보안 IP 개발을 위한 SoC 플랫폼의 구현

이지훈, 문현곤, 이진용, 김용주, 백윤흥
서울대학교 공과대학 전기컴퓨터공학부
e-mail : {jhlee, hgmoon, jylee, yjkim}@sor.snu.ac.kr, ypaek@snu.ac.kr

An implementation of SoC platform for security IP development

Jihoon Lee, Hyungon Moon, Jinyong Lee, Yongjoo Kim, Yunheung Paek
School of Electrical and Computer Engineering, Seoul National University

요 약

스마트폰의 보급과 더불어 개인 정보를 유출하는 악성 프로그램의 위협 또한 증가하고 있다. 악성 프로그램의 위협으로부터 사용자의 데이터를 보호하기 위해 다양한 모바일용 백신이 시중에 나와있는 상황이다. 하지만 일반 컴퓨팅환경의 경우를 보듯이 소프트웨어만으로는 모든 악성 프로그램의 위협에 대처하는 것은 상당히 어렵다. 이러한 단점을 극복하기 위해서 하드웨어의 도움을 받는 선행연구들이 있었지만 스마트폰과 같은 SoC 구조에 적용하기에는 무리가 따른다. 따라서 본 논문에서는 임베디드 시스템의 보안성 향상을 위한 IP 를 개발/실험 할 수 있는 SoC 플랫폼을 구현하도록 한다.

1. 서론

최근 설계 기술의 발달과 반도체 공정의 미세화로 인해 과거 데스크톱 CPU 의 성능은 뛰어 넘으면서 전력은 더 적게 소비하는 모바일 CPU 가 등장하고 있다. 이를 바탕으로 모바일기기는 급속도로 발전하였으며 국내에서는 아이폰의 보급을 시발점으로 스마트폰 열풍이 일어났다. 과거 모바일기기는 제한된 성능과 리소스로 인해서 사용자가 개입할 수 있는 여지를 차단한 폐쇄적인 환경에서 동작을 하였다. 따라서 유저가 직접 프로그램을 추가하거나 시스템의 구성을 변경할 수 없었다. 스마트폰의 경우는 앞서 언급한 모바일 CPU 의 성능을 바탕으로 일반 컴퓨터를 사용하는 것과 비슷한 컴퓨팅 환경을 제공할 수 있게 되었다. 간단한 문서편집은 물론이고 일반 데스크톱에서 인터넷으로 할 수 있는 대부분의 것(예: 인터넷 뱅킹, 증권거래, 웹서핑 등)을 할 수 있다. 또한, 스마트폰을 개발한 회사가 아닌 개인이 만든 프로그램을 설치하여 사용할 수 있으며 더 나아가 jail breaking 을 통해서 초기에 나온 환경과 다른 환경으로 만들어 불법적인 프로그램을 사용하는 경우도 있다. 개방적인 환경으로 인해서 사용자는 자신의 의도하였던 의도하지 않았던 간에 각종 악성프로그램의 위협에 노출되기 쉽다. 보안업체 중 하나인 McAfee 의 보고서에 의하면 그림 1 에서 볼 수 있듯이 안드로이드 기반에서 동작하는 악성프로그램은 더욱 증가하고 있다는 것을 알 수 있다[1]. 이 때문에 악성프로그램으로부터 사용자의 정보를 보호하기 위해서 다양한 스마트폰용 백신이 시중에 나와있다[2]. 하지만 데스크톱의 경우를 보듯이 백신도 OS 에 의존성이 있기 때문에 기존에 알려지지 않은 악성코드들이 OS 나 백신의 취약점을 노린다면 무력화되기 쉬운 측면이 있다. 따라서 소프트웨어의 단점을 극복하기 위해서 하드웨어의 도움을 받아 시스템의 보안을 향상시키려는 선행연구들이 있었다. 그러나 선행연구는 일반적인 컴퓨팅 환경을 목표로 해서 설계를 하였기 때문에 모바일기기와 같은 임베디드 시스템에는 적용하기 어려운 단점이 있다.

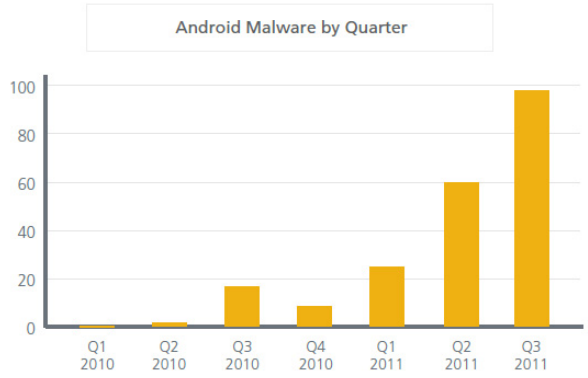

그림 1 분기별 안드로이드 악성프로그램 수

따라서 본 논문에서는 임베디드 시스템의 보안 향상을 목적으로 하는 IP 를 개발/실험 할 수 있는 플랫폼에 대해 설명하고 이를 구현한다. 2 장에서는 기존의 하드웨어기반의 연구에 대해 살펴본 뒤 3 장에서는 구현한 SoC 플랫폼에 대해서 설명하도록 한다. 4 장에서는 결론과 향후 연구방향에 관해서 논하도록 하겠다.

