Computational Forensics

Memory Forensics

COMP 8920 - Computer and Network Forensics

Lesson 03

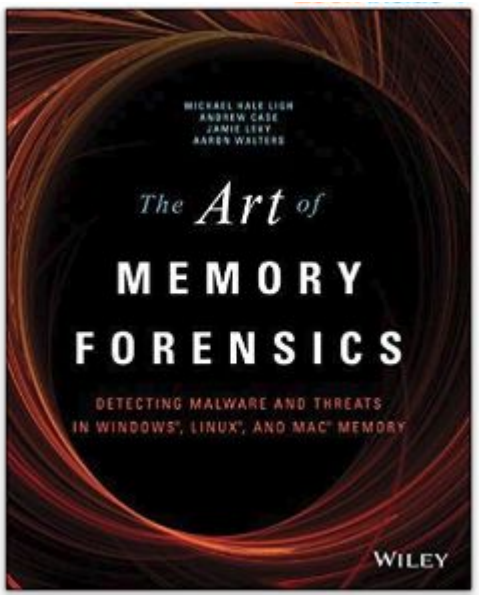
Outlines

- Introduction to Memory Forensics
- Technical Background
- Memory Acquisition
- Memory Analysis

Introduction to Memory Forensic

Overview

- The process of investigating data and potential evidences in live systems (live forensics) by examining volatile memory.
- Memory forensics relies on capturing live memory from a running computer system.
- RAM analysis consists of performing forensic analysis on the data gathered from the live computer.

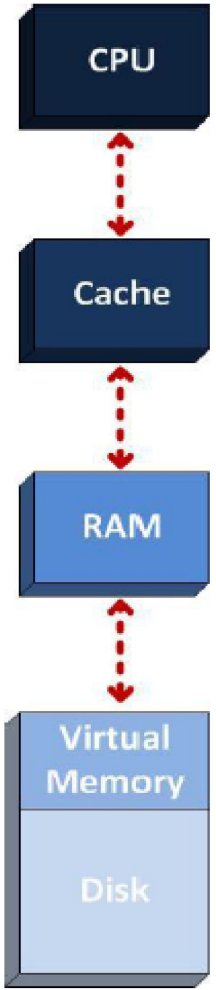


Why Memory Forensics?

- Main memory contains unique data, not just a duplicate of data that can be found elsewhere.
- When examining a network-based attack, main memory provides the missing link between network data (capture/IDS alert) and possible artifacts on a disk.
- Only main memory documents the current status of a computer/device.
- Some attacks don’t leave traces on disk, but only in memory.

Why Memory Forensics?

- Everything traverses RAM
  - Processes and threads
  - Malware (including rootkit technologies)
  - Network sockets, URLs, IP addresses
  - Open files
  - User generated content
    - Passwords, caches, clipboards
  - Encryption keys
  - Hardware and software configuration
  - Windows registry keys and event logs.



Why Memory Forensics?

- Best place to identify malicious software activity
  - Study running system configuration
  - Identify inconsistencies (contradictions) in system
  - Bypass packers, binary obfuscators, rootkits (including kernel mode) and other hiding tools.
- Analyze and track recent activity on the system
  - Identify all recent activity – in context
  - Profile user or attacker activities

Why Memory Forensics?

- Collect evidence that cannot be found anywhere else
  - Memory-resident malware
  - Chat threads
  - Network activities (open ports, network flows)

Lifetime of Volatile Data

- Forensics analysis of physical memory reveals “private” data, even weeks after use.
- Most current systems take no steps to overwrite sensitive data in memory
- Most applications that handle sensitive data weren’t specifically designed to deal with sensitive data (e.g text editors, spreadsheet, etc)
- Password entered into web browser may be duplicated dozens of times (kernel buffers, window manager, application buffers, dynamic memory allocator)

Lifetime of Volatile Data

- Live forensic analysis can reveal sensitive data months after last use.
- Even when process or application crash, memory dump might still contains sensitive information.
- Even after system reboot or shutdown, we can still obtain some sensitive in-memory data. (example from hibernate files, disk swap, etc).

