12-2010

# A Secured Data Protocol for the Trusted Truck(R) System

Srinivasa Anuradha Bulusu
*University of Tennessee - Knoxville*, sbulusu@utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Srinivasa Anuradha Bulusu entitled "A Secured Data Protocol for the Trusted Truck(R) System." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Itamar Arel, Major Professor

We have read this thesis and recommend its acceptance:

Benjamin Arazi, Gregory Peterson, Hairong Qi

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:
I am submitting herewith a thesis written by Srinivasa Anuradha Bulusu entitled "A Secured Data Protocol for the Trusted Truck(R) System." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Itamar Arel, Major Professor

We have read this thesis and
recommend its acceptance:

Benjamin Arazi

Gregory Peterson

Hairong Qi

Acceptance for the Council:

Carolyn R. Hodges,
Vice Provost and
Dean of the Graduate School

(Original signatures are on file with official student records.)

# A Secured Data Protocol for the Trusted Truck(R) System

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Srinivasa Anuradha Bulusu
December 2010

# Dedication

This thesis is dedicated to my beloved parents, brother and my dearest friends.

# Acknowledgments

Firstly, I am grateful to my graduate advisor, Dr. Itamar Arel and my research advisor Dr. Benjamin Arazi for giving me this opportunity to work on this project and also for supporting me through my Masters. Their patience, encouragement and valuable guidance helped me learn a lot during my research.

I would like to thank my committe members Dr. Gregory Peterson and Dr. Hairong Qi for reviewing my thesis.

I express my thanks to all my professors for influencing me and helping me in achieving my goals. Thank you to those in Machine Intelligence Laboratory especially Scott Livingston for his help and suggestions in my research work.

Finally, I owe my heartfelt regards for my family who have constantly been my force of inspiration, determination and encouragement. I extend my special thanks to all my friends for their care and support which has helped me through this ardous task.

# Abstract

Security has become one of the major concerns in the Intelligent Transportation Systems (ITS). The Trusted Truck(R) System, provides an efficient wireless communication mechanism for safe exchange of messages between the moving vehicles (trucks) and the roadside inspection stations. The vehicles and the station are equipped with processing units but with different computational capabilities. To make this Trusted Truck(R) system more secure, this thesis proposes a secured data protocol which ensures data integrity, message authentication and non-repudiation. The uniqueness of the protocol is: it is cost-effective, resource-efficient and embeds itself into the Trusted Truck(R) environment without demanding any additional infrastructure. The protocol also balances the computational load between the vehicle and station by incorporating an innovative key transport mechanism. Digital signatures and encryption techniques are used for authentication and data confidentiality. Cryptography algorithms along with optimization methods are used for the digital signatures. The computational time for the algorithms are analyzed. Combining all these techniques, an efficient secured data protocol is developed and implemented successfully.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Background

The US Department of Transportation (US DOT) conducts nearly 700 thousand inspections per year on commercial vehicles.[**?**] With the increasing population and traffic, this number keeps on increasing. It is the US Department of Transportation's responsibility to ensure that the carriers are complying with the safety regulations without affecting the time and profitability of the carriers. The Intelligent Transportation System (ITS) utilizes modern technologies and systems to improve the current transportations systems of all kinds in the US. The Trusted Truck(R) project is one such project which helped the US DOT in speeding up the carrier verification by providing a wireless communication link to verify the credentials of the vehicle that adheres to ITS standards. The Trusted Truck(R) Program was initiated in 2003 as a joint effort by Volvo Group North America, National Transportation Research Centre, Inc.,(NTRCI), and The University of Tennessee. The main vision behind the Trusted Truck(R) is to provide safe and efficient truck transportation. Presenting and verifying the credentials happen even before the truck approaches the inspection office. Thus a lot of time is saved for the truck as well as the inspectors who can then concentrate more on the safety violations. The concept behind the Trusted Truck(R) system is that once a truck is verified and confirmed as 'trusted', it can bypass other weigh stations. Along with the safe transportation, it is also capable of providing secured communication via the wireless communication link.

Information security forms an important aspect of secured wireless communication between

vehicle and road side inspection stations. Regardless of who is involved, all parties in the transaction must have confidence that certain information security objectives have been met. Some of the information security objectives are: data confidentiality, authentication, authorization, revocation, non-repudiation, etc. For decades, cryptography is in use for information security or secured communication. Cryptography is a widely used tool in communications, computer networks and computer security. Modern cryptography can be divided into two classes: symmetric key cryptography and public key cryptography. The former uses the same key for encryption and decryption whereas the later uses two different keys: one public and the other private. RSA is a public key cryptography which has been proved successful in secured message transfers. RSA uses a pair of keys for encryption and decryption and also for digital signatures. Digital signatures are used to verify the authenticity of a particular sender. In most of the secured communications, RSA is used for digitally signing and transferring the secret key which is used to encrypt the message. After successful verification of the authenticity of the sender, the secret key is extracted and the message is decrypted.

## 1.2 Motivation

Cryptography has been proven successful for ages for data confidentiality and authentication in many security applications. Along with the advancements in cryptography algorithms, its areas of application also increased. In the beginning military applications used methods of encryption for exchanging confidential data. Now-a-days many other applications and sectors like banking, e-mail services, social networking, and even transportation systems greatly rely on cryptography algorithms to maintain security standards. Intelligent Transportation Systems (ITS) have been widely accepted as an advanced technology for providing efficient transportation using intelligent systems. Security is an important aspect of ITS standards. Security in transportation lies in the level of data confidentiality and safe exchange of messages between the systems. Data security using cryptography can be done in: Symmetric Key Cryptography and Asymmetric Key Cryptography. Symmetric key is efficient but not strong and asymmetric key is easy to use but demands long keys. The cryptography algorithms are discussed in detail in the next chapters.

Trusted Truck(R) System has been successful in demonstrating an efficient system which provides a safe wireless communication link between the vehicles ( heavy vehicles like trucks) and road side inspection stations. The vehicles and the stations exchange the required credentials through this link which saves the effort and time of the inspection stations. The advancements in truck sensor technologies also helps the vehicles exchange the information related to the physical condition of the vehicle. For the Trusted Truck(R) system to be secure, there should be a data security infrastructure so that the messages are exchanged *only* between the trusted parties. The necessity for a robust data security infrastructure for such a system initiated the development of a resource-efficient secured data protocol for the exchange of messages between the vehicle (truck) and the inspection station. Asymmetric cryptography along with digital signatures can be used to protect the integrity of the data, its originator or sender, and to achieve non-repudiation. In an environment where critical information is exchanged between the vehicles and the stations, there is always a threat of data manipulation, tampering of vehicle sensors, and also identification of the participant. Using an asymmetric encryption algorithm and a safe key transport channel, the security standards can be achieved [20].

This thesis presents a cost-effective and resource-efficient secured data protocol which perfectly fits in the Trusted Truck(R) System to provide data confidentiality and message authentication. The thesis also introduces an innovative way to use and enhance the speed of the existing asymmetric key cryptography algorithm to balance the computational load between the vehicle and the road side inspection station. The designed protocol uses RSA for the digital signatures and DES for encryption and decryption. The DES key is transported safely through a key-transport mechanism using RSA. The mathematical explanation of the algorithms are explored in this thesis and required speed up is achieved intelligently.

## 1.3 Thesis Outline

Chapter 2 gives a brief review of concepts of public key cryptography and digital signatures It deals with two algorithms in particular, which are required for the understanding of this thesis. It also examines the modular arithmetic which is used at each and every stage of this protocol design.

Chapter 3 introduces the new designed communication mechanism and how it provides the required security for the exchange of messages. It also describes the existing communication system and how the new mechanism fits into the earlier system. Considering the available resources, a step-by-step procedure for the design of new mechanism is discussed. Technical details of the protocol are explained along with the duties of each and every participant in the communication system.

Chapter 4 explains in detail the design of the protocol from a mathematical perspective. The protocol is developed in ANSI C, compatible with all the systems in the Trusted Truck(R) Environment. Various modular arithmetic operations and algorithms are discussed related to the design of the protocol. It also discusses the major challenges faced during the design phase of the system and how they are solved. Speed-up algorithms are used to reduce the computation time for the public key cryptography systems and also to reduce the computational load on the participants. This chapter also examines how minor changes to the execution of modular exponentiation save time and effort which is essential for the successful operation of the protocol.

Finally, chapter 5 concludes the thesis by providing some scope for the future work with some additions to the existing system.

# Chapter 2

# Literary Review

## 2.1 Cryptography Algorithms Explored

In this thesis, two famous cryptographic algorithm are used: RSA, a digital signature scheme with message recovery option, and Data Encryption Standard (DES), a standard approved by NIST [7] for encryption and decryption. Generally, when a message has to be exchanged between two parties, RSA is used to safely transmit the DES key and further DES is used to send or receive encrypted data.

