

# Exploiting Simple Buffer Overflows on Win32

Vivek Ramachandran

SWSE, SMFE, SPSE, SISE, SLAE, SGDE Course Instructor

Certifications: <http://www.securitytube-training.com>

Pentester Academy: <http://www.PentesterAcademy.com>

# Automating Bad Character Detection

# Challenges in Bad Character Detection

- Painfully manual
- Visual Inspection
  - characters can be transformed which one might overlook

# What if we had a tool ...

- Could generate character sets
- Compare memory location with the generated character sets used
- Finds bad characters using the above

**Enter Mona.py!**

# Mona.py

- <http://redmine.corelan.be/projects/mona>
- Created by the Corelan Team (Peter)
- Plugin for Immunity Debugger
- <https://www.corelan.be/index.php/2011/07/14/mona-py-the-manual/> (Documentation)

# Minishare

- Bad Character Detection using Mona.py
- Minishare

# Step 1: Set Working Folder for Mona

```
Immunity Debugger 1.85.0.0 : R'lyeh
Need support? visit http://forum.immunityinc.com/
"C:\Program Files\MiniShare\minishare.exe"

File 'C:\Program Files\MiniShare\minishare.exe'
[21:47:57] New process with ID 000000F0 created
00960000 Main thread with ID 00000510 created
00400000 Modules C:\Program Files\MiniShare\minishare.exe
5D090000 Modules C:\WINDOWS\system32\COMCTL32.DLL
71AA0000 Modules C:\WINDOWS\system32\WS2HELP.dll
71AB0000 Modules C:\WINDOWS\system32\WS2_32.DLL
763B0000 Modules C:\WINDOWS\system32\COMDLG32.DLL
77C10000 Modules C:\WINDOWS\system32\msvcrt.dll
77DD0000 Modules C:\WINDOWS\system32\ADVAPI32.dll
77E70000 Modules C:\WINDOWS\system32\RPCRT4.dll
77F10000 Modules C:\WINDOWS\system32\GDI32.dll
77F60000 Modules C:\WINDOWS\system32\SHLWAPI.dll
77FE0000 Modules C:\WINDOWS\system32\Secur32.dll
7C800000 Modules C:\WINDOWS\system32\kernel32.dll
7C900000 Modules C:\WINDOWS\system32\ntdll.dll
7C9C0000 Modules C:\WINDOWS\system32\SHELL32.dll
7E410000 Modules C:\WINDOWS\system32\USER32.dll
00960000 [21:47:58] Program entry point
OBADF00D Writing value to configuration file
OBADF00D Old value of parameter workingfolder = f:\automate\%p
OBADF00D [+] Saving config file, modified parameter workingfolder
OBADF00D New value of parameter workingfolder = f:\mn\%p
OBADF00D
[+] This mona.py action took 0:00:00.030000
```

```
!mona config -set workingfolder f:\mn\%p
```

# Step 2: Generate Bytearray

```
7E410000 Modules C:\WINDOWS\system32\USER32.dll
OBADFOOD Generating table, excluding 0 bad chars...
OBADFOOD Dumping table to file
OBADFOOD [+] Preparing output file 'bytearray.txt'
OBADFOOD - Creating working folder f:\mn\Echo-Server-Bad-Char-Special
OBADFOOD - Folder created
OBADFOOD - (Re)setting logfile f:\mn\Echo-Server-Bad-Char-Special\bytearray.txt
"\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f"
"\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f"
"\x40\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f"
"\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f"
"\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f"
"\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf"
"\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f"
"\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff"
OBADFOOD Done, wrote 256 bytes to file f:\mn\Echo-Server-Bad-Char-Special\bytearray.txt
OBADFOOD Binary output saved in f:\mn\Echo-Server-Bad-Char-Special\bytearray.bin
OBADFOOD [+] This mona.py action took 0:00:00.040000
```

!mona bytearray



# Step 3: Find Bad Character

76F6000

mona Memory comparison results

Address	Status	BadChars	Type
0x013b3908	Corruption after 0 bytes	00	normal
0x013b3908	Corruption after 0 bytes		unicode

00861090 70 1E 00 00 00 02 00 80 00 00 00 00 09 DF 55 41 ... bU3

```
!mona compare -f f:\mn\minishare\bytearray.bin -a 013b3908|
```