Lifetime of Volatile Data

- Ideal Lifetime: period of time data is actually in use

  [first write after allocation → last read before deallocation]

- Natural Lifetime: period of time that data remains readable

  [first write after allocation → first write after reallocation (first overwrite)]

- Secure Deallocation Lifetime: zero on deallocation

  [first write after allocation → deallocation (when it is zeroed)]

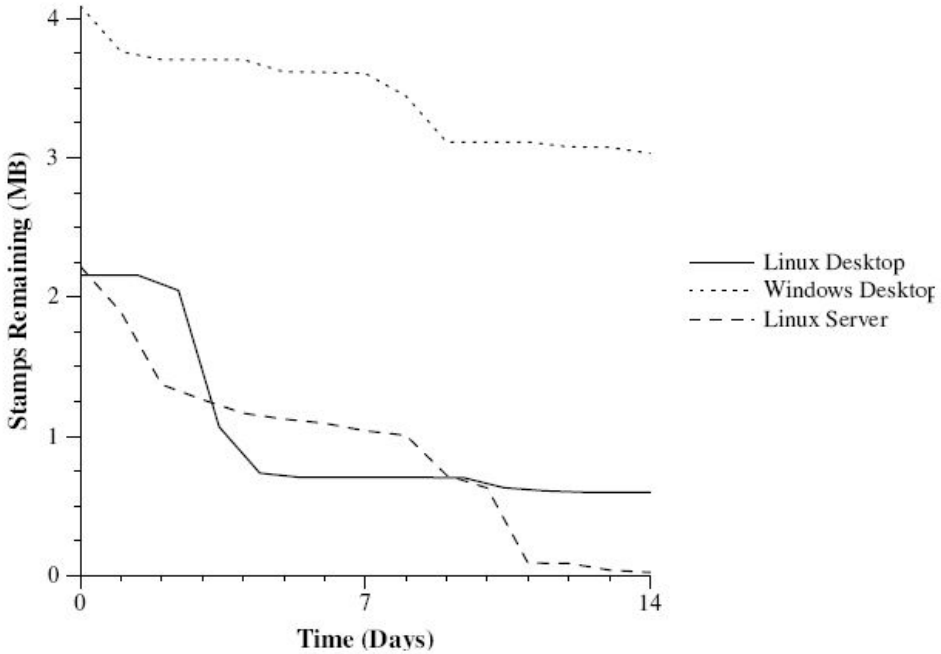
Lifetime of Volatile Data Experiment

- **Settings:**
  - Linux machine, 1GB RAM
  - Windows machine, 1GB RAM
  - Linux server, 256MB RAM
- **Volatile Data:**
  - **Linux version of experiment:**
    - **64MB** buffer filled with **20 byte** “stamps”
    - Allocate and release
    - Each stamp: magic number, serial number, checksum
  - **Windows version:**
    - **4MB buffer** transmitted between machines

Lifetime of Volatile Data Experiment

- **Results:**
  - Immediately after dellocation: 2 - 4MB of stamps remain
  - 14 days later: 23KB – 3MB of stamps remain
  - Additional 14 days on Linux server: 7KB of stamps remain
  - Most remaining stamps “trapped” in blocks of memory in the kernel slab allocator
  - **Reboot** on **Thinkpad T30 laptop**: stamps remain after 30 seconds w/o power

