

Introduction to Information Security, INTROSEC  
Autumn Term 2018

**Report of Practical Laboratory  
Assignment 1**

# Summary

<b>Introduction</b>	<b>3</b>
<b>Laboratory assignments for Windows</b>	<b>3</b>
Exercise 1 - Code Injection	3
Exercise 2 - Windows Registry	3
Exercise 3 - Spoofing: Bypassing the Login Screen	4
Exercise 4 - Monitoring keyboard strokes	5
Exercise 5 - Disclosing masked passwords	5
Exercise 6 - NetBus: Take control over another computer	5
<b>Laboratory assignments for Kali Linux</b>	<b>6</b>
Exercise 1 - Utilising port and vulnerability scanners	6
Exercise 2 - Utilising sniffer tools	8
Exercise 3 - Analysing network traffic	10
Exercise 4 - Metasploit: h4Xing made easy	13
<b>Conclusion</b>	<b>13</b>

# Introduction

The aim of this paper is to report the laboratory assignment 1 of the course Introduction to Information Security. The text is supposed to be not just a presentation of the practical work, but a discussion and a critical view of how the tools presented could be used for malicious purposes. Obviously this also leads us to think about what precautions are necessary to protect a network from this kind of threat.

The paper is divided in two parts reflecting the different OS of the virtual machines with the corresponding exercises.

## Laboratory assignments for Windows

### Exercise 1 - Code Injection

In some cases we would like to understand if an executable has been injected or not. This exercise provides an injected calculator together with an original one. They look equal, their functions are correct and provide good performance in calculation.

By the use of FCIV (File Checksum Integrity Verifier), we easily check if a program such as this injected calculator has been manipulated with a hash program. This action gives us the MD5 hashes of the two calculators, in this way we notice that they are different.

Some code lines were added to the official calc.exe app. The only difference appears to be a "hello world!" message, that is shown when we open the injected executable file.

If a file has been modified from the original version you can not know what changes have been made and for what purpose. For example an executable may have been modified to improve it or to make it compatible with a particular system, but hidden effects may not always be so benign. In this case we can't be sure if the welcome message is the only difference between the two calculators: there might be other malicious things that we are not aware of. In general for this reason it is not safe to run an executable developed by any untrusted source.

It might be the case that the professor modified the executable in order to track all the activities performed on the Windows virtual machine from the group. The injected calculator could activate a procedure that sends a log file to the professor.

### Exercise 2 - Windows Registry

In the Microsoft Windows operating systems, the registry or Windows Registry refers to the database containing settings, options, information, and other values of all programs and hardware installed.

Not all users and programs are allowed to modify the registry, for many reasons like the fact that privacy settings are contained in it. In our case only cs2lab (administrator) is allowed to modify the registry.

This exercise section gives us an introduction in some use of the registry: in particular we work in three actions:

- A. hide user profiles from the login window
- B. hide drives from Windows Explorer
- C. hide the run option in the start menu

The first action avoids the suggestion of a user before logging in. This guarantees a higher level of security, in fact there is no indication of the identity of that particular user. Not only do we need to know the login password, we also need to know the username.

After changing the settings with the action B the hidden drives are still accessible but they are not visible in the Windows Explorer. Hiding drives is a good thing to do in networks shared by many users (for example a office's network) in which there are some shared drives but also employee's private drives that shouldn't be available to the others.

Referring to the task C, hiding the run option shows us how the start panel can be modified from the registry. This action could hinder users to run programs in both way malicious or benign.

The tasks described above are just simple examples, but if you can access the registry and you know very well how it works, you can apply many changes to the computer.

## Exercise 3 - Spoofing: Bypassing the Login Screen

Sticky Key is an application with benign purpose: it provides an accessibility improvement designed to assist users with physical disabilities. It allows to avoid difficulties in pressing several keys at the same time. The exercise takes advantages from that tools in order to satisfy an unsuitable action. Pressing "*shift*" key five times is a shortcut to launch the sticky keys executable.

We copied the command line executable and renamed it with the Sticky Key original name (*sethc.exe*). In this way the action leads to launching the command line.

Accessing the command line a malicious user might access all the computer's contents even before logging in. In fact there is no need of administrator permission.

Having the permission to access to the command line means that you could change the password of a user even if you don't know how to login. It could be a useful point: in the situation in which you forget your password but you have activated this tool, you could be able to resolve easily the problem. On the other hand, you are giving an unlimited access to your computer renouncing security, so if you know that your computer contains important files is better avoid this action.

To avoid such security breaches, the OS should check if the application launched (specially before login) correspond to the original version. This task could be completed using a file checksum integrity verifier, as seen in exercise 1.

## Exercise 4 - Monitoring keyboard strokes

A keylogger is a program that usually runs in background, without the user been aware, and records all the keys struck in a keyboard. This task is usually performed by a malicious person that accessing the files could save passwords or control another user's activity, like chats, internet searches, commands and files used.

