ECDSA 검증이 가능한 효율적인 WAP PKI용 POP 프로토콜

김성덕, 김승주, 원동호
성균관대학교 정보통신공학부
e-mail:netsec@naver.com, {skim, dhwon}@security.re.kr

An Efficient ECDSA Verifiable POP protocol for WAP PKI

Sungduk Kim, Seungjoo Kim, Dongho Won
School of Information and Communication Engineering,
Sungkyunkwan University

요 약

POP는 공개키 인증서의 발급과정에서 인증기관(CA)이 신청자가 제출한 공개키에 대응하는 비밀키를 가지고 있는가와 선택한 암호 알고리즘 및 용도에 적합한 연산을 수행할 능력이 있는 가를 확인하는 절차를 의미한다. WAP PKI에서는 POP 확인을 위해 암호화용 인증서는 WTLS 프로토콜을 수행하고, 전자서명용은 signText()함수를 이용하여 전자서명을 생성하도록 정의하고 있다. 이런 방식은 WTLS 프로토콜의 수행과 전자서명의 생성과정에서 많은 연산과 통신 부담이 발생한다. 본 논문에서는 ECDSA에 기반한 Signcryption 방식을 이용하여 효율적으로 POP를 검증하는 방식을 제안한다.

1. 서론

POP(Proof Of Possession)은 공개키용 인증서 신청자가 인증기관(CA)에 자신이 제출한 공개키에 대응하는 비밀키를 소유하고 있으며, 이를 증명하기 위해 선택한 암호 알고리즘과 용도에 적합한 프로토콜을 수행하는 과정이다. 인증서 신청자가 전자서명용 인증서를 신청한 경우에는 전자서명을 제출해야 하고, 암호화용 인증서를 신청한 경우에는 선택한 암호화 방식(키전달, 키동의 등)에 따라 세션키를 생성하기 위한 프로토콜을 수행하여야 한다.

현재, POP를 정의하고 있는 표준은 IETF의 RFC 4210과 4211, WAP 포럼의 WAP-217-WPKI이며, 관련된 부속 표준들이 있다[1-4,10,11]. IETF의 POP 프로토콜은 개인용 컴퓨터와 같이 계산 및 통신기능이 좋은 환경에 적합하도록 설계되어 핸드폰과 같은 계산 및 통신기능이 제한되는 장비에는 부적합하다[3,4]. 이에 WAP 포럼은 핸드폰과 같은 소형장비에 적합한 PKI를 정의하고 있는데, 인증기관은 WTLS 프로토콜을 수행하여 암호화용 공개키에 대한 POP를 확인하고, signText()함수를 이용하여 생성된 전자서명의 검증을 통해 전자서명용 공개키에 대한 POP를 확인한다. WTLS 프로토콜과 signText() 함수는 WAP PKI를 지원하는 단말기에 기본적으로 탑재되는 모듈이므로, 이를 통해 POP를 확인하면 단말기의 프로그램 크기를 줄일 수 있다는 장점은 있지만, WTLS 프로토콜 자체가 복잡하고, 통신량이 많으며, 전자서명용 비밀키의 POP만 확인하는 경우에도 별도의 WTLS 세션을 만들어야 하는 문제점이 있다.

본 논문에서는 인증서가 없는 핸드폰으로 ECDSA와 ECDH용 공개키 인증서를 동시에 인증기관에 신청하는 상황에서 ECDSA를 이용한 Signcryption 방식을 기반으로 WAP의 POP 프로토콜보다 효율적인 프로토콜을 제안한다.

2장에서는 WAP PKI의 POP 과정에 대해서 설명한다. 3장에서는 Signcryption과 ECDSA 검증이 가능한 Signcryption에 대해서 설명하고, 새로운 ECDSA 기반의 POP 프로토콜을 제안한다. 4장에서는 제안한 방식의 효율성과 안전성을 비교한 후, 5장에서 결론을 맺는다.