Lifetime of Volatile Data Experiment



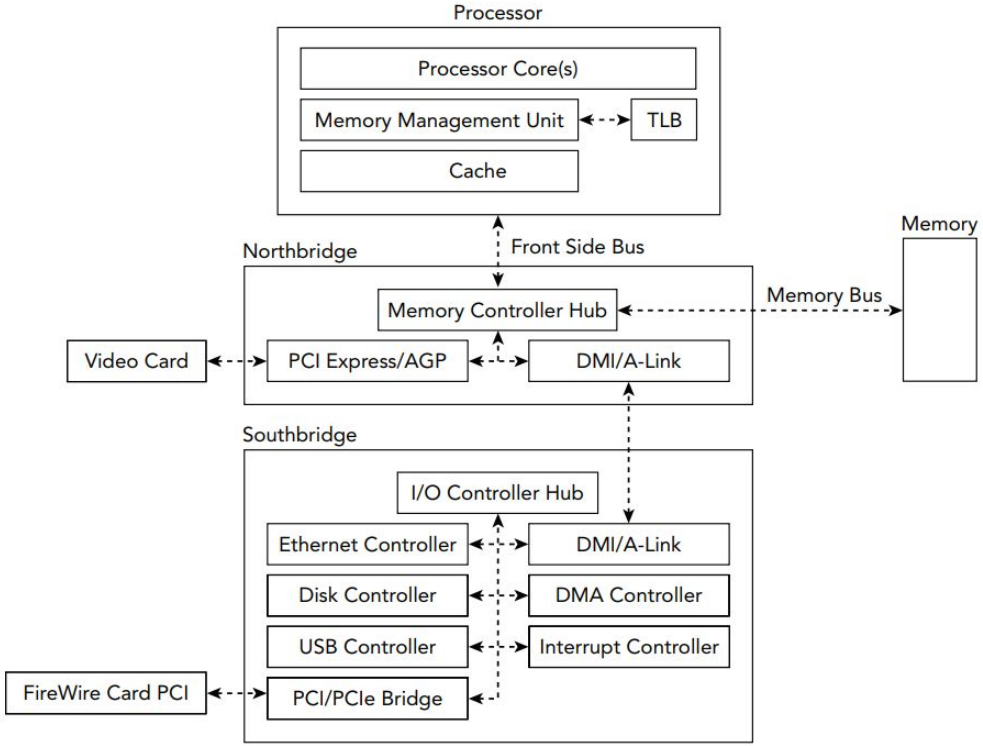
Technical Background

Persistence of (Post-Reboot) Memory

In 2006/2007 Research conducted at the University of New Orleans showed that:

- Many systems retain at least some data after a **warm reboot**, reset, or even **cold reboot**
- Highly dependent on model and **BIOS settings**
- Potentially useful as a “last resort” for obtaining live forensics data, assuming computer model is known to have post-reboot persistent memory

Physical Organization of Modern System



How the Computer Memory Work?

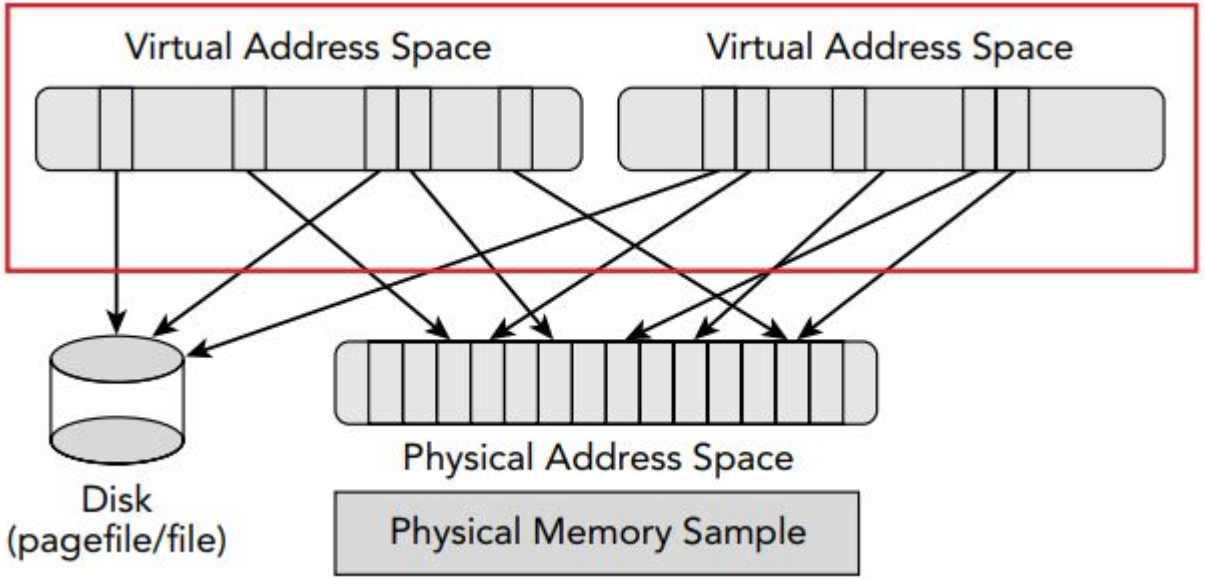
Memory Management: refers to the operating system's algorithms for managing the allocation, deallocation, and organization of physical memory.

While all OS and hardware use in general the same methods and techniques to manage computer memory, the actual implementation is unique for different OS and computer architecture.

Virtual Memory

- Operating systems provide each process with its own private virtual address space.
- This abstraction creates a separation between the logical memory that a process sees and the actual physical memory installed on the machine.
- As a result, you can write programs as if they have access to the entire address space and in which all ranges are memory resident.

