ARM 기반 모바일 단말의 보안성 향상을 위한 경량 하이퍼바이저 설계 및 구현

Design and Implementation of A Thin Hypervisor for Security Enhancement in ARM-based Mobile Devices

저자 (Authors)
양희철, 전민지, 오영섭, 신재복, 박찬익
Heecheol Yang , Minji Jeon, Youngsup Oh, Jae-bok Shin, Chan-ik Park

출처 (Source)
한국정보과학회 학술발표논문집 , 2015.06, 1589-1591 (3 pages)

발행처 (Publisher)
한국정보과학회
KOREA INFORMATION SCIENCE SOCIETY

URL
http://www.dbpia.co.kr/Article/NODE06394485

APA Style
양희철, 전민지, 오영섭, 신재복, 박찬익 (2015). ARM 기반 모바일 단말의 보안성 향상을 위한 경량 하이퍼바이저 설계 및 구현. 한국정보과학회 학술발표논문집, 1589-1591.

ARM 기반 모바일 단말의 보안성 향상을 위한 경량 하이퍼바이저 설계 및 구현

양희철 ○ 전민지 오영섭 신재복 박찬익

포항공과대학교 컴퓨터공학과

{hcyang1012 ○, mjjeon, youngsup, zstormx, cipark}@postech.ac.kr

Design and Implementation of A Thin Hypervisor for Security Enhancement in ARM-based Mobile Devices

Heecheol Yang○ Minji Jeon Youngsup Oh Jae-bok Shin Chan-ik Park

Department of Computer Science and Technology,

Pohang University of Science and Technology

요 약

스마트폰, 태블릿 등으로 대표되는 모바일 단말이 보편화 되면서 사용자들의 많은 중요 데이터들이 단말 내 저장되고 있고, 이에 따른 보안 위협 또한 커지고 있다. 한편, 시스템 가상화는 이러한 보안 위협으로 인한 영향을 분리/감시 할 수 있는 효과적인 방법이지만, ARM 기반의 모바일 단말의 시스템 가상화를 지원하는 가상 머신 모니터 (예. Xen on ARM, KVM-ARM 등) 는 그 크기 및 성능 오버헤드로 인해 보안성 향상을 목적으로 실제 적용하기에는 제약이 크다. 따라서 본 논문에서는 모바일 단말의 보안성 향상을 위한 하이퍼바이서의 요구조건을 분석하여 설계하고, 실제 모바일 단말에 구현하였다. 구현된 하이퍼바이저의 크기는 48K LOC 이며, 2nd-level 페이지 테이블의 접근제어를 통한 이벤트 검출 오버헤드는 1.6 usec 로 평가되었다. 해당 구현과정에서 모바일 단말 내 보안 위협을 감시 할 수 있는 구체적인 사례 방법을 제시하고, 향후에는 구현된 경량 하이퍼바이저를 기반으로 상위 게스트 VM 의 보안 위협 이벤트를 어떻게 관리할 것인가를 지속적으로 연구할 예정이다.

1. 서 론

스마트폰, 태블릿 등의 모바일 단말들이 보편화 되면서 사용자들은 자신의 비밀번호, 공인 인증서 등 각종 개인 정보뿐만 아니라 업무용 메일, 문서, 사진 등 유출 시 기업의 심각한 손실로 이어 질 수 있는 데이터까지 개인 단말에 저장하게 되었고, 이에 악성 어플리케이션에 의한 해킹이나 단말 분실 등으로 인한 데이터 유출 문제가 모바일 환경에서의 새로운 보안 문제로 제기되고 있다[1][2].

한편, 시스템 가상화 (System virtualization) 기술은 하나 또는 다수의 가상 머신 (Virtual machine)을 하나의 물리적 장치 내에서 구동시켜 가상 머신 모니터 (Virtual machine monitor, Hypervisor)와 독립적인 공간에서 사용자 환경을 제공해 줄 수 있다는 장점을 가지고 있기 때문에 이러한 모바일 환경의 보안 문제에 대응하기 위한 기술로서 주목 받고 있다.

그러나 Xen on ARM[3] 및 KVM-ARM[4] 과 같은 ARM 프로세서 기반의 하이퍼바이저는 그 크기 및 성능 오버헤드로 인해 보안성 향상을 목적으로 실제 적용하기에는 제약사항이 크다. 따라서 모바일 장치의 보안성 향상을 위해서는 최소한의 기능만을 갖춘 경량화된 하이퍼바이저가 필수적이다.

