KIPS Trans. Comp. and Comm. Sys.
Vol.11, No.9 pp.305~316
ISSN: 2287-5891 (Print), ISSN 2734-049X (Online)

https://doi.org/10.3745/KTCCS.2022.11.9.305

Efficient Privacy-Preserving Duplicate Elimination in Edge Computing Environment Based on Trusted Execution Environment

Dongyoung Koo†

ABSTRACT

With the flood of digital data owing to the Internet of Things and big data, cloud service providers that process and store vast amount of data from multiple users can apply duplicate data elimination technique for efficient data management. The user experience can be improved as the notion of edge computing paradigm is introduced as an extension of the cloud computing to improve problems such as network congestion to a central cloud server and reduced computational efficiency. However, the addition of a new edge device that is not entirely reliable in the edge computing may cause increase in the computational complexity for additional cryptographic operations to preserve data privacy in duplicate identification and elimination process. In this paper, we propose an efficiency-improved duplicate data elimination protocol while preserving data privacy with an optimized user-edge-cloud communication framework by utilizing a trusted execution environment. Direct sharing of secret information between the user and the central cloud server can minimize the computational complexity in edge devices and enables the use of efficient encryption algorithms at the side of cloud service providers. Users also improve the user experience by offloading data to edge devices, enabling duplicate elimination and independent activity. Through experiments, efficiency of the proposed scheme has been analyzed such as up to 78x improvements in computation during data outsourcing process compared to the previous study which does not exploit trusted execution environment in edge computing architecture.

Keywords : Edge Computing, Privacy, Efficiency, Duplicate Elimination, Trusted Execution Environment

신뢰실행환경기반 엣지컴퓨팅 환경에서의 암호문에 대한 효율적 프라이버시 보존 데이터 중복제거

구 동 영†

요 약

사물인터넷 및 빅데이터 등 디지털 데이터의 범람으로, 다수 사용자로부터 방대한 데이터를 처리 및 보관하는 클라우드 서비스 제공자는 효율적 데이터 관리를 위한 데이터 중복제거를 적용할 수 있다. 중앙 클라우드 서버로의 네트워크 혼잡 및 연산 효율성 저하 등의 문제를 개선하기 위한 클라우드의 확장으로 엣지 컴퓨팅 개념이 도입되면서 사용자 경험을 개선할 수 있으나, 전적으로 신뢰할 수 없는 새로운 엣지 디바이스의 추가로 인하여 프라이버시 보존 데이터 중복제거를 위한 암호학적 연산 복잡도의 증가를 야기할 수 있다. 제안 기법에서는 신뢰실행환경을 활용함으로써 사용자-엣지-클라우드 간 최적화된 통신 구조에서 프라이버시 보존 데이터 중복제거의 효율성 개선 방안을 제시한다. 사용자와 클라우드 사이에서의 비밀정보 공유를 통하여 엣지 디바이스에서의 연산 복잡도를 최소화하고, 클라우드 서비스 제공자의 효율적 암호화 알고리즘 사용을 가능하게 한다. 또한, 사용자는 엣지 디바이스에 데이터를 오프로딩함으로써 데이터 중복제거와 독립적인 활동을 가능하게 하여 사용자 경험을 개선한다. 실험을 통하여 제안 기법이 데이터 프라이버시 보존 중복제거 과정에서 엣지-클라우드 통신 효율성 향상, 엣지 연산 효율성 향상 등 성능 개선 효과가 있음을 확인한다.

키워드 : 엣지 컴퓨팅, 프라이버시, 효율성, 중복제거, 신뢰실행환경

1. 서 론

인터넷의 보편화 및 사물인터넷의 대중화, 멀티미디어 등 빅데이터의 증가 추세에 따라, 개별 사용자가 저장하고 관리하는 데이터의 양은 지속적으로 증가하고 있다. Statista에 따르면, 전 세계에서 생성 및 캡처, 복사, 소비된 데이터의 양은 2010년 2 제타 바이트(zetabytes, ZB; 1ZB = 2^{70}B)에서 2021년에는 79 ZB로 증가하였으며, 2025년에는 181 ZB에 이를 것으로 전망하였다[1]. 증가하는 데이터에 대한 관리 부담을 줄이고 효율성을 극대화하기 위하여 Memopal[2], 드롭박스[3], 구글 드라이브[4] 등 클라우드 스토리지 기반 서

※ 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2021R1F1A1064256).
† 종신회원 : 한성대학교 기계전자공학부 정보시스템트랙 조교수
Manuscript Received : May 10, 2022
Accepted : May 24, 2022
* Corresponding Author : Dongyoung Koo(dykoo@hansung.ac.kr)

비스는 다수 사용자로부터의 중복된 데이터 아웃소싱에 따른 저장 효율성 저하 극복을 위하여 데이터 중복제거(duplicate elimination) 기법을 적용해 왔다.

데이터 프라이버시의 중요성이 강조되면서 사용자에 의한 암호화된 데이터의 아웃소싱은 클라우드 저장소에서 동일성 확인을 불가능하게 하였고, 클라우드 스토리지 환경에서 데이터 중복제거 안전성 및 효율성 개선을 위한 연구가 활발히 연구되고 있다[5-11]. 클라우드 컴퓨팅의 중앙 집중화에 따른 부작용을 개선하기 위하여 클라우드 컴퓨팅의 확장된 개념으로 엣지 컴퓨팅 개념이 도입되면서, 클라우드 서비스의 분산 및 여러 구조의 결합을 통한 효율성 개선 연구 또한 활발히 이루어지고 있으나 프라이버시 보존 데이터 중복제거에서의 안전성 및 효율성 향상을 위한 연구는 상대적으로 찾아보기 어렵다. 전적으로 신뢰하기 어려운 제3자에 의하여 관리되는 원격 공간에서 데이터 프라이버시 보존을 위한 암호화 기법은 연산 복잡도를 높여 서비스 성능을 제한하는 걸림돌로 여겨질 수 있다[12].

본 논문에서는 대표적인 상용 신뢰실행환경(trusted execution environment, TEE)을 제공하는 Intel SGX(Software Guard Extensions)를 활용하여 엣지 컴퓨팅 환경에서 프라이버시를 보존하면서 사용자 경험 향상 및 자원 효율성 극대화를 위한 혼합형 데이터 중복제거 기법을 이용한다. 사용자-엣지 디바이스 사이에서는 서버 측 중복제거(server-side duplicate elimination) 기법을 적용하고, 엣지 디바이스-클라우드 서비스 제공자 사이에서는 사용자 측 중복제거(client-side duplicate elimination) 기법을 적용한다. 서비스 제공자는 사용자의 민감 정보에 대한 관리를 신뢰실행환경 플랫폼의 신뢰 영역에 위임한 상태로 처리한다. 제안 기법의 특징을 요약하면 아래와 같다:

- **사용자 경험 및 가용성 증가**: 사용자가 암호화한 데이터를 엣지 디바이스에 오프로딩(off-loading)함으로써, 사용자는 데이터 아웃소싱 이후 중복제거 여부와 상관없이 독립적 활동 가능
- **통신 효율성 개선**: 엣지-클라우드 간 사용자 측 중복제거를 통한 원거리 통신에서의 중앙 클라우드 서버로의 네트워크 집중에 따른 효율성 저하 개선
- **연산 효율성 개선**: 신뢰실행환경 활용을 통한 엣지 디바이스에서의 연산 최소화 및 중앙 클라우드 서버의 연산 효율성 개선

2. 관련 연구

데이터 중복제거는 반복하여 등장하는 데이터를 확인하여 하나의 사본을 제외한 나머지를 제거하고 논리적 링크로 연결함으로써 데이터 중복에 따른 저장 공간의 낭비를 줄이는 대표적인 효율적 저장 공간 관리 기법이다[13]. 본 절에서는 엣지 컴퓨팅 환경 및 신뢰실행환경에 기반한 데이터 중복제거 기법 관련 연구를 살펴본다.

