情報保護學會論文誌
第20卷 第5號, 2010. 10

클라우드 컴퓨팅 환경에서 보안성 향상을 위한 로컬 프로세스 실행 기술*

김 태 형,[1†] 김 인 혁,[1] 김 정 한,[1] 민 창 우,[2] 김 지 홍,[2] 엄 영 익[1‡]
[1]성균관대학교 전자전기컴퓨터공학과, [2]성균관대학교 휴대폰학과

Security-Enhanced Local Process Execution Scheme in Cloud Computing Environments*

Taehyoung Kim,[1†] Inhyuk Kim,[1] Junghan Kim,[1] Changwoo Min,[2]
Jee-hong Kim,[2] Young Ik Eom[1‡]
[1]Department of Electrical and Computer Engineering, Sungkyunkwan University,
[2]Department of Mobile Systems Engineering, Sungkyunkwan University

요 약

기존 클라우드 환경에서는 어플리케이션의 수행이 원격지 클라우드 서버에서 이루어지며, 해당 어플리케이션이 사용하는 물리 메모리, CPU 등 컴퓨팅 자원 역시 원격지 클라우드 서버에 존재한다. 따라서 원격지 서버가 보안 위협에 노출될 경우, 해당 환경에서 실행되는 모든 어플리케이션들이 보안 위협에 의해 영향 받을 수 있다. 특히, 악의적인 클라우드 서버 관리자에 의한 보안 위협으로 인하여, 비즈니스 모델로 많은 이점이 있음에도 불구하고 개인 및 기업 환경에서 클라우드 서비스를 도입하는데 어려움을 겪고 있다. 본 논문에서는 기존 클라우드 컴퓨팅 환경의 취약점을 근본적으로 해결할 수 있는 로컬 프로세스 실행 기술을 제안한다. 이를 통해, 기밀 데이터를 클라우드 서버가 아닌 로컬에 둘 수 있게 되어, 클라우드 서버에서 발생 가능한 보안 위협으로부터 기밀 데이터를 보호할 수 있게 된다. 즉, 보안 데이터를 사용하는 어플리케이션의 경우, 로컬 프로세스 실행 기술을 이용하여 원격지 서버가 아닌 로컬의 컴퓨팅 자원을 사용하여 실행할 수 있도록 설계한다. 이에 따라 보안 프로세스가 사용하는 자원들이 물리적으로 로컬에 존재하므로 원격지 서버의 취약성 문제를 해결할 수 있다.

ABSTRACT

In the current cloud environments, the applications are executed on the remote cloud server, and they also utilize computing resources of the remote cloud server such as physical memory and CPU. Therefore, if remote server is exposed to security threat, every applications in remote server can be victim by several security-attacks. Especially, despite many advantages, both individuals and businesses often have trouble to start the cloud services according to the malicious administrator of the cloud server. We propose a security-enhanced local process executing scheme resolving vulnerability of current cloud computing environments. Since secret data is stored in the local, we can protect secret data from security threats of the cloud server. By utilizing computing resource of local computer instead of remote server, high-secure processes can be set free from vulnerability of remote server.

Keywords: Local Process Execution, Cloud Computing, Cloud Security, Secret Data, TPM

접수일(2010년 06월 30일), 수정일(1차: 2010년 10월 05일), 게재확정일(2010년 10월 14일)
* 본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음 (NIPA-2010-(C1090-1021-0008))
† 주저자, bigkth@gmail.com
‡ 교신저자, yieom@ece.skku.ac.krr

I. 서 론

다양한 소프트웨어/하드웨어 가상화 기술의 발전으로 서버 환경에 주로 적용되던 가상화 기술이 일반 사용자 환경에까지 적용되면서 최근 클라우드 컴퓨팅 기술이 주목받고 있다. 클라우드 컴퓨팅은 네트워크를 통하여 고도의 확장성 및 유연성이 확보된 IT 자원을 서비스 형태로 제공하는 새로운 컴퓨팅 환경으로써 현재 가장 주목받는 IT 기술 중의 하나이다[1].

클라우드 컴퓨팅은 기존의 그리드 컴퓨팅, 가상화, 유틸리티 컴퓨팅, 분산 컴퓨팅, 고성능 컴퓨팅 기술을 융합 발전시킴으로써 차세대 컴퓨팅 환경의 변화를 주도하고 있다. 하지만 현존하는 다양한 컴퓨팅 기술들이 융합된 클라우드 플랫폼은 운영체제, 네트워크, 하이퍼바이저, 악성코드 등에 의한 다수의 보안 결함에 노출되고 있으며, 이 밖에도 관리자에 의한 보안 위협 등 가상 머신 관리에 있어서 잠재적인 보안문제를 안고 있는 등 보안 취약성에 대한 해결 방법이 부재한 상황이다. 따라서 안전한 클라우드 컴퓨팅 환경을 구축하기 위해서는 개인 정보 보호, 클라우드 플랫폼 취약성 등의 새로운 보안 요구 사항에 대한 대책이 필요하다[2].

이를 위해 본 논문에서는 클라우드 환경에서 보안성 향상을 위한 로컬 프로세스 실행 기술을 제안한다. 기존 클라우드 컴퓨팅 환경에서는 프로세스들이 클라우드 서버에서 동작하면서 악성코드에 의한 코드 변환 위협, 네트워크 스니핑에 의한 개인 정보 노출, 기존 운영체제의 취약점에 의한 위협 등 다양한 보안 위협에 노출되기 쉽다. 특히, 현재 클라우드 환경에서 해결하지 못하고 있는 문제 가운데 가장 이슈화 되고 있는 문제는 관리자에 의한 기밀 데이터 유출이다. 관리자는 서버내의 모든 리소스에 접근 가능하며, 메모리, 네트워크 모니터링을 통해 가상머신 상의 동작 과정 추적 및 대부분의 정보에 대한 접근이 용이하다. 따라서 관리자가 악의적인 목적으로 클라우드 시스템을 접근할 경우, 대부분의 기밀 데이터는 손쉽게 유출될 수 있다. 또한, 기밀 데이터가 암호화 기법에 의해 보호받고 있다고 하더라도, 암호화 기법에 대한 복호화 기법이 수 년 이내에 발견될 수도 있기 때문에 암호화된 기밀 데이터의 노출 또한 클라우드 서비스에 대한 신뢰도를 떨어뜨리는 중요 요소 가운데 하나이다. 이러한 이유로 많은 대기업들이 클라우드 컴퓨팅 환경의 다양한 장점에도 불구하고 회사의 기밀 데이터 관리 어려움 때문에 클라우드 시스템 도입을 늦추고 있다. 따라서 제안 기법에서 보안 데이터를 사용하는 프로세스들은 보안 프로세스로 구분하여 클라우드 서버가 아닌 로컬에서 본인의 컴퓨팅 자원을 활용하여 실질적인 프로세스 실행이 이루어지도록 설계한다.

