Online Authentication with Encryption and Anonymous Authentication in Vehicular Ad-hoc Networks

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Technology
in
Computer Science and Engineering
by

Umang Jain (03CS3005)

Under the guidance of
Prof. Dipanwita Roy Chowdhury

Department of Computer Science and Engineering
Indian Institute of Technology
Kharagpur – 721302
May 2008

Department of Computer Science and Engineering
IIT Kharagpur

CERTIFICATE

This is to certify that the thesis entitled, '**Online Authentication with Encryption and Anonymous Authentication in VANET**' submitted by **Umang Jain** (03CS3005) in partial fulfillment of the requirements for the award of degree of **Masters of Technology** in Computer Science and Engineering, to the Indian Institute of Technology, Kharagpur, is a bonafide work carried out by him under my supervision and guidance. The thesis fulfills the requirements relating to the nature and standard of work prescribed for the award of the above mentioned degree. The results embodied in this thesis have not been submitted elsewhere to any other University or Institute for the award of any other degree or diploma.

Prof. D. Roychowdhury
Dept. of Computer Science and Engg.
Indian Institute of Technology
Kharagpur -721302, India
May 2008

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere gratitude towards my project supervisor Prof. Dipanwita Roy Chowdhury who has been a guide, a teacher and a role model for the past two and a half years. Working on the project was a great learning experience under her constant encouragement and support. I would also like to thank Prof. Abhijit Das for his constructive feedback for the project. I am also thankful to all the faculty members for their cooperation and for all that I learnt from them.

I also owe my gratitude my friend and colleague Nitin Bansal for motivating me throughout and to my friend Arijit Ghosh for his feedback in important stages of the project. I am also thankful to my classmates for their company for five years and all friends who have directly or indirectly assisted me in my endeavors.

Umang Jain

Abstract

Authentication *is the act of establishing or confirming something (or someone) as authentic, that is, that claims made by or about the thing are true. Authenticating an object may mean confirming its originality whereas authenticating a person often consists of verifying their identity. In the ranks of cryptography verifying the authenticity of data or an entity is an unavoidable, indispensable task. Only the mere appreciation of importance of authentication in cryptography gives one enough motivation to put thorough efforts in this field. In the work documented in this thesis, the problem of Authentication in two different scenarios has been undertaken. The first problem is carrying out data authentication together with encryption when the later has to be accomplished in an online manner. In this regard a framework for online Authentication with encryption has been proposed along with a flexible keyed-hash function based on Cellular Automata.*

In the later part, the problem of entity authentication accompanied with the requirement of anonymity in Vehicular Ad-hoc Networks (VANET) has been considered. Here we have proposed a group signature algorithm based on Chinese Remainder Theorem.

Table of Contents

4* GSCRT: A Group Signature Scheme

5* Modified GSCRT

*** The work documented in these chapters has been done jointly with Nitin Bansal (03CS3015)**

List of figures:

List of tables:

Chapter 1

Introduction: Encryption with Authentication

1.1 Authenticated Encryption (AE): An introduction

Often when two parties communicate over a network, they have two main security goals: privacy and authentication. In fact, there is compelling evidence that one should never use encryption without also providing authentication. Many solutions for the privacy and authentication problems have existed for decades, and the traditional approach to solving both simultaneously has been to combine them in a straightforward manner using so-called "generic composition." However, recently there have been a number of new constructions which achieve both privacy and authenticity simultaneously, often much faster than any solution which uses generic composition. This approach to achieving both privacy and integrity is the so-called "Authenticated Encryption" problem.

An AE scheme has two goals: privacy and authenticity. **Privacy** means, intuitively, that a passive adversary who views the ciphertext *C* and the nonce *N*, cannot "understand" the content of the message *M*. One way to achieve this is to make *C* indistinguishable from random bits, and indeed this is one definition of security for an encryption scheme that is sometimes used, although it is quite a strong one. **Authenticity** means that an adversary cannot modify a message such that the receiver is not able to detect it. With the help of an authentication tag the receiver is able to verify of whether the message received by it exactly the same message that was sent by the sender.

Although AE did not get a formal definition until recently [1], the goal has certainly been implicit for decades. The traditional way of achieving both authenticity and privacy was to simply find an algorithm which yields each one and then use the combination of these two algorithms on our message. Intuitively it seems that this approach is obvious, straightforward, and completely safe. Unfortunately, there are many pitfalls accidentally "discovered" by well-meaning protocol designers. One commonly-made mistake is the assumption that AE can be achieved by using a non-cryptographic non-keyed hash function h and a good encryption scheme like CBC mode (Cipher Block Chaining mode; see CBC-MAC and variants) with key K and initialization vector N. One produces CBC$_{K,N}(M, h(M))$ and hopes this yields a secure AE scheme. However, these schemes are virtually always broken. Perhaps the best-known example is the Wired Equivalent Privacy protocol (WEP) used with 802.11 wireless networks. This protocol instantiates h as a Cyclic Redundancy Code (CRC) and then uses a stream cipher to encrypt. Borisov, Goldberg, and Wagner showed, among other things, that it was easy to circumvent the authentication mechanism. Another common pitfall is "key reuse". In other words, using some key K both for the encryption scheme and the MAC scheme. This approach applied blindly almost always fails. We will later see that all the secure "combined modes," do in fact use a single key, but they are carefully designed to retain security in spite of this. It is now clear to researchers that one needs to use a *keyed* hash (i.e., a MAC) with some appropriate key $K1$ along with a secure encryption scheme with an independent key $K2$. However, it is unclear in what order these modes should be applied to a message M in order to achieve authenticated encryption. There are three obvious choices:

• **MtE:** MAC-then-Encrypt. We first MAC M under key $K1$ to yield tag σ and then encrypt the resulting pair (M, σ) under key $K2$.

• **EtM:** Encrypt-then-MAC. We first encrypt M under key $K2$ to yield ciphertext C and then compute $\sigma \leftarrow \text{MAC}K1(C)$ to yield the pair (C, σ).

• **E&M:** Encrypt-and-MAC. We first encrypt M under key $K2$ to yield ciphertext C and then compute $\sigma \leftarrow \text{MAC}K1(M)$ to yield the pair (C, σ).

Also note that decryption and verification are straightforward for each approach above: for MtE decrypt first, then verify. For EtM and E&M verify first, then decrypt. Results show that if the MAC has a property called "strongly unforgeable", then it possible to achieve the strongest definition of security for AE only via the EtM approach. They further show that some known-good encryption schemes fail to provide privacy in the AE setting when using the E&M approach, and fail to provide a slightly stronger notion of privacy with the MtE approach.

In modes that employ an underlying block cipher quite often use the encryption function both for providing authenticity. If separate calls to the encryption function are made for privacy and authenticity, then we call it a "two-pass mode", otherwise it's called a "single pass mode".

A generic composition is one where Encryption and Authentication are employed separately without making any computational gains by making certain operations common to both. A scheme that does make such a gain is said to be a *combined mode scheme.*

Several combined AE schemes have been formulated most of which are based on block cipher and mostly utilize symmetric key cryptography. The first such scheme introduced by Jutla was called IAPM. This and OCB were single pass modes i.e. they did not use extra calls of Encryption function for authentication. But these are patented schemes. The other schemes like CCM, EAX, CWC are two pass schemes, i.e. they incur extra overhead in form of extra calls to the

Encryption Function for achieving integrity. Apart form these there are certain primitives like SNOW and HELIX that are based on stream cipher. They are fast in hardware and software on account of being based on stream ciphers. But they cannot be proved secure very much like any other primitive. For example HELIX even after a long time of its formulation still isn't secure enough to be used. There were defects found after differential cryptanalysis of HELIX but it cannot be said to be secure until more attempts are made of break its security. This is fact that is even accepted by the authors in their paper [2].

A summary of the above mentioned schemes is presented in the Table 1.1.

Scheme	#Passes	Provably Secure	Assoc Data	Parallelizable	On-line	Patent-Free
IAPM	1	✓		✓	✓	
XECB	1	✓		✓	✓	
OCB	1	✓		✓	✓	
CCM	2	✓	✓			✓
EAX	2	✓	✓		✓	✓
CWC	2	✓	✓	✓	✓	✓
Helix	1		✓		✓	✓
SOBER-128	1		✓		✓	✓

Table 1.1: A comparison of AE schemes

There are several applications where there is need for online data, mainly because of the need for online processing of data. For example internet based applications may have such a requirement or maybe some time-critical applications. Most the data authentication algorithms (part of Authenticated Encryptions schemes or otherwise) compute the messaged digests or MAC's based on the complete message. Authentication while using such algorithms for online data communication really makes no sense at all. The receiver will have to wait for the complete message to arrive to verify its authenticity and therefore cannot really use the data with surity of correctness. Many AE schemes claim to

be online in nature but in reality it is only the encryption that is online in true sense.

In this thesis we have proposed a solution for this problem of carrying out authentication and encryption together in online manner. Our solution is not an Authenticated Encryption one in reality and is more of just generic composition. In the following chapter we propose a framework for online AE followed by a proposal of hash function based on cellular automata compatible with the framework.

1.2 An overview of Cellular Automata

A **cellular automaton** is a discrete model studied in computability theory, mathematics, and theoretical biology. It consists of an infinite, regular grid of *cells*, each in one of a finite number of *states*. The grid can be in any finite number of dimensions. Time is also discrete, and the state of a cell at time *t* is a function of the states of a finite number of cells (called its *neighborhood*) at time *t-1*. These neighbors are a selection of cells relative to the specified cell, and do not change. (Though the cell itself may be in its neighborhood, it is not usually considered a neighbor.) Every cell has the same rule for updating, based on the values in this neighborhood. Each time the rules are applied to the whole grid a new *generation* is created.

A one dimensional binary cellular automaton (CA) consists of a linearly connected array of L cells each of which takes the value 0 or 1 and a Boolean function f(x) with q variables. The value of the cell x_i is updated in parallel (synchronously) using this function in discrete time steps as $x_i = f(x)$ for i = 0, 1, 2…..L. The parameter q is usually an odd integer, i.e. $q = 2r + 1$ where r is often named the radius of the function f(x): the new value of the i^{th} cell is calculated

using the value of the i^{th} cell and the values of r neighboring cells to the right and left of the i^{th} cell.

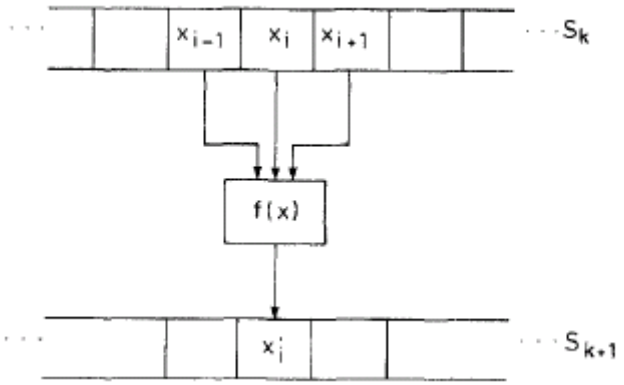

Fig 1.1: 1D Cellular Automata

For example, in a 1-dimensional cellular automaton, the neighborhood of a cell x_i^t, where t is the time step (vertical), and i is the index (horizontal) in one generation—is $\{x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1}\}$. There will obviously be problems when a neighborhood on a left border references its upper left cell, which is not in the cellular space, as part of its neighbor. The boundary conditions are handled in the following way:

Null boundary CA: A CA is said to be null boundary CA if the left (right) neighbor of the leftmost (rightmost) terminal cell is connected to logic 0 state. It means that say there is a cell that all of whose neighbors are not some legitimate cells. Then the remaining neighbors are taken as 0.

Periodic Boundary CA: A CA is said to be Periodic Boundary CA if the extreme cells are adjacent to each other. It can be of a flattened out cylinder.

Rule of a CA: If the next state function of a cell is expressed in the form of a truth table, then the decimal equivalent of the output is called the rule number of the cellular automaton.

Additive and Non Additive Cellular Automata: If the rule of a CA cell involves only XOR logic ,then it is called a linear rule .A CA with all the cells having linear rule is called a linear CA. Rules involving XNOR logic are referred to as complemented rules. A CA having a combination of XOR and XNOR rules is called an additive CA. *The rules with AND-OR logic are non-additive rules or non-linear rules.*

Uniform and Hybrid Cellular Automata*:* If all the CA cells obey the same rule then the CA is said to be uniform CA, otherwise it is a hybrid CA

Reversible CA: A CA is said to be *reversible* if for every current configuration of the CA there is exactly one past configuration (pre-image). If one thinks of a CA as a function mapping configurations to configurations, reversibility implies that this function is bijective.