The executable creates a log file as the result: the txt file contains all the keys struck in the keyboard since the launch of the program.

The keylogger is executed without any notification. Therefore, a unaware user has usually no suspect at all of the threat.

With the operating methods of this exercise is really unusual to be threatened: it is easy to think that we are able to understand when a keylogger is running in our computer. However a keylogger could be execute in different ways: referring to the first exercise, the injected calculator might have launched a keylogger in background.

## Exercise 5 - Disclosing masked passwords

Nowadays we have many username and passwords to log in many sites. It's not always easy to remember all of them. Many browsers allow users to save their passwords, so they don't have to remember them all the times they need to access. To prevent malicious users from peeking the passwords they are not shown in clear text, but they are usually masked by some stars or dots. By the way the passwords are obviously saved somewhere in the computer. It's not easy to access them for a malicious user, but some programs allow to see the obscured passwords in clear text.

Snadboys Revelation present itself as a simple and free application for Windows with this functionality, so it allows to recover hidden passwords without fear of data or files loss and without any risk for the computer. This presentation does not highlight real risks in the application, in fact it is a benign view. On the other hand a malicious person could peek someone else's passwords using this tool.

## Exercise 6 - NetBus: Take control over another computer

Netbus is a software for remote control of a computer with a Microsoft Windows operating system through a network. It consists of two components, according to the client-server architecture. We need to execute a file (in this case patch.exe) on the victim's computer, that modifies the registers and launches the *server* every time the system is started. The component *client* is a separate program that has a graphical interface, which allows the user to perform a series of tasks on the remote computer: for example it is possible to open/close CD tray, swap mouse buttons, take screen dumps, switch off the victim's computer or keylogging.

We might think that this is just a joke between two friends or colleagues, but it it could also be used for malicious purposes and create big problems.

As good students, according with our ethics, we gently asked another group to open the server and give us their IP address. But an attacker might make an unaware person open the *patch.exe* file, with a phishing attack and catch his IP address in some other way. Again we find out that is not a good idea to open executable files from untrusted sources.

## Laboratory assignments for Kali Linux

### Exercise 1 - Utilising port and vulnerability scanners

A useful inspection activity is the scanning. Port scanning is a technique designed to probe a server or host in order to determine which ports are open. Vulnerability scanning is a general search for any weaknesses in a system. These activities could be performed by the system administrators in order to verify and improve their security system. On the other hand, they can be used by malicious people to discover any weak points in a target network.

These activities alone can't be considered harmful, but we could think that to learn about someone's vulnerabilities is the first step to plan an attack.

In this exercise we use two tools in order to test how secure the system is in an educational way. In particular we try Nmap and OpenVAS.

Nmap lets you know which ports or services are running and responding, what operating system is used, and what applications and which versions are installed. It might seem not significant but this information could help an hacker to plan a preciser attack, according to the vulnerabilities of the OS and the applications of the target.

We have scanned Nmap in different ways and systems (Linux and Windows). In order to provide a interesting output we have tried to run Netbus again in the Windows machine.

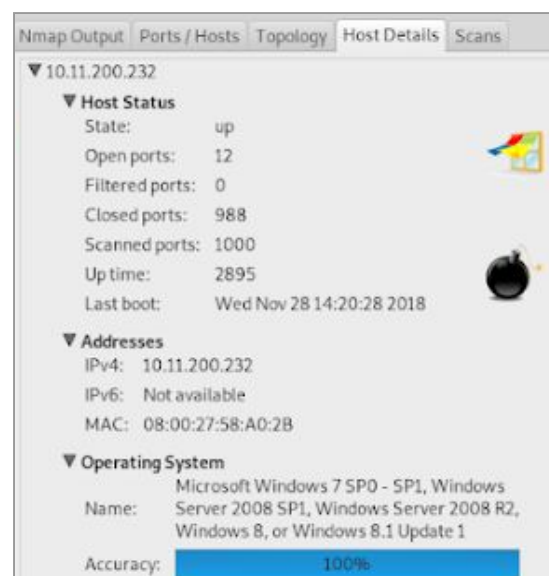
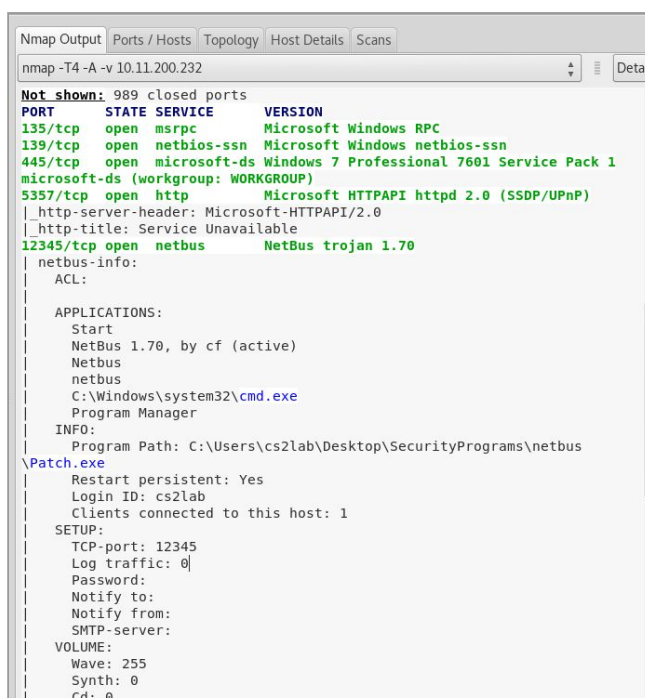


