Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.32, NO.6, Dec. 2022 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2022.32.6.1069

제로 트러스트를 위한 소프트웨어 정의 경계(SDP) 인증 메커니즘 제안 및 ECC 암호 구현*

이 윤 경,† 김 정 녀‡
한국전자통신연구원 (책임연구원)

Software Defined Perimeter(SDP) Authentication Mechanism for Zero Trust and Implementation of ECC Cryptoraphy*

Yun-kyung Lee,† Jeong-nyeo Kim‡
Electronics and Telecommunications Research Institute (Principal Researcher)

요 약

확인하기 전에는 어느 것도 믿지 말라는 의미의 제로 트러스트가 보안에서 핫 이슈로 떠오르고 있다. 직접 확인하고, 신뢰할 수 있는 만큼만 네트워크에 연결될 수 있도록 네트워크 경계를 설정하는 것이 제로 트러스트이다. 이러한 개념은 선 검증을 하고 해당 클라이언트에 접속 권한이 있는 만큼만 네트워크 경계를 동적 방화벽으로 만들어 주는 SDP의 개념과도 맞닿아 있다. 그래서 제로 트러스트 아키텍처에서도 제로 트러스트를 실현할 수 있는 예로 SDP 모델을 추천한다. 본 논문에서는 제로 트러스트를 위하여 SDP에서 보완하여야 할 부분을 지적하고 이를 극복하는 방안을 제시한다. 또한 SDP의 엔터티가 되기 위한 과정의 하나인 온보딩 방법을 제안하고, 제안된 온보딩 방법에서 많은 시간이 소요되는 연산의 하나인 ECDSA 성능을 측정하고, ECC 성능 최적화를 위한 구현 방법을 제시한다.

ABSTRACT

Zero trust, which means never trust anything before verifying it, is emerging as a hot issue in security field. After authenticating users, zero trust establishes network boundaries so that only networks in the trusted range can be accessed. This concept is also consistent with the concept of SDP, which performs pre-verification and creates a network boundary with a dynamic firewall so that clients can access only as many as they have permission to connect. Therefore, we recommend the SDP model as an example of how zero trust can be achieved in a zero trust architecture. In this paper, we point out the areas where SDP needs to be modified for zero trust and suggest ways to overcome them. In addition, we propose an onboarding method, which is one of the processes for becoming an SDP entity, and present performance measurement results.

Keywords: Software Defined Perimeter, Authentication, ECC, Digital Signature

Received(10. 12. 2022), Modified(11. 15. 2022), Accepted(11. 21. 2022)
* 이 논문은 2022년도 정부(방위사업청)의 재원으로 국방기술진흥연구소의 지원을 받아 수행된 연구임(No. 20-107-A00-005-001, 사이버전장 공격 확산방어용 사이버 위협 상황인지 기반 능동대응 기술)
† 주저자, neohappy@etri.re.kr
‡ 교신저자, jnkim@etri.re.kr(Corresponding author)

I. 서 론

최근들어 '제로 트러스트' 라는 용어를 쉽게 접할 수 있다. 제로 트러스트는 말 그대로 신뢰가 제로인 상태를 의미한다. 우리 회사 사람이니까, 우리 회사의 자산이니까 믿고 회사 네트워크로의 연결을 허용해주는 관습에서 벗어나서, 회사 네트워크에 연결하고자 하는 순간의 사용자 상황 및 사용자가 사용하는 기기의 보안 상태까지 확인한 후 회사 네트워크 접속을 허용해주되, 사용자 및 사용자가 이용 중인 기기에 대한 신뢰 상태에 따라 회사 네트워크 접속 범위를 다르게 해야 한다는 의미이다. 또한 회사 네트워크에 접속해서 다양한 회사 서버에 접속하여 어플리케이션들을 이용하는 도중에도 지속적으로 사용자의 행동을 모니터링하고, 그 결과를 사용자의 회사 네트워크 접속 범위 설정에 실시간으로 반영하는 것을 최종 목표로 한다. 사용자뿐만 아니라 사용자가 이용 중인 기기의 보안 상태(예, OS 최신 업데이트 여부, 보안 프로그램 동작 여부, 기기의 무결성, 등)를 실시간으로 모니터링하여 사용자의 네트워크 접속 범위 결정에 반영한다.

NIST의 제로 트러스트 아키텍처[7] 문서에서는 제로 트러스트를 가장 잘 반영할 수 있는 모델의 하나로 소프트웨어 정의 경계(SDP)를 추천하고 있다. SDP를 구현하되, SIEM, SOAR, EDR 등의 솔루션과 연동하여 사용자의 행동 특성, 사용자의 물리적 위치, 사용자가 이용 중인 기기의 보안 상태, 지속적인 사용자 행동 모니터링 결과 등을 네트워크 접속 제어에 반영함으로써 [7]에서 설명한 제로 트러스트 아키텍처를 구축할 수 있다. 본 논문에서는 제로 트러스트를 만족하기 위한 첫 단계인 사용자 및 기기 인증을 SDP에서 어떻게 반영하여 구현할 것인지에 대해서 제시한다. 또한 제로 트러스트를 더욱 잘 만족할 수 있는 SDP를 위한 온보딩 과정에서의 인증 메커니즘을 제안한다. 마지막으로 제안한 SDP 인증 구조를 구현하는 데 있어서 가장 시간이 많이 필요할 것으로 보이는 서명 생성 및 검증에 소요되는 시간을 측정해 보고, 전체 SDP 인증 메커니즘에 미치는 영향을 살펴본다.

II. 관련연구

2.1 제로 트러스트

제로 트러스트는 공격자가 주변 어디에든 존재할 수 있음을 명심하고, 모든 것을 의심해야 함을 강조하는 개념이다. NIST SP 800-207 "Zero Trust Architecture" [7]가 발간되면서 제로 트러스트라는 용어가 본격적으로 알려지기 시작했다. 제로 트러스트는 회사 단위로 적용하는 것이 가장 좋고, 제로 트러스트를 적용하는 회사는 누구도, 어떤 기기도 신뢰해서는 안 되고, 지속해서 신뢰 정도를 평가하고 분석해야 함을 강조한다. 그리고 회사 내부 네트워크에 대한 접근 룰은 클라이언트가 요청한 액션을 수행하는데 필요한 최소한의 권한만을 허용할 수 있도록 가능한 접근 범위를 작게 만들어야 한다.

제로 트러스트는 인증과 인가의 두 영역에 적용되는 개념으로, subject(사용자)의 요청에 대한 액세스 여부를 결정할 때 다음의 네 가지를 고려한다.

