single encryption, the encryption can be performed simply in the *electronic codebook* (ECB) mode of operation [29] without losing any security strength compared with other block cipher modes. The insertion of two different random pads at both ends of the $LEAF$ introduces randomness in the encryption of each of the two AES blocks, preventing any adversaries from observing repetitive ciphertext blocks. Using the speed benchmark from Table 3, a single two-block AES/ECB encryption would cost less than 1μs. Nonetheless, a vehicle could further save time by precomputing a set of $LEAF$ values for use within a certain period of time during which both P_V and K_G stay constant.

Since public-key cryptography is slower than symmetric-key cryptography by orders of magnitude, GLAP's exclusive use of symmetric-key cryptography is guaranteed to be much more efficient than the IEEE 1609.2 standard, which proposes the use of ECDSA to digitally sign every safety message [6]. As an example, assuming that a GLAP safety message is 512 bits long (192 bits for the data D, 256 bits for the $LEAF$, and 64 bits for the timestamp T), it would cost only 0.5μs to compute the HMAC of a safety message according to the speed benchmark from Table 3, which is at least 5000 times faster than ECDSA signature generation and 10000 times faster than signature verification. Although this performance analysis does not take into account the computational overhead of group management, such an overhead is unable to subdue the huge difference in performance between ECDSA and HMAC. In addition, GLAP consumes less bandwidth by eliminating the attachment of public-key certificates to safety messages.

Table 3
Crypto++ library speed benchmarks [1]

Algorithm	**Speed**
AES/ECB	109 MiB/s
ECDSA generation	2.88 ms
ECDSA verification	8.53 ms
ECDH key agreement	2.82 ms
RSA encryption	0.16 ms
RSA decryption	6.08 ms
HMAC/SHA-1	147 MiB/s

6.2.2 PGLAP

PGLAP utilizes a hybrid approach in that it uses both symmetric and asymmetric techniques. All safety messages are encrypted utilizing symmetric techniques as in GLAP to minimize the data delay. ECDH is only utilized once per node during the group joining process, that is once during the creation of an S message and once during the creation of GL_u. As such it is not quite as fast as our GLAP approach. However, it is still several times faster than a straight ECDSA. PGLAP also consumes slightly more bandwidth during control messages than GLAP as the S and GL_u message also require the attachment of a public-key certificate. However, certificates are not required during the transmission of safety messages and as such the extra overhead is kept to a minimum.

We would like to note that the utilization of ECDH for performing encryption rather than RSA is due to stronger security with smaller key sizes, yet providing comparable delay performance. For instance, the delay incurred (i.e., message processing, encryption and decryption) when utilizing RSA with a 2048-bit key is 6.24ms as seen in Table 3. On the contrary, with a ECDH 256-bit key, which is basically equivalent to a 128-bit symmetric key or a 3072-bit RSA key, takes a total time of 2.82 * 2 = 5.64ms to establish a key agreement. Adding the delay of AES shared secret encryption/decryption time which is 1μs makes a total time of 6.64ms.

Additionally, a node could save time by precomputing the shared secret utilized in encryption of the S message, due to the public key information for the law enforcement agency changing infrequently. As the shared secret for each GL_u message will be different for each node, due to each node having a different public key, and it is only generated, in general, once per group leader, there is no benefit to precomputing it.

6.3 NON-REPUDIATION

Both GLAP and PGLAP provide a mechanism that allows law enforcement to track the sender of a safety message. To uncover the identity of the vehicle sending a message-MAC pair that was previously captured, the law enforcement agency can decrypt the $LEAF$ within that message using the law enforcement key K_L whose availability will be discussed shortly under separate subsections.

Once the $LEAF$ is decrypted, the vehicle pseudonym P_V and the group key K_G can be retrieved. Next, the agency would use K_G to verify the MAC to ensure the authenticity and integrity of the message, and then request a court order that would require the transportation authority to disclose the vehicle identifier V corresponding to P_V (as well as possible other vehicle information and/or the registered owner as deemed necessary by the judge). Notice that the vehicle registration database (including the mapping from P_V to V) is maintained only by the transportation authority, not the law enforcement agency. On the other hand, the transportation authority does not possess K_L and so does not have the ability to decrypt the $LEAF$. Such a design prevents a single governmental agency from having too much power. This is similar to the concept of *key escrow* [30], and is an application of the security principle of *separation of duties* (SoD).

6.3.1 GLAP

As GLAP utilizes a universal law key K_L, it is available at the law enforcement agency and thus any $LEAF$ can be easily decrypted.

6.3.2 PGLAP

Since each node generates a different K_L for each group joined, the law enforcement agency must be made aware of these keys. One way to do this would be for the group leader to save the S message when a node attempts to join the group and then send it to an RSU whenever possible. The RSU can then relay this information to the law enforcement agency which can use its private key to retrieve the session key, K_L.

