클라우드 네이티브 환경에서 서비스 메시를 활용한 암호화 민첩성 분석

차호현 [1], 남재현 [2]
[1] 단국대학교 모바일시스템공학과 학부생
[2] 단국대학교 컴퓨터공학과 교수

outcider112@dankook.ac.kr, namjh@dankook.ac.kr

Analyzing Cryptographic Agility with Service Mesh on Cloud-Native Environment

Hohyeon Cha[1], Jaehyun Nam[2]
[1]Dept. of Mobile System Engineering, Dankook University (Undergraduate Student)
[2]Dept. of Computer Engineering, Dankook University (Professor)

요 약

최근 마이크로서비스 아키텍처가 널리 채택되면서 다양한 애플리케이션들이 클라우드 네이티브 환경에서 운영되기 시작하였다. 그리고 이러한 서비스의 보안을 강화하기 위해 서비스 메시를 활용하여 클라우드 서비스 간의 암호화 통신을 구현하였다. 하지만, 양자 컴퓨팅의 등장과 함께 기존 암호화 방식이 취약해질 수 있다는 가능성이 제시되었다. 이에, 클라우드 네이티브 환경에서도 암호화 기술이 빠르게 변화할 수 있는 유연성, 즉 암호화 민첩성(Cryptography Agility)의 중요성이 강조되고 시작하였다. 본 연구에서는 서비스 메시 환경에서 암호화 민첩성을 실현하는데 있어 현 서비스 메시 솔루션들이 직면한 구조적 한계를 분석하고 이에 대한 해결 방안을 제시하고자 한다. 특히, 대표적인 서비스 메시 솔루션인 Istio 에 초점을 맞추어 암호화 민첩성을 향상시키기 위한 개선 방향을 제안하고자 한다.

1. 서론

마이크로서비스 아키텍처와 서비스 메시는 클라우드 네이티브 애플리케이션의 배포와 운영관점에서 높은 편의성을 제공하는 등의 혁신을 가져왔으며, 이러한 장점으로 클라우드 네이티브 환경에서 널리 채택되고 있다. 또한, 서비스 메시는 애플리케이션의 로직과 네트워킹 로직을 분리함으로써, 별도의 프록시를 통해 애플리케이션의 수정 없이 서비스 간 암호화 통신을 지원하고, 이를 통해 전반적인 보안성을 강화하는 기능을 제공하고 있다.

하지만, 양자 컴퓨팅의 등장과 함께 기존 암호화 방식의 취약성이 대두되기 시작하였으며, 이를 보완하기 위한 암호화 민첩성이라는 중요해지기 시작하였다. 암호화 민첩성이란 대규모 시스템 업데이트 없이도 빠르게 암호화 방식을 변경할 수 있는 것을 의미한다. 그리고, 클라우드 네이티브 환경에서는 서비스 메시를 활용하여 이러한 암호화 민첩성을 실현하려는 움직임이 나타나고 있다.

대표적인 서비스 메시 솔루션으로 Istio 가 있으며, 많은 클라우드 네이티브 서비스들은 현재 Istio 의 프록시를 통해 서비스 간 암호화 통신을 구현하고 있다. 하지만, 이러한 서비스 메시 환경에서 암호화 민첩성을 구현하고자 할 때 여러 구조적 한계가 존재하였다. 예를 들어, 기존의 암호화 방식을 이용한 통신 암호화에만 초점을 맞추고 있다 보니, 암호화 라이브러리에 대한 의존성이 높고 암호화 방식을 동적으로 정의하는데 있어 한계점을 보였다.

따라서, 본 연구에서는 암호화 민첩성 적용에 하고자 할 때 야기될 수 있는 서비스 메시 환경의 구조적 한계를 분석하고, 서비스 메시 환경이 암호화 민첩성의 특성을 만족할 수 있도록 하기 위한 해결 방안을 제시한다.

2. 클라우드 네이티브 환경과 서비스 메시

최근 클라우드 네이티브 환경에서는 단일 형태로 동작하던 서비스를 작은 단위로 분리하여 마이크로서비스 형태로 구성하고 네트워킹을 통해 연결함으로써 전반적인 서비스의 확장성을 높였다. 하지만, 이러한 마이크로서비스의 느슨한 결합은 다시 인프라의 관리와 운영을 복잡하게 만들었으며, 이를 해결하기 위해 서비스 메시라는 개념이 도입되었다.

서비스 메시에서는 각각의 마이크로서비스(컨테이너)에 사이드카 방식으로 서비스 간 네트워킹을 위한 프록

시를 삽입, 서비스 간의 연결을 관리한다. 특히, 이 프록시를 통해 서비스 간 네트워크 통신 암호화(상호 TLS)를 수행함으로써 애플리케이션 레벨에서 직접적으로 네트워크 암호화 기능을 구현하지 않아도 된다.

3. 암호화 민첩성 (Cryptography Agility)

양자 컴퓨팅의 출현으로 인하여 기존 암호화 기법들이 무력화될 수 있다는 위험과 이로 인해 네트워크 보안 체계가 취약해질 수 있음은 클라우드 네이티브 환경을 비롯하여 널리 영향을 미치게 되었다. 이에, 기존 암호화 방식에서 새로운 암호화 시스템으로의 신속한 전환, 즉 암호화 민첩성(Cryptography Agility)이 중요한 개념으로 부상하게 되었다. 여기서 암호화 민첩성이란 시스템이 한 암호화 알고리즘에서 다른 알고리즘으로 유연하고 확장 가능하며 동적으로 전환할 수 있는 능력을 의미한다.