Programmable CA (PCA): Positional representations of Rule 90 and Rule 150 show that their neighborhood dependence differ in only one position, viz., on the cell itself. Therefore, by allowing a single control line per cell, one can apply both Rule 90 and Rule 150 on the same cell at different time steps. Thereby, an L cell CA structure can be used for implementing LCA configurations. Realizing different CA configurations (cell updating rules) on the same structure can be achieved using control logic to control the appropriate switches and a control program stored in ROM can be employed to activate the control. The 1(0) state of the i_{th} bit of a ROM word closes (opens) the switch that controls the i_{th} cell. Such a structure is referred as to as a programmable cellular automaton (PCA) Accordingly, allowing one control input per cell that configures the updating

rule, we can apply to that cell, either Rule 90 or Rule 150 The n-bits control word for an n cells PCA has 0(1) on the i_{th} cell if Rule 90(150) is applied to the i_{th} cell.

Linear and Non-linear CA: If the update rule of the cellular automata consists of only linear rules, then it is called linear CA. On the other hand if the rules are non-linear it is called a Non-linear CA. An example of a non-linear rule is rule 30 which can be represented as a= b XOR(c OR d).

Cryptographic applications of Cellular Automata: Cellular Automata have simple and modular structures and are fairly easily implementable. In spite of this they are known to produce complex patterns are believed to have excellent randomness properties. Also it is not easy to reconstruct or reverse 1 dimensional Cellular Automata states, for 2 dimensional the problem whether a rule can be inverted in undecidable. These properties certainly project cellular automata as a system that can be successfully used to develop ciphers as well as authentication mechanisms. Few cellular automata block and stream ciphers as well as hash functions have been proposed in the recent past. Examples are cellular automata based hash functions are in [3], [4] and [5].

Now with a background on Authenticated Encryption and Cellular Automata we now proceed to propose an Online Authenticated Encryption Scheme.

Chapter 2

Online Authentication with Encryption

2.1 A framework for online Authentication and Encryption

In this section we propose a protocol for online Authentication with Encryption. The protocol assumes an underlying block cipher and a hash function. Following the protocol we propose a hash function based on cellular automata which is flexible in terms of key size and the digest size.

2.1.1 The Proposed Framework

The actual protocol for message for online authenticated encryption is outlined below.

- Data is divided into blocks of 128 bits each.
- Encryption is block is done by using a block cipher with block size 128 bits.
- Blocks of messages (128 bit each) are encrypted and the cipher-text is sent to the receiver
- An authentication tag or hash value is computed for a group of 8 blocks
- The hash value is also 128 bit in size
- After 8 message blocks are sent , the hash value is encrypted and sent
- The receiver on receiving message blocks performs similar operations to compute the hash value
- The receiver on gets the encrypted tags after 8 consecutive encrypted message blocks

- Computed and decrypted hash values are compared to verify the integrity of data

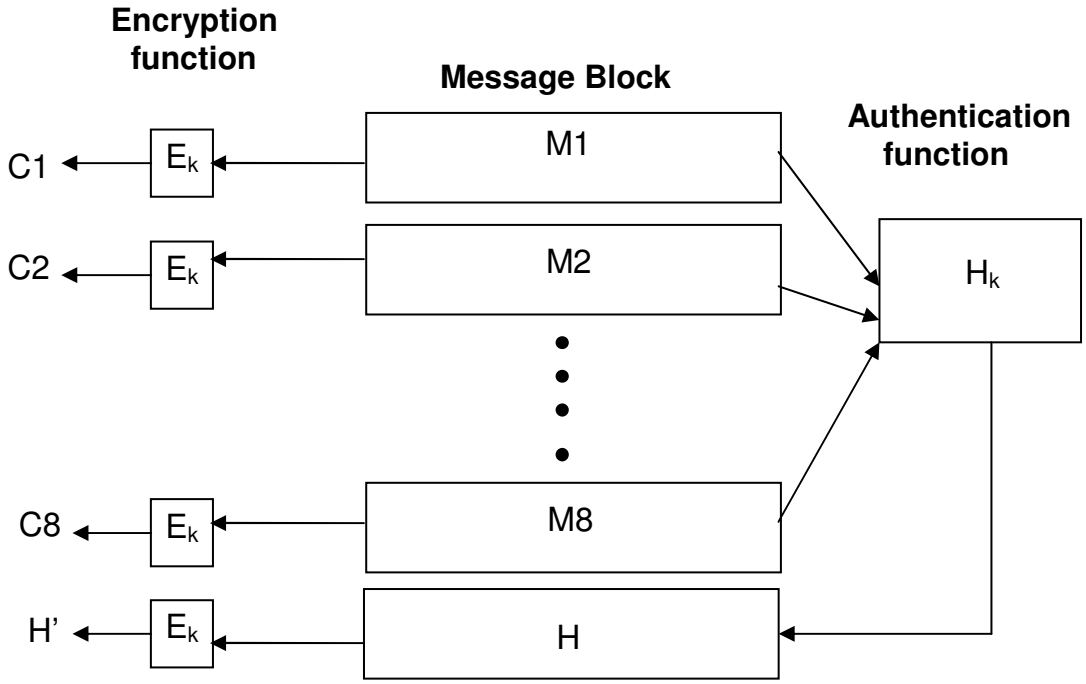

Fig 2.1: AE scheme (block diagram)

2.1.2 Design Rationale

- In the above mentioned scheme, the receiver does have to wait for the arrival of the complete message of compute the hash. As soon as the encrypted hash is received the integrity of the previously received eight data blocks can be verified.
- Also the hash and the cipher-text are sent in identical fashion. This makes life simpler for the receiver and difficult for the advisory. The portion of

the message that was corrupted can be isolated from those that did not encounter any error.

- The receiver needs to discard only the group of those 8 blocks for which the corruption has been detected. The sender also needs to send only those eight blocks thus saving time and resources that were to be incurred in sending the complete message.
- In certain applications the message cannot be used in discrete chunks and the complete message is needed to proceed. If on an error, the receiver decides to discard the complete message, then it does not need to decrypt any message block after it has detected an error.

In the forthcoming sections we look at cellular based hash function that can be used in the above AE scheme along with any secure encryption algorithm. The hash function utilizes cellular automata rule 30 and properties of rule 30 are very critical as they influence to a great deal the performance of the proposed hash function. Therefore, we will proceed by analyzing rule 30 and its properties.

2.2 Rule 30 evolutions in cryptography

Cellular Automata Rule 30 is one of the most interesting amongst the 256 possible 3 neighborhood rules. Much has been written in the literature about the random-ness of rule 30, it use as a Pseudo Random Number Generation (PNRG) and its uses in cryptography. Rule 30 can be described as the following mapping on three bit numbers to provide a one bit result.

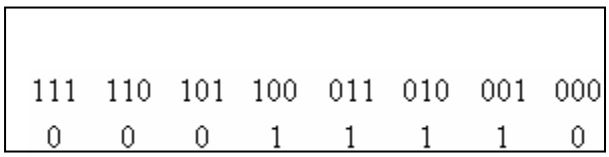

Fig 2.2: CA Rule 30

The properties of rule 30 that make it interesting for use in above mentioned application are its chaotic behavior (random-ness), irreversibility and non-linearity. In [6] the authors have studied in detail the random-ness of rule 30 against basic statistical tests and rule 30 passes them with good results. From these studies it can be concluded rule 30 have enough random-ness to be utilized in cryptographic applications like a hash function. The other property that has been under the scanner is the irreversibility of rule 30. In the following paragraphs we will look at the prospect of reversing the rule 30. It is of paramount importance because as the security of CHASH depends a lot on the irreversibility of rule 30. Also, irreversibility or one-way ness is very important for any hash function.

Global Reversibility of Rule 30: Rule 30 cannot be locally inverted, because (in general) more than one predecessor string leads to the same successor string. Thus, in a sense, information is lost by the application of the rule. The inverse of Rule 30 is "completely indeterminate" in the sense that no 3-bit pattern yields a unique antecedent for the central bit. We cannot find a mapping or rule which can be classified as a reverse of rule 30. However, these comments all refer to local reversibility. Local irreversibility means that given a bit, we cannot predict the unique antecedent for the central bit in the previous state.

If we consider global reversibility, especially in the context of a closed loop of cells, we find that Rule 30 actually is reversible. In other words, although the predecessor values in any range of cells cannot be inferred based on local

knowledge of the successor cells, it is possible to infer the entire predecessor string based on knowledge of the entire successor string.

To evaluate the distribution of (linear) predecessor strings over the n-bit successor strings, we have two options. We can consider n-bit predecessors or (n+2)-bit predecessors. For Rule 30, each n-bit string is the successor of exactly four (n+2) bit strings, and it is a possible successor of exactly three n-bit strings. To put this in another way, for each n-bit string, there are three possible n-bit predecessors (neglecting any boundary interference). For example, the possible 3-bit predecessors of each of the 3-bit strings under Rule 30 are listed below.

	predecessors		
000	000	110	111
001	000	101	110
010	010	011	100
011	001	010	100
100	000	110	111
101	000	101	110
110	010	011	100
111	001	010	100

Fig 2.3 Rule 30 predecessors

Referring to three consecutive cells as a "character", we might think that since a character in one part of the space has three possible predecessor characters, and a character in another region also has three possible predecessors, that there are 3 ◎ 3 = 9 possible combinations of predecessors for these two characters, but that is not the case, because the overall string extending from one character to the other has just three possible predecessors. Thus, for an infinite space with no boundaries, there is necessarily a correlation between distant predecessors

imposed by the global requirements of the rule, even though the predecessors are both indeterminate with respect to local information.

To see how this works, suppose the following 17-bit string has been produced by applying Rule 30 to some unspecified predecessor string:

...11010110100010110...

What can we infer about the predecessor string? Taking the left-most "character" (i.e., the bits 110), we know from the table above that the predecessor string must begin with 010, 011, or 100. The next (overlapping) character in the successor is 101, so the next character of the predecessor must be 000, 101, or 110, and this must overlap with two bits of the first character. Hence there are only three possibilities: 0101, 0110, or 1000, so these are the three possible predecessors of the first four bits. Continuing in this way, we can continue to add bits, and at each stage we will find that exactly three strings are viable as the predecessor. The entire process is summarized in the figure below.

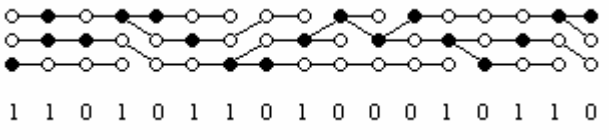

Fig 2.4 Rule 30 viable predecessors

Thus for rule 30 that it's possible for the predecessor string to be locally indeterminate and yet globally determinate.

Identifiable Structures in Rule 30 Evolutions: Some cellular automata rules have a property known as the toggle property. Cellular automata may be either left-toggle or right-toggle or both. A cellular automaton is a left-toggle cellular automaton if equation on flipping the left most bit of the neighborhood flips the

output. Rule 30 is left toggle rule. The toggle property has a clear signature in differential cryptanalysis. The single bit error in the plaintext propagates across the ciphertext, albeit only to the left.

Right Toggle rule-30: Rule 30 cab be described as $f(x_2) = x_1 \; XOR \; (X_2 \; OR \; X_3)$. The right toggle version is described as $f(x_2) = x_3 \; XOR \; (X_2 \; OR \; X_1)$.

2.3 CHASH: A cellular automata based hash function

The complex behavior and the complex patterns that CA's generate in spite of their fairly simple structure and update rules make them an interesting prospect for application in the field of cryptography. With, CHASH we try to tap these interesting properties of Cellular Automata in a hash function.

2.3.1 Overview

CHASH is keyed hash function. The input to the function is the plaintext. The plaintext is padded and divided into blocks of 1024 bits. A hash value is computed for each of the 1024 blocks. For this each of the 1024 bit block is further divided into sub-blocks and these sub-blocks are operated on by some CA rules.

The algorithm utilizes a CA rule generated on the fly using the key the CA rule 30. We proceed by analyzing the properties of rule 30 and then following it by proposing CHASH and its analysis.

2.3.2 CHASH Algorithm: The basic algorithm can be outlined as follows:

- Rule generation using the key
- Message is padded such that message is 64 modulo 128. For this a 1 followed by required number of 0's is added. Then length of the message encoded in 64 bits is added to make the length a multiple of 128 bits.
- Divide the message into 1024 bit blocks

- Divide each 1024 bit block into eight 128 bit sub-blocks.
- Run CA rules on the first 128 bit block to get intermediate hash.
- XOR the intermediate hash and next 128 bit block
- Continue the last two steps until the last block
- The eighth intermediate hash is the final hash for this 1024 bit block.
- For 128 blocks that are left over after dividing message into 1024 bit blocks, hash is computed separately.

2.3.3 Rule Generation: The rule generated by using the key is not any standard CA rule. It is rule with radius =3 or neighborhood =7. It means that each bit in the next state directly depends on 7 bits of previous state.