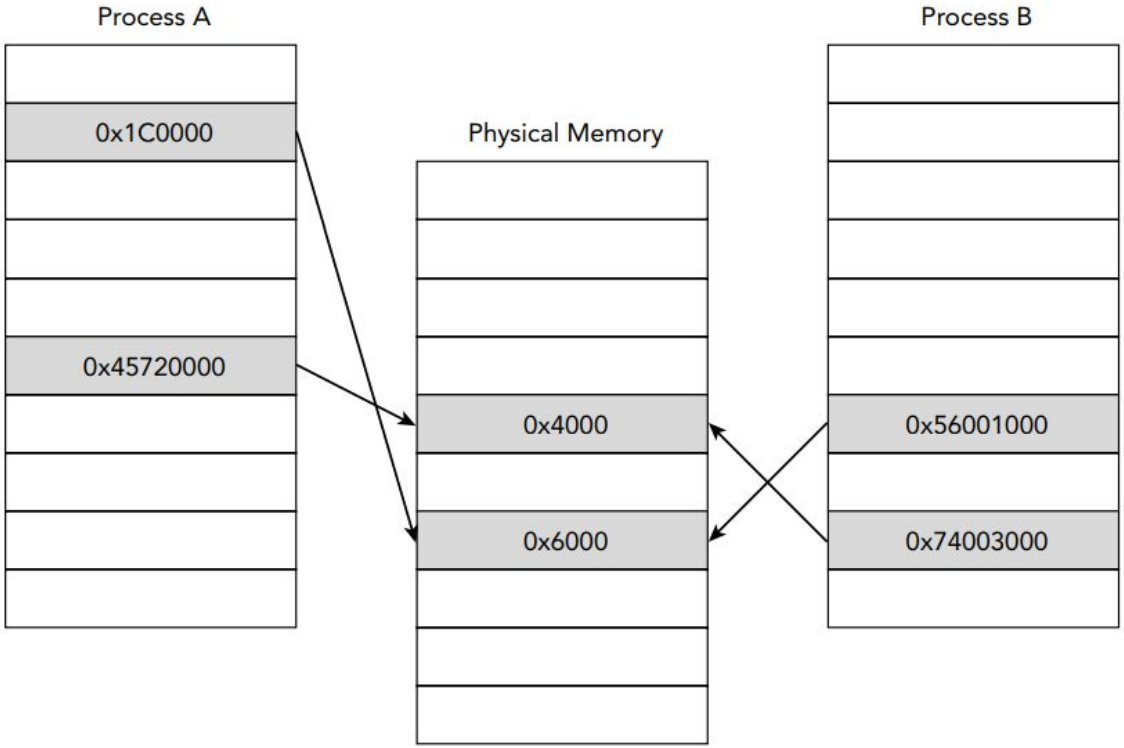
Virtual Memory



Shared Memory

- Modern operating systems also provide mechanisms that allow processes to share memory.
- We can view shared memory as memory that is accessible from more than one virtual address space.
- One common use for shared memory is to provide an efficient means of communication between processes.
- After a shared region is mapped into virtual address spaces, processes can use the region to exchange messages and data.
- Instead of allocating multiple physical pages that contain the same data, we can create a single instance of the data in physical memory and map various regions of virtual memory to it

