INFORMATION
Volume 16, Number 3(B), pp.2289-2294
ISSN 1343-4500
©2013 International Information Institute

The Cross Crypto Scheme Cipher Integration for Securing SCADA Component Communication

MinKyu Choi *, Rosslin John Robles **, Goreti Marreiros*, Zita Vale*, Carlos Ramos* and Hoon Ko[1]*

*GECAD – Knowledge Engineering and Decision Support Group
Institute of Engineering – Polytechnic of Porto, Portugal
E-mail: {minky,zav,csr,hko,mgt}@isep.ipp.pt

**Information Technology Department, College of Arts and Sciences
University of San Agustin, Gen. Luna Street, Iloilo City 5000 Philippines
E-mail: rosslin_john@yahoo.com

Abstract

Critical Infrastructures became more vulnerable to attacks from adversaries as SCADA systems become connected to the Internet. The open standards for SCADA Communications make it very easy for attackers to gain in-depth knowledge about the working and operations of SCADA networks. A number of Internet SCADA security issues were raised that have compromised the authenticity, confidentiality, integrity and non-repudiation of information transfer between SCADA Components. This paper presents an integration of the Cross Crypto Scheme Cipher to secure communications for SCADA components. The proposed scheme integrates both the best features of symmetric and asymmetric encryption techniques. It also utilizes the MD5 hashing algorithm to ensure the integrity of information being transmitted.

Key Words: SCADA Security, SCADA Component Communication, Cross Crypto Cipher Encryption Scheme

1. Introduction

Supervisory Control and Data Acquisition (SCADA) refers to a large scale control system for automated industrial processes like power generation, water supplies, manufacturing industries, gas and oil pipelines, etc. [9].

The connection of the SCADA systems to the Internet has made critical infrastructures more vulnerable to attacks from adversaries. The internet SCADA facility has brought a lot of advantages in terms of control, data viewing and generation. Aside from the introduction of Internet SCADA, operators have implemented wireless connections for SCADA systems that can save and cut operational costs for communication lines [8].

This paper presents an integration of the Cross Crypto Scheme Cipher to secure communications for SCADA components. The crossed crypto cipher scheme was implemented combining the best features of both symmetric and asymmetric cryptography. To ensure integrity of the data, the MD5 hash algorithm was adopted and integrated to the technique.

2. The Cross Crypto Scheme Cipher

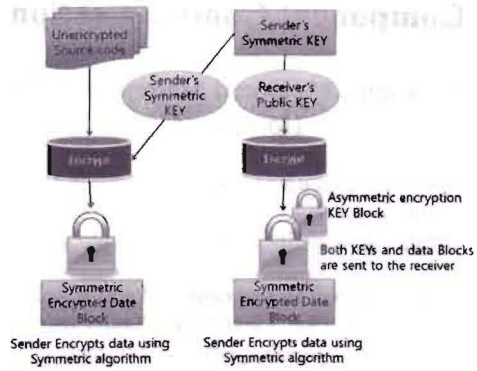

Fig. 2. The crossed-cipher Algorithm.

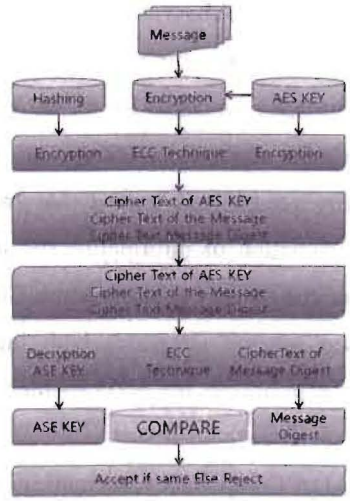

Fig. 3. Cross Crypto Scheme Chain of Operation.

