

DFC 2020 402 Hacker's memory 분석 보고서 (각색)

디지털포렌식 김태룡

본문 요약

1. 사건 개요

회사 A에 대한 사이버 공격이 급증하였으며, 이를 내부자 소행으로 의심한 감사팀은 용의자B를 급습하여 PC 메모리 덤프를 추출하였고, 분석팀이 해당 메모리 덤프를 인계받았습니다.

2. 파일 정보

① 파일 정보

순번	파일 명	크기	SHA256
1	compressed_memory.lime	1.26GB	032C55AA0556F4FB903D59C6D17E880D0625301A46DC857A42808248F7D118E5
2	packet.pcap	16.2MB	017F2EE16AF74846B9C0BFEC496DA8BC46B3EC75D13B2A9400DF0ED8661594DB

3. 분석 결과

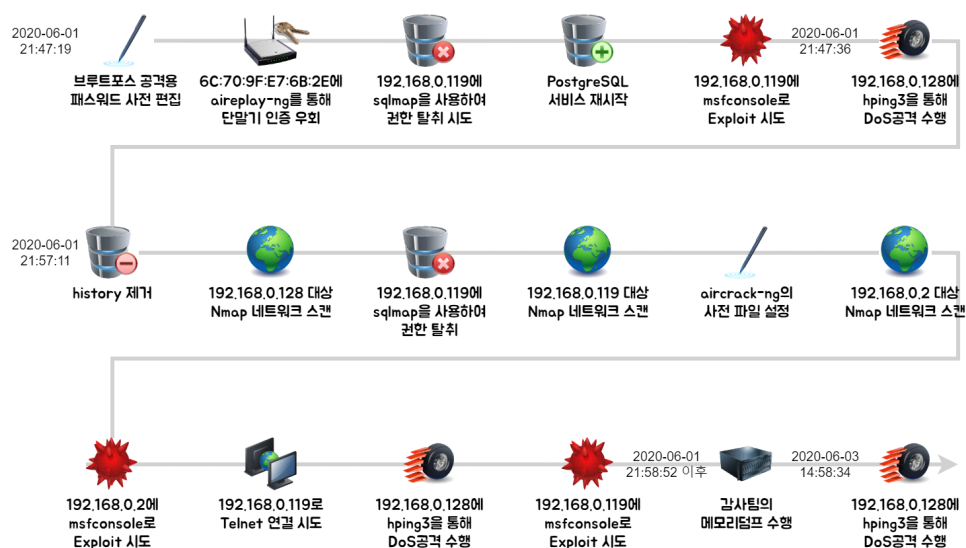
① compressed_memory.lime

✓ sqlmap, msfconsole, hping3 등을 활용한 악성행위 수행 정황을 확인하였습니다

② packet.pcap

✓ 감사 이후에도 추가 공격을 수행한 정황을 확인하였습니다

③ 공격 타임라인



4. 결론

- 회사 A의 용의자 B는 sqlmap, msfconsole, hping3를 이용하여 내부 IP 192.168.0.2, 192.168.0.119, 192.168.0.128에 대한 공격을 감행하였으며, 감사팀의 메모리 덤프 이후에도 공격을 지속하였습니다.

✓ 용의자 B는 회사 A 내부망에 대한 공격을 수행하였습니다.

1. 개요

1-1) 사건 개요

회사 A에 대한 사이버 공격이 급증하였으며, 이를 내부자 소행으로 의심한 감사팀은 용의자B를 급습하여 PC 메모리 덤프를 추출하였고, 분석팀이 해당 메모리 덤프를 인계받았습니다.

1-2) 분석 환경

항목	값	항목	값
OS	Microsoft Windows 10 Home Edition	시스템 종류	64비트 운영 체제
프로세서	Intel(R) Core(TM) i7-1065G7	RAM	16 GB
OS	KaliLinux 5.7.0	시스템 종류	64비트 운영 체제
프로세서	가상환경	RAM	4GB
OS	Ubuntu 20.04 LTS	시스템 종류	64비트 운영 체제
프로세서	가상환경	RAM	4GB

[표 1-1] 분석 환경

1-3) 분석 대상 파일

순번	파일 명	크기	SHA256
1	compressed_memory.lime	1.26GB	7DCF40274C80176EF31336267CE39B26B6B2D493DA9C789B03EFAC4B23146615
2	packet.pcap	16.2MB	032C55AA0556F4FB903D59C6D17E880D0625301A46DC857A42808248F7D118E5

[표 1-2] 분석 대상

1-4) 분석 도구

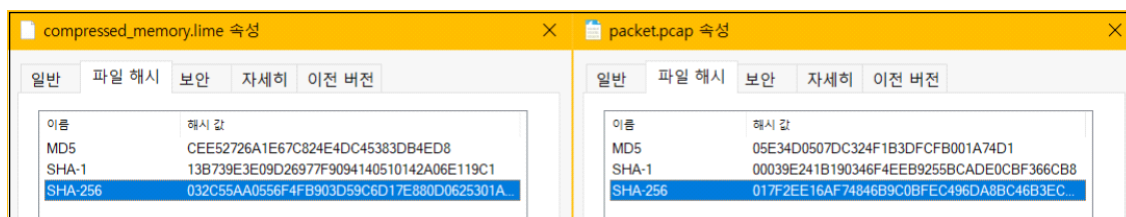
도구	버전	용도	제공
AVML	0.2.1	Linux 메모리 분석 도구	Microsoft https://github.com/microsoft/avml
HashTab	6.0.0	파일 해시 검증	Implbits http://implbits.com/products/hashtab/
HxD	2.4.0.0(x86-64)	파일 Hex 확인 및 편집	mh-nexus https://mh-nexus.de/en/hxd/
Volatility	2.6.1	메모리 덤프 분석	Volatilityfoundation https://github.com/volatilityfoundation/volatility
WireShark	3.4.2	pcap 파일 분석	WireShark https://www.wireshark.org/

