MoVirt: More Mobile than Mobile

Will Dietz, Kevin Larson, Shivaram Venkataraman
{wdietz2, klarson5, venkata4}@illinois.edu
University of Illinois at Urbana Champaign

1 Introduction

Modern mobile devices run third party applications to perform complex tasks like web browsing, banking and gaming. Recent studies have found that smart-phones are the target of an increasing number of malware attacks [13, 3, 4] and their security is important as personal data such as contacts, credit card numbers and passwords are often stored on the device. While some security models [2] provide a stronger process level isolation among applications, operating system bugs such as [7, 6, 8] allow malicious applications to take over the device. Virtualization can be useful for secure isolation of third party code from confidential data and provide greater defense-in-depth against attacks on the system.

In recent years, virtual machines have become prevalent in cluster computing environments [1] as they provide isolation for shared usage of machines in a data center. As a result of hardware improvements, smart phone configurations found today resemble desktop machines from few years ago and many of them run commodity operating systems. There is a growing interest in academia [15] and industry [5] about the benefits of virtualization on these devices. Virtualization provides better security guarantees in mobile devices than current solutions offer as well enabling useful applications like environment migration.

Migrating a system to a mobile device can take advantage of network or computation facilities that are closer to the user's location and provide the user with a consistent experience irrespective of the network connectivity. Environment migration has been studied earlier, in the context of servers in a cluster [14] and enables administrators of clusters to perform maintenance tasks without interruption. On the other hand, migration techniques on mobile devices can help maintain consistent snapshots which allow easy transfer of data when users switch mobile phones and to roll-back the system to a previously known state.

1.1 Existing Work

Currently, there are many solutions available for virtualization on desktop environments. VMware is a popular closed source solution which implements a variety of virtualization techniques and is used in both industry and academia. KVM [18], QEMU [12], and XEN [11] are all open source solutions, implemented using a variety of virtualization techniques. These solutions cannot be directly used in mobile environments for performance and usability reasons.

Recently there has been a surge of research in the area of mobile virtualization. One such solution is MobiVMM [20], which prioritizes performance and security at the cost of usability and portability. Work has been done to port KVM to ARM [9], focusing on performance and functionality. All of these solutions either dual-boot the OS or require disabling the phone's existing runtime stack.

VMware's MVP project [19] is most similar to ours. They introduce a very thin Type I hypervisor with an emphasis on usability, performance, and security–without sacrificing the phone's functionality. However their implementation does not integrate with the host OS, but rather replaces and contains it. This is useful, but tangential to our work. Open Kernel Lab's OKL4 [17] is another implementation of a thin Type I hypervisor and is very similar to MVP. As described in Section 1.2, we aim to provide a Type II Virtual Machine (VM) and prioritize live migration capabilities as well as usability. Instead of virtualizing the existing OS to protect other Virtual Machines, we assume it to be trusted and only isolate third party applications. This provides for a very different architecture in our implementation.

There also is ARM's TrustZone [10] which is aimed at creating a secure "TrustZone", primarily for use in DRM, bank transactions and other similar setups. The goal is to protect a specialized app from the rest of the system (and protect, for example, secrets and keys from leaking out of this zone). We aim to to do the opposite: we trust the host OS and are protecting it from the guest applications.

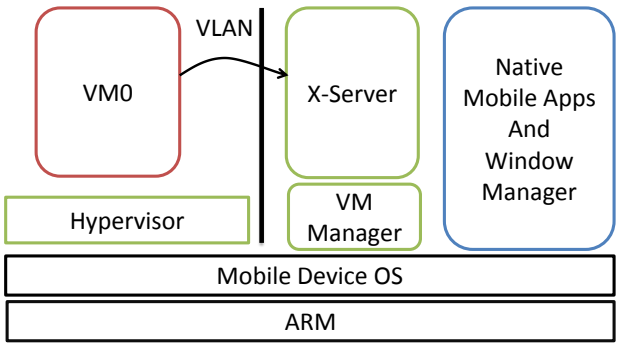

Figure 1: Architecture diagram

1.2 Proposed Architecture

Our proposed architecture has two main components: the virtualization framework, and the integration front-end. The virtualization framework contains the hypervisor (QEMU or KVM based) as well as each of the spawned VMs (see Figure 1). Each VM will contain a thin OS for running the third-party application. We can choose either to run an x86-based Operating System in the VM or an ARM based Operating System. The former will enable a more diverse set of target applications to be run, while the latter would have lesser overhead due to virtualization. We plan to evaluate both of these options. Each of the VMs will headless themselves, but will render their applications by connecting to the native X-server.

The other component is the integration front-end. This contains a light X-server that has been integrated into the host OS, and also contains the VM manager. The VM manager will either run inside the X-server as a native application (ARM-based), or as a separate application within the host OS. The VM manager will be the user interface that controls launching, switching, suspending, resuming, migrating the VMs and ensuring the X-server is running properly. Most of this actual logic and functionality will be implemented by the hypervisor in the virtualization framework.