2.1 엣지 컴퓨팅 환경에서의 데이터 중복제거

클라우드 서비스 대중화에 따른 사용자 증가로 말미암은 네트워크 및 연산 과부하, 중앙 클라우드 서버와의 서비스 이용을 위한 원거리 통신에 기인한 서비스 지연은 사용자 경험(user experience, UX)을 현저히 해치는 결과를 초래하였다. 이를 극복하기 위하여 클라우드 컴퓨팅의 확장된 개념으로, 물리적으로 분산된 환경에서 중앙 클라우드의 서비스를 중계 또는 대리하여 제공하는 엣지 컴퓨팅(edge computing) 개념이 등장하였다[14]. 포그 컴퓨팅(fog computing)이라고도 불리는 엣지 컴퓨팅 환경에서는 사용자 단말과 물리적으로 인접한 곳에 분산되어 존재하는 엣지 디바이스(edge device)가, 주변 사용자들에게 네트워크 중앙의 클라우드 서버를 대신하여 서비스를 제공함으로써 사용자 요청에 적은 통신 비용으로 신속한 대응이 가능하고 적은 횟수의 네트워크 중계를 거쳐 보안 기술 적용 및 변경이 보다 용이하며 이동성을 지원하는 등 여러 장점을 지닌다.

Koo et al.은 엣지 컴퓨팅 환경에서 데이터 중복제거 방법을 제시하였으며[15], 높은 대역폭의 근거리 통신망 환경에서 사용자와 인접한 엣지 디바이스에 데이터를 아웃소싱하는 서버 측 중복제거와 엣지 디바이스와 중앙 클라우드 사이에서 적은 통신 비용으로 중복 데이터를 식별하고 제거하는 사용자 측 중복제거를 결합함으로써 네트워크 자원의 효율적 사용을 가능하게 하였다. Koo and Hur는 이를 확장하여 엣지 컴퓨팅 환경에서 중복제거된 데이터 소유자의 동적 관리를 위한 방안을 제시하였다[16]. 하지만 데이터 아웃소싱의 대중화에도 불구하고 엣지 컴퓨팅 환경에서의 데이터 중복제거에 대한 연구 결과는 많지 않다. 관련 연구 [15, 16]에서는 데이터 중복확인을 위하여 저장된 데이터에 대한 전수조사가 필요하고 이를 위한 복잡한 암호학적 연산을 필요로 하기 때문에 저장된 데이터의 개수가 증가할수록 중앙 클라우드 및 엣지 디바이스에서의 연산 부하 또한 비례하여 증가하여 급격한 성능 저하를 초래한다.

본 논문에서는 엣지 컴퓨팅 환경에서 클라우드 스토리지 저장소에서의 연산 성능 향상을 통하여 유연한 확장이 가능한(scalable) 효율적 데이터 중복제거 방법을 모색한다.

2.2 신뢰실행환경 및 Intel SGX

신뢰실행환경은 프로세서에 의하여 직접 관리되는 별도의 물리적인 공간을 설정함으로써, 외부 공격자를 비롯한 악의적 운영체제, 하이퍼바이저 등 관리자 권한을 가진 공격자로부터도 안전하게 데이터를 처리할 수 있는 신뢰 영역을 제공한다[17]. 신뢰실행환경을 활용함으로써, 응용 계층에서의 복잡한 암호화 연산에 따른 성능 저하를 최소화하면서 독립 개체로 여겨질 수 있는 신뢰 영역에서 민감 정보를 처리하여 프라이버시 보존 데이터 중복제거의 성능을 향상시킬 수 있다.

대표적인 신뢰실행환경으로는 ARM TrustZone[18], Intel SGX[19], AMD SEV(Secure Encrypted Virtualization)[20] 등이 있다. AMD SEV는 가용한 시스템 메모리를 모두 이용

할 수 있는 장점이 있지만, IaaS(Infrastructure as a Service)와 같이 클라우드 시스템 전반에 걸친 고립 환경 제공을 목적으로 하이퍼바이저 및 가상머신 커널에서의 변화가 이루어졌으며 관리자 수준(ring-0)에서의 신뢰 영역을 제공하기 때문에 넓은 영역의 신뢰 컴퓨팅 기반(trusted computing base, TCB)을 요구한다. 반면, Intel SGX는 시스템 메모리의 일부(최대 128 MB)만을 신뢰 영역으로 설정할 수 있다는 제약이 있지만, 사용자 응용(ring-3)에서의 신뢰 영역을 제공하기 때문에 상대적으로 작은 TCB를 구성하므로 보안에 민감한 특정 응용에 활용하기 적합하다. 본 논문에서는 상용 시스템에서 널리 사용되고 있는 Intel SGX를 활용한 프라이버시 보존 선행 연구를 살펴본다.

Intel SGX는 주기억장치에 인클레이브(enclave)라는 신뢰 영역을 할당하여 프로세서에 의한 직접적인 관리가 이루어지며 메모리에 암호화된 형태로 관리되므로 관리자 권한을 가진 사용자로부터도 기밀성이 보장된다. 신뢰 영역에서 동작하는 인클레이브의 코드와 데이터에 대한 무결성 검증을 위하여 동일 시스템 내에서의 지역 검증(local attestation, LA) 및 원격지 시스템에서의 원격 검증(remote attestation, RA) 기법을 제공하며 주기억장치에 상주하는 인클레이브에서 처리된 데이터의 장기 보존 및 재활용을 위하여 밀봉(sealing) 및 개봉(unsealing) 메커니즘이 제공된다[21].

Ren et al.이 제시한 SGXDedup[22]은 Intel SGX 플랫폼을 활용하여 암호화된 데이터의 중복제거 성능을 향상시키는 방안을 제시하였다. Keelveedhi et al.에 의해 소개된 DupLESS[23]의 키 관리 서버와 사용자 응용에 인클레이브를 활용함으로써 기밀성 수준을 동등하게 유지하면서 중복제거 수행 속도를 현저히 향상시켰다. 하지만 SGXDedup은 독립 키 서버 외에도 데이터 아웃소싱을 수행하는 모든 사용자 응용에 중복확인을 위한 인클레이브가 존재하므로, Intel SGX 플랫폼을 지원하지 않는 시스템을 소유한 사용자는 중복제거 서비스 이용이 제한되는 문제가 존재한다. Miranda et al.에 의해 제시된 S2Dedup[24]은 외부 도청자 뿐 아니라 관리자 권한을 지닌 시스템 내부자로부터 안전성을 보장하기 위하여 주기적 키 갱신 또는 휴리스틱에 기반한 중복 허용 빈도 변경 등 다양한 방법론을 적용하여 빈도 분석(frequency analysis)에 의한 부채널 공격에 내성을 지닌 데이터 중복제거 기법을 제시하였다. 하지만 클라우드와 사용자 간의 2-tier 사용자 측 중복제거 기법을 적용하고 있으므로, 엣지 컴퓨팅 환경으로의 확장에 제약이 있다.

본 논문에서는 S2Dedup에서와 같이 서비스 제공자 시스템에서 동작하는 신뢰 영역으로부터 기밀성이 보존된 상태를 유지하면서 엣지 컴퓨팅 환경에서 자원 효율성을 개선한 데이터 중복제거를 위한 방법을 모색한다.

3. 시스템 구성 및 개선 목표

본 논문에서는 엣지 컴퓨팅 환경에서 프라이버시 보존을 위한 암호화된 데이터의 아웃소싱 과정을 설계하고 성능 향상을 위하여 신뢰실행환경인 Intel SGX를 활용한다.

이를 위하여 통상적인 엣지 컴퓨팅 환경이 적용된 Koo et al.의 연구[15]와 동일한 시스템 구조를 이용한다. [15]에서는 신뢰실행환경이 고려되지 않았으나, 본 논문에서는 Fig. 1과 같이 상용화되어 널리 이용되고 있는 Intel SGX를 활용하여 모든 사용자가 서비스를 이용할 수 있도록 중앙 클라우드 서버에서 인클레이브가 동작하는 환경을 설정하였다.

