임베디드 시스템에서 권한 제한을 통한 가상 컨테이너 활용 방안

Utilization of Virtual Containers via Privilege Restriction in Embedded Systems

차 주 형†, 권 용 인*‡
†과학기술연합대학원대학교, ‡한국전자통신연구원
(JooHyoung Cha, Yongin Kwon)
(†University of Science and Technology, ‡Electronics and Telecommunications Research Institute)

Abstract : 리눅스는 현대 컴퓨팅 환경에서 임베디드 시스템부터 클라우드 컴퓨팅까지 다양한 분야에서 널리 활용되는 운영체제이다. 특히 리눅스 커널은 자유로운 수정과 재구성이 가능하여 목적에 따라 운영체제가 제공하는 기능을 다르게 구현할 수 있다. 그러나 임베디드 시스템이나 자원 제약이 있는 하드웨어에서는 도커(Docker)와 같은 컨테이너 기술의 사용이 어려워 새로운 기능의 확장이나 개발 환경 구축에 한계가 있다. 본 연구에서는 리눅스 커널과 루트 파일 시스템과 루트 디렉토리 변경하여 컨테이너를 생성하는 방법을 제시하고자 한다. 이를 통해 도커 사용이 어려운 환경에서도 다른 운영체제의 루트 파일 시스템을 활용하여 시스템을 확장하고, 개발 및 실험 환경을 구축할 수 있음을 확인하였다.

Keywords : linux, kernel, container, file-system, embedded

I. 서 론

리눅스 커널은 다양한 운영체제에서 표준으로 사용되며, 유연성과 개방성으로 인해 임베디드 시스템부터 대규모 서버 환경까지 폭넓게 채택되고 있다[1]. 자원이 제한된 임베디드 시스템에서는 불필요한 커널 기능을 제거하여 자원의 사용을 최소화하고, 이를 통해 성능을 최적화할 수 있다. 반면, 서버 환경에서는 가상화 기능을 활성화하여 커널 수준에서 시스템 자원의 격리와 모니터링을 수행한다.

개발 단계의 임베디드 보드는 커널 수정과 다양한 편의 도구가 많이 제공되는 반면, 제품으로 출시된 시스템은 불필요한 기능을 가능한 모두 제거하여 배포된다. 이로 인해 일부 임베디드 시스템에서는 실험과 연구에 제약이 발생하게 된다. 대표적으로 많은 종속성이 존재하는 소프트웨어나 컨테이너를 사용하는 데 확장성과 이식성에 어려움이 있다[2].

안드로이드나 라우터에서는 보안 강화나 운영체제의 경량화를 위해 많은 기능이 제거되어 배포되므로 접근성 문제가 발생한다. 터미널, 패키지 매니저, 표준 C++ 라이브러리 등의 부재로 인해 인공지능 연산을 수행하거나, 파이썬 기반의 라이브러리를 활용하게 되는 경우 복잡한 종속성 문제로 인해 환경 구성에 제약이 따른다.

또한, 컨테이너는 시스템 자원과 프로세스간 리소스를 독립적으로 사용하기 위해 cgroups과 네임스페이스 그리고 chroot를 활용한다. 그리고 각각의 기능은 리눅스 커널에서 동작하고 있다. 이로 인해, 사전에 제거된 기능을 추가하기 위해 시스템 마다 커널을 수정하여 검증해야하는 제약이 발생한다[3].

본 논문에서는 1982년 BSD 4.2에서 제안된 초기 버전의 컨테이너 개념[4]을 기반으로 연구를 수행하였다. 초기 버전의 컨테이너는 새로운 리눅스의 루트 파일 시스템(rootfs)으로 루트 디렉토리를 변경하는 방식으로 구현되었다. 본 연구에서는 커널의

*Corresponding Author : yongin.kwon@etri.re.kr
차주형: 과학기술연합대학원대학교
권용인: 한국전자통신연구원
※ 이 논문은 2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구 결과 임 (No.RS-2024-00459797, 온디바이스 AI를 위한 ML컴파일러 프레임워크 기술 개발)

수정 없이 파일 시스템과 루트 디렉토리만 변경하여 Debian 계열 운영체제와 SSH 서버를 실행함으로써 새로운 확장 가능성을 제시한다. 이를 통해 커널을 수정하지 않고도 파일 시스템과 루트 디렉토리만 변경하여 최신 운영체제에서도 가상 컨테이너의 구현이 가능함을 확인하였다.

