Tu Lam

CS 373 (Defense Against the Dark Arts)

Dr. Bram Lewis

October 27th, 2021

Homework #5 Write-Up

Week #5 of the class have a special guest; his name was Aditya Kapoor. He works for McAfee as a Research Architect @McAfee Labs. He mostly deals with reversing work on metamorphic virus on any system. Through this week lecture, we will be dealing with the ***windows internals***, ***operating system structures***, how ***threading work***, and ***processor work***. On top of that, we will be looking at some ***rootkit*** during the lab portion.

First, we dive into recapping from week 1-4 through the major concept that we explore throughout those four weeks. This comes from learning the basic of malware, using YARA tools to help identify malware, looking at hex number, learn various stage of attack, learn about exploit, and more. Then, Aditya went into introducing that the point of this week is to do a little stealth into the manipulation of the system that we are trying to exploit. This is where rootkit will come in. Rootkit was on a rise between the year of 2011-2012 where it was attacking many systems such as Android, Linux, etc… Through this, we would need to learn what rootkit is to know how they operate. Based on Aditya's definition, ***rootkit*** is defined as "Malware that actively conceals its existence and actions from users and system processes". They found out that rootkit is use in 10% of the malware attack and most the system that this attack is happen is on a 32-bit windows and sometimes on couple of the 64-bit windows. Using rootkit is a good way to learn how it exploit the malware into the kernel security.

Rootkit can easily enter the 32-bit OS, but with the 64-bit is a little harder. There couple of ways that rootkit can achieve this through entering into the kernel. They can ***bypass driver signing check***, ***modifying the windows boot path***, ***kernel exploits in Windows kernel or third-party drivers***, and ***stealing valid digisigs***. Below is a picture layout of how the user vs. kernel level look like in an operating system structure and where rootkit can be entered. Through this, we can track and see what the process is like for the malware to enter from a ring 3 level (user) into the ring 0 level (kernel) that should be not access. But we can look at the image and get a sense of what the OS structure also have in it.

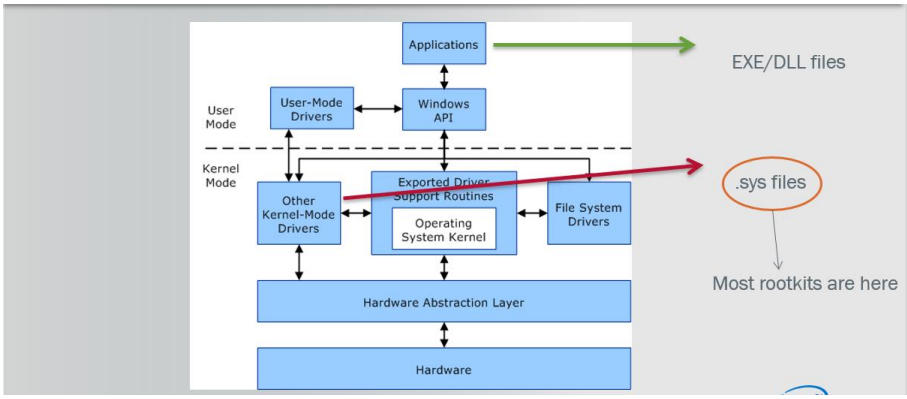


Figure #1: [**From Lecture**] The user vs. kernel level of OS

From here, we then talk about the kernel memory inside the ***OS system***. ***Kernel memory*** is defined as "a flat memory model with no security separation". This means that any kernel driver can access any part of memory inside the kernel, and this is a prime thing that give rootkit a way into and attack the system through structures like SSDT, IRP, IDT, and more.

