KCMVP를 준수하는 TLS Library의 구현

박재필O
(주)시큐위즈 기술연구소O
e-mail: foxyfeel@secuwiz.co.krO

An Implement TLS Library for KCMVP

J.P ParkO
SECUWIZ CO. tech labO

● 요 약 ●

본 논문에서는 국내 암호 알고리즘 검증제도(KCMVP[1])를 준수하는 TLS 1.2[2] 응용 개발 툴(C Libaray)을 개발한다. 국내 암호응용 네트워크 장비들이 암호화용 대칭키를 교환하는 방법으로 대부분 TLS프로토콜을 사용하고 있으나 RFC[3]표준에 국내알고리즘들이 등제 되지 않아 TLS 프로토콜을 사용 할 경우 알고리즘 검증제도를 준수하고 있지 못한 실정이다. 이에 본 논문에서는 TLS 1.2 표준의 cipher spec extension을 이용하여 국산 암호화 알고리즘의 TLS 적용을 구현한다.

키워드: KCMVP, TLS 1.2

I. Introduction

현재 국내 공공기관에서 사용하는 네트워크 암호장비와 암호응용 프로그램들은 모두 국정원의 CC"인증 및 KCMVP를 준용 하여야만 한다.

하지만 단순히 통신 양단간의 데이터 암호화만을 위한 대칭키 암호 알고리즘을 사용하여서는 안전한 대칭키의 통신 양단간 교환이 이루어질 수 없다.

때문에 국내 CC 인증을 준용하기 위하여 기존에 검증되어진 키교환 알고리즘 및 프로토콜을 사용하여야 하는데 KCMVP에서 인증되어진 국산 알고리즘들은 이러한 키교환 프로토콜의 표준화에 빠져있는 실정이다.

때문에 국내에서 암호관련 네트워크 장비나 응용 프로그램을 개발하는 경우 부득이 하게 KCMVP를 준용 못하는 사례가 발생 하고 있다.

본 연구에서는 이러한 제약 사항들을 극복하기 위해 키교환 표준으로 가장 많이 사용되고 있는 TLS 프로토콜의 cipher spec 정의 부분에 KCMVP를 준용하는 국내 알고리즘을 추가 구현하여 암호관련 보안 응용 장비나 프로그램을 구현하는데 KCMVP를 준용할 수 있도록 하는 개발용 crypto library를 구현하고자 한다.

II. Preliminaries

1. Related works

1.1 국내 동향

현재 국내 보안 응용개발 업체들의 대부분이 암호화를 처리하는 모듈을 개발하는데 있어 KCMVP를 준용하는 개발 라이브러리를 이용하여 개발하고 있다.

하지만 이는 단순히 통신 양단간의 대칭키 암호화에 안증된 대칭키 암호를 이용하는 것으로 그 안전성에 문제가 있다고 볼 수 있다.

또한 기존 검증되어진 키교환 프로토콜을 이용한 암호화 통신의 구현은 널리 알려진 SSL, IPSEC프로토콜을 이용하므로써 키교환의 안정성을 확보하였지만 키교환 프로토콜에 사용되어지는 비대칭키 암호화의 표준에 사용하는 알고리즘들이 국내에서 제시하는 KCMVP 표준을 준용 하고 있지 않기 때문에 엄밀히 볼 때 공공기관에서 준수하여야 하는 CC인증의 검증 범위를 벗어나고 있다.

때문에 KCMVP를 준용하는 알고리즘이 포함된 키교환 프로토콜의 사용이 시급한 상황이다.

III. The Proposed Scheme

1. TLS 프로토콜

1) TLS 프로토콜의 개요

세션 연결의 제어하기 위해 사용된 TLS 프로토콜에는 3개의 부프로토콜 즉, handshake 프로토콜, change Cipher_spec 프로토콜, alert 프로토콜이 존재한다.

TLS handshake 프로토콜은 세션 파라미터들을 협의하기 위해 사용되고, alert 프로토콜은 오류 상황을 공지하기 위해 사용된다. change Cipher_spec 프로토콜은 세션의 암호학적 파라미터들을 교환하기 위해 사용된다.

2) Handshake 프로토콜의 개요

Handshake 프로토콜은 클라이언트와 서버사이에 일련의 메시지 교환으로 이루어지며, 이 과정으로 클라이언트와 서버는 하나 이상의 보안 서비스 즉, 기밀성, 메시지 무결성, 인증, 재생방지를 형성할 수 있다.

따라서 클라이언트와 서버는 이것을 위해 알고리즘들을 협상하고 대칭키들을 유도하며, 데이터 해시와 같은 다른 세션 파라미터들을 설정해야 한다. 협상된 인증, 기밀성, 무결성 알고리즘들의 모음을 cipher_suite라 부른다.

아래 그림은 handshake 프로토콜 과정을 보여준다.