본 논문에서 사용하는 기호와 용어는 ANSI X9.62의 정의를 따른다[5,6].

2. WAP PKI의 POP 방식

WAP PKI에서는 암호화 세션을 생성하는 방법으로 사용하는 WTLS 프로토콜은 클라이언트와 서버의 인증(Authentication) 수준에 따라 클래스를 구분하며, 가장 높은 수준인 클래스 3은 클라이언트와 서버가 상호간의 인증을 수행하는 수준으로 정의하였다[10,11].

	Alice	Communication	CA
WTLS	[ClientHello(R_A)]	$\rightarrow R_A \rightarrow$	[ServerHello(R_{CA})] [Certificate($Cert(Q_{CAK})$)] [CertificateRequest]
	[Certificate($SelfCert(Q_{AK})$)] [CertificateVerify($Cert(Q_{CAK})$)]	$\leftarrow R_{CA}$, $Cert(Q_{CAK})\leftarrow$	[ServerHelloDone]]
	$K_p = d_{AK}Q_{CAK}$ $K_m = H(K_p, R_A, R_{CA})$ [ChangeCipherSpec] [Finished][Application Data]	$\rightarrow SelfCert(Q_{AK})\rightarrow$ $\leftarrow WTLS\ Message\leftarrow$	$K_p = d_{CAK}Q_{AK}$ $K_m = H(K_p, R_{CA}, R_A)$ [ChangeCipherSpec] [Finished][Application Data]
ECDSA	ID/PW 입력 $k \in_R [1, n-1]$ $M = Nonce\|\|Name\|\|ID\|\|PW$ $Q_k = (x_k, y_k) = kG$ $r = \overline{x_k}(mod\ n)$ $c = H(M)$ $s = (e + d_{AS}r)k^{-1}(mod\ n)$ $SelfCert(Q_{AS})$ 생성	$\rightarrow Start\ signal\rightarrow$ $\leftarrow Nonce \leftarrow$ $\rightarrow M, r, s \rightarrow$ $\rightarrow SelfCert(Q_{AS})\rightarrow$ $\leftarrow Cert(Q_S)\leftarrow$ $\leftarrow Cert(Q_K)\leftarrow$	Nonce 생성 $e = H(M)$ $u_1 = es^{-1}\ (mod\ n)$ $u_2 = rs^{-1}\ (mod\ n)$ $Q_k = u_1G + u_2Q_{AS}$ $Check: r \equiv \overline{x_k}$

(그림 1) WAP의 ECDH & ECDSA POP 과정

WAP PKI는 2001년 WAP 1.0 표준을 채택하면서 암호화용 인증서의 POP를 수행하는 방법으로 WTLS 프로토콜을 클래스 3수준으로 수행하는 것으로 정의하였다. 클래스 3을 수행하려면 클라이언트도 인증서를 가지고 있어야 한다. 하지만, 인증서를 신청하는 단계에서는 인증서가 없으므로 클라이언트는 임시로 Self-signed 인증서를 생성하여 이용하도록 정의하고 있다.

또한, WAP PKI는 전자서명용 공개키에 대한 POP를 확인하는 방법으로 신청자가 signText() 함수를 이용하여 전자서명을 생성하고, 이를 WTLS 세션을 통해서 인증기관에 전달하도록 하였으며, 이 과정에서 signText()의 설정은 다음과 같다.

Request=Crypto.signText(nonce+":"+Name+":"+ID+":"+Password, 5, 0);

여기서 Request는 signedContent 형식으로 구성된 전문이고, nonce는 인증기관이 보낸 난수이며, ID와 Password는 인증서 신청자의 신원을 확인하기 위해서 인증기관이 발급한 일회성 정보이다. signText()에서 두 번째 인자 5는 INCLUDE_CONTENT (0x0001)과 INCLUDE_CERTIFICATE(0x0004)를 OR 연산한 값으로 각각 전자서명의 대상이 되는 메시지 ("nonce+":"+Name+":"+ID+":"+Password")와 인증서를 signedContent에 포함시킨다는 것을 의미한다. 인증서 신청자가 암호화용 인증서와 전자서명용 인증서를 동시에 신청한다면 WAP의 POP 과정은 <그림 1>과 같다[12].