3.1 시스템 구조

제안 기법의 데이터 아웃소싱에는 방대한 컴퓨팅 자원을 가지고 사용자의 암호화된 데이터의 업로드 및 다운로드 요청을 수행하는 클라우드 서비스 제공자(Cloud Service Pro-

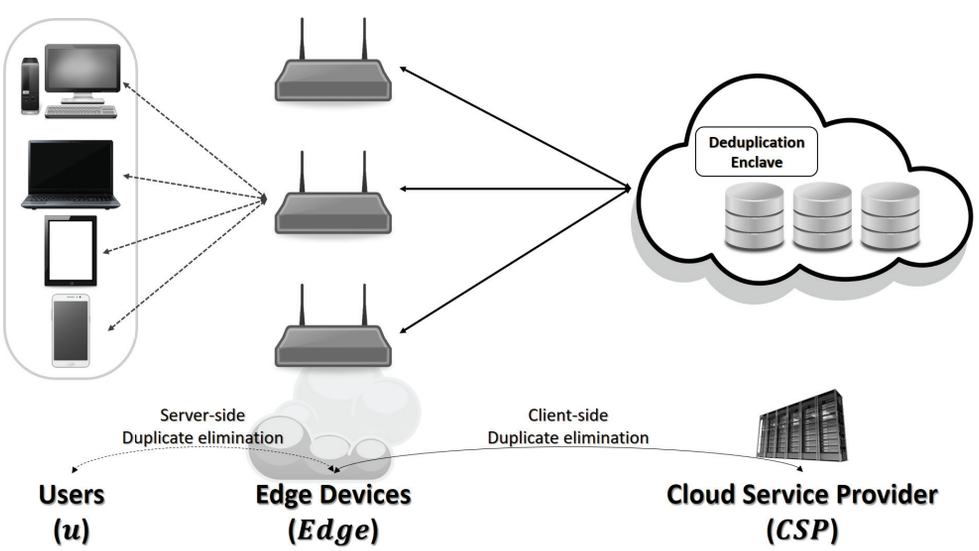

Fig. 1. System Architecture of the Proposed Scheme

vider, *CSP*) 및 *CSP*의 역할을 사용자 근처에서 제공하면서 사용자 경험을 개선하기 위한 엣지 디바이스(Edge devices, *Edge*), 그리고 데이터 아웃소싱을 수행하는 주체인 사용자(User, u)가 참여한다. *CSP*는 (*Edge*를 통해) 사용자로부터 전달받은 암호문의 복호화 및 중복식별, 중복제거를 수행하게 되는데, 이는 아웃소싱된 데이터로부터 데이터 프라이버시가 보존된 상태에서 중복제거 및 데이터 관리를 수행하기 위하여 신뢰 영역(trusted area)인 *deduplication enclave* (이하 *dedup.enclave*) 내부에서만 처리하도록 한다.

3.2 위협 모델

CSP 및 *Edge*는 제안된 프로토콜을 준수하면서 데이터 아웃소싱 과정 또는 저장된 데이터로부터 사용자가 아웃소싱한 평문 데이터의 정보를 획득하려는 시도를 수행할 수 있는 잠재적 공격자(honest-but-curious adversary)로 가정한다. *CSP* 및 *Edge*는 서로 독립적인 제3의 기관에서 관리될 수 있으나, 데이터 프라이버시를 침해하기 위하여 공모할 수 있다. 단, *CSP* 내부에서 동작하는 인클레이브는 안전한 신뢰 공간으로 간주하며 부채널 분석 등의 공격은 별도의 선행 연구를 따르는 대응 방안을 적용할 수 있으므로 고려하지 않는다. 또한, 적법한 사용자 u에 의하여 아웃소싱된 데이터를 유추하여 기밀성을 해치기 위한 도청(eavesdropping) 또는 예측 가능한 데이터의 중복성 확인을 시도하는 외부 공격자가 존재할 수 있다.

3.3 개선 목표

엣지 컴퓨팅 환경에서는 클라우드 컴퓨팅과 달리, 물리적으로 사용자와 인접한 위치에서 중앙 클라우드의 서비스를 대리하는 엣지 디바이스가 존재한다. 따라서 엣지 디바이스를 효율적으로 활용하여 사용자 경험 및 가용성을 증대시키면서 엣지 디바이스의 도입으로 인하여 새롭게 발생 가능한 데이터 프라이버시 침해 위협에 대응하기 위한 메커니즘이 필요하다.

다시 말하면, 사용자와 엣지 디바이스 간 데이터의 송수신에서 SSL/TLS와 같은 단대단(end-to-end) 암호통신을 이용하더라도 잠재적 공격자로 여겨지는 엣지 디바이스는 전송되는 내용을 열람 및 수정할 수 있으므로 엣지 디바이스에 의하여 유도될 수 있는 정보의 양을 최소화하면서 엣지 컴퓨팅 환경의 장점을 효과적으로 활용하기 위한 방안이 필요하다. 이를 위하여 본 논문에서 개선하고자 하는 목표를 요약하면 아래와 같다:

- 엣지 컴퓨팅의 장점 극대화: 사용자 인접 엣지 디바이스의 높은 네트워크 대역폭을 활용한 신속한 데이터 아웃소싱 및 사용자의 데이터 아웃소싱 이후 중복식별 및 중복제거 등을 위한 온라인 상태 요구사항 완화를 통한 사용자 가용성 개선
- 엣지 컴퓨팅의 보안 위험 개선: 엣지 디바이스에 의한 데이터 프라이버시 침해 방지 및 엣지 디바이스와 중앙 클라우드의 사용자 측 암호화된 데이터 중복제거 기법을 활용한 프라이버시 보존 및 통신 효율성 개선

4. 엣지 컴퓨팅 환경에서의 신뢰실행환경 기반 효율적 프라이버시 보존 데이터 중복제거

본 논문에서 제시하는 프라이버시 보존 데이터 중복제거 기법은 Fig. 1과 같이 Koo et al.[15]이 제시한 일반적인 엣지 컴퓨팅 환경에서 동작하며, 신뢰실행환경을 구축함으로써 안전성과 함께 효율성을 개선하는 것을 목표로 한다. 제안 기법에 사용되는 용어 및 설명은 Table 1과 같다.

신뢰실행환경을 활용한 엣지 컴퓨팅 환경에서 프라이버시 보존 효율적 중복제거를 지원하는 데이터 아웃소싱 단계 및 데이터 흐름 개요는 Fig. 2와 같다.

4.1 시스템 초기 구성(System setup)

*CSP*는 사용자의 요청을 처리하기 위한 응용을 구동한다. 사용자 등록 및 중복확인/제거를 담당하는 인클레이브(*dedup. enclave*)를 초기화하며, *CSP*의 사용자 부근에서 중복식별 및 제거를 수행하는 *Edge*가 배포되어 사용자 u에게 서비스를 제공한다.

1) *CSP* 메타데이터 인덱스 생성

*CSP*는 아웃소싱된 데이터 및 소유자 관리를 위한 세 가지

Table 1. Notations

Notation	Description
u_i	Identifier of user i
CSP	Cloud service provider
$Edge_i$	Edge device managing user i's requests and responses from *CSP*
k_i	Secret key of user i
P	Plaintext to be outsourced
λ_{ID}	Size of an identifier (in bits)
λ_K	Size of a key (in bits)
ϕ	Empty set
$\perp$	Invalid output
$a \| b$	Concatenation of a and b
$a \leftarrow b$	Assignment of a as b
$a \xleftarrow{\$} B$	Uniform-random assignment of a as an element of set B
$h(\cdot)$	Cryptographic hash function
$h_i(\cdot) = h(\cdot) \oplus k_i$	Masked hash value with user i's secret key k_i
$E_i(\cdot) = E(k_i, \cdot)$	Secure symmetric encryption algorithm with user key k_i
$D_i(\cdot) = D(k_i, \cdot)$	Secure symmetric decryption algorithm with user key k_i (corresponding to E_i)
$seal_{CSP}(\cdot)$	Data sealed by the *CSP* (i.e., *dedup. enclave*)
$C_i = D_i(P)$	Ciphertext encrypted by user i's key

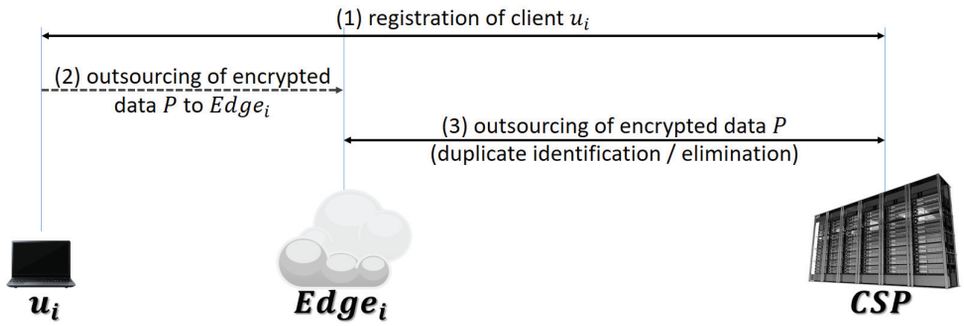

Fig. 2. Overall Procedure of Data Outsourcing

UserTable

User	User's secret key
u_i	k_i
u_j	k_j
...	...

