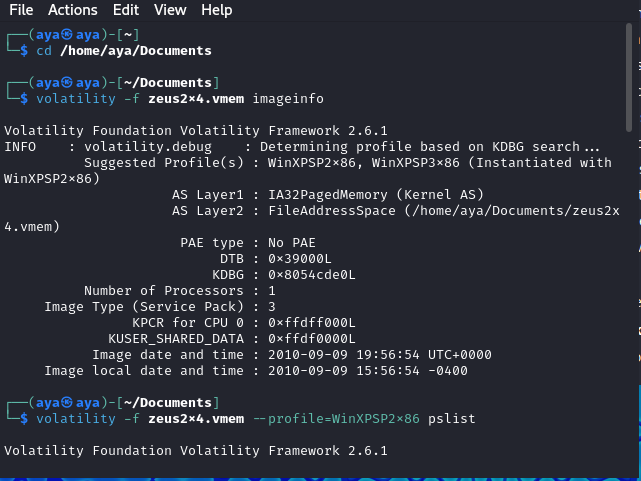
**Name:aya mohamed Abdelrahman**

**Id:20221380245**

**Proactive project**

1-first I install volatility on kali linux

2-i have downloaded the zeus2x4.vmem

3-

The imageinfo command in Volatility is used to **collect metadata** about a memory image file. This metadata helps analysts identify important details about the operating system, configuration, and environment of the captured memory. Specifically, it assists in determining:

1. The **suggested profiles** to use for further analysis.
2. **Memory layers**, which describe how data is stored in the image.
3. Kernel Debugger Block (**KDBG**), an important structure used in Windows kernel debugging.
4. Other metadata like **image date/time**, **DTB (Directory Table Base)**, and **number of processors**.

The imageinfo output provides the following metadata about the memory image zeus2x4.vmem:

**1. Suggested Profiles:**

* The output suggests two profiles: WinXPSP2x86 and WinXPSP3x86.
* These profiles correspond to **Windows XP Service Pack 2 (32-bit)** and **Windows XP Service Pack 3 (32-bit)**.
* Volatility tries to determine the correct profile based on the Kernel Debugger Block (**KDBG**) structure.
* In this case, Volatility has **instantiated the profile** with WinXPSP2x86, meaning this is likely the correct profile to use.

**2. Address Space Layers:**

* **AS Layer1**: IA32PagedMemory (Kernel AS)  
  This indicates that the memory is using a 32-bit **paged memory** structure, typical of Windows XP systems.
* **AS Layer2**: FileAddressSpace (/home/aya/Documents/zeus2x4.vmem)  
  This specifies the location of the memory dump file being analyzed.

**3. Directory Table Base (DTB):**

* **DTB**: 0x39000L  
  The DTB is used by the operating system to manage virtual-to-physical address translation.  
  For analysts, this value helps identify the base location of kernel memory structures.

**4. Kernel Debugger Block (KDBG):**

* **KDBG**: 0x8054cde0L  
  The KDBG structure is a kernel-level debugging structure in Windows memory.  
  It is crucial for locating system processes, threads, and other kernel-related data.

**5. Number of Processors:**

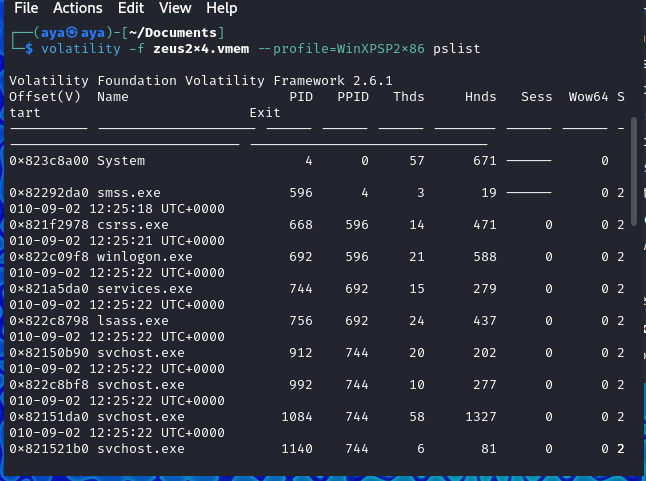
* **Number of Processors**: 1  
  This indicates the system had a **single CPU**.