The combination of the best features of both symmetric and asymmetric encryption techniques is presented in the form of a crossed-cipher for SCADA system. This crossed-cipher is capable of providing implicit authentication for the sender's identity. From the two major types of encryptions, asymmetric encryption provides more functionality than symmetric encryption, at the expense of speed and hardware cost. On the other hand symmetric encryption provides cost-effective and efficient methods of securing data without compromising security and should be considered as the correct and most appropriate security solution for many applications. In some instances, the best possible solution may be the complementary use of both symmetric and asymmetric encryption. The plain text data is to be transmitted in encrypted using the AES algorithm [4]. Where in it will generate a random secret key for a symmetric cipher (AES), and then encrypt this key via an asymmetric cipher (ECC) [5] using the recipient's public key. The message itself is then encrypted using the symmetric cipher and the secret key. Both the encrypted secret key and the encrypted message are then sent to the recipient. This algorithm is shown in Figure 2.

Figure 3 depicts the chain of operation in the proposed cipher scheme. The AES key which is used to encrypt the data is encrypted using ECC. The cipher text of the message and the cipher text of the key are then sent to the receiver. To ensure integrity of the data that is transmitted, the data is subjected to MD5 hash algorithm [2]. The message digest obtained by this process is also encrypted using ECC technique [5]. Thus the sender sends (1) Cipher text of the message, (2) Ciphertext of the AES key, and (3) Ciphertext of the message digest. The receiver upon receiving the Cipher text of the message, Ciphertext of the AES key, and Ciphertext of the message digest, first decrypts the Ciphertext of the AES key to yield the AES key. This is then used to decrypt the cipher text of the message to yield the plain text. The plaintext is again subjected to MD5 hash algorithm. This process yields a message digest. The ciphertext of the message digest is decrypted using ECC technique to obtain the message digest sent by the sender. This value is compared with the computed message digest. If both of them are equal, the message is accepted else rejected. Figure 4 shows the simple application developed based on the chain of operation.

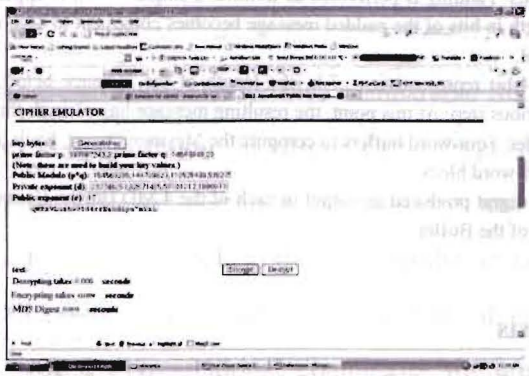

Fig. 4. Cipher Emulator.

Steps in AES:

The algorithm consists of four stages that make up a round, which is iterated 10 times for a 128-bit length key, 12 times for a 192-bit length key and 14 times for a 256- bit length key.

Step 1: "Sub Bytes" transformation is a non-linear for each byte of the block.

Step 2: "Shift Rows" transformation cyclically shifts (permutes) the bytes within the block.

Step 3: "Mix Columns" transformation groups 4-bytes together forming 4-term polynomials and multiplies the polynomials with a fixed polynomial mod(x^4+1).

Step 4: "Add Round Key" transformation adds the round key with the block of data.

Steps in Finding Base:

Step 1: Take the Elliptic curve y2 mod p= x3 + ax +b mod p where p is a prime number.

Step 2: For values from 0 to p-1, compute LHS and RHS

Step 3: Locate points P where LHS = RHS.

Step 4: Count the number of points 'n'. The total number of points is always n + 1 (one point at infinity).

Step 5: Find the prime factors of (n+1) and choose the largest among them.

Step 6: Find the negative point for every point computed in step 3.

Step 7: Now perform addition operation of the each of the points obtained in step 3.

Step 8: Repeat step 7 until one gets the point at infinity.

Step 9: Identify the largest prime factor from step 5. From the table created in step 7, locate for what points of P, the value is O (point at infinity).