(a) User-management table with entry $< u_i, k_i >$

OwnershipTable

$h(P)$	*Owners*
$h(P_t)$	$\{u_o, u_q, ...\}$
$h(P_r)$	$\{u_s, u_t, \cdots\}$
...	...

(b) Ownership-management table with entry $< h(P), \{u_i, \cdots\} >$

DataTable

Data identifier	Data (sealed by *CSP*)
$loc(P_v) = seal_{CSP}(P_v)$	$loc(seal_{CSP}(P_v)) = filename(P_v)$
$loc(P_w) = seal_{CSP}(P_w)$	$loc(seal_{CSP}(P_w)) = filename(P_w)$
...	...

(c) Data-management table with entry $< seaed\ h(P), loc(P) >$

Fig. 3. Metadata Data Structure Used in the Proposed Scheme

키-값 데이터베이스를 생성하고 관리한다. 신뢰 영역인 *dedup. enclave*는 $UserTable$(Fig. 3a) 및 $OwnershipTable$(Fig. 3b)을 생성하며 인클레이브 내부에서만 메타데이터를 관리한다. $UserTable$은 사용자 등록 정보를 관리하며, 개별 사용자의 식별자 u_i를 키로 하여 u_i의 암호화 키(k_i)를 값으로 가진다. $OwnershipTable$은 아웃소싱된 데이터의 소유자 정보를 관리하며, 평문 데이터(P)의 해시값($h(P)$)을 키로 하여 P의 소유자 집합을 값으로 가진다. 또한, 신뢰 영역인 *dedup. enclave*의 메모리 제약을 극복하기 위하여 암호화된 데이터는 인클레이브 외부에 저장되는데 이를 관리하기 위한 $DataTable$(Fig. 3c)은 *dedup. enclave*에 의해 밀봉된 형태의 데이터 해시값($seal_{CSP}(h(P))$)을 키로 하며 보조기억장치에 저장된 데이터($seal_{CSP}(P)$)의 파일명($loc(seal_{CSP}(P))$)을 값으로 가진다.

2) 사용자 u_i 등록

서비스 수준 협약(service level agreement, SLA)을 통하여 클라우드 스토리지 서비스를 이용하고자는 사용자는 Table 2와 같은 사용자 등록 과정을 거친다. 원격 검증(RA)을 통하여 *CSP* 내부의 *dedup. enclave*의 무결성이 보장된 경우, 사용자는 암호화에 사용할 비밀키(k_i)를 생성하여 (RA 과정에서 협의한 세션키를 이용한) 보안 통신을 통하여 *dedup. enclave*에게 $RegReq_i =< k_i >$을 전달한다. *dedup. enclave*는 중복되지 않은 임의값을 선택하여 등록을 요청한 사용자의 식별자(u_i)로 할당하고 u_i에게 성공 여부와 함께 응답한다. *dedup. enclave*는 $UserTable$에 사용자 정보 (u_i, k_i)를 추가한 후, 사용자에게 등록 성공 여부에 대한 응답을 보냄으로써 사용자 등록 과정을 마친다.

Table 2. Pseudocode of Client Registration

```
Client registration(u_i, CSP)

u_i ↔ dedup. enclave@CSP:
    do remote attestation(RA) with session key agreement
    if integrity of dedup. enclave is not guaranteed:
        terminate

u_i:
    k_i ←$ {0,1}^λK
    send RegReq_i =< k_i > to dedup. enclave
      via secure channel (using session key agreed upon RA)

dedup. enclave@CSP:
    k_i ← RegReq_i
    do-while u_i ∉ UserTable:
        u_i ←$ {0,1}^λID
    if there is an unexpected error:
        send <FAILURE> to u_i
    else:
        Append(UserTable, key = u_i, val = k_i)
        send <SUCCESS, u_i > to u_i
```

$UserTable$의 값은 *CSP*의 *dedup. enclave* 내부에서만 사용되는데, SGX의 밀봉(sealing) 기법을 통하여 인클레이

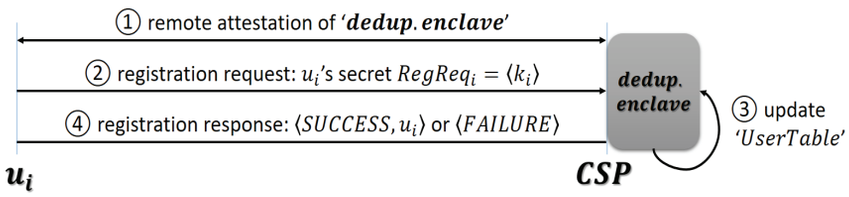

Fig. 4. Data flow of client registration phase

브 외부에 기밀성이 보장된 상태로 저장되고 서비스를 재시작하는 경우에는 보조기억장치로부터 밀봉된 정보를 개봉(unseal)하여 서비스를 지속한다. 사용자 등록에는 $Edge$가 참여하지 않으며, 송수신되는 데이터 흐름은 Fig. 4와 같다.

4.2 사용자 데이터 아웃소싱(Data outsourcing)

아웃소싱할 데이터 P를 소유한 사용자 u_i는 사용자의 비밀키 k_i로 마스킹된 해시값 $h_i(P) = h(P) \oplus k_i$ 및 암호문 $C_i = E_i(P) = E(k_i, P)$을 생성하여 $OutReq_i = < u_i \parallel h_i(P) \parallel C_i >$를 u_i에 인접한 엣지 디바이스 $Edge_i$에 전송한다.

u_i는 $Edge_i$에 데이터를 아웃소싱한 이후에는 프로토콜에 참여하지 않고 오프라인으로 존재할 수 있다. u_i와 $Edge_i$ 사이의 통신은 암호화되지 않은 공용 채널을 이용한다.

4.3 데이터 중복식별(Duplicate identification at $Edge$)

$Edge_i$는 사용자 u_i로부터 전달받은 값에서 실제 암호문을 제외한 정보를 중복식별을 위하여 CSP에 전송한다. CSP의 $dedup.\ enclave$는 $UserTable$ 및 $OwnershipTable$을 참고하여 업로드 요청 데이터가 이미 CSP에 존재하는지 확인한다. $Edge_i$와 CSP 사이에서는 암호화되지 않은 통신을 이용하여 데이터가 송수신된다.

1) 중복식별을 위한 $Edge_i$의 태그 생성 및 CSP 전송

$Edge_i$는 아래와 같이 u_i로부터 전달받은 $OutReq_i$로부터 실제 암호문을 제외한 Tag_i를 생성하여 CSP에 전달한다:

$$(u_i \parallel h_i(P_i) \parallel C_i) \leftarrow OutReq_i$$
$$Tag_i \leftarrow < u_i, h_i(P_i) > .$$

2) $deudp.\ enclave$에 의한 암호문 중복식별

$dedup.\ enclave$는 Tag_i 및 $UserTable$로부터 업로드를 요청한 사용자의 비밀키 k_i를 획득하고, 아웃소싱하고자 하는 데이터 P의 해시값($h(P)$)을 복원한다. 계산한 $h(P)$를 키로 $OwnershipTable$에 동일한 평문 데이터가 CSP에 이미 저장되어 있는지 확인하며, 이미 저장된 데이터에 대한 아웃소싱 요청인 경우 u_i를 $OwnershipTable$에 추가함으로써 소유자임을 인증한다. $OwnershipTable$은 $UserTable$과 함께 $dedup.\ enclave$에 의하여 관리되며, 밀봉 기법을 활용한다.

4.4 중복되지 않은 데이터의 아웃소싱(Outsourcing of non-existing data in CSP by $Edge_i$)

업로드 요청에 대응되는 데이터 P가 CSP에 저장되지 않은 새로운 데이터인 경우, $dedup.\ enclave$는 $Edge_i$에게 암호문 C_i의 업로드를 요청한다. $Edge_i$는 $OutReq_i$로부터 C_i를 추출하여 CSP에 업로드하며, $dedup.\ enclave$는 $Ownership-Table$에 키-값($< key = h(P), val = \{u_i\} >$)을 추가하고 C_i로부터 평문 P를 복원하여 $DataTable$에 추가한다. 실제 아웃소싱된 데이터는 상대적으로 큰 저장공간을 필요로 하기 때문에, $DataTable$은 프라이버시 보존을 위하여 $dedup.\ enclave$로부터 전달받은 밀봉된 값을 인클레이브 외부의 CSP 응용이 관리한다.

