International Journal of Security and Its Applications
Vol.8, No.2 (2014), pp.243-250
http://dx.doi.org/10.14257/ijsia.2014.8.2.25

Implementation of ARIA Cryptographic Modules based on ARM9 Devices

Okyeon Yi[1] , Seunghwan Yun[2], Myungseo Park[3] and HaengGwon Song[4]

[1]*Dept. of Mathematics, Kookmin University, Seoul, Korea*
{oyyi[1], pms91[3], haeng9769[4]}@kookmin.ac.kr, cryptoschool[2]@gmail.com

Abstract

Information over Internet needs to be communicated secure and prompt. But every cryptographic algorithm of the information requires extra time and memory space for execution in the systems such as PC or smart device. In this paper, an actual cost of speed and memory for a popular smart device environment such as ARM9 with Linux is presented so that the performance is enough to serve wired or wireless data transmission from DCU for smart grid systems or Full HD CCTV cameras.

Keywords: *Cryptographic Algorithm, ARIA, ARM9, Cryptographic Module Validation Program*

1. Introduction

Data communication is vulnerable to eavesdropping, forgery and modification in common internet environment over wired or wireless communication. The C-ITS(Cooperative Intelligent Traffic System), Smart Grid and ITS(Industrial Control System) use wired or wireless data transmission with a high speed. Most data communication standards such as 3GPP and IEEE802 series require lots of cryptographic algorithms for information security services.

The DCU(Data Concentration Unit) system which is one of the most important component units of AMI(Advanced Metering Infrastructure) for the smart grid services is normally implemented with ARM processor and Linux. So the security services such as authentication among FEP(Front End Processor) server, DCU, and Meters, encryption/decryption of the transmitted data, message authentication of the data must be implemented on the DCU and Meters. But the ARM9 environment is implemented to perform DCU's original functions without any cryptographic services with low computing power and memory.

So a performance result of running time and memory size to execute cryptographic algorithms is definitely needed to measure the difficulty of information security services on the restricted environment such as DCU or CCTV systems.

Block ciphers, Public key algorithms, hash functions, random number generators, digital signature algorithms provide confidentiality, integrity and authentication services for the transmitting data. But those algorithms require extra resources of time and space in the devices. Those extra resources could generate time delay so that the QoS(quality of service) might become dangerous. Hence correct estimations for the extra resource to perform cryptographic algorithms in a specific device would be critical factors

Cryptographic algorithms can be implemented as software libraries, firmware or dedicated hardware. But adding extra hardware modules for cryptographic services to original device brings out unexpected problems so that most cryptographic algorithms are implemented as software libraries or firmware type.

ISSN: 1738-9976 IJSIA

In this paper, software types of cryptographic algorithm over x86 and ARM architecture are suggested and the performance results in the devices are presented.

The results of implementation of the block cipher ARIA with ECB(Electronic Code Book) mode, CBC(Cipher Block Chaining) mode, CTR(Counter) mode, and CCM(Counter and CBC Message Authentication Code) mode are presented. And the comparison of 3 types of operations such as SO(Shared Object) type file call from application and KO(Kernel Object) type file call from applications, and KO type file call from KO file in the x86 and ARM devices is provided in this paper.

In Chapter 2, the modes of operation of block ciphers and special comparison items of x86 and ARM architecture is presented. And experiment method and results are given in the Chapter 3 and the conclusion is given in the Chapter 4.

2. Cryptographic Backgrounds

2.1. The Modes of Operation of Block Ciphers

The block cipher has its own length of input block other than the length of key. The AES and ARIA both have 128bit as input length with 128 bit output size. Hence to encrypt longer size input, the modes of operation techniques are needed.

ECB mode in Figure 1 uses 128 bit input and 128 bit output and no interaction in different blocks. Hence short messages can be encrypted by using the ECB mode.