# Step 4: Create Bytearray with Exclusions

```
00A90000 00103000
00BA0000 0004C000

Log data
Address Message
013B3908 |00 00 00 00 00 00 00 00 00 | Memory
013B3908
013B3908
OBADFOOD

[+] This mona.py action took 0:00:00.801000

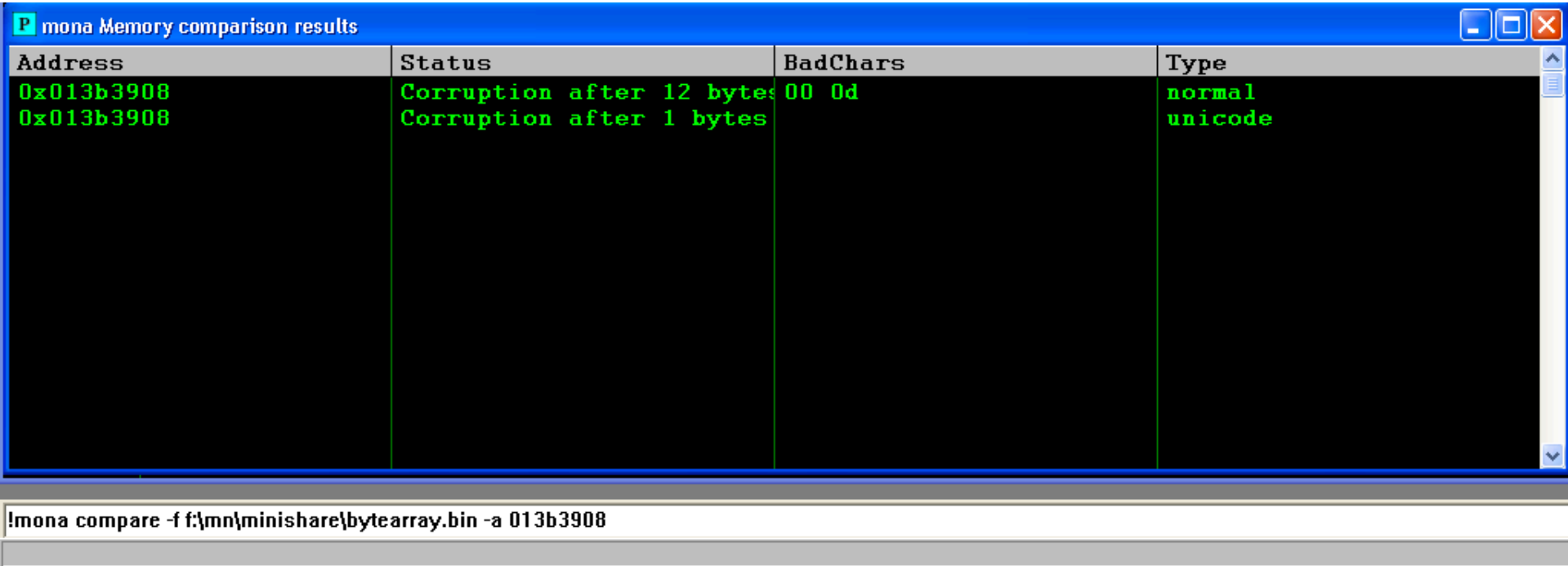
OBADFOOD Generating table, excluding 1 bad chars...
OBADFOOD Dumping table to file
OBADFOOD [+] Preparing output file 'bytearray.txt'
OBADFOOD - (Re)setting logfile f:\mn\minishare\bytearray.txt
"\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x
"\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x
"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x
"\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x
"\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x
"\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\x
"\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\x0\x1\x2\x3\x4\x5\x6\x7\x8\x9\x
"\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\x

OBADFOOD Done, wrote 255 bytes to file f:\mn\minishare\bytearray.txt
OBADFOOD Binary output saved in f:\mn\minishare\bytearray.bin
OBADFOOD

[+] This mona.py action took 0:00:00.040000
```

```
!mona bytearray -cpb "\x00"
```

# Step 5: Repeat and Find All Bad Chars



Address	Status	BadChars	Type
0x013b3908	Corruption after 12 bytes	00 0d	normal
0x013b3908	Corruption after 1 bytes		unicode

```
!mona compare -f f:\mn\minishare\bytearray.bin -a 013b3908
```

# Pentester Academy

PentesterAcademy

a SecurityTube.net initiative



TOPICS

PRICING

WHY SUBSCRIBE

MEMBER ACCESS



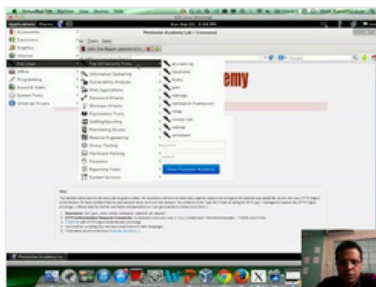
## Revolutionizing Infosec Training

Highly Technical, Hands-on, Affordable

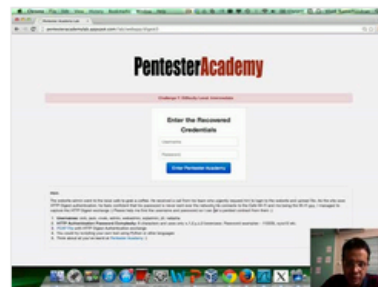
Start Learning Today!

## Latest Videos

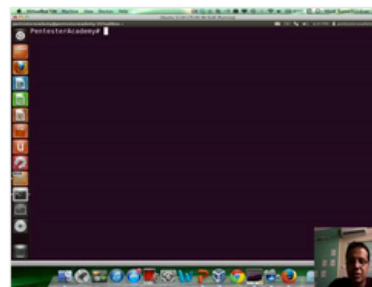
New content added weekly!



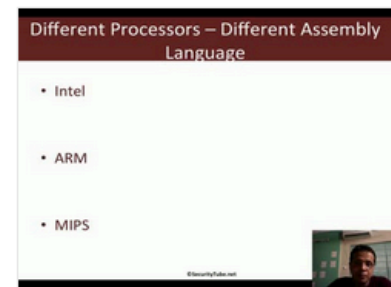
Challenge 7: Cracking Digest Authentication Solution in WAP Challenges



Challenge 7: Cracking Digest Authentication in WAP Challenges



Module 1: GDB Test Solution in x86\_64 Assembly Language and Shellcoding on Linux



Module 1: CPU Information in x86\_64 Assembly Language and Shellcoding on Linux