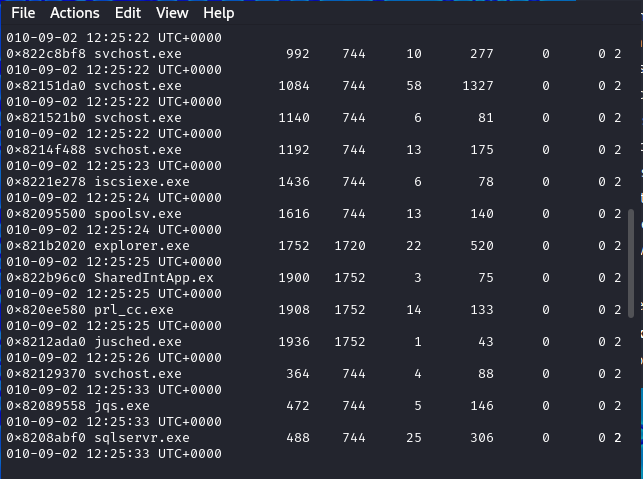
**6. Image Type and Service Pack:**

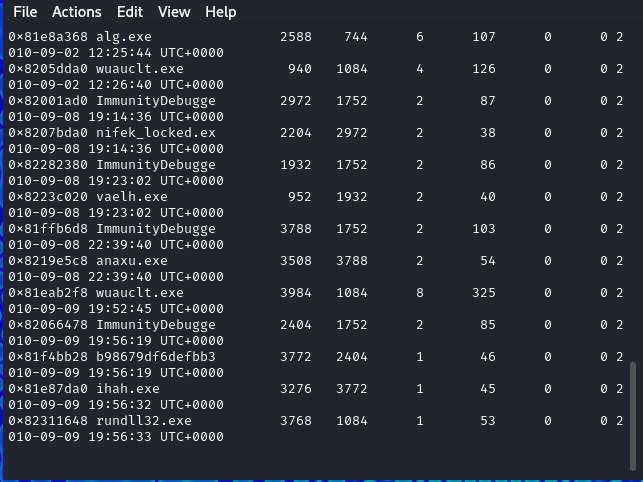
* **Image Type (Service Pack)**: 3  
  This confirms that the operating system was **Windows XP SP3**.

**7. Time Information:**

* **Image date and time**: 2010-09-09 19:56:54 UTC+0000  
  This is the timestamp when the memory dump was taken, expressed in **UTC** time.
* **Image local date and time**: 2010-09-09 15:56:54 -0400  
  This is the local timestamp adjusted for the **Eastern Time Zone (UTC-4)**.

3-





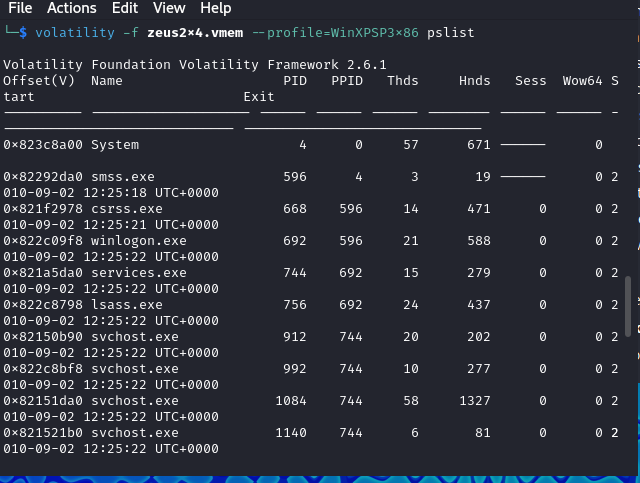
The pslist command in Volatility is used to display a list of **active processes** running on a system when the memory dump was captured. It retrieves the process list by traversing the **EProcess** structures within the Windows memory image. This is useful for identifying normal processes, as well as potentially malicious or injected ones.