Image 1 - Open port Netbus in Windows

Image 2 - Host details in Windows

Generally, the open ports are highlighted in green. To provide an example of an output *Image 1* highlights that the port 12345 has been opened by netbus trojan. In *Image 2* we show the Host Details tab: Nmap provides a resume of the ports. It highlights the OS details, the addresses and the vulnerability status. There are 12 open ports and this is represented by the bomb icon (different icons are referred to different ranges of open ports), in the same way an icon is referred to the OS.

The second tool we used to perform vulnerabilities scanning is OpenVas. We decided to focus our results in the comparison of three attempts. In particular the first scan is referred to the Kali virtual machine that we used at the moment. The second one is the Windows virtual machine with Netbus running. The third one is a Metasploitable server. The results show the differences in vulnerabilities and security according to the type and the features of the machines.

The obtained outputs are resumed in *Image 3*. IP 10.11.200.171 refers to our Kali VM. It shows just a few log problems. They are not alarming, but it's worth to check them. It is reasonable to expect that result: this is the machine in use, we don't have strange open ports and it is relatively well updated and safe. IP 10.11.200.232 refers to our Windows VM with Netbus running. In this case we notice more log problems, and also three threats. We could reason about the comparison with the report of Nmap. The strange fact is that OpenVAS does not send a high alarm, instead Nmap reflects the threat as a true warning.

Host 10.11.200.171					
High	Medium	Low	Log	False	Positive
0	0	0	4	0	

Host 10.11.200.232					
High	Medium	Low	Log	False	Positive
0	2	1	11	0	

Host 10.11.9.38					
High	Medium	Low	Log	False	Positive
13	23	5	47	0	

Image 3 - OpenVAS resume

IP 10.11.9.38 belongs to the Metasploitable. It is clear that it's full of security weaknesses, and many of which appear to be serious. A Metasploitable is a virtual machine deliberately vulnerable based on the Linux OS: the distribution of these machines is born with the intent to carry out simulation of Penetration Testing for didactic purposes, so it is reasonable to think that this result is expected.

For each problem reported openVas provides a wide description, explaining how the problem was detected, what threats it entails, and how this problem can be solved. Sometimes it advises us to change a password from the default, or to update software. Other times the solution is more complex and an expert's intervention is necessary.

*Image 4* shows an example of the report of a threat detected on the Metasploitable server. In this case the threat presented is caused by the presence of the installation file *phpinfo.php*. This file could be attacked in order to catch some important information that it contains such as the IP address, the root directory of the web server and much more. OpenVAS provides a simple solution: delete this file that is no longer necessary instead it is just a risk.



<b>High (CVSS: 7.5)</b> NVT: <code>phpinfo()</code> output accessible (OID: 1.3.6.1.4.1.25623.1.0.11229)		80/tcp
<b>Summary</b> Many PHP installation tutorials instruct the user to create a file called <code>phpinfo.php</code> . This file is often times left in the root directory after completion.		
<b>Vulnerability Detection Result</b> The following files are calling the function <code>phpinfo()</code> which disclose potentially sensitive information to the remote attacker : <code>/phpinfo.php</code>		
<b>Impact</b> Some of the information that can be garnered from this file includes: The username of the user who installed php, if they are a SUDO user, the IP address of the host, the web server version, the system version(unix / linux), and the root directory of the web server.		
<b>Solution</b> Delete them or restrict access to the listened files.		
<b>Vulnerability Detection Method</b> Details: <code>phpinfo()</code> output accessible (OID: 1.3.6.1.4.1.25623.1.0.11229) Version used: \$Revision: 1157 \$		

Image 4 - High level issue example

On the other hand, if we don't want to delete it because we think it might be useful again or we are not expert, there is another solution. Change permission access is the answer in order to avoid the threat improving our security, without deleting the file.

