금 상

이승진 (세종대학교)

요약 보고서

- 침해 서버 분석

(0) 해킹 당한 시스템과 루트킷
(1) 어떻게 들어왔는가?
(2) Jacob 은 누구인가?
(3) 대응 방법

- 악성 코드 분석

(0) 웜 개요
(1) 웜 프로세스의 시작
(2) -1 실행된 적이 있다면 -2 실행된 적이 없다면
(3) Winlogonm.dll 과 winsystemm.exe 은 악성 파일
(4) 레지스트 등록
(5) 직접적인 전파 방법
(6) DOS 공격
(7) 간접적인 전파 방법
(8) Winlogonm.dll 의 특징
(9) 얼마나 악영향을 미치는가?
(10) 대응 방법
(11) 분석 방법

문제 1. 침해사고 분석

침해 서버의 OS : Linux kernel 2.4.18-4 (lilo.conf 에서 확인할 수 있다.)

(0) 해킹 당한 시스템과 루트킷

현재 서버는 해킹을 당한 상태이다. /var/log 를 조사해보면 아래의 정보를 볼 수 있다.

./messages:Jun su(pam_unix)[9147]: session opened for user root by jacob(uid=500)
./messages:Jun su(pam_unix)[983]: session opened for user root by jacob(uid=500)

Jacob 이란 id 가 root 가 되었는 데, Jacob 이 운영자가 사용하는 일반 계정이라면 위 로그는 별로 의심거리가 없는 것이지만, 서버를 조사해보면 rootkit 이 여러 군데 깔려져 있는 것으로 보아, 이 서버는 해킹을 당했다고 볼 수 있다.

해커는 다음 디렉토리에 ark 라는 백도어를 심어놓았다. 백도어는 이 곳 뿐만 아니라 다른 디렉토리에도 있지만 거의 모두 이 안에 있는 것과 관련이 되어 있는 것들이다. 모두 조사하기에는 분량이 너무 많아지므로 간단하게 알아보겠다.

/mnt/usr/lib/librk/rootkit

-rwxr-xr-x	1	0	root	15380 Jul 31	2002 bindshell
-rwxr-xr-x	1	0	root	8092 Jul 31	2002 cgiback.cgi
-rwxr-xr-x	1	0	root	603 Jul 31	2002 hideit
-rwxr-xr-x	1	0	root	352 Jul 31	2002 install
-rwxr-xr-x	1	0	root	9368 Jul 31	2002 logclean
-rwxr-xr-x	1	0	root	184023 Jul 31	2002 ls
-rwxr-xr-x	1	0	root	258612 Jul 31	2002 netstat
-rwxr-xr-x	1	0	root	47388 Jul 31	2002 ps
-rwxr-xr-x	1	0	root	6872 Jul 31	2002 sniffer
-rwxr-xr-x	1	0	root	11028 Jul 31	2002 targets
-rwxr-xr-x	1	0	root	817052 Jul 31	2002 x3

이 폴더에 있는 바이너리들은 특수한 기능들을 수행한다. 먼저 ls, ps, netstat 는 /bin/ 디렉토리에 위치한 바이너리들과 바뀌게 되고, bindshell 과 sniffer 는 각각 /sbin/bshell 과 /sbin/srload 가 된다. 이 두 파일은 /etc/rc.d/rc.sysinit 에 등록되어 시스템이 부팅될 때 마다

실행될 수 있게 설정을 해놓는다. 그리고 Web 을 이용하여 Root 권한을 행사할 수 있는 cgiback.cgi backdoor 는 웹 디렉토리인 /var/www/cgi-bin 에 위치하게 됩니다.

각 백도어 바이너리의 특징 소개

Ls – 본래 ls 는 디렉토리 리스트나 파일의 정보 등을 볼 때 쓰이는 유틸리티다. 그런데 해커의 이 rootkit ls 는 해커가 원하는 특정 문자열을 제거하여 출력해준다. /dev/ptyxx/.file 에서 그 목록을 읽는다.

Logclean – 파일 이름에서 알 수 있듯이 로그를 지워주는 기능을 한다.

Netstat – netstat 는 서버의 네트워크 정보를 출력해주는 프로그램이다. 루트킷 netstat 는 특정 문자열이 존재할 경우 그 것을 제외하고 출력한다. 여기서 문자열은 주로 해커의 IP 가 될 것이다. 해커의 IP 는 /dev/ptyxx/.addr 에 저장되어있다.

Ps – ps 는 서버에서 돌아가고 있는 프로세스 목록들을 보여준다. 이 루트킷 ps 역시 특정 문자열이 존재할 경우 그 것을 출력하지 않는다. /dev/ptyxx/.proc 에서 감출 목록을 읽는다.

Sniffer – sniffer 는 network 를 감시할 수 있는 프로그램이다. 로그는 .snfflogsk 에 남게 된다.

다음 파일들은 rootkit이 운영되는데 필요한 파일들이다.