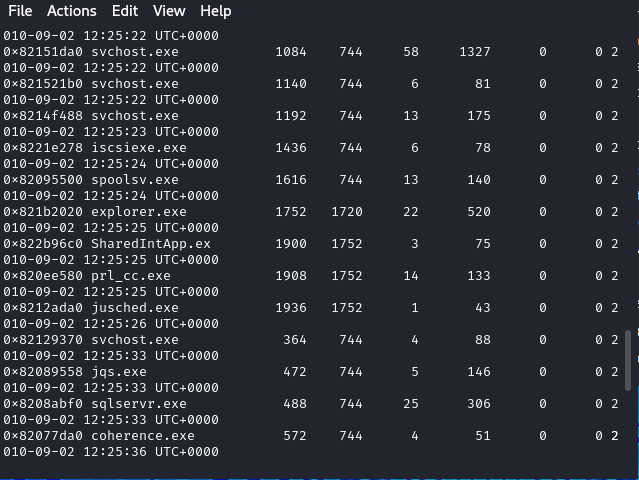
The output consists of several columns, each containing important process-related information:

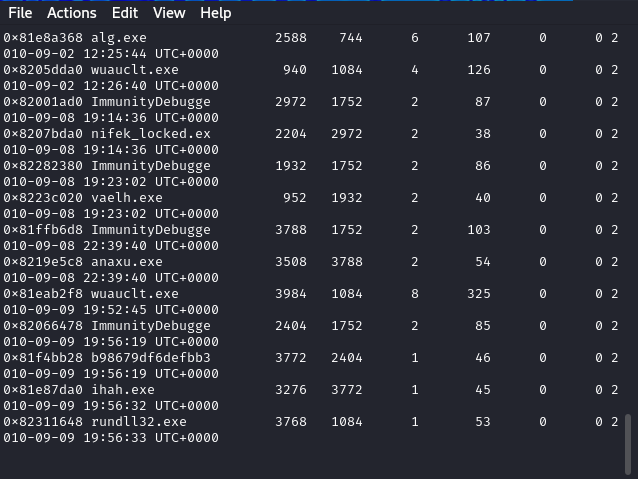
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | **Column** | | --- | --- | |  |  | |  |  | |  |  | |  |  | |  |  | |  |  | |  |  | |  |  | | **Description** |
| **Offset(V)** | The memory address where the process structure (**EProcess**) is located. |
| **Name** | The name of the process (e.g., System, smss.exe, etc.). |
| **PID** | The Process ID |
| **PPID** | The Parent Process ID |
| **Thds** | The number of threads the process is running. |
| **Hnds** | The number of handles (resources like files, registry keys, or sockets) the process is using. |
| **Sess** | The session ID of the process, which can help identify user sessions. |
| **Wow64** | Indicates whether the process is running in 32-bit emulation on a 64-bit system (0 = no). |

**Key Observations in the Output**

1. **System Process (PID = 4):**
   * The System process is the kernel process and always has **PID 4**.
   * It has 57 threads and 671 handles, as expected for a kernel-level process.
2. **smss.exe (Session Manager Subsystem):**
   * **PID = 596**, **PPID = 4** (parent is System).
   * This is the first user-mode process initiated by the Windows kernel during boot.
3. **winlogon.exe and csrss.exe:**
   * winlogon.exe (**PID 668**) manages user login/logout processes.
   * csrss.exe (Client/Server Runtime Subsystem, **PID 608**) is responsible for user-mode side of the Win32 subsystem.
4. **services.exe (PID = 744):**
   * The parent process for system services, spawning service processes like svchost.exe.
5. **svchost.exe:**
   * Several instances of svchost.exe are listed. This process hosts Windows services, and multiple instances are common, each hosting different services.
   * **Parent PID** for svchost.exe is **744** (services.exe).
6. **Handles and Threads:**
   * Processes like svchost.exe have a large number of handles and threads, which is typical as they manage multiple services.
7. **Suspicious Processes:**
   * While no suspicious processes are immediately visible in the list, further analysis is needed to detect injected or hidden processes. Tools like psscan and comparison with pslist output can identify anomalies.