A shared X-server removes rendering overhead from each VM and reduces the complexity in composing the UI. We choose to share the rendering state between third-party applications for performance reasons and simplicity of design, but ensure isolation from native applications. Additionally the X-server will be running native code, which will allow for it to be less CPU intensive than running inside a VM.

We aim to make the virtualization framework device independent, which is important because we anticipate this to be the more complex component. The integration front-end will have to be ported for each new device we support, but we will strive to make its implementation simple. Our first implementation will focus on one particular device and OS and support live migration of applications from desktops.

In summary, we propose to leverage existing hypervisors to build a secure, usable and portable framework for mobile device virtualization. Our contributions will be focused on providing the ability for live migration of applications with deep integration to the mobile device. Our proposed security model is an effective method to isolate applications and we find that our design ideas are similar to other recent efforts [16] in isolating untrusted code.

2 Timeline

2.1 Feb 17 - March 17

- Explore architecture details of ARM and finalize the design details for the hypervisor.
- Evaluate the overhead of x86 vs ARM virtualization.
- Run a thin hypervisor on ARM using the mobile device toolchain.
- Compile and run X Window System on the mobile device.

2.2 March 17 - April 27

- Integrate X Window System with Window Manager of the mobile device.
- Implement VM Manager to spawn the virtual machines and X-Server.
- Implement migration UI on desktop and mobile device.
- Migrate existing applications between desktop and mobile device.
- Investigate performance improvements to make emulation faster.

2.3 April 27 - May 11

- Document appropriately and write up the final report.

3 Evaluation plan

We evaluate our system focusing on our design goals of portability, isolation and usability

- Running existing unmodified binaries on the mobile device.
- Verify correctness of the implementation using standard benchmarks.

- Measure performance impact of emulation on ARM.
- Evaluate the latency of an interactive application which renders using the native X-server.
- Profile the memory and power overhead due to virtualization.

4 Anticipated results

- Working implementation of a Hypervisor on ARM.
- X-server running natively on a mobile device.
- Implementation of a VM Manager integrated with the mobile device window manager.
- Ability to migrate a binary from desktop to mobile device

References

[1] 16 percent of workloads are running Virtual Machines. http://www.gartner.com/it/page.jsp?id=1211813.

[2] Android Security and Permissions. http://developer.android.com/guide/topics/security/security.html.

[3] Cyber-criminals target mobile banking. http://www.v3.co.uk/vnunet/news/2173161/cyber-criminals-target-mobile.

[4] iPhone Privacy. http://seriot.ch/resources/talks_papers/iPhonePrivacy.pdf.

[5] Mobile Phones, The Next Frontier. http://blogs.vmware.com/console/2009/08/mobile-phones-the-next-frontier.html.

[6] Vulnerability Summary for CVE-2009-0475. http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-0475.

[7] Vulnerability Summary for CVE-2009-2204. http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-2204.

[8] Vulnerability Summary for CVE-2009-2692. http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-2692.

[9] D. A. Andreas Nilsson, Christoffer Dall. Android Virtualization. http://www.chazy.dk/android-report.pdf, 2009.

[10] ARM Ltd. ARM - TrustZone. http://www.arm.com/products/processors/technologies/trustzone.php.

[11] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization, 2003.

[12] F. Bellard. QEMU, a fast and portable dynamic translator. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 41–41, Berkeley, CA, USA, 2005. USENIX Association.

[13] A. Bose and K. Shin. On Mobile Viruses Exploiting Messaging and Bluetooth Services. *Securecomm and Workshops, 2006*, pages 1–10, 2006.

[14] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, page 286. USENIX Association, 2005.

[15] L. Cox and P. Chen. Pocket Hypervisors: Opportunities and Challenges. In *Eighth IEEE Workshop on Mobile Computing Systems and Applications, 2007. HotMobile 2007*, pages 46–50, 2007.

[16] C. Grier, S. Tang, and S. King. Secure web browsing with the OP web browser. In *Proceedings of the 2008 IEEE Symposium on Securiy and Privacy*, 2008.

[17] O. K. Labs. OKL4 Microvisor. http://www.ok-labs.com/products/okl4-microvisor.

[18] Qumranet. Kernel-Based Virtual Machine. [Online], 2009. Available: http://linux-kvm.org.

[19] VMware. VMware MVP (Mobile Virtualization Platform). http://www.vmware.com/products/mobile.

[20] S. Yoo, Y. Liu, C.-H. Hong, C. Yoo, and Y. Zhang. MobiVMM: a virtual machine monitor for mobile phones. In *MobiVirt '08: Proceedings of the First Workshop on Virtualization in Mobile Computing*, pages 1–5, New York, NY, USA, 2008. ACM.