## Exercise 2 - Utilising sniffer tools

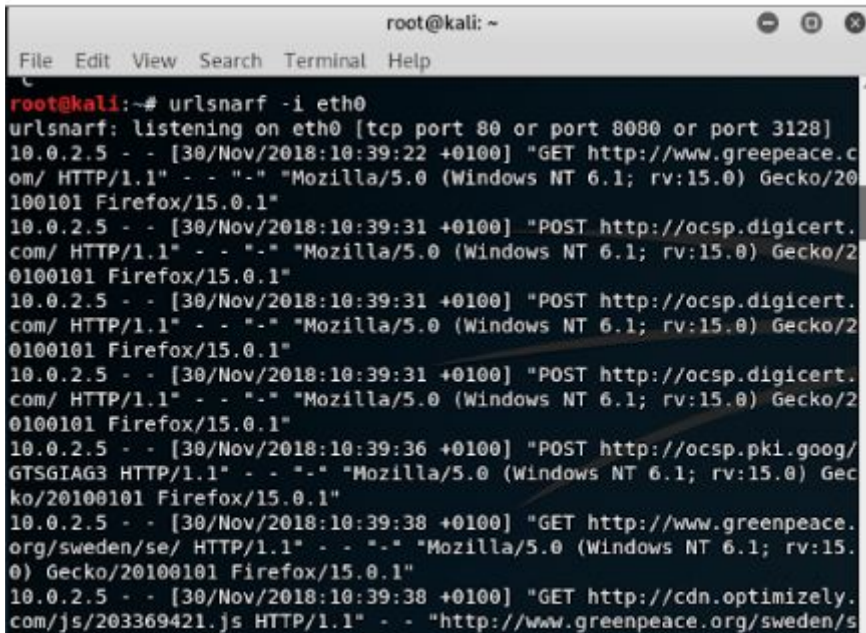
This part of the assignment shows how it is possible to sniff the network traffic with two tools: Dsniff suite and Ettercap. Sniffing consists in the interception of data passing through a telematic network. This activity can be performed for legitimate purposes, in order to solve intrusion problems or to detect intrusion attempts. In different circumstances sniffing can be used by a malicious user as well. For example an attacker could sniff an user's password, or spying the web traffic or emails.

For didactic purpose, we sniffed different targets:

- the VM of another group;
- our Windows VM;
- the Metasploitable VM.

The easiest function of the exercise is Urlsnarf: this activity provides us to know all the web traffic of another machine. So, when the other group browsed some sites our command line output showed all their web traffic.





```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# urlsnarf -i eth0  
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]  
10.0.2.5 - - [30/Nov/2018:10:39:22 +0100] "GET http://www.greepeace.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:31 +0100] "POST http://ocsp.digicert.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:31 +0100] "POST http://ocsp.digicert.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:31 +0100] "POST http://ocsp.digicert.com/ HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:36 +0100] "POST http://ocsp.pki.goog/GTSGIA63 HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:38 +0100] "GET http://www.greenpeace.org/sweden/se/ HTTP/1.1" - - "-" "Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1"  
10.0.2.5 - - [30/Nov/2018:10:39:38 +0100] "GET http://cdn.optimizely.com/js/203369421.js HTTP/1.1" - - "http://www.greenpeace.org/sweden/s
```

Image 5 - Urlsnarf to Windows VM

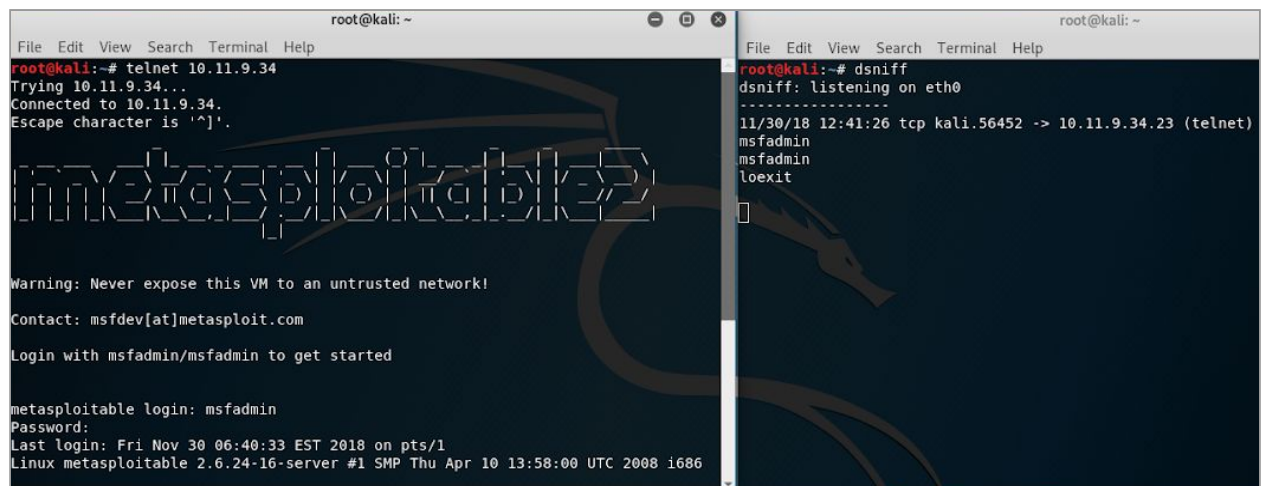
Image 5 shows part of the output of our sniff on the web traffic of our Windows VM.

The result is not so different from the previous one, in fact the other group traffic did not show anything surprising since it was an agreement way of generate web traffic.