As our assumption about a TPD has weakened, it is now possible for a node, say V, to capture another node's LEAF while in a group. It should be noted it is still impossible for the node to decrypt the LEAF without the symmetric key. While including the timestamp does not prevent V from utilizing the captured LEAF in a message, it does prevent the actual owner of the LEAF from being wrongly accused based on information contained in the LEAF as V will not be able to effectively create a current message which contains a valid, matching timestamp and thus impersonate the LEAF owner.

6.4 DENIAL-OF-SERVICE RESISTANCE

GLAP and PGLAP exhibit a high degree of resistance against DoS attacks. The group construction algorithm, Algorithm 1, was designed to guard against attacks by adversaries wishing to prevent nodes from establishing or joining a group. Consider the scenario where there are three non-member nodes, one malicious and two legitimate. The malicious node can prevent each legitimate node from creating a group by broadcasting a S message that contains an arbitrarily high number, thus prohibiting the two nodes from becoming a group leader in opposition to the malicious node. The two legitimate nodes, however, will also exchange S messages, thus allowing for one to become a group leader over the other. Extending this to the general case, the exchange of S messages with every neighbor in range ensures a legitimate node within the neighborhood will eventually create a group for others to join. Below we explain how GLAP and PGLAP address other issues:

6.4.1 GLAP

In GLAP, a malicious node will be unable to create a group for a legitimate node to join because it is incapable of constructing a valid GL message without the knowledge of the join key K_J. In addition, none of the three types of GLAP messages (i.e., S, GL and safety messages) induces any reply messages in the protocol. Both the sender and the recipient of any GLAP message does not waste time and storage waiting for any response. Consequently, there exists no half-opened connections that can be exploited by an adversary, thereby preventing any memory-based DoS attacks. Without knowledge of the join key, K_J and the group key, K_G, a malicious node can only generate and broadcast a bogus GL message with an invalid timestamp or a bogus safety message with an invalid MAC. However, the extreme speed of symmetric-key decryption and computing the MAC implies that such bogus messages can be detected and discarded almost instantly. GLAP could be made slightly more efficient, in terms of grouping, by having group leaders respond to any S message received by re-broadcasting the GL message right away. However, this is disallowed in GLAP to prevent Smurf-like DoS attacks.

6.4.2 PGLAP

In PGLAP a malicious node could form a possibly legitimate group for others to join, however the checking of certificate validity would, in general, show the newly assigned group leader is actually an invalid node. As in GLAP, no PGLAP message (i.e. S, GL_u, GL_b, and safety messages) requires a reply type message. However, PGLAP does potentially require the group leader, to retain S messages from the group members for a period of time. To help prevent a memory-based DoS attack from a possible malicious node spamming arbitrary S messages, the retaining time should be kept small unless the node sending the initial S message has successfully joined the group.

6.5 REPLAY RESISTANCE

For both GLAP and PGLAP, the inclusion of timestamps in GL, GL_u, GL_b, safety messages and the $LEAF$ prevents replay attacks. Since a safety message goes through a group leader before being forwarded to the recipients, the timestamp of a safety message should sustain a slightly longer lifetime than that of any GL message.

If a higher level of security is desired such that a dedicated adversary is prevented from instantly replaying a captured GL message within the narrow timeframe before its timestamp expires, every GL message received and verified can be checked against each entry of the group-info list L. Any such GL message that is found as a replay (whose timestamp and group key exactly match one of the entries in L) is rejected and L will not be updated. The probability that two distinct group leaders send a GL message with identical group keys and identical timestamps should be negligible. For this check to execute at constant time, a cap should be put for the maximum number of group memberships allowed for a vehicle.

6.6 KEY FRESHNESS

This is an issue in GLAP only as we use two universal keys. Regardless of the security strength of the cryptographic primitives used, key freshness is always a desirable property. In other words, a cryptographic key should be renewed at regular times to further thwart statistical cryptanalysis. Among all keys involved in GLAP, the join key K_J is the most attractive target because an adversary can impersonate a group leader if the join key is compromised. The following suggests several ways to preserve the freshness of K_J:

- A collection of join keys is preloaded into all OBUs at periodic technical checkups of a vehicle. Assuming that technical checkups are done yearly and that a join key is 128 bits long with a lifetime of 5 minutes, approximately 1.6 MB of join key data need to be stored into a vehicle's OBU at each technical checkup.

- A *cryptographically secure pseudorandom number generator* (CSPRNG) is used to generate a join key at regular times. A CSPRNG is a pseudorandom number generator that passes the *next-bit test* [29]. More precisely, a CSPRNG is a deterministic algorithm that takes a single input, called the *seed*, and generates a sequence of pseudorandom numbers $n_1, n_2, n_3, \ldots$ such that, given the first ℓ numbers $n_1, n_2, \ldots, n_\ell$ of the output sequence, there exists no polynomial-time algorithm that can predict $n_{\ell+1}$ with high probability of success. Assuming that all OBUs manufactured share a secret seed and implement the same CSPRNG, the join key at time interval i can be defined as the i^{th} number n_i generated by the CSPRNG. The use of a CSPRNG naturally offers the property of *forward secrecy*, meaning that the compromise of a join key will not reveal any future join key subsequently used. The secret seed is resistant to cryptanalysis because it is never used to directly encrypt or compute the MAC of any message available to eavesdroppers. Nevertheless, the seed can be renewed periodically (e.g., yearly) to refresh the pseudorandom number sequence by hashing the seed itself, taking the digest as the new seed.

