International Journal of Security and Its Applications
Vol.8, No.1 (2014), pp.273-282
http://dx.doi.org/10.14257/ijsia.2014.8.1.25

Performance of Converged Secure Socket Layer and CMVP Cryptographic Modules

Okyeon Yi[1], Seunghwan Yun[2], Myungseo Park[3], Nuri Hwang[4],
Taeyean Kwon[5] and Chaewon Yun[6]

[1]Dept. of Mathematics, Kookmin University, Seoul, Korea
{oyyi[1], pms91[3], yubkinuri[4]}@kookmin.ac.kr, [2]cryptoschool@gmail.com,
[5]kte4567@nate.com, [6]chwyun91@nate.com

Abstract

U.S. Nuclear Regulatory Commission's Regulatory Guide 5.71 requires CMVP (Cryptographic Module Validation Program)-approved cryptographic modules for Nuclear facilities. And NIST documents also require CMVP-approved modules for smart grid networks. In this paper, a convergence method of CMVP and SSL(Secure Socket Layer) for web services and performance results of the method are presented.

Keywords: *Cryptographic Module Validation Program, Secure Socket Layer, NIST FIPS 140-2, ARIA*

1. Introduction

Data communication is vulnerable to eavesdropping, forgery and modification in common internet environment. To guarantee the confidentiality and data integrity, cryptographic algorithms and cryptographic modules including symmetric key, public key, hash function, random number generator and digital signature are being used to accomplish the services.

Cryptographic modules basically contain cryptographic algorithms such as AES, ARIA, SEED, RSAES, SHA-256, ECDSA and DH. Also the cryptographic modules have to have extra functions for self-tests, EMI/EMC, *etc*.

There are lots of source codes and library files downloadable from the Internet for those cryptographic algorithms. But 2 big questions arise from this situation. First one is the validity of the open source codes. For example there are 2^128 possible plaintexts and 2^128 possible keys as inputs for AES(Advanced Encryption Standard) symmetric key algorithm. Such huge input and output space bring the impossibility of 100% validity of the cryptographic algorithms and cryptographic modules. So no one can tests all the input and output values of the source codes. It means that there is no way to test all the inputs and outputs for the cryptographic algorithms. Second one is the purity of the open codes since any trapdoor could be inserted in the cryptographic algorithm files downloadable from the Internet.

In 1995, NIST proposed CMVP (cryptographic module validation program) to test cryptographic modules developed for US government security systems. U.S. Nuclear Regulatory Commission's Regulatory Guide 5.71 requires CMVP-approved cryptographic modules for nuclear facilities. And NIST document NISTIR 7628 also requires CMVP-approved modules for smart grid networks. Another security evaluation term is CC(common criteria) for information security products. But CC evaluation does not supersede or replace a validation to either FIPS 140-1 or FIPS 140-2. The security levels in FIPS 140-1 and FIPS

ISSN: 1738-9976 IJSIA

140-2 do not map directly to specific CC EALs or to CC functional requirements. So the CC certificate cannot be a substitute for a FIPS 140-1 or FIPS 140-2 certificate.

1.1. SSL(Secure Socket Layer) Protocol and CMVP

The SSL is a popular common socket level protocol widely being used to protect the transactions of e-Commerce, VPN(Virtual Private Network) services, and most Web-based services. The Open project is a collaborative effort to develop a full-featured open source toolkit implementing the Secure Socket Layer(SSL v2/v3) and Transport Layer Security(TLS v1) protocols as a full-strength general purpose cryptography libraries. The OpenSSL crypto library implements a wide range of cryptographic algorithm used in various Internet standards. The functionality includes symmetric encryption algorithms such as AES, Camellia, SEED, BLOWFISH, CAST, DES, 3-DES, IDEA, RC2, RC4, and RC5, and public key cryptographic and key agreement algorithms such DSA, Diffie-Hellman and RSA, and message authentication code and hash functions such as HMAC, MD2, MD4, MD5, MDC2, RIPEMD-160, and SHA series.