3. 효율적인 POP 프로토콜의 제안

3.1 Signcryption 및 제안동기

1997년 Y.Zheng이 처음으로 제안한 Signcryption 방식은 암호화를 위한 세션키와 전자서명 생성 과정을 통합한 방식으로 적은 통신 및 계산 비용으로 암호화와 전자서명을 한 번에 해결할 수 있다는 장점이 있다[7-9].

Y.Zheng의 Signcryption 방식은 계산 및 통신 측면에서 효율성이 있지만, Signcryption의 수신자가 송신자의 전자서명을 타인에게 검증받으려면 수신자의 비밀키가 노출되지 않도록 ZKIP 프로토콜이 동원해야 하는 문제점이 있다. 이를 보완한 여러 가지 방안이 제안되고 있으며[13], ECDSA 공개검증이 가능한 방식으로는 2002년 신준범 등이 제안한 방식이 있다. 이 방식은 ECDSA를 일부 수정하여 공개검증이 가능하지만, 계산량을 늘어나는 단점이 있다.[15].

ECDSA에 따라 전자서명을 생성하는 경우, 서명자는 먼저 난수(k)를 1개 선택하고 Scalar 곱셈을 통해 임의의 좌표값을 계산한 후, 생성된 좌표의 x축 값을 ECDSA 전자서명의 r값으로 이용한다. 이때 사용하는 난수(k)를 ECDH용 비밀키(d_{AK})로 활용한다면 인증서 신청자는 별도로 ECDH용 비밀키

를 생성하고 이에 대응하는 공개키를 생성하는 과정을 줄일 수 있으며, 인증서 신청자의 신원확인용으로 인증기관이 발급한 ID와 패스워드(PW)를 감추기 위해 WTLS 프로토콜을 사용하지 않고, ECDH 키를 직접 사용할 수 있다.

ECDH 키의 생성과정은 1번의 Scalar 곱셈으로 구성되고, 검증 과정은 각각 2번, 1번의 Scalar 곱셈으로 구성된다.

80MHz ARM7TDMI 프로세서에서 192비트 ECC 연산을 수행하는 경우, Scalar 곱셈에 약 38ms가 소요되는 것으로 보고되었다[14].

인증서 신청자	통신구간	인증기관
$d_S, d_K \in_R [1, n-1]$	$\rightarrow ID, Q_S, Q_K, cs \rightarrow$	$Q_{ECDH} = (x_1, y_1) = d_{CA_K} Q_K - ①$
$Q_S = (x_S, y_S) = d_S G$		$r = \overline{x_1} (mod\ n) - ②$
$Q_K = (x_K, y_K) = d_K G$		$s = D_r(cs) - ③$
$Q_{ECDH} = (x_1, y_1) = d_K Q_{CA_K}$		$e = H(r \| PW) - ④$
$r = \overline{x_1} (mod\ n)$		$u_1 = es^{-1} (mod\ n) - ⑤$
$e = H(r \| PW)\ (mod\ n)$		$u_2 = rs^{-1} (mod\ n) - ⑥$
$s = (e + d_S r) d_K^{-1} (mod\ n)$	$\leftarrow Cert(Q_S) \leftarrow$	$Check : Q_K \equiv u_1 G + u_2 Q_S - ⑦$
$cs = E_r(s)$	$\leftarrow Cert(Q_K) \leftarrow$	

(그림 2) Signcryption을 이용한 효율적인 POP 프로토콜

3.2 Signcryption을 이용한 POP 프로토콜

신준범 등이 제안한 Signcryption 방식은 ECDSA 전자서명으로 전환할 수 있도록 하기 위해, Y.Zheng의 ECC형 Signcryption 방식과 비교하여 Signcryption의 생성과 검증과정에 Scalar 곱셈 연산이 1번씩 추가 된다. 하지만, 인증기관은 POP를 수행할 때 POP 결과를 제3자에게 검증을 의뢰할 필요가 없으므로 <그림 2>와 같이 좀 더 효율적인 프로토콜로 발전시킬 수 있다.