Step 10: From the list of points, one can choose any point which will be the base point.

Steps in Key Generation:

Step 1: Sender and receiver agree on the elliptic curve E and the base point G with order n. The order of n must be large. Hence E,G and n are known to everyone.

Step 2: Sender chooses a random number ds which is $1 < ds < n-1$. He then computes ds * G. For him, ds is the private key and ds * G is the public key.

Step 3: Receiver chooses a random number dr which is $1 < dr < n-1$. He then computes dr*G. For him, dr is the private key and dr * G is

Steps in Encryption:

Step 1: Both sender and receiver agree on the elliptic curve E and the base point G with order n. Hence E,G and n are known to everyone.

Step 2: Sender then encodes the message M as a point.

Step 3: Sender then generates a random number k. He then computes the value of kG (again a point).

Step 4: Sender takes the public key (drG) of the receiver, multiples the same with k (result is a point), and adds that with M. The result is again a point. Sender sends { kG, M + kdrG } to the receiver.

Steps in Decryption:

Receiver gets { kG, M + kdrG } sent by sender

Step1: Extracts kG portion.

Step 2: Multiples the same with his private key dr. He obtains kGdr.

Step 3: He then extracts M + kdrG portion. Subtracts the output of Step 2. i.e. M + kdrG - kGdr. which results in M, the plain text.

Steps in MD5:

Input: b-bit message

Step 1: Appending padding bits: Padding is performed as follows: a single "1" bit is appended to the message, and then "0" bits are appended so that the length in bits of the padded message becomes congruent to 448, modulo 512. In all, at least one bit and at most 512 bits are appended.

Step 2: Append Length: A 64-bit representation of b (the length of the message before the padding bits were added) is appended to the result of the previous step. At this point, the resulting message has a length that is an exact multiple of 512-bits.

Step 3: Initialize the MD Buffer: Four word buffers to compute the Message Digest. Each one is a 32-bit register.

Step 4: Process Message in 16-word block

Step 5: Output: The message digest produced as output in each of the 4 MD Buffer. Begin with low order byte of the Buffer and end with the high order byte of the Buffer.

3. Results and Analysis

The following table 1 depicts information on encryption & decryption of 128 bit AES key and MD5 message digest using ECC. It provides for details on the time taken for encryption, decryption, and calculation of MD5 message digest process calculation.

Table 1. AES encryption and decryption and calculation of MD5 message digest

SIZE (KB)	25Kb	50Kb	100Kb
128 bit AES Encryption	0.951	1.126	2.38
128 bit AES Decryption	0.948	1.121	2.217
MD5 Message Digest	0.683	0.752	1.081

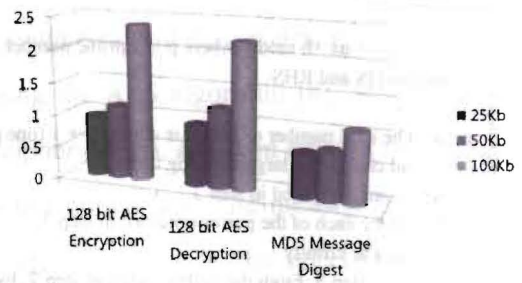

Fig. 5. Graphical Analysis of AES and MD5.

Figure 5 shows the comparison of the time taken for performing the 128-bit encryption, 128-bit decryption and MD5 message digest using AES algorithm applied to different data sizes of 25, 50 and 100 Kb.

Table 2 and Figure 6 depicts the encryption and decryption of AES key and MD5 message digest using the Elliptic Curve Cryptography or ECC.