이에 따라 클라우드 외부에서 서비스를 제공받는 사용자 측면에서는 보안 프로세스가 일반 프로세스와 마찬가지로 클라우드 서버에서 실행이 이루어지는 것처럼 보이지만, 보안 프로세스의 중요 데이터 및 코드들은 모두 로컬의 컴퓨팅 자원을 활용하여 실행됨으로써 기존의 클라우드 서버의 자원을 사용함에 따라 발생 가능한 많은 보안 위협으로부터 벗어날 수 있게 된다. 특히, 로그인 또는 보안 접속을 위한 개인키 등 보안 데이터들을 로컬 저장 장치에 저장함으로써 클라우드 서버로 부터의 연관성을 근본적으로 차단하여 관리자에 의한 보안 위협 등으로부터 벗어날 수 있게 된다. 기밀 데이터의 보안 상태는 크게 세 단계로 나눌 수 있다. 첫 번째, 기밀 데이터가 악의적 공격자에 의해 손쉽게 접근될 수 있는 상태로 보안성이 매우 취약한 단계이다. 두 번째는 암호화 기법에 의해 기밀 데이터가 보호되는 상태로 암호화된 데이터만 노출되므로 기법이 유효할 때까지는 일정 수준 보안성이 유지되는 단계이다. 세 번째는 기밀 데이터가 보안 위협에 노출되지 않는 상태로 안전하게 보관되며 가장 안전한 보안 단계이다. 제안 기법은 세 단계 중 가장 강력한 세 번째 보안 상태를 제공하며, 클라우드 서버에서 관리자를 포함한 악의적 공격으로부터 기밀 데이터를 완전하게 격리시켜 보호할 수 있다.

제안 기법은 클라우드 환경에서 발생 가능한 모든 보안 문제 가운데 클라우드 서버에서 기밀 데이터 노출에 의한 보안 취약점을 해결하는데 초점을 맞추며, 클라우드 로컬 및 네트워크에서 발생 가능한 보안 취약점 문제들은 기존 시스템 보안 분야에서 충분히 연구되고 보완되었다고 가정하고 본 논문에서는 기본적으로 네트워크 및 로컬 시스템은 보안상 안전하다고 가정하였다.

본 논문의 구성은 다음과 같다. 2장에서 클라우드 컴퓨팅의 아키텍쳐와 클라우드 환경에서의 보안 위협을 소개한다. 그리고 3장에서는 클라우드 컴퓨팅 환경에서 보안성 향상을 위한 기존의 기법들을 살펴보고, 4장에서는 제안 기법인 로컬 프로세스 실행 기술을 소개한다. 5장 및 6장에서는 제안 시스템의 구현 및 평가 내용을 서술하며, 마지막으로 7장에서는 결론을 맺는다.

II. 배경 지식

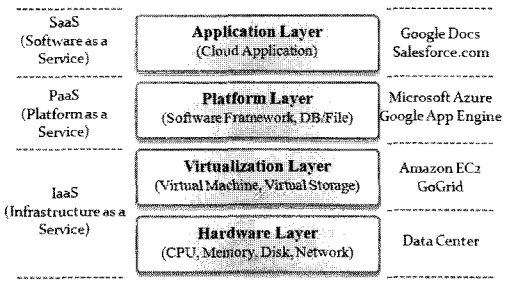
본 장에서는 클라우드 컴퓨팅의 아키텍쳐와 클라우드 환경에서 새롭게 대두되는 보안위협에 대해서 소개한다.

2.1 클라우드 컴퓨팅 아키텍쳐

클라우드 컴퓨팅은 [그림 1]과 같이 하드웨어 계층, 가상화 계층, 플랫폼 계층, 어플리케이션 계층의 4계층 구조를 가진다. 하드웨어 계층은 클라우드의 물리적인 자원 제공하는 계층으로 서버, 라우터 및 방열시스템 등을 포함한다. 가상화 계층은 물리적 자원을 가상화 기술을 이용하여 분할함으로써 논리적으로 독립된 컴퓨팅 및 저장 자원을 제공한다. 이를 가능하게하는 핵심 기술이 가상화 기술이며, Xen, KVM, VMWare와 같은 가상화 기술들이 많이 사용된다[3,4,5]. 플랫폼 계층은 가상화 계층 상위에서 운영체제와 어플리케이션 프레임워크 기능을 제공한다. 특히, 플랫폼 계층은 개별 어플리케이션을 각 가상머신에 배포해줌으로써 관리의 부담을 덜어주며, Google App Engine, Microsoft Azure 등이 대표적이다[6,7]. 어플리케이션 계층은 최상위 계층으로 클라우드 어플리케이션이 존재하는 계층으로써 클라우드 어플리케이션은 보다 나은 성능과 적은 관리 비용을 얻기 위해 클라우드의 확장성을 활용하여 개발될 수 있다.

위와 같은 구조를 기반으로 클라우드 컴퓨팅은 사용자에게 다양한 서비스를 제공한다. 비즈니스 모델 측면에서 클라우드 컴퓨팅의 구조를 분류하면, IaaS(Infrastructure as a Service), PaaS(Platform as a Service), SaaS(Software as a Service)로 나눌 수 있다. IaaS는 하드웨어 자원을 가상화하여 요청에 따라 제공하는 것으로 Amazon의 EC2, GoGrid이 이에 해당된다[8,9]. PaaS는 운영체제, 소프트웨어 개발 및 수행환경을 제공해주는 것으로 Google App Engine, Microsoft Azure등이 해당된다. SaaS는 클라우드환경에서 어플리케이션 서비스를 제공하는 것으로 Google Docs, Salesforce.com등이 있다.

이용대상에 따라 클라우트 컴퓨팅의 유형을 분류하면 크게 공공 클라우드(Public Cloud)와 사설 클라우드(Private Cloud)로 나눌수 있다. 공공 클라우드는 일반에게 서비스를 제공하는 것을 목적으로 하



[그림 1] 클라우드 컴퓨팅 아키텍쳐

며, 서비스 사용자 측면에서 초기투자가 필요 없다는 장점이 있으나, 기업의 기밀 데이터를 처리하기에는 보안성의 문제가 있을 수 있다. 반대로 사설 클라우드는 기업과 같이 한 조직에 서비스를 제공하는 형태로 보안성이 보다 높을 수 있으나, 초기 투자 및 확장성에 있어서 클라우드 컴퓨팅의 장점을 최대한 살리지는 못하는 형태이다. 이러한 두 가지 형태의 장점을 취하고자 하이브리드 클라우드(Hybrid Cloud)를 사용하기도 하는데, 이는 기업의 기밀 데이터 처리와 같은 서비스는 사설 클라우드에서 처리하고 그 외 나머지 부분은 공공 클라우드에서 처리하는 형태이다. 하지만 사설 클라우드에서 처리해야하는 서비스와 공공 클라우드에서 처리해야하는 서비스를 신중히 결정해야하며, 공공 클라우드에서 사설 클라우드에서 처리해야할 데이터가 처리되지 않도록 관리가 필요하다. 이에 대한 대안으로 가상 사설 클라우드(Virtual Private Cloud)는 공공 클라우드에서 VPN(Virtual Private Network)와 같은 보안기술을 사용하여 사설 클라우드와 같이 안전한 클라우드 서비스를 제공하는 것을 목표로 한다. 이와 같이 현재 클라우드 컴퓨팅은 보안 문제와 밀접하게 연관되어있으며 더 많은 보안 기술들을 필요로 하고 있다.