- GPS satellites can be used to securely distribute new join keys to OBUs at regular times. Satellites and OBUs need not share any secret key for the secure distribution of join keys. For example, satellites can periodically send a signal that contains a random bit sequence, and all OBUs manufactured can share a secret key that is used to derive the algorithm for extracting a new join key from the random bit sequence.

Similar to the join key, the law enforcement key K_L can be periodically renewed using any of the aforementioned approaches, yet it can withstand a much longer lifetime. The group key K_G of a node should be continuously changed by itself as the node leaves and joins different VANET groups, but a group leader can choose to periodically renew the group key and distribute it to all group members.

If a higher degree of privacy is desirable, a vehicle's pseudonym can be renewed whenever it is uncovered by law enforcement in order to prevent law enforcement personnel from using an uncovered pseudonym as an alternative label for identifying that vehicle in the future. Since $LEAF$ decryption should be a sporadic event, pseudonym renewal need not occur frequently. For example, a vehicle's pseudonym can be renewed yearly using a small preloaded set (e.g., 50 pseudonyms) stored at the time of OBU manufacture, or renewed by a court order if it is uncovered.

CHAPTER 7
EXPERIMENTATION AND RESULTS

7.1 SIMULATION OVERVIEW

We implemented our protocol in a simulation environment to assess its performance under a variety of conditions.

7.1.1 Simulation Setup

The simulations were evaluated on a single core of an eight core Intel®Xeon®E5450 3.00Ghz processor with 32GB of RAM. In evaluating our algorithm we utilized the ns-2 simulator [31] for simulating a VANET environment. The topography was generated utilizing the mobility modeling software from [32]. It was a 2.4 kilometer square area which resembled an urban environment. The vehicles would move, roughly, anywhere from 25 mph to 45 mph depending on the road they were traveling on. The vehicles would also periodically make stops, simulating the need for stopping at appropriate traffic control devices (e.g. stop signs or stop lights). Each run lasted for approximately 5 minutes of simulation time. The results reported are averaged over 30 different runs. For tests where the node count varied, the transmission range was set to 150 meters. For tests where the transmission range varied, the node count was set to 130 nodes.

For GLAP, while the search messages S were broadcast every 900ms, GL messages were sent approximately every $\frac{1}{3}$ of a second once a group leader had been established. Safety messages were sent every 300ms once a node joined a group. The threshold for checking the validity of GL message timestamps was set to 100ms. This threshold was chosen to be as it achieved over 99% validation rate even in the most delay imposing scenarios. The other threshold for group information freshness (i.e. scanning the list L) was picked as 700ms. In our experiments, we allowed the member nodes to join up to two different groups which helped to ensure inter-group connectivity.

The join key K_J and the law enforcement key K_L were generated, from fixed values, upon node creation as they are to be loaded into the OBU upon vehicle registration.

PGLAP was set up in a similar manner as GLAP. S messages were broadcast every 900ms, both GL_u and GL_b messages were broadcast approximately every $\frac{1}{3}$ of a second, and safety data was broadcast every 300ms. Likewise with the packet transmission rates, most all of the various threshold values in GLAP remained the same as well.

7.1.2 Performance Metrics

We consider average end-to-end packet delay, drop ratio, and average throughput for assessing the performance of the proposed protocols. End-to-end delay includes the time it takes to form our message structure, encrypt and decrypt the message, parse the message structure, and the time it takes for a packet to reach the destination after it has left the source node. Control packets are not included in calculations for end-to-end delay, only data packets (i.e. safety messages).

Drop ratio is calculated as the total number of dropped packets over the total number of packets sent. The only drop type not taken into account for the counting is routing level packet drops as all communication in our protocol is single hop transmissions.

Throughput is calculated via counting the total number of non-control packets which make it to their final destination for processing. This value is then multiplied by the number of bits within each such packet. The result is than divided by the total length of the simulation run to give us a measured value in kilobits per second (kbps).

7.1.3 Baselines for Comparison

To establish a baseline for our own cryptographic solutions, we implement a version of GLAP without any encryption or authentication being performed. We denote this implementation as insecure GLAP (iGLAP). This baseline allows us to judge the overall impact of our encryption and authentication scheme to our grouping and communication scheme.