/dev/ptyxx/.file – rootkit이 디렉토리나 파일의 문자열을 보여주지 않으려 할 때 참고
/dev/ptyxx/.proc – rootkit이 프로세스를 보여주지 않으려 할 때 참고
/dev/ptyxx/.addr – rootkit이 네트워크 주소를 보여주지 않으려 할 때 참고

cgiback.cgi 에 대해서 조금만 더 자세히 알아보겠다.

- cgiback.cgi Backdoor 의 특징 -

Backdoor 는 해커가 타겟 서버에서 Root 권한을 획득하고, 나중에 다시 침입할 때 편하게 하기 위해서 만들어 놓는 것이 보통이다. Cgiback.cgi 백도어는 Web 기반에서 돌아가는 백도어 프로그램이며, 해커는 이를 설치하기 위해서 다음과 같은 작업을 해야 한다.

```
# gcc cgiback.c -o cgiback.cgi –lcrypt
# chown root.root cgiback.cgi
# chmod 4755 cgiback.cgi
```

위와 같은 작업을 하고, cgiback.cgi 를, Web 에서 실행시킬 수 있는 곳으로 이동시키면 백도어 설치가 끝나는 간단한 프로그램이다. 설치가 끝나면, 해커는 이 것을 이용하여 여러

가지 기능을 수행할 수 있다. 예를 들어 /etc/password 파일에 uid 가 0 을 갖는 ID 를 추가할 수도 있고, 웹에서 직접 커맨드를 내릴 수도 있다. 모두 Web 에서 가능하다.

(1) 어떻게 들어왔는가?

현재 서버에는 해커가 어떻게 들어왔는지 정확하게 파악할 수 있는 단서가 많지 않다. 그저 추측만 할 수 있다. 먼저 다음과 같이 두 가지로 나누어 생각을 해 볼 수 있다. '계정이 없는 상태에서 루트를 따냈거나', '계정을 갖고 루트를 따냈거나'

계정이 없는 상태라면 시스템 데몬의 취약점을 이용하여 들어왔을 확률이 크다. (물론 그 이외에도 여러 가지 방법이 더 존재한다.) 서버에 깔려있는 데몬에는 어떤 것들이 있는지 알아보자. 서버에 깔려져 있는 패키지의 정보는 /var/log/rpmpkgs 에 담겨 있다. 조사 결과 주요 데몬으로, openssh-1.2.2-1, apache-1.3.23-11, sendmail-8.12.5-5 이 있었다. 주의 깊게 봐야할 점은 openssh-1.2.2 이다. 현재 서버의 kernel 버전은 2.4.18 인데 비하여 낮은 버전에 속한다. 이 openssh-1.2.2 버전은 취약점이 존재하고, 인터넷에 이미 exploit 이 나와있는 상태이다. 이를 이용한다면 remote 에서 Shell 을 획득할 수도 있다. 이 것은 하나의 가능성일 뿐이며 다른 방법도 더 존재할 수 있다.

계정이 있는 상태라면 훨씬 더 수월하다. 커널 버전이 2.4.18 라면 리눅스 커널 버그가 여러 개 존재하기 때문에 로컬에서 root 를 얻을 수 있다.

(2) Jacob 은 누구인가?

나는 해킹 당한 시스템을 분석하는 것을 좋아하지 않는다. 침해 분석하는 것은 해킹보다 재미없다. 그러나 한번 분석에 빠지기 시작하면 해킹을 하는 것만큼이나 재미있어한다. 분석을 하다가 얼핏 이런 생각이 들었다. "Jacob 이 누구야!!??"

서버에는 Jacob 이라는 일반 계정 하나만 존재한다. 그런데, 이 계정은 운영자인가, 공격자인가? 지금 상황으로는 어느 쪽이 확실하다고 단정 지을 수가 없다. Jacob 은 운영자가 아니라 해커가 사용한 ID 라고 가정해보자. 그렇다면, 해커는 root 권한이 있고, rootkit 을 설치할 실력이 있음에도 불구하고 왜 /var/log/messages 에 Jacob 이 root 가 된 흔적들을 남긴 것인가? 또한 Jacob Home Directory 에 있는 rpm 파일들은, 모두 root 권한으로 소유되어 있고, 이 rpm 파일들의 버전들은 /var/log/rpmpkgs 의 내용과 동일하다. 즉 운영자가 Jacob 의 ID 로 작업한 것일수도 있다.

반대로 해커는 Jacob 이라고 가정하자. Jacob 은 의외로 해킹이 능숙하지 못하고, 아니면 지금 이 환경이 대회를 위한 것이기 때문에 문제 출제자가 의도적으로 해커의 흔적을 남긴 것일수도 있다.

