情報保護學會論文誌
第 22 卷 第 1 號, 2012. 2

사용자 데이터 기밀성을 보장하기 위한 클라우드 스토리지 게이트웨이*

김 홍 성,† 김 형 식‡
충남대학교 컴퓨터공학과

A Cloud Storage Gateway to Guarantee the Confidentiality of User Data*

Hong-Sung Kim,† Hyong-Shik Kim‡
Chungnam National University
Department of Computer Science and Engineering

요 약

클라우드 스토리지는 사용자들로 하여금 저장장치를 소유하는 대신 서비스의 형태로 빌려서 사용하고 사용량만큼만 비용을 지불하게 하므로 자체 데이터 센터를 유지하는 것보다 유리한 측면이 많이 있다. 그렇지만 공용 클라우드로 스토리지를 서비스하면 사용자 데이터에 대한 접근을 소유자가 통제하기 어렵기 때문에 데이터에 대한 기밀성을 보장하지 못하는 문제가 발생된다. 본 논문에서는 공용 클라우드 스토리지에 저장되는 사용자 데이터에 대하여 기밀성을 보장하기 위한 목적으로 클라우드와 사용자 사이에 동작하는 게이트웨이를 제안한다. 이 게이트웨이는 사용자의 개입 없이 데이터를 암호화 혹은 복호화하여 전달하며, 다른 게이트웨이를 통한 접근을 보장할 수 있도록 암호 키를 교환하는 기능도 제공한다. 제시된 방법을 상용 클라우드 서비스에서 시험한 결과 안전성과 호환성을 만족할 수 있음을 확인하였다.

ABSTRACT

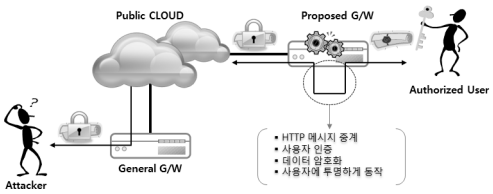
The cloud storage has the client lend and use the device as a form of service rather than owning it, and thus the client pays the charge for the service that he or she actually uses, making it beneficial over the self-managed data center. When the storage service is provided on public cloud, however, the clients does not have any control over the user data, which brings a problem of violating data confidentiality. In this paper, we propose a gateway that works between the public cloud and the client for the purpose of guaranteeing the confidentiality of user data stored in cloud. The gateway encrypts or decrypts, and then delivers the user data without the client's intervention. In addition, it provides the function of exchanging keys to allow the client to access through another gateway. The proposed idea has been tested on a commercial public cloud and verified to satisfy security and compatibility.

Keywords: cloud storage, gateway, confidentiality, proxy service

I. 서 론

클라우드 컴퓨팅에 대한 관심이 증가하고 스마트폰 등 경량화된 단말기가 광범위하게 사용됨에 따라 다양한 형태의 클라우드 컴퓨팅 서비스가 출현하였다. 특히 클라우드 스토리지는 사용자의 수요에 맞춰서 대용

접수일(2012년 1월 5일), 게재확정일(2012년 2월 14일)
* 이 연구는 2011년도 충남대학교 학술연구비에 의해 지원되었음
† 주저자, w1inj4@gmail.com
‡ 교신저자, hkim@cnu.kr



〔그림 1〕 데이터 유출 방지 시나리오

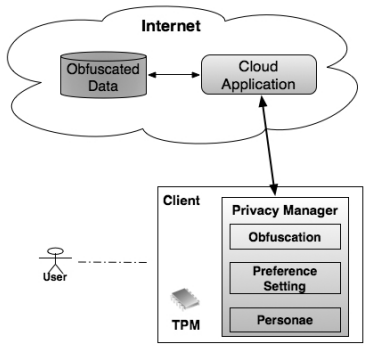
량 파일에 대한 공간을 일시적 혹은 항구적으로 융통성 있게 제공할 뿐만 아니라 사용자의 위치와 상관없이 접근할 수 있도록 지원하기 때문에 호스팅, 웹 하드 형태로 다양화된 서비스가 개발되어 왔다.

그러나 사용자의 데이터를 원격의 클라우드 스토리지에 저장하는 것은 사용자 편의성 측면에서는 괄목할 만한 진전이지만 보안성 측면에서는 몇 가지 심각한 문제점을 노출시켰다〔1〕. 우선 사용자는 자신의 데이터에 대한 통제권한을 행사하는데 한계를 갖게 되었다. 사용자는 자신의 데이터가 저장되는 위치에 대하여 통제할 수 없고 복제되는 내역을 알지 못한다.

둘째, 클라우드에 저장된 데이터에 대한 기밀성을 보장하는데 한계가 있다. 해당 데이터에 접근할 수 없는 사용자가 클라우드에 저장된 데이터를 임의로 활용할 수 있는 가능성이 존재한다. 특히 클라우드 내부에서의 위협은 더 심각한데, 클라우드 내부에 접근할 수 있는 사용자가 클라우드에 저장된 데이터에 접근하는 것을 막는 것은 원천적으로 불가능하다. 따라서 사용자들의 민감한 데이터들이 그대로 노출되어 개인이나 회사에 심각한 피해를 줄 수 있다.