### 2.1.1 Digital Signatures - RSA

Digital Signature, introduced by Diffie and Hellman in 1976 [8][9] is a scheme used in authenticating a particular sender so that the recipient is assured that the message is sent from a known sender and is not altered in between. Any digital signature scheme will have two stages: signature generation and a corresponding verification algorithm. Public-key cryptography algorithms can be used for digital signatures. RSA is one such public-key algorithm where either the public or private can be used for encryption. Another cryptography algorithm used for digital signature is the Digital Signature Algorithm (DSA) which cannot be used for encryption [1][10].

RSA, named after its creators Rivest, Shamir and Adleman [11][12], is the first known public-key algorithm used for signing as well as encryption. The security of RSA lies in the difficulty of factoring the large numbers. The public and private keys are a function of a pair

of large prime numbers. Recovering the plaintext from the cipher text is considered equivalent to factoring the product of these prime numbers. Two random large prime numbers $p$ and $q$ are selected. For maximum security $p$ and $q$ are chosen to be of equal length. The product of $p$ and $q$ is computed:

$$n = pq \tag{2.1}$$

The encryption key $e$ is chosen such that $e$ and $(p-1)(q-1)$ are relatively prime. The decryption key is computed using the Euclidean algorithm such that,

$$ed \equiv 1(\mathrm{mod}(p-1)(q-1)) \tag{2.2}$$

Now $d$ and $n$ are also relatively prime. The numbers $e$ and $n$ form the public key pair; the number $d$ is the private key. The message $m$ is broken down into blocks $m_i$ smaller than $n$. The encrypted message $c$ is also made of similarly sized blocks $c_i$. The encrypted message is simply,

$$c_i = m_i^e \bmod n \tag{2.3}$$

and the decrypted message is computed as,

$$m_i = c_i^d \bmod n \tag{2.4}$$

The message can also be encrypted with $d$ and decrypted with $e$ In this thesis we used $d$ for encryption and $e$ for decryption. RSA is usually slower than symmetric algorithms but speed up can be obtained in various ways. One such way is choosing the value of $e$. As we see the equation (2.3) is a modular exponentiation, the value of $e$ decides the number of multiplications in the equation. The three most common values of $e$ are 3, 17, and 65537. The Internet Privacy Enhanced Mail Standard (PEM) and the Public-Key Cryptography Standard (PKCS) recommend 3 and 65537 [13][14]. In [1], it is explained that when the messages are not identical, a smaller value of $e$ (*equal to* 3) will not allow any Low Exponent Attacks [2]. And also with a larger value of $d$ and a small value of $e$, the attack mentioned by Micheal Wiener in [15] cannot occur. It has also been proved in many papers that even recovering a few bits of information from the RSA encrypted cipher text is as hard as factoring $n$. We used 64 *byte*

prime numbers to generate the public and private keys in this thesis, so all the RSA encrypted messages are believed to be very secure as factoring a 128 *byte* number is very difficult. Private key operations can be speeded up by using Chinese Remainder Theorem which needs calculation of some additional values other than $n$ and $d$ [16].

### 2.1.2 Encryption and Decryption - DES

DES, one of the most frequently used cryptographic algorithms, was released in 1977 by the National Bureau of Standards (NBS) , now The National Institute of Standards and Technology (NIST). DES is a block cipher which means it breaks the message (or plaintext) into blocks of 64 *bits* and encrypts each block separately. The mechanism for doing this is called Feistel System, after Horst Feistel [1].

In DES, a block of cipher consists of 64 *bits*. The key has 56 *bits* expressed as 64 *bits*. Every $8^{th}$ *bit* is a parity bit and is used for error detection purpose, leaving 56 *bits* for the actual key. The output of the encryption is a 64-*bit* cipher text. Encryption is carried out in 16 stages. From the input key (say $K$), sixteen 48-*bit* subkeys $K_i$ are generated, one for each stage. The 64-*bit* plaintext is divided into 32-*bit* halves $L_0$ and $R_0$. Each round is functionally equivalent to taking 32-*bit* inputs $L_{i-1}$ and $R_{i-1}$ from the previous stage and producing $L_i$ and $R_i$ for all $i$ from 1 to 16 as,

$$L_i = R_{i-1} \tag{2.5}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \tag{2.6}$$

where $f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$,

$P$ is a fixed permutation on 32 *bits*,

$E$ is a fixed expansion permutation mapping $R_{i-1}$ from 32 to 48 *bits*,

$S$ is the 6 to 4 *bit* substitution mapping box,

$K_i$ is a string of 48 *bits* obtained from the key $K$.

Figure 2-1 show the detailed process of a DES algorithm. Initially, the message $m$ is permuted by a fixed permutation (IP). Then the left and right halves are exchanged and finally the inverse of the initial permutation is applied to get the cipher text $c$. Decryption is performed in the same way but the keys $K_1, K_2, ...., K_{16}$ are used in the reverse order. The function

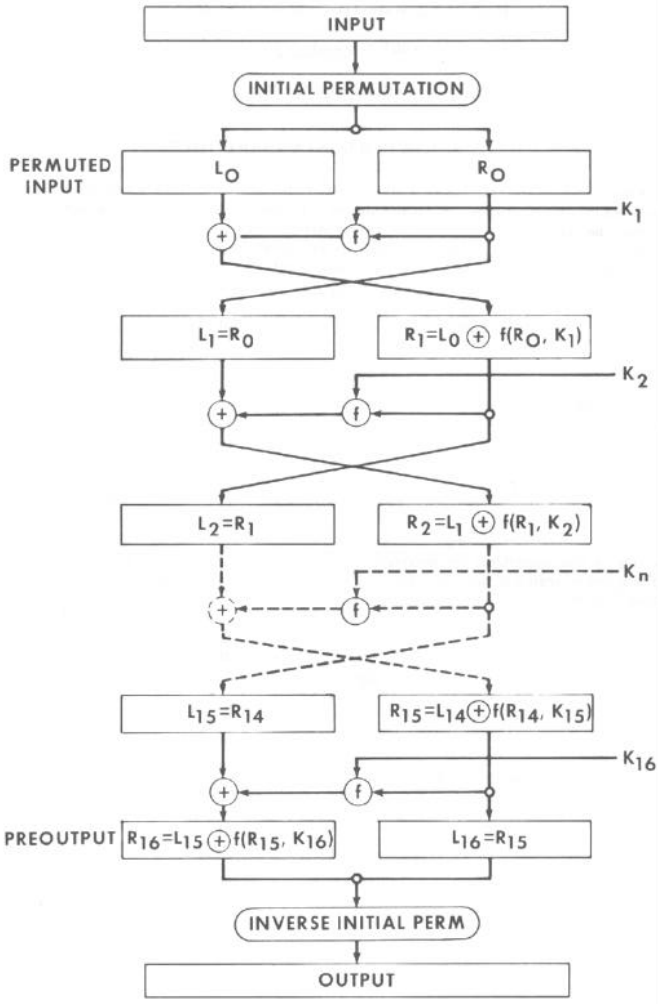


Figure 2-1: Data Encryption Standard Computation Path

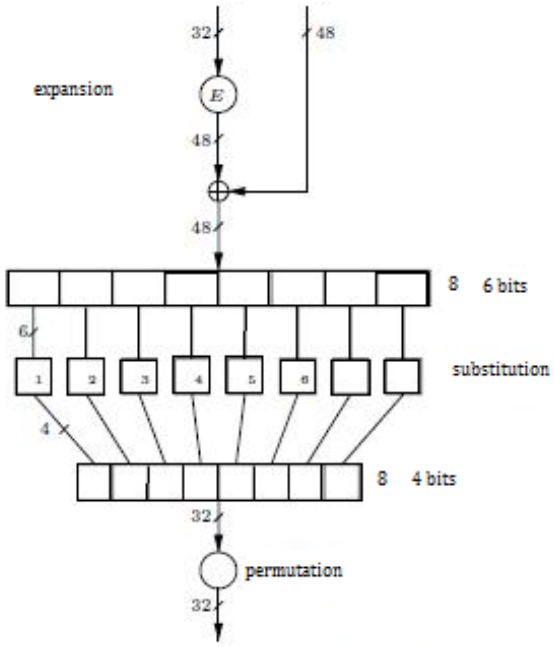


Figure 2-2: DES function

$f(R_{i-1}, K_i)$ is shown in figure 2-2. The Initial Permutation is done according to the table in [6]. The expansion permutation and the substitution mapping boxes ($S$-boxes) are shown in detail in [6]. Each bit of the key is used in approximately 14 out of 16 rounds. The permutation tables are designed in such a way that each bit of the cipher text should be dependent on all the bits of plaintext. But practically this happens only in few rounds. The $S$-boxes are considered to be the heart of the algorithm and they are the key factors for security [2]. So many constraints have been laid down in constructing the $S$-boxes [17]. Block ciphers can also be executed in many different modes as in the real world the message would be either larger than block size or smaller than block size. Many modes have been designed for the block ciphers: Electronic Codebook Mode (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Counter Mode (CTR). Detailed explanation for each and every mode can be found in [2]. In this thesis, we used the ECB mode for the encryption and decryption.