All the websites we browse are reported. We notice that even when we browse one single website, many results are shown.

Our interpretation, as inexperienced users, is that when we browse a website we are redirected to many pages but we don't notice it. Some URL like <http://ocsp.digicert.com/> refer to the control of the permission and secure certificates of that specific site.

With Dsniff function to the Metasploitable machine we were able to get the user and password that we used to log in.



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# telnet 10.11.9.34  
Trying 10.11.9.34...  
Connected to 10.11.9.34.  
Escape character is '^]'.  
  
Warning: Never expose this VM to an untrusted network!  
Contact: msfdev[at]metasploit.com  
Login with msfadmin/msfadmin to get started  
  
metasploitable login: msfadmin  
Password:  
Last login: Fri Nov 30 06:40:33 EST 2018 on pts/1  
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686  
  
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# dsniff  
dsniff: listening on eth0  
-----  
11/30/18 12:41:26 tcp kali.56452 -> 10.11.9.34.23 (telnet)  
msfadmin  
msfadmin  
loexit
```

Image 6 - Dsniff to Metasploitable

In order to simulate a victim machine we launch a Metasploitable VM in a second command shell, meanwhile our machine is sniffing information. The image 6 shows the output: it indicates the keys that we used, in other words the user and the password.

Using webspy we can have on our browser in Kali a copy of the page shown in the sniffed browser in Windows. Such a tool allows a malicious user to spy on someone else's activity. There are many other functionalities of the dsniff package that can be used, on one hand for security purposes, and on the other hand for malicious intents.

The second tool, Ettercap, is a program that can be used as an alternative to dsniff. In fact, it offers the possibility of performing the same operations, providing the user with the expediency of interacting with a graphical interface. A sniffing attack can be performed with Ettercap in order to capture the target's user and password, another time the *Image 7* highlights this function for the Metasploitable VM login.

Anyway in particular, Ettercap is popular and known for being suitable for man in the middle attacks.



Image 7 - Ettercap example

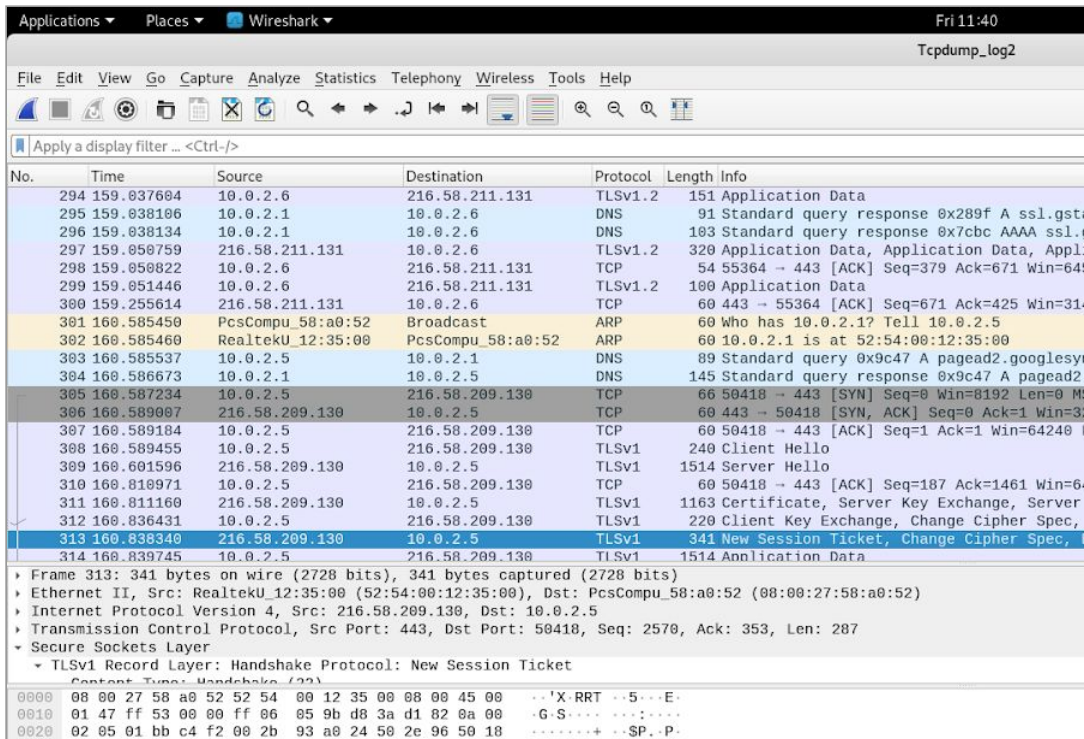
### Exercise 3 - Analysing network traffic

In many cases security incident are caused by insiders or people impersonating them. The majority of harms are not malicious, it could be that honest people are just making mistakes. On the other hand, there are potential malicious outsiders who passed firewalls and access controls. We need a system detecting during an incident that can't be prevented in advance. In particular it is useful analysing our network traffic. Typically we need a device, as another separate computer, that monitors activities to identify malicious or suspicious events.

During this part of the assignment we train in monitor our network traffic with three programs: Tcpdump, Wireshark and Snort.

