Incident Response - More on the Windows PEB

Taz Wake | Halkyn Consulting Ltd | Originally published 2023-11-28

Deciphering the Process Environment Block's Architecture

The Process Environment Block (PEB), an essential structure within the Windows operating system, provides a rich array of information critical for the execution of a process. Delving into the PEB's architecture illuminates the complex mechanisms at play behind the scenes of every running Windows process.

The PEB's Central Role in Process Management

Upon launch, each process in Windows is allocated a PEB, which resides in the user space, making it accessible for read and write operations by the process it belongs to. This structure is pivotal in how Windows manages the high-level details concerning the process's environment.

Composition of the PEB

The PEB is composed of a series of fields that hold specific data types and pointers, each with a distinct purpose. For instance, the **ProcessHeap** field points to the starting address of the process heap, where dynamic memory allocation occurs. This heap is the process's go-to resource for memory requests during its lifetime.

ProcessParameters: The Gateway to Process Configurations

The **ProcessParameters** field is particularly noteworthy, as it is a pointer to a **UNICODE_STRING** structure that holds the command line arguments passed to the process. This is a treasure trove of information for understanding the context in which the process was initiated.

The Ldr Field: A Link to the PEB Loader Data

The **Ldr** field is another critical pointer within the PEB. It connects to the PEB Loader Data (**PEB_LDR_DATA**), a structure that details the modules (DLLs and executables) loaded by the process. This includes a thorough list of each module's base address, entry point, size, and the full path to the module file.

Navigating the PEB for Forensic Analysis

In the realm of forensic analysis, understanding the layout and access points within the PEB can reveal insights into a process's interactions with the system. For example, the **Ldr** field's **InLoadOrderModuleList** can be parsed to uncover the sequence in which modules were loaded, providing clues to a process's behaviour or potential tampering.

Module Analysis via InLoadOrderModuleList

Digging deeper into the PEB, the **InLoadOrderModuleList** is a particularly revealing field. It is a doubly linked list situated within the **PEB_LDR_DATA**, meticulously detailing the modules loaded by the process. Each node in this list, represented by a **LDR_DATA_TABLE_ENTRY** structure, is a compendium of data pertaining to individual modules. Notably, it includes memory allocation specifics such as base addresses and entry points, alongside the module's dimensions and its designated file path. Forensic analysts often scrutinise the **InLoadOrderModuleList** to ascertain the chronological order of module loading, which is a crucial aspect of the process's lifecycle and a potential indicator of unauthorized modifications or exploitative actions.

Beyond InLoadOrderModuleList: Expanding the Forensic Horizon

As well as the **InLoadOrderModuleList**, the PEB's scope encompasses two additional linked lists that serve as vanguards of forensic scrutiny: the **InMemoryOrderModuleList** and the **InInitializationOrderModuleList**. The former is sorted by the memory addresses at which modules reside, thus enabling a more spatially oriented analysis of the process's memory footprint. The latter, arranged according to the initialization sequence of the modules, sheds light on the process's start-up chronology. These lists allow forensic experts to triangulate the integrity of the process's loading and initialization phases. Such detailed examination is instrumental in detecting sophisticated malware, which may clandestinely alter the normal loading sequence to covertly establish its presence within the system. Each list provides a different perspective, together forming a three-dimensional view of a process’s module management—a testament to the depth of insights that the PEB offers for forensic exploration.

Incident Responder's Note: A good way to discover code injection is to review all three lists simultaneously. Injected code can often stand out because it was never initialised or loaded by the process!

The PEB's Relationship with Memory Management

The PEB’s structure is closely tied to the system's memory management, with fields such as **ProcessHeaps** and **NumberOfHeaps** offering details on the arrays of heaps managed by the process. These fields allow for an in-depth look at memory usage patterns, which can be indicative of the process's nature and functionality.

Utilising the PEB in Security Applications

For security applications, the PEB is indispensable. The **BeingDebugged** and **HeapFlags** fields, for instance, can indicate whether a process is being debugged or if certain heap features are activated, which can be signs of debugging or reverse-engineering attempts.

The PEB in the Broader Context of Windows Internals

The PEB's structure and its fields are defined in the **ntddk.h** and **winternl.h** headers, part of the Windows Driver Kit and Windows SDK respectively. This definition underpins the PEB's significance in Windows Internals, serving as a direct interface for numerous system operations.

The PEB: A Dynamic and Evolving Structure

As Windows continues to evolve, so does the PEB. New fields may be added, and existing ones may change, reflecting the operating system's ongoing development. However, the fundamental layout remains consistent across versions, ensuring backward compatibility and a steady framework for developers and analysts alike.

Summary: The PEB's Place in Windows Architecture

The PEB's structure and layout are central to the Windows architecture, providing an organized and accessible way to manage and interact with processes. With each field serving a specific function, the PEB is a microcosm of the process's environment, reflecting its status, configuration, and memory management. For anyone involved in Windows system design, security, or analysis, a deep understanding of the PEB is not just beneficial—it is essential. This article, building upon the foundations laid previously, aims to enhance that understanding and offer a detailed perspective on one of the most integral parts of Windows Internals.

About Halkyn Consulting

Halkyn Consulting Ltd is a UK-based security and risk management consultancy providing 24/7 incident response, digital forensics, threat hunting, virtual security leadership (vCISO) and security compliance support to organisations worldwide. We deliver clear, practical, vendor-independent advice based on real-world threats — no hype, no upsell.

More free security resources are available at www.halkynconsulting.co.uk/security-resources.

Web: www.halkynconsulting.co.uk | **Email:** info@halkynconsulting.co.uk | **Tel:** +44 1244 940858