기존의 클라우드 시스템은 사용자 PC에 클라이언트 인터페이스가 설치되고 서비스 제공자는 클라우드 컴퓨팅을 지원하기 위한 플랫폼을 개발하여 서비스 하는 방식을 채택한다. 이를 이용하기 위해서 사용자는 클라우드 시스템으로부터 인증을 받은 후 PC에 설치된 클라이언트 인터페이스를 통해 데이터를 쓰거나 읽는다. 본 논문에서는 사용자 네트워크에 설치되는 게이트웨이가 클라우드 스토리지에 저장되는 사용자 데이터를 암호화하여 전달하고 클라우드 스토리지로부터 받아온 데이터를 복호화함으로써 클라우드 스토리지에 존재하는 보안위협으로부터 사용자의 데이터를 안전하게 보호하는 방법을 제안한다.

〔그림 1〕은 본 논문이 제안하는 클라우드 스토리지 게이트웨이가 데이터 유출을 방지하는 시나리오이다. 원칙적으로 클라우드 스토리지에 저장되는 데이터를 암호화하여 저장하되, 클라우드 스토리지 게이트웨이



〔그림 2〕 사용자 영역에서의 데이터 암호화

로 하여금 인증된 사용자에 한하여 데이터에 대한 암복호화 기능을 제공한다. 즉, 클라우드 스토리지에 저장되는 데이터뿐만 아니라 클라우드 스토리지와 클라우드 스토리지 게이트웨이 사이에 전송되는 데이터도 암호화된 형태이므로 앞에서의 문제점을 극복할 수 있고 사용자 데이터에 대한 기밀성을 보장할 수 있다.

본 논문의 구성은 2장에서 클라우드 스토리지에서 기밀성을 보장하기 위한 관련 연구들을 살펴보고 3장에서 클라우드 스토리지 게이트웨이의 구조와 동작방식을 설명한다. 4장에서 게이트웨이에 대한 기능 시험과 성능 분석 결과를 보인 후 마지막으로 5장에서 마무리한다.

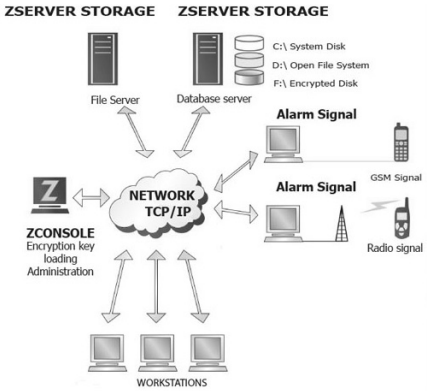
II. 관련연구

클라우드 스토리지에 데이터를 저장할 때 발생되는 문제점들을 해결하기 위한 방안들이 제안되었다〔2〕. 가장 기본적인 방법은 클라우드 스토리지에 저장되는 모든 파일들을 암호화하여 저장하는 것이다.

본 절에서는 사용자 데이터를 암호화하는 방법으로서 대표적인 두 가지 방법을 소개한다. 첫 번째 방법은 사용자 영역에서 데이터를 암호화하는 것이다.

〔그림 2〕는 사용자 영역에서 클라우드 스토리지로 파일을 업로드할 때 암호화하는 사례를 보인다〔3〕. 시스템의 기본적인 구성은 클라이언트 프로그램과 서버 프로그램으로 나누어져 있고, 클라이언트 프로그램에 암호화 또는 난독화 모듈, 암호화적용에 관한 설정모듈이 포함한다.

TPM을 통해 암호화 키를 안전하게 저장할 수 있고, 암호화 알고리즘을 적용하므로 데이터를 안전하게 저장할 수 있지만, 암호화 기능을 제공하도록 모든 클라이언트 프로그램을 다시 구성해야 하는 문제점이 있



〔그림 3〕 클라우드 영역에서의 데이터 암호화

고, 클라이언트 프로그램이 암호화하지 않은 상태로 데이터를 저장하는 것을 방지할 수 없다.

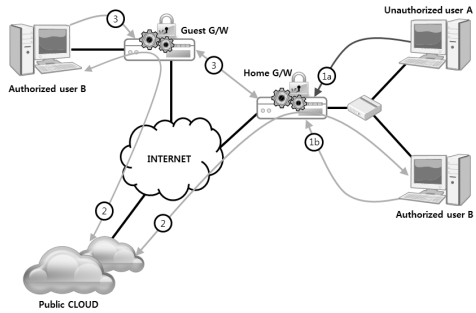
두 번째 방법은 서버 영역에서 사용자 데이터를 암호화하여 저장하는 것이다. 〔그림 3〕은 클라우드 영역에서 데이터를 암호화 하는 모습으로 상용 솔루션에서 채택한 시스템 구조이다[4].