내 의견을 말해보자면, Jacob 은 운영자도 사용을 하고, 해커도 사용을 한다. 지금 상황만 보면 Jacob 은 운영자라고 볼 수 있다. 그렇지만 한번 다음의 흔적을 보자. 현재

/dev/ptyxx/.addr 에는 해커의 IP 를 속이기 위한 것이 데이터가 들어가있다. 즉, 이 안에 있는 IP 주소는 해커의 IP 라고 볼 수 있는 것이다. .addr 을 참조해보면 현재 해커의 IP 는 192.168.131.136 로 되어 있다. 그러나 다음과 같은 한 줄의 커맨드로 우리는 해커가 Jacob 으로 로그인 했다는 것을 알 수 있다.

```
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+21:43)
jacob    pts/0        192.168.131.1    Mon Jun  7 20:19 - down  (00:05)
root     tty1                          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     tty1                          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     tty1                          Mon Jun  7 17:04 - down  (00:42)
reboot   system boot  2.4.18-4         Mon Jun  7 17:03       (00:43)
```

만약 운영자, 해커 모두가 Jacob 을 사용한 것이 맞다면, 의문점은 풀리게 된다. 그러나 이것도 짐작일 뿐이다. 중요한 것은 해커가 어떻게 들어왔느냐인데, 이 것 역시 (1) 에서 설명했듯이, 짐작만이 가능할 뿐이다. 어쩌면 다른 곳에 완벽한 증거가 있을지도 모르겠다..

(3) 대응 방법

이미 해킹을 당했으므로, 완벽한 복구를 하는 것은 힘들다. 중요한 데이터는 모두 백업하고 (바이너리는 해커에게 재이용될 수 있으므로 가급적 백업을 하지 않는다.) OS 를 새로 설치 한다.

서버의 커널과 데몬들을 최신 버전으로 유지한다.

패스워드를 모두 바꾸고 패스워드 보안에 주의한다.

불필요한 서비스들을 제거하고, 가능하다면 Firewall 같은 보안 도구를 사용하여 서버를 안전하게 보호한다.

문제 2. 악성코드(웜) 분석

(0) 웜 개요

악성 코드 파일 이름 : sisworm.exe

먼저 웜은 UPX 로 실행 압축 되어있다. 변형되거나 하지 않은 순수 UPX 압축 파일이다. UPX 를 이용하여 웜을 만들면 코드가 압축되어진 형태이기 때문에, 웜 제작자는 웜 코드를 분석하기 어렵게 만들 수 있다. 공개 되어있는 UPX 프로그램을 이용하여 웜에 걸려져 있는 UPX 압축을 쉽게 해제할 수 있다. 먼저 압축을 푼 후에 분석을 시작해야한다. 압축을 풀면, 웜은 일반적인 WIN32 PE 임을 알 수 있다.

프로그램을 만들다 보면, 문자열을 처리해야 할 경우가 많다. 그런데 이 웜에서는 문자열을 이용할 때 암호화 된 문자열을 이용하였다. 암호화 방식은 시져 암호화 방식과 거의 유사한 단순한 알고리즘을 사용하고 있다. 아래 2 개의 소스로 웜에서 사용한 암호화된 문자열을 쉽게 풀 수 있다. (분석하는 시간이 촉박하여 만들지는 못했으나, 한 개의 소스로도 암호화된 문자열을 충분히 풀 수 있을 것으로 보인다.) 웜 제작자가 암호화된 문자열을 사용하여 프로그램을 만든 이유는 UPX 와 마찬가지로, 분석하기 힘들게 하기 위함이다. 이런 암호화 문자열이 많이 나오게 되는데, 앞으로 설명하는 글에서는 암호화된 문자열을 사용하지 않고, 복호화 문자열을 바로 사용할 것이다.

```
int main(int argc, char *argv[])
{
        char ori[]="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
        char enc[]="MNOPQRSTUVWXABCDEFGHIJKLYZmnopqrstuvwxabcdefghijklyz";
        int a, b, flag;
        for(a=0;a<strlen(argv[1]);a++)
        {
                for(b=0;b<strlen(enc);b++)
                {
                flag=1;
                        if(argv[1][a]==enc[b])
                        {
                                printf("%c", ori[b]);
                                flag=2;
                                break;
                        }
```

```
                }
                if(flag==1)
                {
                        printf("%c", argv[1][a]);
                }
        }
        printf("\n");
}
```

decode2.c

```
int main(int argc, char *argv[])
{
        char ori[]="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
        char enc[]="NOPQRSTUVWXYZABCDEFGHIJKLMnopqrstuvwxyzabcdefghijklm";
        int a, b, flag;
        for(a=0;a<strlen(argv[1]);a++)
        {
                for(b=0;b<strlen(enc);b++)
                {
                flag=1;
                        if(argv[1][a]==enc[b])
                        {
                                printf("%c", ori[b]);
                                flag=2;
                                break;
                        }
                }
                if(flag==1)
                {
                        printf("%c", argv[1][a]);
                }
        }
        printf("\n");
}
```

한 가지 예를 들어 이 웜에서는 “winlogonm.dll” 이라는 원래 문자열을 "kubxcscba.pxx" 와

같이 암호화를 하여서 사용을 하고 있다. 이 암호화된 문자열을 위 프로그램을 이용하여 다음과 같이 풀어 볼 수 있다. (다음은 리눅스 환경이다.)

```
$ gcc -o decode1 decode1.c
$ ./decode1 "kubxcscba.pxx"
winlogonm.dll
```

웜이란 것은 스스로 전파하는 것을 뜻한다. 이 웜에 구현되어 있는 웜의 전파 방법은 크게 다음과 같이 나눌 수 있다.

