2-2023

Poster: Userland Containers for Mobile Systems

Isaac Ahlgren
Loyola University Chicago, iahlgren@luc.edu

Victor Rakotondranoro
Loyola University Chicago, vrakotondranoro@luc.edu

Yasin N. Silva
Loyola University Chicago, ysilva1@luc.edu

Eric Chan-Tin
Loyola University Chicago, chantin@cs.luc.edu

George K. Thiruvathukal
Loyola University Chicago, gkt@cs.luc.edu

See next page for additional authors

Poster: Userland Containers for Mobile Systems

Isaac Ahlgren, Victor Rakotondranoro, Yasin N. Silva, Eric Chan-Tin, George K. Thiruvathukal, Neil Klingensmith
{iahlgren@,vrakotondranoro@,ysilva1@,chantin@cs.,gkt@cs.,neil@cs.}luc.edu
Loyola University Chicago
Chicago, Illinois, USA

EXTENDED ABSTRACT

Mobile platforms are not rising to their potential as ubiquitous computers, in large part because of the constraints we impose on their apps in the name of security. Mobile operating systems have long struggled with the challenge of isolating untrusted apps. In pursuit of a secure runtime environment, Android and iOS isolate apps inside a gulag of platform-imposed programming languages and runtime libraries, leaving few design decisions to the application developers. These thick layers of custom software eschew app portability and maintainability, as development teams must continually tweak their apps to support modifications to the OS's runtime libraries. Nonstandard and ever-changing interfaces to those APIs invite bugs in the operating system and apps alike.

Mobile-only APIs have bifurcated the population of software running on our devices. On one side sits the conventional PC and server programs: compilers, shells, servers, daemons, and many others that use the standard libraries and programming models to interface with the computer and the outside world. On the other side lives the apps: mobile-only and purpose-built, they often serve as user interfaces to some larger cloud-based system. Under the weight of the numerous OS-imposed platform constraints, it is difficult for app developers to innovate: large classes of applications are simply impossible to port to mobile devices because the required APIs are unsupported. To deal with these crossplatform dependencies, it is necessary to maintain multiple code bases. In the past, dependency issues have typically been solved through the use of containers. However, deploying containers on mobile systems present unique challenges. To maintain security, mobile operating systems do not give users permission to launch Docker containers [2].

To solve this issue, we consider an older idea known as *userland containerization*. Userland containerization allows userland containers to be launched by regular unprivileged users in any Linux or Android based system. Userland containerization works by inserting a modified operating system kernel between the host kernel and the guest processes (see Figure 1).

We have done an in depth study on the peroformance of usermode containers like the user mode linux (UML) kernel [1], repurposing it as a userland hypervisor between the host kernel and the guest processes. We prototype a proof-of-concept usermode kernel with an implementation that is guided by the findings of our empirical study. Our kernel introduces a new technique—similar to paravirtualization—to optimize the syscall interface between the guest process and the usermode kernel to improve its I/O performance. The redesigned syscall interface provides I/O performance that approaches that of conventional virtualization techniques. **Our paravirtualization strategies outperform UML by a factor of 3-6× for I/O bound workloads. Furthermore, we achieve 3.5-5× more network throughput and equal disk write speed compared to VMWare Workstation**. Although there is still ample opportunity for performance improvements, our approach demonstrates the promise and potential of a usable userland virtualization platform that balances security with performance.

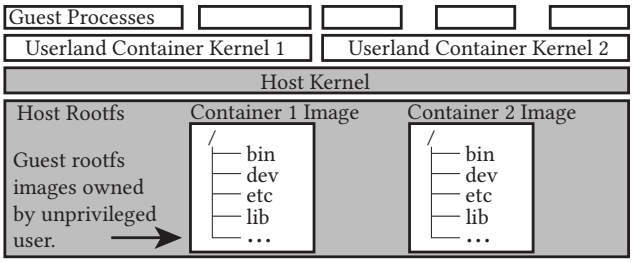

Figure 1: Components of a userland container (privileged shaded gray). All components of the guests, including the guest kernel and rootfs are owned by an unprivileged user.

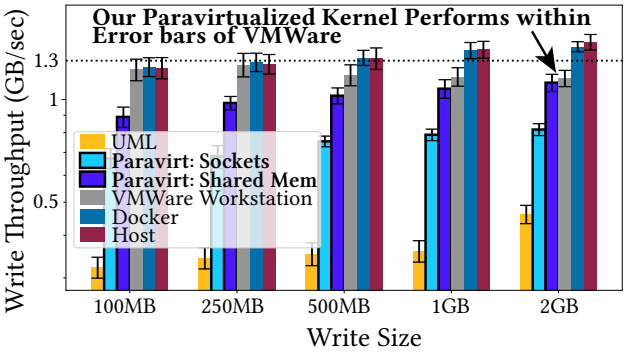

Figure 2: Throughput for each of the virtualization platforms running on desktop. Our paravirtualized userland containers (bars outlined in black) achieve nearly the same performance as VMWare Workstation.

REFERENCES

[1] Jeff Dike. 2001. User-mode linux. In *5th Annual Linux Showcase & Conference (ALS 01)*. USENIX.

[2] Xing Gao, Zhongshu Gu, Zhengfa Li, Hani Jamjoom, and Cong Wang. 2019. Houdini's Escape: Breaking the Resource Rein of Linux Control Groups. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (London, United Kingdom) *(CCS '19)*. Association for Computing Machinery, New York, NY, USA, 1073–1086. https://doi.org/10.1145/3319535.3354227

HotMobile '23, February 22–23, 2023, Newport Beach, CA, USA

ACM ISBN 979-8-4007-0017-0/23/02.
https://doi.org/10.1145/3572864.3581588