Shared Memory



Demand Paging

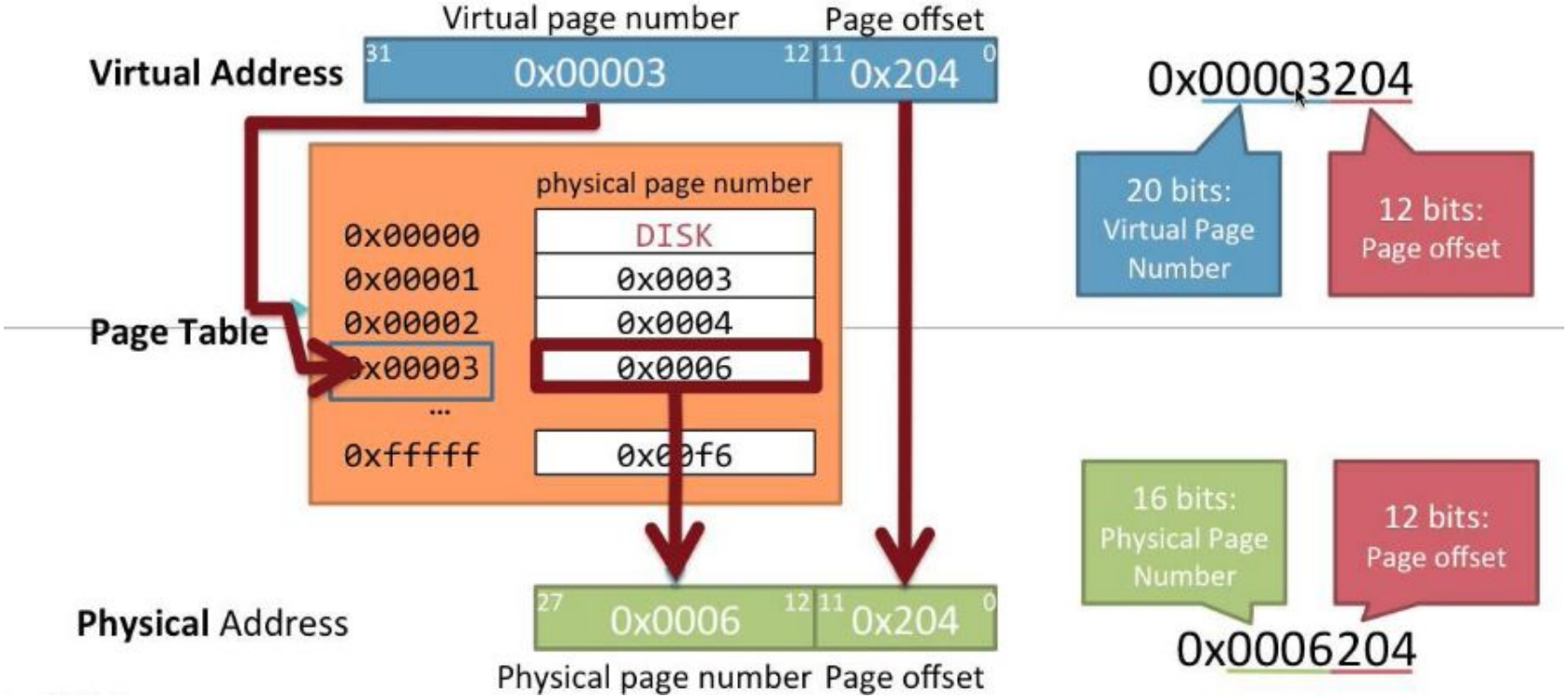
- The mechanism that is commonly used to implement virtual memory is demand paging. It Provides the ability to virtualize the linear address space.
- Simulates a large linear address space with a modest amount of physical memory and disk storage.
- Each 32-bit linear address space is broken up into fixed-length sections, called **pages**, which can be mapped into physical memory in an arbitrary order.
- When a program attempts to access a linear address, this mapping uses memory-resident page directories and page tables to translate the linear address into a physical address.

Demand Paging

If a thread/process attempts to access a page that is not resident, the hardware generates a **page fault**. This means one of three things:

1. The process attempts to access a memory address that does not belong to it (segmentation fault). The OS will kill the process
2. The process tries to access a page that has been loaded into the physical memory yet.
3. The process tries to access a page that the OS swapped to the disk, and it is currently in a page file.

How Virtual Memory Works?



Process Management

- A process is an instance of a program executing in memory.
  - The operating system is responsible for managing process creation, suspension, and termination.
  - When a program executes, a new process is created and associated with its own set of attributes, including a unique process ID and address space.
- A process provides the execution environment, resources, and context for threads to run.
- A thread is the basic unit of CPU utilization and execution. A thread is often characterized by a thread ID, CPU register set, and execution stack(s), which help define a thread's execution context.

Memory Acquisition

Memory Acquisition

- Techniques used to extract volatile memory images from target systems are defined as either hardware-based or software-based solution.
- **Software-based solutions** rely on the operating system in order to perform memory capture.
- **Hardware-based solutions** in contrast, directly access the volatile memory of the target system.
- Hardware-based solutions for memory acquisition have been considered the most reliable as it is difficult to obtain a complete and accurate memory snapshot from software

Memory Acquisition Process Efficiency

- In order to measure the efficacy of acquisition techniques we consider two criteria:
  - The fidelity and reliability of the generated image
  - The availability of the mechanism (software or hardware) used to capture the image.

How do we verify that the image is accurate and reliable?

What is more important reliability or availability?

Memory Acquisition: Fidelity & Reliability Criteria

- Ensure that the image obtained is a “precise copy of the host’s memory”.
- In particular the obtained memory image should be trustworthy (i.e. the capture process is not interfered with by malware or other actors).
- To ensure the fidelity and the reliability the memory acquisition technique should support verify the accuracy of the generated method.

Memory Acquisition: Availability Criteria

- The technique must work on arbitrary computers and/or devices. Essentially meaning that the method be operating system agnostic and not require specialised techniques.
- The technique does not make any assumptions about particular pre-incident preparatory measures and can be applied without knowledge of the execution environment and without requiring that any pre-configurations exist prior to its execution.

Memory Acquisition: Hardware-Based Techniques

- Dedicated hardware techniques are those that involve the installation of a physical device in order to assist investigators in acquiring a memory image from a target system.
- One example for Dedicated hardware techniques is **Tribble**. It is a PCI card that at enables the capture of physical memory using **Direct Memory Access (DMA).**
- It is able to obtain a precise copy of physical memory **without any interaction with the operating system** running on the target machine.

