논문 13-38B-07-04

한국통신학회논문지 '13-07 Vol.38B No.07
http://dx.doi.org/10.7840/kics.2013.38B.7.526

클라우드 서비스 가상화 내부 환경을 위한 BareMetal Hypervisor 기반 보안 구조 설계

최 도 현•, 유 한 나*, 박 태 성*, 도 경 화**, 전 문 석°

A Design of Security Structure in Bare Metal Hypervisor for Virtualized Internal Enviroment of Cloud Service

Do-Hyeon Choi•, Han-Na You*, Tae-Seung Park*, Kyoung-Hwa Do**, Moon-Seog Jun°

요 약

최근 다양한 형태의 클라우드 컴퓨팅 서비스의 출현으로 인한 가상화 기술이 급부상 하면서 데이터에 대한 안전성과 신뢰성 등 보안 문제가 이슈화되고 있다. 클라우드 서비스의 가상화 계층의 손상은 모든 호스트(사용자) 업무의 손상을 가져올 수 있기 때문에 복수의 가상 운영체제가 구동될 수 있는 환경을 제공하는 하이퍼바이저는 해커들에 의해 공격 대상이 될 수 있다. 본 논문에서는 가상화 기술인 하이퍼바이저(베어 메탈 기반)에 해킹 및 악성코드에 예방을 위한 보안 구조를 제안한다.

Key Words : Hypervisor, Virtualization, Bare Metal, Cloud Service, Cloud Computing

ABSTRACT

With rapid rise of virtualization technology from diverse types of cloud computing service, security problems such as data safety and reliability are the issues at stake. Since damage in virtualization layer of cloud service can cause damage on all host (user) tasks, Hypervisor that provides an environment for multiple virtual operating systems can be a target of attackers. This paper propose a security structure for protecting Hypervisor from hacking and malware infection.

Ⅰ. 서 론

클라우드란 사용자가 필요로 하는 다양한 서비스를 제공하기 위한 모든 것이 서비스로 제공되는 가상화된 공간을 의미한다[1].

IDG Research에서 IT의사 결정자들을 대상으로 실시한 설문 조사결과 응답자의 40%가 보안을 가상화 구현과 관련하여 우려사항으로 언급하였고, Computerworld에 따르면 응답자의 82%가 가상화가 기업의 경쟁 우위 확보에 중요하다고 답했다.[2] 클라우드 서비스는 자원의 공동이용, 인터넷을 이용한 정보서비스, 정보시스템의 아웃소싱, 다양한 디바이스 환경 등을 특징으로 하고 있기 때문에 보안 문제가 클라우드 컴퓨팅 활성화에 장애가 되고 있다.

최근 가상화 내부 공간의 하드웨어 가상화(INTEL VT, AMD-V 등)를 위한 하드웨어 기반의 가상화 기술들이 개발됨에 따라 추가적인 보안위협

◆ 주저자 : 숭실대학교 컴퓨터학과 통신연구실, cdhgod0@ssu.ac.kr, 정회원
° 교신저자 : 숭실대학교 컴퓨터학과 통신연구실, mjun@ssu.ac.kr, 정회원
* 숭실대학교 컴퓨터학과 통신연구실, belover7@naver.com, glittering91@naver.com, 학생회원
** 숭실대학교, 행정안전부, doda0905@gmail.com
논문번호 : KICS2012-07-324, 접수일자 : 2012년 7월 18일, 최종논문접수일자 : 2013년 7월 9일

이 제기 되고 있다. 하드웨어 가상화를 지원함에 따라 성능 효율성의 장점을 가진 베어 메탈 기반 하이퍼바이저(전 가상화 기법)는 앞으로 클라우드 서비스의 가상화 핵심 기술이 될 것으로 예상된다[3,4].

클라우드 서비스 구축 시 가상화 영역의 보안 요구사항은 제공하는 서비스의 범위, 제공방법, 구현방법 등에 따라 다양하게 변경 될 수 있기 때문에 추가적인 보안위협에 대한 대책 마련이 필요하다.

기술적인 보안 요구사항은 외부 영역(클라우드 서비스 사용자의 접근제어), 내부 영역(가상화 영역)으로 분류될 수 있다. 외부 영역은 사용자 식별 및 인증, 인증 정책 등에 대한 기술들로 기존 보안 기술들이 응용되어 많은 보안 솔루션들이 제공되고 있지만, 내부 영역은 가상화된 클라우드 서비스 인프라 구조로 인해 보안 기술 적용에 어려움이 존재한다.