① subject의 요청에 비해서 subject의 신원에 대한 신뢰 레벨은 어느 정도 인가?
② subject의 신원에 대한 신뢰 레벨을 고려하여 리소스에 대한 액세스가 허용되는가?
③ 액세스 요청에 사용된 기기가 적절한 보안 상태에 있는가?
④ 고려해야 할 다른 요소나 신뢰 레벨의 변화가 있는가?

이들 요소를 고려하여 개개의 리소스 액세스 요청에 대해서 지속적으로 정확하게 집행되도록 해야 한다.

Fig. 1.에서 볼 수 있듯이, 제로 트러스트의 논리적 구성요소에는 접근 여부를 결정하는 PDP (Policy Decision Point)와 접근 여부 결정 결과를 실행하는 PEP(Policy Enforcement Point)

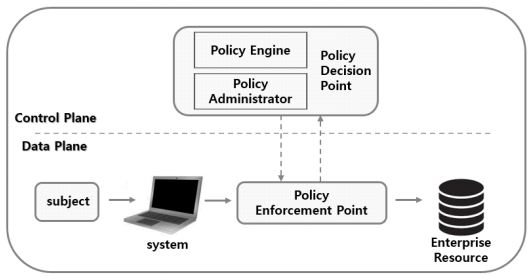


Fig. 1. Zero trust logical component

가 있다. 또한 제로 트러스트는 네트워크 구성 같은 네트워크 컨트롤을 위한 컨트롤 영역(control plane)과 어플리케이션 데이터 통신을 위한 데이터 영역(data plane)으로 구성된다.

PDP는 PE(Policy Engine)와 PA(Policy Administrator)로 구성되는데, PE는 접근 여부에 관한 결정과 이를 로그로 남기는 기능을 수행한다. PE가 네트워크에 대한 접근 여부를 결정할 때는 제로 트러스트가 적용된 회사의 접근 정책과 외부의 다양한 보안 장비(예를 들면, CDM 시스템, Threat Intelligence 서비스, SIEM 시스템 등)들로부터 입력받은 정보를 이용한다. PA는 클라이언트(subject)와 클라이언트 접속을 요청한 서비스 리소스간 통신경로를 생성하고 중단하라는 명령을 PEP에 전달한다. 이 과정에서 클라이언트가 리소스에 접속할 때 필요한 인증 토큰 또는 자격증명을 생성한다.

2.2 제로 트러스트 성숙도 모델

미국 CISA(The Cybersecurity and Infrastructure Security Agency)에서 발간한 "Zero Trust Maturity Model"[6] 은 제로 트러스트를 실현한 상태를 단계로 표현한 것이다. 가장 기본적인 단계를 'Traditional'로 표현하고, 조금 더 발전된 단계를 'Advanced'로 표현한다. 제로 트러스트를 가장 잘 구현한 단계를 'Optional' 로 표현하여 가장 이상적인 단계임을 나타내었다.

제로 트러스트 성숙도 모델에서는 다섯 개의 구분되는 영역을 기둥(pillar)으로 표현하였고, 이 기둥에는 신원(identity), 기기(device), 네트워크 및 환경(network/environment), 어플리케이션(application workload), 데이터가 있다. 이들 중 인증과 관련된 부분인 신원(identity)의 제로 트러스트 성숙도 모델을 살펴보면, 인증을 위해서 패스워드 또는 MFA(Multi-Factor Authentication)를 사용하는 것이 가장 기본(Traditional) 단계이고, MFA만을 이용하여 인증하는 것은 더욱 발전된 단계(Advanced)이다. 접속을 요청하는 시점에만 인증하는 것이 아니라 지속적으로 신원을 확인하는 것이 가장 높은 성숙도를 만족하는 'Optional' 단계이다. 또한 기기 부분의 제로 트러스트 성숙도 모델에서는 회사에서 기기가 회사의 규정을 따르고 있는지를 확인할 수 없고, 기기의 보안 상태를 확인 할 수 없는 단계가 기본(Traditional) 단계이고, 기기가 네트워크에 접속하는 순간 기기의 상태 확인이 가능한 단계가 보다 발전된 단계(Advanced)이다. 그리고 기기에 EDR 솔루션을 설치하는 등 기기의 규정 준수 여부와 기기의 보안 상태를 지속적으로 모니터링 가능한 단계는 가장 높은 제로 트러스트 성숙도를 만족하는 단계이다.

2.3 소프트웨어 정의 경계(SDP)

소프트웨어 정의 경계는 2014년 CSA(Cloud Security Alliance)에서 버전 1.0 스펙[1]을 발표하였고, 2022년 3월 버전 2.0 스펙[2]을 발표하였다. SDP는 기존 보안 솔루션들의 단점을 보완하여 보다 안전한 네트워크를 구축하기 위한 것으로, 새로운 솔루션을 제안하기보다는 기존의 보안 솔루션을 활용하여 보안성을 높이기 위한 네트워크 보안 아키텍처를 제시한다. SDP가 해결하고자 하는, 대표적인 기존 네트워크 보안 솔루션의 한계는 다음과 같다[8]; VPN은 확장성이 좋지 않고, 일단 사용자가 네트워크에 접속하게 되면, 사용자가 네트워크 내부에서 어떤 자원에 접속하는지에 대한 정보를 얻기 힘들며, 이를 제어할 수도 없다. 방화벽은 관리를 많이 하고자 하면 할수록 더 많은 룰이 추가되어야 하고, 로그를 실시간으로 분석하기도 힘들다. 또한 네트워크 접근을 위한 인증 및 ID관리와 접근 제어가 밀접하게 연관되어 있지 않아서 관리의 어려움이 있다. 그리고 사용자가 네트워크 접속 시 사용하는 단말의 보안 취약점으로 인한 네트워크 보안 취약점 증가 등의 문제가 있다.

SDP는 '볼 수 없는 것을 해킹할 수 없고, 볼 수 없는 것을 훔칠 수도 없다'는 사상에서 출발한 개념이다. 즉, 인증 및 접근 제어를 통해서 접근 권한이 있는 네트워크 영역에만 접속할 수 있는 정보를 부여하고, 권한이 없는 부분에 대해서는 철저히 가려져 있어서 사용자는 네트워크 상에 어떤 기기가 연결되어 있는지 등의 정보를 알 수 없다.