Ⅱ. 실 험

본 연구에서는 작업 영역의 격리만을 활용한 컨테이너를 통해 새로운 운영체제가 동작하는지와 SSH 서버를 구동하여 외부에서 접근이 가능한지를 실험적으로 확인하고자 한다. 기존의 Docker와 같은 컨테이너는 하드웨어 자원, 프로세스, 작업 영역의 격리와 같은 가상화를 수행하지만, 임베디드 시스템에서는 커널 레벨의 제약으로 인해 구현에 어려움이 있다. 따라서, 커널 수정 없이 시스템 기능을 확장할 수 있는 방법을 모색하고자 한다[5].

실험에서 리눅스 파일 시스템을 미리 패키지화한 Debian 운영체제의 rootfs를 활용하여 루트 디렉토리를 변경함으로써 새로운 환경을 구축하였다. 이를 통해 두 가지 하드웨어 환경에서 chroot와 rootfs를 사용하여 시스템 기능 확장 가능성을 실험하였다. 표 1은 실험에 사용된 하드웨어 구성을 나타낸다.

표 1. 실험에 사용된 하드웨어 구성
Table 1. Description of Hardware Configuration

	Qualcomm HDK865	IPTime A1004NS
Type	Smart Phone	Router
OS	Android 10	BusyBox v1.8.2
CPU	SD 865	MT7620A
Arch.	Arm	MIPS
Endian	Little Endian	Little Endian

1. Qualcomm HDK865 개발용 보드에서의 실험

안드로이드 운영체제의 터미널로 접근하기 위해 ADB를 활용하여 시스템에 접근하였다. 그러나 경량화와 보안으로 인해 chroot를 지원하지 않기 때문에, 이를 우회하기 위해 PRoot를 활용하였다. PRoot는 사용자 공간에서 chroot, 마운트(bind) 기능을 구현한 도구로, 루트 권한 없이 새로운 루트 파일 시스템을 설정할 수 있게 하는 도구이다[6]. PRoot를 통해 데비안 계열의 rootfs를 활용하여 격리된 실행 환경을 구축할 수 있었다. 추가적인 커널 수정이나 시스템 변조 없이 새로운 운영체제를 실행할 수 있었다. 그림 1은 안드로이드 환경에서 데비안 계열 운영체제인 우분투의 rootfs를 활용하여 새로운 운영체제를 실행한 결과를 보여준다.

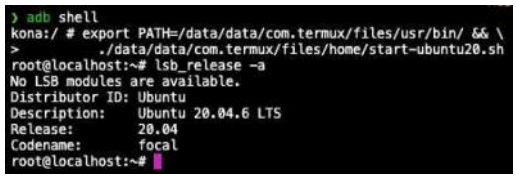

그림 2. Q. HDK 865에서 Ubuntu OS의 lsb_release 실행 결과
Fig. 2. Result of executing lsb_release on Ubuntu OS in Q. HDK 865

2. IPTime A1004NS 공유기에서의 실험

A1004NS 공유기는 MIPS 아키텍처를 기반으로 하며, 디버깅 포트를 통해 터미널에 접근할 수 있었다. 펌웨어의 정보를 통해 해당 공유기가 기본적으로 chroot 기능을 포함하고 있음을 확인하였다. 이를 바탕으로 Debian의 MIPS용 rootfs를 활용하여 공유기에서도 격리된 환경에서 새로운 운영체제를 실행할 수 있음을 검증하였다. Ubuntu의 경우 MIPS CPU에 대한 지원이 부족하여 Debian 운영체제를 선택하여 실험을 진행하였다.

Debian의 MIPS용 rootfs를 적용함으로써, 공유기에서도 chroot를 통해 격리된 실행 환경을 성공적으로 구축할 수 있었다. 그림 2는 IPTime A1004NS 공유기에서 Debian의 rootfs를 활용하여 새로운 운영체제를 실행한 결과를 보여준다.