## 2.2 Modular Arithmetic Revisited

Modular Arithmetic is one of the widely used concepts of number theory in the areas of cryptography. In cryptography, modular arithmetic is often used in symmetric key algorithms like AES, IDEA [6], and public key systems like RSA. The theory of modular arithmetic is widely dependent on congruence relations. Linear system of congruences can be solved in polynomial time using modular arithmetic. In algorithms like RSA where operations like exponentiation on large numbers are involved, modular arithmetic techniques help in computing exponentiation *modulus* $n$ efficiently. In modular arithmetic, $b$ mod $a$ gives the remainder when $b$ is divided by $a$. Talking in terms of congruence, $a$ and $b$ are said to be congruent modulo $n$, when

$$a \equiv b \pmod{n} \quad n > 0 \tag{2.7}$$

This says that $a$ and $b$ leave the same remainder when divided by $n$. Properties like transitivity, reflexivity, and symmetry apply to congruence relation [6]

### 2.2.1 Modular Addition and Subtraction

Modular addition or subtraction does not involve any long division. The numbers are initially summed up and later on divided to get the remainder. Let $n$ be any positive integer, $a$,$b$ be any non-negative integers, then,

$$(a+b) \bmod n = a \bmod n + b \bmod n = \begin{cases} a+b, & if\ a+b < n \\ a+b-n, & if\ a+b \geq n \end{cases} \tag{2.8}$$

Addition and subtraction have a bit complexity of $O(\lg n)$.

### 2.2.2 Modular Multiplication and Division

Modular multiplication and division and complex in nature compared to addition (or subtraction). The computation becomes complex when larger numbers are involved. Let $a$ $(n+1$ $bits)$ and $b$ $(t+1$ $bits)$ be integers expressed in binary notation: $a = \{a_n a_{n-1}....a_1 a_0\}$ and $b = \{b_t b_{t-1}....b_1 b_0\}$. The product of $a$ and $b$ will have at most $(n+t+2)$ $bits$. Multiplication

is done by the simple multiply-and-add method used in integer theory. The bit complexity of multiplication is $O((\lg n)^2)$.

Modular division is the most complicated and time consuming operation in modular arithmetic. Division is done by using the shift-and-subtract method. The bit complexity of division is again $O((\lg n)^2)$. It is shown that the complexity of multiplication and division can be reduced by using techniques like Montgomery Reduction [6]. Complete algorithms for multiplication and division can be found in [1][6]

### 2.2.3 Modular Exponentiation

Modular Exponentiation is crucial for many cryptographic algorithms. In the later sections, it can be observed that most of the cryptographic algorithms involve modular exponentiations at various stages. The public key system RSA which is the key factor for this thesis uses exponentiation for digital signatures as well as encryption. Let $a$, $k$,and $n$ be non negative integers represented in binary form, the modular exponentiation $a^k \bmod n$ can be computed using the repeated square-and-multiply algorithm [6]. In this algorithm exponentiation is computed by performing a series of square and multiply operations. It is also called binary exponentiation as it uses the binary expansion of the exponent $k$. Algorithm 2.1 gives a brief description of the repeated square-and-multiply algorithm. Here, $a$, $n$ are very large integers and $k$ is an integer exponent represented in binary form $\{k_0 k_1 ....k_{t-1} k_t\}$ with $k_0$ as MSB $k$ can also considered as a bit string of length $t$ denoted as $k[1..t]$. In this thesis the value of $t$ goes up to maximum 1024 *bits*. This representation of exponent $k$ is a way to reduce the number of multiplications in the exponentiation computation. Various representations have been proposed [18] to reduce the multiplications.

**Algorithm 2.1: Repeated Square-and-Multiply Method for Modular Exponentiation**
INPUT: $a$, $n$, and $k$
OUTPUT: $a^k \bmod n$
1. Begin
2. Set $b \longleftarrow 1$
3. **IF** $k = 0$ **THEN** return $(b)$.
4. Set $A \longleftarrow a$.
5. **IF** $k_0 = 1$ **THEN** set $b \longleftarrow a$.
6. **FOR** $i$ from 1 to $t$ do the following:
   6.1. Set $A \longleftarrow A^2 \bmod n$.
   6.2. **IF** $k_i = 1$ **THEN** set $b \longleftarrow A * b \bmod n$.
7. **END FOR**
8. Return $(b)$.
9. End

### 2.2.4 Chinese Remainder Theorem

The Chinese Remainder Theorem states that "*Suppose* $\gcd(m, n) = 1$. *Given integers a and b , there exists exactly one solution* $x \pmod{mn}$ *to the simultaneous congruences* $x \equiv a \pmod{m}$, $x \equiv b \pmod{n}$*"*[2]. Generally, if the prime factorization of $n$ is $p_1 * p_2 * p_3 ... * p_t$, then the equations

$(x \bmod p_1) = a_1$

$(x \bmod p_2) = a_2$

.

.

.

$(x \bmod p_t) = a_t$

have a unique solution, $x$, where $x < n$. Thus consider $n = p * q$ (as in RSA), then for any arbitrary $a$ and $b$, there exists an unique solution $x < p * q$, such that

$$x \equiv a \pmod{p}, \text{ and } x \equiv b \pmod{q} \tag{2.9}$$

Then Euclid's algorithm, $u$ is calculated such that,

$$u * q \equiv 1 \pmod{p} \tag{2.10}$$

then the solution $x$ is computed as,

$$x = [\{(a - b) * u\} \bmod p] * q + b \tag{2.11}$$

The Chinese Remainder Theorem (CRT) is used in the computation of the modular exponentiation $a^k \bmod n$ where $n$ can be written in terms of prime factors $p$ and $q$ ($p$ and $q$ are two large prime numbers). In RSA, the exponent $k$ is also obtained from the prime numbers. So it can also be written as a function of the prime numbers. One of the usual solutions to the exponentiation is to compute small integers that are congruent modulo $n$ to various powers of $k$ [19]. In equation (2.4), public key $n$ and private key $d$ are both obtained from the prime numbers $p$ and $q$. By calculating some additional values along with the keys, CRT can be applied to compute the equation making it much faster than traditional modular exponentiation algorithm ( repeated square-and-multiply). Additional values required are $dp = d \bmod (p - 1)$, $dq = d(\bmod q - 1)$, and $q^{-1} \bmod p$. These values are calculated along with the private key $d$.

## 2.3 Trusted Truck(R) Environment

The Trusted Truck(R) Program was initiated in 2003 as a joint effort by Volvo, NTRCI and The University of Tennessee. The aim of the Trusted Truck(R) program is to develop a secure and trusted transport solution from pickup to delivery. The program's objective is to increase safety, security, and efficiency of truck transportation by presenting wireless credentials to roadside inspectors that confirm that the tractor, trailer and cargo meet all appropriate requirements for safe transportation of the cargo. By presenting these credentials without the need for the truck to stop, the number of inspections increase, the efficiency of the system improves, and inspectors can have more time to target more likely safety and security violations. More information on the Trusted Truck(R) System can be found at *http://www.ntrci.org/*

# Chapter 3

# Technical Approach

This section describes the selection of cryptographic algorithms and the designed communication mechanism for the secured data exchange between moving vehicle and the weigh station.

## 3.1 Objective

The existing Trusted Truck(R) system provides a successful wireless communication link between the moving vehicle and the weigh station for credential verification. For the system to be more secure, there is a need for the wireless link to be more robust in design as well as in security related aspects. Thus, design of a secured data protocol has been proposed. The main objective of this approach is to provide an efficient, less-complex security protocol to be used for the secured data communication among the moving vehicles in the Trusted Truck(R) system. The fundamental step to achieve this objective is to select the appropriate algorithms and use them efficiently.

## 3.2 Requirements and Resources

For the system to be termed as secure, the application must be able to trust that the communication has been received unaltered and from a known source. This requires the ability to provide data confidentiality and authentication. In this thesis we tried exploring the cryptographic algorithms to provide authentication and confidentiality in an efficient and innovative

way. The other requirement is the efficient usage of resources and to minimize the computation time.