Fig. 2.는 SDP 구조를 보여준다. Fig. 2. 에서 볼 수 있듯이, SDP는 IH(Initiating Host)와 AH(Accepting Host)의 두 가지 호스트와 컨트롤러로 구성되어 있다. 컨트롤러는 어떤 호스트들 간 통신을 허용할지를 결정하고 네트워크 인프라를 관리한다. IH는 자신이 접속할 수 있는 AH의 리스트와 접속정보를 컨트롤러로부터 받아온다. AH는 SDP 컨트롤러 이외의 모든 호스트와의 통신을 거절하고,

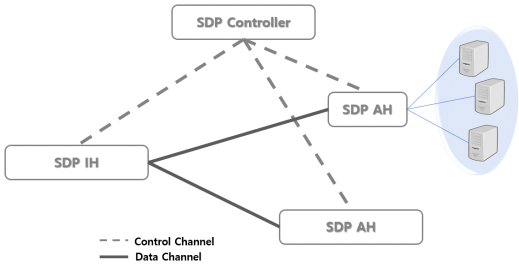


Fig. 2. Structure of SDP

컨트롤러로부터 커맨드를 받은 후, 네트워크 연결을 허락받은 IH에서 오는 패킷만을 받아들인다.

SDP에서 모든 통신은 SPA 메시지로 시작한다. SDP가 적용된 네트워크의 모든 통신 포트에 대해서 'default drop' 방화벽 룰을 적용하여 모든 메시지가 네트워크 내부에 도달되지 못하도록 막는다. 단지, SPA 메시지 수신을 위한 하나의 포트만을 열어둔다. 이 포트로 정상적인 SPA 메시지를 보낸 기기에 대해서만 TLS 통신 포트를 추가로 오픈해 주어 기기 인증 및 보안 채널을 생성할 수 있도록 한다. 이는 SDP가 적용된 네트워크를 대상으로 하는 DDoS 공격을 완화하는 데 큰 도움이 된다.

III. 제로 트러스트를 위한 SDP에서의 인증

3.1 SDP 인증 메커니즘

SDP는 기존의 IP주소 기반 네트워크 접속제어 대신 사용자와 어플리케이션 중심의 접근정책 기반 네트워크 접속제어를 가능하게 해 준다. 또한 SDP에서는 기기를 인증하고, 사용자와 기기를 모두 고려한 네트워크 접속제어를 제공한다. SDP에서 정의한 인증에는 SPA(Single Packet Authorization)를 이용한 선 검증 기능, 상호인증을 제공하는 TLS를 이용한 기기 인증 기능이 있고, 사용자 인증에 대한 구체적인 방안은 제시하고 있지 않다.

3.1.1 SPA를 이용한 선 검증

SPA는 호스트 에이전트 ID(AID, Agent ID)를 검증(validate)한 후, 검증을 통과하면 보호된 서비스에 대한 접속을 허용하는 경량 프로토콜이다. SPA를 이용한 선 검증 과정은 정상적인 SPA 메시지를 전송한 호스트에 대해서 추가적인 네트워크 연결을 허용해주기 위한 것으로, 네트워크에 접속하기 위한 최초의 메시지는 SPA이어야 한다. SPA 이외의 메시지가 도착하면 방화벽에서 차단하므로, 무작위 메시지를 전송하는 DDoS 공격을 완화할 수 있다. 또한 본격적인 네트워크 연결을 허용하기에 앞서, 간단히 '믿을 만한 호스트'의 접속 요청인지를 확인하는 과정이어야 하므로 SPA 프로토콜이 복잡해져서, SPA 프로토콜 수행에 시간이 많이 소요되거나 시스템 리소스를 많이 사용해도 안 된다. SPA 메시지 검증에 시스템 리소스 소모가 많아지면 서버에 부하가 늘어나게 되므로, SPA 메시지를 이용한 DDoS 공격 예방 효과가 감소한다. SPA 검증에 성공하면 호스트 기기를 인증하기 위한 TLS 핸드쉐이크 과정을 수행할 수 있도록 TLS 통신을 위한 포트(port)를 오픈해서 컨트롤러가 호스트의 TLS메시지를 수신할 수 있게 된다.

SPA 메시지는 호스트(IH 혹은 AH)가 SDP를 구성하는 컴퍼넌트들과 통신하기 위해서 전송하는 최초의 메시지이다. Table 1.과 Table 2.에서 볼 수 있듯이 SPA 메시지 구성은 SDP v2.0으로 가면서 더 많은 내용이 추가되고, 일부 변경된 부분도 있다. 또한, SDP v2.0 스펙이 릴리즈 되기 전, 2019년에 릴리즈 된 SDP 구조 가이드[3] 문서에도 SDP v1.0[1]과는 다른 구조의 SPA 메시지를 부록에 제시하고 있는데, [1]과 [3]의 구조가 혼합된 형태가 [2]에서 제시되고 있다. 그리고 [3]에서 SPA 메시지의 기밀성 보장을 위하여 SPA 메시지를 암호화하여 전송할 것을 제안하였고, [2]에서는 SPA 메시지 암호화를 옵션으로 제시하고 있다. 또한, SPA 메시지에 대한 무결성 보장을 위하여 HMAC이 추가되었다. SPA 메시지를 이용한 DDoS 공격 발생시, 서버의 부하를 줄이기 위하여 메시지 복호화보다 리소스가 적게 드는 HMAC 검증을 먼저 수행하여 HMAC 검증 성공시, 메시지 복호화를 통해서 추가적인 검증을 할 수 있도록 하기 위하여 HMAC 값은 암호화 하지 않고 암호화된 SPA 메시지 마지막에 추가하여 전송한다. Fig. 3.과 같은 구조가 된다.

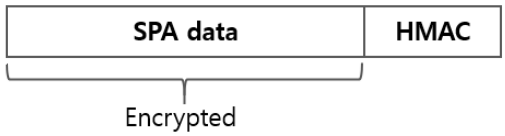


Fig. 3. Simplified SPA message structure

Table 1. SPA structure (SDP v1.0)

SPA structure (SDP v1.0)	Explanation
AID	Agent ID. Identifies an IH or AH
PW	HOTP value generated by the RFC 4226
Counter	a value used as the "look-ahead window" value when calculating HOTP. Must be shared between communicating pair

Table 2. SPA structure(SDP v2.0)

SPA structure (SDP v2.0)	Explanation
ClientID	assigned per user-device pair
Nonce	2bytes random data
Timestamp	current time
Source IP	IP address of SPA source host(IH or AH)
Message Type	optional field
Message String	optional field
HOTP	HOTP value whih is generaged by an algorithm as described by RFC 4226
HMAC	calculated over all fields above. SHA256 algorithm is recommended

Table 1.에서 볼 수 있듯이 SDP v1.0[1]에서는 'AID'를 기반으로 SPA 메시지 전송자를 확인한다. 그리고 AID는 IH 혹은 AH를 확인하기 위한 ID이므로 SPA 메시지를 전송한 기기를 검증(인증)한다.

또한 'Counter' 값과 통신 양단간 공유된 보안 파라미터 값들을 이용해서 HOTP를 검증하여 SPA 메시지에 대한 재전송 공격을 막는 동시에 정당한 보안 파라미터 값을 가진 개체임을 확인하는 과정을 통해서 기기 선 검증 과정을 수행한다.

