ARGOS 하계 정보보호 교육

Digital Forensic

3회차 –메모리 포렌식

발 표 자 | 23학번 나소진

E-Mail | sjna@o.cnu.ac.kr

2024.08.29

CONTENTS

목차

CHAPTER
01

강의 개요

강의 구성 및 계획

강의 개요

- 주차별 계획

Week 1 (8/16)
숨겨진 데이터 찾기와 손상된 파일 복구

- Digital Forensic 개요
- File Format
- Steganography

Week 2 (8/22)
디스크 포렌식

- Disk Image
- File System
- Windows 주요 아티팩트

Week 3 (8/29)
메모리 포렌식

- 휘발성 데이터
- Volatility

강의 개요

- 메모리 포렌식 개요
  - 정의
    - 주기억장치(RAM)에 남아있는 휘발성 데이터 흔적을 분석하는 기법
  - 목적
    - 프로세스(악성코드, 서비스 등)의 행위 탐지
    - 네트워크 연결 정보
    - 사용자 행위
    - 복호화, 언패킹, 디코딩된 데이터
    - 패스워드와 암호 키 획득

강의 개요

- 메모리 포렌식 개요
  - 한계
    - 디지털 포렌식은 일반적으로 사고 발생 이후에 진행

      → 당시 메모리 데이터 획득 어려움으로 수사에서 사용률은 떨어짐
  - (그럼에도) 필요성
    - 암호화된 일부 데이터들에 대한 평문 획득이 가능
    - 저장장치에 흔적을 남기지 않는 악성코드 감염 여부 확인 가능

CHAPTER

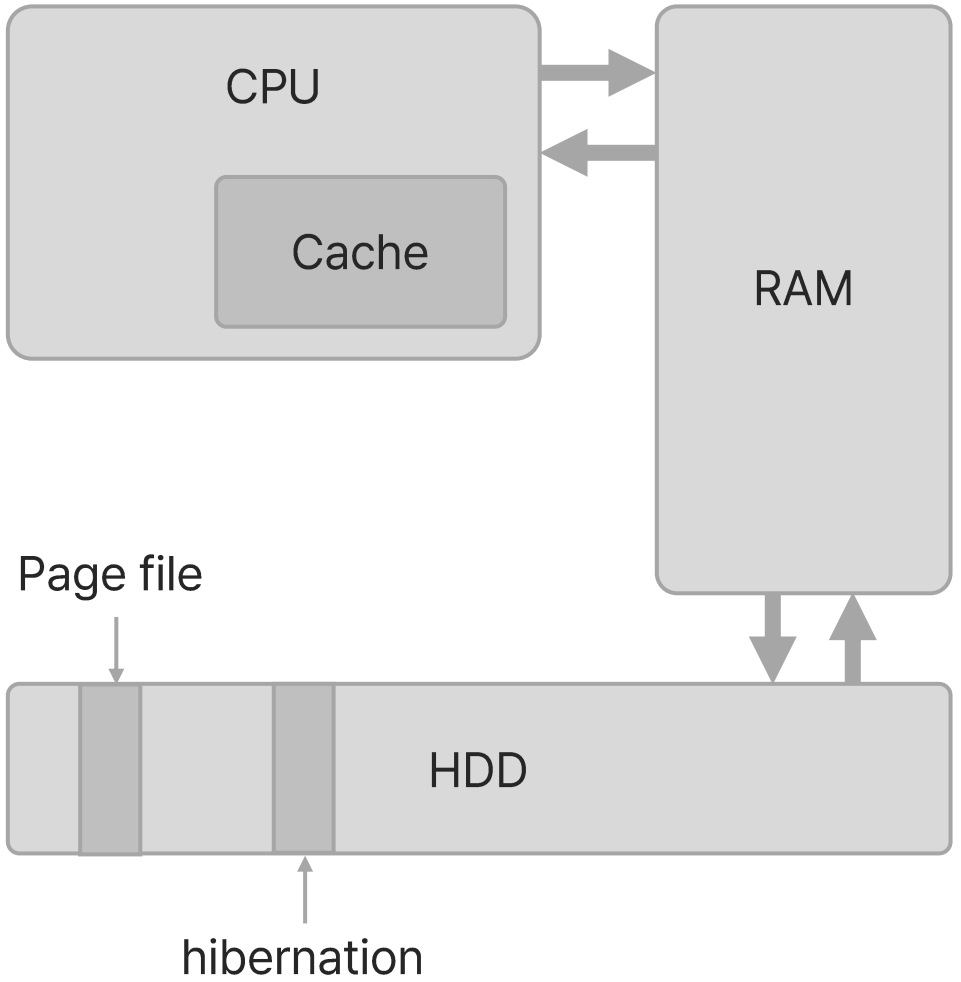
02

휘발성 데이터

휘발성 데이터에 대한 이해 및 획득 방법

휘발성 데이터

- 휘발성 데이터 개요
  - 프로세스 실행 과정
    - RAM
    - Page File
    - Hibernation



휘발성 데이터

- 휘발성 데이터 개요
  - RAM
    - 휘발성 메모리 (전원이 꺼지면 데이터가 사라짐)
    - 현재 실행 중인 프로그램과 데이터를 저장하여 CPU에 직접 접근

