

White Hat Shellcode Workshop

Didier Stevens

Go to
<http://workshop.DidierStevens.com>

Unzip shellcode-workshop.zip to C:\

Password is **workshop**.

First example:
loading/unloading a DLL

Start calc.exe

Start procexp.exe (Process Explorer) and view the DLLs loaded in calc.exe

Execute the following command from a command-line in `c:\workshop`:

```
create-remote-thread.py calc.exe kernel32.dll  
LoadLibraryA str:c:\workshop\msgbox-hello.dll
```

Click on the dialog box: the dialog box indicates that the DLL was successfully loaded.

Execute the following command from a command-line in `c:\workshop`:

```
create-remote-thread.py calc.exe kernel32.dll FreeLibrary  
0xBC0000
```

Reinject the DLL and take note of the base address

Execute the following command from a command-line in
c:\workshop: simple-shellcode-generator.py -o unload-dll.asm
-I "kernel32.dll FreeLibrary 0xBC0000" (replace 0xBC0000
with the base address you wrote down)

Start a NASM Shell from the Start \ All Programs \ Netwide
Assembler menu

From the NASM Shell: cd [c:\workshop](#)

From the NASM Shell: nasm -o unload-dll.bin unload-dll.asm

Execute the following command from a command-line in
c:\workshop: create-remote-thread.py calc.exe unload-dll.bin

Second example:
enforcing DEP

exit calc.exe from the previous example, and start it again

Notice DEP is enabled (DEP column in Process Explorer)

Execute the following command from a command-line in c:\workshop: create-remote-thread.py calc.exe kernel32.dll SetProcessDEPPolicy 0

refresh Process Explorer's view (F5) and notice that DEP has been turned off

Execute the following command from a command-line in
c:\workshop: create-remote-thread.py calc.exe
kernel32.dll SetProcessDEPPolicy 1

refresh Process Explorer's view (F5) and notice that
Permanent DEP is enabled

Execute the following command from a command-line in
c:\workshop: create-remote-thread.py calc.exe
kernel32.dll SetProcessDEPPolicy 0

refresh Process Explorer's view (F5) and notice that
Permanent DEP is still enabled

Execute the following command from a command-line in
c:\workshop: simple-shellcode-generator.py -o dep.asm -l
"kernel32.dll SetProcessDEPPolicy 1"

From the NASM Shell: nasm -o dep.bin dep.asm

Execute the following command from a command-line in
c:\workshop: copy \windows\system32\calc.exe calc-
dep.exe

Start LordPE.exe (LPE-DLX_1.4 folder): open calc-
dep.exe

Note the EntryPoint and the Image Base: $0x00012475 + 0x01000000 = 0x01012475$

Edit dep.asm, replace ret with these 2 lines
mov eax, 0x01012475
jmp eax

From the NASM Shell: `nasm -o dep.bin dep.asm`

From LordPE: add a section from file: `dep.bin`

Notice the VOffset for `dep.bin`: `0x0001F000`

Change the EntryPoint to `0x0001F000`

From LordPE: Save

From LordPE: Rebuild PE

Third example:
testing your security setup

From the NASM Shell: `nasm -o sc-createfile.bin sc-createfile.asm`

Execute the following command from a command-line in `c:\workshop`: `create-remote-thread.py calc.exe sc-createfile.bin`

Check if file `c:\windows\system32\testfile.txt` has been created: it has.

First delete `c:\windows\system32\testfile.txt`

`Psexec -ld c:\windows\system32\calc.exe`

Execute the following command from a command-line in `c:\workshop:` `create-remote-thread.py calc.exe sc-createfile.bin`

Check if file `c:\windows\system32\testfile.txt` has been created: it has not.

Fourth example:
patching an application

Install AdbeRdr910_en_US_Std.exe

Open javascript.pdf, and notice the popup from the embedded JavaScript

Disable JavaScript: Edit / Preferences / JavaScript /
Enable Acrobat JavaScript

Close and open javascript.pdf: notice the nag screen from Adobe Reader

From the NASM Shell: `nasm -o sc-sar.bin sc-sar.asm`
Close javascript.pdf

Start `dbgview.exe`

Execute the following command from a command-line in `c:\workshop`: `create-remote-thread.py calc.exe sc-sar.bin`, and notice the message in `dbgview` after some time

Open `javascript.pdf`: the nagscreen is gone.

Fifth example:
preventing heapspays with
shellcode

uninstall Adobe Reader 9.1

install AdbeRdr812_en_US.exe

start Adobe Reader

unzip util-printf.zip (password is workshop)

open util-printf.pdf and notice Adobe Reader crashing

Execute the following command from a command-line in
c:\workshop: simple-shellcode-generator.py -o sc-
mba.asm -l "user32.dll MessageBoxA 0 str str 0"

Edit sc-mba.asm with notepad and add 50 NOPs

From the NASM Shell: nasm -o sc-mba.bin sc-mba.asm

Start Adobe Reader 8

Execute the following command from a command-line in
c:\workshop: create-remote-thread.py -a 0x30303020
calc.exe sc-nopsled-mba.bin

Notice the message box (and then click the message box
away)

Now open util-printf.pdf

Notice the 2 message boxes (and then click the message boxes away)

Double click on heaplocker.reg and merge the entries to the registry

Start Adobe Reader 8

Execute the following command from a command-line in c:\workshop: `create-remote-thread.py AcroRD32.exe kernel32.dll LoadLibraryA str:c:\workshop\heaplocker.dll`

Notice the messages in DbgView

Now open util-printf.pdf

Notice the warning

Take a look at the threads with Process Explorer