본 논문에서는 ARM 기반의 모바일 단말의 보안성을 높이기 위한 하이퍼바이저의 요구 조건과 실제 모바일 단말에 구현한 결과에 대해 기술하고자 한다.

2. 기반 지식 및 관련 연구

모바일 장치와 같은 임베디드 시스템에서 많이 사용되는 ARM 프로세서는 ARMv7-A 아키텍처에서 하드웨어 기반의 가상화 기술인 ARM Virtualization Extension (ARM-VE)를 지원하고 있다. ARM-VE가 지원되는 ARM 프로세서는 CPU 가상화를 위해 기존의 ARM 아키텍처에 새롭게 추가된 최고 권한 프로세서 모드 (Processor Mode)인 Hypervisor Mode (Hyp Mode)를 포함한다.

또한 ARM-VE는 메모리 가상화를 위해 2nd-level 페이지 테이블 (Page Table)을 지원한다. 이 테이블은 가상 머신이 바라보는 물리 메모리 주소 (Guest Physical Address) 인 IPA (Intermediate Physical Address)를 실제 메모리 주소인 HPA (Host Physical Address)로 전환하기 위해 참조되는 페이지 테이블이다.

한편, 이 페이지 테이블의 각 엔트리(Entry) 들은 각 페이지 별 접근 권한을 설정할 수 있다. 따라서 하이퍼바이저는 이를 통해 가상 머신이 접근 하는 페이지들의 접근 권한을 설정 할 수 있다.

ARM-VE는 I/O 가상화를 위하여 vGIC (Virtualized General Interrupt Controller)를 포함하고 있다. GIC는 외부 장치로부터 발생한 인터럽트에 대한 우선 순위를 결정하고, 실제로 프로세서에게 인터럽트를 발생시키는

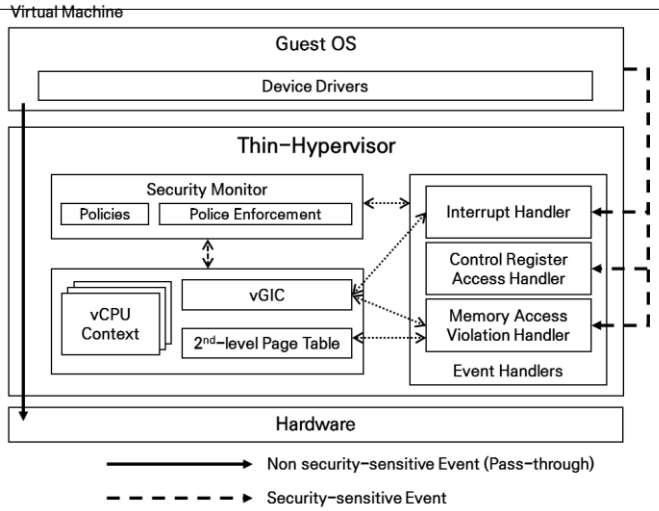

그림 1 경량 하이퍼바이저의 구성. 하이퍼바이저는 가상 머신에 대한 능동적인 개입을 하지 않으며, 발생하는 이벤트들에 대한 처리만을 수행한다.

인터럽트 컨트롤러이다. GIC는 크게 외부로부터 인터럽트를 받아 프로세서에게 전달할 인터럽트를 결정하는 GIC Distributor와, 프로세서에게 실제로 인터럽트를 발생시키는 GIC Interface로 구성되어 있다. vGIC는 GIC Interface를 하드웨어 수준에서 가상화를 지원하며, 하이퍼바이저는 vGIC Interface를 통해 가상 머신에 가상화된 인터럽트를 주입 할 수 있다.

KVM-ARM 및 Xen on ARM 은 기존 x86 프로세서 기반의 하이퍼바이저를 ARM-VE 를 기반으로 옮긴 대표적인 사례이다. 그러나 이 두 가지 모두 Linux Kernel 을 Host OS 및 Domain 0로서 이용하고 있기 때문에 그 크기가 클 뿐만 아니라 한번에 여러 시스템을 동시에 동작시켜야 하므로 성능상 제약이 크기 때문에 모바일 플랫폼의 보안성 향상을 위한 경량 하이퍼바이저로는 적합하지 않다.

3. 경량 하이퍼바이저

3.1. 경량 하이퍼바이저의 필요 요건

모바일 환경의 보안성 향상을 위한 경량 하이퍼바이저는 다음과 같은 조건을 만족하여야 한다.

-**단일 가상 머신 지원**: 모바일 단말과 같은 단일 사용자 환경에서는 다중 VM 지원을 위한 vCPU, 메모리 및 I/O 스케줄링 (Scheduling) 등의 기능은 불필요하다. 따라서 경량 하이퍼바이저는 이러한 기능을 지원하지 않고 최소한의 가상화 역할만을 수행한다.