SMB IPC, P2P(kazaa), LSASS, backdoor, Mail

앞으로 하나 하나 알아볼 것인데, Mail 의 경우에는 프로그램을 리버싱 해본 결과, Mail 을 이용한 악성 코드 자체는 기능이 구현되어 있었으나, 프로그램에서 이를 호출하는 부분이 없는 것 같아서 자세하게 다루지 않았다.

(1) 웜 프로세스의 시작

먼저 웜이 컴퓨터에서 실행을 한다고 가정하자. 먼저 웜은 HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\Version 값을 검사한다. Sisworm.exe 는 위의 레지스트 경로를 이용하는데, 만약 위의 경로에 값이 이미 존재하고 있다면, 해당 컴퓨터는 벌써 바이러스에 걸린 것이다. 즉, 이미 실행된 적이 있다는 이야기이다. 만약 위의 레지스터 값이 없다면 HKEY_CURRENT_KEY\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\Version 에서도 값을 검사하게 된다. 만약 이 경로에도 없다면, Sisworm.exe 은 이 컴퓨터에서 실행된 적이 없다고 가정하고 FIRST_TEMP 플래그에 1 을 설정한다. 1 로 설정되기 전의 값은 0 이다. 즉, 1 이면 sisworm.exe 이 이미 실행됐던 적이 있다고 판단하고, 0 이면 처음 실행되는 것이라 판단한다.

프로그램의 초반에서는 특정한 검사를 하게 된다. 바로 특정한 조건을 검사하여 만약 조건이 만족 된다면 웜의 악성 코드가 실행되지 않는 것인데, 이 웜에서는 2004년 6월 16일 2:28:57 까지만 작동하는 것이 조건이다. 컴퓨터의 시스템 시간과 비교를 하여 만약 시간이 지났다면 웜은 더 이상 작동하지 않을 것이다.

(2)-1 실행된 적이 있다면?

실행된 적이 있었다면 sisworm.exe 은 그 다음의 작업을 진행하기 위해 mutex 를 생성한다. Mutex 의 이름은 SwebSipcSmtxS0 이다.

(2)-2 실행된 적이 없다면?

위의 (1) 에서 만약 웜이 해당 컴퓨터에서 처음 실행되는 경우라면 FIRST_TEMP 플래그가 1 로 설정된다고 하였다. 만약 TEMP 플래그가 1 일 경우에, 웜은 사용자가 '이 것은 뭐지? 바이러스인가? 웜인가?' 라는 의심을 하지 않게 만들기 위해 메모장을 실행한다.

메모장으로 열려진 파일 이름은 '문서내용' 이다. 이 메모장 안에 들어있는 내용은 랜덤한 알고리즘을 이용하여 채운 것이기 때문에, 매번 '문서내용' 파일이 열릴 때마다, 그러니까 각 컴퓨터가 웜에 걸릴 때마다 열려지는 파일 내용이 전부 다르게 된다. 파일 내용을 랜덤하게 만든 이유는, 웜을 잡아내는 백신의 패턴 잡기를 피하기 위함이라 생각된다.

(3) winlogonm.dll 과 winsystemm.exe 는 악성 파일

웜은, 해당 컴퓨터의 윈도우 시스템 디렉토리에 winlogonm.dll 이라는 악성 파일을 만든다. 원래 윈도우에는 winlogon.exe 라는 정상적인 기능을 하는 파일이 존재하는데, winlogonm.dll 과 같이, 이름을 비슷하게 만듬으로써 바이러스 감염 여부의 파악을 힘들게 만들기 위함이다.

Winlogonm.dll 은 특수한 기능을 하는 악성 파일인데, 웜은 이 dll 파일을 생성한 후 LoadLibrary 를 수행한다. Winlogonm.dll 에 대해서는 뒤에서 더 알아볼 것이다.

또한, 웜은 winlogonm.dll 을 생성하는 것 뿐만 아니라, 자기 자신의 복제본을 윈도우 시스템 디렉토리 밑에 winsystemm.exe 이라는 파일로 생성한다.

(4) 레지스트 등록

웜은, 윈도우가 부팅될 때마다 자동으로 실행하기 만들기 위해 레지스트를 조작한다. 이 웜에서는 HKEY_LOCAL_MACHINE 과 HKEY_CURRENT_USER 의 Software\Microsoft\Windows\CurrentVersion\Run 레지스트를 열어 만약 값이 없다면 해당 레지스트에 winsystemm.exe 을 넣는다. 이 winsystemm.exe 은, 악성 웜의 복제본이란 것을 위에서 알아보았다.

레지스트의 이름에서 알 수 있듯이, 이 곳에 등록되는 레지스트 값들은, 시스템이 부팅되고 나서 자동으로 실행되게 된다.