2. 관련연구

기존의 하드웨어를 이용해 보안을 향상시킨 연구로는 Copilot 이 있다[3]. Copilot 은 pci 인터페이스를 사용

하는 EBSA-285 보드로 구현하였으며 호스트 컴퓨터 메인 메모리의 데이터를 가져와 분석할 수 있는 인터페이스와 외부에 보고할 수 있는 통신 모듈이 있다. 일정 주기마다 DMA 를 통해서 커널이 위치하고 있는 메모리 영역의 데이터를 읽어와서 이를 hash 한다. 커널의 특정영역은 동작 중에 그 값이 변하지 않고 보조기억장치로 페이징하지 않는다는 특성을 이용한 것이다. 만약 hash 값이 변한다면 운영체제의 무결성이 깨졌다고 판단해 이를 외부와 연결된 통신 모듈을 통해서 보고를 한다. 실험결과에 따르면 Copilot 의 방식은 메모리의 대역폭을 소비하므로 최대 13%의 성능이 감소된다.

3. SoC 플랫폼의 구현

앞 장에서 언급한 Copilot 과 같은 형태는 임베디드 시스템에 구현하기는 어렵기 때문에 다른 방식으로 바꾸어 구현해야 한다. 기본이 되는 SoC 시스템을 구현하고 그 위에 IP 를 추가하는 방법으로 구현되어야 할 것이다. 여기서 구현되는 SoC 는 리눅스가 구동 가능해야 하며 버스 인터페이스는 일반적으로 잘 알려진 스펙을 가지고 있어야 할 것이다.

구현한 시스템은 gaisler 사에서 제공하는 grlib 을 이용하였다[4]. grlib 에는 sparc 프로세서를 바탕으로 만들어진 Leon3 프로세서가 AMBA2.0 스펙에서 동작 가능하도록 HDL 코드형태로 포함되어 있다. 또한 Leon3 프로세서에서 구동이 가능한 스냅기어 리눅스를 제공하고 있다. 따라서 AMBA2.0 스펙을 참조해 보안기능을 제공하는 IP 의 제작이 가능하며, 따로 리눅스를 포팅해야 하는 문제점도 해결이 된다. Grlib 에 구현된 모듈은 다양한 FPGA 에서 동작 할 수 있도록 호환성을 가지게 디자인되어 있어 자신이 구현하고자 하는 FPGA 에 따라 설정을 달리 해주어야 한다. 구현과정에서는 virtex5 코어타일을 포함한 RPS-3000k 보드를 사용하였고 구현된 시스템은 아래 그림 2 와 같다.

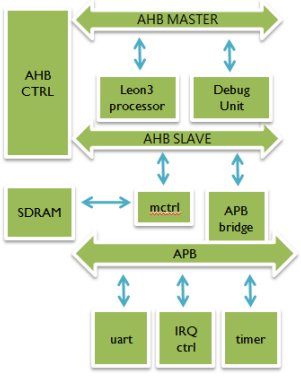

그림 2 초기 구현한 SoC 플랫폼

AHB master 에는 Leon3 CPU 와 하드웨어로 접속 가능하게 인터페이스를 제공해 주는 디버그 모듈로 구성된다. AHB slave 에는 64MB onboard sdram 구동을 위한 메모리컨트롤러와 기타 주변장치를 위한 APB bridge 로 구성된다. APB 장치는 외부통신을 위한 UART, timer 등으로 구성된다. 이 외에 master 와 slave 의 버스사용을 통제하는 AHB 컨트롤러로 구성된다. FPGA 에서 동작하는 하드웨어를 컨트롤하기 위해서는 gaisler 에서 제공하는 grmon 프로그램을 이용한다. shell 에서 JTAG 케이블을 통해 디버그 모듈에 접속할 수 있다. 구현된 하드웨어의 동작을 검증하기 위해 스냅기어 리눅스를 현재 구현한 하드웨어구성에 맞게 설정하고 컴파일하여 구동시켜 보았으며 그림 3 과 같이 정상적으로 동작하는 것을 확인하였다.