이 방식은 ECDSA를 Signcryption 방식으로 변환하여, 인증서 신청과정에서 POP의 생성과 검증과정의 효율성을 보완한 방식이다. POP의 생성과정에서 ECDH 세션키를 생성하고, 이것을 ECDSA의 r값으로 사용하며, 전자서명의 대상은 r값과 패스워드(PW)가 된다.

인증기관은 인증서 신청자의 POP 정보를 받고, ①, ②, ⑦과정을 통해 ECDH 세션키를 생성하고 검증하며, ②~⑦ 과정을 통해 ECDSA의 검증과정을 수행한다. ④과정은 인증서 신청자의 신원을 One Time Password를 통해 확인하는 과정이다. ①~⑦ 과정을 모두 수행하면, 인증기관은 인증서 신청자의 신원을 확인하고, ECDSA용 공개키(Q_S)와 ECDH용 공개키(Q_K)에 대한 POP를 모두 확인할 수 있다.

4. 효율성과 안전성

4.1 계산 비용

ECC 환경에서 가장 시간이 많이 소요되는 연산은 Scalar 곱셈이다. ECDSA 전자서명의 생성과정과 WAP PKI를 준수하는 경우, 인증서 시청자는 총 6번의 Scalar 곱셈을 수행해야 하며, 최소한 228ms가 소요된다. 하지만 제안한 방법을 사용하는 경우, 3번의 Scalar 곱셈만 수행하여 114ms만 소요되므로 50% 계산부담이 줄어든다.

WAP에서는 POP 검증을 담당하는 인증기관이 7번의 Scalar 곱셈(3번의 ECDSA 전자서명 검증 및 1번의 ECDH 키 생성)을 수행해야 하지만, 제안한 방법의 경우, ECDSA 검증과정에서 3번의 Scalar 곱셈을 수행하면 되므로 계산부하를 43%까지 감소시킬 수 있다.

4.2 통신 비용

|a|는 메시지 a의 길이를 의미한다. 인증서 신청자가 ECDSA용 인증서와 ECDH용 인증서를 동시에 신청하는 경우 WAP PKI를 따르는 경우 WTLS 세션을 생성하고, ECDSA 전자서명을 생성/검증해야 한다. WTLS 세션 생성에 필요한 메시지와 발급된 인증서를 제외하고, 인증서 신청자와 인증기관은 |n|+3|Certificate|+|H|+|ID|+|PW| 길이의 메시지를 전송해야 한다. 그리고 인증서 신청자와 인증기관은 4번의 상호작용(송신+수신)을 수행해야 하지만, 제안한 방법을 이용하는 경우 |ID|+|n|+2|Q| 길이의 메시지만 전송하고, 1번의 송수신 과정으로 이루어진다.

4.3 안전성

제안한 프로토콜에서 공격자에게 노출되는 정보는 인증서 신청자의 ID, ECDSA용 인증서, ECDH용 공개키, 그리고 암호화된 서명값이다. 암호화된 서명값 외에는 공개되어도 문제가 없는 정보이다. 공격자가

제안한 프로토콜을 공격하려면 먼저 암호화된 서명문을 복호화해야 한다. 하지만, 암호화된 서명문을 복호화하기 위한 키를 찾는 것은 인증기관의 ECDH용 공개키와, 인증서 신청자의 ECDH용 공개키를 이용하여 ECDH키를 생성해야 하므로 이것은 공격자가 Inverting Diffie-Hellman Problem (또는 Computational Diffie-Hellman Problem)을 해결해야 하는 것을 의미하지만, 아직까지 이 문제를 풀 수 있는 방안은 없는 것으로 평가되고 있다. 그러므로 제안한 프로토콜은 안전하다고 할 수 있다[16].