(5) 직접적인 전파 방법

웜이란 것은 원래 전파의 기능을 갖고 있는 것을 말한다. 이 웜에서는 자신을 전파하기 위해 여러 가지 방법이 구현 되어있는데, P2P 나 메일을 이용한 간접적인 전파 방법도 있고 타겟 컴퓨터의 취약성을 공격하여 직접적으로 관리자 권한을 획득하는 방법 등의 전파 방법이 있다. 전파 대상이 되는 운영 체제는 윈도우 운영 체제이다. 여기서는 직접적으로 공격을 하는 방법에 대해서 알아볼 것인데, 2 가지 정도가 구현이 되어 있다.

[SMB IPC], [LSASS] 공격

먼저, 이 웜에서는 Target 으로 211.241.82.124 를 잡고 있다. 위의 IPC 와 LSASS 는 모두 445 포트를 이용하여 공격을 한다. 그런데 211.241.82.124 는 현재 네트워크 접속이 되지 않는 상태이다. (의도한 것인지 다운된 것인지는 모르겠다.) 웜을 분석해보면, target 서버로 connect 후에, 공격을 하게 되는데 연결 자체가 되지 않아 그 다음 프로시져가 호출이 되지 않아서 웜 분석이 조금 까다로웠다. 그래서 sisworm.exe 이 구동될 때 메모리를 조작하여 target 을 211.241.82.124 가 아니라 192.168.0.3 으로 잡고 테스트를 하였다. 192.168.0.3 은 가상 IP 이고, 내가 관리하고 있는 네트워크이고 OS 는 Windows 2003 이다..

SMB IPC 공격

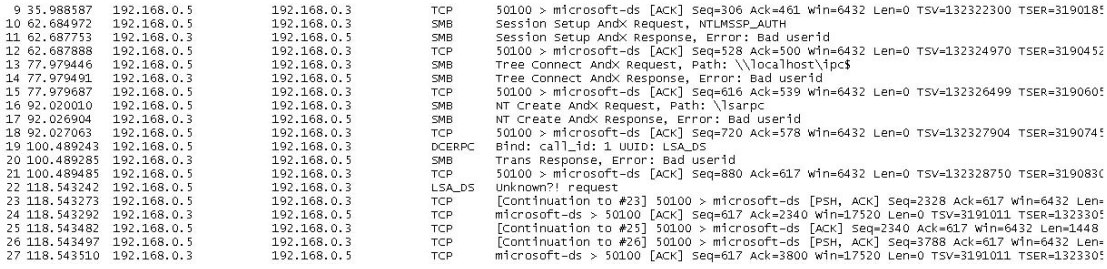
Target 시스템의 IPC$ 공유에 연결하여 악성 코드를 전파하는 방법이다. 물론 이 때, Target 시스템에서는 암호가 설정되어있지 않아야 한다. 암호가 설정되어있지 않은 취약한 상황이라면 웜은 이 점을 이용하여 바이러스를 전파할 수 있다. 그런데, 리버싱을 해 본 결과, 이 웜에서 IPC$ 에 연결을 시도하는 것 자체는 하지만, 결과에 상관 없이 (여기서 결과란, IPC$ 로 접근 허용, 불가 등을 의미한다.) 그에 뒤따르는 공격을 하지 않는 것 같다.

LSASS 공격 (Local Security Authority Subsystem Service)

실존하는 많은 종류의 웜이 이 취약점을 이용한 공격을 수행하고 있다. 이 웜 또한 마찬가지이다. 이 웜에서 주의 깊게 봐야할 점은, 같은 취약점을 노리는 공격을 3 번을 수행한다는 것이다. 다시 말하면 SMB IPC 와 LSASS 공격을 각각 3 번씩 수행하는데, SMB IPC 는 3 번 다 동일한 방법으로 접근을 한다. 그런데 LSASS 공격에는 각각 조금씩 차이점이 있다.

예를 들어 첫번째 LSASS 공격은 평범하다. 그에 비해 두번째 LSASS 공격은 첫번재 공격보다 Packet 의 양이 많았고 (여기서 말하는 Packet 은 공격 코드를 의미한다. Nop 과 Shellcode) Overflow 공격에 흔히 쓰이는 Nop 코드도 첫번째와는 달랐다. 또 패킷이 많아지다보니, 첫번째 현상에서는 없었던 프레그먼테이션 현상도 일어나게 되었다. 두번째와 세번째 역시 약간의 차이가 보였다. 아마도 웜에 LSASS 취약점을 공격하는 코드를 3 개를 장착하고, 모듈화 공격을 수행하는 방식으로 보인다.

