안전한 엣지 컴퓨팅 환경 구축

Build a secure edge computing environment

요 약

엣지 컴퓨팅 환경에서 발생하는 모든 서비스의 기록을 클라우드 서버와 엣지 단에 남겨 관리하고자 한다. 본 연구에서는 클라우드 서버를 파일 저장소로서 사용할 때, 비정상적인 접근에 의한 원본 파일 변조를 탐지할 수 있는 환경을 구축한다. 클라우드 서버에는 원장(ledger)을 저장, 엣지 단에서는 용량의 문제를 최소화할 수 있도록 요약된 정보만을 저장하고자 한다.

1. 서 론

현재 중앙집중형의 클라우드 서버는 많은 양의 데이터를 처리함으로써 지연이 발생하는 문제점을 갖고 있다. 특히 많은 양의 데이터를 클라우드로 보내는 것은 보안 상 안전하지도 않고, 불필요하며, 비현실적이다.
이에 따라, 여러 주요 기업들은 엣지 컴퓨팅을 위한 아키텍처를 제시하고 있으며, 자사의 인프라 구조를 엣지 디자인 패턴을 활용해 재구성하고 있다.

또한, 엣지 컴퓨팅이 점점 상용화됨에 따라 보안상의 문제도 새로이 대두되고 있다. 우리는 이를 위해 서버에 접근을 하는 모든 경우의 로그를 기록하여 관리하고자 한다.

하지만 엣지 단의 컴퓨터는 모든 서비스의 기록을 저장하기에는 용량 상의 문제가 있다.

때문에, 서비스 접근 기록에 대한 원장(ledger) 정보는 클라우드 서버에 저장을 하되, 엣지 노드에서도 그 근거를 남기고자 한다.

관련된 연구로는, 클라우드 사용이 가능한 엣지 기기가 애플리케이션을 만드는 사람들에게 흘러 들어 갈 수 있도록 해야 한다는 개념인 마이크로소프트의 인텔리전트 엣지[1], 아마존의 AWS 그린그래스[2]가 있다.

2. 기존 연구

2.1 인텔리전트 클라우드와 인텔리전트 엣지

마이크로소프트는 2017년 빌드 개발자 컨퍼런스에서 인공지능과 연계하는 새로운 클라우드 환경을 제시했다. 다양한 기기와 인공지능 그리고 서버 중심에서 벗어난 새로운 컴퓨팅 환경을 언급하면서 이는 새로운 진화라고 설명하고 있다.

유비쿼터스 컴퓨팅과 앰비언트 인텔리전스(Ambient Intelligence)의 시대에 클라우드 사용이 가능한 엣지 기기가 데이터, 애플리케이션 그리고 지능이 새로운 클래스의 애플리케이션을 만드는 사람들에게 흘러갈 수 있도록 해야 한다는 개념으로 인텔리전트 클라우드와 인텔리전트 엣지의 개념을 제시했다. 또한, 애저 IoT 엣지를 제시해 클라우드 지능을 엣지 기기로 확장할 수 있음을 보여주었다.

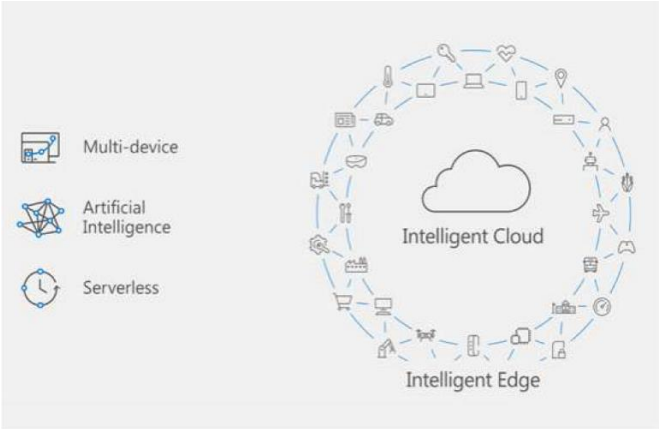

Figure 1 마이크로소프트의 인텔리전트 엣지

2.2 아마존의 그린그래스

아마존은 2016년 12월 're:Invent'에서 그린그래스를 소개함으로써 엣지 게이트웨이와 어플라이언스에 대응하는 소프트웨어를 선보였다. AWS그린그래스에서는 인터넷에 연결되어 있지 않더라도 커넥티드 기기에서 AWS 람다 함수를 실행하고, 기기 데이터를 동기화된 상태로 유지하고, 다른 기기와 안전하게 통신할 수 있다.