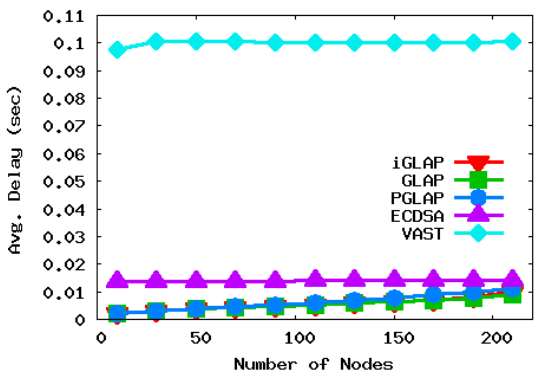
We additionally implement a relatively simplistic and base version of an ECDSA scheme, which mimics IEEE 1609.2, for comparison. In this scheme a node, say V, constructs a message consisting of the information to be sent, V's signature of this information, and V's ECDSA public key. The node then broadcasts the message to surrounding nodes, who verify the signature based on the received key. These messages are sent approximately every $\frac{1}{3}$ of a second, similar to GLAP. No grouping is performed in the ECDSA implemented scheme. We would like to note that in reality the ECDSA implementation should have an even higher delay and drop ratio since real digital certificates have larger sizes and we did not count the time it takes to verify Certified Authority (CA) signatures in our experiments.

In addition to ECDSA, we also ran experiments with VAST, [2], which was also implemented in ns-2. In the implementation, the actual cryptographic operations were simulated by adding in approximate benchmark delay times on both the sender and receiver. The packet's size was generated to simulate a packet containing a signature and/or MAC. As described in [2], packets are broadcast every 100ms.

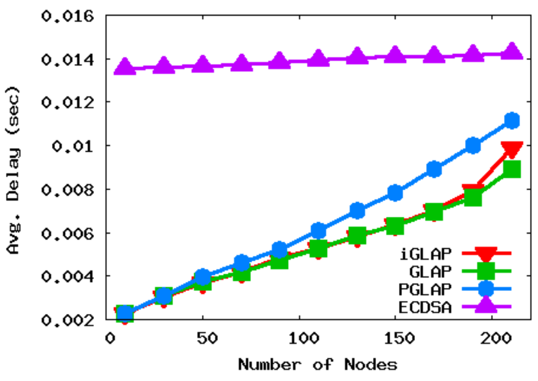
7.2 SIMULATION RESULTS

7.2.1 End-to-end Delay

Figure 7a shows the average end-to-end delay of a typical safety message within our simulation, in seconds, given various node densities. For a better comparison of iGLAP, GLAP, PGLAP, and ECDSA, we also include Figure 7b. The transmission range of the radio device on the vehicle is fixed at 150 meters in these experiments.



(a) Average delay vs Node count



(b) Average delay vs Node count, VAST is not included

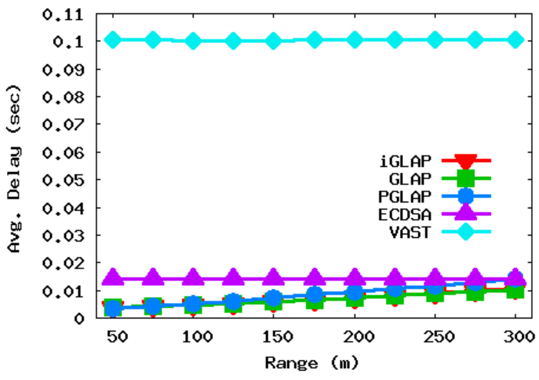
Figure 7. Average delay values when varying node count

We see from Figure 7a that VAST suffers the highest average delay. This is due primarily because, for message M_i, a node must wait for the next heartbeat message M_{i+1}, which is broadcast approximately every 100ms, before the node can verify M_i. From Figure 7b we can see that in all tested scenarios, GLAP and PGLAP have less average delay than the generic implementation of an ECDSA scheme. We also note the efficiency of symmetric-key cryptography in terms of delay as both GLAP and iGLAP produced similar delay results. Even though we are only measuring end-to-end delay for this graph, we can see, in PGLAP, the asymmetric-key cryptography utilized in the control messages does cause PGLAP to incur a slightly higher average delay in most cases. This occurs as while any given message is being processed, any other message received must wait in a queue until the prior message, in a first-in-first-out (FIFO) sense, has finished processing.

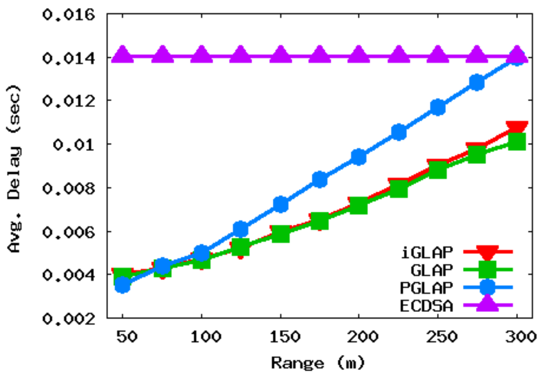
Though iGLAP fluctuates around GLAP's averages, in general iGLAP was found to only be a fraction of a millisecond faster than GLAP which is, in most cases, negligible. The values of iGLAP behave generally as expected save for when the testing reached the upperbound for node count, that is 210 nodes. After further investigation it was noted a few nodes had extremely high delays, one node in one simulation had an average delay of 343.3ms per message, as compared to the other nodes. As such, a standard deviation was performed on the results which proved a nominal variance in the lower node counts, but a significantly high variance in the upper two node counts. Those two counts had a standard deviation of 4.12ms for the 190 node count, which had a average delay of 7.93ms, and a standard deviation of 8.97ms for the 210 node count plot, which had an average delay of 9.90ms. This factor and comparisons with the other data, particularly that of Figure 9 and Figure 10, show iGLAP is receiving a higher delay as only a few nodes are acting as the entire simulations group leader. As such they are being flooded with safety messages. Consequently as the transmission queue increase, so is the delay a message has to wait before being processed. These deviations do not necessarily carry over into GLAP and PGLAP as the grouping of nodes is quite possibly different in those two (i.e. we are not controlling what nodes group where).

