508.3 - Memory Forensics in Incident Response and Threat Hunting

[Incident Response](#_Toc121856516) 4-21

[Enterprise IR challenges](#_Toc121856517) 4-5

[Rapid-Response Tooling](#_Toc121856518) 6

[Enterprise IR Agents](#_Toc121856519) 7-20

[Remote Access Agents](#_Toc121856520) 7

[Remote Analysis Agents](#_Toc121856521) 8

[F-Response](#_Toc121856522) 9-12

[KAPE](#_Toc121856523) 13-15

[Velociraptor](#_Toc121856524) 16-20

[EDR](#_Toc121856525) 22-27

[Why EDR?](#_Toc121856526) 24

[EDR Challenges](#_Toc121856527) 25

[Memory Forensics](#_Toc121856528) 26-27

[Memory Forensics](#_Toc121856529) 30-202

[Why Memory Forensics?](#_Toc121856530) 30-32

[Overview and possible findings/artifacts](#_Toc121856531) 31

[Memory analysis advantages](#_Toc121856532) 32

[Memory Acquisition](#_Toc121856533) 33-39

[Windows](#_Toc121856534) 34-37

[Overview](#_Toc121856535) 34-35

[hiberfil.sys](#_Toc121856536) 36-37

[Virtual machines](#_Toc121856537) 38

[Memory Analysis Introduction](#_Toc121856538) 40-118

[Definition](#_Toc121856539) 40

[Windows Analysis](#_Toc121856540) 42-44

[KDBG / KPCR / PEB / VAD](#_Toc121856541) 42

[Unlinking](#_Toc121856542) 43

[Hooking](#_Toc121856543) 43

[Memory structure simplified 4](#_Toc121856544)4

[Volatility](#_Toc121856545) 45-57

[Overview](#_Toc121856546) 45

[How-to](#_Toc121856547) 46-47

[Help flag (-h)](#_Toc121856548) 48

[Volatility Version 3.0](#_Toc121856549) 49

[Volatility Profiles](#_Toc121856550) 51

[Image Identification](#_Toc121856551) 52

[Build Numbers (how-to)](#_Toc121856552) 52

[kdbgscan](#_Toc121856553) 53-55

[Hibernation file conversion](#_Toc121856554) 56-57

[Starting analysis (finding the first hit)](#_Toc121856555) 58-118

[Step 1: Identify rogue processes](#_Toc121856556) 59-87

[Process Blocks (EPROCESS)](#_Toc121856557) 60-61

[Analyzing Processes](#_Toc121856558) 62-64

[Volatility Plugins](#_Toc121856559) 65-86

[pslist (process overview)](#_Toc121856560) 66-68

[psscan](#_Toc121856561) 71-74

[pstree](#_Toc121856562) 75-79

[Baseline](#_Toc121856563) 84-86

[Know normal (Windows)](#_Toc121856564) 69

[EchoTrail](#_Toc121856565) 70

[WMI & PowerShell processes](#_Toc121856566) 80-81

[Overview](#_Toc121856567) 80-81

[Suspicious WMI processes](#_Toc121856568) 82-83

[wmiprvse.exe](#_Toc121856569) 82

[scrcons.exe](#_Toc121856570) 83

[Review](#_Toc121856571) 87

[Step 2: Analyzing process objects](#_Toc121856572) 89-110

[Process overview](#_Toc121856573) 90-91

[Volatility plugins overview](#_Toc121859986) 92

[dlllist](#_Toc121859987) 93-95

[Cobalt Strike Sacrificial Processes](#_Toc121859988) 96-97

[getsids](#_Toc121859989) 98-101

[Understanding SIDs](#_Toc121859990) 102-103

[Handles](#_Toc121859991) 104-106

[Named pipes](#_Toc121859992) 107

[Mutants/Mutexes](#_Toc121859993) 108-109

[Review](#_Toc121859994) 110

[Step 3: Network Artifacts](#_Toc121859995) 111-118

[Overview](#_Toc121859996) 112

[Unusual behavior](#_Toc121859997) 113

[Volatility plugins](#_Toc121859998) 114-117

[Overview](#_Toc121859999) 114

[netscan](#_Toc121860000) 115-117

[Review](#_Toc121860001) 118

[Code Injection, Rootkits and Extraction](#_Toc121860002) 120-198

[Starting analysis (finding the first hit)](#_Toc121860003) 121

[Step 4: Look for Code Injection](#_Toc121860004) 122-148

[Why code injection?](#_Toc121860005) 123

[Overview (DLL injection/Process hollowing)](#_Toc121860006) 124

[Simple DLL Injection](#_Toc121860007) 125-127

[Volatility plugins](#_Toc121860008) 128-137

[Overview](#_Toc121860009) 128

[ldrmodules](#_Toc121860010) 129-137

[Reflective Code Injection](#_Toc121860011) 138-139

[Overview](#_Toc121860012) 138

[Detecting reflective/hidden code injection](#_Toc121860013) 139

[Detecting code injection](#_Toc121860014) 140-148

[malfind](#_Toc121860015) 140-147

[Review](#_Toc121860016) 148

[Step 5: Hooking and Rootkit Detection](#_Toc121860017) 149-169

[Overview](#_Toc121860018) 150-151

[SSDT/IDT/IAT/IRP](#_Toc121860019) 150-151

[Volatility plugins](#_Toc121860020) 152-168

[Overview](#_Toc121860021) 152

[ssdt](#_Toc121860022) 153-156

[psxview (cross-view analysis)](#_Toc121860023) 158-159

[modscan & modules](#_Toc121860024) 161-163

[driverbl](#_Toc121860025) 164-165

[apihooks](#_Toc121860026) 166-168

[Direct Kernel Object Manipulation (DKOM)](#_Toc121860027) 157

[Review](#_Toc121860028) 169

[Plugin use-cases](#_Toc121860029) 169