In the Trusted Truck(R) system, the possible threats involving the data confidentiality are

- Vehicle, truck and cargo identification
- Tampering or replacing of sensors
- Impersonation of attacks

The developed protocol should balance the asymmetry between the computational resources at the vehicle (truck) and at the inspection stations (weigh station). Thus, in this thesis, apart from merely applying the cryptographic algorithms, analysis has been done on the computational time for each algorithm and the most appropriate one is being used. Also, the computational load on the vehicle side is reduced when compared to that on the inspection station. The vehicles and the inspection stations are not expected to have the state-of–the-art equipment. Additionally, the security protocol has to fit in the existing Trusted Truck(R) infrastructure.

## 3.3 Algorithms

Cryptography has been in use for ages to maintain the confidentiality of data and its authentication in many systems which operate on highly secured data. After a considerable amount of research, Public Key Cryptography has been selected as it proved to be more advantageous and robust in key management when compared to symmetric-key cryptography where the same key is used for the encryption and decryption[1]. In Public Key Cryptography (PKC), a pair of keys, public and private, are used to encrypt and decrypt the message. The public key is so constructed that the calculation of the private key is computationally infeasible from the public key, and vice versa[1]. RSA is one such public key cryptosystem which offers both encryption and digital signature[2]. In RSA, anyone can send an encrypted message or verify a signed message, but only someone in possession of the correct private key can decrypt or sign a message. In the Trusted Truck(R) System, the vehicle and the inspection station should mutually authenticate each other before the actual exchange of messages. Thus, the RSA algorithm is used to serve the purpose of authentication in the form of digital signatures. And for encryption

and decryption, another well known cryptographic algorithm called the DES[3] is used. The DES is the basic building block for data protection. It uses a 56-bit key to transform a 64 bit input into a 64 bit encrypted output[6]. Both the cryptographic algorithms selected involve high ordered computations on the data.

## 3.4 Designed Communication Mechanism

### 3.4.1 Basic Communication Mechanism

The Trusted Truck(R) System involves three main entities – Vehicle (truck), Station (departing station, weigh station, arriving station) and the Certificate Authority. The departing station is the point at which the vehicle starts from, the weigh station is the inspection station on the highway and the arriving station is the destination location of the vehicle. In addition to these we have the driver and cargo information which are crucial in the vehicle identification at the departing station. The protocol is designed such that the identification and authentication mechanisms are embedded in the messages that are transferred between the entities. The vehicle is equipped with an On Board Unit (OBU) which is used for the software implementation on the vehicle. Likewise the stations have a Road Side Unit (RSU). The Certificate Authority (CA) is an entity which generates the digital certificates for use by the participants. The CA generates the digital certificate that contains the public key and other identities of the vehicle/station. The certificate generated by the CA is also like an attestation that the public key contained in the certificate belongs to the vehicle/station. Here are the basic checks performed at different levels while the vehicles originate from the departing point and reach the destination.
At the departing station,

- The driver and the cargo are verified. Along with this, the physical condition of the truck is also noted. Sometimes the departing station may even have a biometrics application which verifies the driver by his/her fingerprint
- Once the credentials are verified, the station grants permission for the vehicle to leave the departing station

At the weigh station,

- The vehicle on approaching the weigh station exchanges the above collected data using wireless communication link (GPRS)
- The weigh station verifies the details and upon successful verification, allows the vehicle to proceed further

At the arriving station(final destination),

- Upon the arrival of the vehicle once again the credentials of the vehicle are verified and the physical condition of the vehicle is checked
- If the verification process is successful, the cargo is received and message is transferred to the vehicle indicating the reception of the cargo

The secured message transfer mechanism is designed such that it does not demand an additional step other than the above mentioned checks, that is, it does not require a step in between the vehicle and the station for the security related checks. This is done by embedding the cryptographic algorithms within the messages that are exchanged between the vehicle and the stations. As the initial step, the communication mechanism with the secured protocol included is designed. The following sections will discuss this in detail.

### 3.4.2 Modified Communication Mechanism

Figure 3-1 shows the designed communication process or the message flow between the vehicle and the station.

Before actually initiating the communication process, the Certificate Authority (CA) generates and approves all the public and private keys for the vehicle as well as the station. The CA is considered as an entity which is credible by all parties wishing to communicate. The CA generates the set of public and private keys for each and every vehicle separately and also for the stations individually. It also generates the keys for itself. The CA's public and private keys are used in the digital signatures.

Once the key generation is complete, each vehicle and each station are loaded with their corresponding public and private keys. The CA also distributes its public key to all the vehicles and the stations. The CA also digitally signs the vehicle/station IDs, expiration date and their

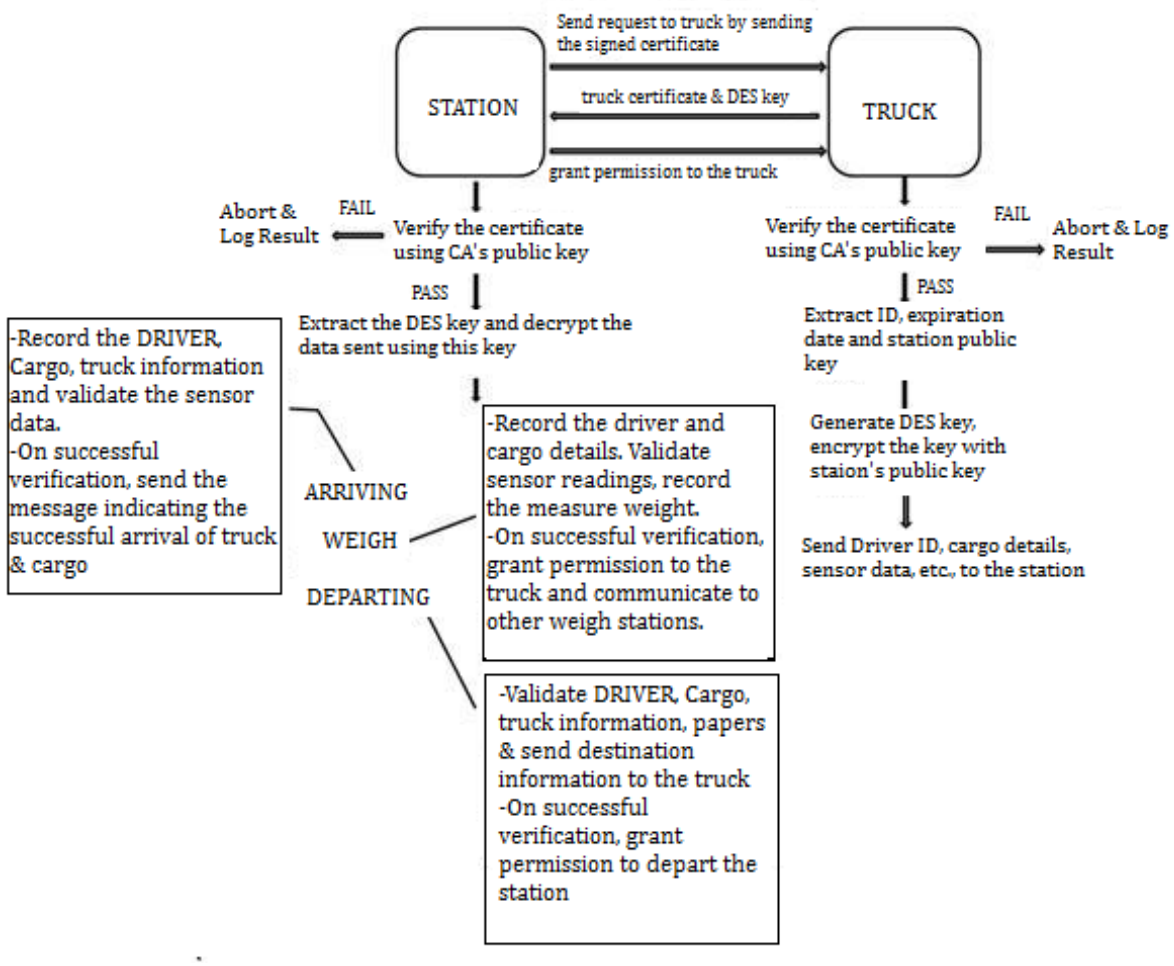


Figure 3-1: Communication Process / Message Flow

public keys using CA's public key. Expiration date is used for verifying the validation of the vehicle/station. ID represents the unique Identification Number given to the vehicle or the station. Along with these keys, the vehicle collects and stores all the sensor readings,and cargo details in the On Board Unit. With all the details ready, the truck initiates the communication process with the station. The nature of the message exchanged depends on the type of station (departing/weigh/arriving) it is communicating with. Here are the series of messages that are exchanged between the vehicle and the station after initiating the communication process,