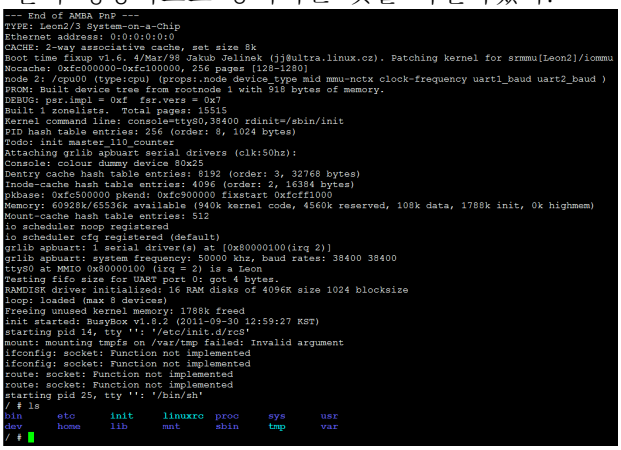

그림 3 스냅기어 리눅스의 구동화면

초기 구현한 플랫폼의 master 나 slave 형식으로 프로세서와 독립적으로 동작하는 보조 IP 의 개발이 가능할 것이다. 하지만 기존에 제시된 copilot 과 같은 coprocessor 형태의 개발은 불가능한 단점을 지니고 있다. 플랫폼의 활용 가능 범위를 넓히기 위해서는 별도의 코어가 존재하고 이를 관리할 수 있어야 하며 독립적인 메모리 공간이 필요하다. 이를 해결하기 위해서 그림 4 와 같이 시스템의 구성을 변경하였다.

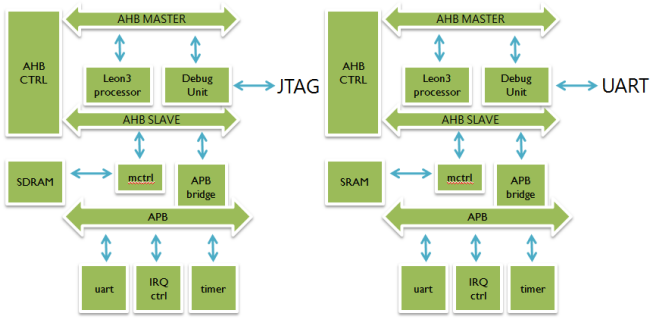

그림 4 최종 완성된 SoC 플랫폼

초기 구현한 시스템은 JTAG 을 통해서 유저와 통신을 한다. 따라서 추가된 시스템의 경우는 JTAG 을 통해서 유저와 통신을 할 수 없는 문제점이 생기게 된다. 이를 해결하기 위해서 grlib 에서 제공하는 JTAG 을 사용하지 않고 UART 를 통해 하드웨어에 접속할 수 있는 모듈로 변경하였다. 또한 독립적인 메모리 영역이 필요하므로 이를 위해 기존의 sdram 이 아닌 sram 을 사용할 수 있도록 메모리 컨트롤러를 구현하였다. 완성된 플랫폼은 50MHz 의 동작주파수를 가지고 있으며 각각 64MB, 2MB 의 메모리 공간을 독

립적으로 가지게 된다. 유저는 독립적인 2 개의 시스템을 서로 다른 채널을 사용해 컨트롤할 수 있으며 한 개의 시스템은 coprocessor 와 같은 형태로 사용이 가능하도록 최종 구현되었다.

4. 결론 및 향후 연구과제

일반적인 컴퓨팅환경에서 보안을 향상시키기 위해 제시된 하드웨어는 SoC 환경에서는 동일하게 적용하기 어려운 점이 있었다. 이러한 문제를 해결하고자 본 논문에서는 임베디드 시스템 보안 IP 개발에 활용 가능한 SoC 플랫폼을 구현하였다. 향후 구현된 플랫폼을 기반으로 coprocessor 형태로 동작하는 리눅스 커널 무결성 검사도구의 개발이 가능할 것으로 생각한다. 이를 구현하게 되면 메인 시스템의 운영체제와는 독립적으로 동작하게 되므로 운영체제에 의존적인 부분에서 나타나는 취약점을 상당부분 해결할 것으로 예상된다. 그 외에도 복잡한 보안 알고리즘의 연산을 하드웨어로 대체하는 IP 를 추가해 메인 프로세서의 부담을 덜어주는 식의 개발이 가능할 것으로 생각된다.

Acknowledgement

본 연구는 교육과학기술부/한국과학재단 우수연구센터 육성사업(과제번호 2012-0000470), 2011 년도 정부(교육과학기술부)의 재원으로 한국과학재단의 국가지정연구실사업(No.2011-0018609) 및 IDEC 의 지원을 받아 수행되었습니다.

참고문헌

[1]http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q3-2011.pdf

[2]http://www.ahnlab.com/kr/site/product/productView.do?prodSeq=66

[3]Nick L. Petroni Jr., Timothy Fraser, Jesus Molina, and William A. Arbaugh. Copilot - a coprocessor-based kernel runtime integrity monitor. In Security '04: Proceedings of the USENIX Security Symposium, 2004.

[4]http://www.gaisler.com/cms/index.php?option=com_content&task=view&id=115&Itemid=103