휘발성 데이터

- 휘발성 데이터 개요
  - Page File
    - 가상 메모리의 일환으로, RAM이 부족할 때 사용되는 하드 드라이브의 파일
    - 사용 목적
      - Win32 시스템의 모든 프로세스는 4GB의 연속된 주소 공간 사용

        → 물리메모리 용량이 4GB인 시스템에서 다수의 프로세스가 동작하는 이유는?

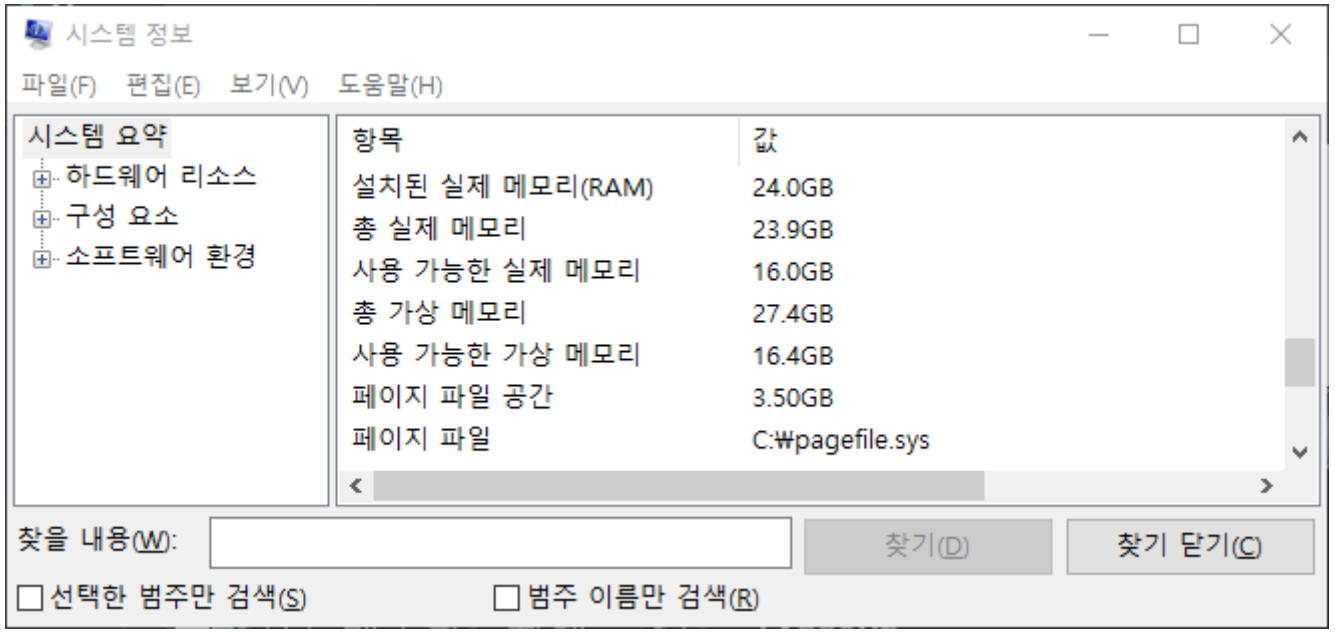
        → 가상 메모리 기법으로 대용량 메모리를 사용하는 프로그램 수행 가능

휘발성 데이터

- 휘발성 데이터 개요
  - Hibernation
    - 컴퓨터 절전 상태 돌입시, 하드 드라이브에 저장되는 시스템의 현재 상태
    - 저장 예시 (Windows)
      ① 컴퓨터 종료/절전 시, 물리 메모리를 압축해 C:hiberfil.sys 저장
      ② 부팅 시, hiberfil.sys가 설정
      ③ NTLDR에 의해서 메모리로 로드
      ④ 이전 상태로 부팅

