7.2 Privileges

To protect critical OS components, data, and processes, there are two privilege modes the processor can run under. These two modes are **user-mode** and **kernel-mode**. User code runs in user-mode, and OS code (this includes drivers) runs in kernel-mode.

> You may have heard of four different privilege levels described as rings (numbered 0 to 3). User-mode is privilege level/ring 3 and kernel-mode is privilege level/ring 0. Rings 1 and 2 are not used by Windows.

Kernel-Mode

Kernel-mode allows the processor to essentially do whatever it wants by allowing access to any code on the system. User-mode processes can only access the kernel through strictly defined interfaces. Kernel mode is a necessity in an OS to keep user-mode processes in check. Kernel-mode processes share the same virtual address space. Pages within the kernel-mode virtual address space are only accessible from kernel-mode. User-mode pages are accessible from both kernel-mode and user-mode.

User-Mode to Kernel-Mode

Sometimes a user-mode process needs to access kernel-mode functionality. This happens very often such as when rendering windows or graphics. When a user-mode process calls a system service, special instructions are executed that switch the thread to kernel-mode. Once the service finishes, the thread is switched back to user-mode.

User-mode code runs in the memory range of 0x00000000 to 0x7FFFFFFF. Kernel-mode code runs in the memory range of 0x80000000 to 0xFFFFFFFF.

Hypervisor

With the boom of virtualization, there was a need for a way to run high-performance OS guests efficiently. To facilitate this virtualization, hypervisors are used. **Hypervisors** allow for the separation and isolation of all system components including virtual memory, physical memory, USB devices, and more. Hypervisors have more privileges and abilities than kernel-mode applications due to their ability to virtualize and isolate components. Because of this, Windows uses the hypervisor for security, this is known as *virtualization-based security* (VBS). Some of the components in VBS are the Hyper Guard, Credential Guard, Application Guard, Host Guardian, Shielded Fabric, and more. I won't explain all of these components, but I will briefly explain two of them.

- **Hyper Guard** - Protects important kernel and hypervisor related data structures and code.
- **Application Guard** - Stronger sandbox for Microsoft Edge.

The hypervisor also implements **Virtual Trust Levels (VTLs)**.

> VTLs are ordered differently than processor privilege levels (rings). VTL 0 has less privileges than VTL 1.

The OS runs in VTL 0, and VBS runs in VTL 1. This gives VBS more privilege than kernel-mode and therefore cannot be touched by kernel-mode. You can think of user-mode and kernel-mode as running within VTLs, and

the hypervisor managing the permissions for all VTLs.

Hypervisors are extremely complex and there's plenty to learn about them. The explanation I've provided is very brief and basic. Still, this is more than you will need to know when it comes to reversing, so if you didn't understand all of it, don't worry. I do encourage you to learn more about hypervisors because they are very interesting and vital to security.

> Although hypervisors have more privileges than kernel-mode, hypervisors do *not* run in ring -1.

<- Previous Lesson
Next Lesson ->

Chapter Home