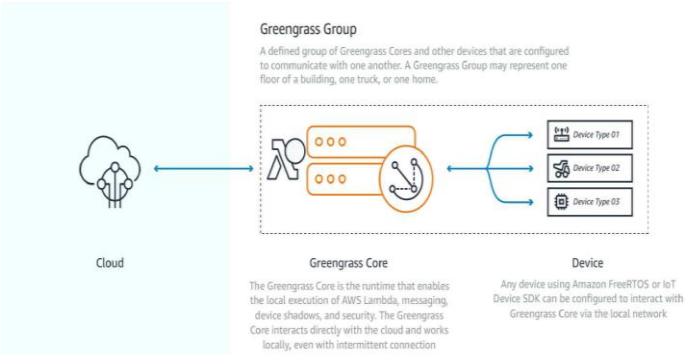

Figure 1 아마존의 AWS 그린그래스

또한 그린그래스 코어 기기와 AWS IoT SDK 지원 기기는 그린그래스 그룹으로 구성해 서로 통신할 수 있고, 코어 기기에서 클라우드와 연결이 끊어지더라도 그린그래스 그룹에 속한 기기는 로컬 네트워크를 통해 서로 계속 통신할 수 있다.

아마존 그린그래스의 초기 고객과 파트너는 인텔, 퀄컴, 테크니칼라, 삼성, 에이서, 노키아, QNAP등이다. 이후 시스템 통합 기업, ISV, 전자 회사, 하드웨어와 반도체 회사 등과 협력하고 있다. 그린그래스는 라즈베리 파이나 ARM 프로세서로 지원되는 비글본 같은 시스템온칩 기기에서 동작할 정도로 사이즈가 작다.

IBM 역시 IoT 비즈니스 유닛이 더블린과 요크타운 연구소와 함께 엣지 컴퓨팅 애플리케이션을 위한 솔루션을 디자인해왔고, 이를 IBM 왓슨 에코시스템에 통합하려고 한다. IBM은 자사가 인수한 웨더 컴퍼니와 함께 본격적으로 엣지 컴퓨팅을 실현하고자 노력하기로 했다. 이를 위해서 IBM 연구소 과학자들과 함께 와이파이나 셀룰라 망이 필요 없이 새로운 피어-투-피어 메시 네트워킹 기술을 개발하고 있다. 파일럿 프로젝트로는 메시 네트워크 얼럿을 만들어 인터넷 연결없이 수십억 명의 사람들에게 기상 정보를 보낼 수 있도록 했다.

3. 시스템 모델

3.1 기존 연구와 차이점 및 해결방안

기존의 엣지 컴퓨팅 연구에서는 중앙 집중형 서버 중심에서 벗어난 새로운 컴퓨팅 환경을 구축하는 데에 초점을 맞추었다. 즉, 클라우드 서버에서 모든 데이터를 처리하던 것을 엣지 단에서 어떠한 방식으로 처리할 것인지에 대한 연구가 주를 이루었다. 엣지 단 컴퓨터에 인공지능을 적용할지, 어떠한 토폴로지로 네트워크를 구성할지, 인터넷에 연결되어 있지 않은 경우에도 엣지가 서비스하는 지역 내에서 어떻게 통신을 가능하게 할 것인가에 대한 연구들이었다.

앞서 언급한대로 이는 클라우드의 중앙 집중형 서비스의 취약점을 개선할 수는 있지만 엣지 자체가 공격의 대상이 될 수 있고, 비정상적인 작업이 일어나면 자칫 서비스 전체가 망가질 수 있다.

따라서 디바이스의 모든 접근 정보를 기록하여 이를 이용한 탐지(detecting)를 원활히 하는 시스템을 구축하고자 한다.

3.2 프로젝트 내용

이 프로젝트는 파일 저장소로서의 클라우드 서버를 구축하고, 저장된 파일의 무결성을 보장하여 안전하게 파일을 관리하는 방향성을 제시한다. 이를 통해 CIA(Confidentiality, Integrity, Availability)를 보장하여 Security Goals를 달성하고자 한다[3].

Figure 3 The CIA triad

클라우드 서버에 원장(파일 원본)을 저장, 엣지 서버에는 해당 파일에 대한 무결성을 보장할 수 있는 최소한의 요약된 정보를 저장하여 행해진 서비스를 기록한다. 이를 통해 엣지 단의 데이터 과부하를 막는다.