크게 세 개의 컴포넌트로 구성되어 있는데 각각은 Zserver Storage, Zconsole, Alarm Module이다. Zserver Storage에 쓰는 모든 파일들은 사용자에 투명하게 암호화 되어 저장되고, 읽는 경우는 복호화되어 제공된다. 암호화 알고리즘, 암호화키 관리 등 여러 가지 설정에 관한 운용은 Zconsole에서 이루어지며 사용자 네트워크의 관리자만이 접속할 수 있도록 한다. Alarm Module은 명령 메시지가 담긴 시그널을 클라우드 서버로 전송하여 암호화된 특정 파일에 대한 접근을 제어하거나 서버를 리부팅한다. 이 방법에서도 암호화 알고리즘이 클라우드에서 실행되고 암호화 키 정보도 클라우드에서 관리되기 때문에 내부적인 위협이 여전히 존재한다.

본 논문의 접근 방식은 클라우드 영역을 신뢰할 수 없다는 전제에 맞춰 서버 영역에서 암호화하는 것을 배제하고, 클라우드 스토리지에 저장되는 데이터의 기밀성을 위하여 클라이언트 프로그램을 변형하는 것도 또한 배제한다. 또한 클라우드 스토리지에 대한 사용 방식을 바꿀 필요가 없도록 투명성을 보장한다는 점에서 기존 연구와 구분된다.

III. 클라우드 스토리지 게이트웨이

클라우드 스토리지 게이트웨이는 클라이언트가 속한 네트워크에 설치되며 클라우드로 향하는 트래픽과



〔그림 4〕 게이트웨이 동작 사례

클라우드로부터의 트래픽을 구분하여 처리한다. 즉, 클라우드 스토리지로 접근하는 트래픽을 인지하여 스토리지에 저장될 내용을 암호화하며, 클라우드 스토리지로부터의 트래픽에서 데이터를 복호화한다.

게이트웨이가 암복호화를 통하여 클라우드 스토리지에서의 기밀성을 보장하기 위해서는 해결해야 하는 과제는 다음과 같다. 우선 인증된 사용자에 대해서만 암복호화 키를 제공해야 한다. 이것은 인증되지 않은 사용자가 복호화하는 것을 방지해야 함을 의미한다. 본 논문에서의 게이트웨이는 사용자 인증 서비스를 제공한다.

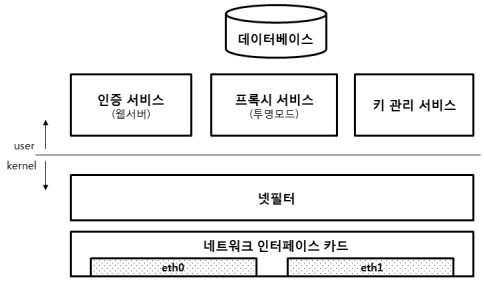
두 번째 과제는 사용자가 다른 네트워크로 이동하였을 때, 즉 데이터를 저장할 때 활용했던 게이트웨이가 아닌 다른 게이트웨이를 통해서도 해당 데이터에 접근할 수 있어야 한다. 이 과제를 해결하기 위하여 본 논문에서는 암복호화 목적으로 대칭키를 쓰고 인증된 사용자가 요청할 때는 다른 게이트웨이에서도 해당 키를 제공할 수 있도록 고려하였다.

본 장에서는 게이트웨이의 동작 방식을 먼저 설명하고 전체 시스템의 구성을 기능 관점에서 세 부분으로 나눠서 설명한다.

3.1 동작 방식

〔그림 4〕는 게이트웨이의 동작 방식을 설명하기 위한 것이다. 사용자는 클라우드 스토리지에 접근하기 전에 게이트웨이로부터 인증받아야 한다. 게이트웨이로부터 인증 받지 못한 사용자는 클라우드 시스템에 접근할 수 없다(1a). 인증을 받은 사용자(1b)가 클라우드에 데이터를 저장 할 때에는 게이트웨이에서 이를 감지하고 파일을 암호화하여 저장한다(2).

사용자의 이동성을 지원하기 위해서는 다른 게이트웨이 사이에 암호화키를 공유할 수 있어야 한다. 사용



〔그림 5〕 게이트웨이 기본 컴포넌트 구성

자가 최초 가입한 네트워크 지역을 홈 게이트웨이 영역이라고 한다면 같은 사용자가 다른 네트워크에서 게이트웨이를 사용할 때 홈 게이트웨이로부터 인증에 필요한 데이터와 암호화키를 받고(3) 인증을 받은 후 클라우드 스토리지에 접근한다(4).

3.2 시스템 구조

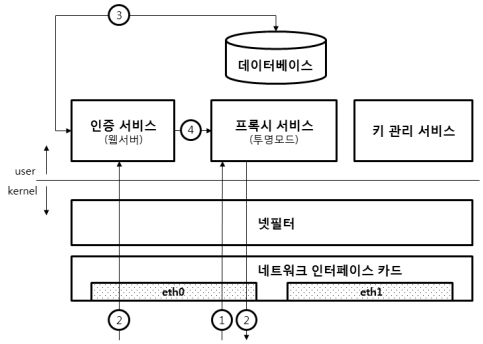
클라우드 스토리지 게이트웨이는 클라이언트와 클라우드 사이에서 트래픽을 중계하기 때문에 두 개의 네트워크 포트가 필요하다. 또한 패킷 수준에서 트래픽을 효과적으로 처리할 수 있는 메커니즘이 필수적이다. 본 논문에서의 게이트웨이는 리눅스에서 동작하도록 구성되었고, 넷필터(netfilter)라는 자체 방화벽을 이용하여 패킷 처리의 효율성을 만족시킨다.