As of December 2012, the OpenSSL is one of two open source programs to be involved with validation under the FIPS 140-2 computer security standard by the National Institute of Standards and Technology's Cryptographic Module Validation Program. The OpenSSL itself in not validated, but a component called the OpenSSL FIP Object Module, based on OpenSSL, was created to provide many of the same capabilities. A certificate was first awarded in January 2006 but revoked in July 2006 when questions were raised about the validated module's interaction with outside software. The certification was reinstated in February 2007.

The CMVP(Cryptographic Module Validation Program) is a joint American and Canadian security accreditation program for cryptographic modules. The program is available to any vendors who seek to have their IT products certified for use by the U.S. Government and regulated industries, such as financial and health-care institutions, that collect, store, transfer, share and disseminate sensitive but not classified information. The module certifications under the CMVP are performed in accordance with the requirements of FIPS 140-2.

Other countries such as Korea and Japan use different CMVP from NIST's CMVP for their public security. The most different item between NIST's CMVP and Korea's CMVP is the symmetric encryption algorithms ARIA, SEED and KCDSA(Korean Certificate-based Digital Signature Algorithm.[4] Those algorithms were developed by Korea, so the OpenSSL doesn't support ARIA and KCDSA. 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.

The KCDSA is a digital signature algorithm created by a team led by the Korean Information Security Agency. It is an ElGamal variant algorithm and is similar to the Digital Signature Algorithm. The KCDSA requires a collision-resistant cryptographic hash function that can produce a variable-sized output from 128 to 256 bits in 32-bit increments.

Since the VPN services based on OpenSSL functionalities require CMVP approved by Korean Government, an efficient way of combining two things should be proposed.

In this paper, a combination method of openssl-1.01e which is most resent version of OpenSSL and KMULiBv1.0 which is a Korea CMVP approved module by the National Cyber Security Center, Korea, and the performance results of the combination are presented [3].

1.2. Korean CMVP

U.S. Government approves the cryptographic modules which include cryptographic algorithms and external functions such as self-testing whether the modules has any trapdoor in the source code level. The confirmation of the cryptographic module's implemented validity is also tested [2].

Software library, firmware or hardware type for a given CPU, OS and compiler version can be tested as a CMVP module so that if anyone of those items is changed, the validation is invalidated. The software library testing rule defined by the National Cyber Security Center should follow the KS X ISO/IEC 19790 [4].

Developers of CMVP determine the security level of the module and submit to the testing laboratory. An important part of the testing of a cryptographic module is reviewing many cryptographic module documentations. These documents need to explain how the module is designed, how the module is built, how the cryptographic module works, how the module is installed and placed and how the module protects cryptographic data. The documents need to explain all of the cryptographic features available in the module [2].

A Security Policy public document contains detailed descriptions of what makes up the cryptographic module what it does, how it does it, and who can operate the module. The Security Policy is reviewed by the testing lab to verify that it meets the minimum formatting requirements and that the information detailed is accurate. When testing is completed and validated, the Security Policy is posted on the CMVP web site so parties interested in procuring the module can be read about it before they make any decisions [4].

Table 1. Cryptographic Algorithms for Korea CMVP

		Cryptographic Algorithms
Symmetric Key Algorithm		ARIA, SEED
Hash Functions		SHA-224, SHA-256 SHA-384, SHA-512
Message Authentication Code	Hash based	HMAC
	Block cipher based	GMAC, CMAC
Random Number Generator	Hash Function or HMAC based	Hash_DRBG HMAC_DRBG
	Block cipher based	CTR_DRBG
Key Agreement		DH, ECDH
Public Key Algorithm		RSAES
Digital Signature Algorithm		RSA_PSS, KCDSA ECDSA, EC-KCDSA

A Finite State Machine document which details how a module behaves is needed. The Finite State Machine document should start at power-up and move through all of the various states contained within the module such as initialization, self-test state, error state, maintenance state, operational state and finally the power-off state. The document must detail what make the module transition from state to state, what the module does within a state, what makes a module move into an error state and how the module gets out of an error state.