기존 보안솔루션은 OS 내에서 보안기능을 수행하기 위해 커널레벨의 루트권한을 스위칭 한다. 이러한 루트 권한의 스위칭은 하이퍼바이저 동작에 필요한 동등한 레벨의 권한을 요구하는 것으로 가상 OS와 하이퍼바이저 사이의 다른 권한을 가진 보안 구조를 구현해야 하며, 현재 상용화된 하이퍼바이저 제품들은 각 다른 보안레벨의 구조를 정의하고 있기 때문에 보안 기능을 구현하는데 어려움이 있다. 본 논문에서는 이런 문제를 해결하기 위해 클라우드 서비스 기술의 내부 영역인 하이퍼바이저를 위한 보안 구조를 제안한다.

2장은 가상화 기술 내의 하이퍼바이저 기술의 보안 취약성과 국내 클라우드 표준 프레임 워크의 문제점에 대해 분석하고, 3장은 베어 메탈 기반의 하이퍼바이저 보안 구조를 제안한다. 4장은 성능분석, 5장은 결론으로 마친다.

Ⅱ. 관련연구

2.1. 클라우드 서비스 가상화 기술 취약점 분석

NIST(National Institute of Standards and Technology), CSA(Cloud Security Alliance)는 클라우드 컴퓨팅의 위협요소로 공격자들이 클라우드 자원을 비도덕적으로 사용하고 자원을 남용하는 것이 가능하다고 분석했다[5,6]. 내부 영역인 가상화 환경에는 서비스 거부와 같은 해킹 공격과 바이러스 및 악성코드에 대한 위협 등 기존 환경의 취약성이 적용될 수 있다[7,8].

일반적으로 가상화 내부의 취약점에는 첫째 하이퍼바이저 해킹으로 인한 통제권 상실, 둘째 가상화 취약점 상속에 대한 호스트 OS의 감염으로 인한 게스트 OS 간 악성코드 감염, 하이퍼바이저 감염으로 인한 게스트 OS로 확산 등이 존재한다[9].

첫 번째 그림 1와 같이 클라우드 서비스가 자원을 통합, 재분배 하여 공유하는 가상화 영역의 특징으로 인해 서버에 저장되어 있는 모든 데이터의 유출 및 손실과 시스템의 취약점을 상속할 수 있는 문제가 존재한다.

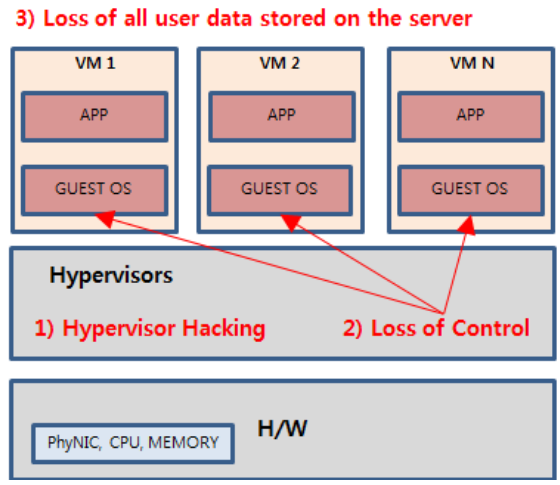

그림 1. 하이퍼바이저의 해킹 위협
Fig. 1. Hypervisor Hacking Threats

두 번째 그림 2와 같이 호스트 기반 하이퍼바이저의 경우 악성코드나 바이러스가 가상 Kernel을 경유하여 통신할 경우 모든 호스트 OS에 감염이 전파될 수 있는 취약점이 존재한다.

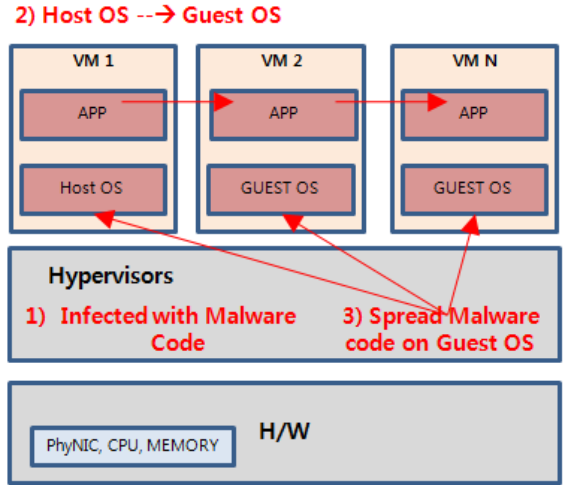

그림 2. 하이퍼바이저의 악성코드 위협
Fig. 2. Hypervisor Malicious Code Threats

가상화된 OS위에 백신을 설치할 경우 Kernel 단

에서 움직이는 악성코드나 바이러스 등에 무방비한 상태가 된다. 이는 기존 오리지널 시스템의 백신 어플리케이션의 시스템 콜 방식과 가상화 시스템의 시스템 콜 방식의 차이로 인해 가상 머신 내부 데이터에 대해서 접근하는 방법이 다르기 때문이다. 결론적으로 가상 OS를 컨트롤 하고 실제 데이터 통신 경로를 제어하는 특권을 가진 하이퍼바이저 레벨에서 보안기술이 적용되어야 한다.