〔그림 5〕는 게이트웨이의 구성도이다. 크게 인증서비스 모듈, 프록시 서비스 모듈, 키관리 서비스 모듈과 넷필터로 나눌 수 있다. 이때 넷필터에서는 내부 네트워크로부터 받은 트래픽 중에 목적지 포트가 80번인 패킷들만 프록시 서버가 받을 수 있도록 올려주고 나머지는 다른 네트워크 포트를 통해서 통과시킨다.

인증 서비스는 클라우드 서버로 접근하는 사용자의 IP 주소를 기반으로 인증된 사용자인지 확인한다. 만약 인증된 사용자라면 해당 요청을 그대로 수행하고 그렇지 않은 경우에는 인증 받을 수 있도록 유도하여 클라우드 시스템의 접근을 제한한다.

처음 게이트웨이를 통하여 클라우드에 접근하는 사용자에 대해서는 인증 서비스가 사용자의 정보를 입력받고 암호화 키를 생성한다. 그 다음 프록시 서비스에서 암호화를 수행할 수 있도록 사용자의 IP 주소와 암호화 키를 전송하여 세션을 유지한다.

프록시 서비스는 사용자가 요청한 HTTP 요청 메시지를 메모리에 저장하였다가 접근하려고 하는 웹 서



〔그림 6〕 사용자 인증 과정

버와 TCP 연결 후 대신 전송함으로써 사용자와 웹서버 사이에서 중계 역할을 한다. 또한 클라우드에 파일을 업로드할 때는 암호화를 수행하고, 반대로 다운로드할 경우에는 복호화를 수행한다.

키 관리 서비스는 게이트웨이 간의 키 교환을 지원하기 위한 기능으로써 특정 사용자의 아이디를 받으면 데이터베이스에서 해당 사용자의 암호화 키에 대한 정보를 전송함으로써 게이트웨이 간에 암호화 키를 공유하는 역할을 담당한다.

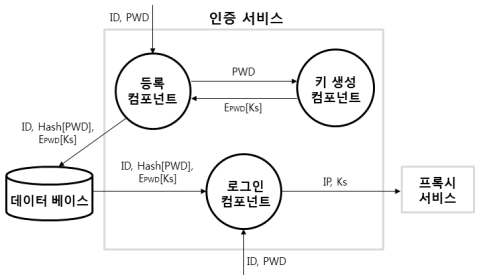
3.2.1 사용자 인증 서비스

클라우드 스토리지에 접근하려는 시도가 탐지되면 〔그림 6〕과 같은 사용자 인증 절차가 시작된다.

우선 프록시 서버로 들어온 메시지를 분석하여 URL 주소가 등록된 클라우드 서비스라면 세션정보를 통해 인증된 사용자 여부를 확인한다(①). 인증된 사용자가 아니면 URL 리디렉션을 통해 게이트웨이의 인증서버로 연결하여 사용자가 인증을 받을 수 있도록 유도한다(②). 데이터베이스에 등록되지 않은 사용자에 대해서는 사용자 등록과정이 필요하다(③). 인증이 된 후에는 암복호화에 필요한 정보를 프록시 서비스에게 전달한다(④).

이미 인증된 사용자가 접근하는 경우라면 이미 확보한 정보를 이용하여 데이터를 암복호화함으로써 사용자와 클라우드 사이의 데이트 흐름을 중계한다.

〔그림 7〕은 인증 서비스에 필요한 컴포넌트들이 주고받는 데이터들을 나타낸다. 등록 컴포넌트는 게이트웨이에 최초 접근하는 사용자를 데이터베이스에 등록하는 컴포넌트이다. 사용자로부터 아이디와 패스워드를 입력받아서 패스워드만 키 생성 컴포넌트에 전달한다. 키 생성 컴포넌트에서는 프록시 서비스에서 사용



〔그림 7〕 인증 서비스 구성

할 비밀키를 생성한다. 원칙적으로 대칭키 암호화 알고리즘을 사용되는데, 랜덤하게 생성한 숫자를 16진수 스트링으로 변환하여 비밀키로 사용한다. 생성된 비밀키는 패스워드기반 암호화 알고리즘으로 암호화되어 다시 등록 컴포넌트로 전달되고 최종적으로 아이디, 패스워드 그리고 비밀키가 데이터베이스에 저장된다.

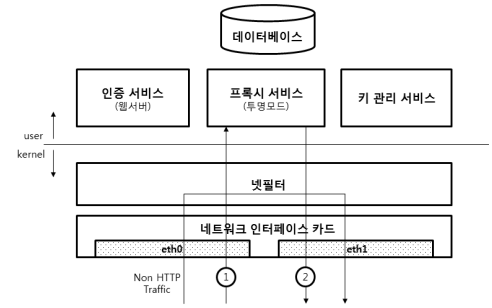
로그인 컴포넌트는 이미 가입된 사용자들로부터 아이디와 패스워드를 입력받아 데이터베이스에 저장된 아이디와 패스워드를 통해 인증을 처리한다. 인증에 성공하면 인증된 사용자의 세션을 유지하기 위해 암복호화에 필요한 비밀키와 사용자의 IP 주소를 프록시 서비스에게 전달하여 클라우드 서비스를 접근하는 사용자를 제어하고 데이터를 암호화하기 위한 목적으로 사용한다. 이 때 세션은 일정 시간 범위 내에서 지속되며, 동일한 IP 주소에서 송수신하는 트래픽은 동일한 사용자가 사용하는 것으로 가정한다.