휘발성 데이터

- 미니 실습
  - 내 메모리 확인하기



휘발성 데이터

- 휘발성 데이터 획득 방법

하드웨어	소프트웨어	크래시
절전모드	콜드부트	가상화 시스템

휘발성 데이터

- 휘발성 데이터 획득 방법
  - 최근 메모리 덤프 방식
    - 하드웨어 방식은 미리 설치되어 있어야 하거나 안정성 문제
    - 절전 모드, 크래시 덤프는 제한된 환경에서만 동작

→ 주로 소프트웨어 방식을 이용해 메모리 덤프

휘발성 데이터

- 휘발성 데이터 획득 방법
  - 소프트웨어를 이용한 덤프
    - 해당 운영체제를 지원하는 소프트웨어를 사용하여 다운로드 가능

      (Windows의 경우 FTK Imager, DumpIt, Belkasoft Live RAM Capturer 등)
    - 장점 : 추가 하드웨어 장치 필요 없음, 오픈소스 및 프리웨어가 다양
    - 단점 : 커널 루트킷에 취약, 운영체제에 따른 제약, 수집하는 메모리에 흔적이 생길 수 있음

휘발성 데이터

- 덤프 파일 확장자
  - .dmp
    - Windows 메모리 덤프 파일
  - .raw
    - 메모리 이미지 그 자체를 파일로 덤프시킨 포맷
  - .vmem
    - Vmware 에서 제공하는 포맷 (가상화 시스템의 메모리)
  - .img 등