[smb, lsass 공격 화면 –ethereal]



(6) 간접적인 전파 방법

이 웜은 간접적인 전파 방법 기능도 구현되어 있다. 대표적으로 P2P 와 메일을 이용한 것인데, 앞서 이야기했듯이, 디스어셈을 해보니, 메일을 이용한 전파 방법이 구현이 되어있고, 관련된 문자열들도 보였는데 실제로 이 기능을 호출 하는 부분이 없었다. 그렇기 때문에 메일을 이용한 전파 방법에 대해서는 자세히 다루지 않겠다. P2P 를 이용한 전파 방법은 실제로 수행을 한다. 이 웜에서 노리는 P2P 프로그램은 Kazaa 라는 프로그램이다.

웜은 먼저, Software\Kazaa\Transfer 의 레지스트에 값이 있는지 확인한다. 값이 존재한다면 kazaa 소프트웨어가 컴퓨터에 설치되어져 있다는 이야기이다. 나는 kazaa 가 설치되어있지 않아서 리버싱을 할 때, 이 부분을 건너뛰게 되었지만 분석을 위하여 플래그를 바꾸어 이 루틴에 들어오게 되었다. 만약 설치가 되어있다면 다시, DlDir0 값을 레지스트에서 가져오는데 아마도 이 값은 kazaa 프로그램에서 사용하는 공유 폴더 쯤일 것이다.

그리고 다음과 같은 문자열 중에서 하나를 고르고, 확장자도 랜덤하게 고른다. 각각 다음과 같다. nuke2004, office_crack, rootkitXP, strip-girl-2.0bdcom_patches, activation_crack, icq2004-final, winamp5, dcom_patches 이 것들은 파일명에 쓰이는 문자열이고, 확장자는 exe, pif, bat, scr 이다.

웜은 자기 자신을 위의 문자열 중에서 임의로 합친 문자열을 이용하여, kazaa 의 공유 폴더 안에 카피를 해 놓는다. 만약 다른 kazaa P2P 사용자가 그 파일을 다운받아 실행을 하게 되면, 그 사람 역시 악성 코드에 감염될 것이다.

(7) DoS 공격

Sisworm.exe 는 특정 타겟에 대한 DOS 공격도 감행한다. 그러나 DOS 공격을 하기 위해서는 특정 조건을 만족해야만 한다. 예를 들어 sisworm.exe 의 경우에는 2004년 6월 15일 16:09:18 이후에만 DOS 공격을 수행한다. 만약 이 조건이 만족된다면 DOS 공격을 수행하는 프로시져를 호출한다.

이 프로시져는 한번 수행되고 나면 1024초를 쉰 뒤에 다시 호출되는 방식이다. DOS 공격의 타겟으로 잡고 있는 사이트는 consult.skinfosec.co.kr 의 80 포트, HTTP 이다. Consult.skinfosec.co.kr 에 연결할 준비가 되었다면 웜은 공격을 준비한다. 쓰레드를 63 개를 생성하는데 이 쓰레드는 웜 공격에 필요한 준비를 수행한다. 이 쓰레드들은 생성된 후 consult.skinfosec.co.kr 의 80 포트로 DOS 공격을 시도할 것이다. 쓰레드가 모두 생성되면,

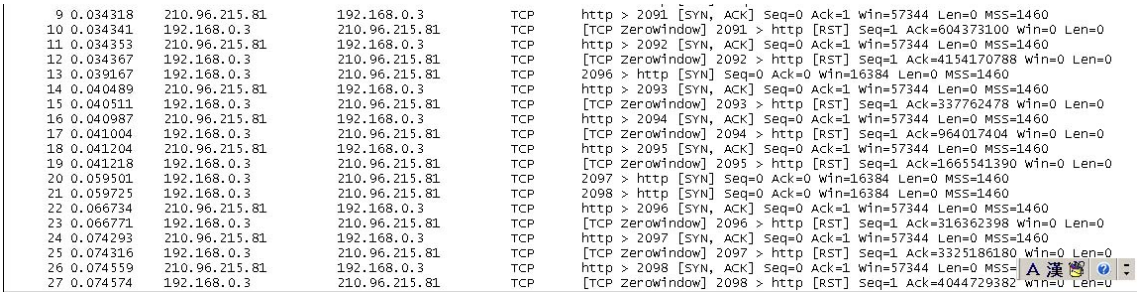
이번엔 쓰레드를 이용하지 않고 직접 그 프로시져를 호출한다. 총 64 번이 호출되는 셈이다. 공격은 HTTP 를 타겟으로 하는데 HTTP 에 연결하기 위해 사용하는 메소드는 다음과 같다.

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

생성된 쓰레드들은 consult.skinfosec.co.kr 의 80 포트로 지속적으로 연결을 하게 되는데 한 쓰레드 당 10 번의 공격을 수행하고 종료한다. 제대로 공격이 된다면 DOS 프로시져가 호출되는 한 번의 공격에 쓰레드의 63 * 10 + 10 (+10 은 쓰레드를 이용하지 않고 직접 호출한 것), 총 640 개의 HTTP DOS ATTACK 이 수행될 것이다.