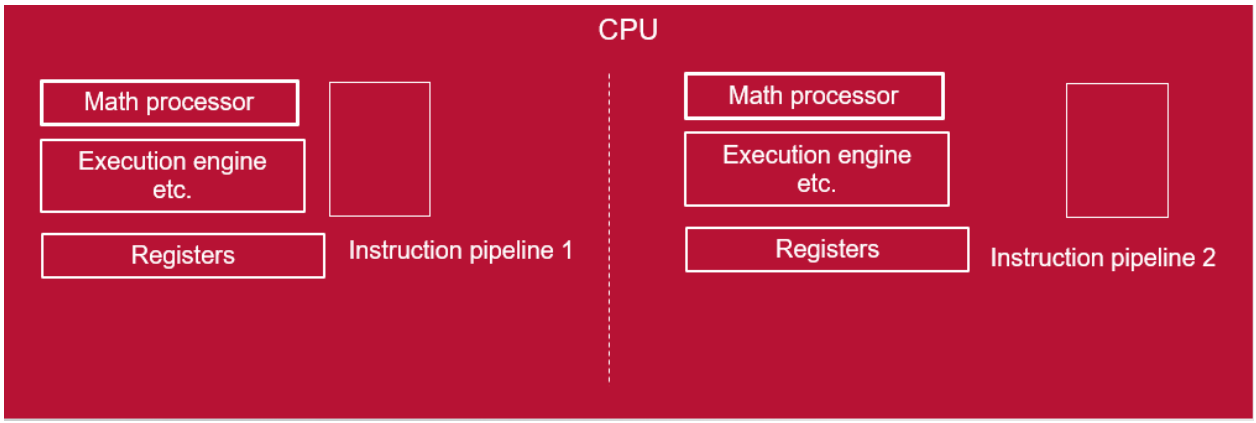


Figure #2: [**From Lecture**] The process of threading

Next, we discuss the ***threading*** process where we could run multiple code/process at the same time in the OS system. Thread is the smallest 'unit' of execution that can execute code and it can have object which defined the thread. Thread is then having context stores all the related register values of the thread and they can have their own stack. In there, they can use thread's function to call local functions and variables, control the data going in and out, and copied and transfer thing for security. The whole threading is then tie in with ***process memory*** where the process includes an object table that has handles to other objects known to this process, and will need one or more threads to execute and run. Below will give you a layout of what the stack looks like for the thread and how each layer is being use.

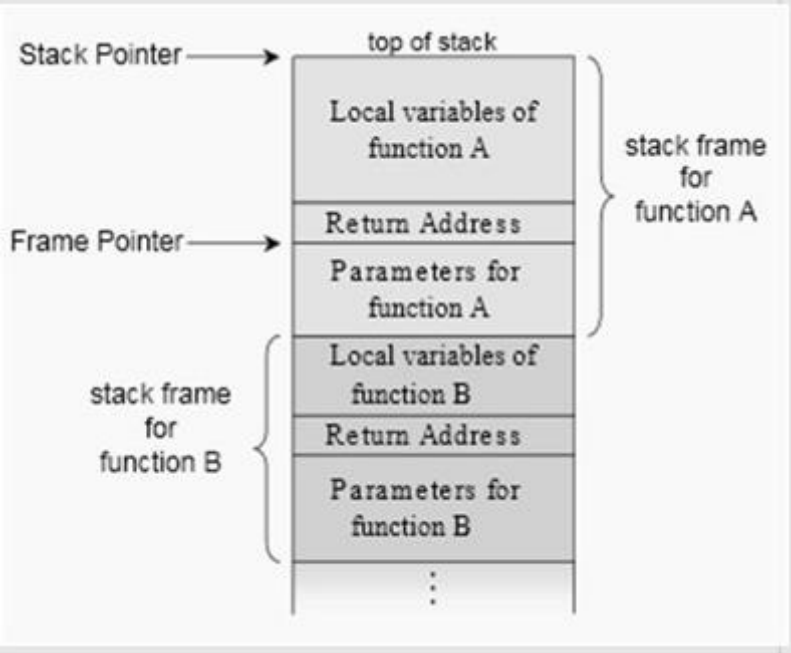


Figure #3: [**From Lecture**] Thread stack

Then, we dive into a little bit of the ***hooking process*** and how to attach it to the OS system. Below will be two images that describe these processes of hooking and detection.