4.2-4.4절의 사용자로부터의 아웃소싱 데이터 생성부터 $dedup.\ enclave$에 의한 중복제거 및 업로드가 완료되기까지의 데이터 아웃소싱 절차는 Table 3과 같으며, 데이터 흐름은 Fig. 5와 같다.

5. 실험 및 분석

제안 기법의 성능 향상을 비교하기 위하여 엣지 컴퓨팅 환경에서 SGX를 활용하지 않은 Koo et al.의 프라이버시 보존 데이터 중복제거 기법[15]과 SGX를 활용한 제안 기법을 구현하여 실험을 수행하였다.

5.1 구현 및 환경 설정

실험 환경은 Ubuntu 18.04 LTS(64bits)가 동작하는 Intel i7-10710U(1.10-4.70GHz) CPU 및 32GB(16GB x2) RAM이 장착된 시스템에 C/C++로 구현하였다. [15]에서는 공개키 암호 시스템을 함께 이용하고 있으므로, NIST 권장 256비트 보안 수준의 AES256 암호 및 SHA256(SHA-2) 해시 함수와 함께 512비트 supersingular 타원곡선(elliptic curve) 암호를 이용하였다. 제안 기법에서는 Intel SGX를 사용하기 위하여 드라이버(v2.6.0_4f5bb53), SDK(v2.7.100.4), SGX SSL(lin_2.5_1.1.1d), OpenSSL(v1.1.1d)을 설치하고 AES256 및 SHA256을 이용하였다. 데이터 중복제거에서는 고정 크기(4 KB 단위)의 블록에 대한 중복식별 및 제거를 수행하였으며, 다수의 사용자를 지원하는 환경을 고려하였다. 제안 기법에 사용된 자료 구조는 unordered_map을 이용하였으며, 각 요소에 대한 설명은 아래와 같다.

Table 3. Pseudocode of Data Outsourcing

Data outsourcing(u_i, $Edge_i$, CSP)

u_i:
  $C_i \leftarrow E_i(P) = E(k_i, P)$
  send $OutReq_i = <u_i, h_i(P), C_i>$ *to* $Edge_i$

$Edge_i$:
  $(u_i, h_i(P), C_i) \leftarrow OutReq_i$
  send $Tag_i = <u_i, h_i(P)>$ *to* CSP

dedup. enclave:
  $(u_i, h_i(P)) \leftarrow Tag_i$
  $val \leftarrow LookUp(UserTable, key = u_i)$
  if $val = \phi$:
    send $<FAILURE>$ *to* $Edge_i$
    terminate
  $k_i \leftarrow val$
  $h \leftarrow h_i(P) \oplus k_i$
  $U \leftarrow LookUp(OwnershipTable, key = h)$
  if $U \neq \phi$: // *already outsourced data*
    $P \leftarrow Lookup(DataTable, key = h)$ // *by CSP*
    if $P = \perp$:
      send $<FAILURE>$ *to* $Edge_i$
    else if $h = h(P)$:
      $Update(OwnershipTable, key = h, val = U \cup \{u_i\})$
      send $<SUCCESS>$ *to* $Edge_i$
  else: // *non-existing data*
    send $<UPLOAD>$ *to* $Edge_i$

$Edge_i$: // *when received* $<UPLOAD>$ *from dedup. enclave*
  send $<C_i>$ *to* CSP

dedup. enclave:
  $P \leftarrow D_i(C_i) = D(k_i, E(k_i, P))$
  $Insert(DataTable, key = seal(h(P)), val = seal(P))$
  $Insert(OwnershipTable, key = h, val = \{u_i\})$
  send $<SUCCESS>$ *to* $Edge_i$

1) 사용자 정보 관리($UserTable$)

사용자 식별 정보는 다수 사용자에 대한 서비스를 제공하는 CSP의 컴퓨팅 자원에 따라 달라질 수 있는데, 본 실험에서는 식별자(u_i, 64비트, 8 B) 및 비밀키(k_i, 256비트, 32 B)를 위한 공간을 각각 할당하였다.

2) 소유자 정보 관리($OwnershipTable$)

소유자 정보는 사용자가 아닌 데이터 블록에 대응되는 다수의 소유자가 존재할 수 있으므로, 데이터 블록의 SHA256 해시값(256비트, 32 B)을 키로 설정하였으며, 다수의 사용자를 키로 가지기 위하여 vector로 사용자의 식별자(u_i, 64비트, 8 B) 값들을 가변적으로 저장할 수 있도록 하였다.

3) 데이터 관리($DataTable$)

아웃소싱된 데이터 관리를 위한 매핑 테이블로서, 인클레이브 외부의 CSP에 의해 관리된다. 평문 정보에 대한 접근이 가능한 $dedup.\ enclave$에 의하여 생성되는 블록 P에 대한 해시값($h(P)$)의 밀봉된 형태($seal_{CSP}(h(P))$, 256비트, 32 B)를 키로 가지며, 대응되는 값은 인클레이브에 의하여 밀봉되어 인클레이브 외부에 저장되는 파일의 위치로 저장된 블록의 파일명($filename(P)$)으로 설정하였다.

5.2 성능 분석

데이터 아웃소싱을 통하여 저장되는 암호문 및 메타데이터는 중복제거 대상이 되는 데이터의 크기에 비례하며, 동일 크기의 블록 수준 중복제거를 통하여 얻을 수 있는 효과는 동일하다. [25]에 따르면, 실시간 시스템에 데이터 중복제거를 통하여 얻을 수 있는 저장공간의 절감은 약 20-30%이며, 장기 백업용 데이터의 경우 약 70%의 저장공간 절약 효과를 얻을 수 있으나 데이터셋에 따른 효과는 상이하다. 각 블록에 대한 중복식별을 위한 메타데이터의 크기에 차이가 있을 수 있으나, 전체 시스템에서의 저장 효율성에 미치는 영향은 크지 않으므로, 연산 및 통신 오버헤드에 대한 비교 및 분석을 수행하고 제안 기법의 저장 오버헤드에 대하여 살펴본다.

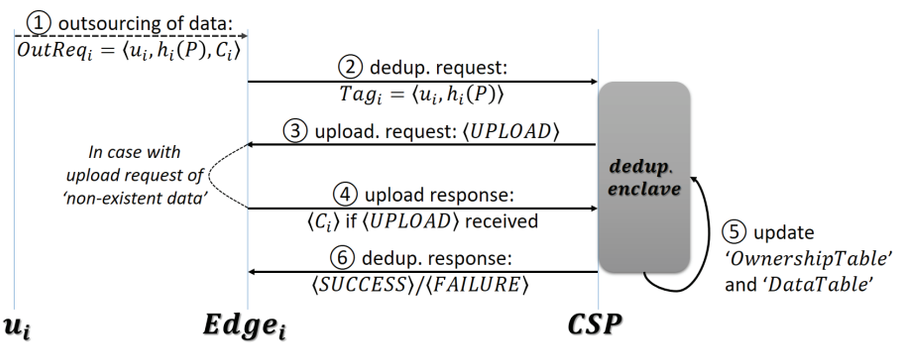

Fig. 5. Data Flow of Data Outsourcing Phase

1) 연산 오버헤드

Koo et al.[15]과 제안 기법에서의 데이터 크기에 따른 업로드 연산 소요시간 비교는 Fig. 6과 같다. 막대그래프(좌측, 주축)는 업로드에 필요한 단위 요소(데이터 읽어오기, 중복제거 태그 생성, 암호화)별 연산시간이며, 꺾은선 그래프(우측, 보조축)은 업로드 연산에 소요된 전체 시간이다. 아웃소싱하고자 하는 데이터의 크기가 클수록 업로드에 소요되는 연산시간은 비례하여 증가한다. Koo et al.[15]의 기법에서는 암호화에 가장 긴 연산시간이 소요된 반면, 제안 기법에서는 태그 생성에 가장 많은 시간이 소요되었다. 파일 입출력에 따른 시간은 동일한 크기의 데이터에 대한 작업으로 유사한 패턴을 보이지만, 중복식별을 위한 태그(제안 기법의 Tag_i에 대응) 생성 및 암호화에서는 제안 기법에서의 연산시간이 현저히 짧다. Koo et al.[15]에서는 타원곡선 상에서의 복잡한 암호학적 연산을 통한 암호문 생성 및 중복식별을 위한 페어링 연산을 수행하는 반면, 제안 기법에서는 SGX 내부에서 대칭키(AES256) 암호화 연산을 수행함에 기인한 것이다.