2.2. 베어메탈 기반 하이퍼바이저의 문제점 분석

호스트 기반 하이퍼바이저 방식은 유저모드와 커널모드 사이에서 데이터 흐름과 특권을 제어하는 VMM(Virtual Machine Monitor)기술이 사용된다. 그림 3, 그림 4는 호스트 기반 하이퍼바이저와 베어 메탈 기반 하이퍼바이저를 나타낸다.

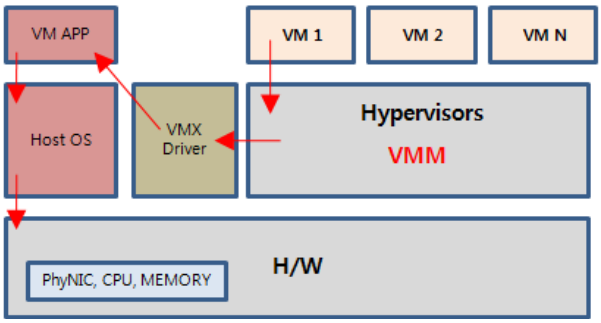

그림 3. 호스트 기반 하이퍼바이저
Fig. 3. Host based Hypervisor

OS 위에 VMM이 설치되는 방식으로 가상화 공간의 모든 I/O요청이 호스트 OS를 통해 전달되기 때문에 악성코드나 바이러스에 대해 침입, 탐지 할 수 있는 영역으로 사용할 수 있다. 실제로 이 VMM에서 각 가상 OS의 통합관제를 가능하게 지원하며 보안 관리의 영역으로 사용할 수 있다.

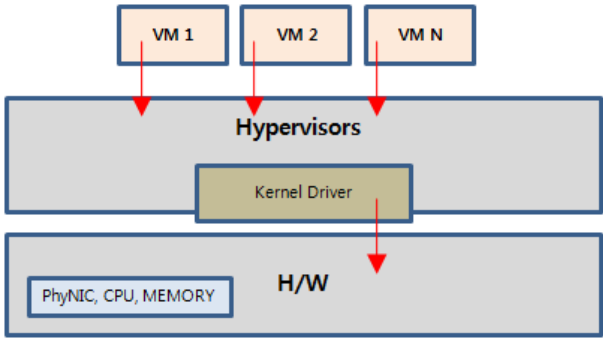

그림 4. 베어메탈 기반 하이퍼바이저
Fig. 4. Baremetal Based Hypervisor

베어 메탈 방식은 하이퍼바이저에 커널 드라이버와 모든 I/O 요청을 직접 통신하는 방식으로 호스트 기반의 하이퍼바이저와 같이 데이터를 제어하는 영역은 없다. 하드웨어와의 직접 통신으로 인한 효율성에 따른 다양한 장점이 있지만 보안적인 측면에서는 취약할 수 있기 때문에 호스트 기반 하이퍼바이저에 제공하는 VMM과 같은 모니터링 기술이 요구된다.

2.3. 국내 표준 침입탐지 프레임워크 문제점 분석

현재 국내 클라우드 표준으로 TTA(한국정보기술협회)에서 정의하는 침입탐지 프레임 워크[10]는 그림 5와 같다.

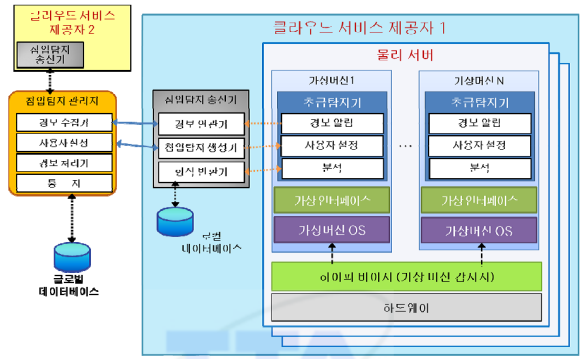

그림 5. 침입 탐지 프레임워크
Fig. 5. Intrusion Detection Framework

각 가상 OS에 기존 보안 솔루션 방식과 같이 보안기능(경보, 분석 등)을 하는 초급탐지기를 두어 관리자에게 통보하는 구조로 정의하고 있다. 이러한 구조는 단순히 각 가상 OS 내부에 대한 보안기능을 지원하는 형태로 가상 OS 이외에 하이퍼바이저 레벨을 보호할 수 없는 구조이다. 또한 각 가상머신 상에 존재하는 초급탐지기는 루트권한 스위칭이 필요하기 때문에 가상 OS간에 동기화 또는 통신 시에 하이퍼바이저에 악성코드나 바이러스를 감염 시킬 수 있는 문제가 존재한다.

이후 감염된 하이퍼바이저는 각 가상 OS들에 위협을 전파할 수 있기 때문에 가상머신 내부에 초급탐지기를 두어 보안기능을 수행하는 것(기존 솔루션 기법)은 다양한 보안 취약성이 존재한다.