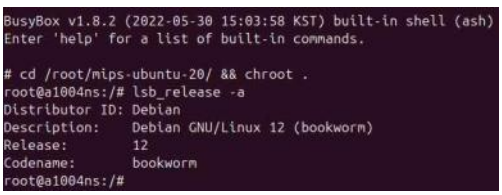

그림 2. IPTime A1004NS에서 Debian OS의 lsb_release 실행 결과
Fig. 2. Result of executing lsb_release on Debian OS in A1004NS

두 가지 하드웨어 환경에서 커널 수정 없이 chroot와 rootfs를 활용하여 컨테이너를 생성함으로써 새로운 운영체제 환경을 성공적으로 구축하였다.

lsb_release 명령어는 Python 3을 기반으로 동작하므로, 모든 하드웨어 환경에서 Python 3이 설치되어 있음을 확인할 수 있었다.

추가적으로, 외부 접근성을 검증하기 위한 실험으로 Debian 계열 운영체제의 패키지 매니저를 통해 SSH 서버를 설치하였다. 그 결과, Android 계열 운영체제와 BusyBox 기반 공유기에서도 컨테이너를 통해 SSH 접근이 문제없이 가능함을 확인하였다. CPU의 아키텍처의 종류와 무관하게 운영체제의 플랫폼의 지원이 충분하다면 다양한 임베디드 장치에서 유연한 확장이 가능함을 나타낸다.

Ⅲ. 한 계 점

시스템 자원과 프로세스 간의 격리 기능이 없는 커널 환경에서는, 커널을 수정하지 않는 한 컨테이너 런타임을 활용할 경우 보안적 문제와 지원되지 않는 문제가 발생할 수 있다. 본 연구에서는 루트 디렉토리의 변경을 통해 이러한 문제를 어느 정도 완화할 수 있음을 확인하였으나, 커널 드라이버나 하드웨어 가상화 기술을 활용하여 시스템을 확장하는 데에는 여전히 어려움이 존재함을 발견하였다.

또한, 리눅스 커널을 공유하는 컨테이너 환경에서 시스템 명령어의 오용은 상위 시스템에 부정적인 영향을 미칠 수 있는 문제를 초래할 수 있다. 예를 들어, 시스템의 강제 종료나 파일 시스템 잠금을 통해 하위 시스템이 상위 시스템의 접근을 통제할 가능성이 존재한다.

이러한 한계로 인해, 외부에 공개된 환경에서 해당 방법론을 적용하기 위해서는 추가적인 보안 조치와 관리가 필수적이다. 또한, 제안된 방법을 통해 생성된 컨테이너를 활용하는 것은 인하우스 환경에서 가장 적합하다는 점을 시사한다.

Ⅳ. 결 론

본 연구에서는 리눅스 커널과 rootfs, chroot를 활용하여 도커 사용이 어려운 임베디드 시스템에서 컨테이너를 통해 새로운 운영체제가 동작이 가능함을 확인하였다. 또한, 격리된 파일 시스템 환경을 구축하여 커널의 수정 없이도 다른 운영체제의 파일 시스템을 효과적으로 활용할 수 있었으며, 이를 통해 전문적인 기술 없이, 변경이 불가능한 시스템에서 개발이 가능함을 확인하였다.

향후 연구에서는 루트 디렉토리 변경으로 인한 한계점을 극복하기 위해 안정성과 보안 문제를 해결하는 추가적인 연구를 진행하고자 한다.

References

[1] J. Watada, A. Roy, R. Kadikar, H. Pham, and B. Xu, “Emerging Trends, Techniques and Open Issues of Containerization: A Review,” IEEE Access, vol. 7, pp. 152443-152472, 2019

[2] U. Ashish Bijlani, "Extension Framework for File Systems in User space," in 2019 USENIX Annual Technical Conference (USENIX ATC 19), 2019, pp. 121 - 134.

[3] T. Sutter, T. Kehrer, M. Rennhard, B. Tellenbach and J. Klein, "Dynamic Security Analysis on Android: A Systematic Literature Review," in IEEE Access, vol. 12, pp. 57261-57287, 2024, doi: 10.1109/ACCESS.2024.3390612.

[4] Joy, W., et al, “4.2 BSD system manual,” Technical report, Computer Systems Research Group, University of California~…, Tech. Rep., 1983.

[5] E. Reshetova, J. Karhunen, T. Nyman, and N. Asokan, ‘Security of OS-Level Virtualization Technologies’, in Secure IT Systems, 2014, pp. 77 - 93.

[6] C. Chenard and P. community, “Proot - chroot, mount - bind, and binfmt misc without privilege/setup for linux,” 2012. [Online]. Available: https://github.com/proot-me/proot