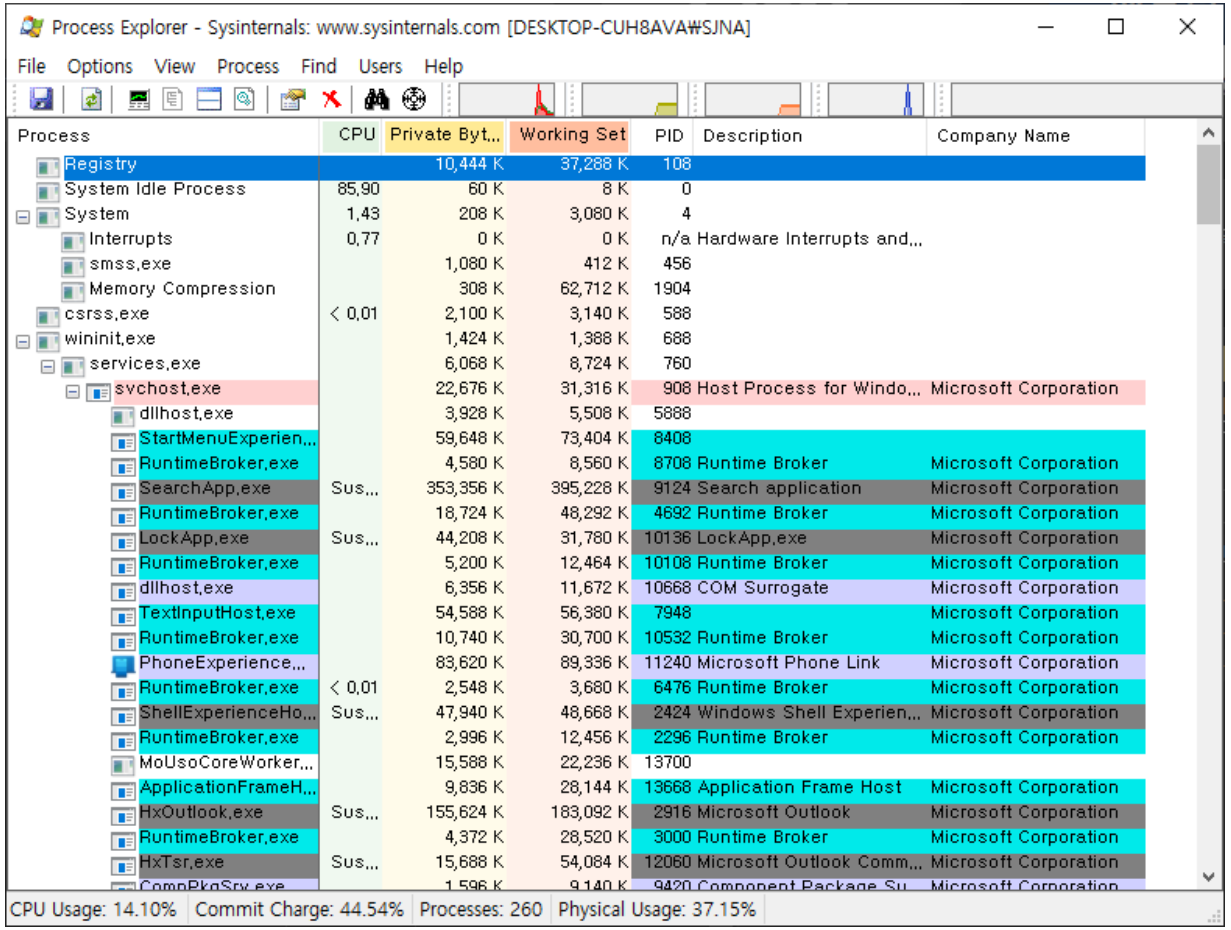
CHAPTER

03

Volatility

메모리 분석 방법 및 Volatility Tool에 대한 이해

- 활성 PC에서의 프로세스 분석
  - Process Explorer



- 메모리 분석 방법
  - 초기 메모리 분석 방법
    - 문자열 추출, 파일 카빙 등
    - 제한된 데이터만을 수집할 수 있는 한계가 존재
  - 변화된 메모리 분석 방법
    - **“물리메모리 상의 오브젝트를 찾기 위한 방법”**
    - 리스트 워킹 (List-Walking)
    - 패턴 매칭 (Pattern Matching)

Volatility

- 메모리 분석 도구

이름	인터페이스	플랫폼	제조사	라이선스
Redline™	GUI	Windows	Mandiant	Freeware
Volatility	CLI	Anywhere	Volatile Systems	Opensource
Responder Pro	GUI	Windows	HBGary	Commercial
Second Look® Linux Memory Analysis	CLI	Linux	Raytheon Pikewerks	Commercial
Volafox	CLI	Mac OS	n0fate	Opensource
Volafunx	CLI	FreeBSD	n0fate	Opensource

- 메모리 분석 도구

이름	XP	2003	Vista	2008	2008R2	7	2012	2012R2	8, 8.1
Redline™	32(SP2)	32(SP2) 64(SP2)	32	32 64	64	32 64	-	-	32 64
Volatility	32(SP2,3) 64(SP1,2)	32(SP0,1,2) 64(SP1,2)	32(SP0,1,2) 64(SP0,1,2)	32(SP1,2) 64(SP1,2)	64(SP0,1)	32(SP0,1) 64(SP0,1)	64	64	32 64
Responder Pro	32 64	32 64	32 64	32 64	32 64	32 64	32 64		32 64

이름	Linux	Mac OSX
Redline™	-	-
Volatility	32(Kernel 2.6.11 – 3.5) 64(Kernel 2.6.11 – 3.5)	32(10.5.x, 10.6.x, 10.7.x, 10.8.x, 10.9.x) 64(10.7.x, 10.8.x, 10.9.x)
Responder Pro	-	-

Volatility

- Volatility
  - CLI 기반 메모리 덤프 분석 도구
  - Windows 및 Linux, Android, macOS 환경 지원
  - Python 기반 스크립트 → Python 설치 필수
  - Python2 지원 종료에 따라 Python3를 이용한 Volatility3 개발
    - 플러그인은 v2가 더 많지만, v3는 ISF를 이용하여 편의성 ↑

Volatility

- Volatility
  - Volatility2 “Profile” → Volatility3 “ISF”
    - Profile
      - 특정 운영체제와 하드웨어 구조에 대한 VType, 오버레이, 객체 클래스의 집합
      - Memory Dump 분석 시, Profile을 직접 지정해야하는 불편함
    - ISF(Image Symbol Format)
      - 커널의 심볼 정보들이 등록되어있는 테이블(심볼 테이블)을 이용
      - Memory Dump 분석 시, ISF를 자동으로 인식(특정 경로에 위치만 하면 인식)하여 편리

Volatility

- Volatility 명령어
  - 기본 명령어 형태
    - python vol.py -f {file} {command}
  - 실행 결과 txt 파일로 추출
    - python vol.py -f {file} {command} > {file name}.txt
    - python vol.py -f {file} {command} --output-file= {file name}.txt

Volatility

- Volatility 명령어
  - Imageinfo로 기본 정보 수집
    - (vol3) windows.info / (vol2) imageinfo
  - Process information
    - 프로세스 리스트
      - windows.pslist / pslist
    - 프로세스 스캔
      - windows.psscan / psscan
    - 프로세스 트리
      - windows.pstree / pstree

Tip. 숨겨진 프로세스 확인

- pslist = True
- psscan = false

Volatility

- Volatility 명령어
  - Process information
    - 프로세스 덤프
      - windows. memmaps / memdump
      - vol3.py –c config.json –f {file} –o {dump file path} windows.memmaps –pid XX
      - vol2.py –f {file} –profile {profile} memdump –p <PID> --dump-dir={dump file path}