프로그램은 클라우드 서버, 엣지 서버, 웹 애플리케이션 크게 3가지로 구성된다. 다음은 본 연구의 대략적인 데이터 흐름도이며, 각 절차와 그에 대한 시나리오는 다음과 같다.

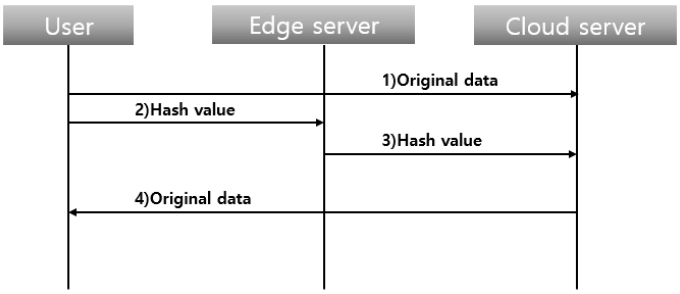

Figure 4 데이터 흐름도

첫번째로 클라우드 서버는 AWS의 S3 스토리지를 이용한다.

두번째로 엣지 서버는 라즈베리 파이를 이용하여 구현한다. 엣지 서버에 저장되는 데이터들은 Node.js를 이용하여 컨트롤한다.

마지막으로 클라이언트 단을 위한 애플리케이션을 구현한다. 애플리케이션은 React.js를 이용한 웹페이지 형식으로 한다.

프로그램의 시나리오는 다음과 같다. 웹페이지에서 사용자가 로그인을 한 후, 서버에 데이터를 저장하는 작업을 한다. 이때, 엣지 서버를 거쳐 클라우드로 저장시킨다. 엣지를 통해 클라우드로 저장함으로써 작업 시간의 단축 및 좋은 서비스를 제공할 수 있다.

엣지 단에는 최소한의 요약된 정보, 클라우드에는 원본을 저장한다. 한 곳이라도 비정상적인 접근이 일어나 데이터 변조가 발생하면 각각의 서버를 비교하여 위변조 여부를 사용자에게 알려주는 시스템을 구축한다. 이러한 구조로 저장함으로써 사용자가 서버에 저장한 데이터의 무결성을 보장한다.

애플리케이션 단에서는 나중에 사용자가 서버에 저장한 데이터를 받아올 때, 진위 여부를 확인시켜주는 화면을 구성한다. 만일 원본 데이터에 문제가 생겼을 때, 이를 사용자에게 알려준다.

4. 결과

다음은 본 연구의 시나리오를 위해 구현한 애플리케이션 동작 및 그에 대한 분석이다.

1) 응용 프로그램 단은 웹 브라우저로 구현하였다. 여기서의 기능은 클라우드 서버로 파일을 저장하는 '저장'과 자신이 저장한 파일들의 목록을 화면으로 불러오는 '조회'의 기능이 있다.

Figure 5 작업 선택 페이지

2) '저장' 작업을 선택하여 저장 페이지로 온 사용자는 자신의 ID를 입력하고 클라우드 서버에 저장하고자 하는 파일을 선택한다.