현재 국내 클라우드 표준은 클라우드 환경에서 하이퍼바이저를 위한 보안구조에 대한 상세한 정의가 없는 상태이기 때문에 세부 보안구조에 대한 표준의 정의가 요구되는 것으로 분석된다.

Ⅲ. VMMA(Virtual Machine Monitor Agent)

본 논문에서 제안하는 악성코드 및 바이러스 탐지 기술을 VMMA(Virtual Machine Monitor Agent)라고 정의한다. 기존의 악성코드 및 바이러스를 탐지, 차단하는 방식은 대표적으로 시그니처 기반 분석과 동적 기반 분석 방식이 있다. 기존의 VMM와 같은 레벨에서 모니터링 할 수 있는 VMMA 구조는 기존 기법이 적용하기 어려운 문제와 호환성 문제를 해결한다. 그림 6은 VMMA의 구조를 나타낸다.

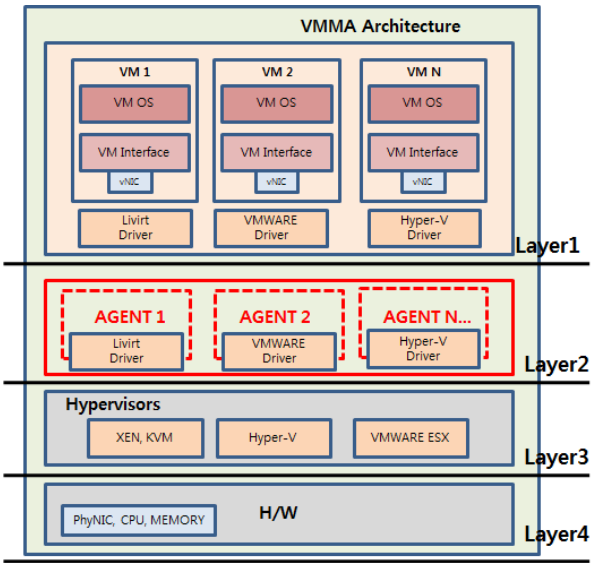

그림 6. VMMA의 구조
Fig. 6. Structure of VMMA

Agent(Layer2)와 하이퍼바이저(Layer3)는 동일한 특권권한을 가지며, 모든 VM에서 발생하는 데이터 트래픽을 Agent(Layer2)에서 모니터링 한다. 계층적으로는 Agent 위에 VM(Layer1)이 동작하는 구조로, 실제로는 하이퍼바이저(Layer3) 위에 설치되어 동작한다. 또한 기존의 하드웨어 커널 드라이버를 Agent 내에 포함하여 VMM 처럼 호스트 기반 하이퍼바이저 방식의 성능 저하 단점을 최소화 한다.

VMMA 구조는 단일 커널 드라이버가 아닌 다중 커널 드라이버를 Agent가 지원하는 구조로 클라우드 서비스 제공자는 운영 중인 다수의 가상 OS를 통합 관리함으로써 다양한 하이퍼바이저에 대한 모니터링 및 통합 관제가 가능하다. 실제 구현 시 데이터 통신 구간의 TCP/IP 패킷을 모니터링 하는 기능 역할만 수행하여 하이퍼바이저 수행에 발생할 수 있는 예외상황을 최소화 하여 안전성을 증가시켰다. 그림 7은 기존 바이러스 및 악성코드 차단 기법이 적용된 VMMA의 데이터의 흐름을 나타낸다.

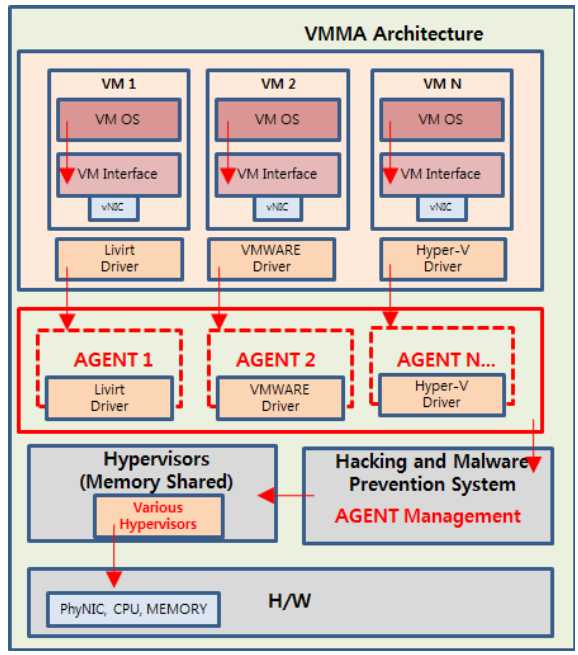

그림 7. 보호된 VMMA의 데이터 흐름
Fig. 7. Protected Data flow of VMMA

보호된 VMMA 구조는 가상 OS를 운영하고 있는 각 클라우드 서비스 제공자가 독립적/공유된 데이터에 대한 보호를 가능하게 한다. 각 분산되어 있는 Agent는 하이퍼바이저의 데이터 흐름을 수집하여 차단 시스템에 보고한다.