그러나 공격은 제대로 수행되지 않는 것 같다. 스니퍼를 이용하여 감시해봤을 때, DOS 공격 시 연결 시도는 수행했지만, 이상하게도 TCP flag 에 Reset 이 check 되어 있고 WINDOW 사이즈가 0 인 패킷을 보내는 것이다. 즉, RST 패킷을 보내는 것이었다. Zero Window 는 TCP/IP 에서 프로세스가 데이터를 더 이상 받아들이기 힘들 때 보내는 패킷인데, 아무래도 웜 제작자가 DOS 모듈을 구현할 때 무엇인가 실수를 한 게 아닌가 생각된다.

[**consult.skinfosec.co.kr DOS attack –ethereal**]



9	0.034318	210.96.215.81	192.168.0.3	TCP	http > 2091 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
10	0.034341	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2091 > http [RST] Seq=1 Ack=604373100 Win=0 Len=0
11	0.034353	210.96.215.81	192.168.0.3	TCP	http > 2092 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
12	0.034367	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2092 > http [RST] Seq=1 Ack=4154170788 Win=0 Len=0
13	0.039167	192.168.0.3	210.96.215.81	TCP	2096 > http [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
14	0.040489	210.96.215.81	192.168.0.3	TCP	http > 2093 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
15	0.040511	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2093 > http [RST] Seq=1 Ack=337762478 Win=0 Len=0
16	0.040987	210.96.215.81	192.168.0.3	TCP	http > 2094 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
17	0.041004	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2094 > http [RST] Seq=1 Ack=964017404 Win=0 Len=0
18	0.041204	210.96.215.81	192.168.0.3	TCP	http > 2095 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
19	0.041218	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2095 > http [RST] Seq=1 Ack=1665541390 Win=0 Len=0
20	0.059501	192.168.0.3	210.96.215.81	TCP	2097 > http [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
21	0.059725	192.168.0.3	210.96.215.81	TCP	2098 > http [SYN] Seq=0 Ack=0 Win=16384 Len=0 MSS=1460
22	0.066734	210.96.215.81	192.168.0.3	TCP	http > 2096 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
23	0.066771	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2096 > http [RST] Seq=1 Ack=316362398 Win=0 Len=0
24	0.074293	210.96.215.81	192.168.0.3	TCP	http > 2097 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=1460
25	0.074316	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2097 > http [RST] Seq=1 Ack=3325186180 Win=0 Len=0
26	0.074559	210.96.215.81	192.168.0.3	TCP	http > 2098 [SYN, ACK] Seq=0 Ack=1 Win=57344 Len=0 MSS=
27	0.074574	192.168.0.3	210.96.215.81	TCP	[TCP ZeroWindow] 2098 > http [RST] Seq=1 Ack=4044729382 Win=0 Len=0

(8) winlogonm.dll 의 특징

Sisworm.exe 은 winlogonm.dll 을 만든다는 것을 위에서 알아보았다. 다시 이 것에 대해서 알아보겠다. Winlogonm.dll 은 5.50KB의 작은 크기를 갖는다. 먼저 HKEY_CLASSES_ROOT 의 E6FB5E20-DE35-11CF-9C87-00AA005127ED}\InprocServer32 와 35CEC8A3-2BE6-11D2-8773-92E220524153}\InprocServer32 레지스트에 각각 WebCheck, winlogonm.dll 을 입력한다.

또한 winlogonm.dll 의 주요 기능으로써, 백도어를 만드는 기능이다. 백도어로 사용되는 포트는 4017~4188 번이며, 4017 부터 증가하며 생성을 하는데 만약 4188 을 넘어가게되면 다시 4017 로 초기화하여 포트를 개설한다. 이때 사용하는 프로토콜은 TCP 이며 각 포트별로 2 개의 쓰레드를 생성한다.

누군가가 이 포트에 접속을 하였다 해도, 아무나 이용할 수 있는 것은 아니다. 왜냐하면 백도어 프로그램에서, 이 포트에 접속한 사람이 입력한 데이터가 0x133C9EA2 인지 확인을 한 후에 작업을 진행하기 때문이다. 만약 이 인증을 통과하지 못한다면 접속이 끊기게 된다.

접속이 되고 나서는, 접속한 사람이 보내온 데이터를 파일로 저장시킨다. 즉 만약 접속한 사람이 바이너리를 전달한다면 백도어는 이 바이너리 데이터를 서버에 파일 형태로 저장한다. 저장이 되고 난 후에, 백도어는 이 바이너리 파일을 실행한다. 이 것이 winlogonm.dll 의 특징이다.

(9) 얼마나 악영향을 미치는 가?

Sisworm.exe 은 대회를 위해서 만들어진 웜이다. 그렇기 때문에 감염이 되었다 해도 실존하는 다른 웜들처럼 심각하게 피해를 미치지는 않는다. 예를 들어서 DOS 공격을 감행할 때, 한 쓰레드당 10 번의 공격을 하는 것만 봐도 알 수 있다. 또한, 분석을 보면 알겠지만 시스템의 파일을 삭제한다거나 변조하는 등의 크게 해를 미치는 요소가 없다. 그렇기 때문에 네트워크 적으로나 시스템 적으로나 큰 피해가 없을 것이다.

