Volume 6 ▪ Issue 3 ▪ June 2012

Editor-in-Chief
Dr Wei Wang

INTERNATIONAL JOURNAL OF SECURITY (IJS)

ISSN : 1985-2320

Publication Frequency: 6 Issues / Year

CSC PUBLISHERS
http://www.cscjournals.org

INTERNATIONAL JOURNAL OF SECURITY (IJS)

VOLUME 6, ISSUE 3, 2012

EDITED BY
DR. NABEEL TAHIR

ISSN (Online): 1985-2320

International Journal of Security (IJS) is published both in traditional paper form and in Internet. This journal is published at the website http://www.cscjournals.org, maintained by Computer Science Journals (CSC Journals), Malaysia.

IJS Journal is a part of CSC Publishers

Computer Science Journals

http://www.cscjournals.org

INTERNATIONAL JOURNAL OF SECURITY (IJS)

Book: Volume 6, Issue 3, June 2012
Publishing Date: June 2012
ISSN (Online): 1985-2320

IJS Journal is a part of CSC Publishers
http://www.cscjournals.org

Published in Malaysia

Typesetting: Camera-ready by author, data conversation by CSC Publishing Services – CSC Journals, Malaysia

CSC Publishers, 2012

EDITORIAL PREFACE

This is the third issue of volume six of The International Journal of Security (IJS). The Journal is published bi-monthly, with papers being peer reviewed to high international standards. The International Journal of Security is not limited to a specific aspect of Security Science but it is devoted to the publication of high quality papers on all division of computer security in general. IJS intends to disseminate knowledge in the various disciplines of the computer security field from theoretical, practical and analytical research to physical implications and theoretical or quantitative discussion intended for academic and industrial progress. In order to position IJS as one of the good journal on Security Science, a group of highly valuable scholars are serving on the editorial board. The International Editorial Board ensures that significant developments in computer security from around the world are reflected in the Journal. Some important topics covers by journal are Access control and audit, Anonymity and pseudonym, Computer forensics, Denial of service, Network forensics etc.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 6, 2012, IJS appears in more focused issues. Besides normal publications, IJS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

The coverage of the journal includes all new theoretical and experimental findings in the fields of computer security which enhance the knowledge of scientist, industrials, researchers and all those persons who are coupled with computer security field. IJS objective is to publish articles that are not only technically proficient but also contains information and ideas of fresh interest for International readership. IJS aims to handle submissions courteously and promptly. IJS objectives are to promote and extend the use of all methods in the principal disciplines of computer security.

IJS editors understand that how much it is important for authors and researchers to have their work published with a minimum delay after submission of their papers. They also strongly believe that the direct communication between the editors and authors are important for the welfare, quality and wellbeing of the Journal and its readers. Therefore, all activities from paper submission to paper publication are controlled through electronic systems that include electronic submission, editorial panel and review system that ensures rapid decision with least delays in the publication processes.

To build its international reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJS. We would like to remind you that the success of our journal depends directly on the number of quality articles submitted for review. Accordingly, we would like to request your participation by submitting quality manuscripts for review and encouraging your colleagues to submit quality manuscripts for review. One of the great benefits we can provide to our prospective authors is the mentoring nature of our review process. IJS provides authors with high quality, helpful reviews that are shaped to assist authors in improving their manuscripts.

Editorial Board Members
International Journal of Security (IJS)

EDITORIAL BOARD

TABLE OF CONTENTS

Volume 6, Issue 3, June 2012

Pages

Blinded Montgomery Powering Ladder Protected Against the Jacobi Symbol Attack

David Tinoco Varela *dativa19@comunidad.unam.mx*
Computational Science Graduate Program
Facultad de Estudios Superiores Cuautitlán
Universidad Nacional Autónoma de México
Edo. Mex. 54740, México

Abstract

Recently, many physical attack types (e.g., timing attacks, power consumption attacks, fault attacks) have been developed against cryptosystems, specifically against the modular exponentiation that is the core operation of many security systems. Indeed, there is a real need to eliminate the vulnerabilities of cryptosystems, such as RSA or the Elliptic Curve Cryptosystem, that make them susceptible to such attacks. In 2006, Boreale described a new type of physical attack based in the Jacobi symbol concept, and later, Schmidt used the same concept as Boreale to break the security of the blinded Montgomery powering ladder. In this paper, a countermeasure against Schmidt's attack is presented to make the blinded Montgomery powering ladder resistant to the Jacobi symbol attack.

Keywords: Modular Exponentiation, Cryptography, Jacobi Symbol, Montgomery Ladder, Fault Attacks.

1. INTRODUCTION

Kocher [1] was the first to point out the existence of physical attacks called *Side Channel Attacks (SCA)*. He observed that when a cryptographic algorithm is implemented in an embedded device, an attacker can obtain the binary string of the secret key by simply observing the power traces or the timing consumption of the device in an electronic test instrument, such as an oscilloscope. SCAs are, first of all, used to attack modular exponentiation (*Add and double* is the analogous function in the Elliptic Curve Cryptosystem, ECC), which is the core operation in cryptosystems such as RSA.

SCAs opened the door to a new type of physical attacks, one of which was the *Fault Attack* (FA) proposed by Bonhe, DeMillo and Lipton [2]. FAs are more aggressive than SCAs because FAs physically disturb the execution of the device that is running the cryptographic algorithm.

To prevent SCAs and FAs, many modular exponentiation algorithms have been created, but Coron [3] provided the first algorithm specifically designed to defeat SCAs when he proposed the s*quare-and-multiply always algorithm*. However, this algorithm was attacked by the denominated *Safe Error Attack* (SEA) [4].

The *Montgomery powering ladder* [5] was a new idea proposed by Joye and Yen to protect cryptosystems against SCAs and FAs. This algorithm works in a regular form: that is, regardless of the value of the bit being processed (0 or 1), the algorithm will always calculate a multiplication followed by a squaring. The Montgomery ladder was widely accepted and attracted the attention of many researchers. Giraud [6] modified the Montgomery ladder to protect it against FAs; he proposed a *Coherence Test* based on a characteristic of the algorithm: the registers in all the iterations have the form $R[0] = m^x$, $R[1] = m^{x+1}$. As a result, if the coherence test $R[0] \cdot m = R[1]$ is true, then return $R[0]$; if not, return "error".

The Montgomery ladder was attacked by the *Relative Doubling Attack* (RDA) [7], a modification of the *Doubling Attack* (DA) [8], but Fumaroli and Vigilant [9] added a random value to the Montgomery ladder to blind the modular exponentiation. The algorithm proposed by Fumaroli and Vigilant was secure against SCAs, DA, RDA, and in a partial form against FA.

A new type of attack was presented by Boreale in 2006 [10]. This attack uses a combination of FA and SCA, and using the Jacobi symbol (JS) it is possible to obtain the binary string of the secret key d. He used his model against the *square-and-multiply right-to-left* algorithm and proved that his attack is effective even in the presence of message blinding. On the other hand, Schmidt and Medwed [11] used the Jacobi symbol concept to create an attack that breaks the security of the Montgomery powering ladder in its blinded form.

There are more modular exponentiation algorithms ([12], [13], [14], [15], [16], [17]) trying to defeat all the physical attacks ([18], [19], [20], [4], [21]) that threaten the security of the cryptosystems, but here, we focus our attention only on the blinded form of the Montgomery ladder algorithm and on the goal of avoiding Jacobi symbol attacks.

2. PRELIMINARIES

2.1 Jacobi Symbol

The first necessary concept is the *quadratic residue*: for a given prime p, a is a quadratic residue if $\gcd(a, p) = 1$ and $a = y^2 \bmod p$ for some y. If $\gcd(a, p) = 1$ but a is not a quadratic residue mod p, a is called a quadratic non-residue mod p.

$\left(\frac{a}{p}\right)$ is called the *Legendre symbol* of $a \bmod p$, and we can see that

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{If } a \text{ is a quadratic residue } \bmod p \\ -1 & \text{If } a \text{ is a quadratic non - residue mod } p \\ 0 & \text{If there is a common factor} \end{cases}$$

Now, we have that $\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right) \cdots \left(\frac{a}{p_k}\right)$ is the Jacobi symbol, where n is odd, $n = p_1 \cdots p_k$, and the p_i are prime factors of n. The Jacobi symbol is a generalization of the Legendre symbol.

2.1 Fault Attacks

Bone, DeMillo and Lipton showed that it is possible to disturb an embedded device while it is executing a cryptographic algorithm [2] and that with the erroneous output value, an attacker can obtain secret information that can break the security of the cryptosystem. A disturbance can be induced, principally, by variation in supply voltage, and it may cause the device to misinterpret data or even skip a complete instruction.

2.2 Montgomery Powering Ladder and its Blinded Form

Many modular exponentiation algorithms have been developed. Joye and Yen proposed a new kind of algorithm to calculate the modular exponentiation, called the *Montgomery powering ladder* [5]. Their model was based on a different idea from those algorithms designed before it. The principal concept was that

$$L_j = \sum_{i=j}^{t-1} d_i 2^{i-j} \quad \text{and} \quad H_j = L_j + 1$$

Some characteristics of the Montgomery ladder introduced in [5] are as follows:

- The algorithm is highly regular; that is, there is always a multiplication followed by a squaring, regardless of the processed bit.
- $R[1]/R[0]=m$ is invariant throughout the execution of the algorithm.
- The two multiplications performed at each iteration share a common operand, for which the *Common-multiplicand multiplication* [22] can be used.
- The two multiplications performed are independent at each iteration, and therefore, they can be calculated in parallel form.

The Montgomery ladder was improved by Fumaroli and Vigilant (FV scheme), who added a random element r to protect the algorithm; they used one more register than the simple Montgomery ladder to save the inverse value of the random element r (Algorithm 1).

Algorithm 1 FV scheme

1: **Input** $m\in G$, $d=(d_{n-1}\dots d_0)_2$
2: **Output** $s=m^d\in G$
3: $R[0]\leftarrow r$; $R[1]\leftarrow m\cdot r$
4: $R[2]=r^{-1}$
5: **for** $n-1$ to 0 **do**
6: $R[\overline{d_i}]\leftarrow R[\overline{d_i}]\cdot R[d_i] \bmod N$
7: $R[d_i]\leftarrow R[d_i]\cdot R[d_i] \bmod N$
8: $R[2]=R[2]\cdot R[2] \bmod N$
9: **end for**
10: **Return** $R[0]\cdot R[2] \bmod N$

2.3 Attacks Based on the Jacobi Symbol

In 2006, Boreale proposed a new kind of attack against the modular exponentiation, implemented over the *binary square-and-multiply right-to-left algorithm* (Algorithm 2) [10]. He put a fault z in $R[1]$ when a squaring is executed in the iteration $i-1$ of the *for* loop. Then, depending on the value of (S/N), where S is the attacked output value, it can be possible to determine the value of the bit d_i. This scheme works by assuming that $(m/N)=1$, where m is the input value, and its behavior is similar to the *Safe error* : if the value of the bit in the i-th iteration is equal to 0, the fault does not affect the calculation of the JS of $(R[0]_i/N)=1$, but if $d_i=1$, z affects the register $R[0]_i$ which can provoke a JS value of $(R[0]_i/N)=-1$,and thus a JS value of $(S/N)=-1$. Here, two options are given: if (S/N) is always equal to 1, then $d_i=0$, but if $(S/N)=-1$, d_i is equal to 1. Thus, an enemy can deduce the secret key of the cryptosystem.

Table 1 shows the behavior of algorithm 2 under the attack described by Boreale. In the example, it was assumed that $(m/N)=1$, $(z/N)=-1$, and $d=25=11001$.

In 2010, Schmidt [11] proposed an attack that consisted of giving a message m with $(m/N)=-1$ to the FV scheme and skipping the operation $R[d_i]=R[d_i]^2$. Then, observing the resulting value could identify the values of d_i and d_{i+1}. If $(S/N)=-1$, then $d_i=d_{i+1}$. The procedure of this attack is shown as algorithm 3.