4-

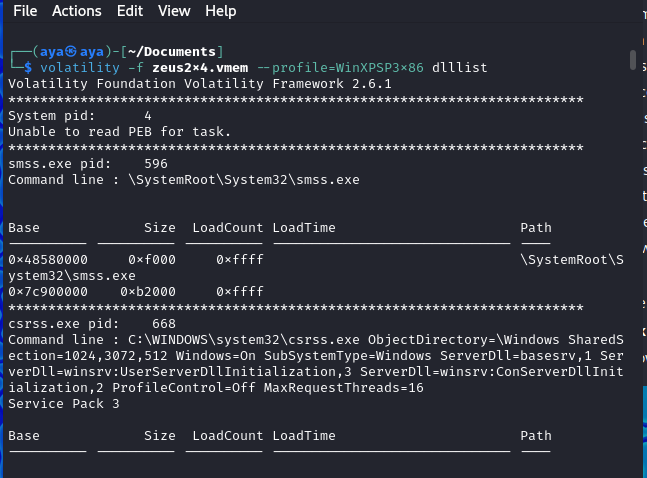


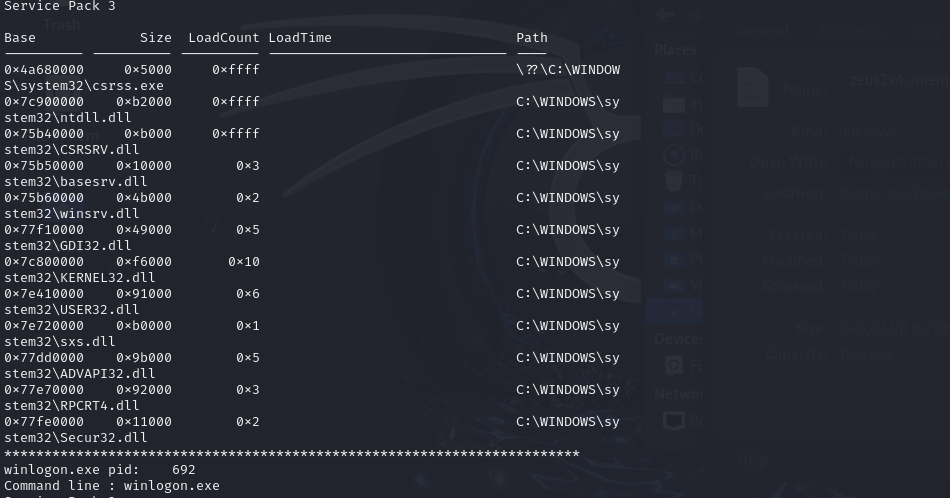


The memory profile explicitly used to analyze the memory dump. Here, the profile matches **Windows XP Service Pack 3 (32-bit)**.

**Purpose and Use of the Command**

1. **Identify Active Processes:**  
   The pslist plugin provides a list of all active processes, which can help to identify **legitimate** processes and potentially **malicious processes**.
2. **Parent-Child Relationship:**  
   By analyzing the **PPID** (Parent Process ID), you can track which processes spawned others. This helps to identify suspicious behavior like malware spawning child processes.
3. **Malware Detection:**
   * Zeus malware may hide itself as a legitimate process like svchost.exe.
   * By analyzing process names, parent-child relationships, and resource usage, you can detect anomalies.
4. **Baseline Behavior:**  
   Comparing this output to a known baseline of active processes on a clean system can help identify unauthorized or unexpected processes.