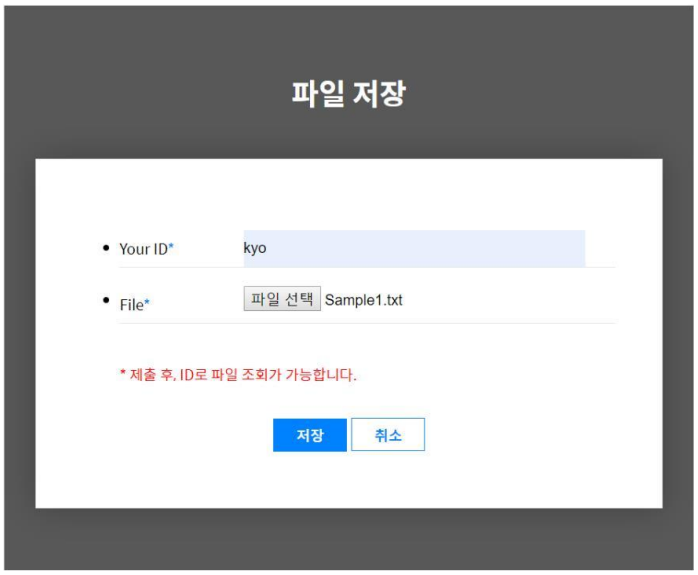

Figure 6 저장 작업 페이지

3) '저장' 버튼을 누르면 특정 사용자가 저장한 파일들을 구분 짓기 위해 서버로 전송할 데이터들은 해당 유저의 'ID', 서버로 전송한 순간의 '시간 값', '파일 이름', '파일 타입', '파일 사이즈', '파일 내용', 그리고 이 모든 정보에 대한 스트링 값을 SHA256 해시 함수로 압축한 '해시 값'이다.

```
/*
  file 정보들, date, ID를 담은 폼 데이터를 이용하여 해시 값 추출!
*/
var hash = sha256(this.state.userID + timeStamp +
  this.state.fileName + this.state.fileType +
  this.state.fileSize + this.state.fileContents);
```

Figure 7 해시 작업

```
axios.post(`${constants.URL_BACK}/store`, {
  body: {
    userID: this.state.userID,
    timeStamp: timeStamp,
    hashValue: hash,
    fileName: this.state.fileName,
    fileSize: this.state.fileSize,
    fileType: this.state.fileType,
    fileContents: this.state.fileContents
  }
},
```

Figure 8 전송 데이터

물론, 작업자의 개인 아이피, 패스워드 등등 압축할 요소를 더 넣는다면 좀 더 유니크한 해시 값을 얻을 수 있다.

특히, 특정 유저가 특정 시간에 서버로 저장한 파일은 유니크하다는 가정을 하여, (ID + 시간 값)은 곧 파일을

특정하는 PK로 가정한다.

4) 엣지 서버는 클라이언트에서 전송한 데이터를 받아 원본 파일이 아닌 최소한의 정보로 사용자가 작업한 파일을 식별할 수 있는 ID, 시간 값, 해시 값 이 세 가지 값 만을 저장하고 클라우드 서버로 수신한 데이터를 그대로 전송한다.

userID	timeStamp	hashValue
kyo	Sun Dec 08 2019 23:38:15	3d39aacf1dd0d24542b44718d64a1937a12e32f0c10ff1e5db0a83bcc991dd21
kyo	Sun Dec 08 2019 23:38:33	8f309fac6443bdaa1c7ad4245f1a423329296451f01dac0a6632948fd75891c9
kyo	Sun Dec 08 2019 23:38:45	87f871a314c0cb44f241d3b69678139c3660048af51c76629e7c3ac8c26801dc
kyo	Sun Dec 08 2019 23:38:56	a7358cf8702a738f632495b7fe317f59555ac4a8e0807376f2cd81181b1c7714a

Figure 9 엣지 서버 DB

5) 클라우드 서버는 클라이언트에서 보낸 모든 정보를 엣지 서버로부터 전송 받아 그대로 저장한다.

userID	timeStamp	hashValue	fileName	fileType	fileSize	fileContents
kyo	Sun Dec 08 2019 23:38:15	3d39aacf1dd0d24542b44718d64a1937a12e32f0c10ff1e5db0a83bcc991dd21	Sample1.txt	text/plain	24	This is a Sample Text 1.
kyo	Sun Dec 08 2019 23:38:33	8f309fac6443bdaa1c7ad4245f1a423329296451f01dac0a6632948fd75891c9	Sample2.txt	text/plain	24	This is a Sample Text 2.
kyo	Sun Dec 08 2019 23:38:45	87f871a314c0cb44f241d3b69678139c3660048af51c76629e7c3ac8c26801dc	Sample3.txt	text/plain	24	This is a Sample Text 3.
kyo	Sun Dec 08 2019 23:38:56	a7358cf8702a738f632495b7fe317f59555ac4a8e0807376f2cd81181b1c7714a	Sample4.txt	text/plain	24	This is a Sample Text 4.

Figure 10 클라우드 서버 DB

6) '조회' 작업을 선택하여 조회 페이지로 온 사용자는 자신의 ID를 입력하면 서버로부터 자신이 저장했던 파일의 리스트를 불러와 브라우저에서 확인할 수 있다.