An example of the attack described in the algorithm 3 against the FV scheme is observed in table 2. In this example, it was supposed that $(m/N)=-1$ and $d=19=10011$.

Algorithm 2 Square-and-multiply right-to-left

1: **Input** $m \in G$, $d = (d_{n-1} \dots d_0)_2$
2: **Output** $s = m^d \in G$
3: $R[0] \leftarrow 1;\ R[1] \leftarrow m$
4: **for** 0 to $n-1$ **do**
5: **if** $d_i = 1$ **then**
6: $R[0] \leftarrow R[0] \cdot R[1] \bmod N$
7: **end if**
8: $R[1] \leftarrow R[1]^2 \bmod N$
9: **end for**
10: **Return** $R[0]$

i	d_i	**Intermediate products**	**Jacobi symbol**
0	1	$R[0] = m$ $R[1] = (m)^2 = m^2$	$(R[0]/N) = (1) = 1$ $(R[1]/N) = (1) = 1$
1	0	$R[0] = m$ $R[1] = (m^2)^2 = m^4$	$(R[0]/N) = (1) = 1$ $(R[1]/N) = (1) = 1$
2	0	$R[0] = m$ $R[1] = (m^4)^2 = m^8 = z$	$(R[0]/N) = (1) = 1$ $(R[1]/N) = (-1) = -1$
3	1	$R[0] = m \cdot z$ $R[1] = (z)^2 = z^2$	$(R[0]/N) = (1)(-1) = -1$ $(R[1]/N) = (1) = 1$
4	1	$R[0] = m \cdot z^3$ $R[1] = (z^2)^2 = z^4$	$(R[0]/N) = (1)(-1) = -1$ $(R[1]/N) = (1) = 1$

TABLE 1: Algorithm 2 performed with a JS attack, FA in $i-1$ and $d_i = 1$.

Algorithm 3 Attack proposed in [11]

1: **Ensure** Exponent $d = (d_{n-1} \dots d_0)_2$ is used by the device.
2: **Set** $d_{n-1} = 1$
6: **for** $n-2$ to 0 **do**
5: Chose $m \in Z_N$ with $\left(\frac{m}{N}\right) = -1$
6: Calculate S with the i-th squaring skipped
6: **if** $\left(\frac{S}{N}\right) = -1$ **then**
7: $d_i = d_{i+1}$
8: **else**
9: $d_i = 1 \oplus d_{i+1}$
10: **end if**
11: **end for**
12: **Return** d

i	d_i	Intermediate products	Jacobi symbol
4	1	$R[0] = m \cdot r^2$ $R[1] = m^2 \cdot r^2$	$(R[0]/N) = (-1)(1) =$ -1 $(R[1]/N) = (1)(1) =$ 1
3	0	$R[0] = (m \cdot r^2)^2 = m^2 \cdot r^4 = FA$ $R[1] = m^3 \cdot r^4$	$(R[0]/N) = (1)(1) =$ 1 $(R[1]/N) = (-1)(1) =$ -1
2	0	$R[0] = (m \cdot r^2)^2 = m^2 \cdot r^4$ $R[1] = m^3 \cdot r^4 \cdot m \cdot r^2 = m^4 \cdot r^6$	$(R[0]/N) = (1)(1) =$ 1 $(R[1]/N) = (1)(1) =$ 1
1	1	$R[0] = m^6 \cdot r^{10}$ $R[1] = (m^4 \cdot r^6)^2 = m^8 \cdot r^{12}$	$(R[0]/N) = (1)(1) =$ 1 $(R[1]/N) = (1)(1) =$ 1
0	1	$R[0] = m^{14} \cdot r^{22}$ $R[1] = (m^8 \cdot r^{12})^2 = m^{16} \cdot r^{24}$	$(R[0]/N) = (1)(1) =$ 1 $(R[1]/N) = (1)(1) =$ 1

TABLE 2: Algorithm 1 executed with FA, where $d_{i+1} \neq d_i$.

In table 2, it can be noted that a modular multiplication in $i-1$ must be performed by two elements with odd exponents to obtain a result with an even exponent and so obtain $(S/N)=1$, which is the key point of the Schmidt's attack. This situation is observed when the modular multiplication $R[1]_{i=2} = R[1]_{i=3} \cdot R[0]_{i=4}$ is calculated after skipping the squaring operation $R[0]_{i=3}$.

The two attacks mentioned above are easy to implement and powerful because they only need to know about the Jacobi symbol in the returned value by the attacked algorithm.

3. PROPOSED ALGORITHM

In this section, a modification of the FV scheme is proposed that is secure against Schmidt's attack, and the behavior of the proposed algorithm is explained.

3.1 Algorithm

In the approach proposed by Schmidt to attack the FV scheme, the idea is not to put a random value z in the execution but to skip a complete squaring operation in the iteration i when the algorithm is being executed. Then, depending on the value of (S/N), it can be determined whether $d_{i+1} = d_i$.

It can be noted that only even intermediate exponents, through an algorithm, can be used to calculate any modular exponentiation. On the basis of this observation, algorithm 4 is proposed. It can be seen that this algorithm begins the register $R[1]$ with an even exponent $R[1] = m \cdot m = m^2$. This even exponent will affect all the calculations through the algorithm, and thus, it will affect the JS of all the intermediate values calculated by algorithm 4. Here, odd values d are considered.

In algorithm 4, it can be seen that the loop is not executed from $n-1$ to 0 but from $n-1$ to 1, because of the behavior of the algorithm; this behavior will be explained in section 3.2. It can be noted that only the value in $R[1]$ was altered, whereas no extra value was placed in $R[0]$.

Algorithm 4 guarantees that when an attacker skips one squaring operation, in any iteration of the loop, he will not be able to obtain any relevant information about the bits of the string of d, because to obtain any information, it is necessary to have in the output value $(S/N)=1$ or $(S/N)=-1$ depending on the value of the attacked bits d_{i+1} and d_i. However, the output value

of algorithm 4 will always be $(S/N)=1$ if $(m/N)=1$ and $(S/N)=-1$ if $(m/N)=-1$, regardless of the values of d_{i+1} and d_i.

Algorithm 4 Modified FV scheme

1: **Input** $m \in G$, $d=(d_{n-1}\dots d_0)_2$
2: **Output** $s=m^d \in G$
3: $R[0] \leftarrow r$
4: $R[1] \leftarrow m^2 \cdot r$
5: $R[2]=r^{-1}$
6: **for** $n-1$ to 1 **do**
7: $\quad R[\overline{d_i}] \leftarrow R[\overline{d_i}] \cdot R[d_i] \bmod N$
8: $\quad R[d_i] \leftarrow R[d_i] \cdot R[d_i] \bmod N$
9: $\quad R[2]=R[2] \cdot R[2] \bmod N$
10: **end for**
11: $R[0]=R[0] \cdot m$
12: **Return** $R[0] \cdot R[2] \bmod N$

All the values obtained in the intermediate steps of algorithm 4 have an even exponent, and obviously, all of them are quadratic residues; therefore, they have a JS equal to 1. Now, in line 11 of algorithm 4, it is possible to see that the register $R[0]$ is altered by the operation $R[0]=R[0]_{i=1} \cdot m$, where $R[0]_{i=1}$ is the resulting value of the iteration $i=1$ of the *for* loop (lines 6 to 10 of algorithm 4). All the values calculated through the *for* loop have a JS equal to 1, and therefore, the JS of $R[0]_{i=1}$ is equal to 1. For that reason, the JS of the returned value depends of the JS of m, disregarding completely the values of d_{i+1} and d_i, because if the JS of m is equal to 1, then $R[0]=R[0]_{i=1} \cdot m=(1)\cdot(1)=1$ (considering only JS values), and if the JS of m is equal to -1, then $R[0]=(1)\cdot(-1)=-1$.

As shown in table 2, elements with even exponents (quadratic residues) and with odd exponents (quadratic non-residues) are needed in the intermediate products to deduce the binary string of d. Thus, the proposed countermeasure is a protection against the Jacobi symbol attack, because the execution of algorithm 4 has only even exponents in the intermediate products. This protection is observed in table 3. In this example, it was supposed that $d=39=100111$ and $(m/N)=-1$.

As shown in table 3, all the JS values of the intermediate steps in the algorithm are equal to 1, and it does not matter if $(m/N)=1$ or if $(m/N)=-1$.

3.2 Behavior of the Proposed Algorithm

The modular exponentiation m^d, where $d=\sum_{i=0}^{n-1} d_i 2^i$ and $d_i \in \{0,1\}$, can be represented by

$$(\cdots((m^{d_{n-1}})^2 \cdot m^{d_{n-2}})^2 \cdots m^{d_1})^2 \cdot m^{d_0} \quad (1)$$

If equation (1) is calculated using algorithm 1, it is possible to know that the last iteration of algorithm 1 can be represented by equation (2), which is the correct result of m^d

i	d_i	**Intermediate products**	**Jacobi symbol**
5	1	$R[0] = m^2 \cdot r^2$ $R[1] = (m^2 \cdot r)^2 = m^4 \cdot r^2$	$(R[0]/N) = (1)(1) = 1$ $(R[1]/N) = (1)(1) = 1$
4	0	$R[0] = (m^2 \cdot r^2)^2 = m^4 \cdot r^4$ $R[1] = m^6 \cdot r^4$	$(R[0]/N) = (1)(1) = 1$ $(R[1]/N) = (1)(1) = 1$
3	0	$R[0] = (m^4 \cdot r^4)^2 = m^8 \cdot r^8 = FA$ $R[1] = m^{10} \cdot r^8$	$(R[0]/N) = (1)(1) = 1$ $(R[1]/N) = (1)(1) = 1$
2	1	$R[0] = m^4 \cdot r^4 \cdot m^{10} \cdot r^8 = m^{14} \cdot r^{12}$ $R[1] = (m^{10} \cdot r^8)^2 = m^{20} \cdot r^{16}$	$(R[0]/N) = (1)(1) = 1$ $(R[1]/N) = (1)(1) = 1$
1	1	$R[0] = m^{34} \cdot r^{28}$ $R[1] = (m^{20} \cdot r^{16})^2 = m^{40} \cdot r^{32}$	$(R[0]/N) = (1)(1) = 1$ $(R[1]/N) = (1)(1) = 1$

TABLE 3: Algorithm 4 executed with JS attack where $d_{i+1} = d_i$.

$$m^{2^{n-1}(d_{n-1})+2^{n-2}(d_{n-2})+\cdots+2^1(d_1)} \cdot m^{(d_0)} \tag{2}$$

Now, it can be supposed that algorithm 4 is executed from d_{n-1} to d_0. Then, the modular exponentiation is represented by

$$(\cdots((m^{2d_{n-1}})^2 \cdot m^{2d_{n-2}})^2 \cdots m^{2d_1})^2 \cdot m^{2d_0} \tag{3}$$

The behavior of equation (3) through algorithm 4 is given by equations (4) to (7), where each step represents an iteration and $k = n-1-i$.

$$\text{Step } 1 \qquad (\cdots((m^{2^{1+1}(d_{n-1})+2^{0+1}(d_{n-2})})^2 \cdot m^{2(d_{n-3})})^2 \cdots m^{2(d_1)})^2 \cdot m^{2(d_0)} \tag{4}$$

$$\vdots \qquad\qquad\qquad\qquad \vdots$$

$$\text{Step } i \qquad (\cdots((m^{2^{i+1}(d_{n-1})+2^{i-1+1}(d_{n-2})+\cdots+2^{0+1}(d_k)})^2 \cdot m^{2(d_{k-1})})^2 \cdots m^{2(d_1)})^2 \cdot m^{2(d_0)} \tag{5}$$

$$\vdots \qquad\qquad\qquad\qquad \vdots$$

$$\text{Step } n-2 \qquad (m^{2^{n-2+1}(d_{n-1})+2^{n-3+1}(d_{n-2})+\cdots+2^{0+1}(d_1)})^2 \cdot m^{2(d_0)} \tag{6}$$

$$\text{Step } n-1 \qquad m^{2^{n-1+1}(d_{n-1})+2^{n-2+1}(d_{n-2})+\cdots+2^{1+1}(d_1)} \cdot m^{2(d_0)} \tag{7}$$

Note that equation (2) is very similar to equation (6). Now, if the last squaring and the last multiplication by $m^{2(d_0)}$ of equation (6) are deleted, then equation (8) is obtained

$$m^{2^{n-1}(d_{n-1})+2^{n-2}(d_{n-2})+\cdots+2^1(d_1)} \tag{8}$$

If equation (8) is multiplied by $m^{(d_0)}$, the correct result of the operation m^d has been obtained. Therefore, algorithm 4 is executed from $n-1$ to 1 (the last squaring and the last multiplication by

$m^{2(d_0)}$ are deleted), and it is necessary to multiply by m in line 11 of algorithm 4 (the multiplication by $m^{(d_0)}$ is made, but it is supposed that $d_0 = 1$, and thus, $m^{(d_0)} = m$).