5-

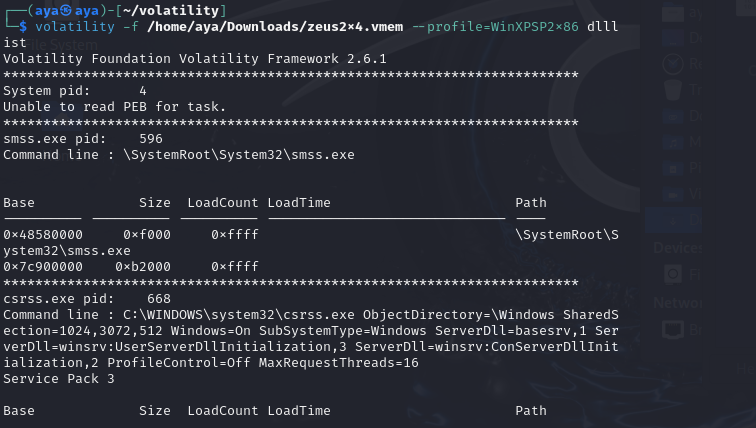


**--profile=WinXPSP3x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP3 running on a 32-bit system.

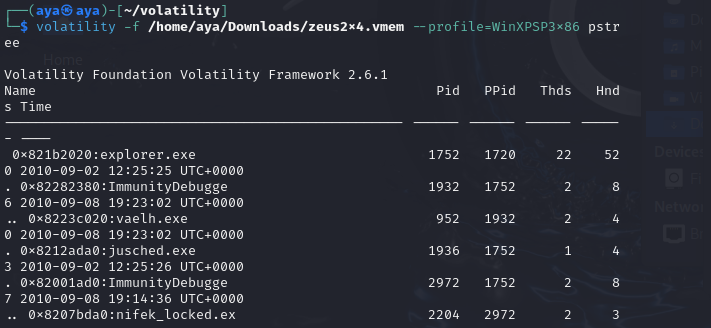
**dlllist:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "dlllist" plugin is designed to list the loaded Dynamic-Link Libraries (DLLs) in the memory dump.

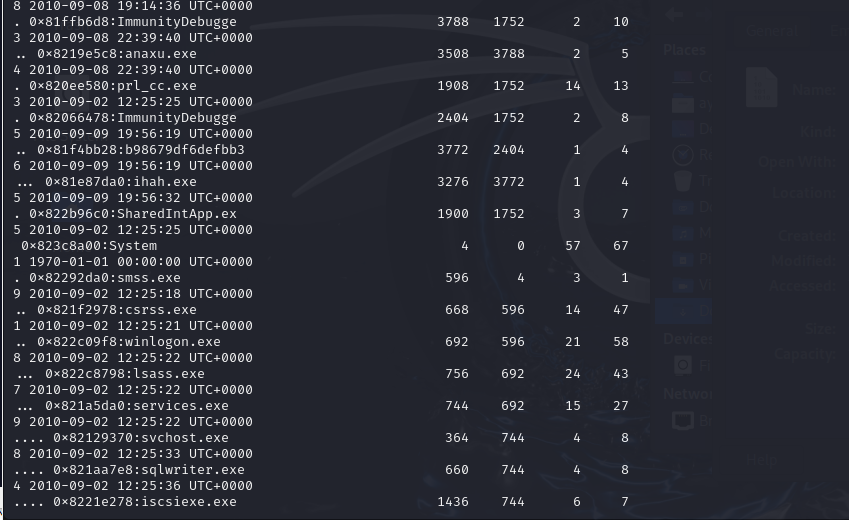
The output provides a list of loaded DLLs in the memory dump. This information can be useful for various memory forensics tasks, such as:

* **Identifying running processes:** The output shows the names and paths of the loaded DLLs, which can help identify the processes that are running on the system.
* **Analyzing malware behavior:** Malware often loads specific DLLs to perform its malicious activities. By analyzing the loaded DLLs, analysts can gain insights into the malware's behavior.
* **Reconstructing system state:** The loaded DLLs can provide information about the system's configuration and state at the time the memory dump was taken.