2.2 클라우드 컴퓨팅의 보안 위협

기존의 컴퓨팅 환경을 클라우드 컴퓨팅 환경으로 전환하는데 있어서 반드시 해결해야할 문제 중에 하나는 보안(security) 문제이다. 시장조사 기관인 IDC의 조사에 따르면 IT 클라우드 서비스에 있어서 가장 큰 문제로 여기는 것이 보안 문제인 것으로 조사되었다[10]. 실제로 2008년 2월 아마존에서 제공하는 AWS S3의 서비스 중단 사고, 2008년 9월 Google Docs의 데이터 유출 사고가 있었다.

클라우드 환경의 보안문제에 대한 가이드라인을

UC Berkely와 Gartner등 여러 기관에서 제시하고 있으며, Gartner에서는 클라우드의 보안 위협을 판단하기 위하여 클라우드 관리자 접근 권한에 대한 관리 기준, 데이터 저장 위치 및 사용자별 데이터 분리 보장 기술에 대한 관리 기술 등 7가지 기준을 제시하였다[11,12].

이러한 보안 기준과 더불어 OCC(Open Cloud Consortium), CSA(Cloud Security Alliance) 등에서는 이기종간 클라우드 표준화 및 보안 표준화 작업을 진행하고 있으며, 국내에서도 클라우드 컴퓨팅 산업 포럼 및 클라우드 서비스 협의회 등이 신설되어 클라우드 표준화를 진행하고 있다[13,14,15]. 이와 같이 다양한 보안 기준 및 표준화 작업이 진행되고 있지만 실질적인 기술 연구에 있어서는 아직까지 미흡한 상황으로 이에 관련 연구가 많이 필요하다.

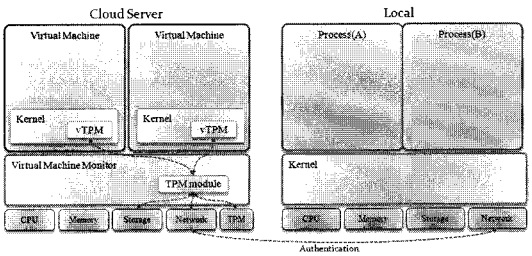
III. 관련 연구

본 장에서는 클라우드 데이터 보안에서 현재 활발히 연구되고 있는 TPM(Trusted Platform Module)을 이용한 보안 기술들을 소개한다. 또한, 서버 관리 효율성 제고 측면에서 활용되는 원격 프로세스 실행 기술을 소개한다. 원격 프로세스 실행 기술은 목적 및 동작 과정이 제안 기법과 명확히 다른 기술로 제안 기법의 정확한 정의를 위해 비교 대상으로 특징들을 서술한다.

3.1 클라우드 데이터 보안

원격지 서버의 저장된 리소스를 아웃소싱하는 클라우드 컴퓨팅에서 데이터의 보안성을 확보하는 것은 현재 가장 시급히 해결되어야 할 문제 중 하나이다. 이를 위해서 데이터의 송/수신 및 저장의 보안성을 높이는 기술들이 연구되고 있으며 주로 암호화 기술 기반의 TPM을 이용한 보안 기술들이 주를 이루고 있다.

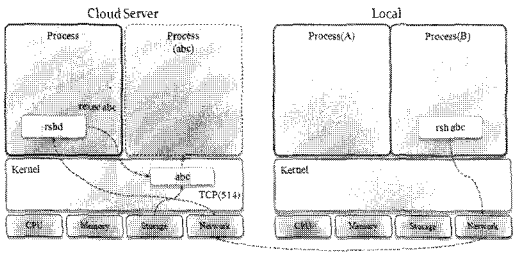
Usenix Hot Cloud'09에서 발표된 [16]에서는 TVMM(Trusted Virtual Machine Monitor)와 TC(Trusted Coordinator)를 통하여 가상 머신 내 보안 및 원격지 가상 머신 인증을 제공하는 신뢰 클라우드 컴퓨팅 플랫폼을 제안하였다. 마찬가지로 Hot Cloud'09에서 발표된 [17]에서는 LoBot을 이용한 가상 머신 보안과 PVI(Private Virtual Infrastructure)를 통한 클라우드 플랫폼 보안을 제공하는 연구를 진행하였다. 이러한 연구들은 TPM을 기반으로 보안성 향상을 제공하고 있다. 다음은 TPM을 이용하는 기존 연구의 예를 보여준다.



[그림 2] TPM을 이용한 기존 연구

TPM을 이용하는 최근 연구들은 위 [그림 2]와 같은 구조를 보인다. TPM은 하드웨어로 구성된 개인키와 암호화/복호화 함수를 통하여 강력한 암호화 기능을 제공한다. 가상 머신 모니터에서는 TPM을 통하여 스토리지 보안, 제3의 공인기관을 통한 사용자 원격 인증 및 각각의 가상 머신에 대한 보안 기술을 제공한다. 결과적으로 클라우드 서버에서 데이터의 비밀 유지 및 원격 인증이 가능하다.

그러나 이러한 암호화를 이용하는 기존의 연구들은 실행 중인 데이터들이 메모리에 상주하여 발생 가능한 보안 위협에 대해서는 대응하지 못하고 있다. 암호화된 기밀 데이터라고 할지라도 프로세스가 실행 중에 해당 기밀 데이터를 읽어 들여 원하는 작업을 수행하기 위해서는 복호화된 기밀 데이터가 메모리의 일정 영역에 저장되는 시점이 발생한다. 이때, 관리자가 악의적인 목적으로 해당 기밀 데이터를 접근하고자 할 경우, 동적 코드 분석 기술 등을 이용하여 얼마든지 해당 정보를 추출할 수 있다. 그리고 암호화된 데이터 자체는 외부에 공개됨에 따라 복호화 및 관리 미흡에 의한 위협 등 암호화 기술 자체의 문제 또한 포함하고 있다. 암호화 기술이 현재는 유효할 수 있지만 기밀 데이터를 영원히 안전하게 보호할 수 있다는 것을 보장하지는 못한다. 수 년, 또는 수 십년 내에 암호화 기술의 취약점이 발견될 경우, 암호화된 기밀 데이터는 복호화 기술에 의해 보안 위협에 그대로 노출되게 된다. 이와 같이 클라우드 서버에 존재하는 보안 데이터들은 악의적인 클라우드 서버 관리자에 의해서 메모리 덤프(Dump) 및 암호화 데이터 유출 등의 문제를 발생 시킬 수 있다. 이와 달리 본 논문에서 제안하는 로컬 프로세스 실행 기술은 실행 중인 컨텍스트(Context)의 정보를 사용자의 로컬 환경에서 생성 및 접근하게 되어 앞서 언급한 클라우드 서버에서 발생 가능한 보안 위협으로부터 근본적으로 보호받을 수 있다.