Equations (9) and (10) are given only to show the relationship between the registers of algorithms 1 and 4, where $R[0]_{(o)_i}$ and $R[1]_{(o)_i}$ are the registers of algorithm 1 running from $n-1$ to 0; $R[0]_{(p)_i}$ and $R[1]_{(p)_i}$ are the registers of algorithm 4 running from $n-1$ to 1; and $d_{(p)}$ is a bit of the exponent in algorithm 4 at the iteration $i-1$.

$$\text{If } d_{(p)_{i-1}} = 0, \text{ then} \begin{cases} \mathrm{R}[0]_{(p)_\mathrm{i}} = R[0]_{(o)_{i-1}} \\ \mathrm{R}[1]_{(p)_\mathrm{i}} = R[1]_{(o)_{i-1}} \cdot m \end{cases} \tag{9}$$

$$\text{If } d_{(p)_{i-1}} = 1, \text{ then} \begin{cases} \mathrm{R}[0]_{(p)_\mathrm{i}} = R[0]_{(o)_{i-1}} \cdot m^{-1} \\ \mathrm{R}[1]_{(p)_\mathrm{i}} = R[1]_{(o)_{i-1}} \end{cases} \tag{10}$$

3.3 Expansion of the Algorithm

Up to this point, the discussion has addressed an algorithm that is effective when the keys d are odd, but it is possible to use algorithm 4 for all types of d values, by adding a few lines. The resulting algorithm is given below as algorithm 5.

Algorithm 5 can be used not only with odd keys, given as exponents, but also with even keys. To understand this option, recall that it is necessary to multiply the value m^{d_0} (where d_0 determines if a key is odd or even) by equation (8) to obtain the correct result of m^d, but $d_0 \in \{0,1\}$. If $d_0 = 1$, equation (8) is multiplied by $m^1 = m$, and if $d_0 = 0$, equation (8) is multiplied by $m^0 = 1$. Therefore, the *if statement* in algorithm 5 allows the algorithm to work with any kind of secret key d.

Algorithm 5 Modified FV scheme to counteract JS attack and to work with any exponent

1: **Input** $m \in G$, $d = (d_{n-1} \ldots d_0)_2$
2: **Output** $s = m^d \in G$
3: $R[0] \leftarrow r$
4: $R[1] \leftarrow r \cdot m^2$
5: $R[2] = r^{-1}$
6: **for** $n-1$ to 1 **do**
7: $\quad R[\overline{d_i}] \leftarrow R[\overline{d_i}] \cdot R[d_i] \bmod N$
8: $\quad R[d_i] \leftarrow R[d_i] \cdot R[d_i] \bmod N$
9: $\quad R[2] = R[2] \cdot R[2] \bmod N$
10: **end for**
11: **if** $d_0 = 1$ **then**
12: $\quad R[0] = R[0] \cdot m$
13: **end if**
12: **Return** $R[0] \cdot R[2] \bmod N$

Algorithm 5 uses more lines than algorithm 4; thus, when the d values are always odd numbers, algorithm 4 is recommended, and when the d values can be either odd or even numbers, algorithm 5 can be used.

4. CHARACTERISTICS OF THE PROPOSED ALGORITHM

The proposed algorithm is highly regular: there is always a multiplication followed by a squaring regardless of the processed bit. The relation between the registers $R[1] / R[0] = m^2$ is invariant throughout the execution of the algorithm.

Table 4 compares some characteristics of the proposed algorithm against the characteristics of other similar algorithms. Table 4 shows the number of registers and the average number of multiplications executed by the proposed algorithm, compared with algorithms derived from the original Montgomery powering ladder and the square-and-multiply algorithms. In table 4, the squarings are considered multiplications; the *if statements* are not considered; and n is the bit length of the exponent.

Algorithms	Number of registers	Average number of multiplications
Square-and-multiply left-to-right	1	1.5*n*
Square-and-multiply right-to-left	2	1.5*n*
Montgomery powering ladder	2	2*n*
Giraud's algorithm	2	2*n*
FV scheme	3	3*n*
Proposed algorithm	3	3*n*

TABLE 4: Comparison of the number of registers and the average number of multiplications executed by algorithms based on the Montgomery powering ladder and the square-and-multiply algorithms.

According to table 4, the proposed algorithm has disadvantages in runtime and number of registers compared with similar algorithms; however, these disadvantages are countered by the security characteristics of the proposed algorithm. Section 4.1 shows the level of security of the proposed technique with respect to other algorithms.

4.1 Security

Simple Power Analysis (SPA) [23] can recognize in a power trace, obtained from a device which executes a cryptographic algorithm, when a bit is equal to 0 and when it is equal to 1 if there are operations that depend on the bit's value being processed. The square-and-multiply algorithm is vulnerable to SPA because it has a conditional branch during its execution. The proposed algorithm does not have conditional operations and is therefore secure against SPA.

Because dummy operations are used in the square-and-multiply always algorithm, it can be attacked with the SEA, which consists of inducing a fault during the execution of the algorithm. If the fault affects a dummy operation ($d_i = 0$), the output result will not be altered, but if the fault affects a necessary operation ($d_i = 1$), the output result will be altered. Thus, an attacker can determine when a bit equal to 0 was attacked. The proposed algorithm does not have dummy operations that can be attacked and is thus resistant to the SEA.

To break the security of a cryptosystem with *Differential Power Analysis* (DPA) [23], it is necessary to collect many power traces of the same algorithm with different input values and perform a statistical analysis over them. The algorithm proposed by Giraud [6] and the Montgomery powering ladder are vulnerable to DPA, but the value r used by the proposed algorithm helps to avoid DPA.

RDA is an attack that uses two related messages M and M^2, and by observing the relationship between the two messages through the execution of the same algorithm, it can obtain the secret key of the cryptosystem. This attack was developed against the Montgomery powering ladder, but Giraud's algorithm is also vulnerable to it. The FV scheme and the proposed algorithm are resistant to this attack because the random value r breaks the relationship between M and M^2.

Kim and Quisquater showed the possibility of inducing two faults during the same execution of an algorithm [19]: the first fault to corrupt a register and the second fault to avoid an operation (such as a coherence test). Under this scheme, the algorithm proposed by Giraud can be vulnerable to the JS attack proposed by Schmidt because the coherence test will not be performed. Thus, the Giraud's algorithm will not recognize that the relationship between the registers has been lost, and an attacker can calculate the JS of the erroneous value, obtaining useful information[1].

It has been shown that the proposed algorithm is secure against the attack proposed by Schmidt and Medwed, whereas the FV scheme, Giraud's algorithm, and the Montgomery powering ladder are vulnerable against that attack.

As demonstrated in this section, the proposed algorithm offers better security than that offered by the other algorithms mentioned here.

5. COMMENTS

There is a concept that can be used to protect algorithms against this kind of attacks: by changing a quadratic non-residue value into a quadratic residue value (or working only with quadratic residue values through an algorithm, such as the proposed algorithm), it is possible to prevent an attacker from using a JS attack against a cryptographic algorithm.

As examples, the algorithms square-and-multiply right-to-left (SaM RtL) and square-and-multiply left-to-right (SaM LtR) are considered. As stated in section 2.3, Boreale attacked the SaM RtL algorithm (Algorithm 2). In this attack, if the squaring $R[1]_{i-1} = R[1]^2$ in the iteration $i-1$ is corrupted with a value z, where $(z/N) = -1$, and if the value of the bit in the i-th iteration is equal to 1, the JS value $(z/N) = -1$ will affect the operation $R[0] \leftarrow R[0] \cdot R[1] \bmod N$ in the i-th iteration, then $R[0]_i = (1) \cdot (-1) = -1$. (It is supposed that $(m/N) = 1$). Henceforth, the register $R[0]$ will have a JS equal to -1, a value that can be exploited by an attacker.

On the other hand, the SaM LtR (algorithm 6) cannot be attacked using Boreale's attack, because if it is placed an error in any operation $R[0] = R[0] \cdot m$ or $R[0] \leftarrow R[0]^2$ in the i-th iteration such that $(R[0]_i / N) = -1$ (It is supposed that $(m/N) = 1$), the operation $R[0]_{i-1} = R[0]_i{}^2$ will convert the JS value $(R[0]_i / N) = -1$ to $(R[0]_{i-1} / N) = 1$ in the next iteration of the algorithm. In other words, the operation $R[0]_{i-1} = R[0]_i{}^2$ will convert a quadratic non-residue value into a quadratic residue value, and this process will be repeated in each step of the *for* loop, which will avoid any kind of JS attack because there will be no any JS value that can be used to obtain relevant information about the cryptosystem.

Thus, the SaM LtR is intrinsically secure against JS attacks, because it converts any quadratic non-residue value into a quadratic residue value through its execution.

6. FUTURE WORK

Here, the blinded Montgomery ladder exponentiation algorithm has been protected against the Jacobi symbol attack. The modification of algorithm 1 was developed according to its specific characteristics, and according to the fault model used over it, but each modular exponentiation algorithm in the literature has different characteristics. To extend our results, we will develop forms to protect other algorithms that are vulnerable to the JS attack and that have different characteristics, such as the algorithms *Add only* and *Add always*, which were presented by Marc Joye in [24] and attacked in 2010 by Kim [25].

[1] Dottax et al. have proposed a method to resist the double-fault attack in [26].

Algorithm 6 Square-and-multiply left-to-right

1: **Input** $m \in G$, $d = (d_{n-1} \ldots d_0)_2$
2: **Output** $s = m^d \in G$
3: $R[0] \leftarrow 1$
4: **for** $n-1$ to 0 **do**
5: $R[0] \leftarrow R[0]^2 \bmod N$
6: **if** $d_i = 1$ **then**
7: $R[0] \leftarrow R[0] \cdot m \mod N$
8: **end if**
9: **end for**
10: **Return** $R[0]$

7. CONCLUSIONS

In this paper, we have proposed an algorithm that is secure against the attack proposed by Schmidt and Medwed. It has disadvantages in runtime and space compared to similar algorithms, but it also provides a higher level of security than these other algorithms.

Acknowledgments: We wish to thank the referee for carefully reading this paper and for his constructive suggestions. This paper was in part supported by the PACIVE project GC-19 of the FES-C UNAM.

8. REFERENCES

[1] P. Kocher. “Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems.” *In Koblitz, N., ed.: Advances in Cryptology-CRYPTO 96. Volume 1109 of Lecture in Notes in Computer Science*, 1996, pp. 104-113.

[2] D. Boneh, R. DeMillo and R. Lipton. “On the importance of checking cryptographic protocols for faults.” *In Fumy, W., Ed.: Advances in Cryptology-EUROCRYPT ’97. Volume 1233 of Lecture Notes in Computer Science*, 1997, pp. 37-51.

[3] J.S. Coron. “Resistance against differential power analysis for elliptic curve cryptosystems.” *In Ko, Paar, C., Eds.: Cryptographic Hardware and Embedded Systems-CHES 2002. Volume 1717 of Lecture Notes in Computer Science*, 1999, pp. 292-302.

[4] S.M. Yen, S. Kim, S. Lim, and S. Moon. “A countermeasure against one physical cryptanalysis may benefit another attack”. *Information Security and Cryptology-ICISC 2001, 2288 of Lecture Notes in Computer Science*, 2001, pp.414-427.