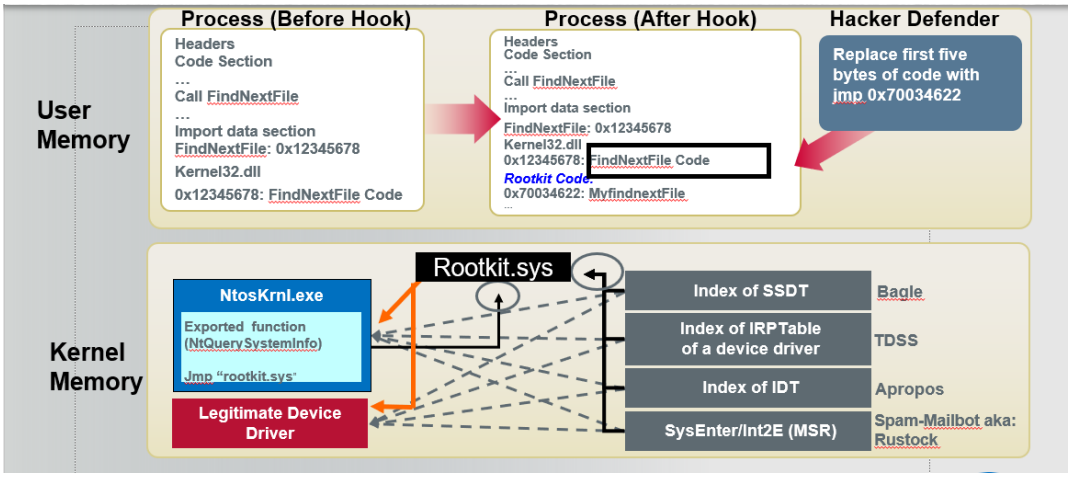
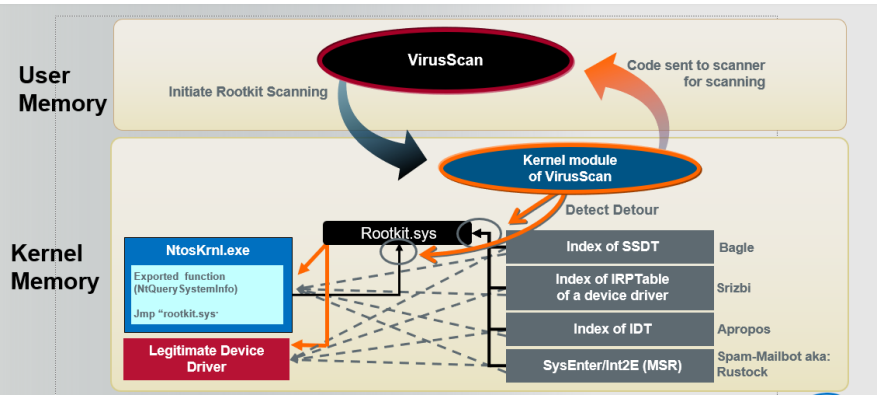


Figure #4: [**From Lecture**] Hooking process & *Figure #5*: [**Below**] Detection Strategy



Finally, we then move onto the ***stealth technique*** that hacker has been using to infiltrate the system OS. These includes inline hooks, import table hooks, DKOM, IRP hook, SSDT hook, and more. There are many ways that hacker use to exploit the system and using rootkit as a fundamental way to access these OS without being detected. Below is a timeline of different technique uses through the year and giving how they went undetected from user. Lastly, we touch a little bit on booting OS where we learn about ***pre-boot***. This is where "power on self-test to diagnose memory, hardware components. It loads the bios that locates the boot device and then loads and runs MBR".

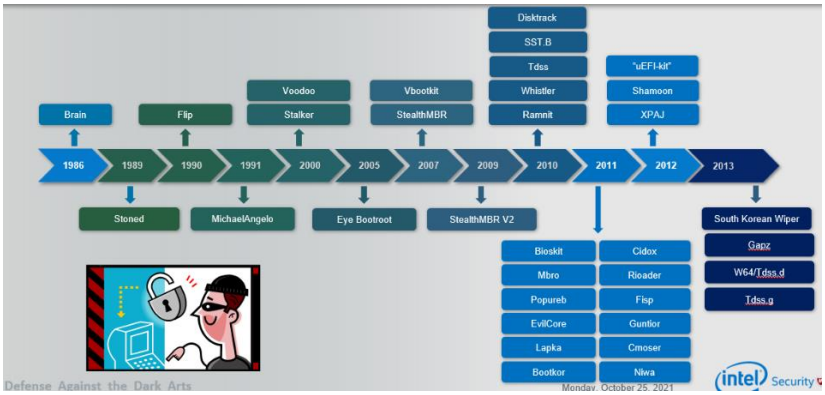


Figure #6: [**From Lecture**] Timeline of different stealth technique

Through this week, we were able to dive into deeper details about windows internals like the OS system and revisit some of the concept that we learn in the past that was helpful to know when dealing with the internal system. Going through rootkit and explore that malware can go undetected when entering into the OS system is crazy and really genius when it comes to it stealth. Overall, be able to identify and found it is a good skill to have and through this week lecture, it was a great introduction to it and hope to inspire other to look into this topic as we move on in the world of cybersecurity.

References

1. Kapoor, A. (2015). WINDOWS MEMORY MANIPULATION. Canvas. Retrieved October 27, 2021, from https://canvas.oregonstate.edu/courses/1877198/pages/module-5-learning-materials?module_item_id=21556809.