가상화 환경의 어플리케이션은 모두 TCP/IP Stack을 통과한다. 각 어플리케이션에서 발생한 트래픽은 Agent에 연동된 해킹 및 악성코드 차단 시스템을 통해서 하이퍼바이저에 접근한다. 차단 시스템은 하이퍼바이저와 독립적으로 운영되기 때문에 구현 시에 특정 도메인에 대한 특권 권한을 따로 추가 구현해야 할 필요가 없다. 기존 Kernel Driver를 통해서 통신하는 악성코드 및 바이러스 등에 감염된 데이터를 VM Agent에서 관리함으로써 하이퍼바이저 감염에 대한 취약성을 해결한다.

그림 8은 Hacking and Malware Detection System이 적용된 Agent 구조를 나타낸다. 각 Agent에서 모니터링 되어 수집되어 저장된 로그는 실제 물리적 호스트 영역에 위치한 로그 데이터베이스에 저장되어 사용될 수 있다.

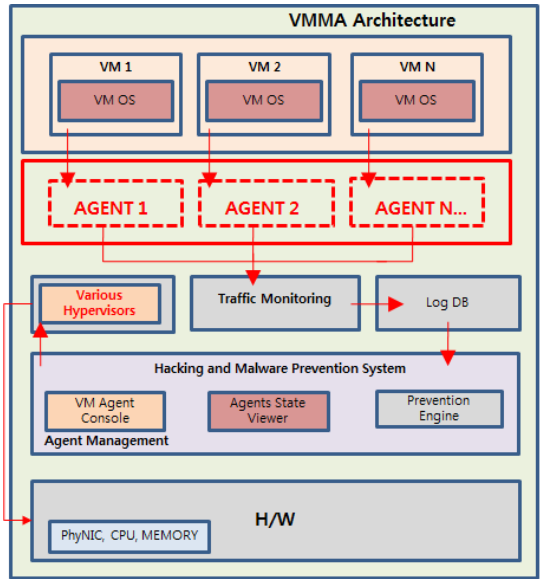

그림 8. 탐지시스템이 적용된 VMMA의 구조
Fig. 8. Applied Detection System of VMMA Structure

차단 시스템에는 베어 메탈 하이퍼바이저 특성상 자체적으로 관리기능이 없기 때문에 VMM 콘솔 같은 별도의 관리콘솔의 구현이 필요하며 차단 시스템에 연동된 탐지 엔진은 시그니처 기반 분석, 동적 기반 분석 등 기존의 악성코드와 해킹에 대한 차단 기술이 적용 될 수 있다.

Ⅳ. 성능평가

본 논문의 시스템 구조에 대한 성능분석은 첫째 VMMA의 추가로 인한 오버헤드 측정을 위하여 하이퍼바이저의 커널 호출에 대한 성능 분석을 비교한다. 커널호출 분석은 차단 시스템이 중간 계층에 추가됨으로써 서비스 가용성에 미치는 영향을 분석하기 위해서이다. 둘째 커널 호출 테스트에서 해킹 및 악성코드로 이용될 수 있는 시스템 콜 및 사용자 라이브러리 함수에 대한 접근 차단 유무를 모니터링 및 분석 하였다. 표 1은 테스트 환경 파라미터를 나타낸다.

표 1. 테스트 환경의 파라미터
Table 1. Parameter of Test Environment

Parameter	Description
Cloud OS : Ubuntu 11.10 64bit Server	Openstack Multi Server (2 Servers)
VM OS(Total 4 VM Clients)	Windows7, Ubuntu10.04 (2 Cpu, 1024 Memory, 10 GB Hdd)
Hypervisors	Windows(Hyper-V), Ubuntu(KVM)
Evaluation List	System Call frequency
	System bottleneck (Cpu, Network, Memory)
Time	5 Minute
Traffic Generator	iperf(Server Mode, TCP, 80Port)

Openstack 기반의 클라우드 환경을 2 서버로 구축하고 VM OS로 윈도우와 리눅스를 생성하였다. 시스템의 성능 분석을 위해 각 VM OS와 Agent 사이에 iperf를 이용하여 트래픽을 생성(default 8kb)하였으며 서버 2대(각 서버 VM OS 4세션)에 80 포트를 이용하여 측정하였다. (최대 대역폭 10Mb)

4.1. 시스템 성능분석

평가 항목은 시스템 콜 호출 빈도와 클라우드 운영 환경의 성능 분석이다. 표 2, 표 3은 각 VM OS의 시스템 콜 성능 분석 결과를 나타낸다.