2. 메모리 프로파일 생성

2-1) 파일 정보 확인

대상	파일 명	크기	SHA256
현장	compressed_memory.lime	1.26GB	032C55AA0556F4FB903D59C6D17E880D0625301A46DC857A42808248F7D118E5
	packet.pcap	16.2MB	017F2EE16AF74846B9C0BFEC496DA8BC46B3EC75D13B2A9400DF0ED8661594DB
검증	compressed_memory.lime	1.26GB	032C55AA0556F4FB903D59C6D17E880D0625301A46DC857A42808248F7D118E5
	packet.pcap	16.2MB	017F2EE16AF74846B9C0BFEC496DA8BC46B3EC75D13B2A9400DF0ED8661594DB

[표 2-1] 채증 당시 현장 기록과 조사 과정에서 추출된 기록 비교자료



[그림 2-1] Hash Tab을 통한 채증 파일 해시 값 비교 화면

✓ 현장 기록과 채증 파일이 동일함을 확인 하였습니다

2-2) 메모리 덤프 파일 분석

- 메모리 덤프의 파일 시그니처는 AVML으로, AVML 도구를 이용하여 추출된 압축 메모리 덤프였음을 확인하였습니다.

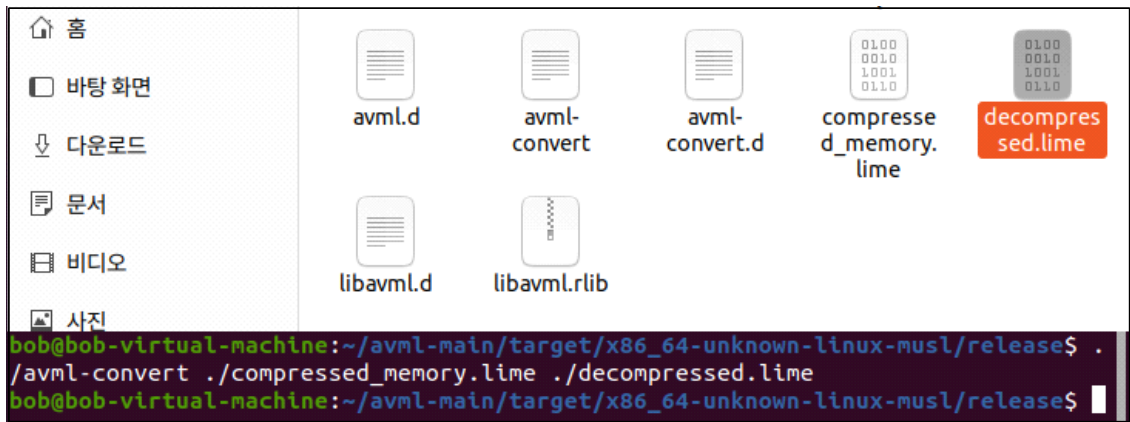
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	41	56	4D	4C	02	00	00	00	00	10	00	00	00	00	00	00	AVML.
00000010	FF	CF	09	00	00	00	00	00	00	00	00	00	00	00	00	00	ÿI.....
00000020	FF	06	00	00	73	4E	61	50	70	59	00	00	79	00	DB	74	ÿ...sNaPpY..y.Ût
00000030	2F	0F	80	80	04	5C	89	C6	05	09	00	00	00	FF	E0	8D	/..€€..\\%E.....ÿà.
00000040	86	48	00	00	00	89	86	40	00	00	00	8D	86	B0	05	0C	tH...%t@....t°..

[그림 2-2] AVML 시그니처

- 분석 Ubuntu 가상환경에 AVML 소스코드를 다운로드 받고나서 musl, rust를 설치하여 빌드 하였으며, 이후 AVML을 이용하여 압축된 메모리 덤프의 압축 해제에 성공하였습니다.

파일 명	크기	SHA256
decompressed.lime	1.74GB	6ADF0046D9302DB9D658DE0C101245FB41360C902A1BB377E9DAE4CACCF3A269

[표 2-2] 압축 해제된 메모리 덤프 정보



[그림 2-3] 압축 해제된 메모리 덤프

- 감사팀으로부터 메모리 덤프 대상에 대한 정보를 얻지 못하여 strings를 통해 OS 및 버전 정보를 확인하였습니다.

```
bob@bob-virtual-machine:~/avml-main/target/x86_64-unknown-linux-musl/release$ strings ./decompressed.lime | grep "Windows version"
bob@bob-virtual-machine:~/avml-main/target/x86_64-unknown-linux-musl/release$ strings ./decompressed.lime | grep "Linux version"
@of Linux version
@of Linux version
@of Linux version
Linux version 5.6.0-kali1-amd64 (devel@kali.org) (gcc version 9.3.0 (Debian 9.3.0-11)) #1 SMP Debian 5.6.7-1kali1 (2020-05-12)
Linux version 5.6.0-kali1-amd64 (devel@kali.org) (gcc version 9.3.0 (Debian 9.3.0-11)) #1 SMP Debian 5.6.7-1kali1 (2020-05-12)
Linux version %s (%s)
MESSAGE=Linux version 5.6.0-kali1-amd64 (devel@kali.org) (gcc version 9.3.0 (Debian 9.3.0-11)) #1 SMP Debian 5.6.7-1kali1 (2020-05-12)
Linux version %s (%s)
Linux version %s (%s)
Linux version %s (%s)
bob@bob-virtual-machine:~/avml-main/target/x86_64-unknown-linux-musl/release$
```

[그림 2-4] OS 및 버전 정보

항목	값	항목	값
OS	Kali Linux (5.6.0-kali-amd64)	저장장치	Toshiba 2.5" HDD 80GB
CPU	Intel(R) Pentium(R) Dual CPU T2390@1.86GHz	RAM	2 GB