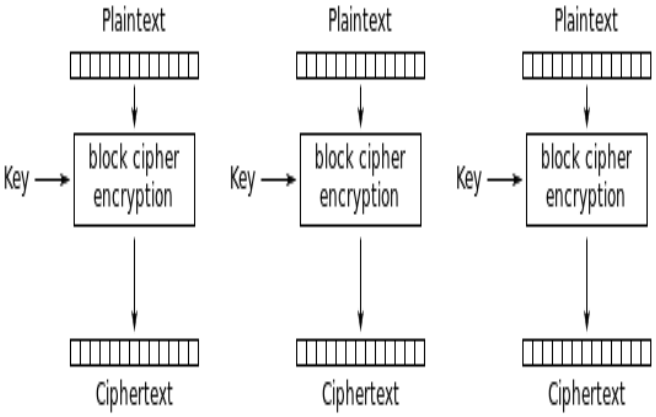


Figure 1. ECB Mode of Operation

CBC mode in Figure 2 cuts input blocks into 128 bit blocks and each input block is XORed with previous cipher block(or with Initial Vector), then the core block cipher encrypts the XORed block.

CTR mode in Figure 3 encrypts a counter value as an input, then XOR the output with a plaintext block to make a ciphertext corresponding the plaintext block. The counter values must be different from each block of plaintexts.

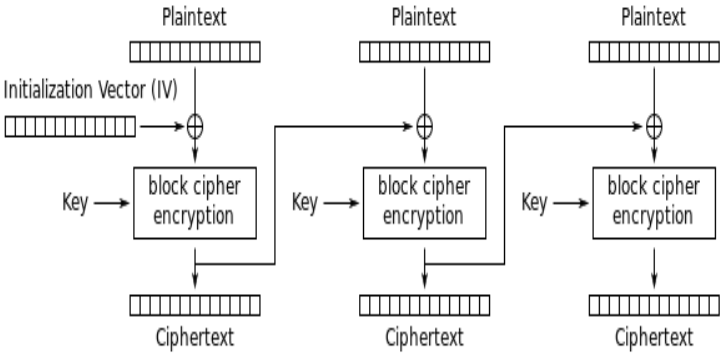


Figure 2. CBC Mode of Operation

CCM(Counter with CBC-MAC) mode is a combination of the CBC-MAC(Cipher Block Chaining-Message Authentication Code) mode with the CTR mode to support the confidentiality, message integrity, and the origination of the message that is whose plaintext it is. CCM mode is widely used at IEEE 802.11(WLAN), IEEE 802.15.3(UWB), and IEEE 802.15.4(ZigBee) to support the security services over the wireless intervals.

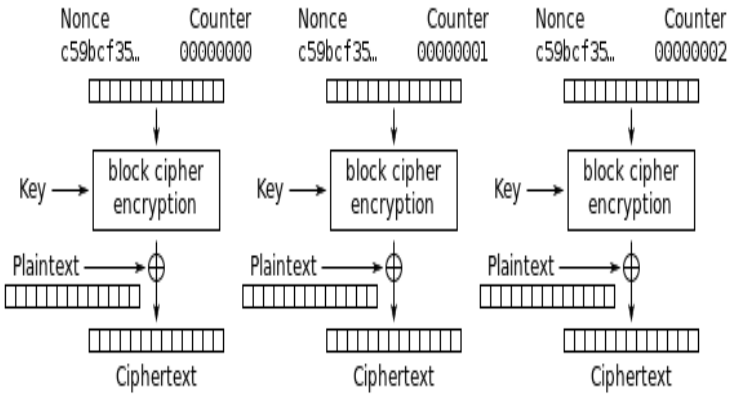


Figure 3. CTR Mode of Operation

2.2. The Architectures of x86 vs. ARM Devices

The smart grid systems and devices require low power, cheaper cost, and low heat processors. The ARM devices are good solution for small systems for example DCU(Data Concentration Unit) or CCTV. One of major difference is the type of CPUs. The ARM(Advanced RISC Machine) is the RISC(Reduced Instruction Set Computer) type. But x86 is the CISC(Complex Instruction Set Computer).

The RISC type processor has fewer commands and simpler circuits, but faster executions and lower components per chip. The main concept of RISC is that of a system uses a small, highly-optimized set of instructions rather than a specialized set of instructions.

The CISC type processors are operated by complex commands so that easy SW implementation is possible, but require high components per chip. It is a computer single

instructions can be executed several low-level operations such as loading from memory, arithmetic operations, and memory store. And it is capable of multi-step operations within single instructions.

In either case, the main principle is to improve the performance. Several papers show that the differences of process type bring the difference of performance of cryptographic modules [4, 5].