Volatility

- Volatility 명령어
  - Network 분석
    - netscan : 활성화 상태의 네트워크 연결 정보 (Window7 이상)
    - netstat : 현재 실행 중인 프로세스 네트워크 정보
    - connections : 활성화 상태의 네트워크 연결 정보 (Windows XP/Vista)
    - connscan : 활성/비활성 네트워크 연결 정보
    - sokets : 네트워크 연결 정보 확인
  - DLL, Thread 분석
    - dlllist : 특정 프로세스에서 로드한 DLL의 정보 분석
    - ldrmodules : 은폐된 DLL 정보 분석
    - dlldump : 특정 프로세스에서 로드한 DLL을 바이너리 형태로 추출
    - malfind : 사용자 모드 형태로 은폐되어 있거나 인젝션 된 코드 또는 DLL 정보 분석

Volatility

- Volatility 명령어
  - File 분석
    - filescan : 메모리에 로드된 파일 정보 스캔, 특정 확장자 및 파일 정보 찾기
      - volatility -f {file} --profile={profile} filescan | findstr ".jpg" (Window)
      - volatility -f {file} --profile={profile} filescan | grep “.jpg” (Linux)
    - dumpfiles : 메모리에서 검색한 파일 복구
    - memdump : 메모리에 존재하는 프로세스 복구 (.dmp 파일 추출)
    - procdump : 해당 프로세스의 실행파일(.exe)를 덤프
    - hashdump : NTLM 비밀번호 해시 추출 (유저 정보, 마지막 값이 비밀번호 해시)

      → NTLM 복호화 사이트가 존재

CHAPTER

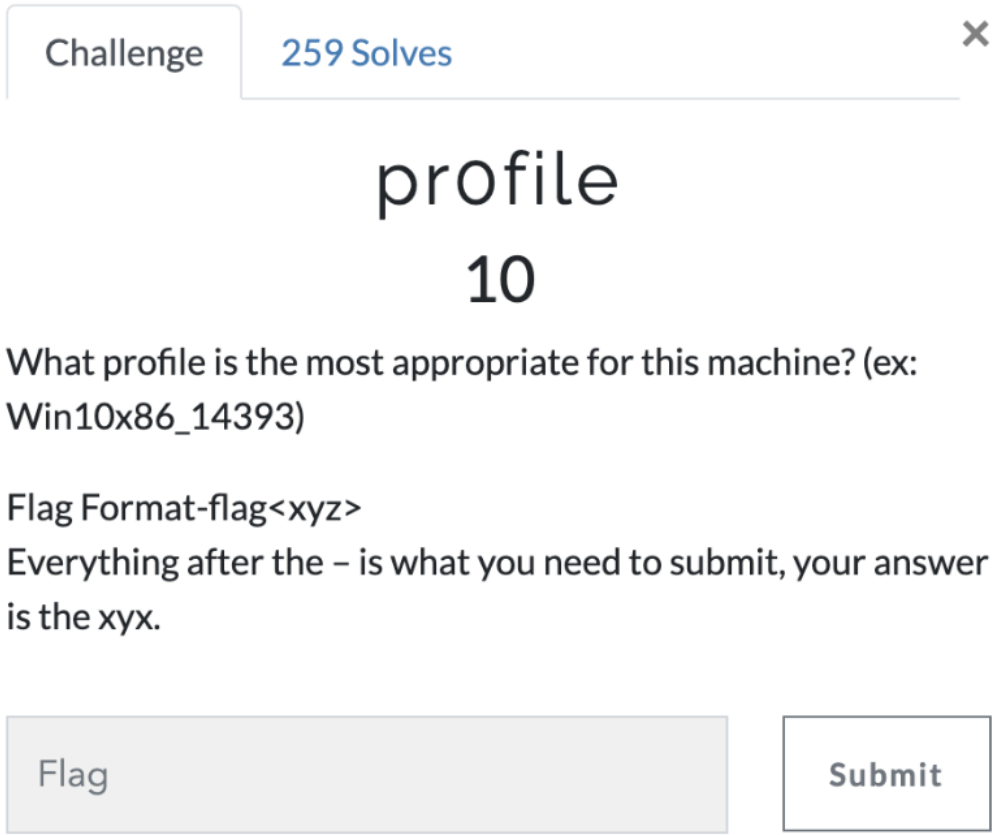
04

메모리 포렌식 with DEFCON

실제 메모리 분석하기 근데 데프콘을 곁들인...