3.1.2 TLS를 이용한 기기 인증

TLS(혹은 SSL 3.0)는 보안 전공자가 아닌 사람들도 실생활에서 접할 수 있을 정도로 보안의 대명사가 되어 가고 있다. 하지만, 대부분의 웹페이지에서 사용되는 TLS는 편리함을 위해서 충분히 안전하게 사용되지 못하고 있다. Fig. 4.에서 볼 수 있듯이 TLS 핸드쉐이크 과정에서는 상대방을 인증하는 과정과 통신 데이터 암호화에 사용할 암호 알고리즘을 결정하고 키를 공유하는 과정이 있다. 그리고 암호화 알고리즘 및 키 공유가 성공적으로 완료된 후 어플리케이션 데이터를 암호화하여 전송하게 된다.

Fig. 4.의 인증 과정은 통신하는 상대방을 인증하는 과정으로 인증 방법에 따라 세 가지 운용 모드가 있다. 통신 양단인 서버와 클라이언트 모두 인증하지 않는 모드, 클라이언트가 서버만 인증하고 서버는 클라이언트를 인증하지 않는 모드(대부분의 웹 서비스에서 사용), 그리고 서버와 클라이언트 모두 통신 상대방을 인증하는 모드가 있다.

통신 상대방을 인증하지 않고 키 교환과 암호화 알고리즘 교환만 하는 경우 가장 보안성이 낮다. 피싱서버 등을 만들어서 개인정보 등을 빼내는 공격을 막기 위하여 서버만 인증하는 운용모드를 가장 많이 사용하지만, 서버 인증 시 사용되는 인증서의 종류 및 인증 방법에 따라서 서버 인증 결과에 대한 신뢰도가 낮아지기도 하고[4, 5], 서버는 클라이언트를 인증하지 않음으로 인해서 피싱, 파밍 등 다양한 공격에 무방비로 노출되기도 한다. 따라서 SDP 스펙에서는 양방향 인증을 지원하는 TLS(mutual-TLS) 사용을 기본으로 하고 있다.

SDP에서는 컨트롤러에 등록된 기기라면 기기 인증서가 기기에 저장되어 있고, 이 인증서를 이용하여 TLS 핸드쉐이크 과정동안 해당 기기 인증에 참여한다. 즉, 인증서에 해당하는 비밀키를 소유하고 있는

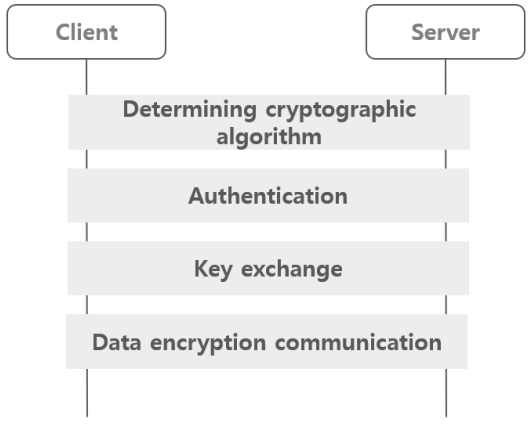


Fig. 4. Simplified TLS handshake process

기기임을 확인한다. 또한, 인증에 성공하면 DTM(Dynamic Tunnel Mode)이 생성되고, 이를 이용하여 데이터를 안전하게 전송한다.

3.1.3 사용자 인증

SDP의 기본 개념은 사용자와 어플리케이션 중심의 접근정책을 기반으로 네트워크 레벨에서 접속제어를 제공하는 것이다. 사용자를 인증하고, 사용자가 접근할 수 있는 서비스, 어플리케이션 등을 제공하는 서버의 특정 포트에 대해서 정해진 시간 동안만 접속할 수 있도록 네트워크 레벨에서(IP 주소 및 Port 번호 중심) 접속을 제어한다.

SDP v.1.0에서는 구체적인 사용자 인증 순서 및 사용자 인증 방법에 대해서 언급되지 않고 있다. 하지만 SDP v2.0의 경우, SPA 메시지에서 사용되는 ID를 ClientID로 명명하고, 이는 사용자-기기 쌍(user-device pair)에 대해서 부여된다. ClientID가 컨트롤러에 등록되어 있는지를 확인하는 과정을 통해서 사용자 확인은 가능해졌다. 그리고, SPA 성공 후 mutual-TLS를 수행하고, TLS 세션 생성에 성공하면, 컨트롤러에 전송하는 "Login Request" 메시지에 사용자 크리덴셜(credential)을 추가할 수 있다고 기술되어 있다. 즉, TLS 핸드쉐이크 성공 후 암호화 통신이 가능해지면, 해당 암호화 통신을 통해서 서비스 이용 요청 전 사용자 인증 메커니즘을 추가 할 수 있을 것이다.

3.2 제안하는 SDP 사용자 인증 메커니즘

3.1.절에서 기술하였듯이 SDP는 사용자 중심의 네트워크 접속을 제어하는 플랫폼을 정의하고 있지만, 사용자 인증 방법이나 인증과정을 구체적으로 명시하고 있지 않다. 따라서 본 논문에서는 SDP에서 사용자 인증을 적용할 수 있는 방안을 제시하고, 또한 제로 트러스트 성숙도[6]를 높이기 위한 SDP에서의 사용자 인증 방법을 제안하고자 한다. 또한, SDP v2.0의 SPA에서 사용되는 ClientID의 한계를 기술하고, 이를 개선할 방안도 제시한다

3.2.1 SDP v2.0 사용자 인증의 한계

SDP v2.0에서는 기기 온보딩(onboarding)시에 사용자를 인증하고, 사용자 인증에 성공 시 사용자-기기 쌍(user-device pair)을 위한 크리덴셜을 배포한다. 크리덴셜에는 TLS encryption key, TLS certificate, SPA encryption key, SPA HMAC key가 포함된다. 그리고 3.1.3절에서 언급했듯이 SDP v2.0에서 SPA에 들어가는 ClientID는 사용자-기기 쌍에 대해서 부여되는 ID이고, 기기 온보딩 과정에서 부여된다.