Now for a 7 neighborhood CA rule the rule-table must consist of $2^7 = 128$ entries. Each entry will either hold a 0 or 1. The rule table is generated using following steps:

- We generate a 256 entity by using the key. Let us call this the intermediate rule table. First 128 bits are same as 128 bits of the key. The remaining 128 bits are generated by negating the bits of the key.

  Intermediate rule table[i] = key[i], 0<= i <128

  = key[i] XOR 1, 128<= i <256

- This rule-table is state on CA rule 30 and on right-toggle version of CA rule 30 alternatively for n1 rounds (where n1=64) rounds. We then pick 128 bits from this 256 bit state to get the final rule-table.

Rule 30 is applied because it is non linear, has very good randomness properties. The method of generation of the intermediate rule-table ensures that the rule has nearly equal number of 0's and 1's. Both rule 30 and the modified rule 30 are applied for good diffusion and remove any patterns that might lead to discovery of the key.

2.3.4 One round of CHASH: One round of CHASH can be described as follows:

1. Perform modulo 2 addition of last round's output and the message block. If the block is the first message block add modulo 2 the initialization vector.
2. Transform the result of step 1 using the rule table for n2 cycles (where n2=20).
3. Transform the result of step 2 using the CA rule 30.
4. Transform the result of step 3 using the right toggle CA rule 30.
5. Repeat step 3 and 4 for n3 cycles (where n3 = 10).
6. Output the result of step 5.

2.3.4 Generation of the hash value

Each of the 128 bit sub-blocks of a 1024 bit block is operated on using one round of CHASH with chaining and the hash output of the eighth round is the hash value for this 1024 bit block.

2.3.5 Design Rationale

- *The rule generation for CHASH is constructed so as to foil brute-force attacks on the intermediate rule table; therefore it is of 256 bits.*
- *Both CA rule 30 and right toggle CA 30 are applied to annul the effect of propagation of difference only to the left, when rule 30 is applied.*
- *The unknown transformation based on the rule table is applied first before applying rule 30 so that even with a chosen plaintext attack it is not possible to guess the internal state of the CHASH round by reversing rule 30 application.*
- *Initialization vector is used to prevent any weakness due to weak messages like the all 0 message. Now it will not be possible to predict the output of an all 0 message.*

- *Total number of possible rules of 7 neighborhood is 128, so it is not possible to guess the rule that is generated using the key.*
- *Both the initial transformation and the last applied to a message is unknown to the attacker, this also lends strength to the hash function.*

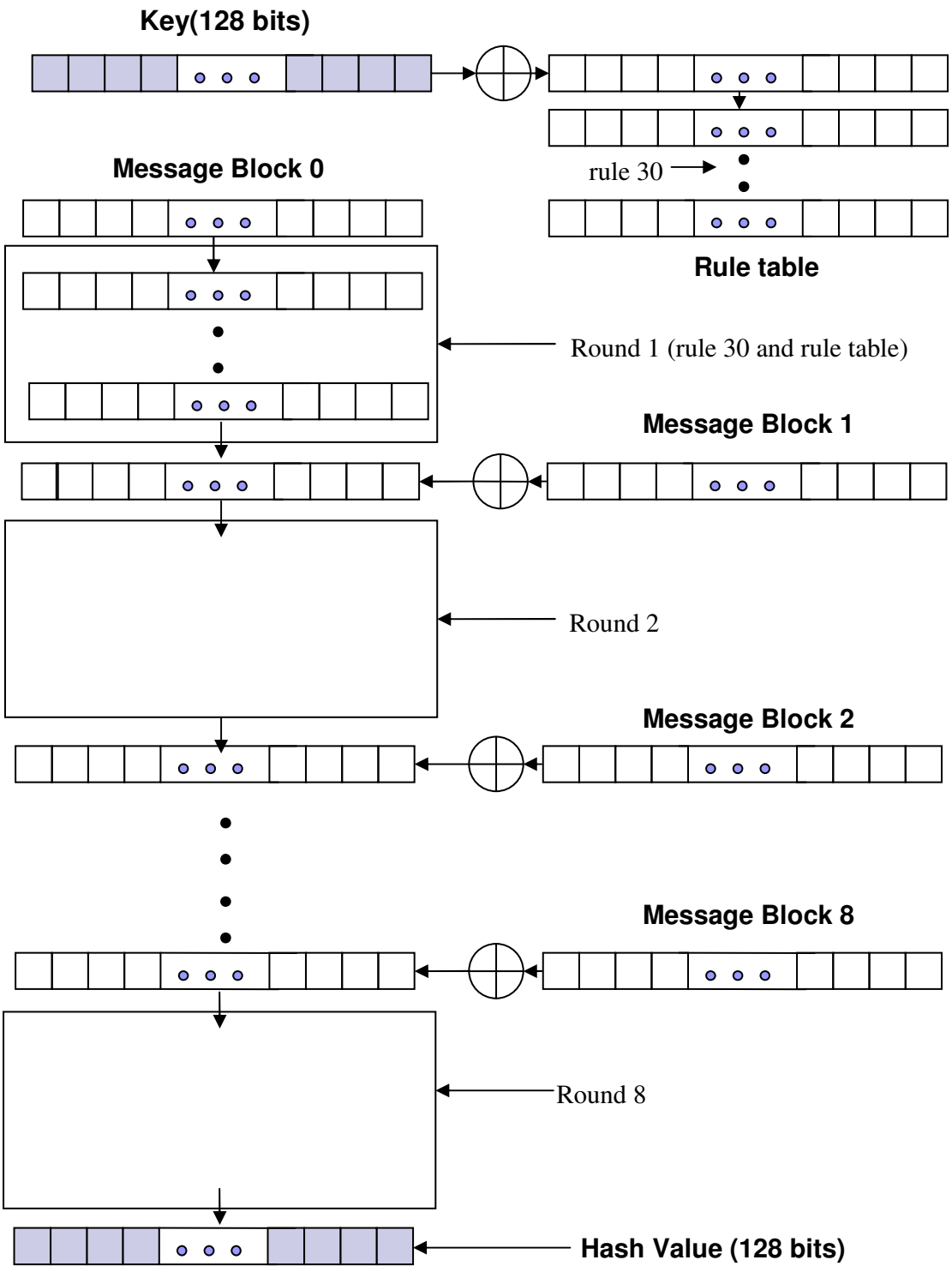

Fig 2.5: CHASH operation

2.4 Evaluation of CHASH

2.4.1 Bit Variance Test: The Bit variance test consists of measuring the impact of on the digest bits by changing input message bits. Bits of an input message are changed and the corresponding message digest bits (for each changed input) are calculated. Finally from all the digests produced, the probability (P_i) for each digest bit to take the value 0 or 1 is calculated. If $P_i(0) = P_i(1) = 0.5$ for all digest bits (0 to n) where n is the digest length, then the hash function is under consideration has attained the maximum performance in terms of bit variance test. It is difficult to check for all possible bit changes on the input. Therefore we only consider 1 bit changes. Since in the scheme proposed a hash is generated for every 1024 bit block separately we consider different randomly generated 1024 bit messages as input to the test. All possible 1 bit changes were applied on 10 different messages and for different keys. The results are tabulated below and looking at them we can safely say the authentication scheme passes the bit variance test.

P0	P1
0.498305	0.502673
0.501092	0.499886
0.498305	0.502673
0.501420	0.499557
0.496060	0.504918
0.499809	0.501168
0.496121	0.504857
0.493837	0.507141
0.494035	0.506942
0.499328	0.501650

Table 2.1 Bit Variance Test (1 bit)

2.4.2 Entropy Assessment Test: Entropy measures the amount of information contained in a message and it is maximum when it equals the total number of bits of the message. In out scheme, the digest if 128 bits long and it is infeasible to calculate the entropy in the absolute sense we employ an approximate method for the same. The Entropy is calculated by finding the occurrence of each 128 bit message digest from a set of digests as we cannot take the set of all possible 128 bit digests. We also cannot find the frequency of each 128 bit digests as it would be computationally infeasible. Therefore we use an approximate test for this purpose.

Approximate Entropy test: Let the message digest be composed of blocks where each block is equal to 1 byte. By taking all possible combinations of byte pairs a set of 16 bit numbers (0-65535) are obtained for each message digest. For a large number of message digests if the frequencies of occurrences of these numbers are equal, then the approximate entropy for the 16 bit sub-blocks of the message digest is equal to 16.

We conduct the approximate entropy test for 8,00,000 message digests. For each message digest we have 16*15 = 240 sixteen bit numbers. For the case of 8,00,000 messages, for the frequency of occurrence of each of these numbers to be same we must have each frequency = (240* 8,00,000)/65536 = 2929.6. As we see in the fig 4.1 the frequencies of these numbers are around the number 2929.

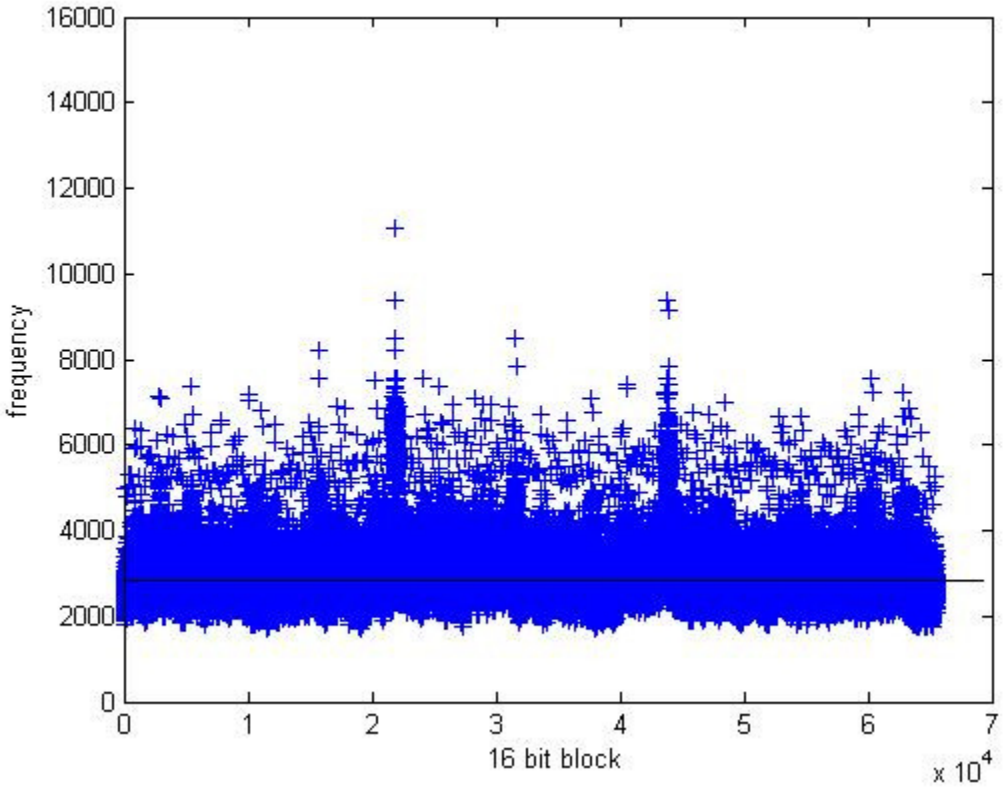

Fig: 2.6: Entropy Assessment

2.4.3 Measuring the diffusion: In this test the effect of changing one bit of input on the output is measured. In the ideal scenario the output difference should be half the total number of bits, which is 64 for our case. In this test, 10000 different randomly generated messages were taken and each of the 1024 bits was changed one by one. For each message the average output difference was calculated and these were plotted on a graph. The results can be seen in fig 3.3. All the output differences are around 64.