[5] M. Joye and S.M. Yen. “The montgomery powering ladder.” *In Cryptographic Hardware and Embedded Systems-CHES 2002, 2523 of Lecture Notes in Computer Science*, 2003, pp. 291-302.

[6] C. Giraud. “An rsa implementation resistant to fault attacks and to simple power analysis”. *IEEE Transactions on computers*, Vol. 55, No. 9, pp. 1116-1120, 2006.

[7] S.M. Yen, L.C. Ko, S.J. Moon, and J.C. Ha. “Relative doubling attack against montgomery ladder.” *In Information Security and Cryptology-ICISC 2005, 3935 of Lecture Notes in Computer Science*, 2005, pp. 117-128.

[8] P.A. Fouque and F. Valette. “The doubling attack–why upwards is better than downwards.” *In Cryptographic Hardware and Embedded Systems-CHES 2003, LNCS 2779*, 2003, pp. 269-280.

[9] G. Fumaroli and D. Vigilant. “Blinded fault resistant exponentiation.” *Fault Diagnosis and Tolerance in Cryptography, 4236 of Lecture Notes in Computer Science, 2006, pp.* 62-70.

[10] M. Boreale. “Attacking right-to-left modular exponentiation with timely random faults.” Fault *Diagnosis and Tolerance in Cryptography, 4236 of LNCS, pp.* 24-35, 2006.

[11] J. M. Schmidt and M. Medwed. “Fault attacks on the montgomery powering ladder”. *Information Security and Cryptology ICISC-2010,* pp. 396-406*,* 2011.

[12] H. Mamiya, A. Miyaji, and H. Morimoto. “Efficient countermeasures against rpa, dpa, and spa.” *Cryptographic Hardware and Embedded Systems-CHES 2004, 3156 of Lecture Notes in Computer Science*, 2004, pp. 343-356.

[13] C.C. Lu, S.Y. Tseng, and S.K. Huang. “A secure modular exponential algorithm resists to power, timing, c safe error and m safe error attacks.” *In 19th International Conference on Advanced Information Networking and Applications, 2005. AINA 2005, pp.* 151-154.

[14] C.H. Kim and J.J. Quisquater. “How can we overcome both side channel analysis and fault attacks on rsa-crt?.” *Workshop on Fault Diagnosis and Tolerance in Cryptography*, pp. 21–29, 2007.

[15] A. Boscher, R. Naciri, and E. Prouff. “Crt rsa algorithm protected against fault attacks.” *Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems, 4462 of LNCS*, pp.229-243, 2007.

[16] J.C. Ha, C.H. Jun, J.H. Park, S.J. Moon, and C.K. Kim. “A new crt-rsa scheme resistant to power analysis and fault attacks.” *Third 2008 International Conference on Convergence and Hybrid Information Technology*, 2008, pp. 351-356.

[17] A. Boscher, H. Handschuh, and E. Trichina. “Blinded fault resistant exponentiation revisited.” *In L. Breveglieri, S. Gueron, I. Koren, D. Naccache, and J.-P. Seifert, editors, Workshop on Fault Diagnosis and Tolerance in Criptography - FDTC’09*, 2009, pp. 3-9.

[18] S.M. Yen, W.C. Lien, S.J. Moon, and J.C. Ha. “Power analysis by exploiting chosen message and internal collisions-vulnerability of checking mechanism for rsa-decryption.” *Progress in Cryptology–Mycrypt 2005, 3715 of Lecture Notes in Computer Science*, 2005, pp. 183-195.

[19] C. Kim and J.J. Quisquater. “Fault attacks for crt based rsa: New attacks, new results, and new countermeasures.” *Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems, 4462*, pp. 215-228, 2007.

[20] S. Chari, J. Rao, and P. Rohatgi. “Template attacks.” *Cryptographic Hardware and Embedded Systems-CHES 2002, 2523 of Lecture Notes in Computer Science*, 2002, pp. 12–28.

[21] S.M. Yen and M. Joye. “Checking before output may not be enough against fault-based cryptanalysis.” *IEEE Transactions on Computers, 49(9), pp.* 967-970, 2000.

[22] S.M. Yen and C.S Laih. “Common-multiplicand multiplication and its application to public-key cryptography.” *Electronic Letters, 29(17),* pp. 1583-1584, August 1993.

[23] P.C. Kocher, J. Jaffe, and B. Jun. “Differential Power Analysis.” *In Wiener, M., Ed.: Advances in Cryptology-CRYPTO ’99. Volume 1666 of Lecture Notes in Computer Science*, Springer 1999, pp. 388-397.

[24] M. Joye. “Highly regular right-to-left algorithms for scalar multiplication.” *Cryptographic Hardware and Embedded Systems-CHES 2007, 4727 of Lecture in Notes in Computer Science*, 2007, pp. 135–147.

[25] C.H. Kim. “New fault attacks using jacobi symbol and application to regular right-to-left algorithms*.” Information Processing Letters, 110(20)*, pp. 882-886, 2010.

[26] E. Dottax, C. Giraud, M. Rivain, and Y. Sierra. “On second-order fault analysis resistance for CRT-RSA implementations*.” Information Security Theory and Practice. Smart Devices, Pervasive Systems, and Ubiquitous Networks,* pp. 68-83, Springer 2009.

Cryptography and Authentication Placement to Provide Secure Channel for SCADA Communication

AAmir Shahzad *mail2aamirshahzad@gmail.com*
Malaysian Institute of Information Technology (MIIT)
University Kuala Lumpur,Malaysia

Shahrulniza Musa *shahrulniza@miit.unikl.edu.my*
Malaysian Institute of Information Technology (MIIT)
University Kuala Lumpur,Malaysia

Abstract

Supervisory Control and Data Acquisition (SCADA) systems are real-time process control systems that are widely deployed throughout critical infrastructure sectors including power, gas, oil, railroads and water. . However, little attention is given to security considerations in the initial design and deployment of these systems, which has caused an urgent need to upgrade existing systems to withstand unauthorized intrusions potentially leading to communication attacks [1]. The current paper take a Hybrid-based Cryptography (combination of Symmetric AES and Asymmetric RSA) solution to enable confidentiality and authentication placed at each end of SCADA communication and provides secure channel for communication between Master Terminal Unit (MTU) to Remote Terminal Units (RTUs) and/or RTUs to MTU.

Keywords: Supervisory Control and Data Acquisition (SCADA), Cryptography, Security Issues.

1. INTRODUCTION

A Supervisory Control and Data Acquisition (SCADA) system allows equipment in many different locations to be monitored and controlled from a central location. The SCADA technology is utilized for industrial measurement and control systems and is commonly used by infrastructure and utility companies such as electric power generation, transmission, and distribution; oil and gas refining and pipelines; water treatment and distribution; chemical production and processing; railroads and mass transit; and manufacturing. A SCADA system normally supports communication between a central control unit and multiple remote units equipped with sensors, actuators, and/or Programmable Logic Controllers (PLCs). SCADA systems were first designed to meet the basic requirements of process control systems where security issues were hardly a concern. However, the growing demands for increased connectivity between a SCADA system and other network components, such as the corporate network or Internet, expose the critical parts of a SCADA system to the public. Therefore, security issues can no longer be ignored [1].

How secure are today's SCADA systems? Typical SCADA security measures consist of physically securing MTUs, RTUs, and transmission media, and employing common cyber security defenses such as password protection and anti-virus utilities. Communication security measures generally include private or leased telephone lines with a "secret" phone number and "secret" proprietary protocols. However, such measures are weak since it is not difficult to identify the secret phone number, tap a telephone line and decode proprietary protocols through reverse engineering. Some firms install firewalls and gateways but they fail to provide end-to-end security. Only a few private SCADA protocols have advanced level of built-in security features, such as message authentication, since most of these protocols were designed primarily to maximize performance, reliability, robustness, and functionality [4]. Here, we analyze security approaches to reduce some of the threats to SCADA communication.

2. SCADA ARCHITECTURE

SCADA architecture consists of one or more MTU terminal units (MTUs) used for supervising personnel, monitoring, and controlling a large number of remote terminal units (RTUs) or intelligent electronic devices (IEDs) installed in field. An MTU is a general purpose computer, running SCADA management software. RTUs and IEDs are generally small dedicated devices designed for rough field or industrial environment. MTUs retrieve real-time analog and binary status data from RTUs or IEDs, analyze these data, and send control commands to RTUs and/or IEDs automatically or manually by the supervisors.

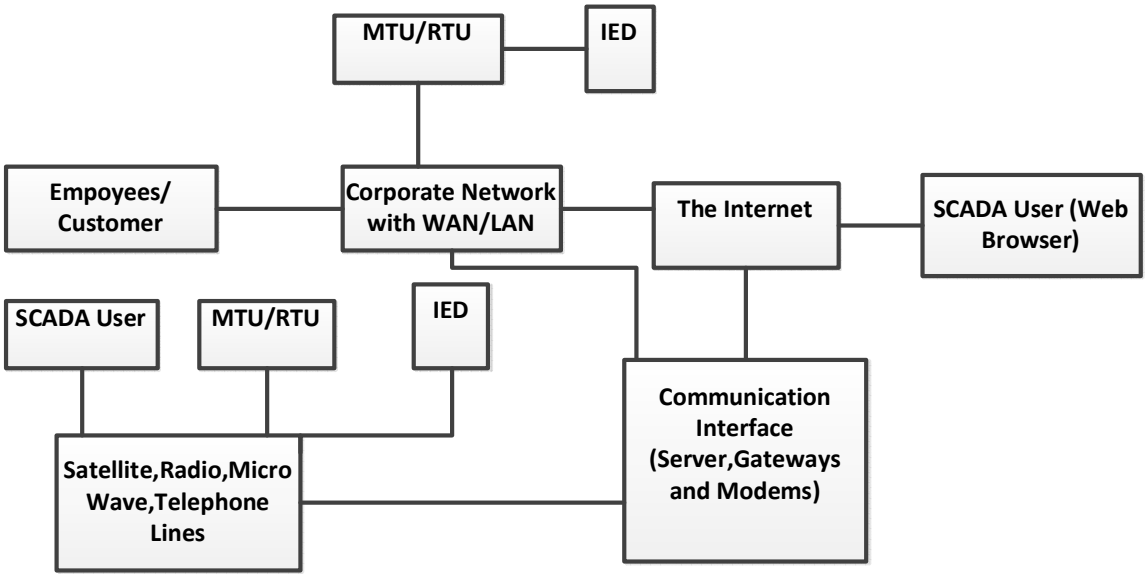


FIGURE 1: SCADA Architecture

The transmission of data and control commands between an MTU and an RTU, designated as SCADA communications are carried over a variety of media, including Ethernet, Frame relay, fiber channel, cellular systems, microwave signals, direct satellite broadcast and many licensed or unlicensed radio systems [4].

Common open communication-protocols include International Electrotechnical Commission 60870-5-101, Distributed Network Protocol Version 3 (DNP3), and Modbus, in addition to several other private protocols. Most of these protocols include application layer, link layer, and transport layer in their specifications. They also allow messages to be transported using TCP/IP specifications to facilitate communication over the Internet [4], [10].

3. LITERATURE REVIEW

Few publications are available on SCADA security, such as the American Gas Association Report No. 12 (AGA 12) [2], [3]. AGA 12 recommends practices designed to protect SCADA's MTU-RTU serial communication links from a variety of active/passive cyber attacks. One of these standards is AGA 12-1, Cryptographic Protection of SCADA Communications. The solution protects against hijacking or modifying the communication channel. AGA 12 requires the installation of multi-channel SCADA Cryptographic Modules (SCM) on a communications channel between the SCADA unit (e.g., host, RTU, IED) and the modem. A SCM receives and transmits SCADA messages on two communication ports: plaintext port and ciphertext port. The plaintext port is used to receive and transmit plaintext messages from a SCADA unit to a SCM, and the ciphertext port is used to transmit and receive ciphertext messages from a SCM to its peer.