Wireshark and Tcpdump are similar, but the first one has a graphical front-end, plus some integrated sorting and filtering options. In addition, although they can perform similar functions, Snort and Wireshark are not always interchangeable. Wireshark is a packet sniffer and analyser that allows you to capture and interpret network traffic. Snort is a complete open-source IDS (Intrusion Detection System). For this reason Snort can be used by a security expert in order to detect any suspicious traffic in the network, and to identify intruders. Snort scans for malicious (or other) patterns in packets it sees and alerts if it sees something suspicious.

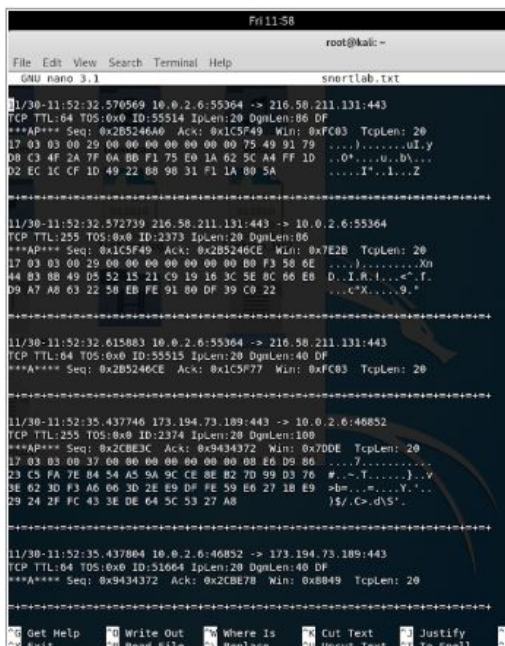
Wireshark reads packets and decodes them in “human readable format” for you to inspect whatever it is that happens in those packets. *Image 8* shows a screenshot of the Tcpdump log launched in our VM.



No.	Time	Source	Destination	Protocol	Length	Info
294	159.037684	10.0.2.6	216.58.211.131	TLSv1.2	151	Application Data
295	159.038186	10.0.2.1	10.0.2.6	DNS	91	Standard query response 0x289f A ssl.gstatic
296	159.038134	10.0.2.1	10.0.2.6	DNS	103	Standard query response 0x7cbc AAAA ssl.gstatic
297	159.050759	216.58.211.131	10.0.2.6	TLSv1.2	320	Application Data, Application Data, Application Data
298	159.050822	10.0.2.6	216.58.211.131	TCP	54	55364 → 443 [ACK] Seq=379 Ack=671 Win=6453
299	159.051446	10.0.2.6	216.58.211.131	TLSv1.2	100	Application Data
300	159.255614	216.58.211.131	10.0.2.6	TCP	60	443 → 55364 [ACK] Seq=671 Ack=425 Win=3145
301	160.585450	PcsCompu_58:a0:52	Broadcast	ARP	60	Who has 10.0.2.1? Tell 10.0.2.5
302	160.585460	RealtekU_12:35:00	PcsCompu_58:a0:52	ARP	60	10.0.2.1 is at 52:54:00:12:35:00
303	160.585537	10.0.2.5	10.0.2.1	DNS	89	Standard query 0x9c47 A pagead2.googlestatic
304	160.586673	10.0.2.1	10.0.2.5	DNS	145	Standard query response 0x9c47 A pagead2.googlestatic
305	160.587234	10.0.2.5	216.58.209.130	TCP	66	50418 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=
306	160.589087	216.58.209.130	10.0.2.5	TCP	60	443 → 50418 [SYN, ACK] Seq=0 Ack=1 Win=32768
307	160.589184	10.0.2.5	216.58.209.130	TCP	60	50418 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
308	160.589455	10.0.2.5	216.58.209.130	TLSv1	240	Client Hello
309	160.601596	216.58.209.130	10.0.2.5	TLSv1	1514	Server Hello
310	160.810971	10.0.2.5	216.58.209.130	TCP	60	50418 → 443 [ACK] Seq=187 Ack=1461 Win=64240
311	160.811160	216.58.209.130	10.0.2.5	TLSv1	1163	Certificate, Server Key Exchange, Server
312	160.836431	10.0.2.5	216.58.209.130	TLSv1	220	Client Key Exchange, Change Cipher Spec, E
313	160.838340	216.58.209.130	10.0.2.5	TLSv1	341	New Session Ticket, Change Cipher Spec, E
314	160.839745	10.0.2.5	216.58.209.130	TLSv1	1514	Application Data

Frame 313: 341 bytes on wire (2728 bits), 341 bytes captured (2728 bits) on interface eth0  
 Ethernet II, Src: RealtekU\_12:35:00 (52:54:00:12:35:00), Dst: PcsCompu\_58:a0:52 (08:00:27:58:a0:52)  
 Internet Protocol Version 4, Src: 216.58.209.130, Dst: 10.0.2.5  
 Transmission Control Protocol, Src Port: 443, Dst Port: 50418, Seq: 2570, Ack: 353, Len: 287  
 Secure Sockets Layer  
 TLSv1 Record Layer: Handshake Protocol: New Session Ticket  
 Content Type: Handshake (22)  
 0000 00 00 27 58 a0 52 52 54 00 12 35 00 08 00 45 00 ...X RRT --5--E  
 0010 01 47 ff 53 00 00 ff 06 05 9b d8 3a d1 82 0a 00 ...G S-----  
 0020 02 05 01 bb c4 f2 00 2b 93 a0 24 50 2e 96 50 18 .....+--SP..P

Image 8 - Tcpdump log