그러나, winlogonm.dll 로 인해서 열리는 백도어는 문제가 될 수 있다. 비록 약간의 인증 과정이 있긴 하지만, 만약 이 인증 과정을 통과하게 되면 서버에 악성 코드를 실행할 수도 있게 된다. 또한 LSASS 공격을 수행할 때 취약점이 존재하는 컴퓨터라면 해당 컴퓨터의 데몬을 비정상 종료시키거나 하는 악영향을 미칠 수도 있을 것이다.

(그리고 ~WRLXXXX.tmp 와 같은 형태로 웜과 관련이 있어 보이는 파일이 웜이 존재하는 디렉토리에 계속 생성되고 있는데 무엇이 원인인지는 시간상 파악하지 못하였다. 파일을 열어보면 바이너리와 아스키가 섞여있으며 웜과 관련된 내용이 보인다.)

(10) 대응 방법

감염을 막는 방법은, 이 웜에서 사용하는 공격 방법을 역으로 생각해보면 알 수 있다. 예를 들어 SMB IPC 의 경우 사용자가 좋은 패스워드를 등록하면 막을 수 있을 것이고, LSASS 의 경우에는 윈도우 보안 패치를 하면 해결할 수 있다. 그러나 445 를 막는 방법은 좋지 않다. 왜냐하면 너무 많이 사용하는 포트이기 때문이다. 그렇지만 LSASS 공격의 경우에는 공격시 대량의 NOP 코드를 첨부함으로써 일반적으로 잘 발생되지 않는 데이터가 가게 된다. 이를 이용한 패턴으로 패킷을 block 시킬 수도 있을 것이다.

DOS 공격은 consult.skinfosec.co.kr 이라는 특정 사이트로 하는 것이기 때문에 개인 사용자는 크게 신경 쓸 필요가 없다. 만약 해당 사이트의 네트워크 관리자라면 대처를 해야 한다. 이 웜에서는 DOS 공격을 할 때, 다양한 패턴을 사용하지 않는다. 즉, 고정된 데이터를 사용하게 됨으로써 패킷의 사이즈나 패턴이 일정하기 때문에 block 을 걸어 둘 룰을 만들기 어렵지 않을 것이다. 또한, 이 웜에서 발생하는 HTTP 메소드는, 인터넷 익스플로어가 생성하는 일반적인 헤더와는 조금 다르기 때문에 헤더를 보고도 block 시킬 수 있을 것이다. 인터넷 익스플로어에서는 보다 많은 헤더를 붙여 전송을 한다.

만약 이미 감염이 된 PC 라면 다음과 같은 방법으로 없앨 수 있다.

-1- Software\Microsoft\Windows\CurrentVersion\Run 의 winsystemm.exe 를 없앤다.

-2- Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\Version 없앤다

-3- 그 외 레지스트 찾기에서 winlogonm.dll 과 winsystemm.exe 이 들어간 값은 모두 지운다. (윈도우에서 정상적인 기능을 하는 파일 중에 위와 같은 이름은 없으니 안심하자.)

-4- 안전 모드로 부팅 후에 winlogonm.dll 과 winsystemm.exe 를 찾아서 모두 삭제한다.

(11) 분석 방법

웜의 파일 형태는 Peid 0.9 를 이용하여 알아보았고, 웜의 UPX 압축 해제는 UPX 1.22 Windows 버전을 이용하였다. 웜의 디스어셈블링과 리버싱에 사용한 툴은 ollydbg 였고, packet 을 capture 할 때 사용한 툴은 ethereal 이다.

프로그램의 처음 실행부터 끝까지 리버싱으로 따라가면서 분석을 하였다. 많은 시간을 투자하였는데도 시간이 부족하여 모든 부분을 세세하게 검사하지 못했고 빠트리거나 틀린 부분이 있을 수도 있다. Ollydbg 로 한 줄 한 줄 따라가거나, 필요 없는 부분은 다음 코드에 중단점을 걸어놓고 진행을 하는 방법으로 디버깅을 하였다. 특정 경우에는 프로시져를 좀 더 정확하게 살펴보기 위해, 조건식이 만족하지 않는데도 임의로 플래그를 고쳐서 가게 하는 방법을 사용하기도 했고, 디버깅을 위해 메모리를 조작하기도 하였다. 또한, 위에서도 언급했지만 211.241.82.124 가 연결이 안되서 내부 네트워크의 192.168.0.3 으로 연결하게끔 조작을 하기도 하였다. 윈도우에는 별로 익숙치 않아서 이런 노가다 방법으로 프로그램을 분석하였다.

부족한 점이 몇가지 있었다면, 정말로 시간이 촉박하였기 때문에 winlogonm.dll 은 어셈블리 코드만 분석을 하고, 실제로 따라가면서 리버싱을 해보지는 못했다. 그렇기 때문에 내가 작성한 정보대로 작동을 할지 의문이다.

[웜 분석 장면 – ollydbg]