3.2.2 프록시 서비스

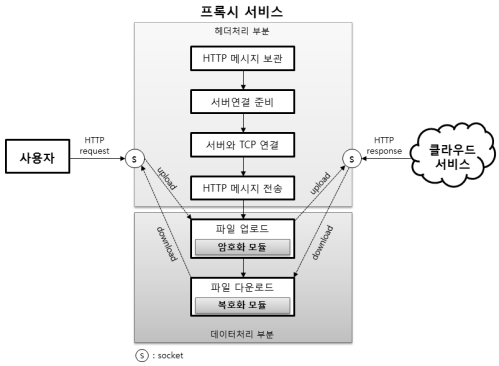
프록시 서비스는 사용자와 클라우드 스토리지 사이에서 패킷을 중계하는 역할을 수행한다. 〔그림 8〕은 프록시 서비스를 통해 메시지가 중계되는 과정을 나타낸다.

프록시 서비스는 넷필터를 통해서 HTTP 요청 메시지를 받고 그 외의 트래픽은 다른 네트워크 포트로 통과시킨다(①). 들어온 메시지를 분석하여 다양한 기능들을 적용할 수 있는데, 본 논문에서는 클라우드 스토리지로 업로드하거나 다운로드하는 경우에 암호화 또는 복호화하는 기능을 적용한다(②).

〔그림 9〕는 프록시 서비스가 사용자의 HTTP 요청 메시지를 처리하는 과정을 나타낸다. 소켓을 통해 들어온 데이터를 통해 HTTP 요청 메시지의 헤더 정보를 메모리에 저장한다. 그 다음 저장된 원본 헤더정보



〔그림 8〕 메시지 중계 과정



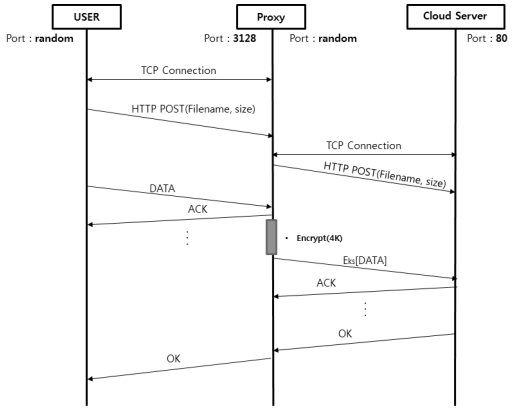
〔그림 9〕 프록시 서비스 구성

를 참조하여 클라우드 서버에 접속하기 위한 최소한의 필수 정보를 구성한 후 클라우드 서버와 TCP 세션을 맺고, 메모리에 저장되어 있던 사용자의 요청 메시지 내용들을 클라우드 서버에 모두 전송한다.

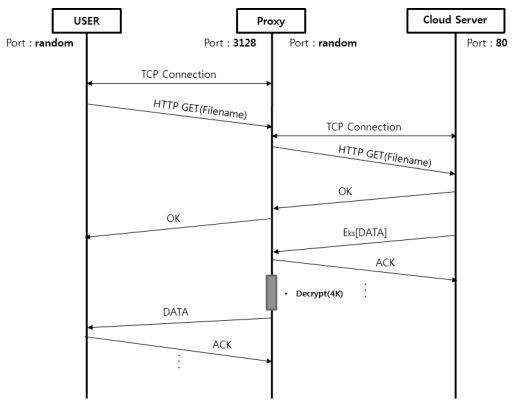
3.2.3 암호화

게이트웨이가 중계하는 파일을 암호화하는 과정은 크게 세 부분으로 나뉘어 볼 수 있다. 첫 번째는 업로드시 발생되는 HTTP 요청 메시지를 감지하는 일이다.

〔그림 10(a)〕는 인터넷을 통해 파일을 업로드하는 과정을 나타낸다. HTTP POST 메시지에는 파일의 이름과 크기 정보가 포함되어 있다. 먼저 프록시 서버와 소켓통신을 위해 TCP 세션을 맺고 HTTP POST 메시지를 보내면 프록시 서버는 특정 포트로 수신하게 된다. 그 다음 클라우드 스토리지와 TCP 세션을 구성하고 메모리에 저장되어 있는 사용자의 메시지를 다시 전송한다. 이 후 클라이언트로 소켓으로부터 들어오는 데이터는 업로드될 파일이므로 데이터가 들어오면 일정 단위로 버퍼에 저장하였다가 암호화



(a) 파일 업로드 요청 과정



(b) 파일 다운로드 요청 과정

〔그림 10〕 파일 처리 과정

하여 서버 소켓에 전송한다.

