SAMPLE SUBMISSION

Title

Battle of the SKM and IUM: How Windows 10 Rewrites OS Architecture

Alex Ionescu, Chief Architect, CrowdStrike

(Black Hat USA 2015)

Track

OS - Host and Container Security

Abstract

Notes:

- *Detailed, yet concise abstract*
- *Defines a problem and offers a solution(s) that will be examined during session*

In Windows 10, Microsoft is introducing a radical new concept to the underlying OS architecture, and likely the biggest change to the NT design since the decision to move the GUI in kernel-mode.

In this new model, the Viridian Hypervisor Kernel now becomes a core part of the operating system and implements Virtual Secure Machines (VSMs) by loading a true microkernel - a compact (200kb) NT look-alike with its own drivers called the Secure Kernel Mode (SKM) environment, which then uses the Hypervisor to hook and intercept execution of the true NT kernel. This creates a new paradigm where the NT Kernel, executing in Ring 0, now runs below the Secure Kernel, at Ring ~0 (called Virtual Trust Level 1).

But it doesn't stop there - as the Ring 0 NT kernel now has the ability to not only create standard Ring 3 user-mode applications, but also Ring ~3 applications (or Virtual Trust Level 0) that run in Isolated User Mode (IUM). Because VTLs are all more privileged than Ring 0, this now creates a model where a user-mode application running inside a VSM now has data and rights that even the kernel itself cannot modify.

Why go through all this trouble? Because it seems like the hottest thing these days is Pass-the-Hash, and attacks must seemingly be mitigated at all costs. And even in Windows 8.1, an attacker with the permissions to load a kernel driver can bypass the existing mitigations (and Mimikatz is signed!). With VTLs, now even the most privileged attacker is only as privileged as the hypervisor will allow it - never able to truly read the hash date that is stored in the secure partition.

How "secure" is this new model really? And what prevents a malicious application from running in such a secure mode to begin with?

Presentation Outline

Notes:

- *Clearly conveys progression of talk*
- *Helps the Review Board visualize the presentation in its entirety*
- *Gives an explanation of each area of the presentation*

1. Introduction to key terms (VTL, VSM, SKM, IUM, etc...)
- This will detail the basic terminology

2. SKM Boot Architecture and Hypervisor Support
- This will detail how the secure kernel is started up, how to configure a system for SKM support, and the key new Hyper-V 4.0 hypercalls that are used to initialize SKM. We will then go over the SKM boot process and its initialization, including discussion of the IDK (Identification Key) and LK (Local Key)

3. SKM to Insecure Kernel Communication
-Once SKM is initialized, this will discuss how it interacts with the vanilla NT kernel and the interfaces that are created in between, managed by the hypervisor. We will describe all the SKM calls that exist from SKM to NT kernel.

4. IUM Initialization
-Next, this will go over how the Isolated User Mode environment starts itself up, and how to create a "secure process" or a "trustlet". We'll also talk about Vmsp.exe (Virtual Machine Secure Process) worker host.

5. IUM to Insecure Kernel Communication
-Here, we'll detail how a trustlet performs system calls and how its environment is sandboxed and allows communication to the real vanilla kernel.

6. IUM to Secure Kernel Communication
-Continuing on the previous topic, this will explain how a Trustlet actually talks to SKM, and how it can obtain the key secure data that the SKM is protecting. We will also go over all the Secure System Calls that are implemented by the SKM.

7. Live Demo of IUM "Trustlet"

-We will take a look at LSASS, implemented as a Trustlet, as well as a custom Trustlet I've written to demonstrate some of the protections afforded and how the isolation works. We will also look at how a potentially malicious Trustlet could attack the system.

8. Closing Remarks
-Finally, this section will show potential avenues for abuse of the system, as well as some thoughts on the practicality of the implementation of this system in Windows 10.

Attendee Takeaways

Notes:

- *Submitter fulfilled requirement of providing 3 takeaways*
- *Explains relevance to the audience*
- *Cleary emphasizes the participant benefits*

1. First, that the security model in Windows 10 is radically changing -- obtaining root/kernel privileges on the machine only gives you Ring 0 rights to the virtualized Hyper-V partition, and not the entire system. There is a new, extra, security boundary that must be broken for full access.

2. Second, how this relates to PtH attacks today, and how future attacks might look like.

3. Third, how this can be enabled and what its requirements are - as well as its weaknesses.

And finally, a good dose of Windows Internals, as always :)

Why Black Hat?

Notes:

- *Summarizes the scale of the issue and its potential impact*
- *Argues relevance/importance of the presentation at Black Hat*

This is a potentially huge change to the most widely used Operating System with far-reaching consequences to OS security, virtualization, safe cryptographic storage, and more. The idea that even the host OS itself is now virtualized becoming mainstream, having started from the first "Blue Pill" esoteric attack to finally being the norm in Windows 10. Black Hat is the perfect platform to make everyone aware of this drastic new model.