SCM immediately begins transmitting a ciphertext message header to its peer as soon as it receives the first SCADA message characters. When enough characters of the cipher block are

received on the ciphertext port, it encrypts and transmits a block of ciphertext. When it finished transmitting all message blocks, it transmits a trailer that includes a Message Authentication Code (MAC) [1].At the receiving SCM, an incoming ciphertext message header signals the start of a new message. Each time when enough characters are received on the ciphertext port to fill a cipher block, the SCM decrypts the block and immediately begins forwarding the decrypted characters via its plaintext port to the receiving SCADA unit. When the trailer of the ciphertext message is received, the SCM computes and checks the MAC. By this time, the decrypted SCADA message may have already been forwarded entirely to the receiving SCADA unit. If the authentication check fails, it is too late to prevent forwarding the unauthentic message. Thus the authentication code only alerts the SCM to a possible failure of data integrity [9]. Such solution is limited and expensive. The standard does not protect an attack from a compromised field site or control center. In addition, SCADA owners need to install AGA 12 compliance multi-channel SCADA Cryptographic Module (SCM) and Key Management Appliance in the SCADA Control Center; and SCM and Maintenance Cryptographic Module attached to every Remote Terminal Unit (RTU). Moreover, AGA 12 is still in the early stages from a system implementation standpoint. The key management is a key component of the standards and is still in the development stage.

In another research, Graham and Patel [4] examined three security enhancements in SCADA communications to reduce the vulnerability of cyber attacks. This includes: (1) solutions that wrap the DNP3 protocols without making changes to the protocols, (2) solutions that alter the DNP3 protocols fundamentally, and (3) enhancements to the DNP3 Application. One of the research directions they identified is to secure the DNP3 protocol which is the focus of this paper. They provided high level description of possible solutions to protect SCADA communications and analysis for existing solutions such as DNP3 over IPSec or DNP over SSL/TLS. The main purpose of the paper is to identify the possible solutions to secure SCADA messages and for further research work to model and proof these solutions. The discussion about providing security for the DNP protocol is theoretical and describes the features of the proposed protocol at a very high level.

Symmetric-Key Encryption for Wireless Internet SCADA discusses internet SCADA, its connection through wireless communication and the security issues surrounding it. To answer the security issues, a symmetric-key encryption for wireless internet SCADA is proposed [5].

Message Encryption, the only good solution to the threats of eavesdropping and traffic analysis is complete encryption of a protocol stream. Unfortunately, encryption can be very processing-intensive and would not be a good solution for some of the smaller devices currently deploying DNP3 since this would Decrease communication Speed to a great extent [1], [3]. Another problem is that there are exporting, licensing, and key exchange issues with encryption that must be dealt.

Authentication using Message Authentication Object (MAO), is to detect modification of a transmitted message, an authentication object can be designed and can be appended to each message or to any DNP3 message that required authentication. The DNP Technical Committee has discussed a possibility of such an object called Message Authentication Object (MAO) [1], [3] which has fields for timestamp, nonce, hash-method, length, and hash value. It would contain the results of a secure hash function performed on the concatenation of the message and a secret, or password with only the valid sender and receiver knowing the secret. The hash would verify that the message has not been changed in transmission. Objects such as MAO will not be protected against eavesdropping or traffic analysis. Nevertheless, it can prevent outputs from being incorrectly activated by unauthorized users even if these users can eavesdrop on the network.

David Bailey and Edwin Wrigh carefully specified the SCADA system response time for the following events. Typically speed that are considered acceptable are: Display of analog and digital value (Acquire from RTU) on the MTU station operator display (1 or 2 second

maximum),Control request from operator to RTU (1 second critical;3 second non-critical),Acknowledge of alarm on operator screen (1 second),Display of entire new display on operator screen(1 second),Retrieval of historical trend and display on operator screen (2 second) and Sequence of event logged (at RTU) of critical event (1 millisecond).It is important that the response is consistent over all activities of the SCADA system[17].In another research, Chengzhi Li Bettati and R. Wei Zhao presents a new schedulability analysis methodology for distributed hard real-time systems with burst arrivals. The schedulability is analyzed by comparing worst-case response times of process with their timing constraints. Compute response times with a new method, which uses the amount of received service time to determine the response time of instances of a process [18].

4. ATTACK SCENARIO

The attack scenario in this part describes several ways the Malicious Intruder (Intruder) uses to compromise the security of SCADA systems and networks. The Intruder could use protocol analyzer tools such as “Ethereal” or other well known techniques to intercept the SCADA frames. As a result, the Intruder grabs unencrypted (plaintext) frames from a SCADA system network application. By doing so, the Intruder will capture the address of the source and destination systems. The Intruder could use the unencrypted data frames contain control and settings information in subsequent attacks on either the SCADA system or the Intelligent Equipment Devices (IEDs). Such attacks could take the form of shutting off the MTU, shutting down the MTU computer or the RTU stops functioning. In addition, the Intruder could change the settings on the IED, controller, or SCADA system such that the equipment either (a) fails to operate when it should, causing bus, line, or transformer damage, or (b) operates when it shouldn’t, causing service interruption [13].

In our scenario, a Intruder, after managing to get between the MTU and the RTU, intercepts the transmission of the frames and implement his/her attack in two phases:

Plan the Attack: An important feature of SCADA is the ability for the RTU to generate unsolicited report by exception (RBE) event and send it to the MTU. Unsolicited message (alarming) generation for event reporting is configurable by the MTU Station through the usage of the configuration functions in the application function code. The Intruder understands the structure of the SCADA and plans his attack by following these steps:

I. The MTU initiates a connection with the RTU.

II. Unknown to the MTU and the RTU, the Intruder is waiting to intercept their connection.

III. The Intruder receives MTU’s request for a connection (authentication capability is not implemented in SCADA, so the Intruder does not have to authenticate himself to the RTU).

IV. The source address (192.168.0.1), the destination address (192.168.0.2), the function codes, and the data objects are available in clear text

Disable SCADA unsolicited messaging (alarming) by attacking one or more RTU units: The Intruder implements his/her attack by following these steps:

V. The Intruder then initiates a connection with the RTU posing as MTU.

VI. The Intruder sends a message to RTU unit (192.168.0.2), with code function code 00021 (disable unsolicited messages).

VII. RTU unit (192.168.0.2) receives the message and disables unsolicited messages function. At this point the RTU will not be able to send any alarming messages to the MTU in case there is a failure or abnormal operation at the RTU unit.

VIII. The Intruder sends another message with code function code 00054 to RTU (192.168.0.2) .Code 00054 gives instructions to the RTU to stop running the application specified in the message.

IX. A simultaneous attack on other RTU units will disable all operations on a communication channel. This could interrupt the utilities services at that region, like shutting down the electricity services.

X. At this stage, the MTU Terminal Unit (MTU) in the MTU Station reports that the application is running normally, while the Remote Terminal Unit (RTU) in the RTU Stations receives tampered frames. The best way to protect a SCADA communications network is the correct and conscious use of cryptographic and an authentication technique in both the MTU and the RTU ends.

5. MTU/RTU COMMUNICATION

The MTU can address individual RTUs, or can initiate a broadcast message to all RTUs. RTUs return a message (response) to requests that are addressed to them individually. The SCADA protocol (used) establishes the format for the MTU's request message by placing it into the RTU (or broadcast) address, a function code defining the requested action, any data to be sent, and an error-checking field. The RTU's response message is constructed using SCADA protocol and contains fields confirming the action taken (if requested), any data to be returned, and an error checking field. If an error occurred in receipt of the message, or if the RTU is unable to perform the requested action, the RTU will construct an error message and send it as its response.

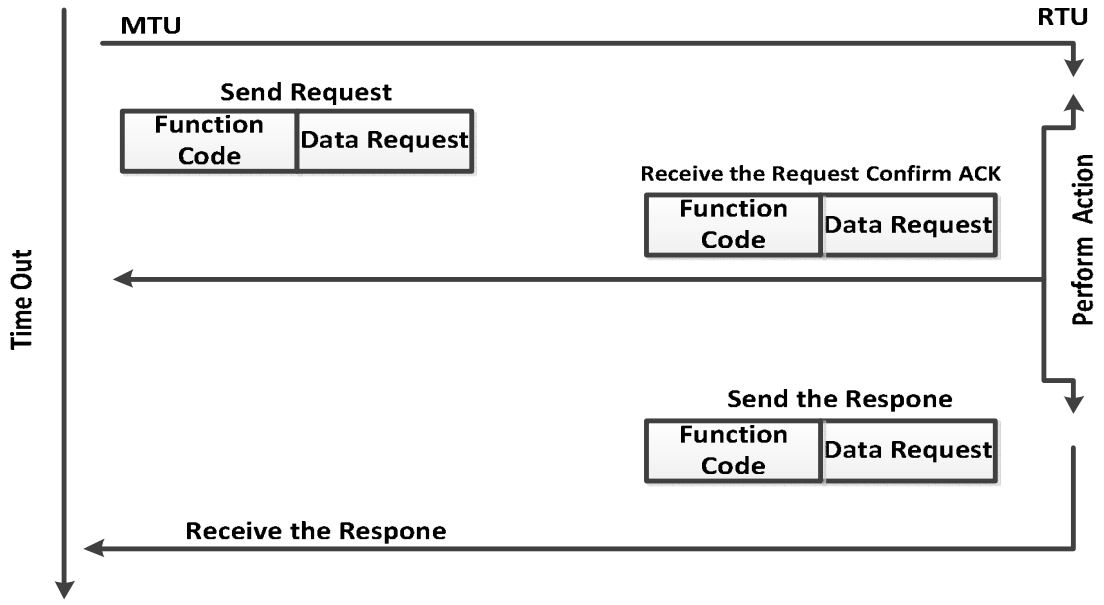


FIGURE 2: MTU/RTU Communication Process

Figure 2 shows the communication process between a MTU and a RTU. The figure illustrates a MTU initiates a request of data from a RTU; this could be a poll for current data. Also, the figure illustrates the communication sequence between a MTU and a RTU with message direction shown between them.

The request message is contained in the application layer information within the message. A confirmation (acknowledge) response is required to this message. The RTU station sends an ACK message to the MTU. Since the last transaction contained an application level request for

the transmission of data, the RTU station then performs the action requested and initiates a communication with the requested data.

6. APPROACH ANALYSIS

Reliability and time to delivery of frames are very important requirements for SCADA Systems. These requirements are vital to market acceptance of a particular SCADA security implementation. SCADA added more efficient reliability and security capabilities by introducing cryptographic and authentication capabilities in the framework. Such capabilities introduced new challenges related to time to delivery of the frames. But we will show soon that these challenges are not significant and our framework maintains a good balance between security, reliability and time to deliver.

SCADA provides several different means of retrieving data. These methods for retrieving data require different means of efficiency, quiescent and unsolicited report-by-exception operation requires real-time efficiency. The time of retrieving data from the RTU or the time the RTU needs to send unsolicited messages to the MTU should not be significantly delayed by the implementation of hybrid cryptography. More detailed performance analysis related to the implementation of SCADA needs to be conducted. Several performance studies on the effect of cryptography on the set-up time and the delivery of the messages from one end to the other indicate that the delay is not significant based on the advanced technologies in the communication networks, processing power at the end systems, and the cryptographic algorithms [14], [15],[16].