다운로드 메시지도 특정 키워드를 확인하는 방법으로 감지할 수 있다. 〔그림 10(b)〕은 클라우드 스토리지로부터 파일을 다운로드하는 과정을 나타낸다. 먼저 사용자는 프록시 서버와 TCP 세션을 맺고 소켓통신을 통해 HTTP GET 메시지를 전송한다. 클라이언트 소켓을 통해 들어온 메시지를 수신한 프록시 서버는 해당 메시지를 통해 클라우드 스토리지와 TCP 세션을 구성하고 메모리에 저장되어 있던 사용자의 HTTP GET 메시지를 다시 전송한다. 클라우드 스토리지로부터 OK 메시지를 받으면 다시 사용자에게 전달하고 이후 클라우드 스토리지는 실제 파일 전송을 시작한다. 프록시 서버로 들어오는 파일 데이터를 암호화 단위와 마찬가지로 일정길이씩 버퍼에 저장하여 복호화한 후에 클라이언트 소켓으로 보내면 다운로드 과정이 끝난다.

암호화를 위한 두 번째로 과정은 소켓을 통해 들어오는 파일의 데이터를 암호화하는 것이다. 일반적으로 받은 만큼 다시 전송하는 것이 아니라 암호화 단위만큼 모은 후 암호화하여 전송해야 하고, 클라이언트가 보내는 데이터에는 전송을 위한 메타 데이터가 파일의 앞뒤로 붙어서 전송되는 경우도 있기 때문에 실제 데이터만 암호화하기 위한 방법이 필요하다.

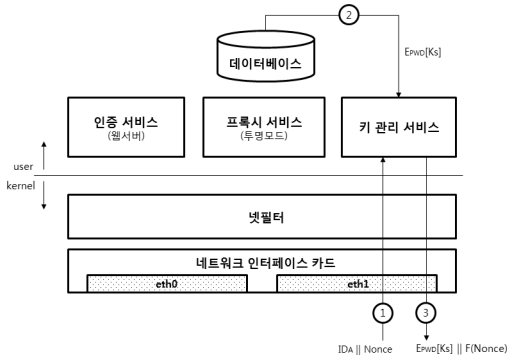
암호화를 위한 마지막 과정은 키 관리다. 암호화를 위한 사용자 키는 사용자의 IP 주소와 함께 세션 정보로 관리된다. 프록시 서버는 세션을 생성하고 해당 세션에 대해서 타이머를 설정한다. 인증된 사용자가 프록시 서버에 접근하여 클라우드 서버로 파일을 업로드하거나 다운로드할 경우 해당 세션에 설정된 타이머를 중지시키고 암복호화를 수행하고, 처리가 완료되면 해당 세션에 대해 다시 타이머를 동작시킨다. 타이머를 통하여 일정 시간이 경과되었음을 인지하면 해당 세션은 종료되고 클라우스 스토리지에 접근하기 위해서는 인증 절차부터 다시 필요하다.

3.3 게스트 게이트웨이 지원

클라우드 스토리지에 대한 인증을 위해서 사용자의 아이디와 패스워드가 저장된 게이트웨이를 홈 게이트웨이라 정의하고 다른 네트워크에 존재하는 게이트웨이를 게스트 게이트웨이라 부른다. 사용자의 이동성을 보장하기 위해서는 사용자가 홈 게이트웨이뿐만 아니라 게스트 게이트웨이를 통하여 클라우드 스토리지에 접근할 수 있어야 한다. 즉 사용자가 게스트 게이트웨이를 통하여 파일을 업로드 혹은 다운로드할 때도 홈 게이트웨이가 암호화에 필요한 인증 정보와 키 정보를 게스트 게이트웨이에게 제공하여야 한다.

게이트웨이의 위치 정보를 포함시키는 방법으로 본 논문에서는 IP 주소를 아이디에 부가하는 방법을 채택한다. 게이트웨이가 사용자 아이디를 입력받으면 아이디에 내재된 IP 주소로부터 홈 게이트웨이를 식별하게 된다.

〔그림 11〕은 게스트 게이트웨이를 사용하는 사용자를 인증할 수 있도록 지원하는 과정이다. 홈 게이트웨이 사용자는 데이터베이스에 저장된 자신의 정보를 통해 인증이 가능하지만 게스트 게이트웨이에는 데이터베이스에 저장된 정보가 없으므로 입력받은 아이디(IDA)를 홈 게이트웨이로 Nonce와 함께 전송하면 (①) 홈 게이트웨이는 데이터베이스에서 아이디를 검색하여(②), 사용자의 패스워드로 암호화되어 저장된 비밀키($E_{pwd}[K_s]$)를 획득하고 이것을 게이트웨이간



〔그림 11〕 게스트 게이트웨이 지원과정

의 약속된 함수를 통해 생성된 Nonce 값과 함께 게스트 게이트웨이로 전송한다(③).