제안 기법에서 사용자의 초기 등록 및 업로드에 소요된 연산 수행시간은 각각 Table 4 및 Fig. 7과 같다. 사용자 등록은 u_i가 $dedup.\ enclave$의 원격 검증을 수행하여 이상이 없는 경우, 임의로 생성한 암호화 키 $< k_i >$를 CSP에 전달하며, CSP로부터 할당받은 사용자의 식별자(u_i)를 향후 사용을 위해 파일에 저장하는 일련의 절차이다. 응용 시작에 평균 1.246 ms가 소요되었으며, 원격 검증(RA)에 평균 36초 760.976 ms, 임의키 생성에는 0.02 ms, CSP로부터 전달받은 $E_i(u_i)$로부터 u_i를 복원하는데 0.162 ms, 복원한 u_i를 파일에 저장하는데 0.041 ms가 각각 소요되었다. $dedup.\ enclave$원격 검증에서는 송수신되는 메시지($msg1, msg2, msg3, msg4$)의 계산 및 통신 시간이 포함되어 있으나 인텔 검증 서비스(Intel Attestation Service, IAS) 서버로부터의 검증 결과의 전달 및 무결성 결과 수신에서의 지연에 약 36초의 대부분 시간이 소요되었다. 하지만 사용자 등록은 서비스를 최초로 이용하는 사용자에 의하여 1번만 실행되며 이후 서비스 이용에서는 동작하지 않기 때문에 장기간에 걸친 서비스 이용에 있어 큰 영향을 미치지 않는다. 사용자의 데이터 아웃소싱 소요시간은 아웃소싱하고자 하는 데이터의 크기에 비례하는데, 파일을 보조기억장치로부터 읽어오는데 대부분의 시간이 소요되었다. 2 MB 데이터에 대하여, 512개의 블록이 생성되어 각각 해시 연산 및 암호화가 수행되는데 보조기억장치로부터 파일을 읽어오는데 평균 140.153 ms가 소요되었고, 각 블록에 대한 해시 계산은 총 4.27740 ms, CSP로의 업로드를 위하여 해시값을 k_i로 마스킹하는데 0.01170 ms, 각 블록을 암호화하는데 2.73270 ms가 소요되었다. 파일 입출력을 제외하면, 해시값 계산이 가장 긴 연산시간을 필요로 하며, 이는 파일 입출력에 소요되는 시간의 1/32 수준이다. Intel x86에서 지원하는 AES_NI(native instruction)를 활용하여 최적화된 암호화 연산으로 해시 연산보다 빠른 연산 속도를 보였다.

Table 4. Computation time (ms) of Client Registration in the Proposed Scheme

Unit operation	Consumed time (ms)	Standard deviation
Setup	1.246	0.021
Remote attestation	36,760.976	55.127
Random key generation	0.002	0.001
Restoration of user ID	0.162	0.095
File I/O(store user ID)	0.041	0.004

CSP에서의 사용자 등록 과정에서도 u_i와 유사한 연산 시간이 필요하며, 사용자의 식별자인 u_i 생성에 평균 0.21427 ms가 소요되었다. 제안 기법에서 CSP와 $Edge$에 의한 중복식별 과정 및 업로드에 따른 연산 시간은 Fig. 8과 같으며, 아웃소싱 요청이 이루어진 데이터의 크기가 클수록 중복식별

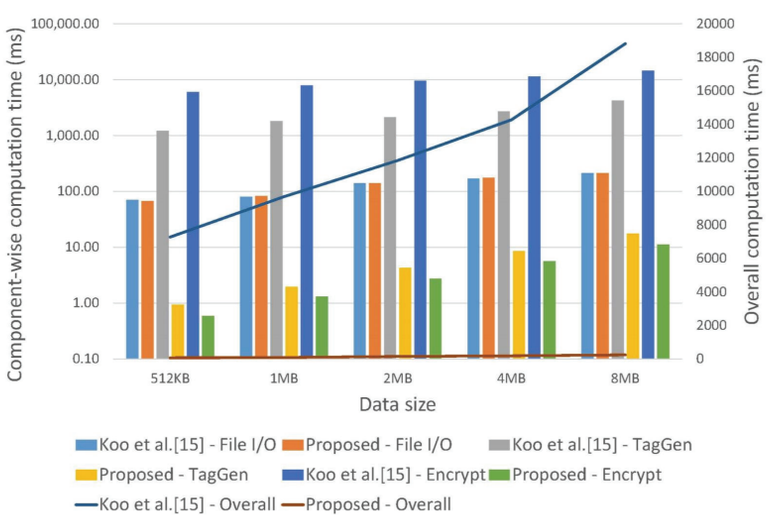

Fig. 6. Comparison of Computation Time (ms) for Client Outsourcing with Varying Data Size

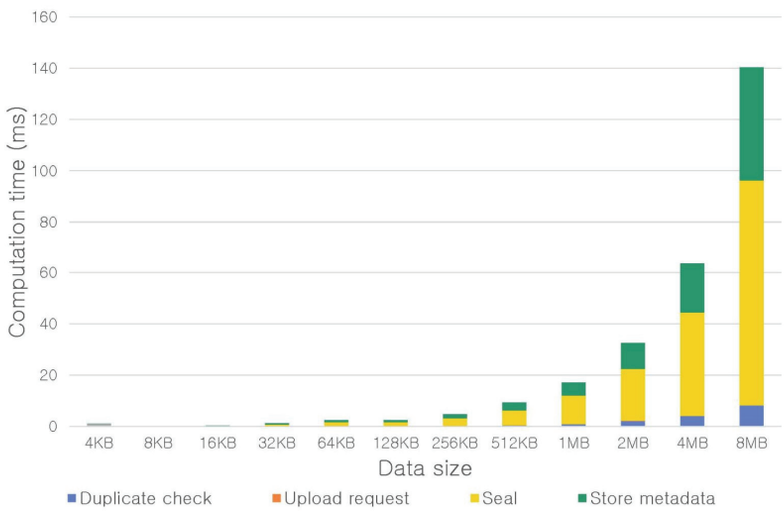

Fig. 7. Computation Time (ms) of CSP Dedup with Varying Size of Data in the Proposed Scheme

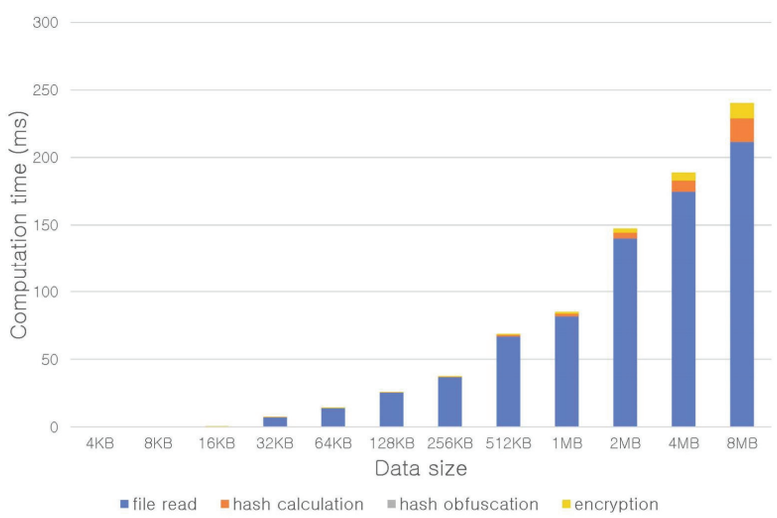

Fig. 8. Computation Time (ms) of Client Upload with Varying Size of Data in the Proposed Scheme

및 중복제거 시간도 비례하여 증가한다. 2 MB 데이터에 대하여, 평균 중복확인은 2.20670 ms, 중복되지 않은 데이터의 업로드 요청 생성 0.00060 ms, 업로드된 블록의 $Data\ Table$ 저장을 위한 밀봉 20.17610 ms, CSP 및 $dedup.\ enclave$에서의 메타데이터 테이블의 갱신을 위하여 10.04160 ms가 평균적으로 소요되었다. 중복된 데이터가 발생하지 않은 경우에는 CSP에 의한 업로드 요청이 존재하지 않으며, $Ownership-Table$을 제외한 메타데이터의 갱신이 이루어지지 않는다.