6- 

**--profile=WinXPSP2x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP2 running on a 32-bit system.

7- 

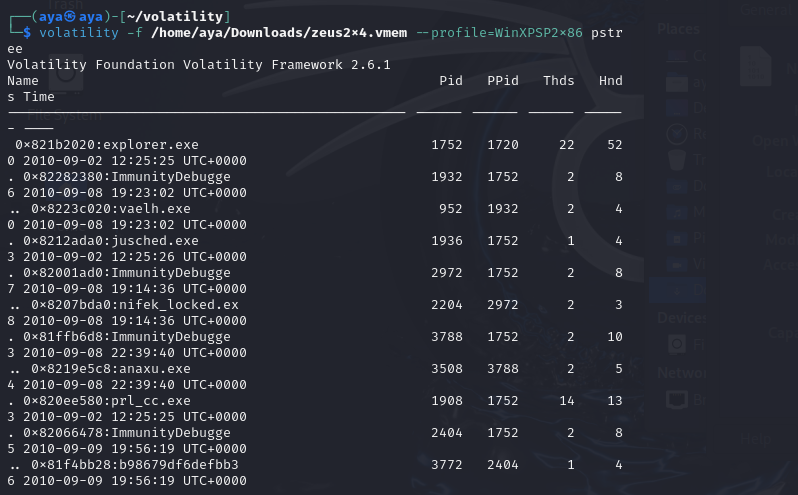


**--profile=WinXPSP3x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP3 running on a 32-bit system.

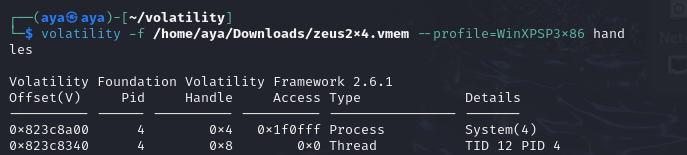
**pslist:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "pslist" plugin is designed to list the running processes in the memory dump.

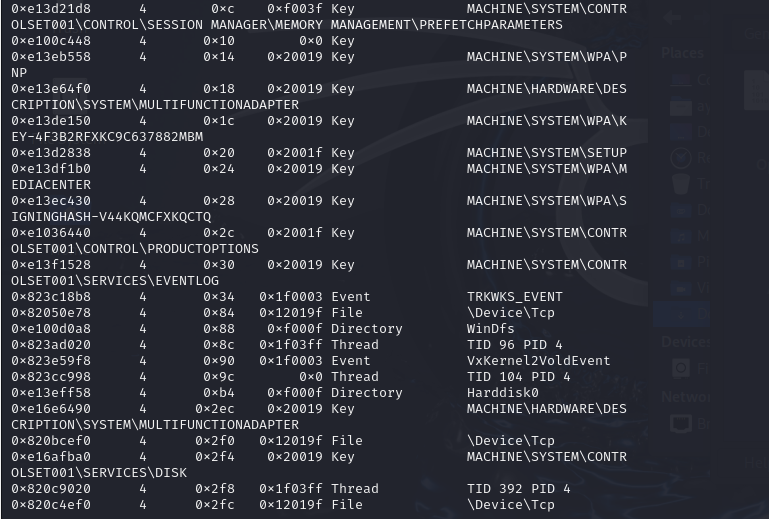
The output shows a list of processes running on the system at the time the memory dump was taken. Each line in the output represents a single process and provides the following information:

* **Base Address:** The base address of the process in memory.
* **Name:** The name of the process.
* **Pid:** The process ID (PID) of the process.
* **PPid:** The process ID (PID) of the parent process.
* **Thds:** The number of threads associated with the process.
* **Hnd:** The number of handles opened by the process.
* **Time:** The time at which the process was created.