- The vehicle receives the station's certificate and using the preloaded CA's public key, it verifies the station's certificate. If all the credentials (ID, Expiration Date) are valid,

the vehicle extracts the station's public key from the certificate. This assures the vehicle that it is communicating with the correct station and not any intruder. After getting the station's public key, the vehicle uses this public key to encrypt the unique 'Session Key' and send the same to the station along with its certificate using the wireless link. The Session Key is the key used to encrypt the vehicle details like the driver ID, cargo details, sensor readings and other related information using the DES algorithm. This key is unique and is generated each time the vehicle communicates with the station. Thus no two stations will receive the same session key. If on the other hand, the station's details fail to verify, the vehicle sends the invalid message and logs the result

- Upon receiving the vehicle's certificate the station extracts the ID, the Expiration Date of the vehicle, using the CA's public key. If the received details are valid, the station proceeds to the next set of messages. The station extracts the session key from the vehicle's certificate using its private key and decrypts the vehicle details (ID, cargo details, sensor readings and other related details) using the session key. If any of these details fail to verify, the vehicle is not granted permission to proceed further and the result is logged

- The nature of messages depends on the station with which the vehicle is communicating

This modified communication mechanism as promised earlier, delivers all the required stages which verify the identity of both the parties at each and every stage and at the same time maintains the confidentiality of the data through encryption at every stage. Even if an intruder tries to manipulate the messages, the original data behind the encrypted text cannot be extracted under any means as the intruder will not have the key details of the vehicle/station. This mechanism does not demand additional hardware or software and uses the same basic communication messages to establish the security.

The next step in the design of this protocol is to balance the computational burden on the vehicle as well as the station. The vehicle takes more time to compute the cryptographic algorithms when compared to the station, as it is generally equipped with lower end systems when compared to the station. To overcome this, an innovative way of utilizing the travel time has been formulated-Offline Key Transport Mechanism. The vehicle communicates with each station in its journey from the source to the destination. When it communicates with a station,

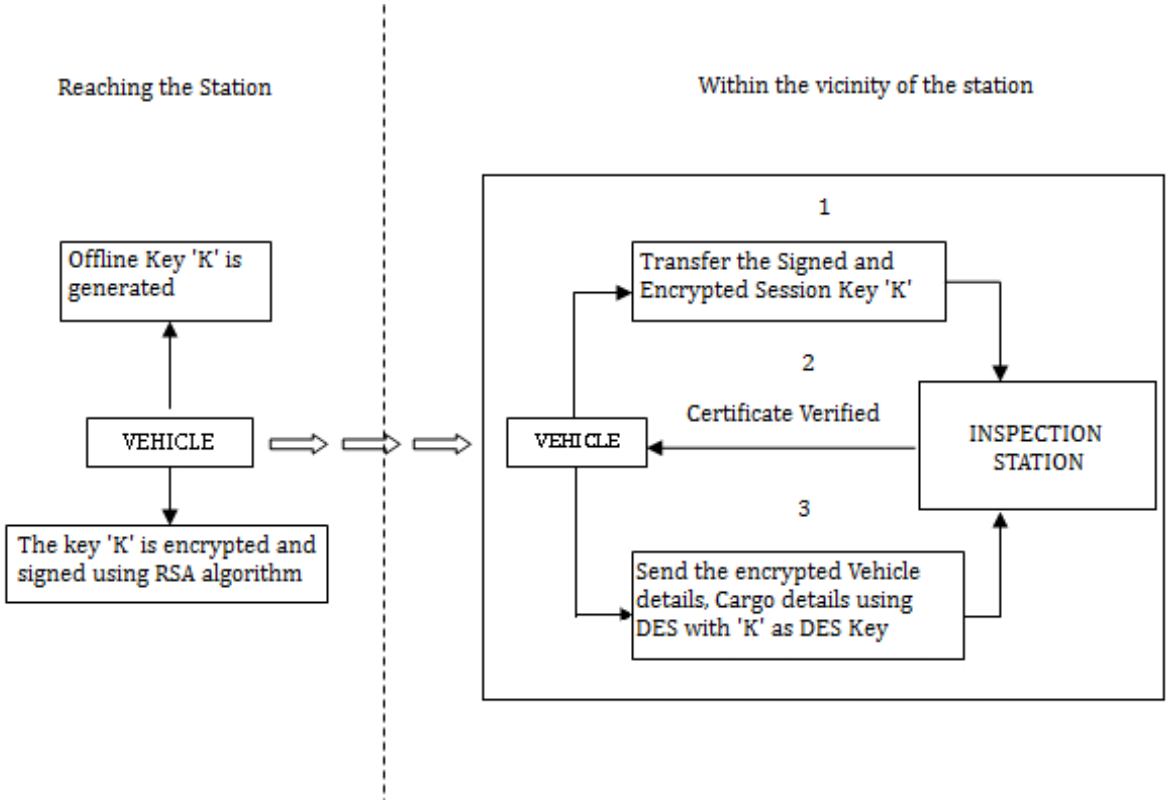


Figure 3-2: Key Transport Mechanism

it has to verify the certificate, encrypt the details and transmit them. The station on the other side has to do the same but it decrypts the details. The vehicle uses the travel time between any two stations efficiently to perform all the computations necessary to communicate with the next station.

### 3.4.3 Offline Key Transport Mechanism

In this method, the vehicle performs the operations in two modes: offline and online. Here offline refers to the stage where the vehicle is not in the vicinity of the inspection station or time before it reaches the inspection station and online refers to the time where the vehicle is communicating with the station. The vehicle uses a method of offline-key generation and an online-key transportation to send the key to the station. Figure 3-2 explains in brief the process of Offline Key Transport Mechanism.

In Offline-Key Generation, the vehicle completes the generation of the session key and generates a certificate by signing a message (containing the session key and the vehicle ID) using the vehicle's public and private keys. If $K$ (8 Bytes) is the session key, then the message $V$ (128 Bytes) is framed as, $V = \{64$ bytes of zeros $||$ 4 repetitions of $VehicleID$ $||$ 4 repetitions of $K\}$. The message framing is further explained in the next chapter. When the vehicle approaches the inspection station, after authenticating the station, the vehicle sends the session key and its ID through Online-Key Transport Mechanism.

In Online-Key Transportation the station verifies the vehicle and the vehicle signs the offline generated certificate (containing the session key) using the station's public key and sends this signed certificate to the station. This certificate is transmitted to the station and upon verification, the station verifies the data sent from the vehicle and takes the decision. As generation of this certificate takes less time when compared to the offline certificate and it also requires station's verification, the vehicle makes this online.

In Session Key Recovery the station recovers the session key and decrypts the messages sent by the vehicle by decrypting the signed certificate using the station's public key. The station extracts the session key from the message $V$. This innovative way of using generating the key offline and performing the computations before approaching the station, reduces the load on the vehicle as well as saves time for the inspection station.

Along with the key generation constraints, one more optimization is used to reduce the computational time on the vehicle side: using the *Chinese Remainder Theorem[5].* The Chinese Remainder Theorem(CRT) as discussed in chapter 2, is one of the techniques used to speed up the RSA calculations by using the divide-and-conquer rule[4]. So to reduce the time of calculation on the vehicle side, the CRT technique is employed. As the stations operate on high end systems and do not have a time constraint for the calculations, the station side systems don't use the CRT technique for the cryptographic algorithms. By using CRT along with the RSA algorithm, only the time taken for the calculations is reduced, without disturbing the efficiency of the original algorithm. And also this technique changes the way the mathematical operations are executed but not the algorithm itself. Before calculating the computational time in both the cases and discussing the challenges faced, various messages involved in this communication system are discussed. The length of the messages and the discovery of an

innovative way to utilize the resources and reduce the computational overhead is also explained in the next section.

### 3.4.4 Message Formats

The entities in the Trusted Truck(R) system operate on data of different lengths starting from the digital certificate to the vehicle related messages. The vehicle's details like the driver ID, vehicle ID, expiration date, cargo details, sensor readings and other vehicle related information are 8 bytes in length. Likewise the station's ID, expiration date is again 8 bytes in length with their most significant bit equal to zero (MSB = 0). The length of the public and private keys generated for the vehicle, station and the CA are,

- Public Key for vehicle/station – 128 Bytes
- Private Key for vehicle/station – 128 Bytes
- Public and private key for the CA – 144 Bytes

The certificate length for any participant except the CA is defined to be the expiration date (8 Bytes, MSB = 0), ID (8 Bytes, MSB = 0), public key (128 Bytes) and the CA signature of the preceding three values. Thus, the certificate generated by the CA for the vehicle/station is 288 Bytes in length. The session key generated for each and every transaction is also 8 bytes with the most significant bit equal to zero(MSB = 0). The messages sent from the station like granting permission for the vehicle, authorizing the vehicle to carry a specific cargo, destination information, etc. are also 8 Bytes in length. Except for the certificates and keys used in the cryptographic algorithms, all the messages exchanged between the vehicle and the stations are 8 Bytes in length.