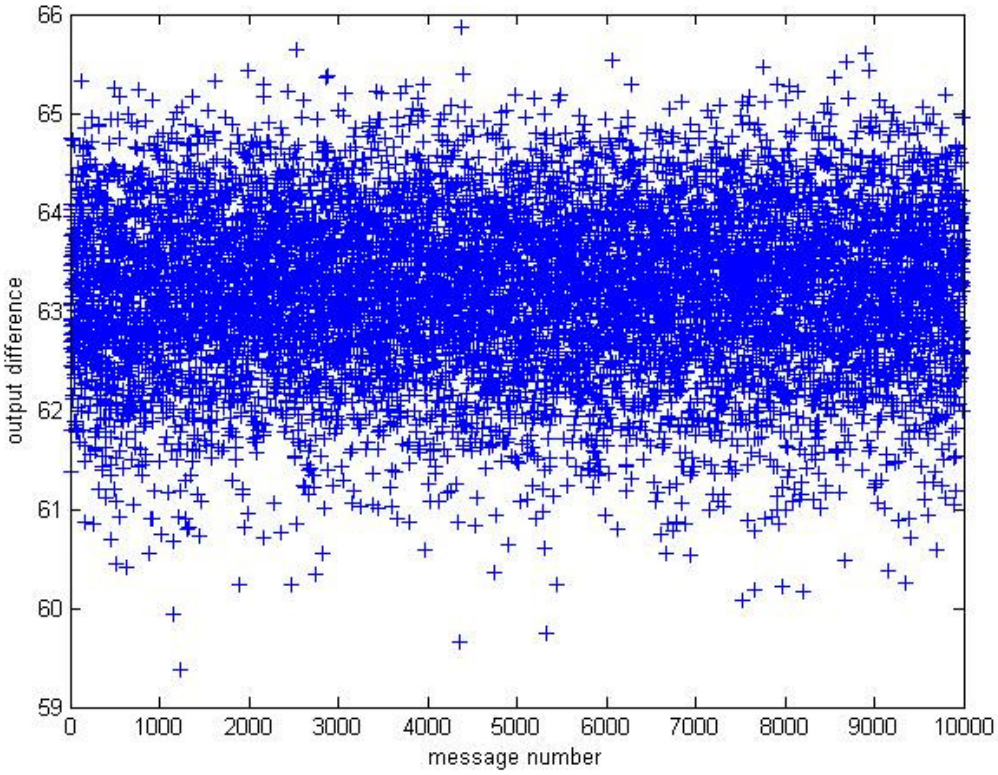

Fig 2.7: Diffusion in CHASH

2.4.4 Measuring the confusion: In this test the effect of changing one bit of the key on the output is measured. In the ideal scenario the output difference should be half the total number of bits, which is 64 for our case. In this test, 5000 different randomly generated keys were taken for a fixed message and each of the 128 bits was changed one by one. For each message the average output difference was calculated and these were plotted on a graph. The results can be seen in fig 3.4. All the output differences are around 64.

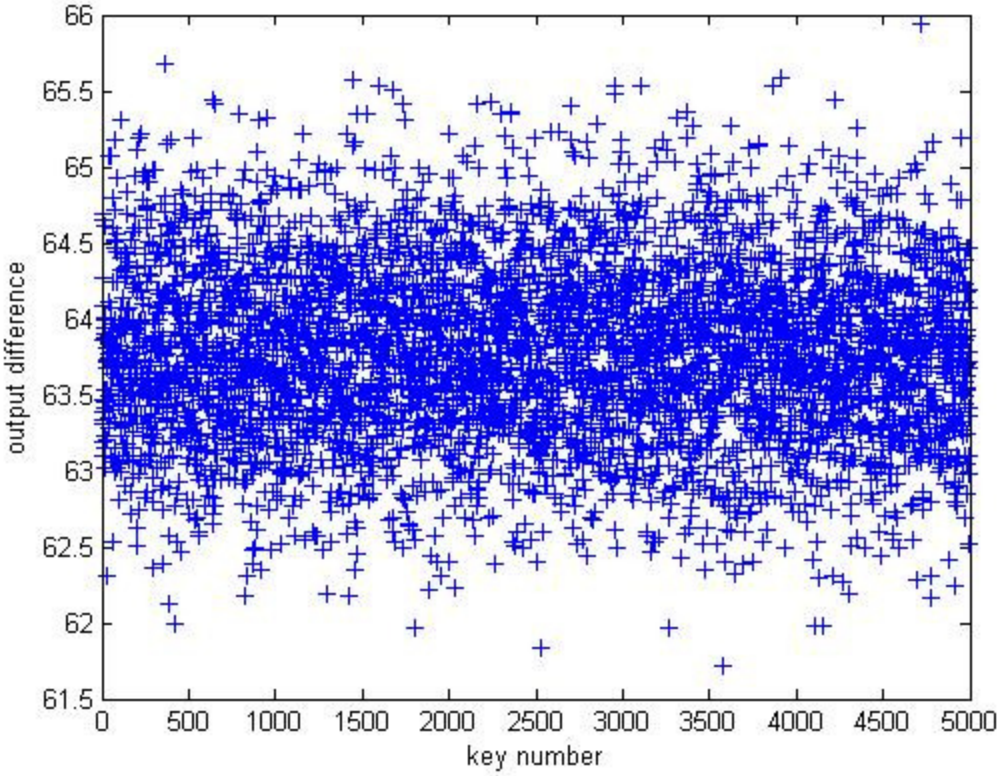

Fig 2.8: Confusion in CHASH

2.5 Irreversibility of CHASH

Irreversibility can be defined as: It is computationally hard to compute the input X of a hash function given its output. Then the function is secure against pre-image attacks. In CHASH the two main operations that are performed on the plaintext are applying rule 30 and the key generated rule for a number of cycles. As mentioned in section 2.2, rule 30 is not irreversible in true sense. In [7] also the authors have explored a method of inverting the states of cellular automata rules. But the application of the key-based rule in every round before application of

rule 30 masks this weakness of rule 30. After application of key-based rule the state in becomes unknown, therefore the seed on which rule 30 is applied is unknown making it impossible to iterate back. Thus we can conclude that as long as the key is unknown CHASH is one-way or irreversible.

2.6 Security Analysis

In this section we analyze CHASH for potential security flaws. As CHASH is not based on any other cryptographic primitive it is difficult to formally prove CHASH to be secure. But we can try and target weak points in the algorithm if any. In the following sub-sections we give an intuitive idea of resistance of CHASH against standard attacks.

2.6.1 Random Attack: In this attack, for a given hash value H, the attacker generates a number of messages and checks whether any of the generated messages produce digest H. The probability of success of this attack is $1/|R|$, where $|R|$ is number of elements in the range of hash function. Since here $|R|$ is 2^{128}, the probability of success of this attack is 2^{-128} which is negligibly low.

2.6.2 Birthday Attack: This attack is based on the birthday paradox. The paradox can be stated as follows: For a group of 23 people the probability that at least two of them have the same birthday is larger than 50%. In a group of r people, the probability that at least two people have the same birthday can be calculated as: $[Q= \prod_{i=0}^{r-1}(1- i/365)]$. Thus the probability that at least two people have the same birthday is P = 1-Q. The attacker selects N random messages and hopes to find a pair of messages that lead to collision. The probability of the success of this is $O(N^{1/2})$. For our case this probability is of order 2^{64} for the message and the same for the key as we are using a 128 bit key and the hash generated is 128 bit long. This is secure as of now and even in future to enhance security the key size can

be easily increased. Only the rule generation algorithm has to modified, the rest of algorithm can be kept unchanged.

2.6.3 Correcting Block Attack: In this attack, the attacker tries to find a message X′ such that X and X′ are distinct and H(X) = H(X′) where H is the hash function. For this attacker finds the message block X'_i such that X'_i and X_i distinct and the property $C(H, X_i) = C(H, X'_i)$ is satisfied where C is the compression function. All other blocks remaining the same he can attacker can replace X_i by X'_i to get the same hash value and hence a collision. In our attack this is equivalent to finding two 128 bit blocks such that with same previous intermediate hash, the intermediate hash generated by them is same. As we have seen already that CHASH has very high entropy which means that all the 128 bit digests are equally likely to be produced. Thus probability of existence of such two blocks would be very low. We can safely conclude that CHASH is secure against this type of attack.

2.6.4 Fixed Point Attack: A fixed point for a compression function C is defined by the values (H,X) where C(H,X) = H. If such fixed points exist and can be found then the attacker can insert an arbitrary number of such blocks without affecting the final hash value. The attack will not work for CHASH as in the padding scheme we append the length of the message. If attacker inserts certain number of blocks, the length will change and hence the final hash value will change.

2.6.5 Meet in the middle Attack: For this attack to succeed, it must be possible to go backwards through the chain. This implies that given a value H, the attacker should be able to invert a round such that $C(H_{i-1}, X_i) = H_i$ to get the values X_i and H_{i-1}. This essentially boils down to problem to reversing a cellular automata state. We have already analyzed in this for CHASH in section 2.5.

2.6.6 Common attacks against a cellular automata based hash algorithms: Few hash functions based on cellular automata have been developed over the years.

Of these only the hash function proposed by Damgard[3] has been attacked in [4]. The hash function proposed in [3] generates a 128 bit hash value. After running the algorithm on the input (it comprises of 384 rounds of rule-30) it selects 128 bits as first bits of different states (a^{257}, a^{258}, a^{384}). Here a^i is the state at the i^{th} time instant. These 128 bits make the final hash output. In [4] , the attackers found that due to local nature of the update rule and the way of choosing the 128 bits, an integer k exists such that for most of the inputs the final hash value is independent of the bits 126-k to 126 of the input. This leads to 2^k simultaneous collisions.

In out hash function the bits are not taken from different states. Now the vector in one state is dependent on the previous state. In other words each and every bit of one state is dependent on some bit of the previous state. Using this fact we can conclude the all bits of the final state are affected by some bit of the initial state. This is same as saying that each bit of the input affects some bit of final state. So, in our algorithm the hash value is dependent on all bits of the initial state. Therefore, we can conclude that the authentication scheme is secure against such attacks.

2.7 Conclusion

In this chapter we proposed a framework for online encryption with authentication and proposed a hash function based on cellular automata. The framework is used can be really inefficient as extra bits will have to be sent across the network as there are multitude of message digests instead of one for the whole message. We believe that this trade-off between how fast one requires the data and how much extra cost can be incurred will always be there. The framework can be modified according to the needs of the application. In this

light we can appreciate CHASH in a better way. If the framework has to be modified say to include 16 instead of 8 blocks to generate the digest, CHASH can still be used without any change at all. Also if the key-length or the digest length has to be changed minimal effort is required. With increasing computational power that is available to attackers these days, it is worthwhile to look at an authentication algorithm like CHASH, which is flexible enough to change with time and requirements without any great effort.

Chapter 3

Introduction to VANET

3.1 Vehicular Ad-Hoc Networks(VANET): An introduction

A **Vehicular Ad-Hoc Network**, or **VANET**, is a form of Mobile ad-hoc network, to provide communications among nearby vehicles and between vehicles and nearby fixed equipment, usually described as roadside equipment. It uses moving cars as nodes in a network to create a mobile network. VANET turns every participating car into a wireless router or node, allowing cars to connect and, in turn, create a network with a wide range. As cars fall out of the signal range and drop out of the network, other cars can join in, connecting vehicles to one another so that a mobile Internet is created.

The main goal of VANET is providing safety and comfort for passengers. To this end a special electronic device will be placed inside each vehicle which will provide Ad-Hoc Network connectivity for the passengers. This network tends to operate without any infra-structure or legacy client and server communication. Each vehicle equipped with VANET device will be a node in the Ad-Hoc network and can receive and relay others messages through the wireless network. Collision warning, road sign alarms and in-place traffic view will give the driver essential tools to decide the best path along the way.

VANET differ from Mobile Ad-Hoc Networks in some details. Rather than moving at random, vehicles tend to move in an organized fashion. The interactions with roadside equipment can likewise be characterized fairly

accurately. And finally, most vehicles are restricted in their range of motion, for example by being constrained to follow a paved highway.

Once VANET is deployed successfully, it is set to revolutionize the way one looks at vehicles. Though in a way it will only extend the current trend of increasing automation in cars. Certain vehicles today already have modern technologies like Global Position System sensors or receivers and VANET is set to drastically increase the environment awareness of vehicles.

There are tremendous benefits to be reaped through the introduction of inter-vehicular communications. Advantages range from increased comfort and entertainment to enhanced safety and better organized traffic scenarios. But is also raises several issues in conceptualization as well as implementation thus giving ample research opportunities. Though at first the concerns regarding inter-vehicular communication may seem similar to those in any network, but the expected amount of data transmission, the huge number of vehicles and the relevance of geographical location of nodes make it much more challenging .Huge amounts of intellectual and monetary capital is being put in around the world to make VANET a reality.

Lot of work and consensus has already been established as far as the setting-up of network protocols are concerned. But at the heart of the VANET lies the communication protocols and therein arises the issue of securing the communications. This area till now has been under-explored and not until recently, have researchers started to pay more and more attention towards it. The problem of information security in VANET poses a different sort of challenge altogether. With the cars being expected to have limited storage and computational capabilities on board, it renders most of the standard algorithms impractical. Hence there is need to view security in VANET from a different perspective altogether.

In the coming sections we will describe the application areas of VANET and the security requirements for each of them followed by the state of the art in the field. In the coming chapters we put forth a proposal for security protocol for VANET.