Memory Acquisition: Hardware-Based Techniques

How it works?

- The Tribble device itself must be installed prior to an investigation.
- When it is activated, the host machine is suspended to prevent any malicious code being executed during memory imaging.
- Once the memory dump is completed control is returned to the host operating system.
- Dedicated hardware solutions for memory acquisition generally have **low availability**.

https://www.youtube.com/watch?v=vJszLtaIyIk

Memory Acquisition: Hardware-Based Techniques

Why it is considered an accurate approach?

Because the assumption that as dedicated hardware does not rely on the operating system on the target machine it is able to produce a true picture of a system's memory.

Vömel and Freiling discuss recent experiments that show that the **northbridge chipset** of motherboards is able to be reprogrammed to provide a subverted view of the system's memory, allowing regions to be swapped in and out with both software and hardware processes alike oblivious to the substitution. **(hardware itself has been compromised)**

Memory Acquisition: Hardware Bus

- A hardware bus facilitates data transfer between components (such as PCI) or between devices (such as USB or FireWire) within computer architectures.
- Memory acquisition techniques have been developed to exploit the use of the **FireWire hardware bus** to access the volatile memory of a system.
- FireWire-based approaches have shown to be quite popular primarily because the bus provides DMA by design.

Memory Acquisition: Hardware Bus

- Compared to dedicated hardware, hardware bus acquisition techniques are much more highly available. This is because hardware bus ports such as FireWire are quite common across both portable and desktop computers.
- Researchers illustrate that when **FireWire** methods of acquisition access the **Upper Memory Area** (UMA) region of memory they can cause random **system crashes**. In addition memory images captured through FireWire are often **corrupt or incomplete.**

Memory Acquisition: Software-Based Techniques

Using Virtualisation:

- An important characteristic of VMs, is that they are capable of having their execution paused or suspended.
- The state of the machine is temporarily frozen and its virtual memory is saved to a hard disk on the underlying host.
- All of this volatile memory data is stored to a .vmem or .vmss file located in the working directory of the guest machine.
- Has a high level of reliability and availability (only works for virtual machines)

Memory Acquisition: Software-Based Techniques

Crash Dumps

- Windows operating systems can be configured to **write memory dumps to a file** when the system unexpectedly stops working.
- The main memory as well as relevant CPU information are saved in the system root directory. These dump files can then be examined with **Microsoft's Debugging Tools** for Windows or manually analysed
- This technique is only suitable to limited situations as the Windows registry must be modified in order to enable the manual crash dump technique.

Memory Acquisition: Software-Based Techniques

User-Mode Applications

- Low level of availability, because it is very hard to capture memory image in user-mode (OS security services prevent that).
- The primary disadvantages of pure software based approaches is that it must be loaded into volatile memory to execute.
- A machine compromised with a rootkit may deny direct access to physical memory and return a modified representation during image generation.

Memory Acquisition: Software-Based Techniques

Kernel-Mode Applications

- Software vendors and researchers are increasingly focusing on developing kernel-mode applications and drivers that can be used to create forensic copies of volatile memory.
- kernel-based approaches suffer from availability issues as they require the investigator to have administrator privileges to install.
- This also a software solutions that will run and execute on the target machine main memory.

Memory Acquisition: Software-Based Techniques

Operating System Injection

- A new approach (proof of concept) for memory acquisition.
- The approach **injects an independent operating system** into the kernel of the target machine.
- The target machine's native kernel execution is then frozen then injected OS can then provide an atomic snapshot of the machine's volatile data.
- Require **specific hardware architecture** and only run on a single cpu core.