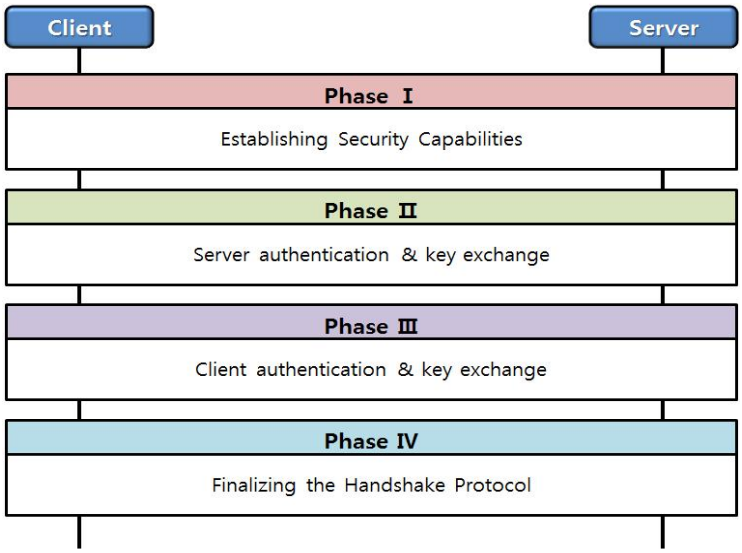

Fig. 2. Handshake 프로토콜 과정

따라서 클라이언트와 서버는 협상된 cipher_suite들로 준비된 보안 서비스들로 보호된 어플리케이션 데이터들을 교환할 수 있다.

이러한 보안 서비스들은 handshake로 협의되고 성립된다. TLS 프로토콜 스택은 다음 그림 1과 같다.

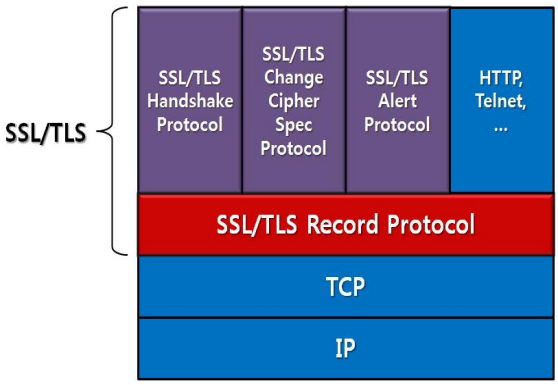

Fig. 1. TLS 프로토콜 스택

3) 추가로 정의된 cipher_suites

많은 암호학적 함수들이 보안 프로토콜에서 사용된다. 널리 알려진 암호학적 특징으로는 기밀성, 무결성, 전자서명이며, SSL에서는 기밀성, 무결성, 서명, 키 일치 등 4가지 보안기능을 사용한다.

SSL은 통신할 때 클라이언트와 서버가 사용할 암호학적 함수들의 모음을 정의한 cipher_suite을 사용한다.

KCVMP 검증필 암호모듈의 보호함수들을 사용하고 새로운 tls cipher_suite들을 정의하여 기밀성, 무결성, 서명, 키 일치 등의 보안기능을 제공한다.

Table 1. 추가 TLS Cipher_suite for ECDH key exchange & ECDSA Certificates

Cipher ID	Name	Key exchange	Encryption	Hash Function for HMAC	Hash Function for PRF
0xD000	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_128_GCM_SHA256	ECDH	ARIA_128_GCM	SHA-256	SHA-256
0xD001	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_192_GCM_SHA256	ECDH	ARIA_192_GCM	SHA-256	SHA-256
0xD002	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_256_GCM_SHA256	ECDH	ARIA_256_GCM	SHA-256	SHA-256
0xD003	TLS_KCMVP_ECDH_ECDSA_WITH_SEED_128_GCM_SHA256	ECDH	SEED_128_GCM	SHA-256	SHA-256
0xD004	TLS_KCMVP_ECDH_ECDSA_WITH_LEA_128_GCM_SHA256	ECDH	LEA_128_GCM	SHA-256	SHA-256
0xD005	TLS_KCMVP_ECDH_ECDSA_WITH_LEA_192_GCM_SHA256	ECDH	LEA_192_GCM	SHA-256	SHA-256
0xD006	TLS_KCMVP_ECDH_ECDSA_WITH_LEA_256_GCM_SHA256	ECDH	LEA_256_GCM	SHA-256	SHA-256
0xD020	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_128_CBC_SHA256	ECDH	ARIA_128_CBC	SHA-256	SHA-256
0xD021	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_192_CBC_SHA256	ECDH	ARIA_192_CBC	SHA-256	SHA-256
0xD022	TLS_KCMVP_ECDH_ECDSA_WITH_ARIA_256_CBC_SHA256	ECDH	ARIA_256_CBC	SHA-256	SHA-256
0xD023	TLS_KCMVP_ECDH_ECDSA_WITH_SEED_128_CBC_SHA256	ECDH	SEED_128_CBC	SHA-256	SHA-256
0xD024	TLS_KCMVP_ECDH_ECDSA_WITH_LEA_128_CBC_SHA256	ECDH	LEA_128_CBC	SHA-256	SHA-256
0xD025	TLS_KCMVP_ECDH_ECDSA_WITH_LEA_192_CBC_SHA256	ECDH	LEA_192_CBC	SHA-256	SHA-256