3.2 Inter-Vehicular Communication: Applications

VANET is envisioned to have a varied range of applications once it is deployed in its full capacity, but it is expected that the full capacity will be achieved in due course of time, hence some applications have been deemed to be of a greater priority than others. It means that the initial focus must be on these applications. Keeping these in mind, the applications can be divided into two broad categories-:

3.2.1 Safety Applications: These applications refer to communication of information required for safety of the vehicles and the travelers. These include collision avoidance, using aggregated positioning and velocity information to ensure better traffic scenarios, fixing liabilities in case of accidents etc. Real life example may include situations like a vehicle transmitting message to inform others about accidents, landslides etc to prevent jams, notification of a road hazard or a road feature condition, warning about potential collisions and so on. These applications not only aim at preventing dangerous situations but also aim at identifying the culprits in case such a situation has occurred to help the law enforcement agencies. The security in these cases is paramount as false information may be propagated in the network for personal gains and of course the world is not devoid of cynical people who would aim to wreak havoc. Even if a single false message goes undetected it may cause dire consequences and the number of vehicles that will receive a message would be considerable.

3.2.2 Services Related Applications: These are applications that aim to increase the comfort level or facilities for a traveler. These include automated payment services, internet availability, and multi-media services. Information services (like finding the closest fuel station etc). These applications are considered less important as of now as compared to the safety applications.

Both categories of applications require the communications to be secure, though the security requirements are of different nature. While the safety applications may only require the authentication of the senders and integrity of data, applications like payment services require data privacy as well. In this work we shall consider security in case of safety applications only.

3.3 System Model Assumptions

We assume that vehicles will communicate with other vehicles and road-side units (RSU's). We also assume the existence of an authority and the vehicles can communicate to the authority through the RSU's.

Network Model: V2V (Vehicle to Vehicle) and V2I (Vehicle to infrastructure) communications over the wireless medium employ the *Dedicated Short range Communications* (DSRC) data link technology. Vehicles transit periodic messages on a common channel dedicated to emergency situations, among the available seven DSRC channels. As in DSRC, we assume that each vehicle periodically sends messages over a single hop every 300ms to all vehicles within a range of 10 seconds of travel from itself. These figures decrease in case of slowed down or stopped vehicles. Based on the content of the message a vehicle may decide to send a similar message on its own to other vehicles within its range. Since, every vehicle is broadcasting it is clear that all vehicles are supposed to receive messages very frequently and less frequently than it will send out messages.

Access to Road Side Units: A fixed infrastructure comprised of a number of base stations positioned in close proximity to highways will act as gateways to the internet and to some certifying authority.

On board communication unit: We assume that a vehicle has an on-board communication unit for V2V and V2I communications and are equipped with wireless technology based on IEEE 802.11 technology with which they can either communicate directly or use multi-hop communication.

Event data recorder: They provide tamper proof storage and will be responsible for recording the vehicle's critical data such as position, speed, time etc during emergency events. These data will help in accident reconstruction and the attribution of liability. These can be extended to record also the safety messages received during critical events.

Tamper proof device: It provides cryptographic processing capabilities. It will take care of storing all the cryptographic material and performing cryptographic operations, especially signing and verifying safety message .By binding a set of cryptographic keys to a given vehicle, TPD guarantees the accountability property as long as it remains inside the vehicle .The access to this device should be restricted to authorized people.

GPS: We expect that in near future, most vehicles will be equipped with GPS receiver providing fairly accurate geographical position coordinates. However, the existence of GPS like device is not mandatory for supporting security in VANET.

Message Formats: The messages are sent periodically and they include location and time and speed information corresponding to the information. Emergency messages may be sent in case of occurrence of an event.

3.4 Security Challenge

VANET represent fully distributed and self organizing networks of vehicle to vehicle and vehicle to roadside communication based on wireless communication. Moreover, VANET nodes are highly mobile which result in frequent change in network topology.

It is clear from the above enumeration of applications that security requirements for the various applications have significantly varying needs with respect to security.

VANET can be vulnerable to attacks and jeopardize user's privacy, For example, an attacker could inject beacons with false information, collect vehicles messages, track their location or infer sensitive user data. Moreover, the system should be able to establish the liability of drivers in case some life critical information is inserted or modified by an attacker but at the same time it should protect as far as possible the privacy of drivers and passengers. Therefore, in order to thwart such attacks security and privacy enhancing mechanisms are necessary, in fact, a prerequisite for deployment.

In this work we will only consider the security of above mentioned safety applications. It is a consensus that vehicles will broadcast messages from time to time to pass-on various kinds of information to other vehicles. As the messages will be broadcast and there will be no one-to-one communication between vehicles so privacy of the messages is not required, but the vehicles need to make sure that the information has been sent by an authentic node in the network. Following are a few attacks that can be employed by adversaries in VANET-:

False Information: The adversary can try and infuse false information in the network for personal gains or just to create havoc. The false information can be

about one's own position or about the environment. This may also include replaying of an older valid message.

Masquerading: One node can pretend to be another node by using false identity to get away with false information attacks or for some other purpose.

Denial of Service: An attacker may try to jam the network by aggressively injecting spurious messages.

Tracking other vehicles: An attacker may try to track a particular vehicle with malicious intent based on the messages transmitted by that vehicle.

Following are a few properties the security protocol must possess in order to thwart the above mentioned and other attacks in VANET:

Authentication: Vehicles must be able to ensure that the message has been sent by a legitimate node.

Anonymity: While the authenticity of sender must be verified it is also imperative that the actual identity of the sender is not revealed. It must also be impossible to link two or more messages to the same sender.

Non-repudiation: No vehicles should be able to deny sending a message if it actually has. It is very important for fixing liabilities to the right vehicles.

Verification of Data: This may not directly concern the security aspect, but in cases like an attacker replaying an older valid message, it must be possible to discard such messages based on context and current information about the environment. Here, we are not concerning ourselves with the verification of content of the message.

3.5 State of the art

VANET (Vehicular Ad-hoc-Networks) is an emerging research area. Currently, most of the research in VANET is focused on the development of a suitable MAC layer with very few efforts focused towards security architecture and protocols for VANET.

The research on VANET security is just starting, with few pioneer papers so far. The most prominent industrial effort in this domain is carried out by Car 2 Car Communication Consortium [9], the IEEE 1609.2 working group [19], the NoW project [20] and the SeVeCom project [21] with all of them developing VANET Security architecture. Their common basic elements include the use of *Certification authorities (CAs)* and public key cryptography to protect vehicle to vehicle (V2V) and Vehicle to infrastructure (V2I) messages. It has now become an established consensus that Public Key cryptography is the way to go about for VANET. This is mainly due to the fact the messages are broadcast and one-to-one communication is not the norm. Due to this fact symmetric key cryptography will incur huge costs in frequent key establishment procedures and they are also difficult to implement as the nodes are constantly on the move. Therefore here on we will concentrate on public key methods only. For all the perspective security protocols, message authentication, integrity and non-repudiation, as well as protection of private user information are identified as primary requirements.

On academic front, there are few publications describing the security architecture of VANETS, [10,11,12] but not may of them proposes specific protocols that considers all the practical requirements needed to secure VANET safety applications. Gerlach [14] describes the security concepts for vehicular networks. Hubaux et al. [13] take a different perspective of VANET security and focus on

privacy and secure positioning issues.Parno and Perrig [17] discuss the challenges, adversary types and some attacks encountered in vehicular networks; they also describe several security mechanisms that can be useful in securing these networks. El Zarki et al. [16] describes an infrastructure for VANETs and briefly mentions some related security issues and possible solutions. The use of digital signatures in the vehicular environment is discussed in [15].

Meanwhile, [23] mentions VANET as an application for group signature, that is, cryptographic primitives for anonymous authentication. This is a stronger property than pseudonymous authentication, as any two group signature generated by a node cannot be linked. A Group signature scheme is basically a method for allowing a member of a group to anonymously sign a message on behalf of the group. In [26] Bellare proposes a static group signature based on underlying digital signature and encryption scheme in which size of group parameters depend on the number of group members. It also provides theoretical foundations for the group signature scheme along with various security requirements that it should satisfy. Bellare in [27] proposes a dynamic group signature scheme which doesn't depend on the number of group members. Xuanwu [28] proposes another dynamic GS approach based on elliptic curve cryptography.

However, [18] proposes schemes for VANET security that relies on the concept of pseudonym authentication. [18] assumes the presence of certification authority which is vested with legal power to disclose node identities and is required to certify the keys of vehicular nodes. It proposes the following schemes for VANET security

a) *Baseline Pseudonym [18]:* Under this scheme every node is equipped with a set of pseudonyms (public private key pairs) along with public keys certified by a

certifying authority. It uses a digital signature scheme like RSA for signing messages and attaches public key certificate for message validation.

b) *Group signature Scheme [23]*: In this each node is equipped with a group public key and its private signing key. Thus this scheme allows any node to sign messages on behalf of group without nodes identity being revealed to the signature verifier.

c) *Hybrid Scheme [18]:* The combination of pseudonym with group signature is basic element of this scheme. It uses digital signature for message authentication and group signature scheme for creating on the fly certificates of public key.

3.6 Motivation

As established in the previous sections, the security of safety applications in VANET require only authentication along with anonymity. There are virtually no anonymous authentication schemes that have been developed keeping the requirements and constraints of VANET in mind. Various frameworks have been proposed and all of them target the pre-available authentication algorithms that show an exemplary performance as far as security is concerned but are not so impressive when it comes down to the storage and computational complexity and ease of implementation. Thus it is reasonable to devote time and effort to the development of such a scheme. It is specially justified in the wake of the fact that VANET is something that is expected to be up and running within a decade from now.

3.7 Objective

In section 1.3 we outlined the various methodologies or protocols for ensuring security in VANET. Apart from the security a scheme provides, it is paramount to consider the costs it incurs in terms of time and memory usage. Also it is reasonable to assume that a vehicle would most probably be confined to one area most of the time, which leads us to another assumption that a group of vehicles (we consider the group to be large) will not be sporadically dynamic in its composition. Hence our target is to develop a secure group signature scheme that is more efficient than those currently available in the literature.

3.8 Conclusion

In this chapter we outlined an introduction to VANET, it applications, constraints and requirements in terms of security and otherwise. Then the problem taken up in this work and its motivations have been explained. In the coming chapters we propose a group signature scheme based for VANET along with its security analysis followed by analysis and comparison of its efficiency with other alternatives.

Chapter 4

GSCRT: A Group Signature Scheme

4.1 Introduction

Based on the background information and system model assumptions established in the previous chapter, we proceed to present a proposal for security in VANET. This group signature scheme is based on certain assumptions about the composition and dynamics of the network which we outline in the forthcoming section. The scheme is based on Chinese Remainder Theorem and it has been built-up upon the hierarchical access scheme proposed in [30] and then we proceed by explaining the theorem and then moving on to our proposal.

4.2 Preliminaries

4.2.1 Network and Infrastructure Assumptions: Apart from the assumptions stated in section 3.3 we assume that a region is divided into several groups of vehicles. This group is of course assumed to be much larger than the set of vehicles a vehicle can communicate with at some time. It means that at any time a vehicle will be able to send messages to and receive from, a set of vehicles that are also in the same group. We assume the existence of a group manager for every group, typically a government authority or some car manufacturer's agent. The role of the group manager will become clear in the forthcoming sections.

4.2.2 Chinese Remainder Theorem: The **Chinese remainder theorem** is a result about congruence's in number theory. Following is the statement of the theorem in one of its forms:

Suppose $n_1,n_2,\ldots n_k$ are integers which are pair-wise co-prime. Then, for any given integers $a_1,a_2,\ldots a_k$, there exists an integer x solving the system of simultaneous congruences -:

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ x &\equiv a_2 \pmod{n_2} \\ &\vdots \\ x &\equiv a_k \pmod{n_k} \end{aligned}$$

Furthermore, all solutions x to this system are congruent modulo the product $N = n_1n_2\ldots n_k$ i.e. the solution x is unique modulo the product $n_1n_2\ldots n_k$.

Following is an algorithm to find the solution given the relatively prime numbers $n_1,n_2,\ldots n_k$ and the set of residues $a_1,a_2,\ldots a_k$.

Let N be the product of the relatively prime numbers $n_1,n_2,\ldots n_k$. Let N_i denote the product of these numbers excluding n_i.

The number c_i is computed as follows:

$$c_i = (N_i)(N_i^{-1} \bmod n_i)$$

Then the solution x can be written as

$$x = (\sum (a_i)(c_i)) \bmod N$$

4.3 GSCRT

In this section we present a group communication scheme based on Chinese remainder theorem.

4.3.1 Proposal: Let there are k group members. There is one group manager per group who is in charge of generating the keys and distributing them to the members.

Public Information (known to all members and manager)

N_G - A prime number

Group Manager has following information:

N_o - Private (known only to manager) number used to reveal identity of the message sender

Nd_i - Private (known only to manager) number required to distinguish the product used in construction of a particular $CRTK_i$ (will use a different Nd_i during construction of a particular $CRTK_i$, Manager need not store them).

Both Nd_i and N_o are prime numbers of order of 512 bits

Group Members:

Each member M_i is given the following information by the group manager.