-**이벤트 기반 동작**: 하이퍼바이저는 가상 머신의 동작에 임의로 개입을 하지 않으며, 가상 머신 수행이나 보안 문제와 관련하여 발생한 이벤트를 처리해 주기 위한 경우에만 수동적으로 실행된다.

-**가상 머신의 디바이스 드라이버 사용**: 특별한 경우를 제외하고, 하이퍼바이저는 인식하고 있는 주변 장치를 가상 머신에게 그대로 보여준다. 즉, 주변 장치들의 관리는 게스트 운영체제가 스스로의 디바이스 드라이버 (Device Driver)를 통해 관리하며 (Pass-through), 보안 정책의 적용이 필요한 경우에만 I/O 과정에서 하이퍼바이저가 개입한다.

-**최소화된 크기 및 성능 저하**: 보안 프레임워크를 포함하는 하이퍼바이저는 최소한의 TCB (Trusted Computing Base)를 통해 그 동작의 안정성을 검증 할 수 있어야 하며, 가상화로 인한 성능 저하 또한 최소화 되어야 한다.

Bitvisor[5] 는 x86 기반에서 제작된 대표적인 경량 하이퍼바이저의 예이다. Bitvisor는 최소한의 가상 머신 동작 환경을 구성 후 대부분의 자원 관리를 게스트 운영체제가 하도록 위임하며, 보안과 관련된 I/O 이벤트가 발생하였을 경우에만 가상 머신의 동작에 동작에 개입하여 그 역할을 수행한다.

3.2. 경량 하이퍼바이저의 구조

그림 1은 3.1 절에서 정의한 요건을 바탕으로 작성된 경량 하이퍼바이저의 구조이다. 하이퍼바이저는 크게 CPU, 메모리, I/O 가상화를 위한 구성요소들과 가상 머신에 적용할 보안 정책을 정의하고 적용하는 Security Monitor, 그리고 가상 머신 내부에서 발생하는 보안 관련 이벤트들을 감시하는 이벤트 핸들러 (Event Handler) 들로 구성된다.

경량 하이버바이저는 기본적으로 하나의 가상 머신만을 지원하며, vCPU Context 저장소, 메모리 가상화 및 접근 제어를 위한 2nd-level 페이지 테이블, 인터럽트 가상화 및 제어를 위한 vGIC 등 최소한의 기능만을 지원 할 뿐 대부분의 하드웨어 제어는 게스트 OS가 수행한다.

Security Monitor는 가상 머신 내부의 보안 정책을 정의하고, 정책에 위배되는 이벤트들이 발생 시 정책에 알맞은 행동을 취한다.

한편, 이벤트 핸들러들은 이러한 보안 정책에 위배되는 이벤트들을 감지하기 위한 역할을 한다. 즉 외부 저장소나 네트워크 등의 장치를 통한 I/O 이벤트, 보호되어야 할 메모리 영역의 허가되지 않은 접근, 프로세서 내 중요한 컨트롤 레지스터들에 대한 쓰기 작업 등이 발생할 경우 이벤트 핸들러는 이러한 이벤트들을 감지하여 Security Monitor에 관련 정책에 알맞은 행동을 수행하도록 요청한다.

4. 경량 하이퍼바이저의 구현

본 논문에서는 Arndale-5250[6] 플랫폼에서 경량화된 하이퍼바이저를 구현하였다. 이 플랫폼은 ARMv7 아키텍쳐를 지원하는 Cortex-A15 프로세서 기반의 Exynos 5250 Dual-Core SoC 및 2GB 메모리를 탑재하고 있다.

.

표 1 ARM-VE 기반 하이버파이저의 가상 머신 내의 이벤트 감시 방법 및 성능 오버헤드

이벤트	감시 방법	검출시간 (usec)
IRQ/FIQ	HCR.IMO / FMO	0.8
Exception / Fault	HCR.TGE	1.5
Control register access	HSTR.Tx	1.4
Memory access	2nd Page Table	1.6

4.1. 가상 머신 내 이벤트 발생 감지

ARM-VE는 가상 머신 내에서 발생하는 많은 이벤트들을 감시하고 해당 이벤트가 발생하였을 경우 Hyp Mode로 전환되도록 하는 기능을 제공하고 있다. 본 논문의 하이퍼바이저는 그 중 **오류! 참조 원본을 찾을 수 없습니다.**과 같은 기능을 통해 가상 머신 내 여러 이벤트들을 감지한다.