[표 2-2] 메모리 덤프 대상 정보

✓ 채증 파일은 Kali Linux 대상 AVML 도구로 수집된 압축 메모리 덤프임을 확인하였습니다

2-2) 커널 버전 조정

- 메모리 분석을 위하여 덤프 대상과 동일한 프로파일을 만들고자 우선 Kali Linux 홈페이지로부터 5.6.0버전과 가장 유사한 이미지¹⁾를 다운로드 받고 설치하였습니다.

```
goldbigdragon@kali:~$ uname -a
Linux kali 5.7.0-kali1-amd64 #1 SMP Debian 5.7.6-1kali2 (2020-07-01) x86_64 GNU/Linux
```

[그림 2-5] 5.7.0 버전이 설치 된 모습

(인터넷을 끈 상태로 설치해야만 커널 자동 업데이트가 이루어지지 않습니다)

- 설치 이후 apt-get install 명령어로 linux-headers-5.6.0-kali1-amd64에 해당하는 헤더 파일을 다운로드 시도하였으나, 연결된 아카이브 페이지에 존재하지 않아 다운로드에 실패하였습니다.

1) 유사 버전 KaliLinux 이미지 : <http://old.kali.org/kali-images/kali-2020.2/>

```
goldbigdragon@kali:/mirrors$ sudo apt install linux-headers-5.6.0-kali1-amd64
[sudo] password for goldbigdragon:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package linux-headers-5.6.0-kali1-amd64
E: Couldn't find any package by glob 'linux-headers-5.6.0-kali1-amd64'
```

[그림 2-5] 헤더 다운로드에 실패한 모습

- 이는 KaliLinux의 기본 아카이브 페이지에 해당 버전의 헤더 정보가 없기 때문이므로, 추가적인 KaliLinux 아카이브 주소를 /etc/apt/sources.list에 작성하였습니다.

```
sudo vi /etc/apt/sources.list
```

/etc/apt/sources.list

```
deb http://http.kali.org/ /kali main contrib non-free
deb http://http.kali.org/ /wheezy main contrib non-free
deb http://http.kali.org/kali kali-dev main contrib non-free
deb http://http.kali.org/kali kali-dev main/debian-installer
deb-src http://http.kali.org/kali kali-dev main contrib non-free
deb http://http.kali.org/kali kali main contrib non-free
deb http://http.kali.org/kali kali main/debian-installer
deb-src http://http.kali.org/kali kali main contrib non-free
deb http://security.kali.org/kali-security kali/updates main contrib non-free
deb-src http://security.kali.org/kali-security kali/updates main contrib non-free
deb http://repo.kali.org/kali kali-bleeding-edge main
```

(입력 이후 저장 및 종료를 원할 경우, [ESC] → wq! → [Enter] 입력)

```
goldbigdragon@kali: ~
# See https://www.kali.org/docs/general-use/kali-linux-sources-list-repositories/
deb http://http.kali.org/kali kali-rolling main contrib non-free
deb http://http.kali.org/ /kali main contrib non-free
deb http://http.kali.org/ /wheezy main contrib non-free
deb http://http.kali.org/kali kali-dev main contrib non-free
deb http://http.kali.org/kali kali-dev main/debian-installer
deb-src http://http.kali.org/kali kali-dev main contrib non-free
deb http://http.kali.org/kali kali main contrib non-free
deb http://http.kali.org/kali kali main/debian-installer
deb-src http://http.kali.org/kali kali main contrib non-free
deb http://security.kali.org/kali-security kali/updates main contrib non-free
deb-src http://security.kali.org/kali-security kali/updates main contrib non-free
deb http://repo.kali.org/kali kali-bleeding-edge main
# Additional line for source packages
# deb-src http://http.kali.org/kali kali-rolling main contrib non-free
~
```

[그림 2-6] 아카이브 주소를 등록한 모습

- 아카이브 주소 등록 후 apt update를 진행시켜주고, 5.6.0 버전 커널과 이미지를 다운로드 받았습니다.

```
sudo apt-get update
```

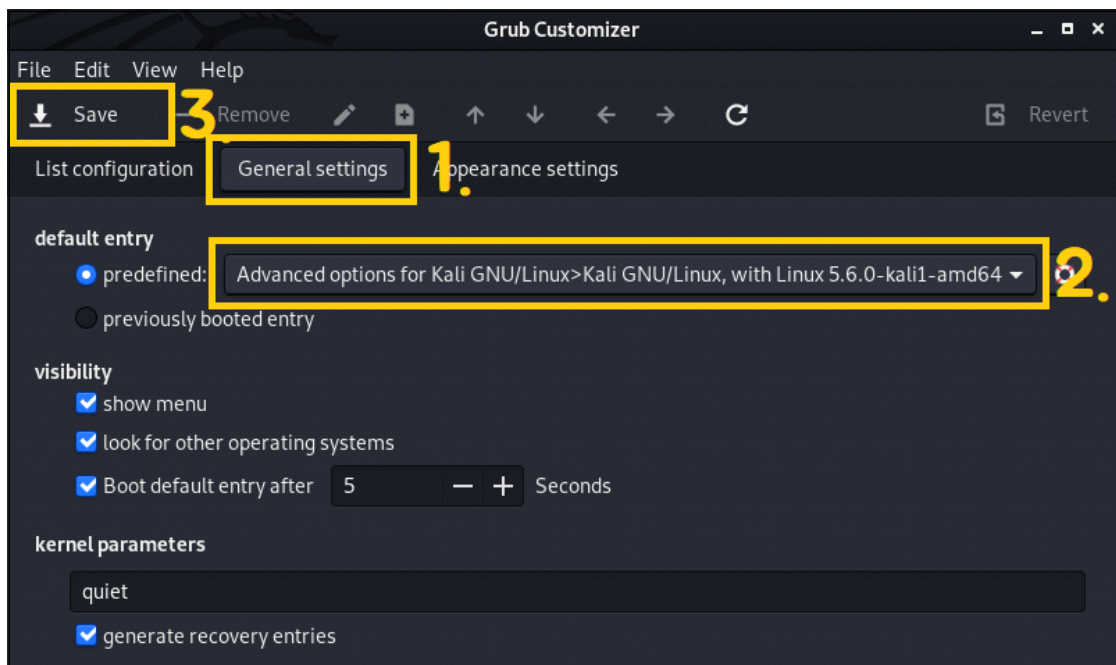
```
sudo apt-get install linux-headers-5.6.0-kali1-amd64
```

```
sudo apt-get install linux-image-5.6.0-kali1-amd64
```

