Controlling a Smartphone from an Architectural Vantage Point

Brief Summary

This talk introduces VERTIGO: a modular custom thin-microvisor for the ARM architecture. It is installed through an Operating System (OS) specific loader that dynamically suspends execution, decouples the OS from its underlying hardware and hoists it into a state analogous to a virtual machine. VERTIGO is considered a virtualization technology but is unique as it does not require *any* source code modifications of the underlying OS to maintain control and synchronization, unlike OKL4 and CODEZERO. The microvisor targets the Cortex -A8 and -A9 series SoCs and has been tested with the Apple iPhone 4 (iOS 5.1.1) and Samsung Galaxy SIII (Android 4.0.4).

The VERTIGO kernel leverages only architectural features, which allows it to maintain broad compatibility with Cortex–A series processors and their supported OSs. It is constructed as position independent flat binary to support a variety of loading methods, including: kernel exploits, privileged user space applications and kernel modules. As an example, the loader for Apple iOS is a user-space application that injects VERTIGO after it gains access to privileged memory and execution in supervisor mode. The microvisor kernel does not link to any API contained in the underlying OS, thus providing clean separation from it. Numerous modules have been constructed to extend functionality and include: Light Weight Shadow Paging (trapping on data), Breakpoint (trapping on underlying OS kernel- and user- space code), Linux Introspection (links to underlying OS API), TrustZone Communication (SCM calls), etc. As modules are meant to extend VERTIGO a set of unprivileged user-space applications are used for management and include *insmod*, *rmmod,* and *lsmod*.

The VERTIGO microvisor represents a unique capability that showcases what is possible at the architectural level. While its primary purpose is to aid in reverse engineering and other security related research tasks, the techniques could be misused to assist with nefarious activities. The benefits of exposing this capability will hopefully be twofold. First, advance the *state-of-the-art* in tools available when performing reverse engineering and other security related research tasks. Second, allow mitigation technologies to be designed and developed to prevent malicious software from leveraging the same techniques.

Detailed Outline

The talk will start with a brief discussion of architectural facilities in ARM Cortex-A series CPUs. After that the construction process for a position independent flat binary will be examined. This will involve a look at custom linker scripts, which are used to construct the flat binary, and assembly code, that allows it to become position independent.

Next will be an overview of the VERTIGO microvisor and then a deep technical dive into its key enabling components. This will include everything from concepts to source code snippets of:

- Architectural brute force approach to address translation
- Paging system setup and transitions
- Vector table hooking, including stack considerations
- Memory management, including: allocation, mapping and unmapping
- Dynamic module loading and linking

After that, the loader applications will be presented. First the Apple iOS variant and how it is able to gain access to the kernel from user space and inject the microvisor. Second the Google Android version and its specific requirements to launch the microvisor.

Module management applications will be examined and will include how it is possible to *install, remove* and *list* modules in the microvisor transparently from an unprivileged user-space application.

Next is an overview of the different modules that have been constructed and details of how introspection and break pointing can used to inject an arbitrary user-space application into an underlying OS.

Finally, debugging techniques will be presented for both the Apple iPhone 4 and Samsung Galaxy SIII. This will include finding/constructing the hardware connectors that expose a UART and developing freestanding libraries that the microvisor can use for communication.

This session will include exposure to advanced features of GNU *gcc* and *ld* to construct a position independent flat binary (shellcode), assembly code and low level systems programming concepts and techniques for the ARM architecture.

Presenter Background

Kirk Swidowski is a Security Researcher, with a Master's degree in Computer Science. His expertise includes: Virtualization (x86 [Intel-VT/AMD- V] and ARM), Trusted Computing Technologies (Intel TXT, ARM TrustZone and TPM), Boot Technologies and Computer Architecture.

Prior work involves the design and construction of multiple custom thin-hyper- / thin-micro- visors for ARM and x86 which provide the foundation for advanced dynamic analysis of hardware peripherals, user/kernel software and the creation of new security capabilities. Hardware experience includes: FPGA development, hot-air reflow and soldering of surface mount ICs and the construction of custom tools, such as SPI/I2C flash chip programmers. He led research which resulted in the discovery of a new AES flow interception attack that undermines the Intel AES-NI instruction set extension. He has also designed and developed multiple

commercialized mobile applications, one of which was published in "*A Windows Mobile Wish List*", Smartphone & Pocket PC Magazine.

Acronym

VERTIGO - Virtualized Extendable Runtime Transport *for* Integrated Germane Operations