In a study by Kim and Montgomery [12] they examined the dynamic behavior and relative performance characteristics of large scale VPN environments based upon IPSec and IKE. The results of their study are summarized in the following table:

Operation, based on 128 bit key	**DES**	**3-DES**
Encryption Speed (Kbit/s)	10508 kbit/sec	4178 kbit/sec
Decryption Speed (Kbit/s)	10519 kbit/sec	4173 kbit/sec

TABLE 1. Performance Characteristics

Based on the performance information above, we will calculate the worst case scenario to measure the time of delivery for the unsolicited message from the RTU to the MTU, which required real-time delivery. Although, the numbers are far from exact, they should be usable as a first approximation. The total time to deliver such message is the sum of the encryption speed (ES), the decryption speed (DS), encryption key set up (EK), decryption key set up (DK), and the transmission time (TT).

$$\text{Unsolicited delivery time} = ES + DS + EK + DK + TT$$

We assume that the size of the SCADA message is 292 bytes, Triple DES is the algorithm of choice with 112 bit key, the network bandwidth is 1.5 Mbps, and the performance speed is measured in kbit/s. The EK and DK are not applicable in our case since we are assuming that we are using manual distribution of the session keys during the installation of SCADA components. The table below shows the performance of each operation:

Operation, based on 128 bit key	Performance	Time
Encryption Speed (Kbit/s)	4173 Kbits/sec	.00007 sec
Decryption Speed (Kbit/s)	4173 Kbits/sec	.00007 sec
Transmission Time	1.5 Mbit/sec	.0002 sec

TABLE 2. Unsolicited Delivery Time

As a result the unsolicited delivery time is equal to .00034 sec. Even if we double this number to accommodate for the authentication calculation time, we believe that this is a very minimum time to have an effect on the delivery time of the unsolicited messages in the SCADA systems. Accordingly, adding the operations above to include cryptographic and authentication operations will not affect the efficiency and the speed of delivery of SCADA messages.

7. PROPOSED WORK

SCADA systems were electronically isolated from all other networks and hence not likely to be accessed by outside attackers. As a result, the security issues of a SCADA system focused on physical security such as physical access control. However, the fact is that the growing demands of the industry for increased connectivity between the SCADA systems and the corporate network (Internet) result in an increase in security threats and vulnerabilities that are not limited to physical attacks. A recent study shows that almost 70% of the reported incidents of SCADA systems were either due to accidents or to disgruntled Insiders acting maliciously [11],[6].

Current paper address the security related to Authentication and Confidentiality of SCADA and provide secure channel for communication between MTU Terminal Unit (MTU) to Remote Terminal Units (RTUs) and/or RTUs to MTU.With these objectives the security of existing SCADA systems can be significantly enhanced to secure network communication.

7.1 Hybrid Cryptosystem

In cryptography, public-key cryptosystems are convenient in that they do not require the sender and receiver to share a common secret in order to communicate securely (among other useful properties). However, they rely on complicated mathematical computations and are thus generally much heavy processing than comparable symmetric-key cryptosystems. In many applications, the high cost of encrypting long messages in a public-key cryptosystem can be prohibited. A hybrid cryptosystem combines the convenience of a public- key cryptosystem with the efficiency of a symmetric-key cryptosystem

A hybrid cryptosystem can be constructed using any two separate cryptosystems: A key encapsulation scheme, which is a public-key cryptosystem, and A data encapsulation scheme, which is a symmetric-key cryptosystem. Note that for very long messages the bulk of the work in encryption/decryption is done by the more efficient symmetric-key scheme, while the heavy processing public-key scheme is used only to encrypt/decrypt a short key value.

7.2 Proposed Implementation

All keys (Such as symmetric and public) are stored locally in database using MySQL , eliminating the need of the Certificate Authority[4].We manually configure each MTU/RTU with common symmetric keys. This could be a good solution for the SCADA systems since these systems are relatively static. The MTU/RTU is only going to be exchanging data with its predefined MTU/RTU.

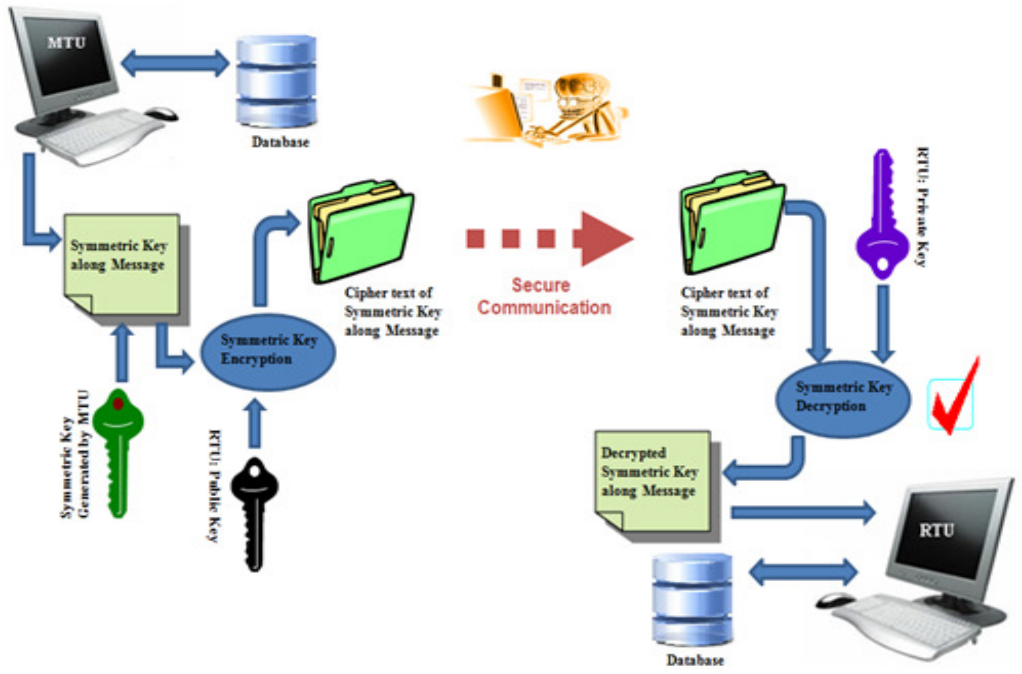


FIGURE 3: SCADA Authentication and Confidentiality

The database has two tables, one MTU-key table in MTU with three fields (RTU public key, MTU-RTU symmetric key and Time-Stamp) and second is RTU-key table in RTU with two fields (MTU public key and MTU-RTU symmetric key).Table 3 and 4 shows the database fields for MTU and RTU and figure 3,4 illustrates the overall view of propose implementation

Public key	Symmetric key	Time-Stamp
7868	5634	0930:0946
……	……	……

TABLE 3: MTU-Key Table

Public key	Symmetric key
7788	5634
……	……

TABLE 4: RTU-Key Table

MTU fetch the RTU public key from MTU-key table and then generates a fresh symmetric key for the data encapsulation scheme, and encrypt symmetric Key just generated under the key encapsulation scheme, using RTU public key than send this encrypted symmetric key along with message to RTU. Here the message itself is not encrypted to save the processing time during encryption/decryption. At other end, RTU fetches the MTU record from RTU-key table and uses its own private key to decrypt the symmetric Key contained in the key encapsulation segment. If symmetric key decrypted successful than RTU has access to open the message transmitted with encrypted symmetric Key from MTU. As result, RTU would conclude that the message came from an authentic source if message can decrypted successful. The RTU would also conclude that the message contents are unaltered if the symmetric key match. Current proposed technique also

applied to the message send from RTU to MTU to prevent an intruder from making MTU send inappropriate messages

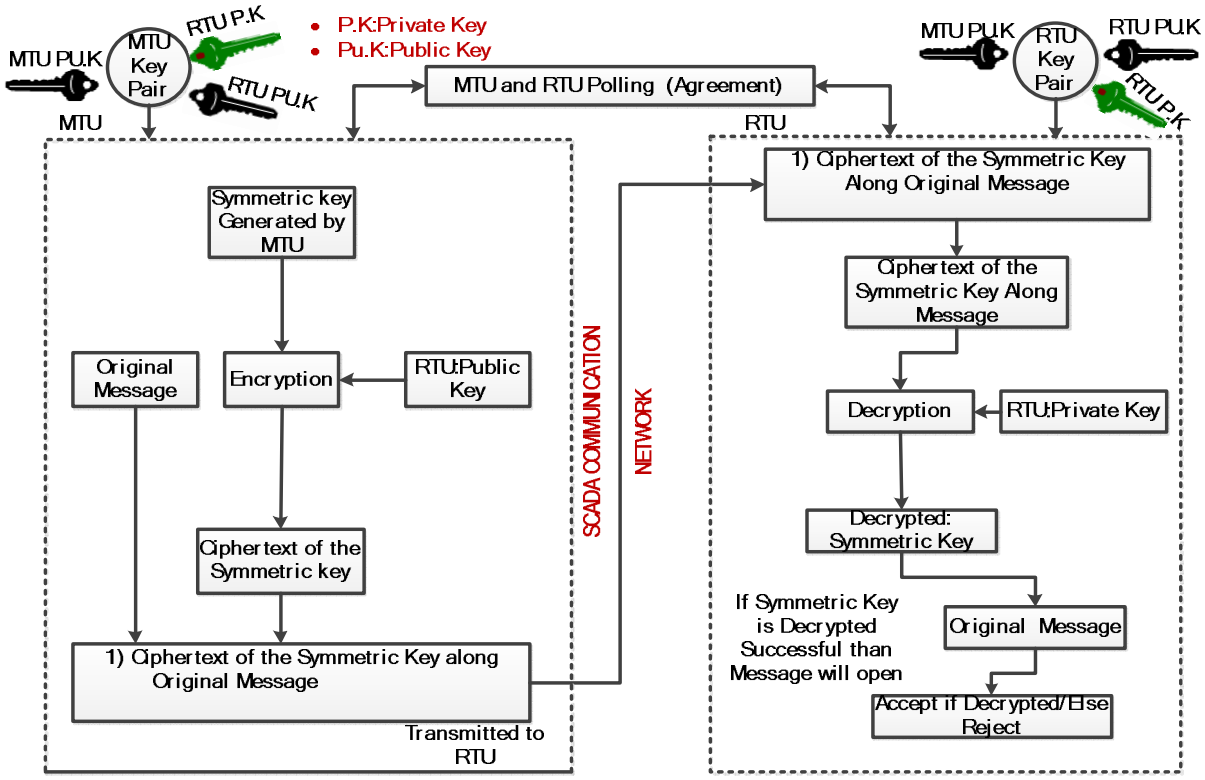


FIGURE 4. Block Diagram of Hybrid Cryptography

7.3 Performance Results

In prototype, two RTUs are locally installed and connected with MTU within LAN. RTU1 (With IP: 192.168.0.2) and RTU2 (With IP: 192.168.0.3) are connected with MTU (With IP: 192.168.0.1) via switch located distance of three and five meters range. Figure 5 illustrates the connectivity between the MTU and RTUs and table 5, 7 and 9 shows the performance results. The experimental setups, table 6, 8 and 10 used to measure propagation delay when data are sent from RTUs to MTU using TCP. The experiment is carried out with the bandwidth of 1.5 Mbps and carefully observed the performance characteristic of data. For each experimental run, data packets are sent from RTU to the MTU. Experiments are run using TCP (Transport Control Protocol) as UDP is not suitable in SCADA as UDP does not provide message guarantee services.