5. 결론

본 논문에서는 핸드폰과 같이 계산 및 통신 능력이 떨어지는 이동형 기기에 적합한 POP 프로토콜을 제안하였다. 대표적인 WPKI 모델인 WAP PKI는 ECDH 공개키와 ECDSA 공개키의 POP를 확인하기 위해 가각 WTLS와 ECDSA 전자서명을 이용하도록 정의하고 있다. 하지만 이 방법은 무선기기에 많은 계산/통신 부담을 초래한다. 하지만, 본 논문에서 제안한 POP 프로토콜은 ECDH 공개키와 ECDSA 공개키에 대한 POP를 Signcryption에 기반한 방식으로 확인하므로 WAP PKI와 비교하여 인증서 신청자와 인증기관의 통신 및 계산 부담을 50% 가까이 경감하는 장점이 있다.

또한, IETF에서 정한 방식과 동일한 수준으로 인증서 신청자의 신원을 확인할 수 있으며, WTLS 프로토콜과 ECDSA 전자서명을 통해 POP를 확인하는 과정에서 발생하는 많은 통신량과 복수의 송수신 과정을 1번의 송수신과정을 통해 해결할 수 있는 방안을 제시하였다.

제시한 방안은 IETF의 POP 과정에도 적용이 가능하므로 향후, 이에 대한 연구가 필요할 것으로 판단된다.

참고문헌

[1] C. Adams, S. Farrell, T. Kause, T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocols", IETF RFC 4210, September 2005.

[2] J. Schaad, "Internet X.509 Certificate Request Message Format"", IETF RFC4211, September 2005.

[3] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed Hashing for Message Authentication", RFC 2104, February 1997.

[4] Cheng, P. and R. Glenn, "Test Cases for HMAC-MD5 and HMAC-SHA-1", RFC 2202, September 1997.

[5] FIPS PUB 186-2 Digital Signature Standard, 2000 January 27

[6] ANSI X9.62 "Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm(ECDSA)"

[7] Y.Zheng. "Digital signcryption or how to achieve cost (signature & encryption) cost (signature) + cost (encryption). In: CRYPTO''97, LNCS 1294, pages 165-179. Springer-Verlag, 1997.

[8] Y. Zheng. "Signcryption and its application in efficient public key solution. In: ISW '97, LNCS 1397, pages 291-312. Springer-Verlag, 1998.

[9] IEEE P1363a : Statndard Specifications for Public-Key Cryptography : Additional Techniques, "Shortened Digital Signature, Signcryption and Compact and Unforgeable Key Agreement Schemes", Yuliang Zheng, 1998.

[10] Wireless Application Protocol Public Key Infrastructure Definition, WAP-217-WPKI Version 24-Apr-2001

[11] Wireless Application Protocol WMLScript Crypto Library, WAP-161-WMLScriptCrypto-20010620-a Version 20-Jun-2001

[12] Dong Jin Kwak, Jae Cheol Ha, Hoon Jae Lee, Hwan Koo Kim, Sang Jae Moon, "A WTLS Handshake Protocol with User Anonymity and Forward Secrecy", CIC 2002, Seoul, Korea, October 29 - November 1, 2002. Revised Papers, LNCS 2524, pp.219-230.

[13]Signcryption Central:www.signcryption.net

[14] MIRACL(Multiprecision Integer and Rational Arithmetic C/C++ Library), www.indigo.ie/mscott/

[15] Jun-Bum Shin, Kwangsu Lee, Kyungah Shim, "New DSA-Verifiable Signcryption Schemes", ICISC 2002, Seoul, Korea, November 28-29, 2002. Revised Papers, LNCS 2587, p.35

[16] Tatsuaki Okamoto, David Pointcheval, "The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes", PKC 2001, Cheju, Korea, Feb 13-15, 2001. LNCS 1992, pp.214