

A Study of Security Architecture for Control Networks over IP

Nobuo Okabe Shoichi Sakane Kazunori Miyazawa
Ubiquitous Lab, Yokogawa Electric Corporation,
2-9-32 Nakacho, Musashino, Tokyo 180-8750, Japan
{Nobuo.Okabe, Shouichi.Sakane, Kazunori.Miyazawa}@jp.yokogawa.com

Atsushi Inoue Masahiro Ishiyama
Corporate R&D Center, Toshiba Corporation,
1 Komukai Toshiba-cho, Saiwai-ku, Kawasaki-shi 212-8582, Japan
{inoue, masahiro}@isl.rdc.toshiba.co.jp

Kenichi Kamada
Graduate School of Information Science and Technology, The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8654, Japan
kamada@hongo.wide.ad.jp

Abstract

There are many kinds of control networks which have been used in various non-IP network areas, such as in buildings, plants and vehicles. These do not incorporate reasonable security mechanisms as they have been mainly used for closed networks. Recently the security of control networks is becoming important because of the popularization of the Internet, the deployment of wireless technologies and the security requirements of such infrastructures. One of the important issues is that the small embedded devices commonly used in control networks for security mechanisms might become overloaded because of their performance limitations. This paper shows security mechanisms which can suit small devices in control networks, assuming that IP is applied to the control networks.

Keywords: Control Network, Embedded System, Security, IPsec, KINK

1 Introduction

Control networks are different from IP (Internet Protocol) with regard to their history, purposes and technologies. Control networks do not have reasonable security mechanisms because they have been used for closed networks such as buildings, plants or vehicles. However, security is becoming more important for control networks because a) many non-IP networks have to be connected to the Internet, b) wireless technologies expose closed networks to traffic. One of the important issues is that the small embedded devices commonly used in control networks for security mechanisms might become overloaded. This paper shows security mechanisms which are suitable for small devices in the control networks, assuming that IP is applied to the control networks. This paper evaluates popular cryptographic

algorithms in Section 2, cryptographic hardware in Section 3, IPsec performance in Section 4, IPsec's key exchange protocols in Section 5, and proposes a security system in Section 6.

2 Performance of Cryptography

The computational cost of cryptography must be reasonable for low-end CPUs, i.e. 8-bit or 16-bit CPUs, which are common among control networks. In this section we estimate the computational cost of RSA (the RSA algorithm) which is usually used for authentication, DH (the Diffie-Hellman key exchange) which is usually used for generating session keys, and symmetric key cryptography and hash functions which are usually used to provide data with privacy and integrity for

protecting communications.

First, Figures 1 and 2 show the encrypting/decrypting time of RSA run on a 16-bit CPU, i.e. H8/3048 [1] which is a popular low-end CPU in Japan [2]. This program was ported from GnuPG 1.0.7 [3] to the CPU. The results do not include the padding specified by PKCS#1 [4] because our purpose is to show the essential computational cost of RSA. Therefore, encrypting time is relative to data size, whereas decrypting time is constant. Both figures show that RSA is not suited to low-end CPUs because it takes several minutes to decrypt a secret.