2.3. CMVP and ARIA Cryptographic Algorithm

ARIA is a block cipher designed in 2003 by a group of Korean researchers. And it was selected as a standard cryptographic algorithm by the Korean Agency for Technology and Standards. The ARIA uses a substitution-permutation network structure based on AES. The interface is 128-bit block size with key size of 128, 192, or 256 bits. The number of rounds is 12, 14, or 16, depending on the key size. ARIA uses two 8x8 bit S-boxes and their inverses in alternate rounds [7].

Many countries such as Korea and Japan use different CMVP(Cryptographic Module Validation Program) from NIST's CMVP by FIPS 140-1 or FIPS 140-2 for their public securities. The most different item between NIST's CMVP and Korea's CMVP is the symmetric encryption algorithms ARIA and KCDSA(Korean Certificate-based Digital Signature Algorithm.

The CC(Common Criteria) evaluation does not supersede, substitute, or replace a validation to either FIPS 140-1 or FIPS 140-2.[1] The four security levels in Korean CMVP, FIPS 140-1 and FIPS 140-2 do not map directly to specific CC EALs or to CC requirements.

3. Experiments on x86 and ARM9 Devices

An experiment result is presented in this chapter from the implementations of KCMVP-approved cryptographic algorithms ARIA with ECB, CBC, CTR, CCM modes on x86 and ARM9 devices.

Table 1. Specification of Experiments

	ARM9	x86
Operating System	Linux AT91SAM9 2.6.39	Linux version 2.6.32-38
Processor	ARM926EJ-S rev 5	MIPS 74K V4.12
Memory	125940kB	10486784kB
Compiler	arm gcc version 4.3.5	gcc version 4.3.3

Exactly 1280-byte inputs are encrypted 100,000 times and the elapsed time in seconds are measured without the key scheduling since the modes of operation use the same key during a session period. The interface call processes are differed by 3 types, APP-LIB, APP-DRV, DRV-DRV on x86 and ARM architecture.

Obviously, x86 environment is better than ARM9 environment, so the performance results clearly show facts. But the main point of this paper is the comparison of the 3 call processes in x86 and ARM9 environments.

The Table 2 shows the results of the comparisons in each experiment environment.

Table 2. Performance on each Environment (Encrypted 100,000 times, in sec.)

	Linux x86			Linux Arm9		
	APP-LIB	APP-DRV	DRV-DRV	APP-LIB	APP-DRV	DRV-DRV
ECB	2.5489	3.5919	1.208	93.8757	44.3294	44.5
CBC	2.5506	3.503	1.244	108.5654	45.3034	45.03
CTR	2.5002	4.0886	1.212	97.0579	45.8638	45.56
CCM	4.9687	6.2194	2.416	210.1792	92.1122	91.59

In Figure 4, DRV-DRV interface call time is 2 times faster than APP-LIB interface call in x86 environment. And DRV-DRV interface call time is 3 times faster than APP-DRV interface call. And obviously APP-DRV interface call time is the longest among the interfaces in x86 environment.

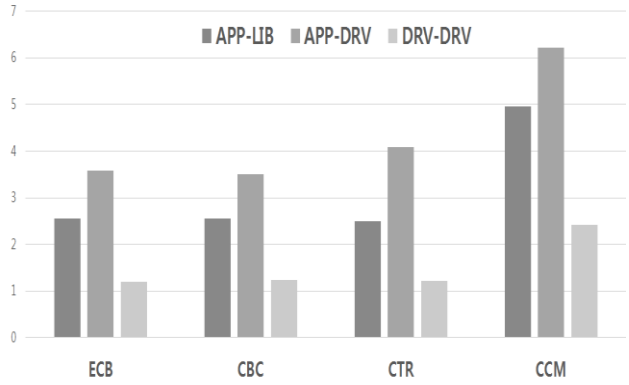


Figure 4. Three Different Interface Calls on x86 System

But in Figure 5, we can notice that APP-DRV and DRV-DRV interface calls take almost the same performance but APP-LIB interface call takes 2 times longer than the other ones in ARM9 environment.