# Chapter 4

# Mathematical Analysis

In this chapter, we explore the mathematical explanation for the communication system discussed in the previous chapter. As the entire protocol use cryptographic algorithms for the data security, we see that the necessary computations or the operations on the vehicle and the station would be no more than modular exponentiations, multiplications and divisions. It involves modular arithmetic operating on data less than or equal to 144Bytes. Here we discuss how the computational load is balanced between the vehicle and the station which is the key factor for this thesis. Various validations made at different levels on the data and on the intermediate results are also discussed in this section. The major operations in this design are signature generation, signature verification, offline-key transportation and encryption and decryption. Other operations include extracting the details from the messages, exchanging messages, starting the sessions, closing the sessions and logging the transaction. This section explains the mathematical operations involved in each step along with the computational load on each participant (vehicle/station/CA). It also discusses an intelligent way of using the modular arithmetic properties in the cryptographic algorithms to balance the load between the participants so as to make the protocol more time-efficient, resource-efficient and also cost-effective.

The parameters used by the cryptographic algorithms are the public key, private key and the message. Let $n_s$ and $d_s$ be the public and private key respectively for the station. Likewise $n_t$ and $d_t$ represent the public and private key respectively for the vehicle. Here the subscript 's' indicates station and 't' represents the vehicle. Similarly $n_{CA}$ and $d_{CA}$ represent the public and private key respectively for the Certificate Authority (CA).

## 4.1 Signature Generation

For digitally signing the message, the RSA algorithm is used. Both the vehicle as well as the station use RSA for mutual authentication. Let $M$ be any message in the communication link and $S$ be any signed message. The contents of the message can be a single value or it can be a concatenation of various values. The framing of the message also plays a key role in digital signatures. All the keys except for the session key, are generated on the CA system. To generate $n_s$, $n_t$, $d_s$, $d_t$, $n_{CA}$ and $d_{CA}$ the standard RSA algorithm is used as mentioned in chapter 2. Recollecting the RSA algorithm, we know that the signed message $S$ for a given message $M$ is,

$$S = M^d \bmod n \tag{4.1}$$

where $d$ is the private key and $n$ is the public key.

To generate these keys, large prime numbers $p$ and $q$ are generated as mentioned in chapter 2. As we discussed earlier, the vehicle and station use 128 Byte keys whereas the CA uses 144 Byte keys. So the prime numbers are each 64Byte for the vehicle (and station) and 72 Byte for the CA. The procedure to generate the keys is kept the same except for their lengths. In view of the computational loads on the entities, minor changes have been made while generating the keys for the vehicle and station. As the CA works as a separate entity without any restrictions on the computational load, no major changes have been made on the algorithms used on the CA side. The need for this change is discussed in detail in the next sections. The large prime numbers $p$ and $q$ are randomly generated such that $(p-1) \bmod 3 \neq 0$ and $(q-1) \bmod 3 \neq 0$. The primality of the numbers is tested using the Fermat Primality Test[2]. The $p_s$ and $q_s$ for the station are generated in a way that they are always greater that the values generated for the vehicle, that is $p_t$ and $q_t$. To ensure that this relational inequality is maintained, the second most significant bit of $p_t$ and $q_t$ is *always* made 0(zero) and that of $p_s$ and $q_s$ is always made 1 (one). So the prime numbers for the vehicle take the form $\{10p_2p_3p_4.....p_{len-2}1\}$ and the prime numbers for the station take the form $\{11p_2p_3p_4.....p_{len-2}1\}$. As the numbers are prime in nature, the LSB (Least Significant Bit) and MSB (Most Significant Bit) will always be 1 (one). Using the RSA algorithm[6], the public keys are calculated as,$n_s = p_s * q_s; n_t = p_t * q_t; n_{CA} = p_{CA} * q_{CA}$. And we also have $n_t < n_s$. The private key is selected such that $3d \bmod[(p-1)(q-1)] = 1$[6]. This

is reduced as,

$$d = \frac{2 * (p-1) * (q-1) + 1}{3} \tag{4.2}$$

From observation, the MSB of $d$ is always equal to 1 and for $n$ both the MSB and LSB are equal to 1. From a mathematical perspective, the signature generation equation represents a modular exponentiation equation, where a value $M$ is exponentiated to $d$ first and is then divided by $n$ to get the remainder. An algorithm similar to the simple repeated square-and-multiply algorithm[6] is used to compute the modular exponentiation. Equation (4.1) is computed as,

---

**Algorithm 4.1: Modular Exponentiation** (Using repeated square-and-multiply algorithm)
INPUT : $M$, $d\{d_0 d_1 .....d_{len-2} d_{len-1}\}$, $n$, $len$(size of n or d in $bits$)
OUTPUT : $S = M^d \bmod n$
1. Begin
2. Choose $S \longleftarrow 1$
3. **FOR** $i$ from 0 to $len - 1$ do the following
   3.1 $S \longleftarrow S^2 \bmod n$
   3.2 **IF** $d_i = 1$ ($d_i$ is the $i^{th}$ coefficient of the binary representation of $d$)
   **THEN** $S \longleftarrow S * M \bmod n$
4. **END FOR**
4. Return $S$
5. End

---

In this application, the certificate authority (CA) generates the digital certificate for the vehicle as well as the station using CA's set of keys, that is $(n_{CA}, d_{CA})$. The message $M$ is constructed from the participant's details. Let $M_t$ be the message for the vehicle and $M_s$ be the message for the station, the messages are constructed as,

$$M_t = \{ExpDt||VehicleID||n_t\} \tag{4.3}$$

$$M_s = \{ExpDt||StationID||n_s\} \tag{4.4}$$

where $ExpDt$ is the expiration date for the participant's ID

$VehicleID$ is the 8 Byte vehicle's identification number

$StationID$ is the 8 Byte station's identification number

$n_s$ is the 128 Byte station's public key

$n_t$ is the 128 Byte vehicle's public key

Digital certificate is generated using the equation (4.1) as,

$$S_t = M_t^{d_{CA}} \bmod n_{CA} \tag{4.5}$$

$$S_s = M_s^{d_{CA}} \bmod n_{CA} \tag{4.6}$$

When the vehicle receives the station's certificate, it extracts the station's ID and checks the expiration date. If it is valid, it proceeds to the further step in the communication. Similar extraction and verification happens on the station's side.

## 4.2 Signature Verification

Digital signature verification process consists of a verification algorithm followed by recovering the original message. When the vehicle and station mutually authenticate each other, they exchange their digital certificates and once the original message or data is verified, further communication is carried on. For every signature generation algorithm, there is a corresponding signature verification algorithm. RSA provides verification along with digitally signing a message. So the same RSA verification algorithm[6] is used to verify the signature generated without message recovery. Here the extracted message is compared to the original message to verify the certificate. Mathematically, verification is similar to generation in a sense that verification also uses modular exponentiation. Let $R$ be the message obtained after extraction from the certificate and let the encryption constant as discussed in chapter 2 be equal to 3, that is $e = 3$. To improve the efficiency of the decryption, generally a smaller exponent value is used, here $e = 3$. As the message is not the same for any two stations, smaller exponent will not cause any kind of intruder to recover the original message. Also the message is almost of the same length as $n$ but less than $n$, so the attacks which target the small messages are also not possible. The other advantage of having a smaller exponent is that it accelerates the encryption process. If $e = 3$, the encryption process will consist of only 1 modular multiplication and 1 modular squaring. Thus $R$ can be obtained as,

$$R = S^3 \bmod n \tag{4.7}$$

where $S$ is the digital signature for the message $M$, $n$ is the public key of the entity which generated the certificate. Algorithm 4.2 explains the calculation of $M$.

Once we extract $R$, it is compared with $M$ to verify the certificate. Equation (4.7) is again a modular exponentiation similar to equation (4.1) with a smaller exponent. The same repeated square-and-multiply algorithm is used to compute equation (4.7) but with a single iteration as,

**Algorithm 4.2: Signature Verification**
INPUT: $M$, $S$, $n$
OUTPUT: $R = S^3 \bmod n$
1. Begin
2. Choose $B$
3. Compute $B \longleftarrow S * S \bmod n$
4. Compute $R \longleftarrow B * S \bmod n$
5. **IF** $R = M$
   **THEN** return '*verified*'
   **ELSE** return '*failed to verify*'
6.End

## 4.3 Key Transport Mechanism

The vehicle uses preliminary offline-key transportation mechanism to generate the session key while approaching the station. As the session key is used for the encryption and decryption of the critical messages, care has to be taken while transporting the session key to the station. Instead of sending the message directly, a message $V$ is framed with the session key in it such that only that specific station is able to recover $V$ and at the same time the station must be assured that the specific vehicle sent that message $V$. As discussed earlier the 128 Byte message $V$ is framed as

$$\begin{aligned} V \;=\; & \{64\ bytes\ of\ zeros \,||\, VehicleID \,||\, VehicleID \,||\, VehicleID \,||\, VehicleID \,|| \\ & K \,||\, K \,||\, K \,||\, K\} \end{aligned} \tag{4.8}$$

where $K$ is the session key,

$VehicleID$ is the 8 Byte ID used to identify the vehicle.