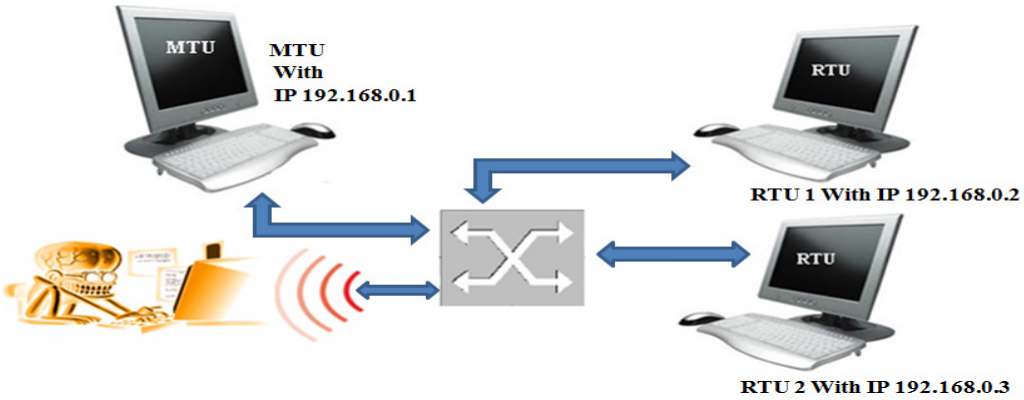


FIGURE 5: Connectivity between the MTU and RTUs

Current paper uses Visual Studio 2010 platform which provide C# and VC++ library for implementation. Here is a simple implementation .

```
   public class Cryptography
   {
      public const string KeyElementName = "EncryptionKey";
      public const string EncryptedElementName = "Encrypted";
      public const string CredentialsElementName = "Credentials";
      public const string AllElementName = "s:Envelope";
      public const int AesKeySize = 128; //192 and 256 bits also use
      public const int RsaKeySize = 1024;
      protected const bool Content = false;
    public static RSACryptoServiceProvider  RsaServiceProvider
{
get; private set;
}
protected static ConcurrentDictionary<string, byte[]> AesKeys
 {
get; private set;
}
  static Cryptographer()
      {
RsaServiceProvider=new RSACryptoServiceProvider(RsaKeySize);
AesKeys = new ConcurrentDictionary<string,byte [ ]>();
      }
    }
  }
```

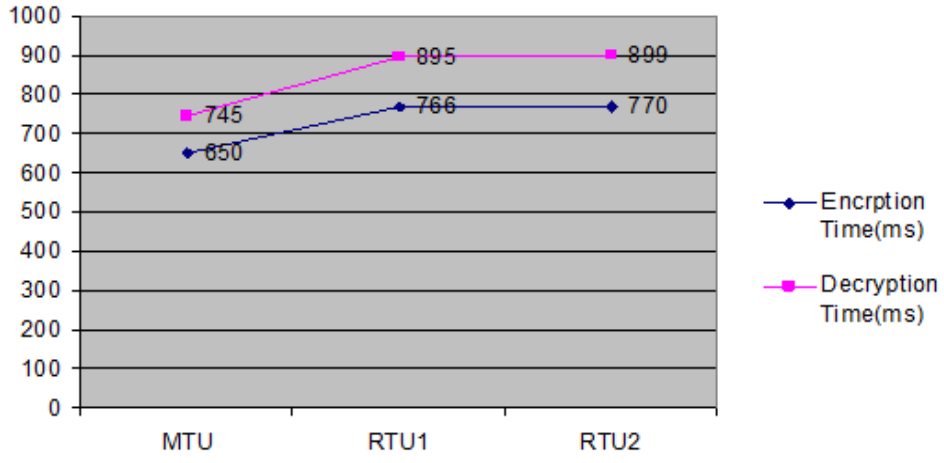
7.3.1 Performance Results Using AES (128) and RSA (1024) bits

Operations	**Performance**		
AES(128) & RSA(1024) bits	**MTU**	**RTU1**	**RTU2**
Encryption (Message)	5899 Kbits	5899 Kbits	5899 Kbits
Encryption Time in (millisecond)	650	766	770
Operations	**Performance**		
Decryption (Message)	6701 Kbits	6701Kbits	6701 Kbits
Decryption Time (millisecond)	745	895	899
Bandwidth	1.5 Mbps	1.5 Mbps	1.5 Mbps

TABLE 5: Performance Results Test1

Based on the performance information test1 above, we have calculated the scenario to measure the time of message encryption/decryption, which required real- time delivery. Although, the timing results are not far from exact because results are calculated by Prototype. We assume that the size of message is 5899 Kbits for encryption and 6701 Kbits for decryption, AES key size is 128bits, RSA key size is 1024 bits, the network bandwidth is 1.5 Mbps, and the encryption/decryption operation times are measures in milliseconds. As results from table 5 we conducted two sets of measurements .We first measured the MTU/RTUs message encryption times and then we measured the MTU/RTUs message decryption times.

In first measurement, MTU send encrypted message (5899 Kbits) to RTUs. The total time spend for massage encryption is 650 milliseconds while message (5899 Kbits) encryption time for RTU1 is 766 milliseconds and RTU2 take 770 milliseconds for message (5899 Kbits) encryption with the bandwidth of 1.5 Mbps. In Second measurement, MTU send decrypted message (6701 Kbits) to RTUs. The total time spend for massage decryption is 745 milliseconds while message (6701 Kbits) decryption time for RTU2 is 899 milliseconds and RTU2 take 899 milliseconds for message (5899 Kbits) decryption with the bandwidth of 1.5 Mbps.



GRAPH 1: Performance Results Test1

The above two measurements, MTU/RTUs have different timing for encryption/decryption operation. In encryption operation, RTU1 has 116 milliseconds and RTU2 has 120 milliseconds difference compared with MTU encryption operation. While in decryption operation, RTU1 has

150 milliseconds and RTU2 has 154 milliseconds difference compared with MTU decryption operation.

Operations AES(128) & RSA(1024) bits (hh:mm:ss:ms)	**Data Received at MTU (hh:mm:ss:ms)**	**Propagation delay (hh:mm:ss:ms)**
Data sent from RTU1 10:47:08:130	10:47:08: 138	00:00:00:008
Operations AES(128) & RSA(1024) bits (hh:mm: ss: ms)	**Data Received at MTU (hh:mm: ss: ms)**	**Propagation delay (hh:mm: ss: ms)**
Data sent from Remote RTU2 11:47:11:150	11:47:11:160	0:00:00.010

TABLE 6: Performance Results Test1 Propagation Delay

7.3.2 Performance Results Using AES (192) and RSA (1024) bits

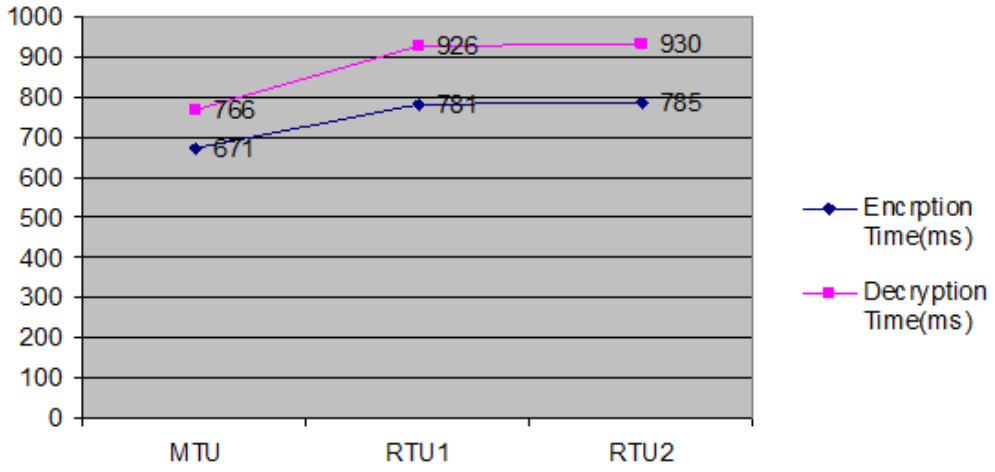
Operations	**Performance**		
AES(192)& RSA(1024) bits	**MTU**	**RTU1**	**RTU2**
Encryption (Message)	5899 Kbits	5899 Kbits	5899 Kbits
Encryption Time in (millisecond)	671	781	785
Decryption (Message)	6701 Kbits	6701Kbits	6701 Kbits
Decryption Time (millisecond)	766	926	930
Bandwidth	1.5 Mbps	1.5 Mbps	1.5 Mbps

TABLE 7: Performance Results Test2

Based on the performance information test2 above, we have calculated the scenario to measure the time of message encryption/decryption, which required real- time delivery. Although, the timing results are not far from exact because results are calculated by Prototype.

We assume that the size of message is 5899 Kbits for encryption and 6701 Kbits for decryption, AES key size is 192 bits, RSA key size is 1024 bits, the network bandwidth is 1.5 Mbps, and the encryption/decryption operation times are measures in milliseconds. As results from table 7 we conducted two sets of measurements .We first measured the MTU/RTU message encryption times and then we measured the MTU/RTU message decryption times.

In first measurement, MTU send encrypted message (5899 Kbits) to RTUs. The total time spend for massage encryption is 671 milliseconds while message (5899 Kbits) encryption time for RTU1 is 781 milliseconds and RTU2 take 785 milliseconds for message (5899 Kbits) encryption with the bandwidth of 1.5 Mbps. In Second measurement, MTU send encrypted message (6701 Kbits) to RTUs. The total time spend for massage decryption is 766 milliseconds while message (6701 Kbits) decryption time for RTU1 is 926 milliseconds and RTU2 take 930 milliseconds for message (5899 Kbits) decryption with the bandwidth of 1.5 Mbps.



GRAPH 2: Performance Results Test2

The above two measurements, MTU/RTUs have different timing (Propagation delay) for encryption/decryption operation. In encryption operation, RTU1 has 110 milliseconds and RTU2 has 112 milliseconds difference compared with MTU encryption operation. While in decryption operation, RTU1 has 160 milliseconds and remote RTU has 164 milliseconds difference compared with MTU decryption operation.

Operations AES(192) & RSA(1024) bits (hh:mm:ss:ms)	**Data Received at MTU (hh:mm:ss:ms)**	**Propagation delay (hh:mm:ss:ms)**
Data sent from RTU1 11:48:10:143	11:48:10:155	00:00:00:012
Data sent from RTU2 11:50:11:161	11:50:11:175	0:00:00.014

TABLE 8. Performance Results Test2 Propagation Delay

7.3.3 Performance Results Using AES (256) and RSA (1024) bits

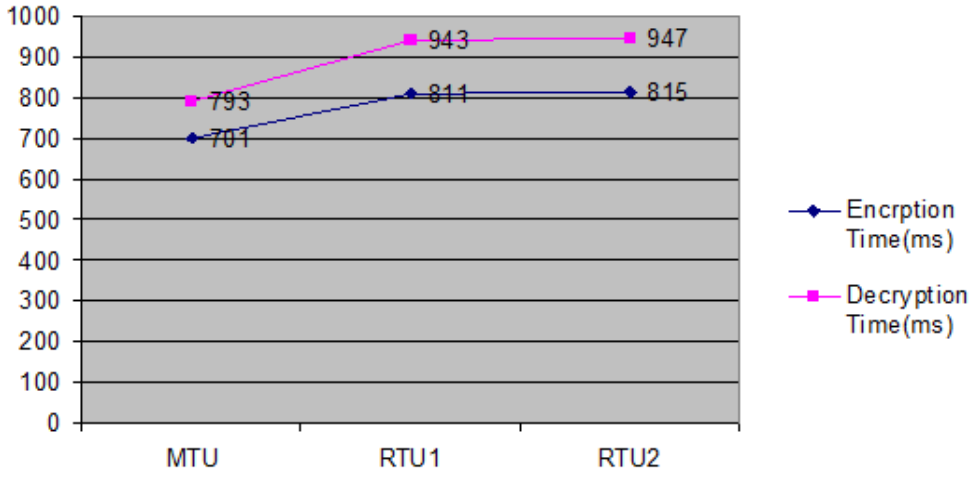
Operations	**Performance**		
AES(256)& RSA(1024) bits	**MTU**	**RTU1**	**RTU2**
Encryption (Message)	5899 Kbits	5899 Kbits	5899 Kbits
Encryption Time in (millisecond)	701	811	815
Decryption (Message)	6701 Kbits	6701Kbits	6701 Kbits
Decryption Time (millisecond)	793	943	947
Bandwidth	1.5 Mbps	1.5 Mbps	1.5 Mbps

TABLE 9: Performance Results Test3

Based on the performance information test3 above, we have calculated the scenario to measure the time of message encryption/decryption, which required real- time delivery. Although, the timing results are not far from exact because results are calculated by Prototype.

We assume that the size of message is 5899 Kbits for encryption and 6701 Kbits for decryption, AES key size is 256 bits, RSA key size is 1024 bits, the network bandwidth is 1.5 Mbps, and the encryption/decryption operation times are measures in milliseconds. As results from table 7 we conducted two sets of measurements .We first measured the MTU/RTU message encryption times and then we measured the MTU/RTU message decryption times.