Memory Acquisition: Software-Based Techniques

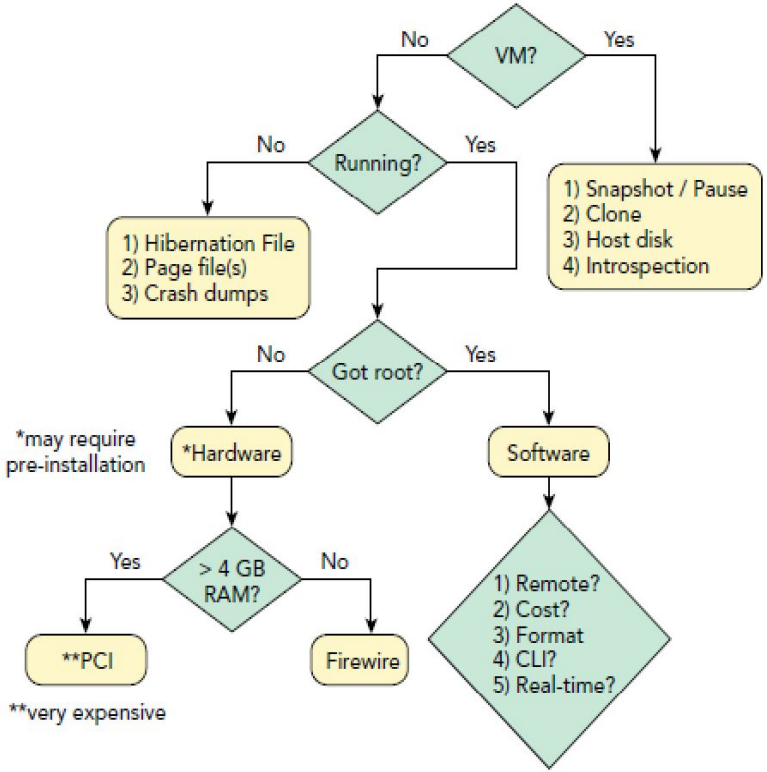
Cold Booting Technique

- The approach is based on the observation that information is not erased from volatile memory immediately after powering off a machine and may be recovered for a non-trivial amount of time.
- We could **reboot the machine and launch a custom kernel** with a small memory footprint that gives access to residual memory.
- We could **cut power, restore power and then launch the custom kernel**.
- We could **cut the power of the target machine and translating the RAM modules to a second computer** which is configured to extract their state.

Memory Acquisition: Techniques Summary

Technique	Atomicity	Availability	Comments
Dedicated Hardware (3.1.1)	High	Low	Atomicity may be compromised with hardware tampering.
Hardware Bus (3.1.2)	Moderate	High	Atomicity affected by random crashes in some cases.
Virtualisation (3.2.1)	High	High	In environments that utilise virtualisation.
Crash Dumps (3.2.2)	Low	Low	Not all memory is dumped, can overwrite system page file and requires registry modifications to work.
User-mode Applications (3.2.3)	Low	High	Easily subverted, will modify memory when capturing it and will not have access to entire memory range.
Kernel-mode Applications (3.2.4)	Low	Moderate	Easily subverted and will modify memory when capturing it.
Operating System Injection (3.2.5)	High	Low	Reliance on hardware platforms and very slow.
Cold Booting (3.3)	High	High	Requires off the shelf items and Syslinux

Memory Acquisition: Cheat Sheet



Memory Analysis

Memory Analysis

The most common types of information that researchers are developing techniques to extract include:

1. Running Processes (normal, compromised, and malicious).
2. Cryptographic keys.
3. Network connections.
4. Command history.
5. Open files.
6. Systems status.