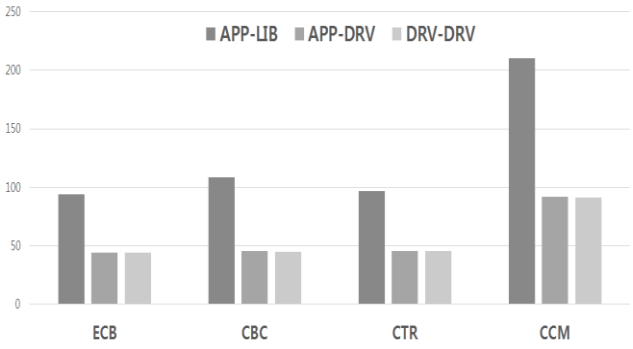


Figure 5. Three Different Interface Calls on ARM9 System

So when we implement the cryptographic modules ARIA with ECB, CBC, CTR, or CCM in ARM9 environment, KO(Kernel Object) type implementation improves 2 times faster performance than SO(Shared Object). This experiments used Korean CMVP approved ARIA module tested on the ARM9 devices. Since around 98% of Korean CMVP approved modules are implemented as SO type on x86 PC or server environment, the results of this experiment shows the different result over the ARM devices.

4. Conclusions

This paper presented experiment results of the difference among the interface calls APP-LIB, APP-DRV, and DRV-DRV on ARM9 environment with Linux.

To provide security services, most systems implement the cryptographic algorithms or cryptographic modules via additional software types such as SO(Shared Object) file type or KO(Kernel Object) type. And those security services are also implemented small smart devices with low-power and low-memory consumption.

The experiment result show that in the ARM9 environment, APP-DRV and DRV-DRV have almost the same performance and 2 times faster than the APP-LIB type. But the results show a big difference from x86 environment in which APP-DRV type is 3-times slower than DRV-DRV, and even 2 times slower than APP-LIB type.

So this paper concludes that the KO type implementation of cryptographic algorithms or cryptographic modules with APP interface call or DRV interface call is the best choice on the ARM9 environment for the security of smart grid or CCTV services.

Acknowledgements

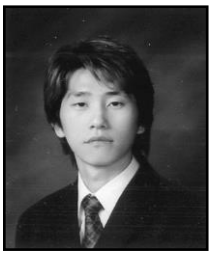
This work was supported by 2013 research program of Kookmin University in Korea.

References

[1] Cryptographic Module Validation Program, http://csrc.nist.gov/cmvp.
[2] KMULiB v1.0 Validation Module list, http://service1.nis.go.kr/.
[3] U. Degenbaev, “Formalization of parts of the x86-64 Instruction Set Architecture”, Master’s thesis, Saarland University, Germany, **(2007)**.
[4] P. R. Lakhe, “A Technology in Most Recent Processor is Complex Reduced Instruction Set Computers (CRISC): A Survey”, International Journal of Innovation Research and Studies, vol. 2, no. 6, **(2013)** June, pp. 711-715.
[5] S. P. Dandamudi, “Guide to RISC processors”, ISBN 0-387-21017-2, **(2005)**, pp. 121-123.
[6] Z. Ruimei and W. Mei, “Design of ARM-based embedded Ethernet interface”, vol. 4, **(2010)** April, pp. 268-270.
[7] The Law to mandatory the usage of KCMVP Modules to IT systems for Public Services http://www.law.go.kr/lsInfoP.do?lsiSeq=141803&efYd=20130706#0000.
[8] Information technology-Security techniques-Security requirements for cryptographic modules, ISO/IEC 19790, 201208.
[9] P. R. Lakhe, “A Technology in Most Recent Processor is Complex Reduced Instruction Set Computers (CRISC): A Survey”, ISSN 2319-9725.

Authors

Okyeon Yi received his Ph.D. degree in Mathematics from the University of Kentucky, Lexington, KY, USA in 1996. After working at the Electronics & Telecommunications Research Institute(ETRI) from 1999 to 2001, he joined the Kookmin University in 2001, where he is currently a Professor of the Information Security major, the Graduate School, Kookmin University. His interests are Information Security, Mobile Communication and Cryptographic Module Implementation.



Seunghwan Yun received his M.S. degree in Mathematics from Kookmin University, Korea, in 2008. He is currently undertaking an Ph. D. course in the Center for Information Security Institute from Korea University, Korea His current research interests are in the areas of implementation of cryptographic modules and information security.

Myungseo Park received his B.S. degree in Mathematics from Kookmin University, Korea, in 2013. He is currently undertaking an M.S. course in Mathematics from Kookmin University, Korea. His current research interests are in the areas of implementation of cryptographic modules and information security.

HaengGwon Song received his B.S. degree in Mathematics from Kookmin University, Korea, in 2013. He is currently undertaking an M.S. course in Mathematics from Kookmin University, Korea. His current research interests are in the areas of implementation of cryptographic modules and information security.