N_i - A prime number known only to M_i

a_i - A random number ($< N_i$) known only to M_i used for sign verification

Pr_i = Product* Nd_i

Here Product = $\prod (N_j)$ which will be same for every user of the group and j € {0 ...k }.

N_G - A number known to all members.

All N_i's, N_G and Nd_i's are prime numbers.

$CRTK_i$ which is created as follows:

$CRTK_i \bmod N_0 \equiv ID_i$

$CRTK_i \bmod N_1 \equiv a_1$

$CRTK_i \bmod N_2 \equiv a_2$

.....

$CRTK_i \bmod N_i \equiv a_i$

.....

$CRTK_i \bmod N_k = a_k$

$CRTK_i \bmod Nd_i = ad_i$ *(here ad_i can be any random number $< Nd_i$)*

$CRTK_i = <ID_1, a_1, a_2, a_3, a_4....., a_i... a_k, ad_i>$ *(k+2 Tuple) as in CRT*

Note that this entity $CRTK_i$ is unique modulo Pr_i (follows from Chinese Remainder theorem).

All this information is available with a particular member.

4.3.2 Signature Generation:

To send the message the member creates a signature Y in the following manner.

$$Y \bmod Pr_i \equiv CRTK_i$$
$$Y \bmod N_G \equiv Hash(Message)$$
$$Y = < CRTK_i, Hash(Message) >$$

4.3.3 Signature Verification:

To verify the signature a member M_j does the following -:

$$X = Y \bmod N_j$$
$$If\ (X == a_j)\ the\ signature\ is\ verified.$$

It is important to note that the verifier does not need to and cannot extract $CRTK_i$ of the sender, to verify the authenticity of the sender.

4.3.4 Identity Extraction:

Only Manager will be able to reveal the identity of message sender by doing following operation:

$$ID_i = Y \bmod N_o$$

This ID_i then can be mapped to the actual identity of the sender.

4.3.5 Correctness:

In order to verify the receiver does the following check -:

If($Y \bmod N_i == a_i$)

We have to prove that in case of an authorized sender, this check does stand to be true.

$$CRTK_i = (\sum a_j * ((Pr_i/N_j)* (((Pr_i/N_j)^{-1} \bmod N_j)))) \bmod Pr_i \quad \text{---------- (1)}$$

Where j varies from 0 to k+1 (assuming $Nd_i = N_{k+1}$)

$Y = (CRTK_i (N_G * (N_G^{-1} \bmod Pr_i)) + Hash<Message>(Pr_i * (Pr_i^{-1} \bmod N_G))) \bmod Pr_i * N_G$ -(2)

Let Z be a number such that

$Z \bmod N_0 \equiv ID_i$

$Z \bmod N_1 \equiv a_1$

$Z \bmod N_2 \equiv a_2$

.....

.....

$Z \bmod N_k \equiv a_k$

$Z \bmod Nd_i = ad_i$

$Z \bmod N_G \equiv Hash<Message>$

Let $N_{k+2} = N_G$, $a_0 = ID_i$, $a_{k+2} = Hash<Message>$

Let $P = Pr_i * N_G$

Therefore Z can be written as-:

$Z = (\sum a_j * ((P/N_j) * (((P/N_j)^{-1} \bmod N_j)))) \bmod P$ where j varies from 0 to k+2

$Z = (\sum a_j * ((P/N_j) * (((P/N_j)^{-1} \bmod N_j)))) \bmod P + a_{k+2} * (Pr_i * (Pr_i^{-1} \bmod N_G)) \bmod P$,

j varies from 0 to k+1

Let Z = Z1 + Z2, where Z1 and Z2 are the two terms in the above equation

$Z1 = (\sum a_j * ((Pr_i * N_G/N_j) * (((Pr_i * N_G/N_j)^{-1} \bmod N_j)))) \bmod P$

Since Pr_i is a multiple of N_j,

$N_G^{-1} \bmod N_j = N_G^{-1} \bmod Pr_i$

$Z1 = (((\sum a_j * ((Pr_i/N_j) * (((Pr_i */N_j)^{-1} \bmod N_j)))) * (N_G * (N_G^{-1} \bmod Pr_i)))) \bmod P$ – (3)

Now we have from equation 1

$\sum a_j * ((Pr_i/N_j)* (((Pr_i*/N_j)^{-1} \bmod N_j)) = qPr_i + CRTK_i$ for some integer q ----- (4)

From (4) and (5)

$Z1 = ((qPr_i + CRTK_i)* N_G*(N_G^{-1} \bmod Pr_i)) \bmod P$

$= (((qPr_i*N_G + CRTK_i * N_G) \bmod P * (N_G^{-1} \bmod Pr_i) \bmod P) \bmod P$

$= (((qP + CRTK_i * N_G) \bmod P * (N_G^{-1} \bmod Pr_i) \bmod P) \bmod P$

$= (((CRTK_i * N_G) \bmod P * (N_G^{-1} \bmod Pr_i) \bmod P) \bmod P$

$= ((CRTK_i * N_G* (N_G^{-1} \bmod Pr_i)) \bmod P$

$Z1 = ((CRTK_i * N_G* (N_G^{-1} \bmod Pr_i)) \bmod P$ --------- (5)

$Z = Z1 + a_{k+2} * (Pr_i * (Pr_i^{-1} \bmod N_G)) \bmod P$

$= ((CRTK_i * N_G* (N_G^{-1} \bmod Pr_i)) \bmod P$ + Hash<Message> * $(Pr_i * (Pr_i^{-1} \bmod N_G)) \bmod P$

$Z=((CRTK_i * N_G* (N_G^{-1} \bmod Pr_i)$+Hash<Message> * $(Pr_i * (Pr_i^{-1} \bmod N_G)))\bmod Pr_i* N_G$ = Y from (2)

Therefore Y = Z

Hence,

$Y \bmod N_j = Z \bmod N_j = a_j$

4.3.6 Application to VANET: For deploying GSCRT in VANET, we assume the vehicles are divided in groups of 10000 each (this number may of course and accordingly the parameters will change). Therefore there are 10000 N_i's per group each of 80 bits, while N_0 and Nd_i are of 512 bits each. We assume that there are Road side units available at boundaries of regions so that when a vehicle

travels outside its group it can contact the manager through the RSU and obtain new parameters for the new group.

4.3.7 Addition of a new Member: A new member will obtain its parameters directly from the manager. In this scheme the addition of member will require the inclusion of a new N_i-a_i pair, thus leading to a need to change the parameters of all other members. This implies that the scheme is truly static in nature.

4.3.8 Removal of a Member: The advantages of using a group signature scheme for VANET are accompanied by some challenging problems, notably certificate revocation. For example, the certificates of a detected attacker or malfunctioning device have to be revoked, i.e., it should not be able to use its keys or if it still does, vehicles verifying them should be made aware of their invalidity. In this particular proposed protocol, CRTK given to member vehicles can be considered as a certificate. Following is one of the approaches for such revocation:

Once the Trusted authority has decided to revoke certificate of a given vehicle M, it sends to it a revocation message encrypted with the vehicle's public key(assuming symmetric key communication).After the message is received and decrypted by the TPD of the vehicle, the TPD erases all the keys and stops signing safety messages. Then it sends an ACK to the CA. All the communications between the CA and the vehicle take place in this case via road side units (RSUs). In fact, the CA has to know the vehicle's location in order to select the RSU through which it will send the revocation message. If it does not know the exact location, it retrieves the most recent location of the vehicle from a location database and defines a paging area with base stations covering these locations. Then it multicasts the revocation message to all these base stations.

4.4 Security Analysis

In this section we analyze the security and robustness of our algorithm with respect to various requirements of a group signature scheme and a few attacks.

4.4.1 Anonymity: This property requires that a member should not be able to reveal the identity of another member from the signature that the later has sent. In GSCRT the identity is embedded into the key in the following way:

$$CRTK_i \bmod N_0 \equiv ID_i$$

The number N_0 is not available to any of the members (it is available only with the manager). N_0 is definitely a part of the products available with all the members. So to reveal the identity a member must be able to factorize any of the products or must guess N_0. The size of the product is around 100 kilo bytes with two factors of size 512 bits which makes it computationally infeasible to be factorized. Guessing N_0 correctly has a probability 2^{-512} as N_0 is a 512 bit number. This probability is certain negligible which leads us to the conclusion that GSCRT provides anonymity.

4.4.2 Non-frameability: This property requires that no member should be able create any valid signature that links to identity of some member other than his own. To create a valid signature that frames some other vehicle, a member needs to know all N_i-a_i pairs and N_0. Let us assume that from the product, the adversary is able to extract the smaller prime factors (Ni's). Then to frame another member, he must get N_0 and the identity of the member. Getting N_0 is equivalent to factoring the product of N_0 and Nd_i, which is computationally infeasible. Therefore GSCRT provides non-frameability.

4.4.3 Unlinkability: This property requires that deciding whether two different valid signatures were computed by the same group member is computationally hard. In GSCRT the members receive Y which yields CRTK of the member and the hashed message as residues modulo Pr_i and N_G respectively. If the recipient is able

to extract CRTK then he can definitely conclude that the messages are from the same sender just by comparing CRTK's. It must be noted though that even in this event it is not possible for him to determine the identity of the sender.

Now let us focus on the question that whether CRTK can be extracted from from Y. The following equality gives the relation between CRTK and Y:

$$Y \bmod Pr_i = CRTK_i$$

The parameter Pr_i, where i the sender, is unknown to all the recipients. If the recipient attempts at guessing Pr_i, we can simply strike it off as it is large number and therefore computationally infeasible to guess. This leads us to conclude that CRTK cannot be extracted from Y. Therefore under GSCRT different messages sent by the same member are unlinkable.

4.4.4 Traceability: This property requires that the group manager is always able to open a valid signature and identify the actual signer. In GSCRT a vehicle can create a valid signature that cannot be traced to any identity if somehow it extracts all N_i-a_i pairs. In fact extracting only N_i's will suffice as a_i's can be extracted using N_i's from any valid signature received from some other vehicle. N_i's can be extracted by factorizing the product but it seems infeasible even though N_i's are the small factors, as both the size of the product and the number of N_i's is large. But if the adversary forms a coalition or colludes with several others members and they all share their information, it will definitely reduce the complexity of factoring the product. This complexity will decrease with increase in the number of colluding members. N_i's can also be determined as in attack mentioned later in section 4.4.5.2.

Therefore GSCRT does not provide *coalition-resistance* and does not provide traceability in all conditions. Note that the properties of anonymity, non-frameability and un-linkability still continue to hold even in case of colluding members.

4.4.5 Attacks: In this section GSCRT is analyzed with respect to some attacks. Note that many of the attacks mentioned in the first chapter are automatically ruled out due to above mentioned properties of GSCRT.

4.4.5.1 Insider Replay Attack: In this attack a member of the group intending to cause confusion by propagation of contextually incorrect information, replays a signed message that he has received from some other member. This attack only gains the stature of an "attack" if the message is replayed after a considerable amount of time. The attack can be easily thwarted by including the "timestamp" in the message. The recipient of a message can check using the timestamp whether the message is "too old" to be used.

Let us now look at the scenario when the attacker tries to modify the timestamp in the original message. This will lead the message signature to change as it includes the hashed message. To make the signature to comply with the changed message the attacker has to change Y accordingly which means he needs to extract CRTK of the sender and then recreate Y. This is not possible as the attacker does not know and cannot feasibly guess Pr_i of the sender.

4.4.5.2 Guessing N_i's: Each N_i is of 80 bits. As N_i's are primes, total number of possible values for it is definitely less than 2^{80}. Therefore it is not too difficult to guess N_i's by enumerating all possible values. Whether the guessed values are correct or not can be determined by storing a set of messages $Y_1 \ldots Y_k$ all from different members and checking the residues modulo the guess value of an N_i. If all yield the same residue, the guessed value is a correct one. It may be argued that determining whether two messages are from different members is not obvious, this problem may be overcome by storing a larger number of members or by using the location information in the messages to distinguish. For example if locations from two messages received during the approximately same time are far apart, it may be concluded that the senders are distinct.

4.5 Time Complexity

4.5.1 Basic Definitions: The following definitions of bit complexity for basic operations on integers have been outlined in [29].

Integer Addition

Bit complexity of computing x+ y for integers x and y is O (lg x + lg y).

Integer Multiplication

Bit complexity of multiplying integers x and y is O((lg x) (lg y))

Integer Division

Bit complexity of dividing integer x by integer y is O((lg x) (lg y))

Modular Integer Addition

Input: A positive integer N and integers x,y € Z_N = {0,...,N 1}.

Output: x + y (mod N).

The bit complexity of this problem is O(lgN).

Modular Integer Multiplication

Input: A positive integer N and integers x,y € Z_N

Output: xy (mod N).