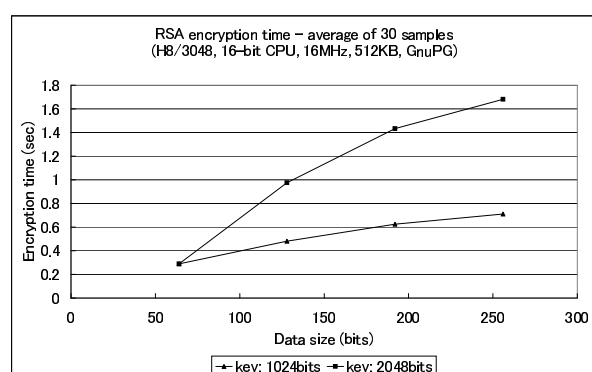


Figure 1: Encrypting performance of RSA

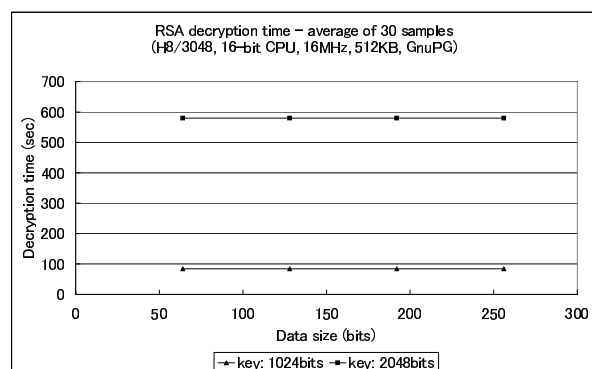


Figure 2: Decrypting performance of RSA

Second, Figure 3 shows the performance of DH run on H8/3048. This estimation uses MODP (modular exponentiation) groups, which is deemed mandatory by IKE (the Internet Key Exchange) [11], rather than elliptic curve groups which is not mandatory for IKE, because we are assuming that IKE may be introduced to the security system. The processing time in the figure is the sum of the generating time of a DH pair (a se-

cret value and a public value) and the generating time of a DH shared value for each bit length of prime numbers. However, it excludes the generating time of the prime number and the primitive root number. This program is an original code based upon arithmetic libraries in GnuPG 1.0.7 [5]. The figures show that DH is not suited to low-end CPUs because it takes several minutes for DH processing.

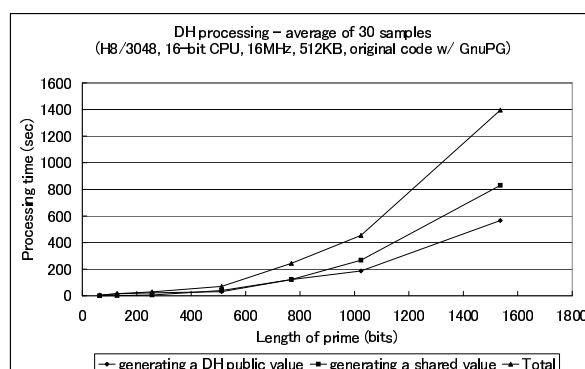


Figure 3: Performance of DH exchanges

Third, Figure 4 shows the performance of symmetric key cryptography (AES-238 CBC and 3DES CBC) run upon a 8051-compatible 8-bit CPU, i.e. DS80C390 [6]. The 8051 architecture is also popular low-end CPU. Figure 5 shows the performance of hash functions (MD5 and SHA1). These programs are original code written in assembler. Figure 4 shows that it takes several seconds to encrypt/decrypt 1000 bytes of data. Figure 5 shows that it takes about one second to generate a hashed value from 1000 bytes of data. The figures show that low-end CPUs cannot achieve reasonable throughput if applying symmetric key cryptography and a hash function to protect communication.

3 Cryptographic Hardware

Section 2 showed that cryptographic means with computational capabilities that are very limited do not suit low-end CPUs. One solution is to introduce cryptographic hardware. In this section we estimate the gate sizes of cryptographic hardware, i.e. RSA, DH, symmetric key cryptography and hash functions.

Symmetric key cryptography and hash functions needs up to ten thousand the gates if optimizing for area [7, 8]. RSA and the MODP group of DH use the MODP function $g^x \text{ mod } p$, where g is a primitive root