This message is digitally signed using the $(n_t, d_t)$

$$L = V^{d_t} \bmod n_t \tag{4.9}$$

where $L$ is the digital signature of the message $V$

$n_t, d_t$ are the public and private keys of the vehicle respectively

Unlike the previous signature generation, where we generate the digital signature and transmit to the other party, here we do not transmit the signature immediately after signing it for the following reason. The session key should be kept secret and only the specific station should be able to recover and use it to decrypt the vehicle details. So in order to maintain the confidentiality, both the parties agree upon the message format and the repetitions of the ID and the session key in the original message. This prevents from any intruder from modifying the message. The number of repetitions of the ID signifies the known message which is embedded in the original message. Unlike the standard RSA[6], here we cannot transmit the original message for signature verification as we did in Algorithm 4.2 as it consists the secret key (session key). So even the signature is encrypted and later on the key is recovered from the extracted message. Let $C$ be the encrypted version of the signed message, then $C$ is given by,

$$C = L^3 \bmod n_s \tag{4.10}$$

where $C$ is the encrypted message,

$n_s$ is the station's public key

that is,

$$C = \{V^{d_t} \bmod n_t\}^3 \bmod n_s \tag{4.11}$$

It is important to mention that equation (4.10) is not be confused with the equation (4.7) as the latter refers to the signature verification with the public key $e = 3$ and the former refers to the encryption using RSA. Equation (4.10) can be stated otherwise as a message $(L)$ obtained after signing the original message with the $(n_t, d_t)$ first and later on signing with the station's public key with the encryption exponent as 3. The station can recover the message from the encrypted message by using the combination of previously discussed signature generation and

signature verification algorithms as,

$$V = \{C^{d_s} \bmod n_s\}^3 \bmod n_t \qquad n_t < n_s \tag{4.12}$$

Equation (4.12) can be explained as decrypting or extracting the message first using $(n_s, d_s)$ and later on extracting the original message $V$ using $(n_t, 3)$ using algorithm 4.1. Here it is worth mentioning the inequality, $n_t < n_s$ again as extracting the original message using equation (4.12) is infeasible if $n_t \not< n_s$. Now, as both the parties agree upon the format of the original message, the station can easily recover the key by checking the last 64 bytes of the message (here $V$). By verifying the $VehicleID$ in the message, the station can assure that the message is from the required vehicle and not any intruder. Once the station recovers the session key, it can decrypt the vehicle's details and exchange the messages with the vehicle.

As discussed in section 3.3, Data Encryption Standard (DES) is used to encrypt the data exchanged between the vehicle and the station. DES as such does not involve highly complicated computations when compared to RSA and other digital signature algorithms. The computations involved in a DES algorithm are logical AND, logical XOR and logical OR which take time that is insignificant when compared to modular exponentiations. The DES algorithm takes a 8 Byte value and generates a cipher text of 8 Byte using a 64 bit key. In this application, the 64bit key is generated from the session key which is created for each and every transaction with the station. All the message exchanges except for the messages used for authentication are 8 Bytes in length.

## 4.4 Optimization using Chinese Remainder Theorem

The modular exponentiation has a higher order of complexity and also consumes more time compared to other calculations. The Chinese Remainder Theorem optimizes the time taken for the modular exponentiation, thus speeding up the RSA process. When data of large size if used in the exponentiation, the CRT techniques breaks down the exponentiation into parts and then combines them. Even though this looks complex, the final result is obtained is lesser time. By using the CRT, it reduces the time taken for the modular exponentiation by 4 times, that is,

*time taken using CRT technique* $= \frac{1}{4} *$ *time taken without using the CRT technique.*

Consider equation (4.1), where $n = p * q$, $d = \frac{2*(p-1)*(q-1)+1}{3}$ and $n$ and $d$ are both 128 Byte numbers. Among the prime numbers $p$ and $q$, usually $p$ is the larger prime number and the message $M$ is always less than $n$. The equation can be written as,

$$S = M^{(\frac{2(p-1)(q-1)+1}{3})} \bmod (p * q) \tag{4.13}$$

Since both $p$ and $q$ are 64 byte wide here, the exponent ($d$) as well as the divisor($n$) will be 128byte. This exponentiation can be broken down into two parts and then later on can be combined to get the result $S$. All the certificates generated on the vehicle uses CRT acceleration along with the other crypto routines. Here, the value of $S$ is calculated without actually calculating $n$. Algorithm 4.3 is based on to the traditional Chinese Remainder Theorem[6].

**Algorithm 4.3: Signature Generation using Chinese Remainder Theorem**
INPUT: $M, p, q, d$
OUTPUT: $S = M^d \bmod (p * q)$
1. Begin
2. Compute $dp \longleftarrow d \bmod (p-1)$
3. Compute $dq \longleftarrow d \bmod (q-1)$
4. Compute $Mp \longleftarrow M \bmod p$; $Mq \longleftarrow M \bmod q$
5. Consider $a$ and compute $a \longleftarrow (M \bmod p)^{dp} \bmod p$
6. Consider $b$ and compute $b \longleftarrow (M \bmod q)^{dq} \bmod q$
7. Consider $U$ and compute $U \longleftarrow q^{p-2} \bmod p$
   And also verify if $(U * q) \bmod p = 1$
8. **IF** $a < b$ **THEN** $a \longleftarrow a + p$
9. Compute $X \longleftarrow [\{(a-b) * U\} \bmod p] * q + b$
10. Return $X$
11. End

After calculating $X$ using algorithm 4.3, we observe that both $X$ and $S$ are the same which shows that using CRT in the computation of modular exponentiation, just reduces the number of iterations but does not disturb the result. Comparing algorithm 4.3 with algorithm 4.1, one can notice that the number iterations are greatly reduced, thus reducing the complexity of the operation. And also, the number of multiplications and divisions computed in algorithm 4.1 are far more compared to that of algorithm 4.3 where we used the Chinese Remainder Theorem. The values $a, b$ and $U$ can be computed using the repeated square-and-multiply algorithm. As the CRT operates on data which is exactly half of the length of the date which is used in

the original modular exponentiation algorithm, it reduces the computation time by four times. Thus, all the signature generation operations executed on the vehicle use signature generation with CRT while all the operations executed at the station follow algorithm 4.1.

## 4.5 Computational Challenges - Solution

In this application, all the operations on the vehicle use a 16 bit processor while the stations use a 32 bit processor. This creates the imbalance in the computational time for the vehicle and the station. The following show the time taken for the similar operations on both 32 bit as well as 16 bit without using Chinese Remainder Theorem,

*time taken for modular exponentiation on a 32 bit processor without CRT is* $0.2-0.3\,\mathrm{sec}$ .(*approx.*)

*time taken for modular exponentiation on a 16 bit processor without CRT is* $0.4-0.5\,\mathrm{sec}$ .(*approx.*)

This time difference causes a great asymmetry in the communication process where the station should be validating a number of vehicles in a given time. So in order to balance this inequality, optimizations have been made in many ways on the vehicle side. The attempts to reduce the computational load on the truck starts at the CA level by incorporating few constraints on the keys generated for the truck. When the CA generates the public and private keys for the truck, it ensures that those keys are *always* less than the keys generated for the station so that the values calculated on the truck are always less than that at station and always take less time. All the trucks perform the session key generation required for the authentication offline (16 bit processing). This offline-key generation is done for every communication session before approaching a station, thus saving the time. On the other hand, the stations perform all the operations online (32 bit processing). As we observe the only operation performed by the truck 'online' is the modular exponentiation with the exponent equal to 3 i.e., $A = B^3 \bmod N$ where A,B and N are 128 byte numbers. Mathematically as the operation $A = B^3 \bmod N$ just involves one modulo calculation, it takes very less time when compared to the operation $A = B^E \bmod N$. Thus when the truck starts communicating with the station, it does not have to spend much time on calculation. Thus all these methods combine to reduce the computational load on the vehicle and help the vehicle in finishing the computations in a less and reasonable time. After implementing the CRT on the vehicle, the time taken for the modular exponentiations is greatly

reduced,

*time taken for the modular exponentiation on a 16 bit processor using CRT is* $0.12 - 0.14\,\text{sec}$ (*approx.*)

It is noticed that this time, 0.12 seconds is nearly equal to $\frac{1}{4}^{th}$ times the time taken without using CRT, that is 0.4 seconds. which clearly proves that including Chinese Remainder Theorem reduces the computation time by 4 times. And also the time taken for the offline-key generation which is here the time taken for modular exponentiation in 16 bit processor using CRT ($0.12\,\text{sec}.$) is very less than the time taken for the vehicle to travel from one station to the other. So the vehicle can easily generate the offline key and transfer the key online while approaching the station.