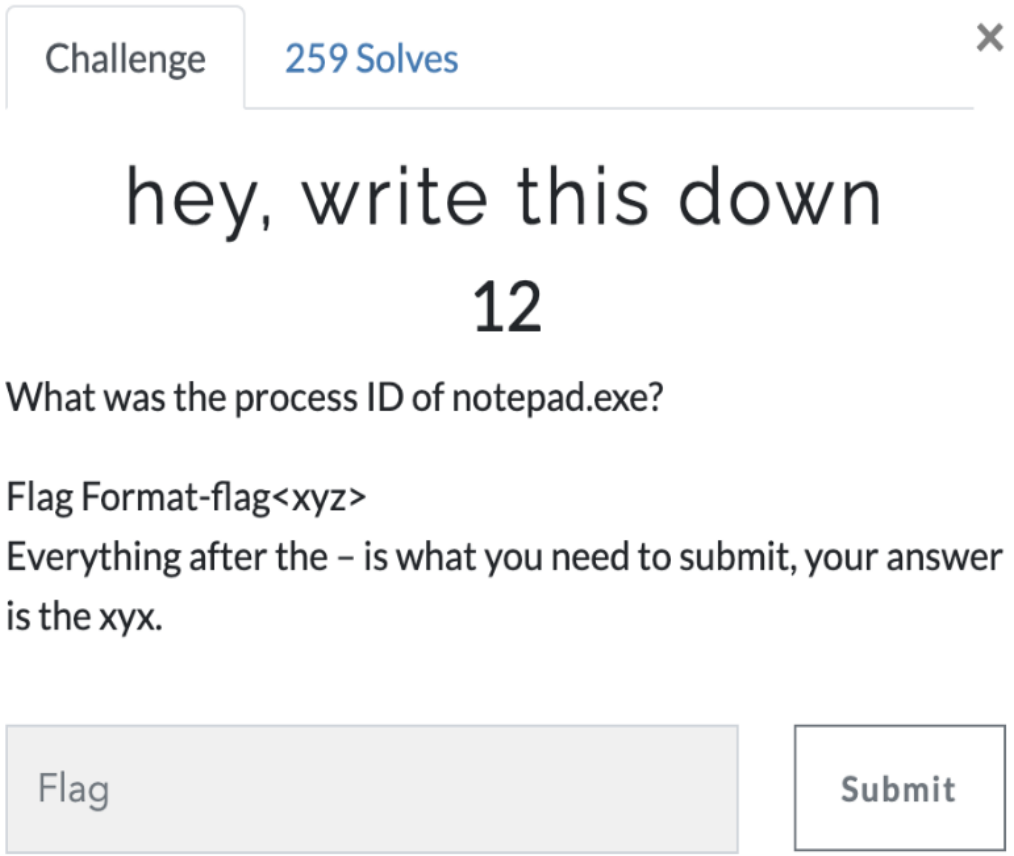
실제 메모리 분석하기 근데 데프콘을 곁들인...

메모리 포렌식 with DEFCON



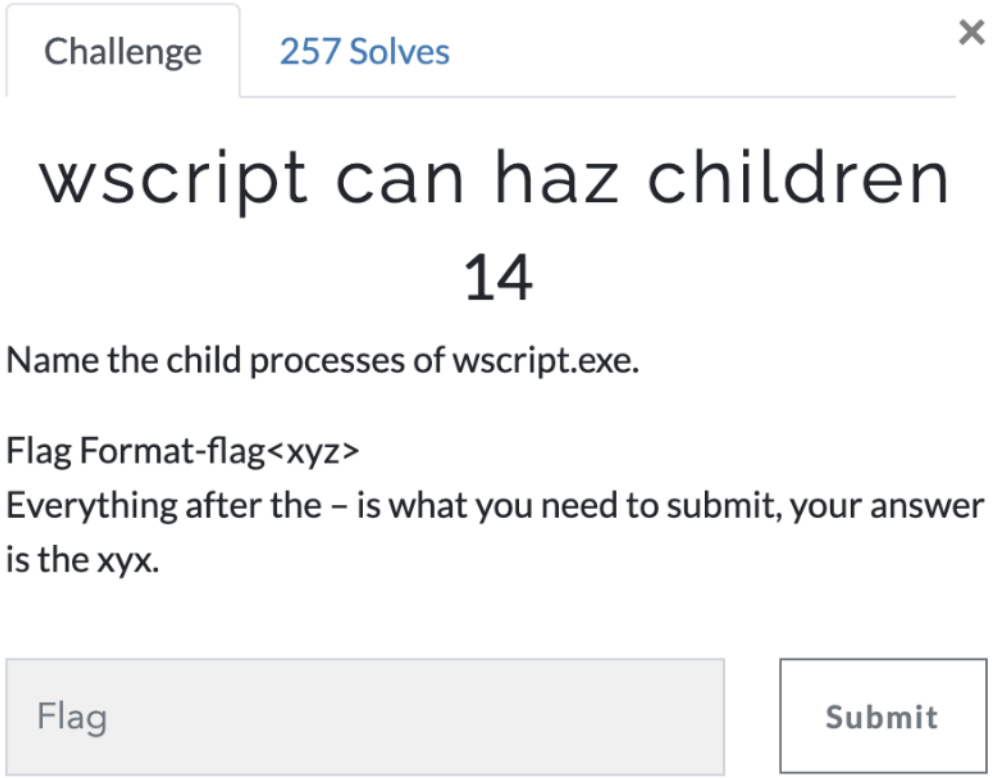
실제 메모리 분석하기 근데 데프콘을 곁들인...