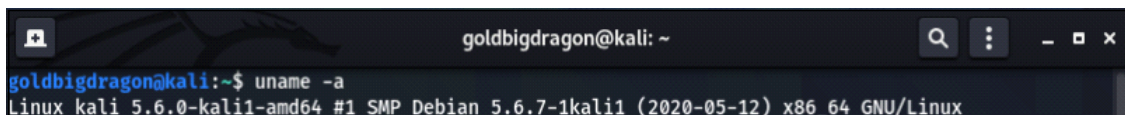
- 이후 5.6.0 버전 KaliLinux로 부팅하기 위해 부트로더 설정 프로그램²⁾을 설치하고, 재부팅 시 5.6.0 버전 KaliLinux를 구동하도록 설정 이후 재부팅하였습니다.

```
sudo apt install grub-customizer
```

```
grub-customizer
```



[그림 2-7] 재부팅 시 5.6.0 KaliLinux로 시작하도록 설정 한 모습



[그림 2-8] 재부팅 이후 5.6.0버전 KaliLinux로 구동된 모습

2) 부트로더 종류 : LILO, GRUB

LILO(Linux LOader) : Linux 운영체제에 한정되어 사용되는 경우.

GRUB(GRand Unified Bootloader) : 리눅스 이외 다양한 운영체제에 사용 가능한 경우.

2-3) KaliLinux 5.6.0 프로파일 생성

- 프로파일 생성을 위해 Volatility와 dwarfdump를 설치하였습니다.

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

```
sudo apt-get install dwarfdump
```

- 설치 완료 이후 /volatility/tools/linux에 위치한 리눅스 심볼 추출 코드를 사용하여 프로파일 생성의 필수 요소 중 하나인 dwarf 파일을 생성하였습니다.

```
cd ./volatility/tools/linux
```

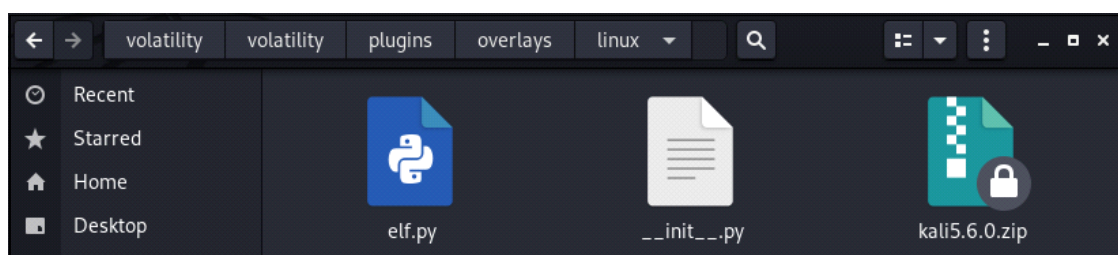
```
make
```

```
goldbigdragon@kali:~/volatility/tools/linux$ ls
kcore Makefile Makefile.enterprise module.c module.dwarf
```

[그림 2-9] 생성된 module.dwarf 파일

- dwarf파일 추출 이후 /boot에 위치한 System.map파일과 함께 압축하여 KaliLinux 5.6.0에 대한 프로파일을, 프로파일을 모아두는 plugins/overlays/linux 디렉터리에 생성하였습니다.

```
sudo zip ../../volatility/plugins/overlays/linux/kali5.6.0.zip ./module.dwarf
/boot/System.map-5.6.0-kali1-amd64
```



[그림 2-10] KaliLinux 5.6.0에 대한 프로파일이 생성된 모습

```
python --info
```

```
Profiles
-----
Linuxkali5_6_0x64 - A Profile for Linux kali5.6.0 x64
VistaSP0x64 - A Profile for Windows Vista SP0 x64
VistaSP0x86 - A Profile for Windows Vista SP0 x86
```

[그림 2-11] KaliLinux 5.6.0 프로파일이 성공적으로 등록된 모습

3. 메모리 분석

3-1) 실행 프로세스 확인

- 압축 해제된 메모리 덤프 파일을 volatility 홈 디렉터리에 옮긴 다음, 생성된 KaliLinux 5.6.0 프로파일을 이용하여 수행 중이던 프로세스를 확인하였으며, pslist 기능으로도 탐지되지 않는 숨겨진 프로세스 혹은 강제 주입식 공격에 의해 실행 된 프로그램까지 확인하기 위해 psxview 옵션을 추가로 사용하였습니다.

```
python vol.py -f decompressed.lime --profile=Linuxkali5_6_0x64 linux_pstree
```

```
python vol.py -f decompressed.lime --profile=Linuxkali5_6_0x64 linux_pslist
```

```
python vol.py -f decompressed.lime --profile=Linuxkali5_6_0x64 linux_psxview
```

Offset(V)	Name	PID	pslist	psscan	pid_hash	kmem_cache	parents	leaders
INFO : volatility.debug : SLUB is currently unsupported.								
0x00000000fd7d400	systemd	1	True	True	True	False	True	True
0x00000000fd7c600	kthreadd	2	True	True	True	False	True	True
0x00000000fd7aa00	rcu_gp	3	True	True	True	False	False	True
0x00000000fd7f000	rcu_par_gp	4	True	True	True	False	False	True
0x00000000fd78e00	kworker/0:0H	6	True	True	True	False	False	True
0x00000000fd78000	mm_percpu_wq	8	True	True	True	False	False	True
0x00000000fd7b800	ksoftirqd/0	9	True	True	True	False	False	True
0x00000000fd7b600	rcu_sched	10	True	True	True	False	False	True