Source code documentation is also required to be reviewed by the testing lab. The source code serves as an excellent supporting evidence documents on how a cryptographic module works. The testing lab is required to verify that the source code is properly annotated with comments.

The testing of the module is conducting physical testing and operating of the cryptographic module following the KS X ISO/IEC 24759 Derived Test Requirements (DTR) document. The DTR should contain a specific security requirement (*i.e.*, ports and interface, key management, roles and services) [4].

2. Convergence of KCMVP Module into OpenSSL

The convergence of KMULiB v1.0 and openssl-1.0.1.e uses ARIA as a block cipher algorithm, CBC(Cipher Block Chaining) mode as a mode of operation and SSLv3 for openssl-1.0.1.e.

To make an extension version of OpenSSL containing KCMVP, top level API is not changed but extra parameter indices are added to openssl-1.0.1.e. So the original openssl-1.0.1.e client module is still interworking with the extension version including KCMVP.

The OpneSSL uses EVP API to support object files of cryptographic algorithms to use cryptographic functions. Hence an insertion process for ARIA symmetric key algorithm into EVP source tree is needed and a code change for the OpenSSL is also needed.

First to insert ARIA into object listed below **`crpto/objects/object.txt`** file needs to be changed.

```
Identified-organization     : KMU         : kmu
411 20005                   : ARIA-ECB    : aria-ecb
kmu 1 1                     : ARIA-CBC    : aria-cbc
kmu 1 2
!Cname aria-cfb128          : ARIA-CFB    : aria-cfb
Kmu 1 3
!Cname aria-ofb128          : ARIA-OFB    : aria-ofb
Kmu 1 4
```

And to declare EVP for ARIA several lines need to be inserted into **`crypto/evp/evp.h`** as follows;

```
#ifndef OPENSSL_NO_ARIA
const EVP_CIPHER *EVP_aria_cbc(void);
#endif
```

Another insertion process into **`crypto/evp/c_allc.c`** file to call ARIA as an object for a cryptographic algorithm of EVP as follows ;

```
#ifndef OPENSSL_NO_ARIA
   EVP_add_cipher(EVP_aria_cbc());
   EVP_add_cipher_alias(SN_aria_cbc,"ARIA");
   EVP_add_cipher_alias(SN_aria_cbc,"aria");
#endif
```

Now for a cryptographically secure communication by OpenSSL with ARIA capability the cryptographic algorithm list in the Ciphersuite is modified. By using this new list, SSL server and SSL client can negotiate ARIA Ciphersuite during the handshake process and KMULiB v1.0 ARIA cryptographic module can used to protect SSL procedure.

The list of ARIA Ciphersuite is as follows;


```
{1, "RSA_ARIA_128_CBC_SHA256", 0x0300C03C, SSL_kRSA, SSL_aRSA,
  SSL_ARIA, SSL_SHA1, SSL_TLSV1, SSL_NOT_EXP|SSL_HIGH|SSL_FIPS,
  SSL_HANDSHAKE_MAC_DEFAULT|TLS1_PRF, 128, 128},
{1, "DHE_RSA_ARIA_128_CBC_SHA256", 0x0300C044, SSL_kEDH,
SSL_aRSA,
  SSL_ARIA, SSL_SHA1, SSL_TLSV1, SSL_NOT_EXP|SSL_HIGH|
SSL_HANDSHAKE_MAC_DEFAULT|TLS1_PRF, 128, 128},
{1, "ECDHE_RSA_ARIA_128_CBC_SHA256", 0x0300C04C, SSL_kEECDH,
SSL_aRSA,
  SSL_ARIA, SSL_SHA1, SSL_TLSV1, SSL_NOT_EXP|SSL_HIGH|
SSL_FIPS, SSL_HANDSHAKE_MAC_DEFAULT|TLS1_PRF, 128, 128},
{1, "ECDH_RSA_ARIA_128_CBC_SHA256", 0x0300C04E, SSL_kECDHr,
SSL_aECDH,
  SSL_ARIA, SSL_SHA1, SSL_TLSV1, SSL_NOT_EXP|SSL_HIGH|
SSL_FIPS, SSL_HANDSHAKE_MAC_DEFAULT|TLS1_PRF, 128, 128},

/* end of list */
```

After the modification above, interworking process is needed to use KCMVP modules with OpenSSL. First to call KCMVP's API from OpenSSL, call command and KMULiB v1.0 library are added into the crypto folder. Then Makefile is changed to use OpenSSL and KMULiB v1.0 library together. A call procedure of SSL API including EVP API and KMULiB is as following diagram.