메모리 포렌식 with DEFCON



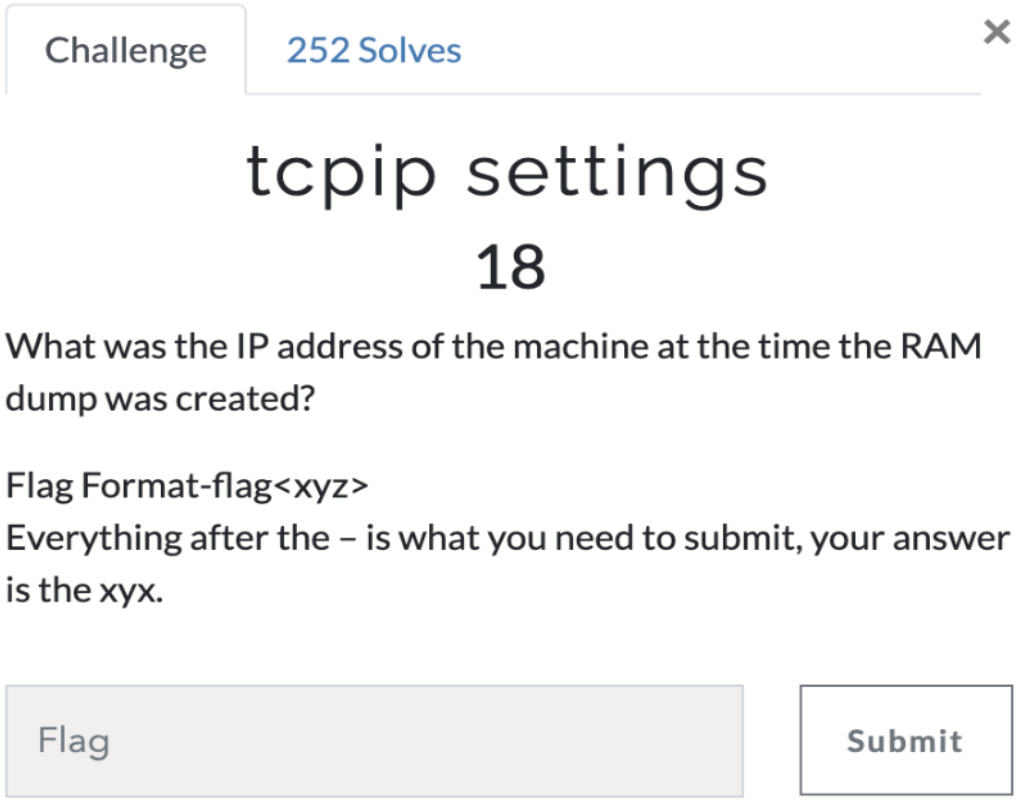
실제 메모리 분석하기 근데 데프콘을 곁들인...