표 2. Windows7 시스템 콜 성능 분석 결과
Table 2. Analysis Result of Windows 7 System Call

Parameters	Original	Modified
Write	6985	6951
Read	186	188
Open	210	209
Close	207	201
Execve	2	2
Access	39	40
Brk	6	6
Munmap	116	105
Mprotect	48	47
Mmap2	206	198
Stat64	114	100
Fstate64	153	127
Set-thread_area	2	2
Total	8274	8176

표 3. 우분투 10.04 시스템 콜 성능 분석 결과
Table 3. Analysis Result of Ubuntu 10.04 System Call

Parameters	Original	Modified
Write	6897	6921
Read	183	188
Open	201	198
Close	199	191
Execve	2	2

Access	32	34
Brk	5	5
Munmap	102	105
Mprotect	41	40
Mmap2	195	189
State64	102	109
Fstate64	136	141
Set-thread_area	2	2
Total	8097	8125

두 OS의 시스템 콜 횟수의 차이는 트래픽 제너레이터의 불규칙한 트래픽 생성으로 인한 차이로, 이는 VMMA 구조 자체가 시스템 콜에 영향을 줄 필요가 없이 설계되었기 때문에 제안한 VMMA는 시스템 콜에 영향을 끼치지 않는 것으로 분석되었다. 그림 9, 그림 10은 클라우드 운영 환경의 시스템 성능 분석 결과를 나타낸다. 본 성능분석의 대상인 리눅스는 Agent가 설치되어 VM OS(리눅스, 윈도우)를 운영하는 서버를 의미한다.

그림 9. 기존 클라우드 서비스 운영환경(리눅스)
Fig. 9. Existing Cloud Service Operating Enviroment

각 VM OS는 약 5분 동안 80포트에서 발생하는 네트워크 트래픽을 수집하고 시스템 콜 횟수를 분석한 결과이다. Windows7의 경우 수정된 하이퍼바이저가 호출 회수 98개가 적게 나타났고, Ubuntu의 경우 28개가 증가하였다. 각 VM OS에서 발생하는 시스템 콜 횟수는 세션마다 증가하거나 감소하는 것으로 나타났지만 해당 시스템 호출 횟수의 변화는 전체 시스템 콜 횟수의 극히 일부분으로 커널 상에 동작하는 일부 어플리케이션의 영향으로 분석되었다.

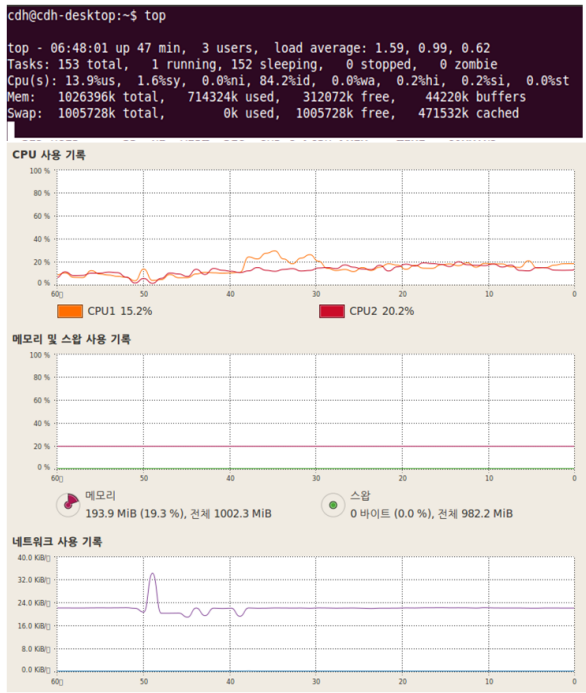

그림 10. 탐지시스템이 적용된 VMMA의 구조
Fig. 10. Applied Detection System of VMMA Structure

본 성능 분석에서는 CPU 과부하와 I/O 성능의 정확한 분석을 위해서 Rwhod, routed 등 불필요한 데몬 프로세스를 모두 제거 하고, CPU의 의존적인 프로세스의 우선순위를 낮추기 위해서 NICE를 사용하였다.

수정된 운영환경의 성능 분석 결과 평균 20% 이하의 Cpu 점유율, Load Average 1분, 5분이 평균 부하가 모두 2개 이하(과부하 : 약 5~10개 이상)의 프로세스 대기상태로 Cpu 부하가 큰 차이가 없는 것으로 나타났으며, 메모리 사용률 평균 20%로 차단을 위한 트래픽 모니터링 기능만을 수행하기 때문에 전체적인 성능에 영향을 끼치지 않는 것으로 분석된다. 네트워크 사용 기록의 경우 80포트를 사용하고 있는 특정 프로토콜(일부 서비스)에 대한 트래픽에 대한 영향으로 약간의 변화이외에 iperf에서 발생한 트래픽으로 일정한 분포를 보이는 것으로 확인했다.