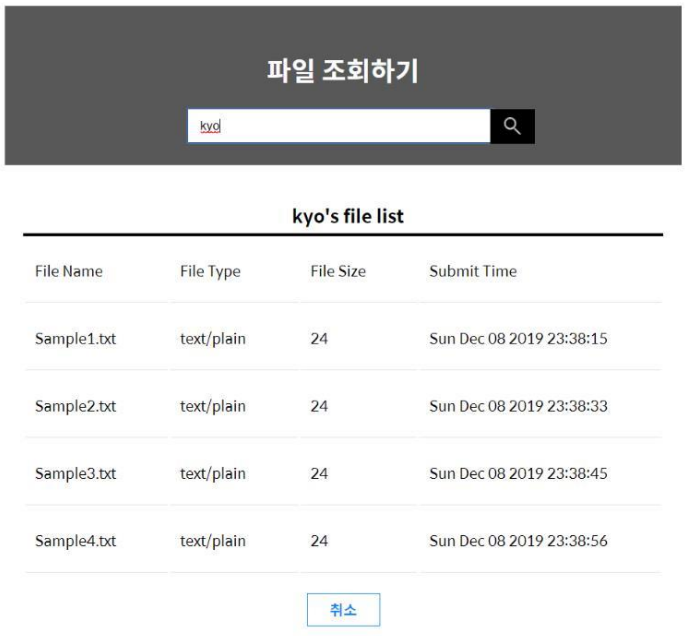

Figure 11 조회 작업 페이지

이때, 사용자는 특정 시간에 자신이 보낸 파일 정보를 보고서 로컬로 불러오고자 하는 파일을 특정할 수 있다. 만일 데이터 변조가 되지 않았다면, 클라우드 서버에서 원본 파일 정보에 대한 해시 작업을 하고서 엣지 서버로 내려 보낸 후, 엣지에서도 마찬가지로 기존에 저장되어 있던 해시 값을 가지고 비교를 했을 때, 서로 일치하면 무결성 검증이 된 것으로 간주하여 사용자에게 다음과 같이 검증 결과를 알려준다.

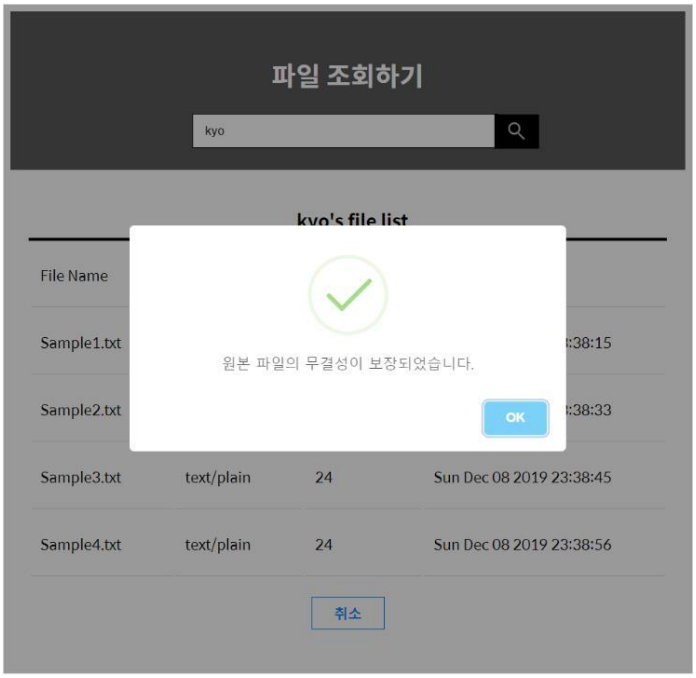

Figure 12 무결성 검증 성공

만일 Figure 10에서와 같이 클라우드 서버에 저장된 파일 원본 내용(클라우드 DB의 fileContents 컬럼)이 제3자에 의해 다음과 같이 변경된다면 파일의 변조가 일어난 것으로 판단하여 사용자에게 알려준다.

userID	timeStamp	hashValue	fileName	fileType	fileSize	fileContents
kyo	Sun Dec 08 2019 23:38:15	3d39aacf1dd0d24542b44718d64a1937a12e32f0c10ff1e5db0a83bcc991dd21	Sample1.txt	text/plain	24	changed
kyo	Sun Dec 08 2019 23:38:33	8f309fac6443bdaa1c7ad4245f1a423329296451f01dac0a6632948fd75891c9	Sample2.txt	text/plain	24	changed
kyo	Sun Dec 08 2019 23:38:45	87f871a314c0cb44f241d3b69678139c3660048af51c76629e7c3ac8c26801dc	Sample3.txt	text/plain	24	changed
kyo	Sun Dec 08 2019 23:38:56	a7358cf8702a738f632495b7fe317f59555ac4a8e0807376f2cd81181b1c7714a	Sample4.txt	text/plain	24	changed