In first measurement, MTU send encrypted message (5899 Kbits) to RTUs. The total time spend for massage encryption is 701 milliseconds while message (5899 Kbits) encryption time for RTU1 is 811 milliseconds and RTU2 take 815 milliseconds for message (5899 Kbits) encryption with the bandwidth of 1.5 Mbps. In Second measurement, MTU send encrypted message (6701 Kbits) to RTUs. The total time spend for massage decryption is 793 milliseconds while message (6701 Kbits) decryption time for RTU1 is 943 milliseconds and RTU2 take 947 milliseconds for message (5899 Kbits) decryption with the bandwidth of 1.5 Mbps.



GRAPH 3: Performance Results Test3

The above two measurements, MTU/RTUs have different timing (Propagation delay) for encryption/decryption operation. In encryption operation, RTU1 has 110 milliseconds and RTU2 has 114 milliseconds difference compared with MTU encryption operation. While in decryption operation, RTU1 has 150 milliseconds and RTU2 has 154 milliseconds difference compared with MTU decryption operation.

Operations AES(256) & RSA(1024) bits (hh:mm:ss:ms)	**Data Received at MTU (hh:mm:ss:ms)**	**Propagation delay (hh:mm:ss:ms)**
Data sent from RTU1 12:33:08:130	12:33:08: 146	00:00:00:016
Data sent from RTU2 12:41:11:155	12:41:11:175	0:00:00.020

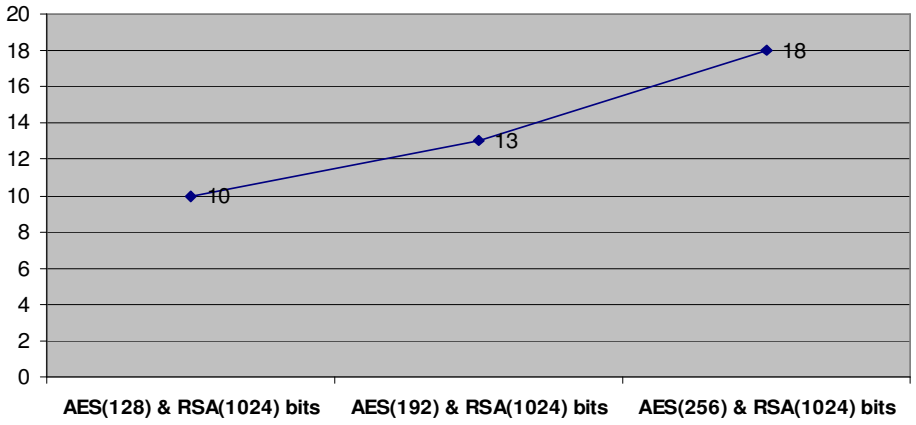
TABLE 10: Performance Results Test3 Propagation Delay

7.3.4 Summary of Experimental Results With Mean

Operations	RTU(Connect via switch) to MTU Propagation delay(ms)	RTU(Remote) to MTU Propagation delay(ms)	Mean Propagation delay(ms)
AES(128) & RSA(1024) bits	08	10	9
AES(192) & RSA(1024) bits	12	14	13
AES(256) & RSA(1024) bits	16	20	18

TABLE 11: Summary of Experimental Results

Table 11 and Mean Propagation delay (ms) Graph 4 summarizes the performance results in the form of mean delay. As result, propagation delay increased with the increasing of key size (RSA and AES). The mean delay with RSA 1024 and AES 128 key size is 10 ms while mean delay increases to 13 ms and 18 ms using RSA 1024 and AES 192 bits key and RSA 1024 and AES 256 bits key.



GRAPH 4: Mean Propagation delay (ms)

8. CONCLUSION

SCADA systems are significantly important systems used in national infrastructures such as electric grids, water supplies and pipelines. However, the SCADA systems have lots of security vulnerabilities. Any attacks or damages of the SCADA systems can affect to the society severely. The study of the security for SCADA systems is essential for that reasons The current paper take a Hybrid-based Cryptography (combination of Symmetric AES and Asymmetric RSA) solution to enable confidentiality and authentication placed at each end of SCADA communication and provides secure channel communication between MTU Terminal Unit (MTU) to Remote Terminal Units (RTUs) and/or RTUs to MTU. By implementing proposed Hybrid Cryptography solution, includes carried out real time experimental analysis, encryption/decryption operation to enable confidentiality and authentication and total delay are within the allowable delay for SCADA systems. Therefore, current paper suggests new research direction to more adequately secure SCADA communication over the long run.

9. ACKNOWLEDGMENTS

I am sincerely and heartily grateful to my supervisor, Shahrulniza Musa, for the support and guidance he showed me throughout my research paper writing. I am sure it would have not been possible without his help. Besides I would like to thank to my parents and my best friend, Muhammad Irfan boosted me morally and provided me great information resources.

10. REFERENCES

[1] J.H. Graham and S.C. Patel. "Security Considerations in SCADA Communication Protocols," Sept 2004.

[2] P.Blomgren and S.M Kotronx. "Cryptographic Protection of SCADA Communications Part 1: Background, Policies and Test Plan," American Gas Association (AGA), Draft 4, AGA Report 12, Mar.14.2006.

[3] M.D. Hadley and K.A. Huston and T.W. Edgar. "American Gas Association (AGA), Report No. 12 Part 2.Cryptographic Protection of SCADA Communications: Retrofit Link Encryption for Asynchronous Serial Communications", Aug. 2007.

[4] S.C. Patel and G.D. Bhatt and J.H. Graham. "Improving the cyber security of Scada communication Network," Communication of ACM, Vol .52 No.7, July.2009.

[5] R.J.Robles and M.K.Choi. "Symmetric-Key Encryption for Wireless Internet SCADA," Springer-Verlag Berlin Heidelberg, Communications in Computer and Information Science, Volume 58, 289-297, DOI: 10.1007/978-3-642-10847-1_36, 2009.

[6] M.Jethanandani and C.FI. "CERT Vulnerability Report in TCP," April .20. 2004.

[7] DNP3 Organization homepage: http://www.dnp.org/

[8] Modbus Organization. http://www.modbus.com/

[9] A. Wright and J. Kinast and J. M.Carty. "Low-Latency Cryptographic Protection for SCADA Communication," Springer Lecture Notes, 2006.

[10] R.D. Colin and B.E. Dawson. "A Key Management Architecture for SCADA Systems," ACM journal, 2006.

[11] E.B.Fernandez and J.Wu and M.M.Larrondo. "On Building Secure SCADA Systems using Security Patterns,"ACM conference, April .2009.

[12] Kim and Montgomery. "Behavioral and Performance Characteristics of IPSec/IKE in Large-Scale VPNs," Proceedings of the IASTED, International Conference on Communication Network and Information Security," pp. 231-236, Dec. 2003.

[13] P.Oman and E.O. Schweitzer. "Substations, and SCADA Systems against Electronic Intrusions," Schweitzer Engineering Laboratories, Inc. Pullman, WA USA.

[14] G.Clarke and D. Reynders. "Practical Modern SCADA Protocols," May.2003.

[15] E.Nahum and S.O.Malley. "Towards High Performance Cryptographic Software," ftp://ftp.cs.arizona.edu/reports/1995/TR95-03.ps.

[16] Schneier and Bruce. "Performance Comparison of the AES Submissions," http://www.schneier.com/paper-aes.

[17] D.Bailey and Edwin. "Wright, Practical SCADA for industry." Available: http://books.google.com.my/books

[18] C.Li.Bettati and R. W. Zhao. "Response time analysis for distributed real-time systems with bursty job arrivals," Dept. of Compute. Sci., Texas A&M Univ., College Station, TX, http://ieeexplore.ieee.org/Xplore.

INSTRUCTIONS TO CONTRIBUTORS

Information Security is an important aspect of protecting the information society from a wide variety of threats. The International Journal of Security (IJS) presents publications and research that builds on computer security and cryptography and also reaches out to other branches of the information sciences. Our aim is to provide research and development results of lasting significance in the theory, design, implementation, analysis, and application of secure computer systems.

IJS provides a platform to computer security experts, practitioners, executives, information security managers, academics, security consultants and graduate students to publish original, innovative and time-critical articles and other information describing research and good practices of important technical work in information security, whether theoretical, applicable, or related to implementation. It is also a platform for the sharing of ideas about the meaning and implications of security and privacy, particularly those with important consequences for the technical community. We welcome contributions towards the precise understanding of security policies through modeling, as well as the design and analysis of mechanisms for enforcing them, and the architectural principles of software and hardware system implementing them.

To build its International reputation, we are disseminating the publication information through Google Books, Google Scholar, Directory of Open Access Journals (DOAJ), Open J Gate, ScientificCommons, Docstoc and many more. Our International Editors are working on establishing ISI listing and a good impact factor for IJS.

The initial efforts helped to shape the editorial policy and to sharpen the focus of the journal. Starting with volume 5, 2011, IJS appears in more focused issues. Besides normal publications, IJS intend to organized special issues on more focused topics. Each special issue will have a designated editor (editors) – either member of the editorial board or another recognized specialist in the respective field.

We are open to contributions, proposals for any topic as well as for editors and reviewers. We understand that it is through the effort of volunteers that CSC Journals continues to grow and flourish.

IJS LIST OF TOPICS

The realm of International Journal of Security (IJS) extends, but not limited, to the following:

- Anonymity
- Attacks, security mechanisms, and security service
- Authorisation
- Cellular/wireless/mobile/satellite networks securi
- Public key cryptography and key management
- Cryptography and cryptanalysis
- Data integrity issues
- Database security
- Denial of service attacks and countermeasures
- Design or analysis of security protocols
- Distributed and parallel systems security
- Formal security analyses
- Information flow
- Intellectual property protection
- Anonymity and pseudonymity
- Code security, including mobile code security
- Biometrics
- Authentication
- Confidentiality, privacy, integrity, authenticatio
- Data confidentiality issues
- Data recovery
- Denial of service
- Dependability and reliability
- Distributed access control
- Electronic commerce
- Fraudulent usage
- Information hiding and watermarking
- Intrusion detection

- Key management
- Network and Internet security
- Network security performance evaluation
- Peer-to-peer security
- Privacy protection
- Revocation of malicious parties
- Secure location determination
- Secure routing protocols
- Security in ad hoc networks
- Security in communications
- Security in distributed systems
- Security in e-mail
- Security in integrated networks
- Security in internet and WWW
- Security in mobile IP
- Security in peer-to-peer networks
- Security in sensor networks
- Security in wired and wireless integrated networks
- Security in wireless communications
- Security in wireless LANs (IEEE 802.11 WLAN, WiFi,
- Security in wireless PANs (Bluetooth and IEEE 802.
- Security specification techniques
- Tradeoff analysis between performance and security
- Viruses worms and other malicious code
- Multicast security
- Network forensics
- Non-repudiation
- Prevention of traffic analysis
- Computer forensics
- Risk assessment and management
- Secure PHY/MAC/routing protocols
- Security group communications
- Security in cellular networks (2G, 2.5G, 3G, B3G,
- Security in content-delivery networks
- Security in domain name service
- Security in high-speed networks
- Security in integrated wireless networks
- Security in IP networks
- Security in optical systems and networks
- Security in satellite networks
- Security in VoIP
- Security in Wired Networks
- Security in wireless internet
- Security in wireless MANs (IEEE 802.16 and WiMAX)
- Security policies
- Security standards
- Trust establishment
- WLAN and Bluetooth security

CALL FOR PAPERS

Volume: 6 - **Issue:** 5 - October 2012

i. Paper Submission: July 31, 2012 **ii. Author Notification:** September 15, 2012

iii. Issue Publication: October 2012

CONTACT INFORMATION

Computer Science Journals Sdn BhD
B-5-8 Plaza Mont Kiara, Mont Kiara
50480, Kuala Lumpur, MALAYSIA
Phone: 006 03 6207 1607
006 03 2782 6991

Fax: 006 03 6207 1697

Email: cscpress@cscjournals.org

CSC PUBLISHERS © 2012
COMPUTER SCIENCE JOURNALS SDN BHD
M-3-19, PLAZA DAMAS
SRI HARTAMAS
50480, KUALA LUMPUR
MALAYSIA

PHONE: 006 03 6207 1607
006 03 2782 6991

FAX: 006 03 6207 1697
EMAIL: cscpress@cscjournals.org