```

Fri 11:58
root@kali: ~
File Edit View Search Terminal Help
GNU nano 3.1 snortlab.txt

11/30-11:52:32.570569 10.0.2.6:55364 -> 216.58.211.131:443
TCP TTL:64 TOS:0x0 ID:55514 Iplen:28 Dglen:86 DF
***AP*** Seq: 0x2B5246A0 Ack: 0x1C5F40 Win: 0xFC03 TcpLen: 20
17 03 03 00 29 00 00 00 00 00 00 75 49 91 79 .....uLy
D8 C3 4F 2A 7F 0A 8B F1 75 E0 1A 62 5C A4 FF 1D ...0.....b\..<f
D2 EC 1C CF 1D 49 22 88 9B 31 F1 1A 80 5A .....I..1..Z

=====
11/30-11:52:32.572739 216.58.211.131:443 -> 10.0.2.6:55364
TCP TTL:255 TOS:0x0 ID:2373 Iplen:28 Dglen:86
***AP*** Seq: 0x1C5F40 Ack: 0x2B5246CE Win: 0x7E2B TcpLen: 20
17 03 03 00 29 00 00 00 00 00 00 75 49 91 79 .....uLy
D8 C3 4F 2A 7F 0A 8B F1 75 E0 1A 62 5C A4 FF 1D ...0.....b\..<f
D2 EC 1C CF 1D 49 22 88 9B 31 F1 1A 80 5A .....I..1..Z

=====
11/30-11:52:32.615883 10.0.2.6:55364 -> 216.58.211.131:443
TCP TTL:64 TOS:0x0 ID:55515 Iplen:28 Dglen:40 DF
***A*** Seq: 0x2B5246CE Ack: 0x1C5F77 Win: 0xFC03 TcpLen: 20

=====
11/30-11:52:35.437746 173.194.73.189:443 -> 10.0.2.6:46852
TCP TTL:255 TOS:0x0 ID:2374 Iplen:28 Dglen:100
***AP*** Seq: 0x2CBE3C Ack: 0x9434372 Win: 0x7DDE TcpLen: 20
17 03 03 00 29 00 00 00 00 00 00 75 49 91 79 .....uLy
D8 C3 4F 2A 7F 0A 8B F1 75 E0 1A 62 5C A4 FF 1D ...0.....b\..<f
D2 EC 1C CF 1D 49 22 88 9B 31 F1 1A 80 5A .....I..1..Z