Figure 13 변조된 내용

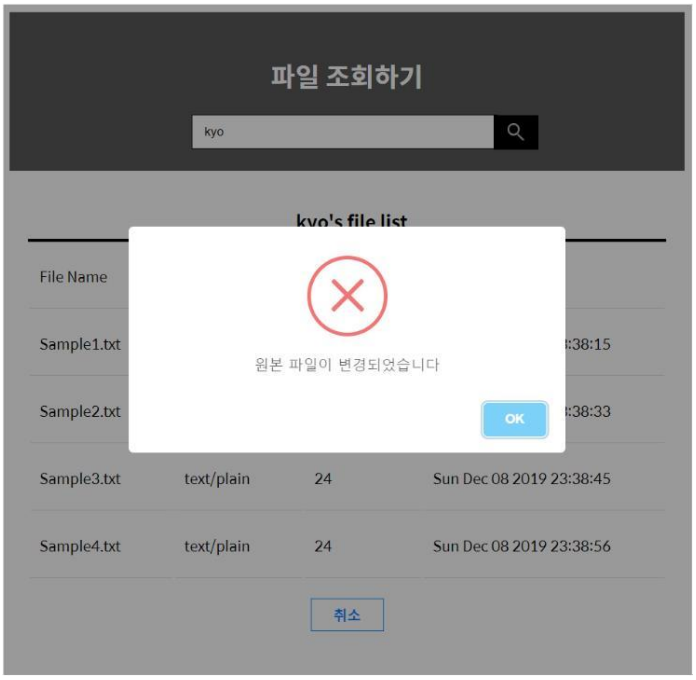

Figure 14 무결성 검증 실패

5. 결론 및 향후 연구

엣지 컴퓨팅 환경에서 엣지 단에서 데이터 처리를 함으로써 클라우드로 중앙 집중된 시스템의 취약점을 보완할 수는 있지만 아직 엣지 단을 위한 보안 정책은 부족한 상황이다. 앞서 언급한 대로 이를 위하여 클라우드와 엣지 부분에서 디바이스의 접근 기록을 효율적으로 관리하는 시스템을 구축하고자 하였다.

엣지 단에 여러 사용자들이 접근하여 많은 데이터가 쌓이게 되어도 클라우드 서버에 적절히 전송함으로써 데이터 과부화를 막을 수 있을 것이다. 단, 엣지 서버에도 소량의 정보를 저장한다.

본 연구에서는 파일 저장소로서의 서버를 가정하고, 하나의 싱글 클라우드 서버와 엣지 서버를 통하여 사용자 데이터를 안전하게 저장하는 작업을 하고자 하였다.

이 연구 결과를 통해 보안 관련 행동에 관한 사건탐지, 보안 위반 사건 분석 후 대응이 가능하며, 이를 위한 추가 서버의 운영이 불필요하다.

하지만 현재 연구 단계의 시나리오는 하나의 싱글 클라우드만으로 운영하여 동작한다. 이를 멀티 클라우드로 구축하여 원본 파일을 분산 저장하여 보관하게 되면 좀 더 정확한 무결성 검증을 할 수 있을 것이다.

또한, 기술 상의 문제로 원본 파일을 다룬 것이 아닌 그저 그 내용물들을 따로 스트링 형태로 나눠서 저장한 한계도 존재한다. 때문에 시연 과정에서 파일 타입은 간단하게 텍스트 파일로 제한하였다.

이러한 저장작업에서의 원본 파일에 대한 비정상적인 접근과 조작을 탐지하는 것에만 목적을 두어 시스템 구축을 하였기에, 그를 방지하거나 복구하는 과정까지는 나아가지 않았다.

마지막으로 파일 무결성(Integrity) 검증에 SHA256 해시 알고리즘만을 사용하여 작업하였다. 해시 값의 비교만으로 완전한 무결성 검증 및 위변조 여부를 파악할 수 없다. 이를 위한 새로운 알고리즘 적용은 앞으로 연구할 과제이다.

참 고 문 헌

[1] Microsoft, “Microsoft Azure IoT Edge – Extending cloud intelligence to edge devices”, May 10, 2017

[2] Forbes, “Amazon Makes Foray into Edge Computing With AWS Greengrass”, Jun 7, 2017

[3] Jim Breithaupt and Mark S. Merkow, 『Information Security: Principles and Practices, 2nd Edition』, Pearson IT Certification