8- 

**--profile=WinXPSP2x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP2 running on a 32-bit system.

9-

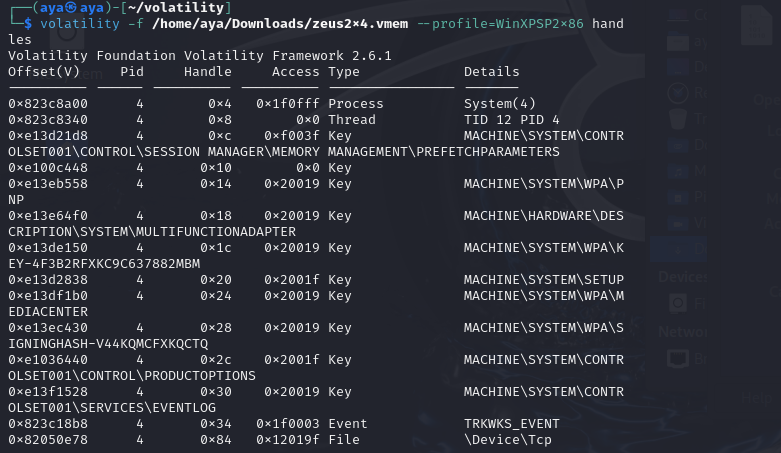


**--profile=WinXPSP3x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP3 running on a 32-bit system.

**handles:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "handles" plugin is designed to list the open handles in the memory dump.

The output shows a list of open handles in the memory dump. Each line in the output represents a single handle and provides the following information:

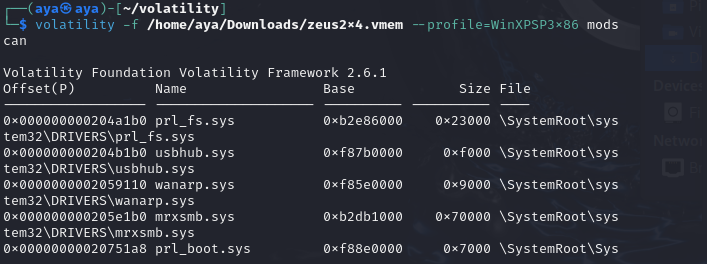
* **Offset(V):** The offset of the handle in the virtual address space of the process.
* **Pid:** The process ID (PID) of the process that owns the handle.
* **Handle:** The handle value.
* **Access Type:** The type of access granted to the handle (e.g., read, write, execute).
* **Details:** Additional information about the handle, such as the name of the process or the type of object the handle refers to.

10-

**--profile=WinXPSP2x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP2 running on a 32-bit system.

**Specific Handles in the Output:**

* **Process and Thread Handles (0x4, 0x8):** These are handles to the current process and its main thread.
* **Registry Handles (0xc, 0x10, 0x14, 0x18, 0x1c, 0x20, 0x24, 0x28, 0x2c, 0x30):** These are handles to various registry keys, providing insights into the system configuration and installed software.
* **Event Handle (0x34):** This is a handle to a system event object, likely used for internal system notifications.
* **File Handle (0x84):** This is a handle to a network file, indicating an active network connection.

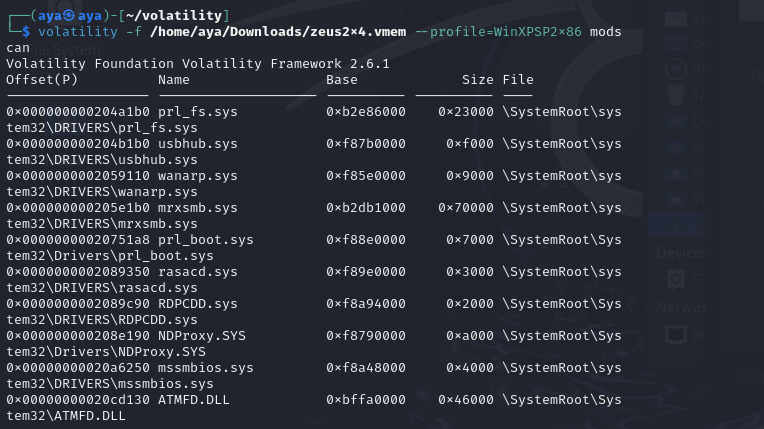
11-

**--profile=WinXPSP3x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP3 running on a 32-bit system.

**mods:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "mods" plugin is designed to list the loaded modules in the memory dump.