[그림 3-1] 프로세스를 확인 한 모습

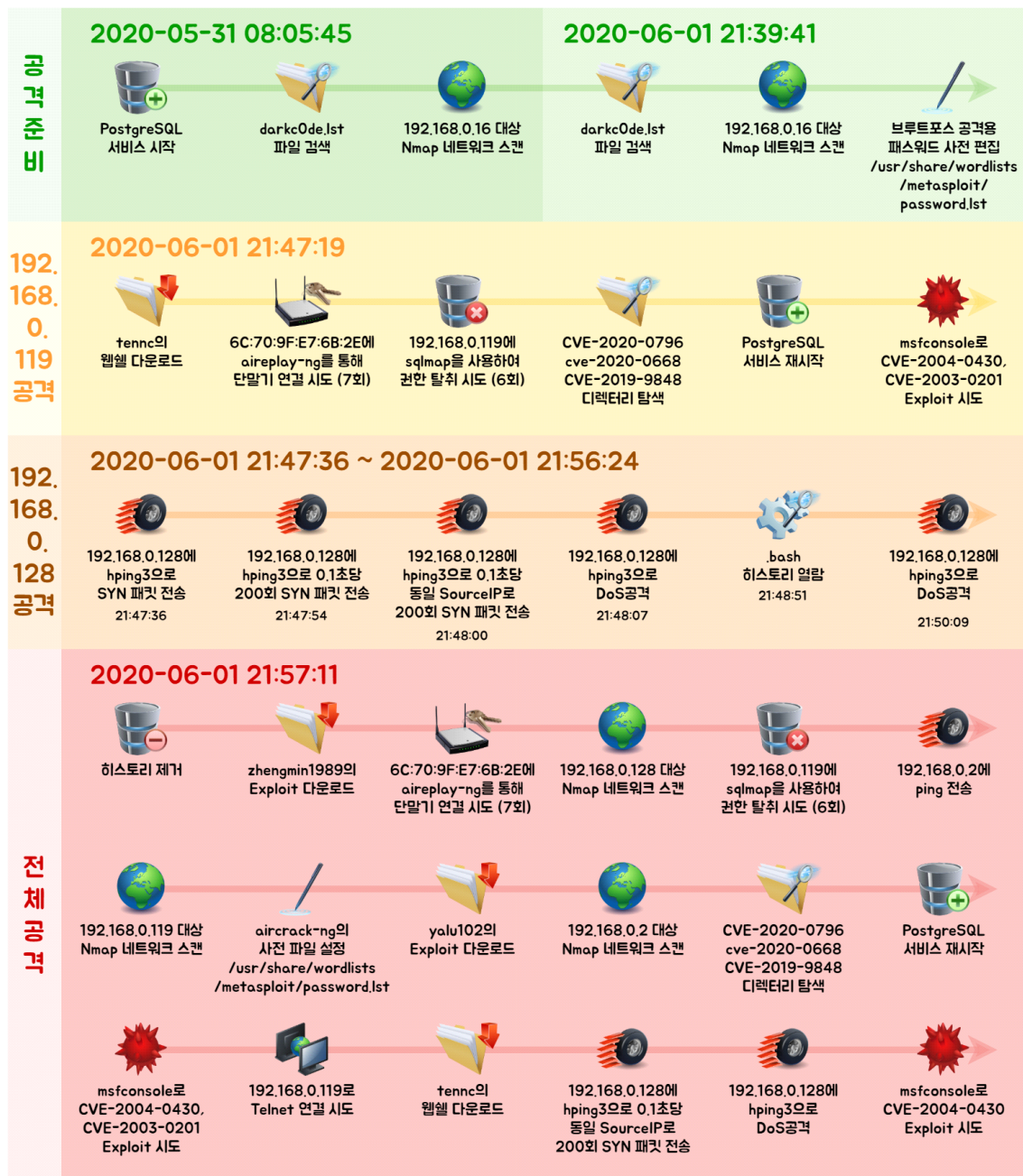
3-2) 입력 명령어 확인

- 수상한 프로세스를 발견하지 못하였으므로 프로세스 수행 직후 상황일수도 있었습니다. 이에 사용자 명령어 사용 내역을 확인하였습니다.

```
python vol.py -f decompressed.lime --profile=Linuxkali5_6_0x64 linux_bash
```

Pid	Name	Command	Time	Command
3554	bash	cd ..	2020-05-31 23:05:45 UTC+0000	
3554	bash	ls	2020-05-31 23:05:45 UTC+0000	
3554	bash	python vol.py --conf-file=../capture/kali-2020-05-16_12.17.36/volatilityrc linux_banner	2020-05-31 23:05:45 UTC+0000	
3554	bash	cp volatilityrc volatilityrc1	2020-05-31 23:05:45 UTC+0000	
3554	bash	su - root	2020-05-31 23:05:45 UTC+0000	

[그림 3-2] 명령 사용기록을 확인 한 모습



[그림 3-3] 주요 명령 사용기록 타임라인

- 내부자 B가 192.168.0.2, 192.168.0.119, 192.168.0.128 대상으로 hping3, msfconsole, Nmap, aircrack-ng, sqlmap 등을 사용한 공격을 진행한 정황을 확인할 수 있었으며, 간간히 외부 오픈소스 Exploit 도구 및 웹 셸 다운로드 정황을 추가로 확보할 수 있었습니다.