〔그림 3〕 원격 프로세스 실행의 예

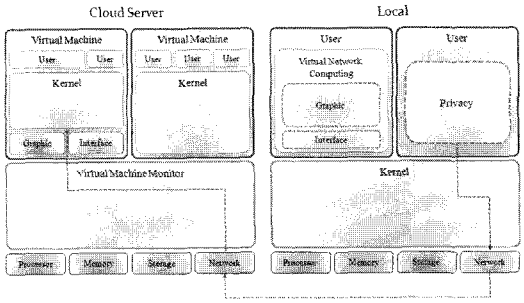
3.2 원격 프로세스 실행 (Remote Process Execution)

원격 프로세스 실행 기술은 원격지의 사용자가 서버의 파일 시스템에 존재하는 실행 가능한 바이너리를 실행하는 기술이다[18]. 원격 프로세스 실행 기술은 1983년 BSD Unix에서 rsh(remote shell)을 통하여 처음 제공되었으며 서버의 514번 TCP 포트와 rshd 데몬을 이용한다[19]. 다음 〔그림 3〕은 원격 프로세스 실행의 예를 보여준다.

위 〔그림 3〕과 같이, 원격지의 사용자는 rsh 클라이언트 프로그램을 통해 abc라는 임의의 프로그램의 실행을 서버에게 요청한다. 서버는 네트워크를 통해 받은 사용자의 요청을 rshd 데몬을 통하여 전달 받아, rexec 명령을 통하여 서버 내의 파일 시스템에 존재하는 abc를 실행한다. 이와 같은 원격 프로세스 실행 기술은 사용자의 요청을 서버에서 처리하는 단방향 커뮤니케이션으로써 서버 관리의 효율성을 목적으로 하며 보안 이슈와는 관련성이 적다. 이와 달리, 본 논문에서 제안하는 로컬 프로세스 실행 기술은 클라우드 컴퓨팅의 데이터 보안성 향상을 목적으로하며 서버 내의 프로세스 자원을 원격지 사용자의 로컬 환경으로 옮겨와 실행하는 기술이다. 자세한 세부 기술에 대한 설명은 다음 장에서 소개한다.

IV. 로컬 프로세스 실행 기술

본 논문에서는 기존 클라우드 컴퓨팅 기술의 보안 취약성을 해결하는 새로운 구조를 제안한다. 기존 클라우드 컴퓨팅 환경에서는 애플리케이션의 실행이 원격 클라우드 서버에서 이루어지며 로컬에서는 애플리케이션 실행을 위한 인터페이스만 제공하였다. 이러한 환경에서는 애플리케이션의 실행을 위해서 보안 데이터 및 코드가 클라우드 서버의 메모리에 존재하게 됨으로써 기존 운영체제 취약성에 의한 위협, 악성코드



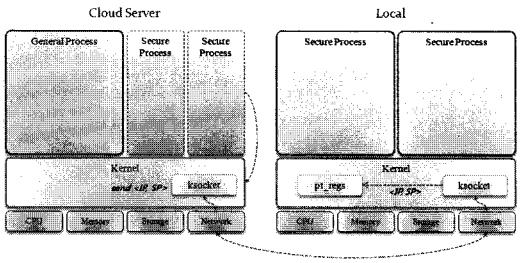
〔그림 4〕 제안 시스템의 기본 구조

접근 및 관리자에 의한 임의적 접근 등 기존 클라우드 환경에서 제기되어온 보안 위협에 그대로 노출되게 된다. 이와 달리 제안 시스템에서는 높은 보안을 필요로 하는 프로세스를 보안이 취약한 서버에서가 아닌 로컬에서 실행할 수 있는 환경을 제공하며, 이를 본 논문에서는 로컬 프로세스 실행 기술이라고 지칭한다. 이에 따라 해당 프로세스에서 필요로 하는 데이터 및 코드를 관리자 서버가 아닌 로컬 자신의 물리 메모리에 로딩하여 연산 처리 할수 있게 됨으로써 클라이언트 환경에서 발생 가능한 보안의 취약점을 근본적으로 해결할 수 있게 된다. 〔그림 4〕는 제안 시스템의 기본 구조를 보여준다.

제안 시스템에서는 클라우드 서버에서 실행될 사용자 프로세스들을 두 가지 유형으로 분류한다. 기존과 동일하게 클라우드 서버에서 실행할 일반 프로세스와 보안 위협에 노출되기 쉬운 보안 프로세스 두 가지로 분류하여 보안 프로세스는 로컬에서 실행하게 된다. 이를 위해 보안 프로세스가 로컬에서 실행되는 과정에서 발생하는 페이지 폴트 및 시스템 콜에 대한 처리가 필요하다. 본 장에서는 제안 시스템에서 보안 프로세스가 동작하는 과정 및 페이지 폴트/시스템 콜 처리 기법에 대하여 각각 살펴본다.

4.1 보안 프로세스 실행 시나리오/과정

로컬 환경에서 프로세스가 실행되는 시점에서부터 보안 프로세스의 실행 방법은 일반 프로세스와 구분된다. 보안 프로세스 실행을 위하여 클라우드 서버에서는 실행파일의 헤더 섹션을 읽어 들여 메모리 맵을 설정하고 힙 메모리를 할당하는 등 초기화 과정을 거친 후, 로컬에서 보안 프로세스가 실행될 수 있도록 프로세스 시작 주소 및 스택 포인터 등 필요한 정보를 네트워크 소켓을 이용하여 전송한다. 로컬에서는 클라우드 서버에서 전송해준 정보를 기반으로 보안 프로세스



〔그림 5〕 보안 프로세스 관점에서 제안 시스템에서의 실행 과정

를 실행시킨다. 로컬에서는 기본 실행을 위한 정보만 주어졌을 뿐, 코드 및 데이터가 아직 메모리에 로딩되어 있지 않은 상태이므로 페이지 폴트가 발생한다. 제안 시스템에서는 보안 프로세스의 프로세스 컨텍스트는 로컬에서 실행하고 커널 컨텍스트는 클라우드 서버에서 실행하도록 설계하였다. 따라서 해당 페이지 폴트에 대한 처리는 클라우드 서버와 로컬 간에 네트워크 통신을 통해 필요한 정보를 주고받음으로써 보안 프로세스가 정상 동작할 수 있도록 처리한다. 이와 마찬가지로 보안 프로세스에서 시스템 콜이 발생하는 경우에는 페이지 폴트 처리와 유사한 방식으로 클라우드 서버와 로컬 간에 필요한 정보를 주고받아서 해당 시스템 콜을 수행한다. 또한, 공인인증서 및 로그인을 위한 개인 키 등 보안 데이터는 로컬의 디스크에 저장하고 해당 데이터를 필요에 따라 읽어 들일 수 있도록 설계하였다. 따라서 로컬에 보안 데이터가 디스크나 메모리에 존재함으로써 발생 가능한 보안위협을 근본적으로 해결할 수 있게 된다. 결국, 이와 같은 과정을 거쳐 보안 프로세스는 로컬의 물리 메모리에 실행파일의 코드 및 데이터를 로딩하여 해당 프로세스의 동작을 수행하게 된다. 〔그림 5〕는 보안 프로세스 관점에서 프로그램의 실행 과정을 간략화 하여 보여준다.

