임베디드 리눅스에서 서명 검증 방식을 이용한 악성 프로그램 차단 시스템*

이종석° 정기영 정다니엘 김태형 김유나 김종
포항공과대학교 컴퓨터공학과
{ spicajk, zzangky, dennis, kth, existion}@postech.ac.kr jkim@postech.ac.kr

Preventing ELF-File-Infecting Malware using Signature Verification for Embedded Linux

JongSeok Lee° Ki Young Jung Daniel Jung Tae-Hyung Kim Yuna Kim Jong Kim
Department of Computer Science and Engineering
Pohang University of Science and Technology (POSTECH)

해를 거듭할수록 급속히 발전하는 정보통신 기술은 인간의 삶의 질을 향상시키는 기반이 되고 있지만, 그에 따라 보안 위협도 꾸준히 증가하고 있다. 이러한 보안 위협은 모바일 기기가 발달함에 따라 기존의 PC 및 서버 등의 플랫폼을 넘어서 모바일 임베디드 시스템에서도 확산되는 추세이다.

이에 맞춰서 여러 회사에서 Palm, Symbian OS, Windows CE, Window Mobile, Pocket PC 등의 모바일 임베디드 시스템을 위한 보안 프로그램을 제작하기 시작했다. 그러나 임베디드 시스템이 점점 더 좋은 성능과 대용량의 자원을 갖추게 되고 그에 따른 시스템의 활용 방안에 대한 요구도 더욱 커지고 있는 상황에 비추어 볼 때, 기존의 임베디드 시스템 운영체제는 확장성, 안정성, 호환성 등의 측면에서 미흡하다. 그래서 향후 더욱 발전될 임베디드 시스템을 효과적으로 사용하여 성능을 극대화시키기 위한 방안으로 리눅스를 임베디드 시스템에 적용시키고자 하는 연구가 활발히 진행되고 있는데, 이러한 임베디드 리눅스가 성공적으로 정착이 되기 위해서는 그것을 보호해줄 수 있는 보안 프로그램 개발이 반드시 필요하다.

그런데 기존의 모바일 임베디드 시스템에서 사용되어 온 운영체제들과 달리 임베디드 리눅스는 PC 및 서버에서 주로 사용되어 온 범용 운영체제를 임베디드 시스템에 맞게 적용했기 때문에 기존에 존재하던 보안 문제와 위협들이 그대로 전이될 수 있다. 대부분의 프로그램 소스 코드가 공개되어 있어서 누구나 수정 및 배포가 가능하다는 점은 이러한 위험성을 증대시킨다. 따라서 임베디드 리눅스를 공격하는 악성 프로그램들이 증가하기 전에, 기존에 존재하던 악성 프로그램들을 참고하여 잠재적인 공격에 대비해야 한다.

리눅스를 위협하는 악성 프로그램들은 주로 ELF (Executable and Linking Format) 형식의 바이너리 파일을 변형시켜 악성 코드를 삽입, 실행하려 한다. 이러한 방법은 다른 파일을 감염시켜 자가 복제를 계속 발생시키는 고전적인 형태의 바이러스 외에도 웜, 트로이 목마, 루트킷 등 다양한 유형의 공격에서 활용된다. 예를 들어 RST (Remote Shell Trojan)의 경우 /bin 디렉토리 안에 있는 모든 ELF 바이너리 파일을 감염시켜 루트 권한을 가지는 쉘을 백도어를 통해 실행할 수 있게 한다. 기존의 많은 연구들이 이러한 악성 프로그램들을 막기 위해 방화벽, 침입 탐지 시스템, 바이러스 백신 등을 제시해 왔으나, 시그니쳐 기반의 탐지 방법의 한계로 인해 제로데이 (zero-day) 공격과 다양한 변형 공격에 약점을 보여 임베디드 시스템의 안정성과 가용성을 완벽히 보장하기가 어렵다. 악한 공격에도 큰 피해를 입을 수 있는 임베디드 시스템의 환경을 고려할 때, 이런 문제점은 반드시 해결이 되어야 한다. 또한 안정성과 가용성을 보장하기 위해서는 시스템이 안전하게 유지가 되는지 실시간으로 확인해야 하나, 기존 어플리케이션 기반의 방법들은 하나의 독립적인 프로세스로서 메모리에 상주해야 하기 때문에 임베디드 시스템의 자원을 일정부분 지속적으로 소비하는 문제점이 생긴다. 따라서 기존의 PC 및 서버 등의 환경에서 사용되었던 보안 시스템보다 임베디드 시스템의 자원을 효율적으로 사용하는 방법이 필요하다.