기기 온보딩 과정은 기기가 처음 네트워크에 연결될 때 한 번 이루어지는 과정이므로, 온보딩 시 사용자를 인증하면, 이후에는 SPA 메시지의 사용자-기기 쌍에 부여된 ID를 확인하는 과정을 통해서 사용자를 식별하게 된다. 사용자 인증 없이, 사용자-기기 쌍에 대해 부여된 ID 확인만으로 사용자를 식별하고, 해당 사용자에게 접근 권한이 부여된 서버 혹은 서비스에 접속할 수 있도록 허용하는 것은 심각한 보안 문제를 야기할 것이다. 또한, 다른 사람의 노트북을 빌려서 사용하는 경우 실제 사용자를 식별 할 수 없고, 회사 전산실 혹은 PC방 등 공용 PC를 이용하여 회사 내부 서버에 접속하여 회사 업무를 수행하고자 하는 경우, 사용자 확인의 문제가 발생한다. 그리고, 모든 네트워크 접속에 대해서 접속을 허용하기 전에 인증 및 인가하여야 한다는 제로 트러스트의 원칙에도 위배된다. 따라서, SDP 프로토콜 상에서 사용자가 서비스 서버에 접속을 요청할 때, 사용자 인증은 반드시 추가되어야 할 것이다. 또한, 사용자와 기기 쌍에 대해서 ClientID를 부여하고, SPA 메시지에 ClientID를 포함하여 ClientID를 기준으로 선 검증을 하는 SDP v2.0 스펙은 기존의 agentID(host에 부여된 ID)를 사용하는 SDP v1.0 스펙보다 더 많은 제약이 생기고, 이에 따라 제로 트러스트 아키텍처를 구축하는 데 어려움이 있다.

제로 트러스트 아키텍처에서는 사용자가 사용하는 기기의 신뢰 정도에 따라서 사용자가 그 순간 접속할 수 있는 서버 및 서비스의 종류가 달라진다. 사용자가 자유롭게 다양한 기기를 사용하여 네트워크에 접속할 수 있으나 사용하는 기기의 신뢰도에 따라서 접속범위가 달라져야 하는데, 사용자에 대한 검증 없이 사용자와 기기 쌍에 부여된 ClientID를 확인하고 접속 범위가 결정되는 SDP v2.0 스펙을 따른다면, 기존 온보딩 과정을 거친 기기라도 사용자가 달라질 때마다 새로운 ClientID를 부여받기 위해서 온보딩 과정을 거쳐야 하고, 사용자-기기 쌍에 대해서 컨트롤러에 등록되어 있지 않다면 접속조차 할 수 없을 것이다.

또한, SDP v2.0의 온보딩 과정에서 일부 오류가 있다. 사용자 인증에 성공하면 사용자-기기 쌍에 대해서 크리덴셜이 발급되는데, 크리덴셜의 내용에 "TLS Encryption Key"가 있다. 이는 TLS 통신 채널 암호화에 사용되는 어플리케이션 데이터 암호화용 키를 의미한다. 하지만, 이 키는 TLS 서버와 클라이언트간 핸드쉐이크를 수행하는 도중에 인증 및 키 공유 과정을 통해서 생성되고 공유되는 값으로, 온보딩 과정을 통해서 배포해야 할 값이 아니다.

그리고, 온보딩 과정에서는 외부 Identity provider(IdP)를 이용하여 사용자 인증을 한 후, 인증에 성공하면 SDP 프로토콜 내의 TLS에서 사용할 기기 인증서를 크리덴셜에 넣어서 사용자가 실행한 클라이언트 어플리케이션에 전달하게 된다. 사용자 인증만 성공하면, 기기에 대한 정보 확인이나 기기 정보 수집 절차 없이 기기의 인증서를 크리덴셜에 넣어서 전송해주고, 기기 인증서를 전송해주는 주체도 공인 CA 혹은 회사 내부 사설 CA가 아니라, 외부의 IdP가 된다. 사용자와 기기로 구성된 클라이언트의 인증 결과에 따라 해당 클라이언트가 접속할 수 있는 네트워크상에서의 접속범위가 결정되는 SDP의 기본 개념을 고려하면 사용자와 기기의 인증이 보안의 시작인데, 기기 인증의 과정 없이 기기의 인증서를 전달하고, 이 인증서를 기기 인증에 사용하게 되는 것은 또 다른 보안의 홀이 될 것으로 판단된다. 이 부분은 SDP 스펙에서 보완되어야 한다.

본 논문에서는 이러한 SDP v2.0 스펙에서 사용자 인증의 한계를 해결하는 방안을 제시한다.

3.2.2 Login Request 메시지를 이용한 사용자 인증

본 절에서는 SDP 스펙의 Login Request 메시지를 사용자 인증에 이용하는 구체적인 방법을 제시함으로써, SDP 프로토콜을 따르면서 사용자 인증이 가능하게 하여 SDP를 제로 트러스트 아키텍처에 적용할 수 있는 방안을 제시한다. SDP 스펙의 Login Request 메시지는 Fig. 5.에서 볼 수 있듯이 메시지 헤더만 존재한다. 따라서 Login Request 메시지에 사용자 인증정보를 포함하여 컨트롤러에 전송함으로써 사용자 인증이 가능해진다. ID/PW를 이용한 인증을 한다면, Login Request 메시지 헤더 다음에 사용자의 ID와 패스워드(혹은 hashed 패스워드)를 추가하여 전송하면, 컨트롤러에서는 이를 통해서 사용자 인증이 가능하다. 또한, mutual-TLS를

0x00	No command-specific data

Fig. 5. Login Request message structure

0x01	Status Code	IH/AH Session ID

Fig. 6. Login Response message structure

통해서 이미 IH-컨트롤러 간 혹은 AH와 컨트롤러 간 암호화 통신이 가능한 상태이므로 Login Request 메시지는 암호화된 TLS 채널을 통해 컨트롤러에 전송되어 사용자 ID, 패스워드에 대한 기밀성 및 무결성이 보장된다. 또한, OTP를 이용한 사용자 인증이 필요한 경우, Login Request 메시지 전송 전, 사용자로부터 OTP 값을 입력받아서 이 값을 Login Request 헤더 및 사용자 ID와 함께 전송함으로써 사용자 인증이 가능해질 것이다.

지문인식, 얼굴인식, 홍채인식 등도 OTP와 동일하게 사용자 ID와 사용자의 생체정보를 전달함으로써 사용자 인증을 할 수 있으며, 최근 많이 사용되고 있는 FIDO 인증 기술 또한 적용이 가능할 것이다. 또한, 인증서를 이용한 사용자 인증에도 동일한 방법을 적용할 수 있다. 대부분의 인증 방법이 단방향 인증 메시지 전송만으로 해결할 수 있으므로 Login Request 메시지를 활용하여 SDP 스펙을 크게 벗어나지 않으면서도 효과적으로 사용자 인증이 가능해진다. 다만, Challenge-Response 형태의 인증 메커니즘을 적용하고자 한다면 Login Request와 Login Response 사이에 추가의 메시지를 정의할 필요가 있다.