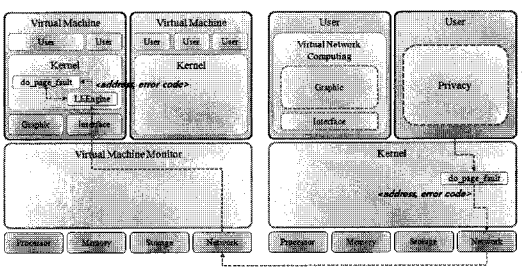
4.2 보안 프로세스 실행을 위한 페이지 폴트 처리 기법

일반적으로 프로세스의 실행 과정에서 메모리 접근을 위한 주요 동작은 크게 두 단계로, 초기화 단계에서 프로그램 헤더 정보를 기반으로 이루어지는 실행 초기화 동작과 페이지 폴트에 의한 물리 페이지 프레임 할당을 위한 동작으로 나누어 볼 수 있다. 보안 프로세스에서는 두 단계의 실행 초기화 동작과 메모리 접근을 위한 페이지 할당을 위한 동작을 각각 클라우드 서버와 클라우드 클라이언트에서 나누어 수행하게 된다. 보안 프로세스는 기본적으로 클라우드 서버에서 프로그램 실행을 위한 초기화가 이루어지며, 이후 프로세스 컨텍스트에 해당 하는 실질적인 프로세스의 실행은 로컬에서 동작하도록 환경을 구축하였다. 즉, 실행 파일 실행을 위하여 헤더 정보 기반으로 메모리 맵 생성 등은 클라우드 서버에서 이루어지며, 초기화된 설정에 따라 실질적인 물리 페이지 프레임 할당은 로컬에서 이루어진다. 메모리 맵은 실행할 코드와 데이터가 위치할 메모리 주소를 명시할 뿐 해당 페이지에 할당할 페이지 프레임은 on-demand로 페이지 폴트 발생 시 로컬에서 할당되며, 클라우드 서버에서 해당 페이지 프레임에 저장할 코드 및 데이터를 네트워크를 통해 전달받는다.

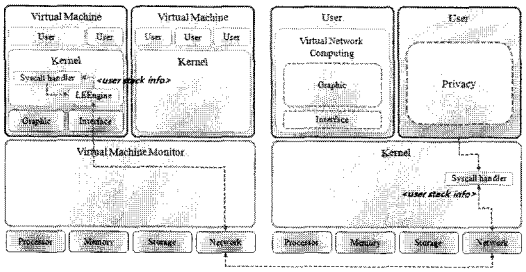
이에 따라 보안 프로세스가 로컬에서 최초로 실행되는 경우에는 초기화 정보인 메모리 맵 정보만 클라우드 서버로부터 전달 받아 구성되며, 페이지 테이블에는 실질적으로 물리 페이지 프레임이 할당되어 있지 않다. 따라서 프로그램의 시작 메모리 주소인 EIP에 해당하는 코드를 실행 요청할 때 페이지 폴트가 발생하게 된다. 프로세스 메모리 관리는 서버에서 이루어지며, 로컬에서는 해당 정보를 전송 받아 실질적으로 물리 자원들을 할당 및 해제하게 된다. 해당 페이지에 대한 정보는 클라우드 서버의 해당 저장 장치에서 읽어 들여 로컬로 전송 받은 후, 로컬의 새로운 페이지 프레임을 할당 받아서 전달받은 정보를 저장하고 페이지 맵핑 과정을 거쳐 프로세스가 실행하게 된다. 이후에도 보안 프로세스의 실행에서 필요한 페이지에 대한 접근은 페이지 폴트 발생에 의한 클라우드 서버와 로컬의 유기적인 동작에 의해 이루어진다. 〔그림 6〕에서는 보안 프로세스 실행 시 페이지 폴트 처리 과정을 보여준다.

4.3 보안 프로세스 실행을 위한 시스템 콜 처리 기법

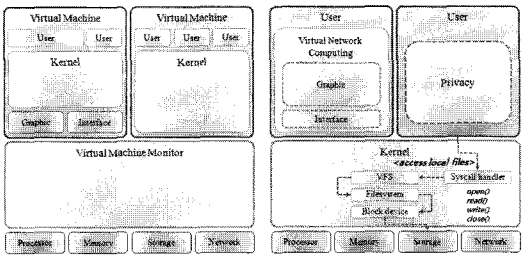
로컬에서 보안 프로세스가 실행되는 과정에서는 디



〔그림 6〕 보안프로세스 실행 시 페이지 폴트 처리 과정



〔그림 7〕 보안 프로세스 실행 시 시스템 콜 처리 과정



〔그림 8〕 보안 프로세스 실행 시 로컬 저장장치에 존재하는 보안 데이터 접근 과정

스크 I/O 요청 등을 위한 시스템 콜이 발생하게 된다. 제안 시스템은 보안 프로세스의 코드 및 데이터가 로컬에서 실질적으로 수행되게 할 뿐 커널에 의해 이루어지는 기능들은 클라우드 서버에서 동작하도록 설계하였으므로 시스템 콜에 대한 처리 역시 클라우드 서버에서 수행하게 된다. 이를 가능하게 하기 위해서 로컬에서 보안 프로세스 실행 과정에서 시스템 콜 발생 시 해당 시스템 콜 처리를 위한 컨텍스트 정보를 클라우드 서버로 전송해주고, 클라우드 서버에서는 시스템 콜 처리 결과를 다시 로컬로 전달해 주는 일종의 시스템 콜 리다이렉션 기능을 제공한다. 〔그림 7〕은 보안 프로세스 실행 시 시스템 콜 처리 과정을 보여준다.