대상	hping3	telnet	netcat	Nmap	msfconsole	aircrack-ng	sqlmap
192.168.0.2				21:47:19	21:47:19		
192.168.0.119		21:57:11	21:57:11	21:57:11	21:47:19	21:47:19	21:57:11
192.168.0.128	21:47:36			21:57:11			

[표 3-1] 대상에 대한 공격시각 및 사용된 도구

- 또한 msfconsole 공격 직전 PostgreSQL을 재시작하는 패턴을 통해 내부자 B는 Exploit 결과를

내부 PostgreSQL DB에 저장하는 점을 알 수 있었습니다.

- 하지만 sqlmap 사용 시에는 --dbms=mysql 옵션을 넣음으로 써 mysql 데이터베이스에 탈취한 정보를 입력하는 모습을 확인 할 수 있었습니다.

```
8759 bash 2020-06-01 12:57:11 UTC+0000 sqlmap -u "http://192.168.0.119/admin/index.php" --dbms=mysql --data "user=USER&password=PASS"
-p user
8759 bash 2020-06-01 12:57:11 UTC+0000 exit
8759 bash 2020-06-01 12:57:11 UTC+0000 telnet 192.168.0.119 80
8759 bash 2020-06-01 12:57:11 UTC+0000 sqlmap -u "http://192.168.0.119/sso.php?id=1" --dbms=mysql --dbs
8759 bash 2020-06-01 12:57:11 UTC+0000 sqlmap -u "http://192.168.0.119/sso.php?id=1" --dbms=mysql -D information_schema --tables
8759 bash 2020-06-01 12:57:11 UTC+0000 aireplay-ng --deauth 100 -a 6C:70:9F:E7:6B:2E wlan0mon
8759 bash 2020-06-01 12:57:11 UTC+0000 simple-command
8759 bash 2020-06-01 12:57:11 UTC+0000 sqlmap -u "http://192.168.0.119/admin/index.php" --dbms=mysql --data "user=USER&password=PASS"
```

[그림 3-4] sqlmap 사용 시 mysql 정보를 입력하는 모습

- 다양한 Exploit 시도가 확인되었으나, 명령어를 통해 실질적으로 사용된 종류는 2종이며, 두 종류 모두 MAC OS 취약점을 이용하는 점을 통해 공격 대상이 MAC OS를 사용한다는 점을 알 수 있었습니다.

CVE	대상	사용 여부
CVE-2003-0201	MAC OS	msfconsole를 통해 사용됨
CVE-2020-0796	SMB	확인되지 않음
CVE-2020-0668	Windows Kernel	확인되지 않음
CVD-2019-9848	LibreOffice	확인되지 않음
CVE-2004-0430	Apple File Server	msfconsole를 통해 사용됨

[표 3-2] 사용된 CVE 목록

✓ 악성 행위 수행 정황을 확인하였습니다

3-3) 사용자 IP 확인

- 명령 사용기록 타임라인을 통해 내부자 B가 악성 행위를 수행한 정황을 확인하였습니다. 용의자 B는 내부 네트워크망을 이용하여 공격을 시도하였기에, 이미지로부터 네트워크 연결 정보를 확인하였습니다.

```
python vol.py -f decompressed.lime --profile=Linuxkali5_6_0x64 linux_ifconfig
```

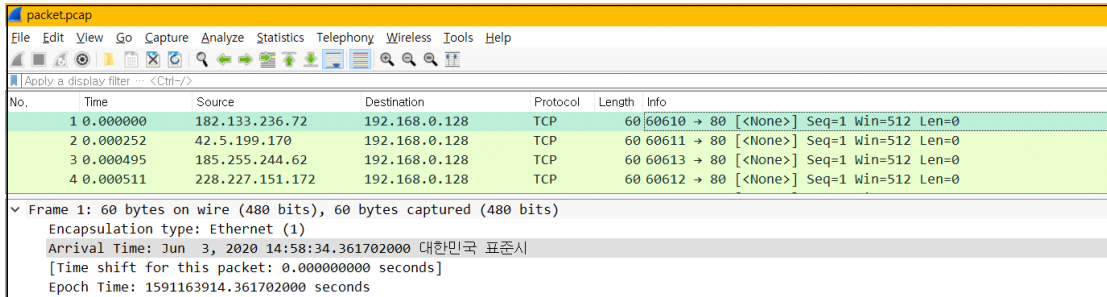
Interface	IP Address	MAC Address	Promiscuous Mode
lo	127.0.0.1	00:00:00:00:00:00	False
eth0	222.111.61.249	00:13:77:9f:fe:7b	False
wlan0	192.168.0.63	00:21:63:56:4f:8b	False
lo	127.0.0.1	00:00:00:00:00:00	False
lo	127.0.0.1	00:00:00:00:00:00	False

[그림 3-5] 인터페이스별 할당된 IP

4. pcap파일 분석

4-1) 공격 시간 확인

- pcap 파일에 기록 된 첫 패킷과 끝 패킷의 도달 시각을 확인 해 본 결과, 2020-06-03 14:58:34에 시작하여 2020-06-03 15:00:34에 끝나는 모습을 통해 감사팀이 용의자 B의 PC 메모리를 덤프 한 이후에 추가적으로 수행된 공격임을 알 수 있었습니다.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	182.133.236.72	192.168.0.128	TCP	60	60610 → 80 [<None>] Seq=1 Win=512 Len=0
2	0.000252	42.5.199.170	192.168.0.128	TCP	60	60611 → 80 [<None>] Seq=1 Win=512 Len=0
3	0.000495	185.255.244.62	192.168.0.128	TCP	60	60613 → 80 [<None>] Seq=1 Win=512 Len=0
4	0.000511	228.227.151.172	192.168.0.128	TCP	60	60612 → 80 [<None>] Seq=1 Win=512 Len=0

Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Encapsulation type: Ethernet (1)
Arrival Time: Jun 3, 2020 14:58:34.361702000 대한민국 표준시
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1591163914.361702000 seconds

[그림 4-1] 메모리 덤프 이후 수행된 공격 정황

✓ 용의자 B는 감사 이후에도 추가 공격을 수행한 정황을 확인하였습니다

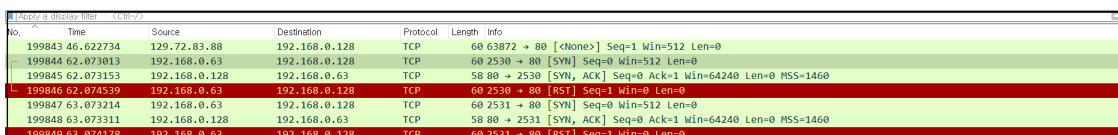
4-2) 공격 기법 확인

- 앞서 내부자 B의 IP주소와 hping3을 통한 공격 정황을 확인하였으므로, 이를 바탕으로 pcap파일을 분석하였습니다.