4.2. 제안하는 보안 구조의 안전성 분석

평가 항목은 핵킹 및 악성코드 접근으로 호출될 수 있는 시스템 콜 함수와 라이브러리 함수들로 안전성 분석을 위해 11개의 프로그램과 악성 함수 호

출 코드 총 12개를 대상으로 안전성 분석을 진행하였다. 취약성 함수는 Agent 내에 화이트리스트로 추가하는 방식을 이용하여 호출되는 함수상태를 체크하였다. 표 4, 표 5, 표 6, 표 7은 가상 OS 별 사용된 함수의 리스트를 나타낸다.

표 4. Windows7 함수 목록(기능 별)
Table 4. Windows7 Function List(by Function)

File (Malisicous Code 1)
GetModuleFileName()
CreateFile(), OpenFile(), CopyFile()
MoveFile(), WriteFile()
Registry (Malisicous Code 2)
RegOpenKey(), RegCreateKey()
RegSetValue(), RegQueryValue()
Network (Malisicous Code 3)
Inet_addr()
htons
gethostbyname
Bind, Connect

Win32 API 함수 리스트에서 해킹 및 악성코드로 이용될 수 있는 대상은 크게 두 가지로 분류된다. 첫 번째 분류는 파일 핸들링 및 네트워크, 두 번째 분류는 행위별 함수로 분류하여 테스트를 진행하였다.

표 5. Windows7 함수 목록(행위 별)
Table 5. Windows7 Function List(by action)

Trojan (Malisicous Code 4)
CreateProcess, OpenProcess, TerminateProcess
TerminateProcess
WinExec, ShellExcute
LoadLibrary
SearchPath
Impersonation (Malisicous Code 5)
RpcImpersonateClient
ImpersonateLoggedOnUser
CoImpersonateClient
ImpersonateDdeClientWindow
ImpersonateSecurityContext
ImpersonateAnonymousToken
ImpersonateSelf
Denial Service (Malisicous Code 6)
InitializeCriticalSection
EnterCriticalSection
CreateService, DeleteService
_Alloca
TerminateThread, TerminateProcess
Buffer Overflow (Malisicous Code 7)
lstrcatA, lstrcpyA
Wcscat, Wcsncat
MultiByteToWideChar, strncpy

표 6. 리눅스 시스템 콜 함수 목록
Table 6. Linux System call Function List

File (Malisicous Code 8)
read(), create(), write(), open(), close()
link(), unlink(), execv(), chdir()
mknod(), chmod(), chown()
fstat(), access()
rename(), mkdir(), rmdir(), readdir()
Memory and Network (Malisicous Code 9)
mlock(), munlock(), mremap()
munmap(), syslog()
pipe(), socketcall()
etc (Malisicous Code 10)
reboot(), sysinfo(), setlimit(), kill()

표 7. 리눅스 라이브러리 함수 목록
Table 7. Linux Library Function List

File (Malisicous Code 11)
fopen(), fprintf(), fscanf(), fclose()
Memory (Malisicous Code 12)
memcpy(), memmove(), memcmp(), memset()
malloc(), realloc(), free()

각 표의 함수리스트는 호스트에서 실행될 수 있는 해킹이나 악성코드로 인한 악성행위를 위해, 파일 및 메모리 관련 데이터 변조나 레지스트리 정보 등록, 다른 네트워크와의 안전하지 않는 채널을 생성하는 등 시스템의 자원에 접근하기 위해 호출할 수 있는 함수들로 다양한 시그니처 추출을 통해 누적된 데이터는 클라우드 서비스 환경 내에 악성코드 예방을 위한 패턴 데이터베이스로 사용할 수 있다.

표 8은 차단된 악성행위 함수에 대한 리스트를 나타낸다. A : 전체 함수 콜 횟수, B : 취약성 함수 콜 횟수

표 8. 취약성 함수 차단율
Table 8. Vulnerability Function Blocking rate

Program	A	B
Explorer	127	35
Gomplayer	504	15
Nateon	1483	84
Chrome	331	11
HWP	12663	27
Word	225	4

PowerPoint	223	14
Access	9060	78
V3LITE	3378	73
ALZIP	62	11
DROPBOX	1602	36

취약성 코드는 Agent 내에 악성 함수 리스트에서 필터링 하는 것으로 진행되었으며 차단 결과 악성 함수에 대한 필터링을 정상적으로 수행하는 것을 확인했다. 본 성능분석에서는 악성함수에 대한 탐지 및 감지가 아닌 정상적으로 차단하는지 확인하는데 목적이 있다. 탐지 및 감지의 경우 기존의 악성행위에 대한 분석 기법의 적용에 따라 결과가 다양하게 나타날 것으로 예상된다.