엣지 디바이스에서는 사용자 업로드 요청($OutReq$)에 대한 일시적인 저장 외에는 CSP로의 전달 및 중복되지 않은 블록에 대한 업로드만 이루어진다. 2 MB 데이터가 모두 중복되지 않은 경우라 하더라도, CSP로부터 요청된 블록 인덱스의 비교 및 메모리 복사를 통한 송신 과정에서 0.001 ms 동안 연산이 이루어지므로 엣지 디바이스에서의 연산 오버헤드는 전체 데이터 아웃소싱 연산에서 무시할 수준(0.001%)이며, 엣지 디바이스의 성능이 제한된 경우라 하더라도 데이터 아웃소싱에 미치는 영향은 매우 제한적이다.

2) 통신 오버헤드

사용자의 식별자(u_i) 크기(λ_{ID})를 8 B, AES256 키 및 SHA256 해시값을 적용하기 위한 저장 공간(λ_K)을 32 B(=256비트)라 할 때, 원격 검증을 제외한 사용자 등록을 위해 송신되는 데이터는 40 B이다. 블록 단위로 분할된 데이터에 대응되는 메타데이터의 크기는 사용자가 아웃소싱하고자 하는 전체 데이터의 크기에 비례하며, Koo et al.[15]과 매우 유사한 패턴을 보인다. 이는 사용자 측에서는 CSP의 데이터 중복여부와 무관하게 엣지 디바이스에 암호화한 데이터를 전달하기 때문이며, 이후 중복식별 및 제거는 전적으로 $Edge$와 CSP 사이에서만 이루어진다. 사용자와 엣지 디바이스는 물리적으로 인접한 상태에서의 통신이 이루어지므로, 근거리 통신망으로

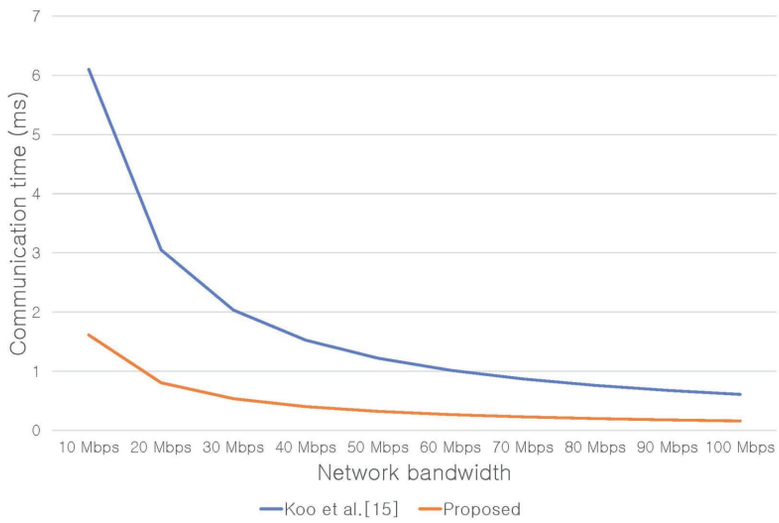

Fig. 9. Comparison of Communication Time (ms) for Client Outsourcing with Varying Network Bandwidth

간주할 경우 네트워크 대역폭에 따라 송수신 시간이 큰 영향을 받게 되며 대역폭에 따른 2 MB 데이터의 아웃소싱에 소요되는 시간은 Fig. 9와 같다. 제안 기법에서는 2 MB 데이터의 경우 4,096 B의 데이터 블록이 512개 존재하며, 이에 대한 해시값이 512개 생성되어 마스킹된다. 또한 실제 아웃소싱되는 암호문의 크기는 데이터의 크기와 같으므로, 사용자의 식별자를 포함한 송신 데이터의 크기는 16,400 KB(= 16.015625 MB)이다.

3) 저장 오버헤드

데이터 중복제거율은 주어진 일정 기간 입력으로 들어온 저장 요청 데이터의 바이트 대비 출력으로 실제 저장되는 데이터의 바이트 크기로 표현할 수 있다[26]. 특정 환경에서는 다수의 중복제거 기법이 유사한 중복제거율을 보이지만, 데이터 유형, 데이터 변경 주기, 동적/정적 데이터 비율, 데이터 전송 주기 등에 따라 달라진다. Tarasov et al.[27]은 데이터 중복제거의 성능은 데이터 콘텐츠, 접근 패턴, 메타데이터 특징 등에 따라 달라질 수 있는데, 공통된 데이터셋을 통한 비교가 이루어지지 않은 선행 연구의 공정한 성능 평가를 위한 기준을 제시하였다.

NEC의 white paper에서는 대부분의 데이터가 반복되어 저장되는 전형적인 백업 시스템의 경우에는 95% 이상의 중복제거율을 보인다고 발표하였고[28], 마이크로소프트사의 시스템을 통한 실험에서는 사용 중인 표준 시스템에서 50%의 중복제거율을 보이고 백업 이미지에 대해서는 87%의 중복제거율을 보인다고 분석하였다[29]. 실험 데이터셋 및 중복제거 기법에 따라 중복제거율에 근소한 차이를 보이지만 수시로 업데이트되는 동적 데이터보다 백업 용도의 정적 데이터에 대한 중복제거의 성능이 높은 사실을 확인할 수 있다. 또한, 파일 전체가 일치하는 경우에만 중복제거가 이루어지는 파일 수준(file-level) 중복제거보다 블록 수준(block-level) 중복제거의 성능이 높게 나타나는 경향을 보인다.

본 논문에서는 블록 수준 중복제거에 기반하고 있으므로, 블록 수준에서의 중복제거율은 [15]와 동등한 성능을 보인다. 단, 블록 수준 중복제거에서는 각 블록에 대한 메타데이터가 필요하므로 블록 데이터 대비 메타데이터의 크기에 따른 오버헤드를 살펴볼 수 있다. 제안 기법에서는 $DataTable$의 키로 블록의 해시값(256비트, 32 B)을 이용하고 있으므로, 4 KB 블록에 대해서는 0.77519%의 추가적인 저장 공간이 필요하며 1 MB 블록에 대한 중복제거를 수행하는 경우에는 0.00038%의 추가 저장 공간이 필요하다. 이는 저장되는 실제 데이터에 비하여 매우 적은 공간 오버헤드를 차지하는 것을 알 수 있다.

6. 안전성 분석

제안 기법의 안전성을 확인하기 위하여 내부 공격자로 간주될 수 있는 엣지 디바이스 및 중앙 클라우드 서버(클라우드 서비스 제공자)와 외부 공격자가 획득할 수 있는 정보에 따른 분석을 수행한다. 제안 기법에 사용된 암호학적 요소는 AES256 암/복호화, SHA256 해시함수이며 안전성이 보장된다고 가정한다.

6.1 엣지 디바이스

엣지 디바이스 $Edge_i$는 사용자 u_i로부터 업로드 요청에 따른 $OutReq_i =< u_i, h_i(P), C_i >$만을 전달받고, 일부 또는 전부를 CSP에 전달한다. 사용자의 식별자인 u_i는 유일하게 CSP에 의하여 사용자의 비밀키(k_i)를 검색하기 위하여 공개되므로 $Edge_i$가 알 수 있지만, $Edge_i$는 k_i를 알지 못하기 때문에 난독화된 해시값 $h_i(P) = h(P) \oplus k_i$으로부터 아웃소싱하고자 하는 데이터 P의 해시값($h(P)$)을 계산할 수 없다. 마

찬가지로, k_i를 알 수 없는 $Edge_i$는 $C_i = E_i(P) = E(k_i, P)$로부터 평문 블록 P를 구할 수 없다.