We find the primary source of delay in all versions of GLAP is due to a bottleneck which is created from all data being forced to flow through the group leader. This can be seen in the relatively linear delay increase as the number of nodes increase. As the number of nodes per group increase more messages passing through the group leader is incurred. Additionally, this is supported by the standard deviation results generated from the closer inspection on iGLAP. From Figure 7 we can see the impact of having to perform single node asymmetric communications for PGLAP as the slope of PGLAP's plot is quite higher than that of GLAP or iGLAP. With the larger group sizes, the number of nodes performing asymmetric communications to join a group adds to the queuing delay of the group leader more than that of GLAP or iGLAP. Out of all cases when varying the node densities, GLAP's highest average delay time was less than about 9ms and PGLAP's highest average delay was about 11.1ms, while the ECDSA implementation's lowest average delay was about 13.5ms.

Figure 8a and Figure 8b are similar in concept to the previous two figures, Figure 7a and Figure 7b respectively, except instead of varying node densities the transmission range of a node was altered. In these experiments, the number of nodes was fixed to 130 nodes.



(a) Average delay vs Transmission range



(b) Average delay vs Transmission range, VAST is not included

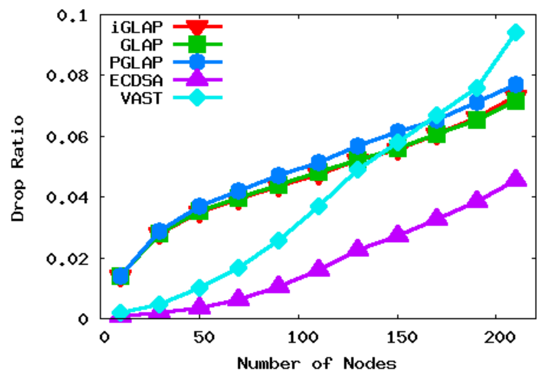
Figure 8. Average delay values when varying transmission range

VAST again has the highest average delay, as seen in Figure 8a, due to the aforementioned need to wait for an additional heartbeat message. From Figure 8b we see the optimum range for minimum average delay with GLAP, PGLAP, and iGLAP, would be 50 meters which was measured to be about 3.9ms. The three intersect with an average delay of 4.0ms at around 65–70 meters. Again, all versions of GLAP have a relatively linear growth when the transmission range is varied. GLAP and iGLAP max out average delay around 10.0ms at 300 meters in range while PGLAP maxes out around 16.7ms at a 300 meter range. As before the average delay growth depicted in Figure 8 is due to group leaders causing a bottleneck in data flow. Again, this is more in PGLAP due to large group sizes and thus more queuing delays with increased transmission range.

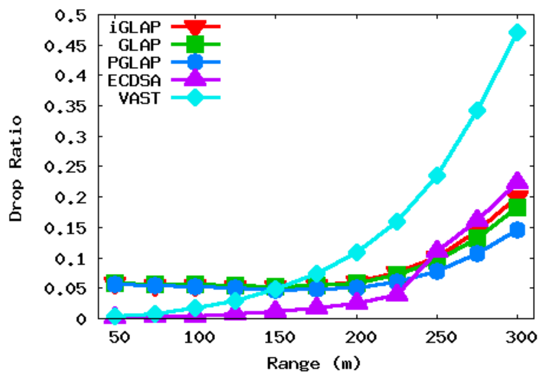
We see, from Figure 8b the ECDSA algorithm has a relatively constant delay of about 14.0ms. Although the increased number of nodes should increase queuing delay, since more packets are dropped with the increased range they effectively cancel each other and thus the delay stays relatively constant.

7.2.2 Packet Drop Ratio

Figure 9a shows the drop ratio of packets within the simulation given the various node densities while Figure 9b shows the drop ratio given various transmission ranges. As previously mentioned, the packets checked for dropping were all packets besides routing related packets.



(a) Average drop ratio vs Node count



(b) Average drop ratio vs Transmission range

Figure 9. Packet drop percentages

We see from Figure 9a that all algorithms dropped relatively few packets, that is less than 10%. All versions of GLAP had relatively similar packet drop ratios, with PGLAP being slightly higher. This is attributed to the utilization of asymmetric-key cryptography as it takes slightly longer to process the packets than that of symmetric-key cryptography. Additionally, there will be more control packets as an extra message has to be utilized to transfer grouping information between two nodes. Packet drop reasons were primarily collisions with several retry count expirations as well.