V. 결 론

본 논문에서는 클라우드 서비스 가상화 기술 중에 기존 베어 메탈 기반 하이퍼바이저 방식의 취약점을 개선하기 위해 VMMA기반 베어 메탈 하이퍼바이저를 제안하였다.

Agent 안에 커널 드라이버를 포함하여 해킹 및 악성코드를 차단을 위한 구조를 설계하였고, 성능의 효율성과 다양한 하이퍼바이저 기술의 호환성을 지원할 수 있는 장점을 제공하였다. 성능 분석 결과 시스템 콜 빈도, 운영환경의 성능 모두 큰 영향을 끼치지 않음을 확인하였으며 다양한 취약성 함수의 안전성 분석 결과 차단 기능을 정상적으로 수행하는 것을 확인하였다.

향후 본 논문의 제안한 VMMA의 Agent 구조는 기존의 시그니처 및 동적 엔진 분석기법 등과 연계하여 감지 및 탐지할 수 있는 기술의 연구가 필요하며, 국내 클라우드 보안프레임 워크의 하이퍼바이저 보안구조에 대한 상세한 정의가 필요한 것으로 확인하였다.

Reference

[1] I. Y. Jung, I. Jo, and Y. Yu, “Trust assurance of data in cloud computing environment,” *J. KICS*, vol. 36, no. 9, pp. 1066-1072, Sep. 2011.

[2] HP(Analyst report), *IDG Tech Dossier: Security in the New V-Era(2012)*, Retrieved June, 27, 2012, from http://www8.hp.com/h20195/v2/GetDocument.aspx?docname=4AA4-6716EEW.

[3] Intel, *Intel Virtualization Technology(2006)*, Retrieved June, 30, 2012, from http://www.intel.com/technology/itj/2006/v10i3/1-hardware/5-architecture.htm.

[4] Advanced Micro Devices, *AMD-VTM NestedPaging(2008)*, Retrieved June, 25, 2012, from http://developer.amd.com/wordpress/media/2012/10/NPT-WP-1%201-final-TM.pdf.

[5] T. Grance and W. Jansen, “Guidelines on security and privacy in public cloud computing,” *NIST Special Publication 800-144*, Dec. 2011.

[6] CSA(Cloud Security Alliance), “CSA Guidance Version 3,” *Security Guidance for Critical Areas of Focus in Cloud Computing*, Nov. 2011.

[7] S. Subashini and V. Kavitha, “A survey on security issues in service delivery models of cloud computing,” *J. Network Comput. Applicat.*, vol. 34, no. 1, pp. 1-11, Jan. 2010.

[8] N. Gruschka and M. Jason, “Attack surfaces : a taxonomy for attacks on cloud services,” in *Proc. 2010 IEEE 3rd Int. Conf. Cloud Comput. (CLOUD)*, pp. 276-279, Miami, U.S.A., July 2010.

[9] KISA(Korea Internet Security Agency) Research and Developer Team, “Cloud Service Information Security Guideline,” *KISA Guideline and Explanation*, Oct. 2011.

[10] TTA(Telecommunication Technology Association), *Intrusion Detection Framework in Collaborative Cloud Environment*, TTAK.KO-10.0534, Dec. 2011.

최 도 현 (Do-Hyeon Choi)

2008년 2월 동서울대학 컴퓨터소프트웨어 공학사
2010년 8월 숭실대학교 컴퓨터학과 석사
2010년 9월~현재 숭실대학교 컴퓨터학과 박사과정
<관심분야> Mobile Security, Virtualization, 802.16x, PKI, Secure Coding

유 한 나 (Han-Na You)

2008년 8월 평생교육원 정보보안전공 공학사
2010년 8월 숭실대학교 컴퓨터학과 공학석사
2010년 9월~현재 숭실대학교 컴퓨터학과 박사과정
<관심분야> 금융보안, 인증, 네트워크보안

박 태 성 (Tae-Sung Park)

2011년 8월 숭실대학교 일반대학원 컴퓨터학과 석사
2011년 8월~현재 숭실대학교 일반대학원 컴퓨터학과 박사과정
<관심분야> 정보보호, 네트워크 보안, PKI

도 경 화 (Kyoung-Hwa Do)

2004년 2월 숭실대학교 공학박사
2004년 4월~현재 안전행전부 전문위원
2004년~2010년 한국정보보호학회 논문심사위원
2004년~2007년 산자부 정보보안기술(JTC1/SC27) 전문위원회 운영위원 등
<관심분야> 전자보호 및 개인정보보호 정책 시행, 빅데이터, 전자정부, 정보공동이용 및 활용 기술 등

전 문 석 (Moon-Seog Jun)

1981년 2월 숭실대학교 전자계산학과 졸업
1986년 2월 University of Maryland Computer Science 석사
1989년 2월 University of Maryland Computer Science 박사
1991년 3월~현재 숭실대학교 정교수
<관심분야> 정보보호, 네트워크 보안, 전자여권, 암호학