6.2 중앙 클라우드 서버($dedup.\ enclave$를 제외한 CSP)

CSP는 사용자 u_i의 등록 요청 메시지로부터 암호화된 사용자의 암호화 키 k_i를 전달받는다. 이는 $dedup.\ enclave$에 대한 원격 검증(RA) 과정에서 $dedup.\ enclave$와 u_i 사이에서 합의된 세션키로 암호화되어 있기 때문에, CSP라 하더라도 $dedup.\ enclave$ 내부에서 일시적으로 사용된 세션키를 알 수 없으므로 u_i의 암호화 키(k_i)를 알 수 없다. $Edge_i$를 통한 u_i의 암호문 아웃소싱 및 중복제거 과정에서도 CSP는 사용자의 식별자인 u_i를 제외한 $h_i(P)$ 및 C_i는 중복 식별 및 중복제거를 위하여 $Edge_i$로부터 전달받은 값을 그대로 $dedup.\ enclave$에 전송하기 때문에 k_i에 대한 정보를 획득하지 못한다.

CSP에 의하여 직접 관리되는 $DataTable$에 있어, CSP는 $dedup.\ enclave$에 의하여 밀봉된 해시값($seal_{CSP}(h(P))$과 암호문($seal_{CSP}(P)$)을 전달받아 메타데이터를 갱신하기 때문에, 밀봉키(seal key)가 인클레이브 외부로 노출되지 않는 SGX 환경에서 $dedup.\ enclave$ 외부에 존재하는 CSP는 평문 및 해시값을 얻을 수 없다.

6.3 외부 공격자

사용자(u_i)와 클라우드 서비스 제공자(CSP) 사이에서의 통신은 SSL/TLS 암호통신이 적용되며 세션마다 새로운 키로 암호화되기 때문에, 도청을 통하여 의미있는 정보를 획득하고자 하는 외부 공격자(A)는 사용자의 식별자(u_i) 또는 암호화 키(k_i)에 대한 정보를 획득할 수 없다.

적법한 사용자로 가장한 공격자(A)는 예측 가능한 데이터에 대하여 임의의 평문 블록 P'을 생성하여 아웃소싱을 수행할 수 있으나, 제안 기법에서는 사용자로 가장한 공격자는 인접한 엣지 디바이스($Edge_A$)에 A가 직접 생성한 키로 암호화한 데이터를 전달한 이후에는 $Edge_A$로부터 중복제거 결과를 전달받지 않기 때문에, 동일 데이터의 존재 유무에 대한 확인(confirmation-of-file, CoF) 공격 또한 차단된다.

7. 결 론

엣지 컴퓨팅 환경에서 다수 사용자로부터의 데이터 아웃소싱 요청을 효율적으로 처리하기 위하여 암호문에 대한 연산을 신뢰실행환경이 제공되는 Intel SGX의 인클레이브에 위임함으로써 응용 수준에서의 복잡한 암호학적 연산을 최소화하고 효율성을 개선하기 위한 방법을 모색하였다. 실험을 통하여 엣지 디바이스 및 중앙 클라우드 서버에서의 통신 및 연산 효율성이 개선됨을 확인하였으며, 안전성 검토를 통하여 데이터 프라이버시가 보존됨을 확인하였다.

References

[1] Statista, "Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025," [Internet], https://www.statista.com/statistics/871513/worldwide-data-created/

[2] Memopal, "Technology," [Internet], https://www.memopal.com/technology/

[3] Dropbox, "Dropbox," [Internet], https://www.dropbox.com/

[4] Google Drive, "Google Drive," [Internet], https://drive.google.com/

[5] P. Puzio, R. Molva, M. Önen, and S. Loureiro, "ClouDedup: Secure deduplication with encrypted data for cloud storage," *IEEE International Conference on Cloud Computing Technology and Science*, pp.363-370, 2013.

[6] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pp.296-312, 2013.

[7] M. Wen, K. Lu, J. Lei, F. Li, and J. Li, "DBO-SD: An efficient scheme for big data outsourcing with secure deduplication," *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp.214-219, 2015.

[8] M. Bellare and S. Keelveedhi, "Interactive message-locked encryption and secure deduplication," *IACR International Workshop on Public Key Cryptography (PKC)*, pp.516-538, 2015.

[9] S. Mishra, S. Singh, and S. T. Ali, "RCDSD: RSA based cross domain secure deduplication on cloud storage," *International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp.1-7, 2018.

[10] P. Singh, N. Agarwal, and B. Raman, "Secure data deduplication using secret sharing schemes over cloud," *Future Generation Computer Systems*, Vol.88, No.2018, pp.156-167, 2018.

[11] Y. Wang, M. Miao, J. Wang, and Xuefeng Zhang, "Secure deduplication with efficient user revocation in cloud storage," *Computer Standards & Interfaces*, Vol.78, pp.1-8, 2021.

[12] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of fog computing and its security issues," *Concurrency and Computation: Practice and Experience*, Vol.28, No.10, pp.2991-3005, 2015.

[13] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simin, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," *Technical Report MSR-TR-2002-30, Microsoft Research*, pp.1-14, 2002.

[14] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, Vol.3, No.5, pp.637-646, 2016.

[15] D. Koo, Y. Shin, J. Yun, and J. Hur, "A hybrid deduplication for secure and efficient data outsourcing in fog computing," *IEEE International Conference on Cloud Computing Technology and Science(CloudCom)*, pp.285-293, 2016.

[16] D. Koo and J. Hur, "Privacy-preserving deduplication of encrypted data with dynamic ownership management in fog computing," *Future Generation Computer Systems*, Vol.78, No.2, pp.739-752, 2018.

[17] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted execution environment: What it is, and what it is not," *IEEE Trustcom/BigDataSE/ISPA*, pp.57-64, 2015.

[18] ARM, "Learn the architecture: TrustZone for AArch64," [Internet] https://developer.arm.com/documentation/102418/0101/What-is-TrustZone-?lang=en

[19] Intel, "Intel® Software Guard Extensions (Intel® SGX)," [Internet] https://www.intel.sg/content/www/xa/en/architecture-and-technology/software-guard-extensions.html

[20] AMD, "AMD Secure Encrypted Virtualization (SEV)," [Internet] https://developer.amd.com/sev/

[21] V. Costan and S. Devadas, "Intel SGX explained," *Cryptology ePrint Archive*, pp.1-118, 2016.

[22] Y. Ren, J. Li, P. P. C. Lee, and X. Zhang, "Accelerating encrypted deduplication via SGX," *USENIX Annual Technical Conference (USENIX ATC)*, pp.303-316, 2021.

[23] S. Keelveedhi, M. Bellare, T. Ristenpart, "DupLESS: Server-aided encryption for deduplicated storage," *USENIX Security Symposium (USENIX Security)*, pp.179-194, 2013.

[24] M. Miranda, T. Esteves, B. Portela, and J. Paulo, "S2Dedup: SGX-enabled secure deduplication," *ACM International Conference on Systems and Storage(SYSTOR)*, pp.1-12, 2021.

[25] D. Meister, J. Kaiser, A. Brinkmann, T. Cortes, M. Kuhn, and J. Kunkel, "A study on data deduplication in HPD storage systems," *IEEE International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, pp.1-11, 2012.

[26] M. Dutch, "Understanding data deduplication ratios," *SNIA (Storage Networking Industry Association) - Data Management Forum*, [Internet] https://www.snia.org/sites/default/files/Understanding_Data_Deduplication_Ratios-20080718.pdf, pp.1-13, 2008.

[27] V. Tarasov, W. Buik, P. Shilane, G. Kuenning, and E. Zadok, "Generating realistic datasets for deduplication analysis," *USENIX Annual Technical Conference (ATC)*, pp.1-12, 2012.

[28] Advanced Storage Products Group, "Identifying the Hidden Risk of Data Deduplication: How the HYDRAstor™ Solution Proactively Solves the Problem," *White paper - NEC*, pp.1-9, [Internet] https://silo.tips/downloadFile/identifying-the-hidden-risk-of-data-deduplication-how-the-hydrastor-tm-solution, 2009.

[29] D. T. Meyer and W. J. Bolosky, "A study of practical deduplication," *ACM Transactions on Storage*, pp.1-20, 2012.

구 동 영

https://orcid.org/0000-0003-3283-5494
e-mail : dykoo@hansung.ac.kr
2009년 연세대학교 컴퓨터·산업공학(학사)
2012년 한국과학기술원 전산학(석사)
2016년 한국과학기술원 전산학(박사)
2016년 ~ 2017년 고려대학교 컴퓨터학과 연구교수
2017년 ~ 현 재 한성대학교 전자정보공학과 조교수
관심분야 : Information Security, Applied Cryptography, Network Security, Cloud/Fog/Edge Computing Security