한편 각 시스템에서 사용되는 임베디드 리눅스가 범용 리눅스와는 다르게 특정한 목적을 위한 하드웨어에 최적화된다는 점을 감안하여 기존 시스템의 변경을 필요로 하지 않으면서 쉽고 빠르게 시스템에 적용/해제가 가능한 방법을 개발하여야 한다.

따라서 본 논문에서는 임베디드 시스템에 쉽게 적용이 가능하며, 오버헤드를 최소한으로 줄일 수 있는, ELF 바이너리 파일 대상 악성 프로그램 실시간 감시 방법으로 커널 레벨에서 서명 검증 기법을 이용한 악성 프로그램 차단 시스템을 제안한다.

그림 1은 제안하는 시스템의 구조를 표현하고 있다. 메모리에 상주하여 악성 프로그램을 검색하는 일반적인 실시간 감시 프로그램과는 달리, 제안하는 시스템은 ELF 바이너리 파일이 실행 명령 후 커널 레벨에서 파일이 실행되는 순간 그 파일에 대해 검사를 한다.

이 때 프로그램을 실행할 때마다 어떤 악성 프로그램에 감

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2007-C1090-0701-0045)

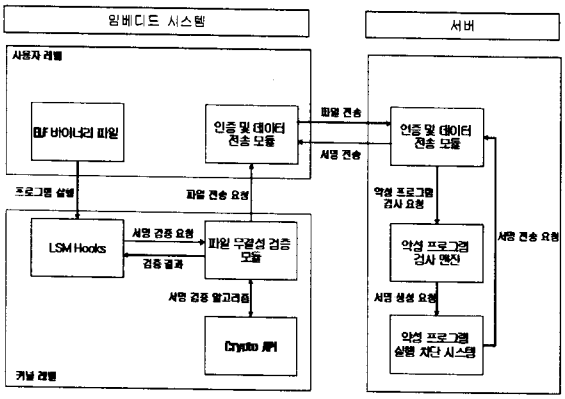

그림 1. 제안 시스템 구조

염되었는지 자세히 검사를 하는 것은 임베디드 시스템에 부담을 주게 되므로, 서명 검증 기법을 사용하여 실행하고자 하는 바이너리 파일의 변조 여부를 먼저 확인한 뒤, 파일이 변경되었거나 아직 서명을 생성하지 않았을 경우에만 악성 프로그램 검사를 하도록 한다. 악성 프로그램 검사는 임베디드 단말에서 하지 않고, 별도의 서버로 파일을 전송하여 수행한다. 이와 같은 방법을 사용하는 이유는 임베디드 시스템 내부에 악성 프로그램 검사 엔진을 둘 경우 전체 악성 프로그램 목록을 검사하는 과정을 수행하는 데 필요한 자원을 임베디드 시스템이 감당하기 힘들고, 악성 프로그램 엔진에서 사용하는 데이터베이스를 유지하고 갱신하는 것 또한 시스템 자원을 필요로 하기 때문이다.

서버는 임베디드 시스템으로부터 전송받은 악성 프로그램을 검사하여 안전한 파일로 판정될 경우, 검사한 파일에 대한 서명을 생성하여 임베디드 시스템으로 전송하여 바이너리 파일의 끝에 첨가한다. 이후 해당 파일의 실행 시에는 시스템에서 파일의 서명 검증과정을 통해 감염이 안 된 안전한 파일로 판단하고 호출을 계속 수행한다. 그러나 서명 검증과정이 실패한 경우에는 서명 생성 후 파일에 대한 변형이 발생한 것이므로 다시 서버로 보내어 파일이 안전하다는 검증 과정을 거쳐야 한다.

제안하는 방법이 효과적으로 ELF 바이너리 파일의 변조를 방지하는지 확인하기 위해 ELF 바이너리 파일을 변조하는 공격들을 수행하였다. 실험 결과, ELF 바이너리 실행 파일을 감염시키는 바이러스들은 물론, 공유 라이브러리를 변조하여 백도어를 설치하는 루트킷 또한 차단하는 것을 확인할 수 있었다. 또한 제안하는 방법을 임베디드 시스템에 적용하였을 때 발생하는 오버헤드를 분석하기 위해 적용 전후의 프로그램 실행부터 종료까지 걸리는 수행 시간을 비교하였다. 수행 시간을 비교할 때 사용한 ELF 실행 파일은 /bin/busybox ls와 /bin/busybox pwd이다. 표 1은 일반 커널과 제안하는 방법을 적용한 커널에서 측정한 수행 시간 결과이다. 표 1에서 real은 프로그램의 전체 수행 시간을 나타내고 user와 sys는 각각 사용자 레벨에서의 수행 시간과 커널 레벨에서의 수행시간을 나타낸다.