4.4 보안 프로세스 실행을 위한 로컬 장치 접근 기법

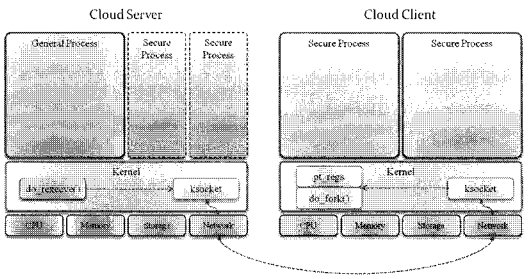
제안 시스템에서는 보안 위협에 노출되었을 때 문제가 되는 보안 데이터는 클라우드 서버가 아닌 로컬에서 저장 및 관리하도록 설계하였다. 따라서 보안 데이터는 클라우드 서버에 연관되지 않고 오로지 로컬에서 처리되어 클라우드 컴퓨팅에서의 보안 위협으로부터 벗어나게 된다. 이를 위해 제안 시스템에서 보안 프로세스가 데이터를 읽고 쓰기 위해 파일에 접근할 경우 일반 데이터 접근과 보안 데이터 접근은 각기 다른 방식으로 수행된다. 프로세스에서 파일에 대한 접근은 기본적으로 시스템 콜에 의해 이루어지며 일반 데이터의 경우 앞서 설명한 시스템 콜 처리 기법에 따라 클라우드 서버의 저장장치에서 파일을 접근하여 필요한 정보를 읽고 쓰게 된다. 하지만 보안 데이터의 경우 시스템 콜 호출 시 클라우드 서버가 아닌 로컬에 있는 저장장치에 접근하여 파일을 읽고 쓰도록 〔그림 8〕과 같이 설계하였다. 즉, 보안 데이터 처리를 위한 시스템 콜은 예외적으로 로컬 커널에서 처리하도록 하여 클라우드 서버로부터의 개입을 원천봉쇄하였다. 또한, 보안 데이터 영역에서 페이지 폴트가 발생할 경우에도 기본 페이지 폴트 처리 방식이 아닌 로컬에서 해당 데이터를 다시 읽어 들일 수 있도록 시스템을 구축하였다. mmap() 등을 이용하여 보안 데이터를 메모리 로딩 과정에서 보안 데이터 영역을 일반 데이터 영역과 구분하여 관리함으로써 해당 페이지에서 페이지 폴트 발생 시 클라우드 서버로 페이지 폴트 처리를 넘기지 않고 로컬에서 수행하도록 한다.

V. 로컬 프로세스 실행 기술 구현

제안 시스템은 linux 2.6.33 커널을 기반으로 do_execve(), do_fork(), do_page_fault() 등 프로세스 실행, 페이지 폴트 처리 및 시스템 콜 처리를 위한 코드들을 재설계함으로써 보안 프로세스가 클라우드 서버와 로컬에서 정상적으로 동작할 수 있도록 구현하였다.

5.1 보안 프로세스 실행을 위한 기본 구현

기본적으로 일반 리눅스에서 프로세스의 실행은 do_execve()의 실행을 통해 이루어진다. 제안 시스템에서는 do_execve()가 프로세스 실행을 위해 수행하는 기능들을 클라우드 서버와 로컬에서 나누어 수행할 수 있도록 하였으며 do_rexecve()의 이름으로 〔그림 9〕와 같이 재설계하였다. 클라우드 서버와 로컬 양 측면에서 각각의 수행 동작을 살펴 보면 다음과 같다. 우선, 클라우드 서버에서는 기존 do_execve()에서 수행하던 실행 전 상태까지 동일하게 수행한 후, 원격 프로세스를 실행시킬 준비를 한다. 로컬에서 보안 프로세스를 실행하기 위해서는 실행할 프로세스의 IP 및 스택 포인터가 필요하므로 소켓을 통하여 해당 정보를 전송한다. 그리고 클라우드 서버에서는 로컬에서 보안 프로세스가 동작하며 중간에 페이지 폴트 및 시스템 콜 요청 등을 처리하기 위하여 대기한다. 반대로 클라우드 로컬 측면에서는 클라우드 서버로부터 보

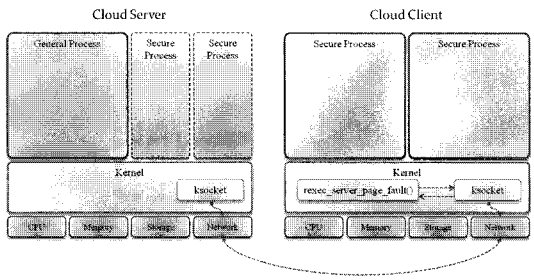


〔그림 9〕 do_execve 함수 처리과정

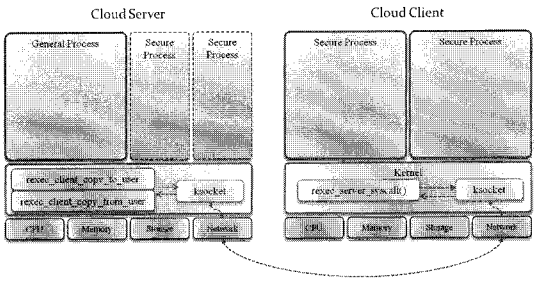
안 프로세스 실행 요청이 있을 때까지 대기하다가 소켓을 통해 보안 프로세스의 실행요청을 받으면 IP 및 스택 포인터 값을 넘겨받는다. 그리고 프로세스 실행을 위해 IP 및 스택 포인터 값을 제외한 나머지 컨텍스트 정보들은 임의 셋팅하여 pt_regs 값을 모두 설정하고 수정된 do_fork()를 실행하여 보안 프로세스를 생성시킨다. 이렇게 생성된 프로세스는 초기 실행에서 페이지 폴트를 통해 필요한 코드와 데이터를 클라이언트 메모리에 로딩한 후 자신의 기능을 수행한다.

5.2 보안 프로세스 실행을 위한 페이지 폴트 처리 기법 구현

보안 프로세스가 클라우드 클라이언트에서 수행되는 과정에서 페이지 폴트가 발생하면 수정된 do_page_fault()가 수행된다. 해당 함수는 〔그림 10〕과 같이 중간에 rexec_server_page_fault() 함수를 호출하여 현재 컨텍스트 정보 및 에러코드 정보를 클라우드 서버로 전송하여 준다. 클라우드 서버에서는 해당 페이지 폴트에 대한 처리 후 결과 값을 로컬로 송신하여 준다. 로컬에서는 rexec_server_insert_page()를 수행하여 페이지 설정하고 보안 프로세스의 실행을 복귀시킨다.



〔그림 10〕 페이지 폴트 처리과정



〔그림 11〕 시스템 콜 처리과정

5.3 보안 프로세스 실행을 위한 시스템 콜 처리 기법 구현

로컬에서는 기존 시스템 콜과 구분하여 보안 프로세스에서 발생하는 시스템 콜에 대한 처리도 가능하도록 구현하였다. system_call 엔트리에서 기본적으로 모든 시스템 콜을 핸들하고, 보안 프로세스가 호출한 시스템 콜들은 구분하여 remote_syscall 엔트리에서 처리하도록 하였다. remote_syscall 에서는 rexec_server_syscall()을 호출하며 현재 컨텍스트 정보인 pt_regs 값을 클라우드 서버로 넘겨준다. 클라우드 서버에서는 로컬에서 실행된 컨텍스트 정보를 기반으로 해당 시스템 콜을 처리한 후 결과 값을 다시 로컬로 전송해 준다. 그리고 클라우드 서버의 시스템 콜 처리 과정에서 보안 프로세스의 유저 영역 데이터를 필요로 하는 경우가 발생할 수 있는데, 이를 위해서는 로컬의 유저영역에 접근을 필요로 한다. 이를 위해서 〔그림 11〕과 같이 클라우드 서버에서는 copy_to_user()/copy_from_user() 의 역할을 해주는 rexec_client_copy_to_user()/rexec_client_copy_from_user()를 구현하여 시스템 콜 수행 과정에서 필요한 유저 영역의 데이터를 접근한다. 그리고 로컬에서는 시스템 콜 처리 결과를 클라우드 서버로부터 넘겨받은 후 보안 프로세스의 실행을 이어간다.