메모리 덤프에서 발견된 hping3 명령어	
hping3 -S 192.168.0.128 -p 80	
hping3 192.168.0.128 -S -i u100 -c 200	
hping3 192.168.0.128 -a 192.168.0.128 -S -i u100 -c 200 -p 80	
hping3 --flood --rand-source 192.168.0.128 -p 80	

[표 4-1] 사용된 hping3 명령어

- hping3의 -S 옵션은 SYN 플래그를 활성화 시킨 패킷을 전송하는 형태이며, -i는 전송 간격, -c는 전송 횟수로, 0.1초당 200개의 SYN 패킷을 전송하는 공격을 의미하였습니다. 이를 찾기 위하여 공격자의 IP인 192.168.0.63을 SourceIP로 하는 SYN 플래그 패킷을 확인하였습니다.

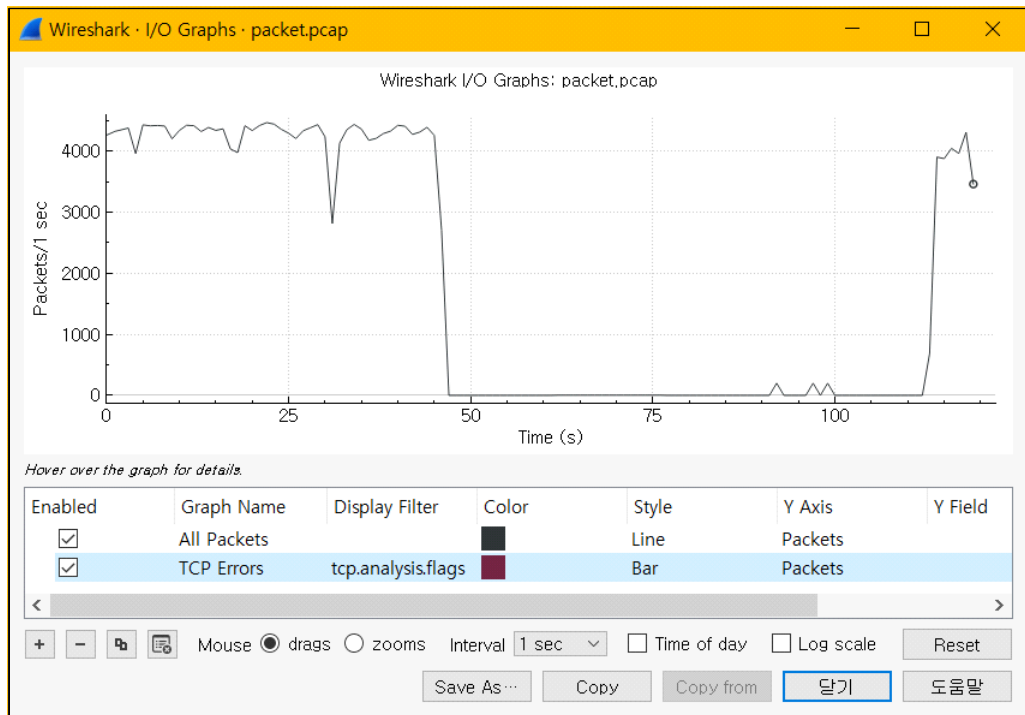


No.	Time	Source	Destination	Protocol	Length	Info
199843	46.622734	129.72.83.88	192.168.0.128	TCP	60	63872 → 80 [<None>] Seq=1 Win=512 Len=0
199844	62.073013	192.168.0.63	192.168.0.128	TCP	60	2530 → 80 [SYN] Seq=0 Win=512 Len=0
199845	62.073153	192.168.0.128	192.168.0.63	TCP	58	80 → 2530 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
199846	62.074539	192.168.0.63	192.168.0.128	TCP	60	2530 → 80 [RST] Seq=1 Win=0 Len=0
199847	63.073214	192.168.0.63	192.168.0.128	TCP	60	2531 → 80 [SYN] Seq=0 Win=512 Len=0
199848	63.073311	192.168.0.128	192.168.0.63	TCP	58	80 → 2531 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
199849	63.074178	192.168.0.63	192.168.0.128	TCP	60	2531 → 80 [RST] Seq=1 Win=0 Len=0

[그림 4-2] SYN Flood 공격 정황을 확인 한 모습

- hping3의 --flood 옵션은 패킷 전송 속도 제한을 없애는 것으로, DoS를 유발하며, --rand-surce는 임의 IP주소를 송신IP 주소에 작성하여 전송하는 것으로, pcap 파일의 I/O

Graph와 각 SourceIP 주소를 확인하여 DoS 공격이 수행되었는지 확인하였습니다.