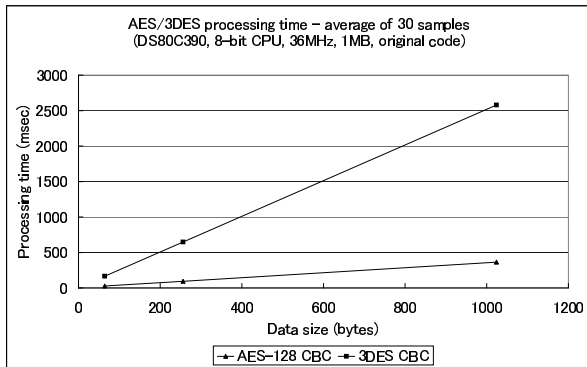


Figure 4: Performance of symmetric key cryptography

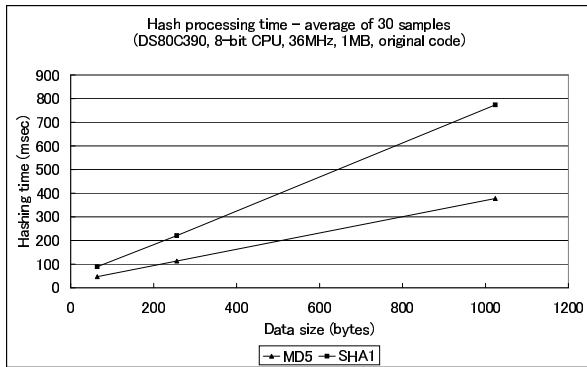


Figure 5: Performance of hash functions

number, p is a prime number and x is a random number. The MODP function for a 1024-bit prime number needs more than one hundred thousands gates [9].

Symmetric key cryptography and hash functions are necessary to maintain the privacy and integrity of packets, even if RSA or DH is introduced. This is common for recent security protocols, e.g. SSL, IPsec with IKE. In other words, it is inevitable to implement hardware of symmetric key cryptography and hash functions when introducing hardware of RSA or DH for low-end systems. From an economic point of view, we anticipate that a security system which uses only symmetric key cryptography and hash functions will be more scalable than the one using public cryptography, symmetric key cryptography and hash functions because smaller gate size is better for low-end systems.

4 Performance of IPsec

IPsec provides IP packets with privacy and integrity for protecting communications, and is useful because its enforcement is independent from applications. In this section we estimate the performance of IPsec written in assembler, which we implemented on a DS80C390. Figure 6 shows the outbound performance of IPsec ESP [10] whose cryptographic programs are the same ones used for Figures 4 and 5. Inbound performance is omitted because both performances are almost same.

The processing time of IPsec (Figure 6) is almost the sum of the processing time of symmetric key cryptography (Figure 4) and hash functions (Figure 5). This means that the greatest part of the processing time is consumed by cryptography and hash functions. Therefore, any security mechanism like IPsec faces the same performance degradation, and cryptographic hardware can improve the performance.

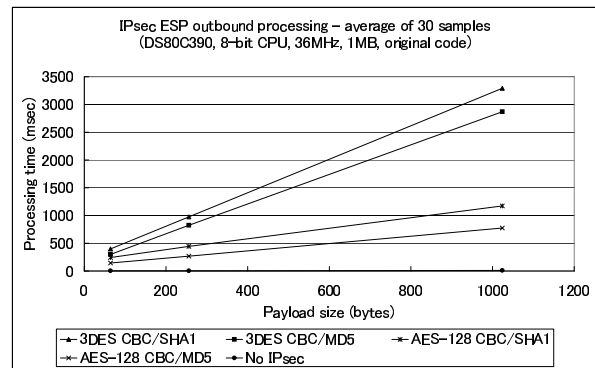


Figure 6: Outbound performance of IPsec ESP

5 A Choice of the Key Exchange Protocols

It is important for IPsec to share a secret between both ends. Key exchange protocols will be important in facilitating the sharing of a secret if running IPsec on small embedded devices because these devices do not have a powerful user interface like a PC, which makes manual keying difficult. IKE, which has been standardized in IETF IPSEC WG, is the most popular key exchange protocol for IPsec. However, IKE does not suit the small embedded devices described in Section 2. There is another key exchange protocol for IPsec, i. e. KINK (Kerberosized Internet Negotiation of Keys) [12], which

is in a standardizing process of IETF. We expect that KINK can work well on small embedded devices because KINK is based upon Kerberos¹ [13], which uses symmetric key cryptography and hash functions only.