[Step 6: Extracting objects from memory](#_Toc121860030) 172-198

[Volatility plugins](#_Toc121860031) 173-180

[dlldump](#_Toc121860032) 174-175

[moddump](#_Toc121860033) 176-178

[procdump](#_Toc121860034) 179

[memdump / vaddump](#_Toc121860035) 180

[strings](#_Toc121860036) 181

[grep](#_Toc121860037) 182

[Using strings and grep with memdump](#_Toc121860038) 183-184

[Command history and active consoles](#_Toc121860039) 185-188

[cmdscan & consoles](#_Toc121860040) 185-186

[Windows 10 memory compression](#_Toc121860041) 189

[Extracting files](#_Toc121860042) 191-196

[dumpfiles](#_Toc121860043) 191-193

[filescan](#_Toc121860044) 192

[filescan + dumpfiles](#_Toc121860045) 193

[Extracting Registry](#_Toc121860046) 197

[shimcachemem](#_Toc121860047) 197

[Review](#_Toc121860048) 198

[Scaling Analysis](#_Toc121860049) 201-203

[Live memory forensics](#_Toc121860050) 201-202

[Using IOCs](#_Toc121860051) 203

[openioc\_scan](#_Toc121860052) 203

[yarascan](#_Toc121860053) 203

[page\_brute.py](#_Toc121860054) 203

# Incident Response

## Enterprise IR challenges

## Rapid-Response Tooling

## Enterprise IR Agents

### Remote Access Agents

### Remote Analysis Agents

### F-Response

### Velociraptor

### KAPE

# EDR

## Why EDR?

## EDR Challenges

## Memory Forensics

# Memory Forensics

## Why Memory Forensics?

### Overview and possible findings/artifacts

### Memory analysis advantages

## Memory Acquisition

### Windows

#### Overview

#### hiberfil.sys

### Virtual machines

## Memory Analysis Introduction

### Definition

### Windows Analysis

#### KDBG / KPCR / PEB / VAD

#### Unlinking

#### Hooking

#### Memory structure simplified

### Volatility

#### Overview

#### How-to

#### Help flag (-h)

#### Volatility Version 3.0

#### Volatility Profiles

#### Image Identification

##### Build Numbers (how-to)

##### kdbgscan

#### Hibernation file conversion

### Starting analysis (finding the first hit)

#### Step 1: Identify rogue processes

##### Process Blocks (EPROCESS)

##### Analyzing Processes

##### Volatility Plugins

###### pslist (process overview)

###### psscan

###### pstree

###### Baseline

##### Know normal (Windows)

##### EchoTrail

##### WMI & PowerShell processes

###### Overview

##### Suspicious WMI processes

###### wmiprvse.exe

###### scrcons.exe

##### Review

#### Step 2: Analyzing process objects

##### Process overview

###### Process objects

##### Volatility plugins overview

###### dlllist

###### Cobalt Strike Sacrificial Processes

###### getsids

##### Understanding SIDs

##### Handles

##### Named pipes

##### Mutants/Mutexes

##### Review

#### Step 3: Network Artifacts

##### Overview

##### Unusual behavior

##### Volatility plugins

###### Overview

###### netscan

##### Review

## Code Injection, Rootkits and Extraction

### Starting analysis (finding the first hit)

#### Step 4: Look for Code Injection

##### Why code injection?

###### Overview (DLL injection/Process hollowing)

###### Simple DLL Injection

Attach

Allocate

Execute

##### Volatility plugins

###### Overview

###### ldrmodules

datasources

LoadLibraryA() example

TDSS example (unlinked DLL)

Stuxnet example (hollowing)

##### Reflective Code Injection

###### Overview

###### Detecting reflective/hidden code injection

##### Detecting code injection

###### malfind

Overview

Example (MZ present)

Example (no MZ)

Example (False positive)

Countermeasures

###### Review

#### Step 5: Hooking and Rootkit Detection

##### Overview

###### SSDT/IDT/IAT/IRP

##### Volatility plugins

###### Overview

###### ssdt

###### psxview (cross-view analysis)

###### modscan & modules

###### driverbl

###### apihooks

##### Direct Kernel Object Manipulation (DKOM)

##### Review

###### Plugin use-cases

#### Step 6: Extracting Processes, Drivers and Objects

##### Volatility plugins

###### dlldump

###### moddump

###### procdump

###### memdump / vaddump

##### strings

##### grep

##### Using strings and grep with memdump

##### Command history and active consoles

###### cmdscan & consoles

##### Windows 10 memory compression

##### Extracting files

###### dumpfiles

###### filescan

###### filescan + dumpfiles

##### Extracting Registry

###### shimcachemem

##### Review

## Scaling Analysis

### Live memory forensics

### Using IOCs

#### openioc\_scan

#### yarascan

#### page\_brute.py