[그림 4-3] DoS 공격 정황을 확인한 모습

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	182.133.236.72	192.168.0.128	TCP	60	60610 → 80 [None] Seq=1 Win=512 Len=0
2	0.000252	42.5.199.170	192.168.0.128	TCP	60	60611 → 80 [None] Seq=1 Win=512 Len=0
3	0.000495	185.255.244.62	192.168.0.128	TCP	60	60613 → 80 [None] Seq=1 Win=512 Len=0
4	0.000511	228.227.151.172	192.168.0.128	TCP	60	60612 → 80 [None] Seq=1 Win=512 Len=0
5	0.000844	5.175.237.59	192.168.0.128	TCP	60	60614 → 80 [None] Seq=1 Win=512 Len=0
6	0.001091	111.215.139.110	192.168.0.128	TCP	60	60615 → 80 [None] Seq=1 Win=512 Len=0
7	0.001347	113.58.251.51	192.168.0.128	TCP	60	60616 → 80 [None] Seq=1 Win=512 Len=0
8	0.001603	245.97.237.27	192.168.0.128	TCP	60	60618 → 80 [None] Seq=1 Win=512 Len=0
9	0.001620	144.116.237.200	192.168.0.128	TCP	60	60617 → 80 [None] Seq=1 Win=512 Len=0

[그림 4-4] 임의 IP 설정 정황을 확인한 모습

- 하지만 모두 임의 IP주소라고 좀비PC를 이용한 공격이 아니라고 단정 지을 수 없었기에 Endpoint를 확인 하였으며, 그 결과 AskeyCom 사의 공유기와 Raspberr 기기만을 거친 모습을 확인하였습니다.
- 피해 PC가 DMZ 설정이나 포트포워딩이 되어있지 않은 평범한 사내 PC이므로 이는 모두 내부 소행이며, hping3를 통한 공격 전력이 있으며, 동시에 pcap파일에 SYN Flood 공격 시 IP 주소가 드러났던 용의자 B의 행위임을 확인하였습니다.

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
AskeyCom_564f8b	224,719	13M	224,704	13M	15	870
Raspberr_5cb1f3	224,719	13M	15	870	224,704	13M

Buttons: Copy, Map, 달기, 도움말

[그림 4-5] 모든 패킷이 공유기 및 Raspberry 기기만을 거친 모습

- 마지막으로 -a 192.168.0.128 는 전송지 IP를 192.168.0.128로 지정하는 것으로, 잘못된 패킷을 보낼 경우, 대상자에게 반드시 답장을 보내는 통신구조의 취약점을 악용한 Land Attack이며, 수신지와 발신지 주소가 같기에 지속적으로 본인에게 응답 패킷을 보내도록 하여 DoS를 유발하는 공격이었습니다.

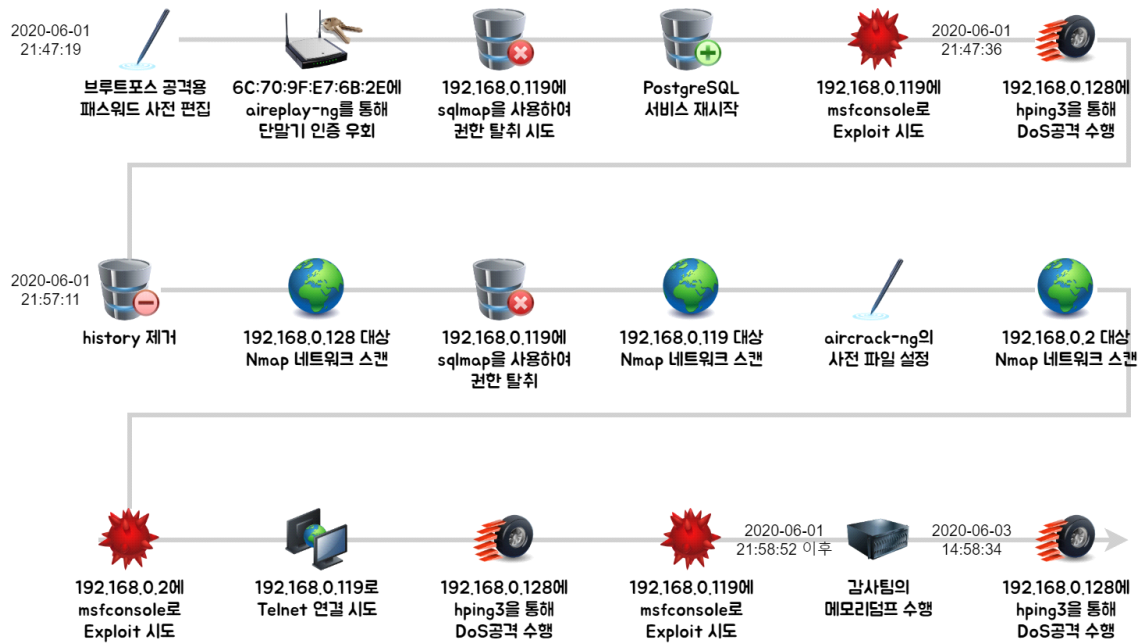
Wireshark · Conversations · packet.pcap											
Ethernet · 1		IPv4 · 224005		IPv6		TCP · 224689		UDP			
Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
192.168.0.63	192.168.0.128	45	2670	30	1800	15	870	62.073013	14.0047	1028	496
192.168.0.128	192.168.0.128	600	36k	600	36k	0	0	92.750306	6.4913	44k	0
192.168.0.128	192.169.46.47	1	60	0	0	1	60	0.391353	0.0000	—	—
192.168.0.128	192.169.11...	1	60	0	0	1	60	29.155904	0.0000	—	—

[그림 4-6] Land Attack 정황을 확인 한 모습

✓ 용의자 B는 SYN Flood, DoS, Land Attack 공격을 진행하였습니다

5. 결론

5-1) 전체 타임라인



[그림 5-1] 주요 타임라인

5-2) 결론

- 회사 A의 용의자 B는 sqlmap, msfconsole, hping3을 이용하여 내부 IP 192.168.0.2, 192.168.0.119, 192.168.0.128에 대한 공격을 감행하였으며, 감사팀의 메모리 덤프 이후에도 공격을 지속하였습니다.

✓ 용의자 B는 회사 A 내부망에 대한 공격을 수행하였습니다.