Here the bit complexity is $O((lgN)^2)$

Modular Inverse

Input: A positive integer N and an integer x € Z_N = {a € Z_N : gcd(a,N) = 1}.

Output: y € Z_N such that xy = 1 (mod N).

The bit complexity is $O((lgN)^2)$

Modular Exponentiation

Input: A positive integer N, an integer x € {0,...,N 1}, and any integer k.

Output: x^k (mod N).

Using the method of repeated squaring, the bit complexity is $O((lgk) (lgN)^2)$.

As Trusted Authority or Group Manager has sufficient storage and computation power, so we are not taking in to consideration the time required to generate CRTK and other parameters needed to join the group. Our emphasis will be on estimating the time required to generate message signatures and verifying them.

4.5.2 Signature Generation Complexity:

We use Chinese remainder theorem while generating message signatures and so we need to look into the operations involved, in order to calculate the bit complexity of signature generation.

$$Y \bmod Pr_i = CRTK_i$$
$$Y \bmod N_G = Hash\ (Message)$$
$$Y = <CRTK_i, Hash(Message)>$$

Now, we know that $CRTK_i$ and Pr_i are of order of (k'b) bits and size of N_i is b bits.

$$Y= (CRTK_i\ (N_G * (N_G^{-1} \bmod Pr_i)) + Hash<Message>(Pr_i*(Pr_i^{-1} \bmod N_G))) \bmod Pr_i*N_G$$

Bit complexity Calculations:

1. $(N_G * (N_G^{-1} \bmod Pr_i)$

It involves Modular inverse of N_G w.r.t Pr_i which is of $O((\lg Pr_i)^2)$=**$O((k'b)^2)$** and multiplication of N_G with its inverse which is of $O(k'b* b) = O(k'b^2)$. However, size of this product is (k'+1)*b bits.

2. $(CRTK_i\ (N_G * (N_G^{-1} \bmod Pr_i)))$

So bit complexity of $(CRTK_i\ (N_G * (N_G^{-1} \bmod Pr_i)))$ is $O((\lg CRTK_i)\ \lg(N_G * (N_G^{-1} \bmod Pr_i))$

=O(k′b * (k′+1)*b) = **O((k′b)²).** Size of this product is (2k′+1)*b bits

3. Pr_i(Pr_i^{-1} mod N_G))*

It involves Modular inverse of Pr_i w.r.t N_G which is of $O((\lg N_G)^2)=O(b^2)$ and multiplication of Pr_i with its inverse which is of O(k′b* b) = **O(k′b²)**. However, size of this product is k′+1)*b bits

4. Hash<Message> (Pri(Pr_i^{-1} mod N_G))*

So bit complexity of Hash<Message>(Pr_i*(Pr_i^{-1} mod N_G))

is O((lg Hash<Message>) lg (Pr_i*(Pr_i^{-1} mod N_G))) =O(b*(k′+1)*b) = **O(k′b²)**

Size of this product is (k′+2)*b bits

*5. ($CRTK_i$ (N_G * (N_G^{-1} mod Pr_i)) + Hash<Message>(Pr_i*(Pr_i^{-1} mod N_G)))*

Bit complexity of addition is O (Size of (1) + Size of (2)) = O (3(k′+1)*b) =O (k′b)

6. Message signature (Y)

Bit complexity of computing Y is equivalent to calculating of Modulus of (3) w.r.t Pr_i*N_G

= O ((lg Pr_i*N_G) ²) = **O((k′b)²)**

Thus we can conclude that bit complexity of generating message signature is ***O((k′b)²)***

4.5.3 Signature Verification Complexity:

For each N_i and a_i

$X = Y \bmod N_i$

If ($X == a_i$) the signature is verified.

We only require taking modulus of *Y w.r.t. to* N_i and we know that modulus is similar to dividing *Y w.r.t. to* N_i and calculating the remainder.

Bit complexity of division will be of order O(lg (Y) lg (N_i)). We can approximate Y size to be order of k''b bits and we know size of N_i to be b bits. Hence bit complexity of verification is O(k''b * b)) = $\boldsymbol{O(k''b^2))}$.

Note : Here k'' is equal to (k + c') and k' is equal to (k + c),

where k is number of N_i *used while creating any CRTK and c, c' are constant.*

4.6 Overhead and Storage Requirements

Let each N_i's have size b bits and there are k members. CRTK's and Pr's will have size of the order of (b*k+1024) bits. This poses a problem for large groups as CRTK and Pr will lead unacceptable size requirements. The overhead will be the size of Y which will of the order of size of Pr*N_G in the worst case.

For b = 80 bits and k = 10000 storage size is of the order of **202 kilo bytes** and overhead is of the order **101 kilo bytes.**

4.7 Conclusion

In this chapter GSCRT, a group signature has been proposed. The scheme does well on the security front, but does not provide coalition resistance. Though signature generation and verification involve fairly simple operations, the scheme still is not efficient due to the huge size of the parameters involved. We have analyzed for 10000 members as typically the number of vehicles in a group is expected to be large. In the next chapter we present a modified group signature scheme which performs much better comparatively.

Chapter 5

Modified GSCRT

5.1 Introduction

Because of the memory requirement problems with the original GSCRT scheme mentioned in the previous chapter, we propose a new version of the previous scheme which tries to overcome memory requirement problems. Following the specifications for modified GSCRT we show that this version performs drastically while providing the same security.

5.2 Modified GSCRT

In this scheme, the number of N_i used in the construction of the $CRTK_i$ is not equal to but less than the number of users (or vehicles) present in the group. Therefore the storage requirements will not increase linearly with the number of group members.

5.2.1 Proposal: Let there are n group members. Let k+2 be the number of prime numbers used to construct a particular CRTK

Public Information (known to all members and manager)

N_G - A prime number known to all members

Group Manager has following information:

N_o - Private (known only to manager) number used to reveal identity of the message sender

Nd_i - Private (known only to manager) number required to distinguish the product used in construction of a particular $CRTK_i$ (will use a different Nd_i during construction of a

particular $CRTK_i$, Manager need not store them).

Both Nd_i and N_0 are prime numbers of order of 512 bits

Group Members:

The manager creates k_C2 pairs of N_i's and corresponding a_i's , and distributes 2 $<N_i, a_i>$

pairs to each member.

Each member M_i is also given the following information by the group manager.

Pr_i = Product* Nd_i

Here Product = $\prod (N_j)$ which will be same for every user of the group and j € {0 ...k }.

N_G - A number known to all members.

All N_i's, N_G and Nd_i's are prime numbers.

$CRTK_i$ which is created as follows:

$$CRTK_i \bmod N_0 \equiv ID_i$$
$$CRTK_i \bmod N_1 \equiv a_1$$
$$CRTK_i \bmod N_2 \equiv a_2$$
.....
$$CRTK_i \bmod N_i \equiv a_i$$
.....
$$CRTK_i \bmod N_k = a_k$$
$$CRTK_i \bmod Nd_i = ad_i \quad (\text{here } ad_i \text{ can be any random number} < Nd_i)$$
$$CRTK_i = <ID_1, a_1, a_2, a_3, a_4....., a_i... a_k, ad_i> (k+2 \text{ Tuple}) \text{ as in CRT}$$

The modulus is taken with respect to N_0 ,N_1..... N_i..... N_k, Nd_i.

Note: Nd_i is a prime number and thus can be used along with all other N_i in CRT.

All this information is available with a particular member.

5.2.2 Signature Generation: To send the message the member creates a signature Y in the following manner.

$Y \bmod Pr_i = CRTK_i$

$Y \bmod N_G = Hash\ (Message)$

$Y = < CRTK_i, Hash\ (Message) >$

5.2.3 Signature Verification: To verify the signature a member M_j does the following -:

For each N_i and a_i it has (there are two)

$X = Y \bmod N_i$

If ($X == a_i$) the signature is verified.

It is important to note that the verifier does not need to and cannot extract CRTK of the sender, to verify the authenticity of the sender.

5.2.4 Identity Extraction: Only Manager will be able to reveal the identity of message sender by doing following operation:

$ID_i = Y \bmod N_o$

This ID_i then can be mapped to the actual identity of the sender.

Note:

$N_0, N_1, N_2, \ldots N_k, N_G, Nd_i$ they are all prime numbers.

Here there can be multiple users using the same set of values for $<N_i, a_i>$ for verification but $CRTK_i$, they will use will be different because of different ID_i and Nd_i

Moreover, they will have different Pr_i because of Nd_i which will ensure that no other user can get back $CRTK_i$ from Y as he doesn't have any idea about Nd_i used and hence no knowledge of product (as modulus) used in Y.

5.2.5 Correctness: As underlying operations remain the same as mentioned in the original GSCRT, correctness proof is similar to the one mentioned before.

5.2.5 Application to VANET: For deploying GSCRT in VANET, we assume the vehicles are divided in groups of 10000 each (this number may of course and accordingly the parameters will change). The number of N_i's is taken to be 25 each of 80 bits, while N_0 and Nd_i are of 512 bits each.

5.2.6 Addition of a new member: In contrast to the original version the modified GSCRT scheme is not absolutely static in nature. When a new member has to be added, the manager can assign it to any of the existing k sets and will provide the parameters accordingly.

5.2.7 Removal of a member: Apart from the approach mentioned in section 4.3.8, there is one more approach based on timestamps for certificate revocation or removal of members.

***Other Approach*:** We can opt for using short certificate lifetime that will make certificates (CRTK's) expire thus revoking the certificates. This can be achieved by including a timestamp during the creation of CRTK by the trusted authority which specifies the date up to which a particular CRTK is valid.

Once the trusted authority has included a timestamp in CRTK, that CRTK will remain valid till date. After CRTK expires, in order to continue communicating with that CRTK, vehicle has to go to road side units to get it CRTK refreshed with a new timestamp else vehicle will not be able to communicate with its expired CRTK. Thus, this approach can also be used to revoke a particular vehicle's certificate (CRTK) by not refreshing its certificate with a new timestamp. However, a malicious node will be able to send erroneous message as long as its certificate is valid. Thus, it creates a vulnerability window. However, this vulnerability window can be reduced by asking vehicles to frequently refresh their CRTK with new timestamps

5.3 Timestamp inclusion in $CRTK_i$

The manager creates k_{C2} pairs of N_i's and corresponding a_i's, and distributes 2 $<N_i, a_i>$

pairs to each member. Each vehicular member M_i is also given the following information by the group manager.

Pr_i = Product* Nd_i. Here Product = $\prod (N_j)$ which will be same for every user of the group and j varies from 0 to k.

N_G - A number known to all members

Manager includes a timestamp t_i by XORing it with all the a_i's in $CRTK_i$

Timestamp t_i is basically a date till which this $CRTK_i$ is valid. For this protocol its taken to be 10 days from the date on which vehicles comes to refresh its timestamp .So that the vehicles knows when its $CRTK_i$ will expire and can accordingly refresh its timestamp.

$CRTK_i$ which is created as follows:

$CRTK_i \bmod N_0 \equiv ID_i \oplus t_i$

$CRTK_i \bmod N_1 \equiv a_1 \oplus t_i$

$CRTK_i \bmod N_2 \equiv a_2 \oplus t_i$

.....

$CRTK_i \bmod N_i \equiv a_i \oplus t_i$

.....

$CRTK_i \bmod N_k = a_k \oplus t_i$

$CRTK_i \bmod Nd_i = ad_i \oplus t_i$ *(here ad_i can be any random number < Nd_i)*

$CRTK_i = <ID_1, a_1, a_2, a_3, a_4....., a_i... a_k, ad_i>$ *(k+2 Tuple) as in CRT*

The modulus is taken with respect to $N_0, N_1, ... N_i ..... N_k, Nd_i$.

All N_i's, N_G and *Nd_i's* are prime numbers.

Note: *This $a_1 \oplus t_i$ is XORring of a_1 with t_i*

All this information is available with a particular member.

5.3.1 Signature Verification with Timestamp:

To verify the signature a member M_j does the following -:

For each N_i and a_i he has (there are two)

$X = Y \bmod N_i$

$Z = X \Theta a_i$

If (Z is a valid timestamp) the signature is verified.

5.4 Security Analysis

5.4.1 Properties: In the original GSCRT scheme we showed that GSCRT provides anonymity, non-frameability and unlinkability. These properties are unchanged as far as modified GSCRT is concerned. The scheme still does not provide *coalition-resistance* and traceability in all conditions. In fact achieving non-traceability becomes easier in the modified version because extracting small factors is easier as they are less in number for the same group size.

5.4.2 Attacks: Two attacks were mentioned in the section 4.4.5. Modified GSCRT like the original version is resistant against the insider replay attack. For the attack in section 4.4.5.2, the original scheme failed, but the modified GSCRT withstands that attack due to presence of timestamp in a_i's. Due to this when the adversary tries to guess N_i's, he is unable to verify correctness of the guess as even for the same N_i, different messages may give different residues due to possible presence of varying timestamps. The group manager can in fact make sure that each member has a different timestamp. An attempt to try all possible combinations of N_i-a_i pairs will be computationally infeasible.

