

# Mp3 file can execute code in Winamp [Sandblad advisory #5]

**Source:** <http://www.derkeiler.com/Mailing-Lists/securityfocus/bugtraq/2002-04/0388.html>

---

**From:** Andreas Sandblad ([sandblad@acc.umu.se](mailto:sandblad@acc.umu.se))

**Date:** 04/26/02

Date: Fri, 26 Apr 2002 08:30:56 +0200 (CEST)  
From: Andreas Sandblad <[sandblad@acc.umu.se](mailto:sandblad@acc.umu.se)>  
To: [bugtraq@securityfocus.com](mailto:bugtraq@securityfocus.com)

– Sandblad advisory #5 –

-----

Title: Mp3 file can execute code in Winamp.

Date: [2002-04-26]

Software: Nullsoft Winamp 2.79

Rating: High because mp3 files are widely trusted as safe.

Impact: Specially crafted mp3 file can execute arbitrary code when played in Winamp due to a buffer overflow condition.

Vendor: Nullsoft has confirmed the vulnerability.

Patch: Winamp 2.80 released 02-04-25 will fix the issue.

Download at: <http://www.winamp.com/>

Workaround: Disable the minibrowser \_ \_

(enabled by default). o' \,=./ `o

Author: Andreas Sandblad, [sandblad@acc.umu.se](mailto:sandblad@acc.umu.se) (o o)

-----ooO--( )--Ooo-----

## NON TECHNICAL DESCRIPTION:

=====

It is possible to modify an existing mp3 file in such a way that it can carry a virus. The virus is activated when the mp3 file is played in Winamp and can then infect other mp3 files found on harddrives or network shares. In order to protect yourself you need to upgrade to Winamp 2.80 or disable the minibrowser.

## TECHNICAL DESCRIPTION:

=====

A mp3 file can contain the ID3v2 tag. It's a newer version of the ID3v1 tag and carries information like title, artist and album. The tag is parsed by Winamp when a mp3 file is loaded.

If the minibrowser is enabled, Winamp will try to query a script on <http://info.winamp.com> for extra information about the song, based on data

## SecurityFocus Bugtraq: Mp3 file can execute code in Winamp [Sandblad advisory #5]

from the ID3v2 tag. The buffer overflow condition occurs when the url string intended to be sent to the minibrowser is created. That means the buffer overflow occurs before any actual internet connection to info.winamp.com is being made.

The easiest way to test the buffer overflow condition is to apply at least 159 "?" characters in the title field of the ID3v2 tag. When playing the mp3 file in Winamp 2.79 the following error log was created:

```
Access violation – code c0000005 (first chance)
eax=00000021 ebx=00000000 ecx=0012bd00 edx=00000032 esi=00000113
edi=00000113
eip=32253132 esp=00129c08 ebp=25313225 iopl=0         nv up ei pl zr na po
nc
cs=001b  ss=0023  ds=0023  es=0023  fs=0038  gs=0000
efl=00010246
32253132 ?? ???
```

If we debug Winamp we notice that the created url to be sent to the minibrowser looks something like this:

[http://info.winamp.com/winamp/WA.html?Alb=%21%2...\(alot\)...%21Cid=winamp&Tid=&Track=%21%21%21%...\(a](http://info.winamp.com/winamp/WA.html?Alb=%21%2...(alot)...%21Cid=winamp&Tid=&Track=%21%21%21%...(a)

We also understand that the buffer is overwritten by maximum one NULL character, making the register esp change to 0x129c00. The esp is then incremented by 0x4 and a ret is done, sending the program to the address found in 0x129c04.

We need the stackpointer 0x129c04 to be in a dataspace we are in control of. Seems like we are lucky! The url string is stored from 0x1298d8 to 0x129cd8, thus it seems like we can control the eip. And in fact we can. Simply check the eip of the error log. It displays eip=0x32253132, that is hex for "2%12". Seems like the stack pointer is located somewhere in our url string containing "...%21%21%21%21...". (Remember that memory addresses are retrieved backwards!)

So how do we exploit this thing? Well normal buffer overflow exploit is to return to anywhere in memory where we can find JMP ESP (opcode 0xFFE4) in order to get back to the string that caused the buffer overflow. This method is used because normal buffer overflows are only limited to the fact that they can't insert 0x00 in the return address because of string operations.

The problem we face is we can only use characters a-z, A-Z, 0-9, ".", because all others will be escaped to %HEX. That means the addresses containing the JMP ESP instruction we need to find are a bit limited (but of course in most cases not impossible to find as we only need one location). Since the versions of the system .dll files Winamp import at launch is OS and system dependent, it really depends on the system if we are able to find an available address containing the JMP ESP assembler instruction.

SecurityFocus Bugtraq: Mp3 file can execute code in Winamp [Sandblad advisory #5]

Once we control the eip we have to do something useful. Since we still are limited in what kind of opcodes we can construct, it's better to try to get somewhere in memory where our url is not escaped (ex. ! to %21). When debugging we notice the register ecx is 0x12bd00 and points to a copy of our url partly unescaped. So if we somehow can increase the ecx and change the memory to do JMP ECX we are on a address where we can create any opcode we want. This can be done with opcodes like:

```
0x66335142 ("f3QB") XOR DX,[ECX+0x42]
0x4A ("J") DEC EDX
0x665A ("fZ") POP DX
0x6652 ("fR") PUSH DX
```

It's not an easy task to perform the above and on some OS it has to be done differently, but still it's possible.

Disclaimer:

=====

Andreas Sandblad is not responsible for the misuse of the information provided in this advisory. The opinions expressed are my own and not of any company. In no event shall the author be liable for any damages whatsoever arising out of or in connection with the use or spread of this advisory. Any use of the information is at the user's own risk.

Feedback:

=====

Please send suggestions and comments to: \_ \_

[sandblad@acc.umu.se](mailto:sandblad@acc.umu.se) o' \,=./`o

(o o)

-----ooO--(\_)--Ooo-----

Andreas Sandblad,  
student in Engineering Physics at the University of Umea, Sweden.

-----

- 
- **Previous message:** [Markus Friedl: "Revised OpenSSH Security Advisory \(adv.token\)"](#)
  - **Messages sorted by:** [\[ date \]](#) [\[ thread \]](#) [\[ subject \]](#) [\[ author \]](#) [\[ attachment \]](#)