Fig. 7.에서 볼 수 있듯이 Login Request 메시지를 수신한 컨트롤러는 IH/AH에 Login Response 메시지를 전송한다. Login Response 메시지는 Fig. 6.과 같이 구성된다. Login Response 메시지에서 "Status Code"의 값은 SDP 스펙에서 정해져 있지 않고, 구현하기에 따라 달라질 수 있다고 되어 있다. 따라서 본 논문에서는 Login Request 메시지에 사용자 인증정보를 전송하였으므로, Login Response 메시지의 'Status Code"에 인증 결과를 넣고, 인증 성공 시에만 "IH/AH Session ID"필드에 의미 있는 값이 입력될 것이다. "Status Code"는 16bit 값이어야 하므로 Table 3.에서 볼 수 있듯이, 인증 성공 시 헥사

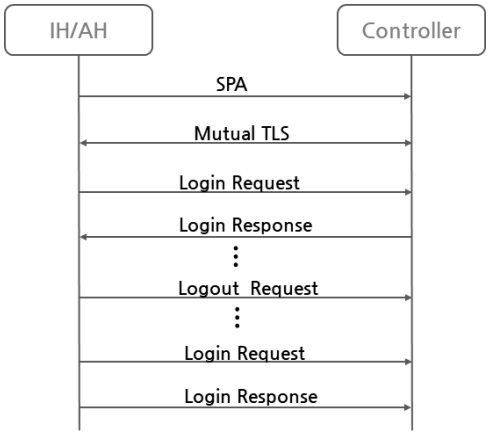


Fig. 7. User login process

Table 3. Example of Status Code

user authentication result	Status Code (16bits)
Success	0x01 0x01
Fail	0x01 0x02

값으로 '0x01 0x01', 인증 실패 시 헥사 값으로 '0x01 0x02' 등의 값을 할당할 수 있다.

3.2.3 제로 트러스트 성숙도를 높이기 위한 사용자 인증 방법

NIST SP 800-207 "Zero Trust Architecture" [7]에서는 Zero Trust 구현에 적합한 구조의 하나로 SDP를 제시하고 있다. 2장에서 언급하였듯이, 제로 트러스트는 구현 방법에 따라 성숙도의 레벨이 달라진다. 특히 인증과 관련된 Identity 파트의 제로 트러스트 성숙도 레벨을 높이기 위해서는 MFA(Multi Factor Authentication) 및 지속적인 인증도 필요하다. 따라서 3.2.2절에서 제안한 사용자 인증 방법과, 기존 SDP 스펙에서 사용하는 TLS를 이용한 기기인증 방법(즉, 인증서를 이용한 기기 인증 방법)을 모두 적용함으로써 MFA을 만족할 수 있다. 다만, 사용자 인증에서는 인증서를 이용한 인증을 적용하면 MFA를 만족할 수 없으므로, 다른 인증 방법을 적용하여야 한다. 그리고 사용자 인증 자체에서 두 가지 이상의 인증 방법을 적용하여 MFA을 만족시킬 수도 있다. Login Request 메시지를 이용하여 사용자 인증 정보 전송시, 두 가지 이상의 인증 정보(예를들어, ID/PW와 지문정보)를 포함하여 전송하는 등의 방법을 이용할 수 있다. 또는 challenge-response 기반의 사용자 인증 메커니즘을 Login Request 메시지와 Login Response 메시지 사이에 추가하여 사용자를 인증하는 방법도 가능할 것이다.

제로 트러스트 성숙도 최상위를 달성하기 위해서는 클라이언트가 서비스를 이용하는 동안 지속적으로 클라이언트를 확인하고, 클라이언트의 행동 속성을 분석하여 접속 가능 영역을 제어하는 과정이 필요하다. 서비스를 이용중인 클라이언트에 대한 지속적인 확인 및 행동 속성 수집은 EDR(Endpoint Detection and Response) 솔루션과 연동하여 얻을 수 있고, EDR 솔루션에서 받은 결과를 이용하여 컨트롤러에서 클라이언트의 접속 범위를 실시간으로 결정하게 된다. 이 부분은 컨트롤러에서 접속제어 결정을 위해서 사용하는 신뢰 알고리즘(Trust Algorithm)의 영역이므로 본 논문에서는 자세히 다루지 않는다.

3.2.4 Shared secret 공유를 위한 온보딩 과정 제안

본 절에서는 SDP v2.0 스펙에 기술되어있는 온보딩 과정에 오류가 있으므로 이를 보완하기 위하여 온보딩 과정에서 SPA 검증에 필요한 보안정보를 공유하는 방법을 제안한다.

Fig. 8.은 본 논문에서 제안하는 온보딩 과정 및 온보딩 과정에서 shared secret을 공유하는 방법을 보여준다. 제안하는 방법에서 기기 인증서는 기기 생산 공장에서 기기에 대한 공인 인증서를 저장하여 출시한 상태를 가정한다. 또는 SDP를 구축하여 사용하는 회사 내에서, 회사 자산관리 서버에 등록된 기기에 대해 회사 사설 인증서(혹은 공인 인증서)를 발급하여 기기에 저장해 둔 상태를 가정한다. 그리고, 회사에서 SDP를 적용하는 경우, 컨트롤러에 회사에서 관리하는 기기들의 정보가 등록된 상태에서 시작하게 된다. Fig. 8.에서 제안한 온보딩 프로세스는 다음과 같다.

1. SDP가 적용된 시스템에 접속하여 서비스를 받고자 하는 클라이언트(사용자 및 기기)가 SDP Client App을 설치한다.
2. SDP 사용자가 SDP Client 앱을 실행한다.

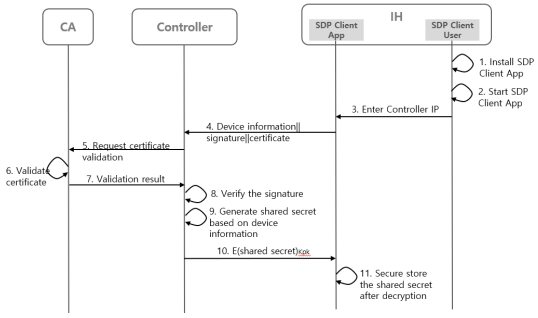


Fig. 8. Proposed onboarding process

3. SDP 사용자는 해당 SDP 시스템의 컨트롤러 IP를 입력한다.
4. SDP 클라이언트 기기는 해당 기기의 정보(agentID, 기기IP주소 등)와 기기 정보에 대한 해쉬값을 입력 메시지로 하여 생성된 서명값, 그리고 기기의 인증서를 컨트롤러에 전송한다. 이 과정은 IH에 설치된 SDP Client App 내부에서 이루어지는 과정이다.
5. 컨트롤러는 수신한 기기의 인증서에 대한 유효성 검증을 CA에 요청한다.
6. CA는 인증서의 유효성을 검증한다.
7. 인증서 유효성 검증 결과를 컨트롤러에 전달한다.
8. 유효한 인증서인 경우, 컨트롤러는 해당 기기가 보내온 기기정보와 인증서 내의 공개키를 이용하여 서명을 검증한다.