게스트 게이트웨이는 홈 게이트웨이로부터 수신한 비밀키 정보($E_{pwd}[K_s]$)를 사용자가 입력한 패스워드로 복호화하고 비밀키를 획득한다. 만약 복호화에 실패한다면 사용자 인증을 실패로 처리한다. 또한 Nonce 값이 서로간의 약속되어 있는 함수를 거쳐 변환된 올바른 값인지 확인한다. Nonce 값을 확인하는 이유는 장치간의 신뢰성을 보장하고 리플레이 공격을 방지하기 위함이다. 이제 게스트 게이트웨이는 클라우드 스토리지와의 파일 송수신에 비밀키를 적용할 수 있다.

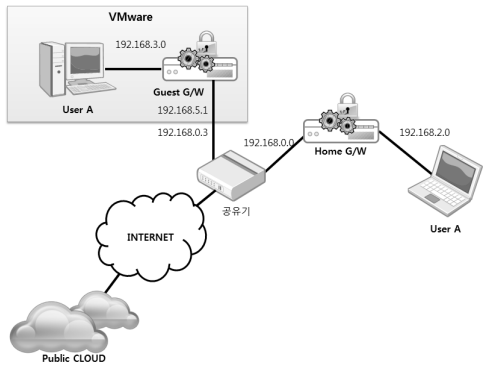
IV. 시험

게이트웨이에 대하여 기능 시험과 성능 시험을 수행하였다. 우선 게이트웨이의 기능을 시험하기 위해 실제 게이트웨이를 통과한 데이터가 암호화되어 저장되는지 확인하고 암호화되었던 파일을 다운로드하여 복호화했을 때 정상적으로 복호화되는지를 시험하였다.

4.1 시험환경

〔그림 12〕는 게이트웨이의 기능을 시험하기 위한 네트워크 구성이다. 공유기를 통해 게이트웨이를 중심으로 서로 다른 서브넷을 구성하였고 사용자는 게이트웨이를 통해서만 클라우드 스토리지에 접근할 수 있다.

또한 복수개의 게이트웨이가 존재하는 경우를 시험하기 위해 게스트 게이트웨이를 구성하였다. 공유기는 보통 192.168.0.0 네트워크만 라우팅을 할 수 있으



〔그림 12〕 시험을 위한 네트워크 구성

므로 192.168.2.0 네트워크로 들어오는 패킷을 라우팅 할 수 있도록 라우팅 테이블을 수정하였다. 클라우드 서비스로는 네이버에서 제공하는 Ndrive를 사용하였다.

4.2 기능 시험

사용자가 업로드하는 파일은 게이트웨이를 통과하면서 암호화되어 클라우드 시스템에 저장된다. 파일의 기본적인 속성만으로는 결과를 확인 할 수 없으므로 Winhex라는 도구를 사용하여 실제 파일의 내용이 암호화되는지 확인하였다.

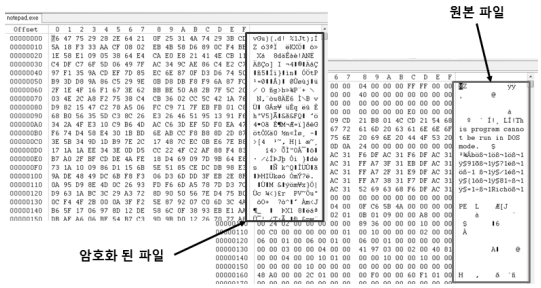
〔그림 13(a)〕는 Winhex 도구로 업로드하기 전의 원본 파일과 암호화환 게이트웨이가 아닌 다른 게이트웨이를 통해 다운로드한 파일의 내용을 비교하는 것이다. 파일은 읽거나 실행할 수 없도록 암호화 되어 있음을 확인하였다.

게이트웨이를 통해 업로드된 파일을 다시 다운로드하였을 경우 정상적으로 사용이 가능한지 호환성을 확인하였다. 〔그림 13(b)〕는 클라우드에 암호화되어 저장되었던 파일을 다시 다운로드하였을 때 정상적으로 복호화되는지 원본 파일과 대조하기 위한 것이다. 암호화된 파일이 게이트웨이를 거쳐 복호화되었을 때 그 내용이 원본 파일과 동일함을 확인할 수 있었다.

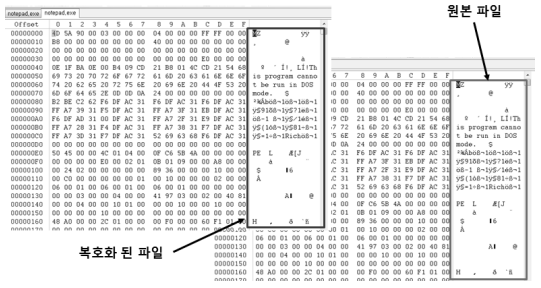
4.3 오버헤드 측정 시험

게이트웨이의 핵심 기능은 암복호화를 위한 프록시 서비스 기능이다. 클라우드 스토리지와 주고받는 모든 데이터가 암복호화되어야 하기 때문에 암복호화에 수반되는 오버헤드를 측정하였다.