실험 결과 일반 커널에 비해 제안하는 방법을 적용한 커널에서의 수행 시간이 증가하는 것을 확인할 수 있다. 그러나

표 1. 프로그램 수행 시간 비교

	/bin/busybox ls -al	/bin/busybox pwd
일반 커널	real 0.03s user 0.01s sys 0.02s	real 0.00s user 0.00s sys 0.01s
제안하는 방법을 적용한 커널	real 0.59s user 0.02s sys 0.51s	real 0.23s user 0.00s sys 0.18s

위의 예제들이 사용자 레벨에서의 수행 시간은 거의 없고 커널 레벨에서의 수행 시간도 매우 짧은 프로그램들이라 성능 평가에 있어서 가장 나쁜 조건 (Worst Case)이라는 점과 실험에 참여한 사용자들이 느끼기엔 수행 속도의 차이가 체감적으로 거의 없을 정도라는 사실에 주목하자. 게다가 일반적으로 사용자가 오래 사용하는, 즉 수행 시간이 긴 프로그램들은 사용자 단계에서 대부분의 작업을 수행하기 때문에, 제안하는 방법은 처음에 프로그램을 실행시키는 과정 중에 커널 레벨에서 단 한 번만 수행되므로 전체적인 시스템 성능에 큰 지장을 주지 않는다. 또한 기존의 리눅스 시스템에서 사용되는 ELF 바이너리 파일의 크기도 대부분 512KB 이하라는 점을 감안할 때 임베디드 시스템에서 사용되는 바이너리 파일의 크기 증가에 따른 수행 시간 증가도 크게 우려하지 않아도 될 것이다. 따라서 제안하는 방법이 효과적으로 시스템을 보호하는 것을 확인할 수 있다.

실험 결과로 보아 제안하는 방법은 추가적인 하드웨어 장비 없이 메모리에 프로세스를 상주시키지 않으면서도 실시간으로 악성 프로그램을 효과적으로 탐지할 수 있다. 즉, 효율적이고 안전한 시스템 유지가 가능하다. 또한 알려지지 않은 신종 공격에 바로 노출되지 않아 대응할 수 있는 시간을 확보할 수 있으며, 공인된 서버를 사용하여 검증하기 때문에 검사 결과에 신뢰성이 높고 시스템의 유지보수가 쉽고 빠르다는 장점을 가지며, 커널 모듈로 구현하여 기존에 사용되던 임베디드 리눅스에 쉽고 빠르게 적용 및 제거가 가능하다.

제안한 방법은 ELF 바이너리 파일의 변조 여부로 공격 여부를 판단하기 때문에, 인증을 받은 정상적인 프로그램 내부에 존재하는 취약점을 통해 직접 쉘 코드를 실행시키는 형태의 공격을 방어하지 못한다. 또한 제한하는 방법을 실제 모바일 환경에서 적용할 경우, 공격자가 임의의 악성프로그램 파일들을 이동 단말 기기에 대량으로 배포하여, 임베디드 시스템들로 하여금 동 시간대에 패킷 전송량을 비정상적으로 증가시켜 서버와 임베디드 시스템 사이의 네트워크를 공격하는 일종의 DoS (Denial of Service) 공격을 시도할 수 있다.

향후 연구는 이러한 문제점들을 해결하기 위해 정상 프로그램들의 취약점을 통해 공격을 받더라도 시스템을 안전하게 보호할 수 있는 메모리 보호 기법과, 기존에 연구되었던 모바일 환경에서의 DoS 공격 차단 기법들을 본 논문에서 제안한 방법에 적용시키는 방향으로 진행할 것이다. 또한 제안 시스템의 효율성을 증대시키기 위해 전력 소비량과 같은 수행 시간 외의 다른 측면에 대한 성능 평가를 수행하고, 그것을 토대로 서명 검증 단계나 서버와 임베디드 시스템 사이의 인증 단계의 알고리즘 성능 개선을 통한 최적화를 진행할 계획이다.