ECDSA had the overall lowest drop ratio with only dropping, at most, about 4.5% of packets. The primary cause of packet loss in this scheme was deteremined to be due to collision over the wireless channel. Due to the nature of how the simulation is ran, nodes are staggard started, and the fact this implementation of ECDSA only has a node broadcast every approximate $\frac{1}{3}$ of a second, it is quite probable the majority of nodes began falling into communication synchronization in a round robin sort of fashion. This notion is further backed by the relatively constant delays seen in Figure 7 and Figure 8 and the throughput experienced, which will be discussed shortly. Additionally, ECDSA has the slowest overall packet transmission rate of all protocols, that is packets are only sent every approximate $\frac{1}{3}$ of a second as opposed to VAST's every 100ms for example.

VAST initially had a relatively low drop ratio, however this changed as the drop ratio grew at an almost exponential rate to start with. This growth rate however tapered off at around the 130 node count and VAST maxed out at around 9.4%. Similar to ECDSA, most all packet drops were due to collisions. As VAST transmits every 100ms, it has the fastest overall transmission rate out of all of the protocols. This in turn increases congestion and is a key factor in the quickly rising drop ratio.

From Figure 9b we again see similar drop rates among the GLAP algorithms. All had a relatively constant drop ratio until a transmission range of approximately 175 meters was reached. After such time, the drop ratios did begin to increase faster. Of interest, PGLAP had the lowest drop ratio, about 16.4%, at the farthest transmission range of 300 meters. One possible explanation for this is the fact that extra steps are needed for joining a group and as such it is possible less nodes are joining groups and remaining as temporary unassociated nodes if the joining messages get dropped due to increased collision at higher ranges. Should the node stay ungrouped, only search messages S will still be broadcast, which occur every 900ms. This notion is further backed by the slowly growing throughput seen in Figure 10.

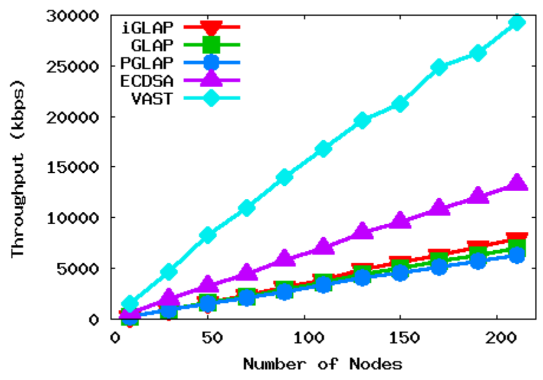
ECDSA began relatively well, but at approximately 225 meters the drop ratio sharply increases and maintains a steep linear growth rate. VAST performed quite bad when varying transmission range with regard to drop ratios. It in fact has a relatively exponential growth factor as at a transmission range of 100 meters only about 1.7% of packets where dropped, however at 300 meters over 47% of packets were lost.

From Figure 7 we can see merely increasing node density does not have as dire of an impact on this primarily due to the sparsity of nodes when taking the entire simulation setup into account. However, Figure 8 shows quite a different story. Increasing the transmission range has a drastic impact over the area even with only 130 nodes in the simulation. When thinking of this in the sense of a graph, an increase in transmission range will increase the connectivity of all nodes throughout the entire simulation, while an increase in node count will only increase the connectivity of a few nodes intermittently during the simulation.

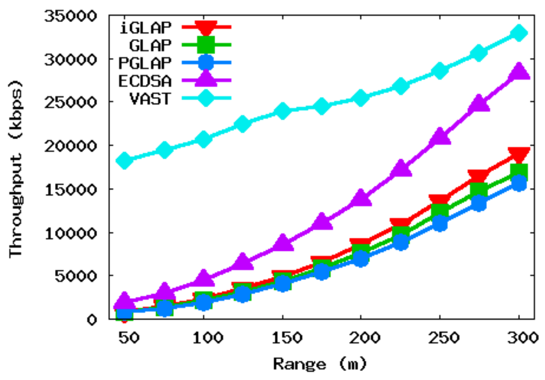
7.2.3 Message Throughput

Figure 10a shows the average throughput of message packets within the simulation given the various node densities while Figure 10b shows the average throughput of message packets given various transmission ranges. Throughput does not count any control messages.

From both Figure 10a and Figure 10b we see that VAST and ECDSA have the



(a) Average throughput vs Node count



(b) Average throughput vs Transmission range

Figure 10. Average message throughput in kilobits per second (kbps)

highest overall throughput. This is due in part as all messages in VAST and ECDSA are considered data messages in regards to our simulation test, as there is no grouping in either. Additionally, all packets in ECDSA contain not only the data, but also the signature on the data as well as the signer's certificate for usage in verifying the signature. VAST is similar and both protocols data packet sizes are over 30 bytes larger. This coupled with the fact that VAST sends messages every 100ms and ECDSA every approximate $\frac{1}{3}$ of a second help in showing the somewhat drastic difference in throughput between the protocols.