기본적으로, ARM-VE는 가상머신 내에서 발생하는 이벤트들의 감시 여부를 HCR (Hypervisor Configuration Register)를 통해 설정 가능하다. HCR의 TGE (Trap General Exception) Bit는 가상 머신 내에서 Exception 및 Fault가 발생하였을 경우 가상 머신의 핸들러가 아닌 Hyp Mode Hyp Trap 핸들러가 가장 먼저 수행되도록 설정하는 역할을 한다.

CP15 와 같은 코프로세서 (Coprocessor) 내 컨트롤 레지스터들에 대한 접근은 HSTR (Hyp System Trap Register)의 T0~T15 Bit를 통해 감시가 가능하며, 해당 Bit들이 1로 설정된 상태에서 컨트롤 레지스터에 접근 시 Hyp Trap 핸들러가 실행되게 된다.

또한 MMIO (Memory-Mapped I/O)등의 감시를 위해 가상 머신이 특정 메모리 주소로 접근 하는 것을 감시하기 위해서는 2nd-Page Table Entry 내의 XN(Never-Execute) Bit 및 AP(Access Permission) Bit를 통해 제어가 가능하다. 만약 허가되지 않은 메모리 영역에 대한 접근이 발생할 경우 Hyp Trap이 발생하게 된다.

한편, 앞서 언급된 여러 원인에 의해 Hyp Trap이 발생한 경우 Hyp Trap 핸들러는 HSR (Hyp Syndrome Register)의 EC (Exception Class) Bit를 통해 그 원인을 파악 할 수 있다.

4.2. 성능 측정 결과

본 논문에서 구현된 하이퍼바이저는 Xen on ARM의 부팅 및 가상 머신 초기화 코드를 기반으로 구현되었다. 이 과정에서 Xen의 vCPU 스케줄링 관련 코드 및 디바이스 드라이버들은 삭제되었으며, 여기에 이벤트 감시를 위한 추가적인 코드들이 추가되었다. 그 결과 본 논문의 경량 하이퍼바이저는 48K LoC 수준의 TCB를 가지고 있으며, 이는 기존 Xen on ARM(131K), KVM-ARM (130K), Bitvisor(194K) 과 비교하여 매우 작은 수준이다.

또한 구현된 게스트 가상 머신 내에서 발생하는 이벤트들을 검출해 내는 오버헤드 측정 결과 표 1**오류! 참조 원본을 찾을 수 없습니다.**과 같이 최대 1.6 usec 가 소모되는 것을 확인 할 수 있다. 특히 IRQ/FIQ 이벤트를 제외한 다른 이벤트는 발생 시 Hyp Mode의 Hyp Trap 핸들러에서 공통적으로 처리를 하기 때문에 HSR 레지스터를 통해서 이벤트 발생 원인을 파악해야 하는 시간이 IRQ/FIQ 이벤트와 비교하여 추가적으로 소요되는 것을 확인 할 수 있다.

5. 결 론

본 논문은 모바일 장치의 보안성 확보를 위한 경량화된 하이퍼바이저의 요건 및 필요 기능을 정의하고, 실제 모바일 단말에서의 구현을 통해 실제 구현 과정에서의 필요 지식들을 제시하였다. 추후 계획으로 구현된 경량 하이퍼바이저를 기반으로 실제 모바일 플랫폼에서 발생할 수 있는 보안 위협을 다루는 보안 프레임워크를 구현 할 예정이다.

6. Acknowledgement

본 연구는 "미래창조과학부 / 정보통신기술진흥센터의 R&D 프로그램 (B0101-15-0239, 인간친화형 디바이스(스킨패치, 멀티모달 서피스) 및 디바이스 소셜 프레임워크 기술 개발])" 과 "Brain Korea 21 PLUS 포스텍 컴퓨터공학 사업단 (F15SN02D1108)" 의 지원을 받아 수행되었음.

<참고 문헌>

[1] http://fninside.hyundaicapital.com/408
[2] http://kr.aving.net/news/view.php?articleId=203687
[3] Dall, Christoffer, and Jason Nieh. "KVM/ARM: the design and implementation of the linux ARM hypervisor." Proceedings of the 19th international conference on Architectural support for programming languages and operating systems. ACM, 2014.
[4] Hwang, Joo-Young, et al. "Xen on ARM: System virtualization using Xen hypervisor for ARM-based secure mobile phones." Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE. IEEE, 2008.
[5]Shinagawa, Takahiro, et al. "Bitvisor: a thin hypervisor for enforcing i/o device security." Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. ACM, 2009.
[6]http://www.arndaleboard.org