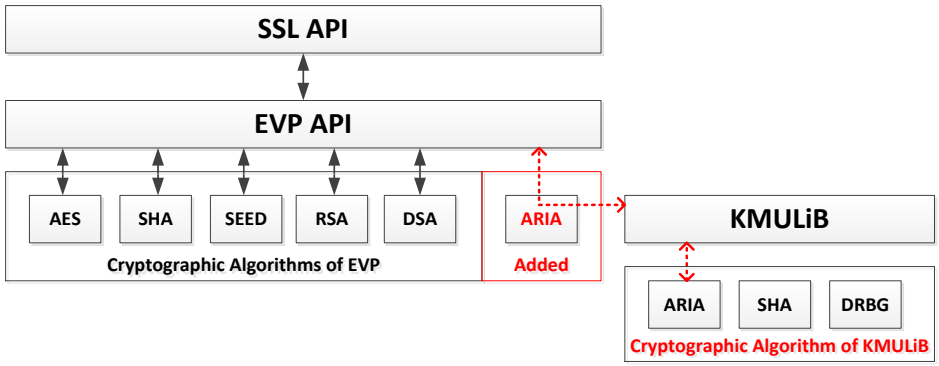


Figure 1. Call Process of SSL API, EVP API and KMULiB

Each KCMVP module has its own operation rule defined in KMULiB module. So the operation rules in the OpenSSL and KCMVP module are different from each other. So the following items should be considered for the interworking OpenSSL with KCMVP.

- Key Management: the KCMVP module controls the key for encryption and decryption as a critical parameter. But the OpenSSL manages its own keys so the boundary of key management should be considered.
- Lack of Information for Key Initialization: OpenSSL notifies key length and whether the key is for encryption or decryption only when the key is initialized. But the mode of operation is not notified by OpenSSL.
- Stopping plaintext communication: communication session must stop whenever an error occurs in block cipher in OpenSSL process. Any error in KCMVP module stops its KCMVP process, but OpenSSL may not be notified so that plaintext communication of OpenSSL continues.

- Time for key renewal: SSL protocol basically uses the same master session key except for Renegotiation Alert. And applications over SSL can change encryption and decryption keys irregularly. So lots of key initialization should be considered by two different OpenSSL and KCMVP modules.

3. Experiment Results

In this chapter, two experiment results in a real system combined OpenSSL and KCMVP modules. The first communication model is OpenSSL + KCMVP vs. OpenSSL + KCMVP, *i.e.*, server and client have OpenSSL protocols and KCMVP module. The second model contains OpenSSL + KCMVP vs. OpenSSL, so only one side contains OpenSSL. So KCMVP ARIA is not available for SSL protocol. But the second model proves the extended OpenSSL is still working with the original OpenSSL.

3.1. Experiment Environment

The specification of server and client system is as follows;

Table 2. Specification of Server and Client System

	Server	Client
OS	Linux version 2.6.32-49-generic	Linux version 2.6.32-49-generic
CPU	Intel® Core™2 Duo CPU E8400@3.00GHz	Intel® Core™ i7-2600 CPU@3.4GHz
Memory	991400kb	3603668kb
Compiler	Ubuntu 4.4.3-4 ubuntu 5.1	Ubuntu 4.4.3-4 ubuntu 5.1

The two service models perform wired communication between server and client systems. In the OpenSSL + KCMVP vs. OpenSSL + KCMVP model, the validity of the encrypted communication should be tested. And in the OpenSSL + KCMVP vs. OpenSSL model, only OpenSSL wired communication is being tested for the validity.