Though all versions of GLAP have a generally linear growth factor in the graphs of Figure 10, iGLAP has the highest throughput rate. This is as expected though as iGLAP does not waste time with any encryption or verification of HMAC. As GLAP utilizes symmetric key encryption it is only slightly lower in throughput compared to iGLAP. PGLAP performs the worst, as expected, out of the GLAP versions. This is primarily due to the delay incurred from control messages utilizing an asymmetric-key cryptographic scheme. The aforementioned issue with nodes being able to join groups properly falls under this category.

CHAPTER 8
CONCLUSION

Herein, we identified the need for a secure, flexible, and infrastructure independent solution for VANET communication which can be deployed anywhere at anytime. We proposed an efficient and group-based lightweight authentication protocol, GLAP, and its extension, PGLAP, to meet these requirements by exploiting the group-based behavior of vehicles in traffic. Our scheme maintains the privacy of an individual while allowing for non-repudiation of a message by law enforcement if so needed. Additionally, it allows for a high degree of DoS and replay resistance. As the use of symmetric-key cryptography is proposed in GLAP, we have also identified several ways to maintain key freshness, a desirable property. In addition, when VANET deployment matures and PKI support is enabled, GLAP can easily be extended to utilize public-key cryptography as described with PGLAP.

As PGLAP is based on GLAP, it reaps the majority of the benefits attributed to GLAP. PGLAP is efficient in data communication as it utilizes symmetric-key cryptography yet it solves GLAP's reliance on two universal keys by utilizing PKC for group construction. As such PGLAP requires some infrastructure support to function, but is quite flexible in that should PKI support be sparse PGLAP is still able to maintain functionality. The utilization of PKC additionally allows for other vehicle's credentials to be validated before communication is allowed thus aiding in preventing malicious communications from traversing the network.

The performance of both GLAP and PGLAP have been tested under ns-2 using real-life traffic data. Simulation results have shown that they prove to be fast and efficient protocols with all message delays being less than 14ms within our simulation environment and a low drop ratio for the majority of scenarios tested.

The results also have shown that for the optimal performance in terms of both delay and drop ratio, smaller groups (i.e. smaller transmission ranges such as 50m) should be preferred. Compared to ECDSA and VAST, GLAP and PGLAP can significantly reduce the end-to-end delay for packets while providing the desired authentication with privacy and non-repudiation.

This thesis can be further extended in the following ways. First, both schemes, GLAP and PGLAP, could further improve efficiency by allowing the transmission range to be dynamically adjusted depending on traffic conditions. Second, efficiency may also be improved by analyzing, more in-depth, the effect various transmission times have on the protocols. Additionally, a jitter factor could be added in such that nodes only transmit new messages at approximately $300 \pm .5$ms rather than every 300ms, for example. Third, distributed trust can be investigated to cope with internal misbehaving nodes (i.e. broken or cheating sensors, black holing data). Basic groundwork has already been explored in adding the concept of trust to GLAP. Integrating trust into our schemes would allow for isolation of misbehaving nodes, regardless of intention. That is should a node's sensors be faulty the node could be isolated so the data it is sending is not propagated throughout the network. The concept of trust within a VANET has received relatively little attention, with less attention even more so on a trust scheme functioning with an authentication scheme.

REFERENCES

[1] “Speed Comparison of Popular Crypto Algorithms.” Available: `http://www.cryptopp.com/benchmarks.html`.

[2] A. Studer, F. Bai, B. Bellur, and A. Perrig, “Flexible, Extensible, and Efficient VANET Authentication,” *Journal of Communications and Networks*, vol. 11, pp. 574–588, December 2009.

[3] X. Lin, X. Sun, P.-H. Ho, and X. Shen, “GSIS: A Secure and Privacy-Preserving Protocol for Vehicular Communications,” *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 6, pp. 3442–3456, 2007.

[4] Y. Xi, K.-W. Sha, W.-S. Shi, L. Schwiebert, and T. Zhang, “Probabilistic adaptive anonymous authentication in vehicular networks,” *J. Comput. Sci. Technol.*, vol. 23, no. 6, pp. 916–928, 2008.

[5] L. Fischer, A. Aijaz, C. Eckert, and D. Vogt, “Secure revocable anonymous authenticated inter-vehicle communication (. . . ,” *In proceedings of the Workshop on Embedded Security in Cars (ESCAR)*, 2006.

[6] “IEEE Trial-Use Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages,” *IEEE Std 1609.2-2006*, pp. 0_1 –105, 2006.

[7] “IEEE 802.11p Standard for VANETs.” Available: `http://grouper.ieee.org/groups/802/11/Reports/tgp_update.htm`, 2009.

[8] *Ad Hoc Wireless Networks: Protocols and Systems.* Prentice Hall Publishers, 2001.