6 Proposed System

Figure 7 is our proposal for a security system which is suitable for the control networks. Device-to-device communication on IP is protected by IPsec whose key exchange protocol is KINK. The Kerberos server gives KINK-specific tickets to the devices for mutual authentication. Each device in the system has symmetric key cryptography and hash functions which can be optionally implemented by hardware if the device's computational capabilities are inadequate. This system requires only symmetric key cryptography and hash functions, and does not need public key cryptography, e.g. RSA or DH, which are known for their highly computational consumption.

We are currently developing a prototype system to estimate its validity.

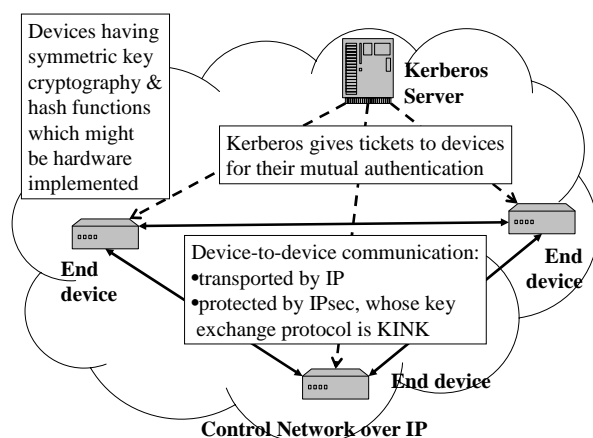


Figure 7: Proposed System

7 Summary

This paper shows an IP based security system which are suitable for low-end CPUs or small embedded devices. We are developing a prototype system to verify our ideas.

¹Kerberos is a trademark of the Massachusetts Institute of Technology (MIT).

The one of important study items is the performance evaluation of real systems because competition between IPsec and KINK for cryptographic hardware can stall the system's throughput. That will be done on the prototype system which we are developing.

References

- [1] 16-bit micro-controller H80/3048, Hitachi, Ltd., <http://www.renesas.com/eng/products/mpumcu/16bit/h8300h/3048/>
- [2] "Survey Results on Real-time OS Use in Embedded Systems (2002)", TRON Association, <http://www.assoc.tron.org/jpn/research/data/survey2002J.pdf>
- [3] GnuPG: a RFC2440 (OpenPGP) compliant application, <http://www.gnupg.org/>
- [4] B. Kaliski, "PKCS #1: RSA Encryption Version 1.5", RFC2313, 1998.
- [5] GnuMP: A free library for arbitrary precision arithmetic, <http://www.swox.com/gmp/>
- [6] 8-bit micro-controller DS80C390, Maxim/Dallas Semiconductor, <http://pdfserv.maximic.com/en/ds/DS80C390.pdf>
- [7] The summary of encryption cores, Ocean Logic Pty Ltd., <http://www.ocean-logic.com/des.htm>
- [8] H. Shimizu, F. Sano, et al. "Implementation of SPN block cipher" ISEC 2001-55, Technical report of IEICE, 2001.
- [9] CS-9010 IP core, chipSign A/S, <http://www.chipsign.com/downloads/PBF-9010-01-02.pdf>
- [10] S. Kent, R. Atkinson, "Security Architecture for the Internet Protocol", RFC2401, 1998.
- [11] D. Harkins, D. Carrel, "The Internet Key Exchange (IKE)", RFC2409, 1998.
- [12] M. Thomas, J. Vilhuber, "Kerberized Internet Negotiation of Keys (KINK)", draft-ietf-kink-kink-05.txt, 2003.
- [13] J. Kohl, C. Neuman, "The Kerberos Network Authentication Service (V5)", RFC1510, 1993.