And the difference of performance of the extended OpenSSL + KCMVP from OpenSSL is being tested by using the wireshark[6] which is a network analysis program.

3.2. Experiment Result

The experiments confirm that the cryptographic protection is working properly between the server and client on both sides. The confirmation of Handshake procedure, Ciphersuite list selection procedure and encryption/decryption of application data are tested. And to check the performance of interworking, each service system is tested.

The Handshake procedure between OpenSSL + KCMVP vs. OpenSSL + KCMVP service model is tested. The test starts from the ClientHello message, and selects the TLS_RSA_WITH_ARIA_128_CBC_SHA256 as a Ciphersuite from the ServerHello message. The final procedure of the Handshake message ends by the the ChangeCipherSpec. And the encryption of application data is confirmed ‘OK’.

```
Handshake Protocol: Server Hello
   Handshake Type: server Hello (2)
   Length: 77
   Version: SSL 3.0 (0x0300)
   Random
   Session ID Length: 32
```



```
  Session ID:
98f63e0e01012a81db4c0319f568164db1e1fc9b03bba210…
  Cipher suite: TLS_RSA_WITH_ARIA_128_CBC_SHA256
(0xc03c)
  Compression Method: null(0)
  Extensions Length: 5
  Extension: renegotiation_info
```

Figure 2. Selection of TLS_RSA_WITH_ARIA_128_CBC_SHA256

```
SSLv3 Record Layer: Application Data Protocol: http
   Content Type: Application Data(23)
   Version: SSL 3.0 (0x0300)
   Length: 48
   Encrypted Application Data:
4eba093c4e0af7ecf35befd5bf4c0dc438aebed418034cd8…
```

Figure 3. OpenSSL + KCMVP vs. OpenSSL + KCMVP

The Handshake procedure between OpenSSL + KCMVP vs. OpenSSL service model is also tested. The 2nd test starts from the ClientHello message, and selects the TLS_RSA_WITH_AES_256_CBC_SHA as a Ciphersuite from the ServerHello message. The final procedure of the Handshake message ends by the the ChangeCipherSpec. And the encryption of application data is confirmed 'OK'.

```
Handshake Protocol: Server Hello
   Handshake Type: server Hello (2)
   Length: 77
   Version: SSL 3.0 (0x0300)
  Random
  Session ID Length: 32
  Session ID:
6e20f446b122beff6ae8b86011a6432db605f390620034ae…
  Cipher suite: TLS_RSA_WITH_AES_256_CBC_SHA
(0x0035)
  Compression Method: null(0)
  Extensions Length: 5
  Extension: renegotiation_info
```

Figure 4. Selection of TLS_RSA_WITH_ AES_256_CBC_SHA

```
SSLv3 Record Layer: Application Data Protocol: http
   Content Type: Application Data(23)
   Version: SSL 3.0 (0x0300)
   Length: 32
   Encrypted Application Data:
e9579db04991281b2feb821e78e827ae6e0afd9a08c206ce…
```

Figure 5. OpenSSL + KCMVP vs. OpenSSL

The result of speed check for the OpneSSL extension to KCMVP module is as follows;

No.	Time	Protocol	Length	Info
137	10.212277	SSL v3	226	Client Hello
139	10.212558	SSL v3	924	Server Hello, Certificate, Server Hello Done
141	10.219755	SSL v3	278	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
142	10.231252	SSL v3	141	Change Cipher Spec, Encrypted Handshake Message
143	10.231610	SSL v3	140	Application Data, Application Data
144	10.231688	SSL v3	156	Application Data, Application Data

Figure 6. Handshake and Encryption of Application Data by ARIA

No.	Time	Protocol	Length	Info
678	44.724090	SSL v3	226	Client Hello
680	44.724352	SSL v3	924	Server Hello, Certificate, Server Hello Done
682	44.725336	SSL v3	278	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
683	44.727696	SSL v3	141	Change Cipher Spec, Encrypted Handshake Message
684	44.728053	SSL v3	140	Application Data, Application Data
685	44.728141	SSL v3	156	Application Data, Application Data

Figure 7. Handshake and Encryption of Application Data by AES

From the figures above, the gap between 2nd message and 3rd message and the gap between 3rd and 4th message show time difference. The initialization process of KMULiB v1.0 between the 2nd and 3rd message and between the 3rd and 4th message bring the time difference.