VI. 로컬 프로세스 실행 기술 평가

기존 클라우드 컴퓨팅 환경에서 발생 가능한 위협들에는 운영체제, 네트워크, 프로세스, 하이퍼바이저, 관리자 등에 의한 보안 위협들을 생각해 볼 수 있다. 본 장에서는 기존 클라우드 컴퓨팅 환경에서의 보안 위협과 제안 기법의 보안 위협을 정형화된 수식으로 비교 평가하여 본다. 우선, 클라우딩 환경에서 발생 가능한 보안 위협들은 다음 〔표 1〕과 같이 정의할 수

〔표 1〕 클라우딩 환경에서 발생 가능한 보안 위협 유형

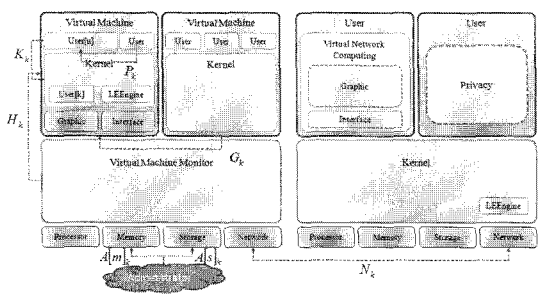
유형	정의	
P_k	인덱스 k 프로세스에 의한 보안 위협	
K_k	인덱스 k 커널에 의한 보안 위협	
H_k	인덱스 k 하이퍼바이저에 의한 보안 위협	
G_k	인덱스 k 가상머신에 의한 보안 위협	
N_k	인덱스 k 네트워크에 의한 보안 위협	
A_k	인덱스 k 관리자에 의한 보안 위협	
	$A\{m\}_k$	인덱스 k 관리자에 의한 메모리 보안 위협
	$A\{s\}_k$	인덱스 k 관리자에 의한 저장장치 보안 위협

있으며, 〔그림 12〕는 각각의 위협들이 클라우드 컴퓨팅 환경에서 발생 가능 위치를 보여준다.

위와 같이 각각의 보안 위협을 정의하였을 때, 클라우드 서버에서 실행 중인 사용자 프로세스의 보안 위협을 U()k로 표현하였다. 이때, 유저 프로세스의 유저 영역에서의 보안 위협을 U{u}k로 표현하고 유저 프로세스의 커널 영역에서의 보안 위협을 U{k}k로 표현한다. 그리고 기존 클라우드 컴퓨팅 환경에서의 보안을 N()s 로, 제안 시스템에서의 보안을 L()s 로 정의하였을 때, 각각의 환경 및 영역에 따른 보안 위협을 다음과 같이 표현할 수 있다.

$$N(U\{u\}_k)_s = \{ P_k \mid K_k \mid H_k \mid G_k \mid N_k \mid A\{m\}_k \mid A\{s\}_k \}$$
$$N(U\{k\}_k)_s = \{ P_k \mid K_k \mid H_k \}$$
$$L(U\{u\}_{k)s} = \{ \varnothing \}$$
$$L(U\{k\}_k)_s = \{ P_k \mid K_k \mid H_k \}$$
$$L(LEEngine_k)_s = \{ P_k \mid K_k \mid H_k \}$$

기존 시스템의 유저 영역의 코드가 실행될 때에는 〔표 1〕에서 정의한 모든 보안 위협으로부터 영향을 받게 되며, 제안 시스템의 유저 영역의 코드가 실행될 때에는 로컬 프로세스 실행 기술에 의해 〔표 1〕에서 정의한 어떠한 보안 위협으로 부터도 영향을 받지 않게 된다. 이 밖에도 기존 시스템의 커널 영역의 코드가 실행될 때에는 프로세스, 커널 및 하이퍼바이저에 의해 공격받을 수 있으며, 제안 시스템의 커널 영역의 코드가 실행될 때에는 기존 시스템과 마찬가지로 클라우드 서버에서 코드가 실행되므로 기존 시스템과 동일한 보안 위협을 받는다. 그리고 로컬 프로세스 실행 엔진인 LEEngine에 해당하는 코드가 실행될 때에도 마찬가지로 클라우드 서버에서 동작함에 따라 기존 시스템의 커널 영역 코드 수행 및 제안 시스템의 커널 영역 코드가 실될 때와 동일한 보안 위협을 받게 된다. 일반적으로 클라우드 컴퓨팅 환경에서 로컬 또한 보안 취약점이 존재할 수 있지만, 앞서 언급한 바와 같이 본 논문에서는 클라우드 서버에서 발생 가능한 보안 위협 및 대응에 초점을 맞추었으며 로컬에서 발생 가능한 문제들은 기존 시스템 보안 분야에서 충분히 연구되었다고 가정하고 로컬은 보안상 안전하다고 정의하였다.



〔그림 12〕 클라우드 컴퓨팅 환경에서 발생 가능한 보안 위협 유형 분류

제안 시스템에서는 보안 프로세스 실행을 위해 원격 서버와 로컬 사이에서 필요한 데이터를 주고 받는 네트워크 통신 과정이 추가됨에 따라 이에 대한 오버헤드가 발생한다. 기존 시스템과 제안 시스템에서 시스템 콜 및 페이지 폴트 처리에 대한 수행 시간을 표 2와 같이 비교해 봄으로써 제안 시스템의 오버헤드를 측정하였다.

위 실험은 Intel i7 2.8Ghz CPU, 4G RAM, Realtek PCI-Express Gigabit Ehternet Controller의 시스템 환경에서 수행되었으며, Linux 기

〔표 2〕 기존 클라우드 시스템과 제안 시스템에서 주요 시스템 콜 함수 및 페이지 폴트 처리에 대한 수행 시간 비교 결과

(단위 시간: ms)

	유형	기존 클라우드 컴퓨팅 시스템	제안 시스템
시스템 콜	open()	4.67×10^{-3}	3.21×10^{-1}
	write()	1.48×10^{-2}	3.13×10^{-1}
	close()	2.71×10^{-3}	1.58×10^{-1}
	dup()	9.93×10^{-4}	1.59×10^{-1}
	time()	6.30×10^{-4}	1.61×10^{-1}
	brk()	7.20×10^{-4}	1.57×10^{-1}
	getpid()	6.66×10^{-4}	1.44×10^{-1}
Page Fault		2.27×10^{-3}	3.79×10^{-2}

반의 kernel 2.6.33 버전에서 각각의 수행시간을 측정하였다. 실험 결과 기본적으로 시스템 콜 처리를 위해서 필요한 시간보다 제안 기법에서 데이터 전송 과정에서 소요되는 시간이 상대적으로 크게 소요됨을 확인할 수 있다. 페이지 폴트의 경우 제안 기법의 수행 시간이 기존 클라우드 컴퓨팅 시스템에서의 수행 시간에 비해 대략 10배 정도 차이가 나고, 시스템 콜의 경우 대략 103배 정도의 시간이 더 소요되었다. 그리고 open()과 write()의 경우에는 내부적으로 유저영역의 데이터 접근을 위해 copy_from_user()가 수행되어야 한다. 하지만 제안 시스템에서는 copy_from_user() 수행하기 위해서 리다이렉션 기능을 이용해야 하기 때문에 추가적인 데이터 전송을 위한 네트워크 통신 시간이 소요된다. 따라서 제안 시스템에서는 두 번의 리다이렉션 기능을 이용하게 됨에 따라, 이에 따른 수행시간이 소요되어 제안 기법의 open()과 write()의 수행 시간이 제안 기법의 다른 시스템 콜 수행 시간에 비해 2배의 시간이 소요됨을 확인할 수 있다.