## 4.6 Results and Discussion

### 4.6.1 Implementation

The security protocol is developed in basic ANSI C which is inexpensive and does not need any additional software for compiling. The software developed is fully compatible with a Unix platform. The OBUs and the RBUs are equipped with systems working on Unix platform. Three different packages are developed individually for the Certificate Authority (CA), Vehicle (truck) and the Station. All packages have their own code running independently on the respective system. The CA package is responsible for the generation of keys, digital signature for the vehicle and the station. The vehicle package (truck) is responsible for three major operations - Offline Key generation, Signature Verify and Encryption. The station package is responsible for the following operations-Signature Verification and Decryption. In all the communication sessions, the vehicle is assumed to initiate the communication while it is entering the vicinity of the station. Along with message transfer, each and every communication session is logged which can be used for further reference. Any link failure or invalid credentials are all logged in the log file.

### 4.6.2 Time Estimate

The computational time is noted at each and every stage in the mathematical operations and proper care is taken while selecting the appropriate cryptographic algorithms which will fit the available environment. The time taken for the modular exponentiations in both 32 bit and 16 bit is observed and noted. Table 1 shows the time taken by a 32 bit processor for the modular exponentiation operating both without using CRT and using CRT. The average time taken by a 32 bit processor to compute the equation (4.1) that is $S = M^d \bmod n$ is 0.218 seconds (*approx.*). The same 32 bit processor takes 0.05715 sec (*approx.*) to compute $X = [\{(a-b)*U\} \bmod p]*q+b$ in Algorithm 4.3. This supports the proof of Chinese Remainder Theorem reducing the computational time by nearly 4 times.

| Trial No | Time taken without CRT (secs) | Time taken using CRT (secs) (includes the time taken for a and b) |
|---|---|---|
| 1 | 0.208438 | 0.057042 |
| 2 | 0.223449 | 0.057879 |
| 3 | 0.206945 | 0.057248 |
| 4 | 0.215324 | 0.056165 |
| 5 | 0.229685 | 0.056701 |
| 6 | 0.226190 | 0.056904 |
| 7 | 0.233663 | 0.058223 |
| 8 | 0.224079 | 0.05743 |
| 9 | 0.212913 | 0.056463 |
| 10 | 0.205311 | 0.057443 |
| **Average =** | **0.2186** | **0.05715** |

But, in this application the participant which requires a speed up process is the vehicle which uses a 16 bit processor. Thus time estimates are noted for a 16 bit processor also. It is found that the average time taken by a 16 bit processor to compute the equation $S = M^d \bmod n$ is in the range of $0.4 - 0.5\,\text{sec}(approx.)$, that is without incorporating CRT. When CRT acceleration is applied the same 16 bit processor, the time taken is observed to be in the range of $0.12 - 0.13\,\text{sec}(approx.)$. This once again proved the efficiency of Chinese Remainder

Theorem in reducing the computational time. By observing these values, it can be concluded that computational imbalance has been totally balanced and the vehicle is left with ample time for any other computational duties other than message exchange. Looking at the station side, which takes 0.4 sec to compute the crypto routines, it would be easy for any station to complete the verification process within a very short time that too while the vehicle is approaching and this helps the station concentrate on any other trivial details other than validating the vehicles. This greatly improves the count of vehicles being checked by a station in a given time period.

# Chapter 5

# Conclusions

## 5.1 Thesis Summary

This thesis explains how a secured data protocol for the Trusted Truck(R) system is developed successfully. It explains the necessity of a secure data infrastructure in an efficient wireless communication system like Trusted Truck (R). By using RSA, a public key cryptography algorithm and the Data Encryption Standard (DES) for encryption and decryption, a secured communication mechanism has been designed. Speed up algorithms are used along with RSA to improve the computation time on the vehicle which balances the computational effort on both the station and the vehicle. A secured key transport mechanism has been introduced along with the enhancements in the computations. The time taken for the algorithms with and without using the speed up algorithm is shown. The protocol is designed in a way that it embeds itself in the existing architecture of the Trusted Truck(R) system without any additional hardware or software. The protocol is developed in standard ANSI C and using the TCP / IP protocol stack and is platform independent.

## 5.2 Future Work

Further advancements to the protocol can be made by optimizing the algorithms used in the design. Optimizing the algorithms further reduces the time taken by the vehicle or the station to compute the cryptography operations necessary for the message transfer.

# Bibliography

# Bibliography


[1] Bruce Schneier "*Applied Cryptography: Protocols, Algorithms, and Source Code in C*", John Wiley & Sons Publications, Second Edition 1996.

[2] "*Introduction to cryptography with coding theory"*, second edition Wade Trappe, Lawrence C. Pearson Education Publications, Washington 2006.

[3] Data Encryption Standard, Federal Information Processing- T. Takagi, "*Fast RSA-Type Cryptosystem Modulo pkq*", CryptoStandards Publication (FIPS PUB) 46, National Bureau of Standards, 1998, 1462 of LNCS. 1998, pp. 318-326.Washington, DC (1977).

[4] Chung-Hsien Wu; Jin-Hua Hong; Cheng-Wen Wu,"*RSA cryptosystem design based on the Chinese Remainder Theorem,*" *Design Automation Conference*, 2001. Proceedings of the ASP-DAC 2001. Asia and South Pacific 30 Jan.-2 Feb. 2001 Page(s): 391 - 395.

[5] Sining Liu; King, B.; Wei Wang, "*A CRT-RSA Algorithm Secure against Hardware Fault Attacks,*" *Dependable, Autonomic and Secure Computing, 2nd IEEE International Symposium* on Sept. 29 2006-Oct.1 2006 Page(s): 51 - 60.

[6] "*Handbook of Applied Cryptography*", Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone CRC Press; 1 edition 1996.

[7] "*Data Encryption Standard (DES)*", Federal Information Processings Standard Publications, 46-3, October 25, 1999.

[8] W. DIFFIE AND M.E. HELLMAN, "*Multiuser cryptographic techniques*", Proceedings of AFIPS National Computer Conference,109–112, 1976.

[9] "*New directions in cryptography*", IEEE Transactions on Information Theory, 22 (1976), 644–654.

[10] M.O. Rabin "*Digital Signatures,*" Foundations of Secure Communication, New York: Academic Press, 1978, pp.155-168.

[11] R.L. Rivest, A.Shamir, and L.M. Adleman, *"A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,*" Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126.

[12] R.L. Rivest, A.Shamir, and L.M. Adleman, "*On Digital Signatures and Public Key Cryptosystems,*" MIT Laboratory for Computer Science, Technical Report, MIT/LCS/TR-212, Jan 1979.

[13] D. Balenson, "*Private Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*" RFC 1423, Feb 1993.

[14] RSA Laboratories, "*PKCS #1: RSA Encryption Standard,*" version 1.5, Nov 1993.

[15] M.J. Wiener, "*Cryptanalysis of Short RSA Secret Exponents,*" IEEE Transactions on Infornation Theory, v. 36, n. 3, May 1990, pp. 553-558.

[16] J,-J, Quisquater and C. Couvreur, "*Fast Decipherment Algorithm for RSA Public-Key Cryptosystem,*" Electronic Letters, v.18, 1982, pp. 155-168.

[17] D. Coppersmith, "*The Data Encryption Standard (DES) and its strength against attacks,*" IBM Journal of Research and Development, vol. 38, no. 3, May 1994, pp. 243-250.

[18] MITCHELL C.J, "*Another improvement to square-and-multiply exponentiation,*" Technical Report CSD-TR-93-cc, Royal Holloway, University of London, Computer Science Department, Egham, Surrey, August 1993.

[19] Donald E. Knuth, "*The art of computer programming*", volume 2: Seminumerical Algorithms, 3rd edition, Addison-Wesley Publications, Reading, 1997. ISBN 0-201-89684-2.

[20] Gerrit Bleumer, "*Advances in Information Security, Electronic Postage Systems Technology, Security, Economics,*" Volume 26, 2007 pp 91-118.

# Vita

Srinivasa Anuradha Bulusu was born in India on July 27, 1985. She received her Bachelor of Science degree in Electronics and Communication Engineering from Jawaharlal Nehru Technological University, India in May of 2006 and received her Master of Science degree in Electrical Engineering from University of Tennessee, Knoxville in December of 2010.