Table 3. Time Difference in Messages

Interval	OpenSSL + KCMVP vs. OpenSSL + KCMVP	OpenSSL + KCMVP vs. OpenSSL
Between 2nd and 3rd message	7.197ms	0.984ms
Between 3rd and 4th message	11.497ms	2.36ms

The time consuming for encryption and decryption of real application data is getting from the 5th and 6th messages.

Table 4. Time Difference Between Messages

Interval	OpenSSL + KCMVP vs. OpenSSL + KCMVP	OpenSSL + KCMVP vs. OpenSSL
Between 5th and 6th message	0.078ms	0.092ms

For the experiment, the starting point of the KCMVP module is selected as the exact point when the first API of the KMULiB v1.0 is called. So as the key exchange process of SSL procedure finishes, server and client system initialize their KMULiB v1.0 modules. After the

initialization, server and client start encryption and decryption of application data by the cryptographic algorithms exchanged by the Ciphersuite. So the interval between the 5th and 6th messages shows that there is no big time delay due to the extension of OpenSSL to KCMVP module.

4. Conclusions

This paper suggested a converged model with the open project OpenSSL and KCMVP module which is mandatory to the server and client systems for public services. It showed an easy and simple way of the convergence to them by only a small change to interfaces of the OpenSSL. The KCMVP modules can't be changed in any single bit since the hash code of the KCMVP module should be given to guarantee the integrity of the module. So for the convergence OpenSSL with KCMVP, a suitable change in the interface of the OpenSSL is needed.

And an experiment result about the performances between the changes is provided to show that the extension is working properly.

Acknowledgements

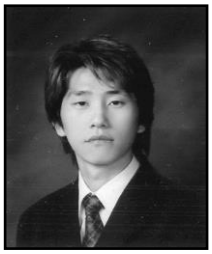
This work was supported by 2012 research program of Kookmin University in Korea.

References

[1] Open Project for SSL, http://www.openssl.org/.
[2] Cryptographic Module Validation Program, http://csrc.nist.gov/cmvp.
[3] KMULiB v1.0 Validation Module list, http://service1.nis.go.kr/.
[4] The Law to mandatory the usage of KCMVP Modules for IT systems for Public Services http://www.law.go.kr/lsInfoP.do?lsiSeq=141803&efYd=20130706#0000.
[5] Information technology -- Security techniques-- Security requirements for cryptographic modules, ISO/IEC 19790, 201208.
[6] A program to analysis of network packets, http://www.wireshark.org/.

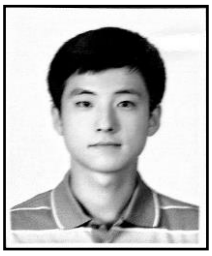
Authors

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 Mobile Communication and Cryptographic Module Implementation.



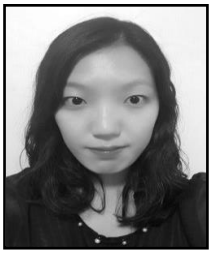
Seunghwan Yun received his M.S. degree in Mathematics from Kookmin University, Korea, in 2008. He is currently undertaking an Ph. D. course in the Center for Information Security Institute from Korea University, Korea His current research interests are in the areas of implementation of cryptographic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



Nuri Hwang 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.

Taeyean Kwon is currently undertaking an B.S. course in Department of Mathematics from Kookmin University, Korea. Her current research interests are in the areas of implementation of cryptographic modules and information security.



Chaewon Yun is currently undertaking an B.S. course in Department of Mathematics from Kookmin University, Korea. Her current research interests are in the areas of implementation of cryptographic modules and information security.