=====
11/30-11:52:35.437804 10.0.2.6:46852 -> 173.194.73.189:443
TCP TTL:64 TOS:0x0 ID:51664 Iplen:28 Dglen:40 DF
***A*** Seq: 0x9434372 Ack: 0x2CBE78 Win: 0x8B49 TcpLen: 20

```

Image 9 - Snort output

The graphic interface lets us understand something from the output. Different colours are associated with different protocols to help user identify the types of traffic.

It is also possible to know which source generated the traffic and which destination received it. Moreover, Wireshark has more powerful features, for example we were able to view the reconstructed stream of a TCP session.

When we use Snort the same information is provided on the shell. It's only harder to read for a user. Probably is most suitable for automated tasks performed by a program. Anyway, *Image 9* shows the output of Snort running in our network.



In this exercise it resulted difficult to understand an example of harm but in order to catch an example of use we could think of a real situation. A company network usually analyze its network traffic: it is possible that an associated employee used company resources for downloading torrents. In that case a packet with the connected source IP address could be identified with the protocol BitTorrent. Thinking about the scenario it is reasonably assumed that the IP address associated with the BitTorrent user can be someone who is knowingly downloading and possibly uploading files using a P2P exchange client. The situation could be a point of vulnerability for the network, in fact it is known that many torrents are related to a transmission of malware. Therefore we think that it is important to educate and train employees about the policies that a company considers significant. On the other hand it is better to prevent unpleasant situations: there are ways of configuring network devices in order to block or reduce P2P file sharing (always thinking to not have a bad impact on the productivity of the company).

## Exercise 4 - Metasploit: h4Xing made easy

This aim of this exercise is to use an attack tool, the Metasploit, to take advantages of a vulnerability present in an application in Windows 7 system. We used version 2.2rc2, intentionally not updated, because it contains bugs that make it vulnerable. In particular we searched on Internet trying to understand that vulnerability. *Image 10* shows the result of our search: it seems that the point is a possible attack to SQL commands in application's database.

Copy

Advisory ID: HTB23181  
Product: Dokeos  
Vendor: Dokeos  
Vulnerable Version(s): 2.2 RC2 and probably prior  
Tested Version: 2.2 RC2  
Advisory Publication: October 30, 2013 [without technical details]  
Vendor Notification: October 30, 2013  
Public Disclosure: November 27, 2013  
Vulnerability Type: SQL Injection [CWE-89]  
CVE Reference: CVE-2013-6341  
Risk Level: High  
CVSSv2 Base Score: 7.5 (AV:N/AC:L/Au:N/C:P/I:P/A:P)  
Solution Status: Solution Available  
Discovered and Provided: High-Tech Bridge Security Research Lab ( <https://www.htbridge.com/advisory/> )

-----

Advisory Details:  
  
High-Tech Bridge Security Research Lab discovered vulnerability in Dokeos, which can be exploited to perform SQL Injection attacks.  
  
1) SQL Injection in Dokeos: CVE-2013-6341  
  
The vulnerability exists due to insufficient validation of "language" HTTP GET parameter passed to "/index.php" script. A remote unauthenticated attacker can execute arbitrary SQL commands in application's database and gain complete control over the vulnerable web application.

Image 10 - Windows 2.2 RC2 vulnerability [source: <https://www.exploit-db.com/>]

Referring to OpenVAS scanning report in Kali Linux Exercise 1, we checked vulnerabilities in Windows VM: we can't find that weakness in our report.

The exercise made us attacking a victim Windows OS. We used Armitage in order to establish a connection with the victim. Meterpreter allows us to open a command shell that points on the target machine. As shown in *Image 11*, to create a proof of our hack we produce a text file *hack.txt* on the Desktop following the instructions. Of course this didn't affect the victim's computer. By the way

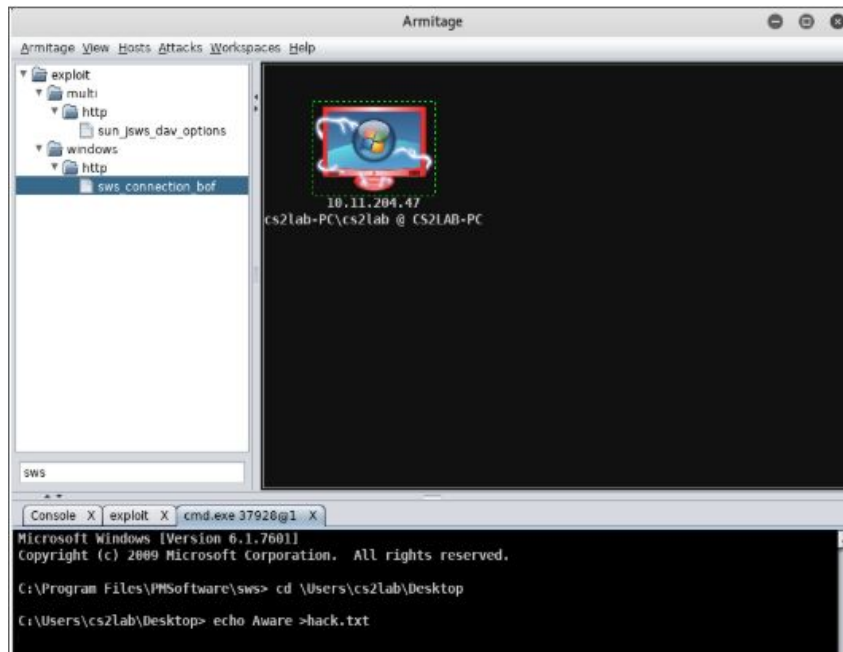


Image 11 - Armitage attack

accessing the command line gives the possibility to do almost anything. A vulnerability, as this one, could be exploited by an attacker in order to cause serious problems.

Of course in this case the victim's computer had a weakness that was let accessible intentionally. But if a network is not well secured an expert attacker could always find a vulnerability to plan an attack.

For this reason it's always important to check the network security to detect any vulnerability, and solve them.

## Conclusion

The assignments provides us a training to introduce in hacking and in security systems tools. The paper is divided in two parts reflecting the different OS of the virtual machines with the corresponding exercises.

The first part highlights how it is not safe to open executable files provided by untrusted sources and let us practicing in access control and permission problems. The second part focuses on identifying weaknesses such as open ports, vulnerabilities but also warnings in the network traffic, and on the other hand, try to show how to sniff and attack a victim machine.

Most (if not all) of the tools we used were developed for security purposes. By the way these software can also be used with malicious intents in order to perform different kinds of attacks.

Check the presence of vulnerability of your network, and updating the software is always good to keep your system secured.