**Specific Modules in the Output:**

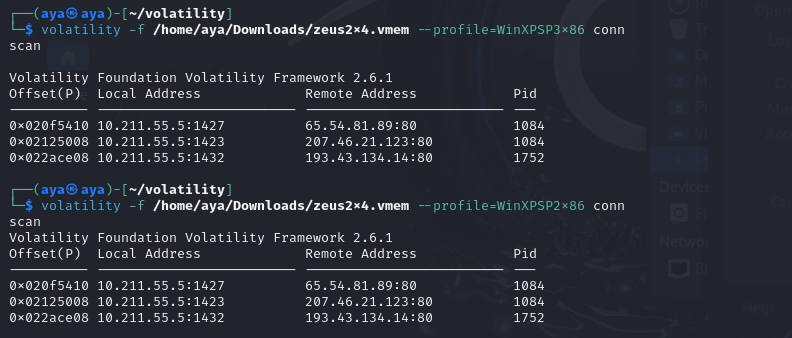
* **prl\_fs.sys, usbhub.sys, wanarp.sys, mrxsmb.sys, prl\_boot.sys:** These are all kernel-mode drivers. Kernel-mode drivers run in the highest privilege level and have direct access to system hardware and resources. By analyzing the loaded kernel-mode drivers, analysts can gain insights into the system's hardware configuration and installed drivers.

12-

**--profile=WinXPSP2x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP2 running on a 32-bit system.

**Specific Modules in the Output:**

* **Kernel-mode Drivers:** Several kernel-mode drivers are listed, such as prl\_fs.sys, usbhub.sys, wanarp.sys, mrxsmb.sys, prl\_boot.sys, rasacd.sys, RDPCDD.sys, NDProxy.sys, mssmbios.sys. Kernel-mode drivers run in the highest privilege level and have direct access to system hardware and resources. By analyzing the loaded kernel-mode drivers, analysts can gain insights into the system's hardware configuration and installed drivers.
* **User-mode DLL:** One user-mode DLL is listed, ATMFD.DLL. User-mode DLLs are typically associated with specific applications and provide functionality to those applications.

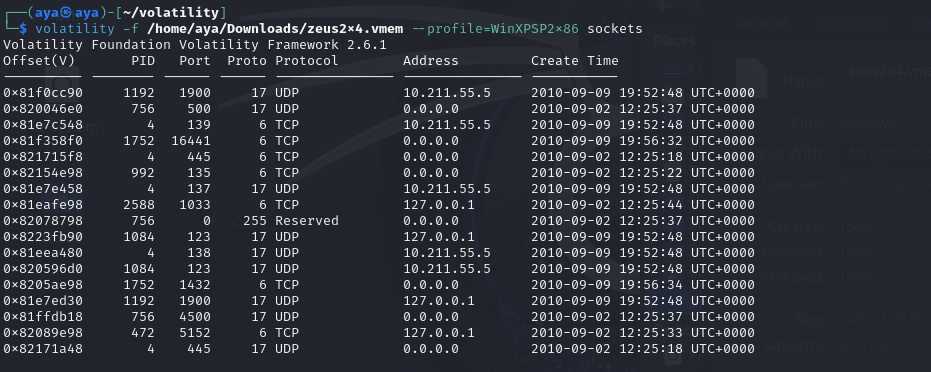
13-

**--profile=WinXPSP3x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP3 running on a 32-bit system.

**conn scan:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "conn scan" plugin is designed to list the open network connections in the memory dump.

The output of the command with --profile=WinXPSP2x86 is identical to the output with --profile=WinXPSP3x86. This suggests that the network connections were not affected by the specific Windows XP service pack version.

The output shows three active network connections, each with a different remote IP address and port number. This suggests that the system was communicating with multiple systems at the time the memory dump was taken.

14-

**--profile=WinXPSP2x86:** This argument tells Volatility which profile to use for the analysis. Profiles contain information about the structure and layout of the operating system's memory. In this case, the profile is for Windows XP SP2 running on a 32-bit system.

**sockets:** This is the plugin being used. Plugins are modules within Volatility that perform specific tasks. The "sockets" plugin is designed to list the open network sockets in the memory dump.

The output shows a variety of sockets, including TCP and UDP sockets, listening sockets, and sockets connected to remote systems.

Some of the sockets are associated with well-known services, such as SMB (port 445) and NetBIOS (ports 137 and 138).

Some of the sockets have local addresses of "0.0.0.0", which indicates that they are listening sockets that are accepting connections on all network interfaces.