8-1. 유효하지 않은 인증서인 경우, 컨트롤러는 해당 IH에게 SDP에 참여할 수 없음을 알린다.

9. 인증서 유효성 검증에 이어 서명 검증에도 성공하게 되면, 컨트롤러는 기기가 보내온 기기 정보를 이용하여 보안 정보들(shared secret)을 생성한다. 생성된 보안 정보에는 SPA 검증에 필요한 SPA 암호화 키 및 HOTP 생성 키, HMAC 키 등이 있다.
10. 생성한 보안 정보는 해당 IH의 공개키로 암호화하여 IH에 전송한다.
11. IH는 비밀키로 복호화하여, 앞으로 사용하게 될 자신의 보안 정보를 안전하게 저장한다.

이 논문에서 제안한 온보딩 과정은 사용자 인증과 기기 인증을 분리하여 고려함으로써 기존 SDP에서 사용자-기기 쌍(pair)으로 묶어서 접근제어 정책을 부여하는 것보다 사용자 편의성을 높일 수 있고, 사용자가 현재 사용하는 기기의 보안 상태에 따라서 접근할 수 있는 서비스의 종류 등이 달라지므로 높은 보안성을 확보할 수 있다. 즉, 사용자가 회사에 등록된 기기가 없는 외부에서 급하게 회사 메일을 확인하고자 하는 경우, 다른 사람의 PC 등을 빌려서 회사 메일서버에만 접속할 수 있도록 하는 등의 서비스가 가능해 진다. 또한 SDP에서 보안의 시작점이라 할 수 있는 사용자의 보안 정보들을 암호화 해서 IH, AH에 전송하기 때문에 보안성을 더욱 높일 수 있다. 기존 SDP 온보딩 과정에서는 보안 정보들을 크리덴셜에 담아서 전송하는데 크리덴셜의 암호화 여부에 대해서는 언급이 없다. 만약 크리덴셜을 암호화하여 전송한다면 암호화 키에 대한 공유 과정이 추가되어야 하고, 크리덴셜을 암호화하지 않고 전송한다면 보안정보의 유출 위험이 있다.

IV. ECC 암호 구현 및 성능평가

본 논문에서 제안한 기기 온보딩 과정에서는 보안 정보를 공유하기 위해서 기기 인증서를 이용하여 기기를 인증하는 과정이 추가되었다. 기기 온보딩 과정은 기기가 회사내 네트워크에 처음 연결될 때 한 번 하는 과정이지만, 새로운 인증과정이 추가되므로 추가 시간이 소요된다. 본 장에서는 새로 추가된 인증과정에서 가장 많은 시간이 걸리는 서명 생성 및 검증의 고속 구현 방법에 관하여 기술하고, 서명 생성 및 검증에 소요되는 시간의 측정 결과를 기술한다.

4.1 성능향상을 위한 암호 알고리즘 구현 기법

SDP는 클라우드 보안에 적용하기 위한 솔루션이지만, IP기반 네트워킹이 가능한 모든 기기에 대해서 적용할 수 있다. 따라서 일반 네트워크 환경에서 IP 기반 통신이 가능한 모든 시스템, 서비스 등에 적용이 가능하다. 컴퓨팅 시스템의 성능이 좋아지면서 보안성을 확보하기 위한 RSA 인증서의 키 길이도 길어지고 있어서, 상대적으로 짧은 키 길이를 가진 ECC 기반의 인증서를 이용한 인증 서비스가 늘고 있다. 따라서 본 논문에서는 ECC 인증서를 이용한 서명 생성 및 검증을 고려한다. 또한 KCMVP 인증 알고리즘 커브의 하나인 ECC secp256r1 커브에서의 서명생성/검증 속도를 측정한다.

ECC 암호 구현 성능은 좌표계 선택, 타원곡선 연산 최적화, 유한체 연산 최적화에 많은 영향을 받

는다. 좌표계의 경우 일반적으로 아핀 좌표계를 고려한다. 아핀 좌표계에서의 타원 곡선 덧셈 연산은 수식 (1)과 (2)와 같다.

$$x_3 = (\frac{y_2 - y_1}{x_2 - x_1})^2 - x_1 - x_2 \quad (1)$$

$$y_3 = (\frac{y_2 - y_1}{x_2 - x_1})(x_1 - x_3) - y_1 \quad (2)$$

또한 아핀 좌표계에서 타원 곡선의 더블링(doubling) 연산은 수식 (3) 및 (4)와 같다.

$$x_3 = (\frac{3x_1^2 + a}{2y_1})^2 - 2x_1 \quad (3)$$

$$y_3 = (\frac{3x_1^2 + a}{2y_1})(x_1 - x_3) - y_1 \quad (4)$$

수식 (1)~(4)에서 볼 수 있듯이, 덧셈 및 더블링 연산에 필요한 연산량은 Table 4.와 같다. Table 4.에서 볼 수 있듯이 덧셈과 더블링 연산에 각각 1회의 역원(inversion)연산이 필요하다. 하지만 곱셈(multiplication) 연산에 비해서 상대적으로 많은 연산 시간이 필요하다. 따라서 역원 연산 없이 타원곡선 덧셈연산을 수행하기 위하여 더블링 연산에서는 야코비안 좌표계를 사용하고, 덧셈연산에서는 야코비안 좌표계와 아핀 좌표계를 동시에 사용하는 혼합(mixed) 좌표계를 사용하여 구현했다.

Table 4. Amount of required computation

	inversion	multiplication	squaring
addition	1	1	1
doubling	1	2	2

4.2 서명 생성 및 검증 성능

4.1.절에서 구현한 ECC커브로 ECDSA 서명 생성 및 검증 기능을 구현하고 성능을 측정해 보았다. 성능 측정을 위해서 윈도우즈 노트북에 WSL2 ubuntu를 설치하여 테스트 하였고, 사용한 노트북 사양은 Table 5.에 기술한다. 서명 생성 및 검증에

Table 5. System specifications used to measure performance

CPU	12th Gen Intel(R) Core(TM) i7-1260P 2.10GHz
Memory(RAM)	16.0 GB
OS	5.10.16.3-microsoft-standard -WSL2(Ubuntu)

걸리는 시간 측정의 정확도를 높이기 위하여 1000회 연속 서명 생성 및 검증에 소요되는 시간을 측정하였다. 시간 측정에는 서명생성 직전과 직후에 clock( ) 함수를 호출하여 클럭 수를 구하고, 두 클럭 수의 차를 계산한 후, 이를 초당 클럭 수로 나누어 구하였다.