Table 2. 추가 ECDH 및 ECDSA TLS Cipher Suites 상세

구분	형식	설명	해당 표준
키 교환	ECDH	NIST-P256 또는 P224 curve	ANSI X9.63
전자서명	ECDSA	NIST-P256 또는 P224 curve	NIST FIPS 186-3
블록암호	ARIA, SEED, LEA	Data Channel 암호화 방식	KS X 1213-1(ARIA) TTAS.KO-12.0004/R1(SEED) TTAS.KO-12.0223(LEA)
키 길이	128, 192, 256 비트	암호화 키 길이	-
운영모드	GCM, CBC	블록암호 사용 시 운영 방식	TTAS.KO-12.0131(GCM) KS X 1213-1(ARIA) TTAS.KO-12.0025(SEED) TTAK.KO-12.0246(LEA)
해시함수	SHA256	인증서 용 전자서명에서 사용	ISO/IEC 10118-3 ISO/IEC 9797-2

Table 3. 추가 TLS Cipher Suites for RSA certificates

Cipher ID	Name	Key exchange	Encryption	Hash Function for HMAC	Hash Function for PRF
0xD040	TLS_KCMVP_RSA_WITH_ARIA_128_GCM_SHA256	DH-2048	ARIA_128_GCM	SHA-256	SHA-256
0xD041	TLS_KCMVP_RSA_WITH_ARIA_192_GCM_SHA256	DH-2048	ARIA_192_GCM	SHA-256	SHA-256
0xD042	TLS_KCMVP_RSA_WITH_ARIA_256_GCM_SHA256	DH-2048	ARIA_256_GCM	SHA-256	SHA-256
0xD043	TLS_KCMVP_RSA_WITH_SEED_128_GCM_SHA256	DH-2048	SEED_128_GCM	SHA-256	SHA-256
0xD044	TLS_KCMVP_RSA_WITH_LEA_128_GCM_SHA256	DH-2048	LEA_128_GCM	SHA-256	SHA-256
0xD045	TLS_KCMVP_RSA_WITH_LEA_192_GCM_SHA256	DH-2048	LEA_192_GCM	SHA-256	SHA-256
0xD046	TLS_KCMVP_RSA_WITH_LEA_256_GCM_SHA256	DH-2048	LEA_256_GCM	SHA-256	SHA-256
0xD060	TLS_KCMVP_RSA_WITH_ARIA_128_CBC_SHA256	DH-2048	ARIA_128_CBC	SHA-256	SHA-256
0xD061	TLS_KCMVP_RSA_WITH_ARIA_192_CBC_SHA256	DH-2048	ARIA_192_CBC	SHA-256	SHA-256
0xD062	TLS_KCMVP_RSA_WITH_ARIA_256_CBC_SHA256	DH-2048	ARIA_256_CBC	SHA-256	SHA-256
0xD063	TLS_KCMVP_RSA_WITH_SEED_128_CBC_SHA256	DH-2048	SEED_128_CBC	SHA-256	SHA-256
0xD064	TLS_KCMVP_RSA_WITH_LEA_128_CBC_SHA256	DH-2048	LEA_128_CBC	SHA-256	SHA-256
0xD065	TLS_KCMVP_RSA_WITH_LEA_192_CBC_SHA256	DH-2048	LEA_192_CBC	SHA-256	SHA-256

Table 4. 추가 RSA TLS Cipher Suites 상세

구분	형식	설명	해당 표준
키 교환	DH	공개키 : 2048-bit 개인키 : 224-bit, 256-bit	PKCS#3
전자서명	RSA-PSS	공개키 : 2048-bit, 3072-bit 해시함수 : 224-bit 또는 256-bit	PKCS#1
블록암호	ARIA, SEED, LEA	Data Channel 암호화 방식	KS X 1213-1(ARIA) TTAS.KO-12.0004/R1(SEED) TTAS.KO-12.0223(LEA)
키 길이	128, 192, 256 비트	암호화 키 길이	-
운영모드	GCM, CBC	블록암호 사용 시 운영 방식	TTAS.KO-12.0131(GCM) KS X 1213-1(ARIA) TTAS.KO-12.0025(SEED) TTAK.KO-12.0246(LEA)
해시함수	SHA256	인증서용 전자서명에서 사용	ISO/IEC 10118-3 ISO/IEC 9797-2

IV. Conclusions

본연구의 목표는 국내에서 제한하고 있는 암호알고리즘과 RFC표준에 기술되고 있는 보안표준간의 괴리로 인하여 본의가 아니게 국내 보안표준을 준수하지 못하고 있는 보안 응용 개발자들의 어려움을 보완할 수 있는 방법이 되리라 기대되어진다. 또한 가장 널리 사용되고 있는 표준인 TLS 프로토콜에 국내 암호검증 표준을 적용하므로써 보다 많은 보안 응용 프로그램이 국내 보안 표준을 준수하는데 기여할 수 있을 것으로 기대된다.

REFERENCES

[1] Korea Cryptographic Module Validation
[2] Transport Layer Security
[3] RFC(Request for Comments)
[4] Common Criteria SO/IEC 15408