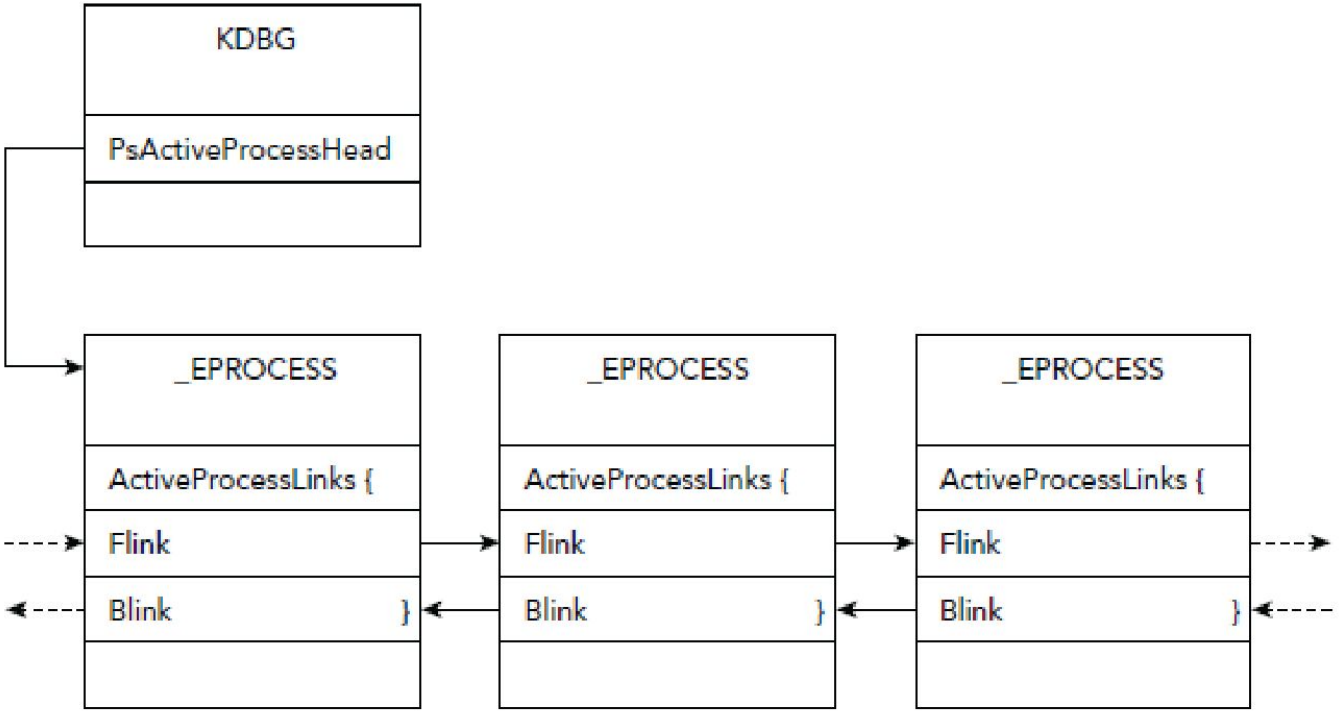
Memory Analysis: Processes Detection

- Windows operating systems maintain lists of **EPROCESS** and ETHREAD structures in memory, to help schedule and execute programs.
- This means we could extract the list of running process by enumerating the lists of the processes and threads maintained in areas such as the **Process Control Block (PCB)** in a memory image on Windows OS machines.
- However, some malicious processes (e.g. malware) could unlink itself from the **ActiveProcessLinks** list

Memory Analysis: Processes Detection

- Another approach developed by digital forensic researchers is a **signature based approach**.
- Using a signature (a set of rules) to precisely describe the structure of a system process or thread for different operating systems.
- These signatures are used to extract all of the processes and threads from memory, even those that have unlinked themselves.
- By comparing the process list from the OS and the extracted process list from the signature-based tool we could detect a malicious process that attempts to hide itself.

Memory Analysis: Processes Detection



Memory Analysis: Cryptographic Key

- Encryption keys that are used to encrypt data are stored during the encryption and decryption process in the main memory.
- Analyzing volatile memory might be the only way to decrypt encrypted data found on the digital evidence.
- The encryption and the cryptographic parameters used to generate the key for secure and encrypted hard drives are usually stored in the main memory as long as the hard drive is mounted.
- Research proposed different approaches to to find keys and **decrypted hard drives** that were protected with tools like BitLocker, TrueCrypt, dm-crypt and Loop-AES

Memory Analysis: Network Analysis

- By investigating memory dump/image we could extract information about network connectivity.
- This includes local and remote sockets (IP addresses and Ports numbers).
- Then, we need to link the discovered connections to specific processes and programs.

Memory Analysis: Open File Analysis

- By analyzing the **(Process Environment Block) PEB** table in the memory image of a given process we find pointer references multiple lists that contain the full name, size and base address of all **loaded libraries.**
- This enable the investigator to find malicious libraries inject by an attacker or a malware, or **benign** utilities or libraries abused by the malware.
- By analyzing data in the **Virtual Address Descriptor (VAD)** structure we could find references to the **files** the processes interact with.

Memory Analysis: System State Analysis

- Analyzing the memory image to extract system state is very important.
- As an example is analyzing the **Window clipboard** could enable use to discover user's password, copied sections between documents, URLs, etc.
- Even basic **string search** could enable the investigator to extract web browsing history, credit card information, etc.

Memory Analysis Tools

The development of memory forensics tools and techniques started in around 2005.

There are several tools for memory forensics, both commercial and freeware.

1. Windows Memory Toolkit from MoonSols (commercial)
2. Responder (Professional and Community) from HB Gary (commercial)
3. Memoryze from Mandiant (Free)
4. The Volatility Framework from Volatile Systems (Free)

Summary

In this class we covered:

- Memory Forensics techniques and methods

Note: Make sure to read the reading materials posted on the course website.

What is Next?

In our next Class we continue talking about computational forensics topics and we will focus on file carving.

References

1. Memory Forensics: Review of Acquisition and Analysis Techniques by Grant Osborne 2013
2. Live Forensics Tutorial, by Frank Adelstein and Golden G. Richard, in USENIX Security 2007
3. Memory Forensics Lecture Notes, by Dr. Issa Traore,
4. The Art of Memory Forensics, by Ligh and et al, Wiley