5.5 Time Complexity:

5.5.1 Signature Generation: In this proposal, we do signature generation in a fashion similar to that in proposal 1. So, time taken to sign a message will be of

order **O((k'b)^2**). But here k' is significantly less than that in proposal 1(reduction is from 10,000 to 25)

5.5.2 Signature Verification: Signature Verification is different from the one in proposal 1. In this scheme, vehicle has to verify signature for every $<N_i , a_i>$ it has . However, verification for different pairs can be done in parallel thereby keeping the verification time equivalent to verification by a single $<N_i , a_i>$ pair. Hence time taken for verification is of order of **O(k''b^2))** as given earlier.

5.6 Communication Overhead and Storage Requirements

5.6.1 Overhead: While sending the message, a vehicle needs to send Y also along with it in order to get verified and accepted. So, communication overhead consists of byte size of Y. Let the number of Ni's be k and each be of b bits. Let the number of members be n. Then CRTK's and Pr's will be of order (k*b + 1024) bits each. A set of N_i's and the corresponding a_i's will be shared by (n/k$_C$2) members. We can choose k and b to reduce the storage and to restrict the number of members sharing the same set of N_i's and corresponding a_i's.

$$Y \bmod Pr_i = CRTK_i$$
$$Y \bmod N_G = Hash\ (Message)$$
$$Y = < CRTK_i, Hash(Message) >$$

For b = 80 bits, n = 10000 and k = 25

- Size of $CRTK_i$ is of order of 25*80 + 1024 = 3024 bits =378 bytes
- As N_G is of the order of 80 bites and Y is created using CRT we can say
- Overhead = Size of Y = order of 3024+80 bits = 3104 bits **= 388 bytes** in the worst case.

5.6.2 Storage Requirements: Storage size comes out to be = 3024 ($CRTK_i$) + 3024(Pr_i) + 320 bits (N_i's and corresponding a_i's) + 80 bits (N_G) = 6448 bits = **= 806 bytes.**

However, use of fixed numbers of Ni's while constructing $CRTK_i$ leads to sharing of same pair of $<N_i, a_i>$. Therefore, numbers of members with same parameters is approximately 33. i.e. (10,000/25C2). Thus using the second scheme we can achieve considerable reduction in communication overhead and storage requirements.

5.6.3 Communication Overhead Comparison: In [18] for baseline pseudonym based approach, overhead consists of public key(25 bytes), certificate on public key (64 bytes) along with message signed with public key (48 bytes) , giving total overhead of 137 bytes. For Group Signature approach, overhead is signature on message signed with private signing key .Hence, total overhead is 225 bytes. Similarly for hybrid scheme overhead comprises of public key (25 bytes) , certificate on public key using group signature (225 bytes) along with message signed with public key (48 bytes) giving a total overhead of 296 bytes.

Scheme	Overhead (in bytes)	Storage (in bytes)
GSCRT	101 Kilo	202 Kilo
Modified GSCRT	388	806
Baseline Pseudonym (using ECDSA)[18]	137	22 MB
Group Signature [23]	225	864
Hybrid [18]	298	913

Table 5.1 Overhead and Storage Comparison with Other Schemes

5.6.4 Storage Overhead Comparison: For baseline Pseudonym based approach mentioned in [18], one need to store certified public private key pairs. So assuming 8-10 hrs of daily car usage and refilling after one year , a vehicle needs to have approximately 200,000 such pairs where each pairs has size of 113 bytes (25 bytes public key + 24 bytes private key + 64 bytes of certificate) , there by giving combined storage size of 22 MB. Security level for certificate is 128 bits.

For Group Signature Scheme, [18] uses a security level of 128 bits , therefore one need to store at least a group public key and a private signing key for signing messages on behalf of group. Therefore, total storage size

= 800 bytes (group public key) + 64 bytes (private key) = 864 bytes.

In hybrid approach, we have a digital signature on message with security level of 80 bits along with a certificate of public key created on the fly using a group signature. Therefore, we need to store at least a public and private key pair for digital signature scheme along with group signature parameters. Thus, total storage requirement is

25 bytes public key + 24 bytes private key + 800 bytes(group public key) + 64 (group signing private key) = 913 bytes.

Therefore our group communication scheme performs well on the storage front while it is marginally more costly as far as the communication overhead is concerned.

5.6.5 Time Complexity Comparison: For quiet some time now, RSA is the most used and preferred public key encryption scheme for its security and simplicity. Many group signatures proposed till now are based on strong RSA assumption [24, 25]. There are others based on the Diffie-Hellman assumption or bilinear pairings [22,23]. In this section we compare the time complexity of our algorithm

with that of RSA. We assume 1024 bit for RSA and same before-mentioned specifications for our scheme.

The basic operation in RSA is modular exponentiation modulo the 1024 bit key. The generation and verification of signature employ similar operations hence we assume similar complexity for them.

RSA signature generation/verification complexity:

Generation: $Y = M^e \bmod N$ Verification: $M = Y^d \bmod N$

Complexity: $O(\lg(e)\lg(N)^2)$ Complexity: : $O(\lg(d)\lg(N)^2)$

	RSA	GSCRT
Signature Generation	$O(\lg(e)\lg(N)^2)$	$O((k'b)^2)$
Signature Verification	$O(\lg(d)\lg(N)^2)$	$O(k'b^2)$

Table 5.2 Time Complexity Comparison with RSA

The generation and verification complexity of RSA goes to $O((\lg(N))^3)$, if e and d are O(N). In comparison with our scheme $k'b = O(\lg(N))$, leading to the fact that signature generation in GSCRT is comparable to that in RSA while it performs better than RSA when it comes to signature verification. As explained earlier lower verification time is suitable for VANET. Both provide the same security as in both cases the security comes down to factorizing an integer with two large prime factors (512 bits each).

5.7 Conclusion and Future Work

In this chapter we presented the modified GSCRT group signature scheme which performs much better than the original GSCRT scheme. The weakness of the algorithm towards coalition resistance still persists. Also the algorithm does not seem to be very scalable if the number of cars increases drastically. In that case

the vehicles need to be redistributed into new groups which will require lot of effort. Redistribution will become unavoidable as with increase in the number of vehicles the size of the parameters involved will become too big to be acceptable. Another point to be considered is that all analyses here have been done keeping the worst case in mind. The size of Y, in reality and in general will be much less (even half) than the stated size.

The link between the theoretical security of an algorithm and the practical security required in VANET is uncertain to say the least. For example researchers around the world have said enough about the anticipated presence of Tamper Proof Devices in the vehicles to act as cryptographically secure storage for secret information. If this is truly the case then the underlying algorithm may actually stop worrying about coalition-resistance altogether as if the secret information cannot be extracted, they of course cannot be shared as well.

This algorithm presents something that is quiet against the norms prevalent in our times. Almost all public key algorithms coming are based on much more complex mathematical problems as compared to Chinese Remainder Theorem. We believe that to meet stringent efficiency requirements of VANET we will have to look beyond conventional methods and schemes. In its current form GSCRT is too raw to be accepted as a candidate to provide security in VANET, but it certainly does throw light on new areas and possibilities which are there to be explored.

Bibliography

1. M. Bellare and C. Namprempre., *Authenticated encryption: Relations among notions and analysis of the generic composition paradigm*, In Advances in Cryptology– ASIACRYPT '00 (2000), vol. 1976 of Lecture Notes in Computer Science, Springer-Verlag.

2. Ferguson, N., Whiting, D., Schneier, B., Kelsey, J., Lucks, S., and Kohno, T., *Helix: Fast encryption and authentication in a single cryptographic primitive*, In Fast Software Encryption, 10th International Workshop, FSE 2003, T. Johansson, Ed., Lecture Notes in Computer Science, Springer-Verlag.

3. I. Damgard, *A Design principle for hash functions*, Advances in Cryptology, *CRYPTO' 89*, LNCS 435, pp. 416–427, Springer-Verlag, 1990.

4. Daemen, J., Govaerts, R., Vandewalle, J., *A framework for the design of One way Hash function Including Cryptanalysis of Damgard's One-Way Function Based on Cellular Automaton*, ASIACRYPT 1991, pp. 82-96.

5. Monalisa Mukherjee, Niloy Ganguly, and P. Pal Chaudhuri, *Cellular Automata Based Authentication*, Proceedings of the 15th international conference on Computer communication, 2002.

6. McGarry, B., Gage, D., Laub, E., *Is Rule 30 Random?*, 2005 Midwest NKS Conference.

7. C.K. Koc and A.M. Apohan, *Inversion of cellular automata iterations*, IEE Proc. Comput. Digit. Tech., vol. 144, pp. 279-284,1997.

8. M.Raya and J.-P.Hubaux, *Securing vehicular ad hoc networks*, in: Journal of Computer Security 15(2007),39-682005,pp.11–21.

9. http://www.car-2-car.org/.

10. J.Blum and A.Eskandarian , *The threat of intelligent collisions*, IT Professional 6(1)(2004)

11. Y.C.Hu and A.Perrig, *A survey of secure wireless ad hoc routing*, IEEE Security & Privacy 2(3)(2004),28–39.

12. K.Sampigethaya, L.Huang, M.Li, R.Poovendran, K.Matsuura and K.Sezaki, *CARAVAN:Providing location privacy for VANET*,in:Proceedings of the Workshop on Embedded Security in Cars (escar)'05, 2005.

13. J.-P.Hubaux,S.Capkun and J.Luo, *The security and privacy of smart vehicles*, IEEE Security and Privacy Magazine 2(3)(2004),49–55.

14. M.Gerlach, VaneSe, *An approach to VANET security*, in: Proceedings of V2VCOM'05, 2005.

15. L.Gollan and C.Meinel, *Digital signatures for automobiles*, in: Proceedings of Systemics, Cybernetics and Informatics (SCI)'02, 2002.]

16. M.ElZarki, S.Mehrotra, G.Tsudik and N.Venkatasubramanianm, *Security issues in a future vehicular network*, in: Proceedings of European Wireless'02, 2002.

17. B.Parno and A.Perrig, *Challenges in securing vehicular networks*, in: Proceedings of the Workshop on Hot Topics in Networks (HotNets-IV), 2005.

18. G.Calandriello, P.Papadimitratos, J.-P Hubaux, A.Lioy, *Efficient and robust pseudonymous authentication in VANET*, in: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks, Canada (2007),19-28

19. IEEE 1609.2.IEEE trial-use standard for wireless access in vehicular environments-security services for applications and management messages,July2006.

20. M.Gerlach, A.Festag, T.Leinmuller, G.Goldacker, and C.Harsch, *Security architecture for vehicular communications*. In WIT2005, Hamburg, Germany.

21. P.Papadimitratos, L.Buttyan, J-P.Hubaux, F.Kargl, A.Kung and M.Raya, *Architecture for secure and private vehicular communications*. In ITST'07,Sophia Antipolis ,France

22. D.Boneh, X.Boyen, and H.Shacham, *Short group signatures*, 2004.

23. D.Boneh and H.Shacham, *Group signatures with verifier-local revocation*, in CCS'04, pages 168–177, New York, NY, USA, 2004, ACM Press.

24. J. Camenisch, M. Michels, *A Group Signature Scheme Based on an RSA-Variant*, 1998

25. N.Baric and B.Pfitzman, *Collision-free accumulators and fail-stop signature schemes without trees*, In W.Fumy,editor, Proceedings of Euro crypt 1997, volume 1233 of LNCS, pages 480–494, Springer-Verlag, May 1997.

26. M.Bellare, D. Micciancio and B. Warinschi, *Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions*, Advances in Cryptology - Eurocrypt 2003 Proceedings, Lecture Notes in Computer Science Vol. 2656, E. Biham Ed, Springer-Verlag, 2003.

27. M.Bellare, H.Shi and C.Zhang. Foundations of Group Signatures: *The Case of Dynamic Groups*, Topics in Cryptology - CT-RSA 2005 Proceedings, Lecture Notes in Computer Science Vol. 3376, A. Menezes ed, Springer-Verlag, 2005.

28. X.Zhou, X.Yang, P.Wei and Y.Hu, Dynamic *Group Signature with Forward Security and Its Application*, in: Proceedings of the Sixth International Conference on Grid and Cooperative Computing (2007), 473-480

29. Lecture 7, http://www.cs.uwaterloo.ca/~watrous/lecture-notes.html

30. Xukai Zou , Byrav Ramamurthy , Spyros S. Magliveras, *Chinese Remainder Theorem Based Hierarchical Access Control for Secure Group Communication*, Proceedings of the Third International Conference on Information and Communications Security, p.381-385, November 13-16, 2001