메모리 포렌식 with DEFCON



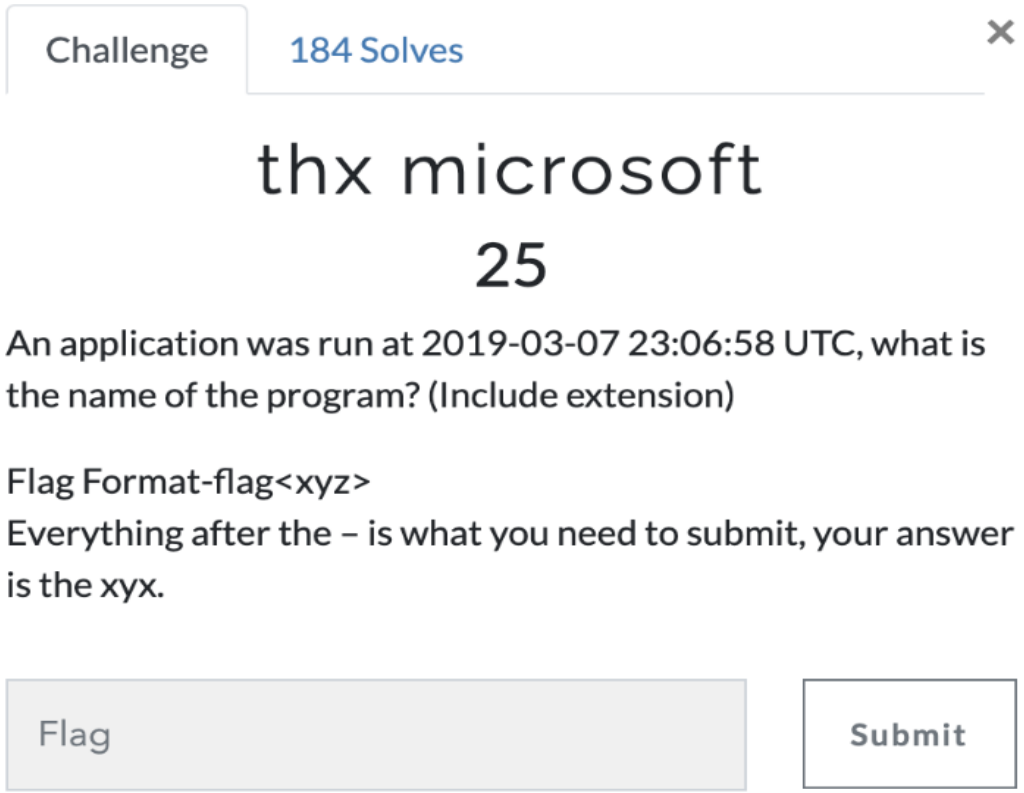
실제 메모리 분석하기 근데 데프콘을 곁들인...

메모리 포렌식 with DEFCON

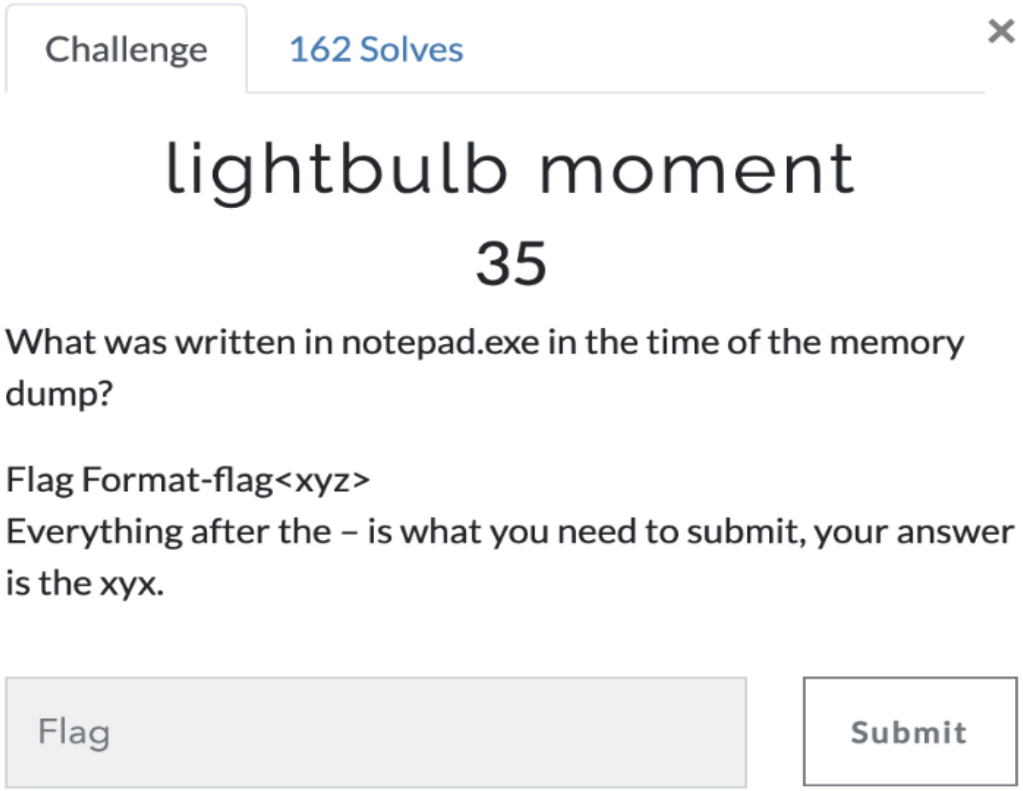


실제 메모리 분석하기 근데 데프콘을 곁들인...

메모리 포렌식 with DEFCON



메모리 포렌식 with DEFCON



Reference

1. Forensic Proof, http://forensic-proof.com/
2. 디지털 포렌식 A-Z, 충남대학교
3. [DEFCON DFIR CTF 2019] Memory Forensics, https://information-security.tistory.com/469