그리고 미국 국립표준기술연구소(NIST)에서는 암호화 민첩성의 세 가지 특징으로 동적 선택성, 확장성, 폐기성을 언급하였으며, 이를 바탕으로 클라우드 네이티브 환경에서 암호화 민첩성을 실현하기 위해서는 크게 두 가지 주요 조건을 충족되어야 한다. 첫째, 네트워크에 대한 투명한 가시성이 확보되어야 하며, 사용 중인 암호화 알고리즘을 명확히 인식해야 한다. 둘째, 암호화 설정의 관리가 자동화되어야 한다. 이는 인증서 만료 또는 암호화 알고리즘의 취약점이 발견되었을 경우 시스템이 자동으로 인증서를 갱신하거나 취약한 알고리즘을 폐기할 수 있도록 하기 위함이다.

4. 서비스 메시와 암호화 민첩성

본 연구에서는 클라우드 네이티브 환경에서 암호화 민첩성을 달성하기 위해 우선 대표적인 서비스 메시 솔루션인 Istio 의 암호화 과정을 살펴보았다. 그리고 분석 결과, 크게 두 가지의 한계점을 확인하였다.

암호화 라이브러리의 한계. Istio 의 프록시(Envoy)의 TLS 옵션에 대한 문서에 따르면, Envoy 는 TLS 1.3 프로토콜에 대해서 암호화 스위트 지정이 불가능하다. 이는 Istio 가 사용하는 암호화 라이브러리인 BoringSSL 이 TLS 1.3 프로토콜을 처리할 때, 암호화 스위트를 지정할 수 없도록 API 를 제공하고 있기 때문이다.[2]

암호화 알고리즘의 정적 구현. Istio 는 서비스 간 통신을 위해 mTLS 를 사용한다. 이를 위해, Istiod (Istio 의 제어평면) 내 Citadel (인증서 서명 모듈)이 각각의 프록시가 사용할 인증서를 발급한다. Istio 는 mTLS 인증서를 발급할 때 특정한 서명 알고리즘(타원 곡선 알고리즘)을 사용하도록 정적으로 구현되어 있다. 결국, 이러한 방식은 암호화 민첩성을 위한 동적 선택성과 확장성을 저해한다.

5. 암호화 민첩성 반영을 위한 서비스 메시 구조

(1) 명시적 암호화 정책 설정. 서비스 메시 환경에서 통신 암호화에 사용될 Cipher Suites 를 MeshConfig 같은 설정을 통해 정의함으로써 암호화 가시성을 확보해야 한다. 이를 통해 프록시 간의 (m)TLS 통신이나 다른 서비스와의 통신에서 사용할 암호화 방식을 명시적으로 결정할 수 있어야 한다.

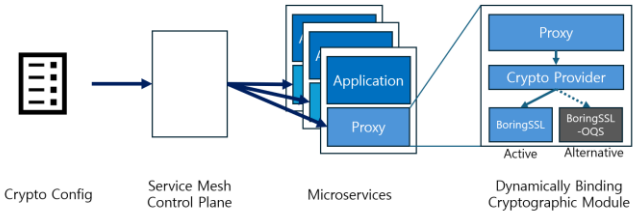

그림 1. 서비스 메시 환경에서의 암호화 민첩성 구현

(2) 암호화 알고리즘의 동적 적용. 서비스 메시에서 사용되는 암호화 알고리즘은 동적으로 적용될 수 있어야 한다. 예를 들어, Istio 의 한계점에서 언급하였듯이, 인증서 서명 알고리즘을 정적인 형태가 아닌 MeshConfig 같은 설정을 통해 지정할 수 있어야 한다. 또한, Cluster, Namespace, Label 등을 기반으로 서비스 간 mTLS 통신을 위한 암호화 알고리즘을 단일 클러스터 내에서도 다양하게 지정할 수 있어야 한다.

(3) 암호화 모듈의 교체 가능성. 새로운 암호화 알고리즘 적용이나 기존 암호화 라이브러리의 취약점에 대응하기 위해, 암호화 모듈(라이브러리)은 교체가 가능해야 한다. 즉, 특정 암호화 라이브러리에 대한 의존성을 없애고 사용자가 필요에 따라 원하는 암호화 모듈과 알고리즘을 선택할 수 있어야 한다. 이는 암호화 라이브러리들에 대한 추상화 계층인 Crypto Provider 를 통해 애플리케이션과 암호화 라이브러리 간의 의존성을 해결할 수 있다.

6. 결론

본 연구에서는 서비스 메시 환경에서 암호화 민첩성을 높이기 위해 주요 서비스 메시 솔루션인 Istio 를 분석하였다. 그 결과, 현재 Istio 의 암호화 라이브러리와 설정 방식에 한계가 있음을 확인하였다. 이를 해결하기 위해, 세 가지 구조적 접근 방식을 제안하였으며, 이러한 접근 방식은 서비스 메시 환경에서의 암호화 민첩성 향상을 위한 중요한 기반으로 작용할 것으로 기대한다.

Acknowledgement

이 성과는 2024 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임. (RS-2023-00212738)

참고문헌

[1] Anne Frances Johnson and Lynette I. Millett, Cryptographic Agility and Interoperability: Proceedings of a Workshop, Washington, D.C., The National Academies Press, pp. 19-20.

[2] BoringSSL, boringssl/include/openssl/ssl.h, Accessed April 24, 2024, https://boringssl.googlesource.com/boringssl/+/refs/heads/master/include/openssl/ssl.h