VII. 결 론

클라우드 컴퓨팅 환경은 사용자의 편의성을 높임과 동시에 그린 IT 환경을 조성하기 위한 기반구조로 전 세계에서 주목받고 있다. 하지만 아직까지는 개인정보 보호 등 여러 가지 심각한 문제들로 제한적인 환경에서만 활용이 되고 있는 추세이다. 이에 본 논문에서는 클라우드 환경에서 보안성 향상을 위한 로컬 프로세스 실행 기술을 설계하였다. 클라우드 환경에서 동작하는 프로세스 가운데 보안 위협에 취약한 프로세스는 클라우드 서버가 아닌 로컬의 컴퓨팅 자원을 활용하여 동작하게 함으로써 클라우드 서버에서 존재할 때 발생가능한 수많은 위협들로부터 벗어날 수 있게되었다. 특히, 개인키 등 중요 보안 데이터를 로컬 저장장치에 둠으로써 기밀 정보 보호에 취약한 기존 클라우드 환경에서의 보안 위협을 근본적으로 해결하였다.

참 고 문 헌

[1] Wikipedia, Cloud computing, http://en.wikipedia.org/wiki/Cloud_computing.

[2] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud Computing : state-of-art and research challenges," Journal of Internet Services and Applications, pp 7-18, May 2010.

[3] XenSource Inc, Xen, www.xensource.com

[4] Kernal Based Virtual Machine, http://www.linux-kvm.org/page/MainPage

[5] VMWare ESX Server, http://www.vmware.com/products/esx

[6] Google App Engine, http://code.google.com/appengine

[7] Windows Azure, http://www.microsoft.com/azure

[8] Amazon Elastic Computing Cloud, http://aws.amazon.com/ec2

[9] Cloud Hosting, CLoud Computing and Hybrid Infrastructure from GoGrid, http://www.gogrid.com

[10] Asia Pacific End-User Cloud Computing Survey, IDC, 2009.

[11] M. Armbrust et al, Above the clouds: a Berkeley view of cloud computing, UC Berkeley Technical Report, 2009.

[12] Gartner, Assessing the Security Risks of Cloud Computing, 2008. 6, http://www.gartner.com/DisplayDocument?id=685308.

[13] Cloud Security Alliance, Security Guidance for Critical Areas of Focus in Cloud Computing, 2009.

[14] Open Cloud Consortium, http://opencloudconsortium.org/

[15] 이강찬, 이승윤, "클라우드 컴퓨팅 표준화 동향 및 전략," 전자통신동향분석, 25(1), pp 90-99, 2010.

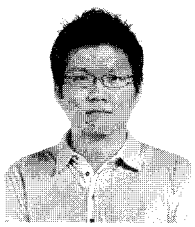
[16] N. Santos, K. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," In Proc of HotCloud09, June 2009.

[17] F. J. Krautheim, "Private virtual infrastructure for cloud computing," In Proc of HotCloud09, June 2009.

[18] Remote Process Execution, http://en.wikipedia.org/wiki/Remote_Process_Execution.

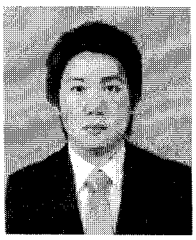
[19] rsh, http://unixhelp.ed.ac.uk/CGI/ man-cgi?rsh.

〈著者紹介〉

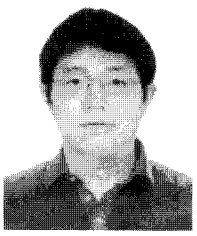


김 태 형 (Taehyoung Kim) 학생회원
2007년 8월: 성균관대학교 컴퓨터공학과 졸업
2009년 2월: 성균관대학교 전자전기컴퓨터공학과 석사
2009년 3월~현재: 성균관대학교 전자전기컴퓨터공학과 박사과정
〈관심분야〉 시스템보안, 가상화, 운영체제

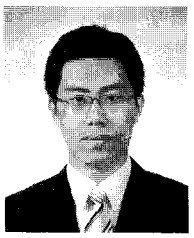
김 인 혁 (Inhyuk Kim) 학생회원
2006년 2월: 성균관대학교 컴퓨터공학과 졸업
2010년 8월: 성균관대학교 전자전기컴퓨터공학과 석사
2010년 9월~현재: 성균관대학교 전자전기컴퓨터공학과 박사과정
〈관심분야〉 시스템보안, 가상화, 운영체제



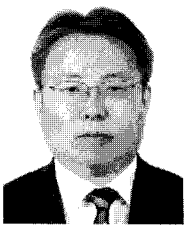
김 정 한 (Junghan Kim) 학생회원
2008년 2월: 세종대학교 컴퓨터소프트웨어공학과 졸업
2010년 2월: 성균관대학교 전자전기컴퓨터공학과 석사
2010년 3월~현재: 성균관대학교 전자전기컴퓨터공학과 박사과정
〈관심분야〉 시스템보안, 가상화, 운영체제



민 창 우 (Chang-woo Min) 학생회원
1996년 2월: 숭실대학교 컴퓨터학부 졸업
1998년 2월: 숭실대학교 전자계산학과 석사
2010년 3월~현재: 성균관대학교 휴대폰학과 박사과정
〈관심분야〉 시스템보안, 가상화, 운영체제



김 지 홍 (Jee-hong Kim) 학생회원
2008년 2월: 광운관대학교 전자공학과 졸업
2010년 2월: 광운대학교 전자공학과 석사
2010년 3월~현재: 성균관대학교 휴대폰학과 박사과정
〈관심분야〉 시스템보안, 가상화, 운영체제



엄 영 익 (Young Ik Eom) 종신회원
1983년 2월: 서울대학교 계산통계학과 졸업
1985년 2월: 서울대학교 전산과학과 석사
1991년 8월: 서울대학교 전산과학과 박사
2000년 9월~2001년 8월: Dept. of Info. and Comm. Science at UCI 방문교수
1993년 3월~현재: 성균관대학교 정보통신공학부 교수
〈관심분야〉 시스템소프트웨어, 미들웨어, 가상화, 시스템보안