〔그림 14(a)〕는 파일을 업로드할 때 암호화하는 경

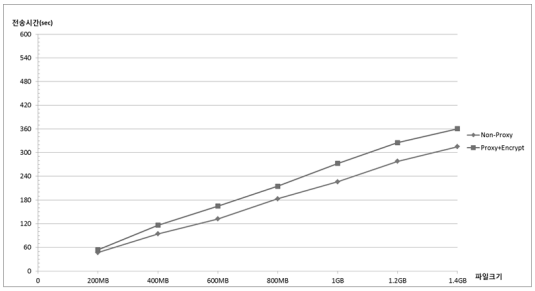


(a) 안전성 시험 결과

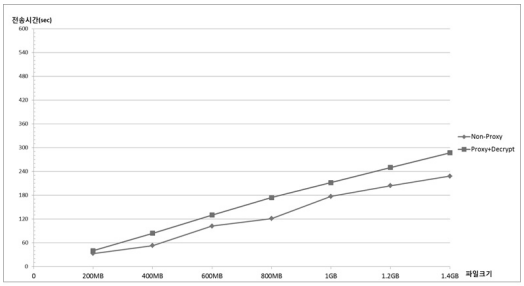


(b) 호환성 시험 결과

〔그림 13〕 Winhex 도구를 이용한 기능 시험



(a) 암호화 오버헤드 측정 시험 결과



(b) 복호화 오버헤드 측정 시험 결과

〔그림 14〕 오버헤드 측정 시험

우를 그렇지 않는 경우와 비교한 그래프이다. 파일크기를 1.4G 바이트까지 증가시키면서 클라우스 스토리지까지의 전송시간을 측정하였다. 암호화에 수반되는 오버헤드는 전송시간 대비 최대 16% 정도였는데, 이는 1.4G 바이트 파일을 전송할 때 약 50초 정도의 지연시간이 발생됨을 의미한다.

〔그림 14(b)〕는 파일을 다운로드할 때 발생하는 오버헤드를 보인다. 복호화를 할 때도 암호화와 비슷한 수준의 오버헤드를 관찰할 수 있었는데, 1.4G 바이트 파일에 대하여 오버헤드는 약 60초 정도로 측정되었다.

V. 결 론

사용자 데이터를 클라우드 스토리지에 저장할 때 기밀성을 보장하지 못한다는 점은 클라우드 스토리지의 활용 범위를 공개 가능한 파일로 제한하는 문제점을 유발했었다. 본 논문에서는 이러한 한계를 극복하기 위한 방법으로 클라우드 스토리지 게이트웨이를 제안하였고, 이 게이트웨이가 클라우드 스토리지나 사용자의 개입없이 사용자 데이터에 대한 기밀성을 보장할 수 있음을 보였다. 클라우드 스토리지 게이트웨이는 클라우드 스토리지와 사용자 사이에서 데이터를 중계할 뿐만 아니라 사용자 인증에 대한 기능도 제공한다.

클라우드 스토리지에 저장된 파일은 암호화되므로 게이트웨이를 경유하는 경우만 정상적으로 읽거나 실행할 수 있었고, 게이트웨이를 경유하지 않는 사용자는 비록 파일을 다운로드하더라도 사용할 수 없다. 또한 클라우드 스토리지 외부에서 암호화되기 때문에 클라우드 스토리지 내부의 위협으로부터도 사용자 데이터를 보호할 수 있다. 또한 게이트웨이 간에 암호화를 위한 비밀키를 공유할 수 있는 메커니즘을 도입함으로써 클라우드 스토리지 게이트웨이는 사용자의 이동성도 보장할 수 있도록 설계되었다.

사용자 데이터의 암복호화에는 키가 필수적이므로 클라우드 스토리지 게이트웨이의 안전성을 담보하기 위해서는 구체적인 키 관리 메커니즘이 도출되어야 한다. 본 논문에서는 클라우드 스토리지 게이트웨이의 동작성을 검증하는데 초점을 맞추어 키 관리 메커니즘에 대한 것은 최소한의 요건만을 만족하도록 한정하고 구체적인 내용은 고려하지 않았으므로 이 부분에 대해서는 추후 개선할 예정이다.

참고문헌

〔1〕 Manny Siddiqui, “Cloud Computing Secu-

rity," pp. 1-6, Jul. 2011.
〔2〕 Brandel Mary, "Cloud security in the real world: 4 examples", http://www. csonline.com/article/print/596820, June 15, 2010.
〔3〕 HP Labs, "A Privacy Manager for Cloud Computing", pp. 1-12, Nov. 2009.
〔4〕 Zecurion, "Zserver Storage", http://www.zecurion.com/zserver/architecture-server-data-encryption.php

〈著者紹介〉

김 홍 성 (Hong-Sung Kim) 학생회원
2010년 2월: 충남대학교 전기정보통신공학부 졸업
2012년 2월: 충남대학교 컴퓨터공학과 석사
〈관심분야〉 시스템보안, 클라우드컴퓨팅, 해킹

김 형 식 (Hyong-Shik Kim) 종신회원
1988년 2월: 서울대학교 컴퓨터공학과 졸업
1997년 8월: 서울대학교 컴퓨터공학과 박사
1999년 9월 ~ 현재: 충남대학교 컴퓨터공학과 교수
2005년 ~ 2009년: 한국정보보호학회 논문지 편집위원
2011년 ~ 현재: 한국정보보호학회 이사
〈관심분야〉 인터넷보안, 컴퓨터시스템구조, 클라우드컴퓨팅