[9] F. Li and Y. Wang, “Routing in vehicular ad hoc networks: A survey,” *Vehicular Technology Magazine, IEEE*, vol. 2, pp. 12–22, June 2007.

[10] M. Raya and J.-P. Hubaux, “The security of vehicular ad hoc networks,” in

SASN '05: Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks, (New York, NY, USA), pp. 11–21, ACM, 2005.

[11] H.-J. Reumerman, M. Roggero, and M. Ruffini, "The application-based clustering concept and requirements for intervehicle networks," *Communications Magazine, IEEE*, vol. 43, pp. 108–113, April 2005.

[12] J. J. Blum, A. Eskandarian, and L. J. Hoffman, "Challenges of intervehicle ad hoc networks," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, no. 4, pp. 347–351, 2004.

[13] M. Raya and J.-P. Hubaux, "Securing vehicular ad hoc networks," *Journal of Computer Security*, vol. 15, no. 1, pp. 39–68, 2007.

[14] X. Lin, R. Lu, C. Zhang, H. Zhu, P.-H. Ho, and X. Shen, "Security in vehicular ad hoc networks," *Communications Magazine, IEEE*, vol. 46, pp. 88–95, April 2008.

[15] C. Zhang, X. Lin, R. Lu, P.-H. Ho, and X. Shen, "An Efficient Message Authentication Scheme for Vehicular Communications," *Vehicular Technology, IEEE Transactions on*, vol. 57, no. 6, pp. 3357–3368, 2008.

[16] D. Boneh and H. Shacham, "Group signatures with verifier-local revocation," in *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 168–177, ACM, 2004.

[17] A. Wasef and X. Shen, "PPGCV: Privacy Preserving Group Communications Protocol for Vehicular Ad Hoc Networks," in *Communications, 2008. ICC '08. IEEE International Conference on*, pp. 1458–1463, May 2008.

[18] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA broadcast authentication protocol," *RSA CryptoBytes*, 2002.

[19] P. Tsang and S. Smith, "PPAA: Peer-to-peer anonymous authentication," *Applied Cryptography and Network Security*, 2008.

[20] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy, "Efficient and

robust pseudonymous authentication in VANET," in *VANET '07: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 19–28, ACM, 2007.

[21] P. Kamat, A. Baliga, and W. Trappe, "An identity-based security framework For VANETs," in *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 94–95, ACM, 2006.

[22] A. Studer, E. Shi, F. Bai, and A. Perrig, "{TACK}ing Together Efficient Authentication, Revocation, and Privacy in VANETs," in *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON 2009)*, 2009.

[23] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*, (New York, NY, USA), pp. 47–53, Springer-Verlag New York, Inc., 1985.

[24] A. Studer, M. Luk, and A. Perrig, "Efficient Mechanisms to Provide Convoy Member and Vehicle Sequence Authentication in {VANETs}," in *Proceedings of the International Conference on Security and Privacy in Communication Networks ({SecureComm})*, 2007.

[25] R. Anderson, M. Bond, J. Clulow, and S. Skorobogatov, "Cryptographic Processors-A Survey," *Proceedings of the IEEE*, vol. 94, no. 2, pp. 357–369, 2006.

[26] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in VANETs," in *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 29–37, ACM, 2004.

[27] R. Canetti, "HMAC: keyed-hashing for message authentication," *RFC*, 1997.

[28] "Advanced Encryption Standard (AES)," 2001.

[29] A. Menezes, P. V. Oorschot, and S. Vanstone, "Handbook of applied

cryptography," 1997.

[30] D. E. Denning and D. K. Branstad, "A taxonomy for key escrow encryption systems," *Commun. ACM*, vol. 39, no. 3, pp. 34–40, 1996.

[31] "The Network Simulator - ns-2." Available: `http://www.isi.edu/nsnam/ns/`.

[32] "Modelling mobility for mobile ad hoc networks." Available: `http://www.cs.rice.edu/~amsaha/Research/MobilityModel/`.

VITA

Graduate School
Southern Illinois University

MARSHALL K RILEY   Date of Birth: April 23, 1981

1200 EAST GRAND AVENUE, APARTMENT 6-3 B, CARBONDALE, ILLINOIS 62901

13150 REED CEMETERY ROAD, MARION, ILLINOIS 62959
mkriley99@yahoo.com

Southern Illinois University at Carbondale
Bachelor of Science, Computer Science and Mathematics, May 2008

Thesis Title:
Group-Based Authentication Mechanisms for Vehicular Ad-Hoc Networks

Major Professor: Dr. K. Akkaya

Publications:

B. Woodward, M. Whitney, M. Riley, and K. Mei, *Virtualization Labs for Information Security*, Cengage Learning, 2010, 2010.

M. Wainer, H. Hexmoor, and M. Riley, "Tablet PCs and robots: Technology as platform and motivator for explorations in collaboration," *2007 International Symposium on Collaborative Technologies and Systems*, 2007, pp. 115-121.