Table 2. Encryption & Decryption of AES key and MD5 message digest using ECC

	128 bit AES KEY	**MD5 Hashing or Ciphertext**
Encryption	32	33
Decryption	36	38

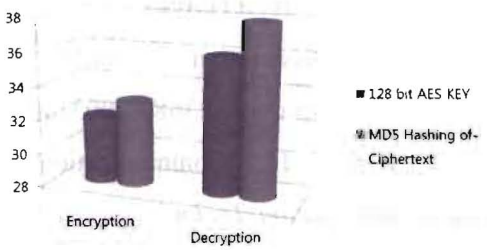

Fig. 6. Graphical analysis of Encryption & Decryption of AES key and MD5 message digest using ECC.

Since the ECC key sizes are so much shorter than comparable RSA keys, the length of the public key and private key is much shorter in elliptic curve cryptosystems. This results into faster processing times, and lower demands on memory and bandwidth. With any cryptographic system dealing with 128 bit key, the total number of combination is 2128. The time required to check all possible combinations at the rate of 50 billion keys/second is approximately 5 x 1021 years. Computational complexity for breaking the elliptic-curve cryptosystem for an elliptic curve key size of 150 bits is 3.8 x 1010 MIPS (Million Instructions Per Second years) [3]. The challenging and somewhat complicated nature of elliptic curve groups makes it harder to crack the ECC discrete logarithm problem. With less bits required to give the same security, ECC has fared favorably compared to RSA.

4. Conclusions

The connection of the SCADA systems to the Internet has made critical infrastructures more vulnerable to attacks from adversaries. This paper has proposed the integration of the Cross Crypto Scheme Cipher to secure communications for SCADA components. The crossed crypto cipher scheme was implemented combining the best features of both symmetric and asymmetric cryptography. To ensure integrity of the data, the MD5 hash algorithm was adopted and integrated to the technique. The implementation of a crossed

crypto cipher scheme is one way to retrofit on to the system and be able to address the Confidentiality, Authenticity, Integrity, and Non-repudiation issues in SCADA systems.

5. Acknowledgments

"This work is supported by FEDER Funds through the "Programa Operacional Factores de Competitividade - COMPETE" program and by National Funds through FCT "Fundação para a Ciência e a Tecnologia" under the project: FCOMP-01-0124-FEDER-PEst-OE/EEI/UI0760/2011.

References

[1] Washington Post, "Dissertation could be security threat", http://www.washingtonpost.com/ac2/wp-dyn/A23689-2003Jul7, (2007).

[2] Robles R. J., Seo K. T., Kim T. H., "Communication Security solution for internet SCADA", *Korean Institute of Information Technology 2010 IT Convergence Technology - Summer workshops and Conference Proceedings*, (2010) May, pp.461-463.

[3] Stallings W., Cryptography and Network Security. *Prentice Hall*, Upper Saddle River, New Jersey, USA, second edition, (1999).

[4] Shay Gueron, "Intel® Advanced Encryption Standard (AES) New Instructions Set", *Intel Corporation*, (2010) May.

[5] Aydos M., Sunar B, and Ko□ □. K., "An Elliptic Curve Cryptography based Authentication and Key Agreement Protocol for Wireless Communication", *2nd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Dallas, Texas, (1998) October 30.

[6] Abu Al-Saud K., Tahir H., Saleh M. and Saleh M., "A Performance Comparison of MD5 Authenticated Routing Traffic with EIGRP, RIPv2, and OSPF", *The International Arab Journal of Informatiòn Technology*, vol.7, no.4, (2010) October.

[7] http://www.tscm.com/outsideplant.html

[8] Robles R. J. and Choi M. K., "Symmetric-Key Encryption for Wireless Internet SCADA", Communications in Computer and Information Science, Vol.58, *Security Technology*, (2009), pp. 289-297, ISSN: 1865-0929.

[9] Hentea M., "Improving Security for SCADA Control Systems", *Interdisciplinary Journal of Information, Know ledge, and Management*, vol.3, (2008).

*Corresponding author: Hoon Ko

GECAD – Knowledge Engineering and Decision Support Group
Institute of Engineering – Polytechnic of Porto, Portugal
E-mail: hko@isep.ipp.pt