Fig. 9.에서는 1000회의 연속된 서명 생성에 소요되는 시간을 측정하고, 이 시간을 1000으로 나누어 1회 서명 생성에 소요되는 시간을 계산한 것이다. 그리고, 이 과정을 20회 반복하여 그래프로 나타내고, 서명생성에 걸리는 시간의 평균을 구하여 표현하였다. Fig. 9.에서 처음 1000회의 연속된 서명 생성 소요 시간은 0.457초 이므로 1회 서명 생성에 평균 0.457ms가 소요됨을 알 수 있다. 그리고 두 번째 연속된 1000회의 서명 생성에는 0.48초가 소요되었으므로, 평균 0.48ms의 서명 생성 시간을 얻었다. 이를 반복하여 구한 평균 서명 생성 소요 시간은 0.44985ms이다.

Fig. 10.에서는 서명 생성에 걸리는 시간을 구할 때와 동일한 방법을 사용하여 서명 검증에 걸리는 시간을 측정하여 구하였다. 1000회의 연속된 서명 검증에 2.507초가 소요되었으므로, 평균 2.507ms의 서명 검증 소요 시간을 구하였다. 또한 두 번째 연속된 1000회의 서명 검증에 2.484초가 소요되었으므로, 1회의 서명 검증에 평균 2.484ms가 소요되었음을 알 수 있다. 이러한 과정을 20회 반복하여

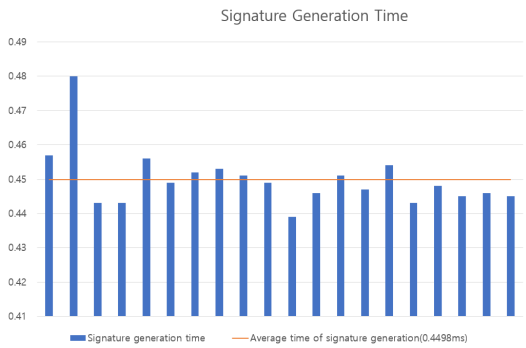


Fig. 9. Time required for signature generation

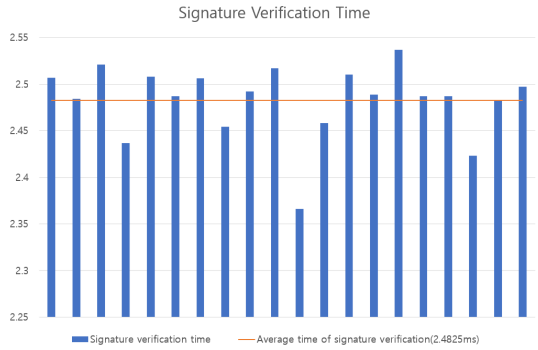


Fig. 10. Time required `for signature verification

Table 6. Average Time required for signature generation and verification

	Average Time(ms)
signature generation	0.44985
signature verification	2.4825

평균 2.4825ms의 서명검증 시간이 소요됨을 확인하였다.

V. 결 론

본 논문에서는 사용자의 지리적 위치와 관계없이 동일한 사용자는 동일한 과정을 통해서 회사 네트워크에 접근할 수 있어야 하고, 사용자 중심의 접근제어 정책을 네트워크 레벨에서 실행하는 SDP의 기본 개념을 거스르는 SDP v2.0 스펙의 SPA 메시지 구조를 보완하는 방법을 제시하였다. 또한 SDP 스펙에서 사용자 인증을 추가하는 방안을 제시하고, SPA 메시지 검증에 필요한 보안정보를 공유하는 온보딩 과정을 제안하였다. 제안한 온보딩 과정에서 기기 인증에 ECC 서명생성 및 검증을 사용하고, 이 연산에 가장 많은 시간이 소요되므로 ECC를 최적화 구현한 방법을 제시하고, ECDSA 서명 생성 및 검증에 소요되는 시간을 측정하였다. 서명 생성 및 검증에 걸리는 시간을 합해도 3ms 미만이고, 이는 무시할 만한 수준이므로 제안한 온보딩 과정으로 인한 시간 오버헤드는 거의 없다고 할 수 있다. 본 논문에서는 노트북에서의 서명 생성 및 검증에 소요되는 시간을 측정하였으나, 실제 SDP 온보딩 과정은 서버급 PC와 클라이언트 기기(노트북 등) 사이에서 일어나므로 온보딩에 필요한 시간은 더욱 짧아질 것이다.

References

[1] Brent Bilger, Alan Boehme, et al., "SDP Specification 1.0," CSA, April, 2014

[2] Jason Garbis, Juanita Koilpillai, "Software Defined Perimeter(SDP) Specification v2.0," CSA, Mar. 2022

[3] Jason Garbis, Juanita Koilpillai, "Software-Defined Perimeter Architecture Guide," CSA, 2019.

[4] Yoon Hong, "A method for detecing phishing/pharming attacks using SSL/TLS protocols," Journal of Defense and Security, vol.1, no.2, pp. 118-134, Dec. 2019.

[5] Ji Yeon Yang, Hyoung Kee Choi, "A Study on the Certificate Verification Testing in SSL/TLS Implementations," KICS Summer Conference 2017.pp.138-139, 2017.

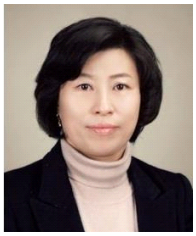
[6] CISA Cybersecurity division, "Zero trust maturity level," Pre-decisional Draft, version1.0, June 2021.

[7] Scott Rose, O.Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST.SP.800-207, Aug. 2020.

[8] Juanita Koilpillai, "Software Defined Perimeter-A New Paradigm for Securing Digital Infrastructures/ Systems," GTSC 2017, Aug.

〈저자소개〉

이 윤 경 (Yun-kyung Lee) 종신회원
2001년 2월: 포항공과대학교 전자전기공학과 석사
2018년 2월: KAIST 전산학과 박사
2001년 1월~현재: 한국전자통신연구원 정보보호연구본부 책임연구원
〈관심분야〉 IoT 보안, 모바일보안, 네트워크 보안, 인증(Authentication)



김 정 녀 (Jeong-Nyeo Kim) 종신회원
1987년 2월: 전남대학교 전산통계학과 졸업
2000년 2월: 충남대학교 컴퓨터공학과 석사, 박사
1988년 ~ 현재: 한국전자통신연구원 정보보호연구본부 책임연구원
1996년: OSF/RI 공동연구 파견(미국)
2005년: Univ. of California, Irvine Post-Doc.
2015년~현재: 과학기술연합대학원대학교(UST) ICT 공학과 (정보보호전공) 전임교수
〈관심분야〉 IoT 보안, 모바일 보안, 시스템·네트워크 보안, 보안 OS 등