Experiment- 3

Memory Analysis

1. List all running processes from a memory image

2. List all network connections from a memory image

3. Find out whether a firewall is set by analyzing a memory image Hint: volatility

Here's a step-by-step guide to perform memory analysis using **Volatility** on a memory image for your Experiment-3:

Prerequisites

bash

 # Install Volatility 3 (recommended for modern systems)

pip3 install volatility3

 # Or use Volatility 2 (for older memory images)

git clone https://github.com/volatilityfoundation/volatility.git

cd volatility

python2 setup.py install

Step 1: Profile Identification

First, identify the memory profile/OS version of your memory image:

 # Volatility 3

vol -f memory_image.raw imageinfo

 # Volatility 2

volatility -f memory_image.raw imageinfo

Output: Note the suggested profile (e.g., Win10x64_19041, LinuxUbuntu_x64, etc.)

Step 2: List All Running Processes

Volatility 3:

bash

 # List processes with full details

vol -f memory_image.raw windows.pslist

 # List processes with DLLs loaded (more verbose)

vol -f memory_image.raw windows.pslist.DllList

 # List processes with command-line arguments

vol -f memory_image.raw windows.cmdline

Volatility 2:

 # Replace PROFILE with your identified profile

volatility -f memory_image.raw --profile=PROFILE pslist

 # Processes with command line

volatility -f memory_image.raw --profile=PROFILE cmdline

 # Processes with loaded DLLs

volatility -f memory_image.raw --profile=PROFILE dlllist

Pro Tip: Look for suspicious process names, PIDs, parent-child relationships, or processes with no parent (PPID=0).

Step 3: List All Network Connections

Volatility 3:

 # Active TCP connections

vol -f memory_image.raw windows.netscan

 # Detailed network connections

vol -f memory_image.raw windows.netscan --dump

 # UDP connections

vol -f memory_image.raw windows.netscan --connections udp

Volatility 2:

 # TCP connections

volatility -f memory_image.raw --profile=PROFILE netscan

 # Connections with process info

volatility -f memory_image.raw --profile=PROFILE connections

 # Full network scan

volatility -f memory_image.raw --profile=PROFILE netstat

Key Indicators:

- Connections to unusual IPs/ports
- Processes with many outbound connections
- Suspicious DNS resolution

Step 4: Firewall Status Analysis

Method 1: Check Windows Firewall Service Status (Volatility 3)

 # Check if MpsSvc (Windows Firewall service) is running

vol -f memory_image.raw windows.pslist | grep -i mpssvc

 # Check service status

vol -f memory_image.raw windows.services

Method 2: Registry Analysis (Most Reliable)

 # Volatility 3 - Dump firewall registry keys

vol -f memory_image.raw windows.registry.hivelist

vol -f memory_image.raw windows.registry.printkey --key "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess"

 # Check firewall enable/disable value

vol -f memory_image.raw windows.registry.printkey --key "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile"

Method 3: Volatility 2 Registry Dump

volatility -f memory_image.raw --profile=PROFILE hivelist

volatility -f memory_image.raw --profile=PROFILE printkey -K "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess\Parameters\FirewallPolicy"

Method 4: Check Running Firewall Processes

bash

 # Look for firewall-related processes

vol -f memory_image.raw windows.pslist | grep -i -E "(firewall|wf|mpssvc|iphlpsvc)"

 # Check loaded firewall drivers

vol -f memory_image.raw windows.driverscan | grep -i firewall

Complete Analysis Script (Volatility 3)

Save this as memory_analysis.py:

#!/bin/bash

MEM_IMAGE=*$1*

echo "=== MEMORY ANALYSIS REPORT ==="

```
echo "Target: $MEM_IMAGE"
echo

echo "1. SYSTEM INFO"
vol -f $MEM_IMAGE imageinfo
echo

echo "2. RUNNING PROCESSES"
vol -f $MEM_IMAGE windows.pslist > processes.txt
echo "Processes saved to processes.txt"
echo

echo "3. NETWORK CONNECTIONS"
vol -f $MEM_IMAGE windows.netscan > network.txt
echo "Network connections saved to network.txt"
echo

echo "4. FIREWALL STATUS"
echo "Checking firewall service..."
vol -f $MEM_IMAGE windows.pslist | grep -i mpssvc
echo "Registry firewall status:"
vol -f $MEM_IMAGE windows.registry.printkey --key
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SharedAccess"
```

Run: chmod +x memory_analysis.py && ./memory_analysis.py memory_image.raw

Firewall Status Interpretation

EnableFirewall = 1 → Firewall ENABLED

EnableFirewall = 0 → Firewall DISABLED

MpsSvc service running → Firewall likely active

No firewall processes → Firewall likely disabled

Quick Summary Commands

 # One-liner for all key info

vol -f memory_image.raw imageinfo && vol -f memory_image.raw windows.pslist | head -20 && vol -f memory_image.raw windows.netscan | head -10

Expected Output Samples

Processes:

PID PPID ImageFileName Offset(V) Threads Handles SessionId Wow64 CreateTime ExitTime

4 0 System 0x8a123456 92 456 - False 2024-01-15 10:00:00 -

672 4 smss.exe 0x8b789abc 2 23 - False 2024-01-15 10:00:01 -

Network:

Proto LocalAddress LocalPort ForeignAddress ForeignPort State PID Owner

tcp 0.0.0.0:135 0.0.0.0:0 LISTENING 772 RpcSs

tcp 192.168.1.100:80 0.0.0.0:0 LISTENING 1234 httpd.exe

This gives you a complete memory forensics workflow. Start with profile identification, then systematically extract processes, network activity, and firewall status!