TÉCNICO LISBOA

RC 19/20	LAB ASSIGNMENT	Guide.Part #:	1.2
Application Layer		**Issue Date:**	30 Sep 2019
Network Tools, e-mail (SMTP, POP3)		**Due Date:**	
Author: Prof. Rui Cruz		**Revision:**	4.0

1 Introduction

The objective of the experiments in this Lab is to use Network Tools, such as a **ifconfig**, **ping**, **traceroute**, **whois** and **nmap** and the e-mail protocols **SMTP** and **POP3**.

For that purpose, a Vagrant environment with a Ubuntu server will be created, provisioned and configured with the necessary Network Tools.

Preliminary Notes

The instructions in this document are applicable to Computers at the IST Labs.

Nevertheless, one nice feature of the software stack we are going to use is that it is portable to many platforms including **YOUR OWN** personal computers, running the following Operating Systems:

- Microsoft Windows from version 10 up
- Apple macOS from versions 10.13 'High Sierra' up
- Debian-based Linux, such as Ubuntu (recommended) from versions 12.04 'Precise' up.

It is not recommended to apply this setup to a virtual machine (nested virtualization), although possible, as the configuration requires access to a hypervisor environment (recommended Virtualbox) in the host system.

Note: Avoid copying text strings from the command line examples or configurations in this document, as pasting them into your system or files may introduce/modify some characters, leading to errors or inadequate results.

2 Spin off of the Lab Environment with Vagrant

To start, go to your project directory (as explained in previous Lab) and create a new folder named, for example, **nethack**. Proceed with the following steps:

1. Download from the course website the file `RC_19_20_LAB_01_P2_support_files.zip` and uncompress the content to the **nethack** project folder.
2. Verify that a `Vagrantfile` and a file named `bootstrap_nethack.sh` exist in that folder.

In that **nethack** folder, edit if needed, the `Vagrantfile` in the block that defines the "nethack" VM to suit the environment in your host system. Confirm that the content of the `Vagrantfile` is similar to the following:

```
config.vm.define "nethack" do |nethack_config|
  ....
  nethack_config.vm.provider "virtualbox" do |vb|
      ....
  end # of vb
  nethack_config.vm.provision "shell", path: "bootstrap_nethack.sh
     "
end # of nethack_config
```

For Microsoft Windows hosts, you need to start the "**xming**" application before spin off the lab environment. The "**xming**" application will start a **X-Windows** server that will forward application GUIs running inside the VM to the host system.

For the experiments in this lab you will use **Wireshark** (A network Sniffer and Protocol dissector tool, provisioned and configured in the VM) to capture the packets in the network interface in order for you to analyse them. For that purpose, you will need to have **two Terminal windows opened**. On the first window you will start Wireshark with the following command:

```
$ vagrant ssh nethack -c wireshark -- -X
```

In the second Terminal window start a **ssh** connection with the machine:

```
$ vagrant ssh
```

3 Network Tools Experiment

These experiments are valid for all the systems (Windows, Mac and Linux) if they have the required tools installed. We will use the "nethack" VM to perform the experiments.

3.1 ifconfig

In this section we will list and analyse the network interfaces in our system. First use the `man` command to know a bit more about `ifconfig`.

```
vagrant@nethack:~$ man ifconfig
```

By issuing the command `ifconfig` we will get a detailed list of the currently active network interfaces of the machine. By giving the interface a name as argument, the details of that specific interface will be displayed (similar to the example below).

```
vagrant@nethack:~$ ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 08:00:27:44:69:7b
          inet addr:192.168.1.220  Bcast:192.168.1.255  Mask
             :255.255.255.0
```

```
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11130 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3740 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1023325 (1.0 MB)  TX bytes:243430 (243.4 KB)
```

The command `ifconfig` is also used to manage these interfaces, and so we can enable and disable interfaces with the command parameters and set those interfaces to **promiscuous** mode (allowing to "sniff" or "dump" all the packets passing in that interface).

3.2 ping

You will use `ping` to try to have responses from other hosts. Try to reach `www.wustl.edu` and discover what is its IP address.

NOTE: While ping is working, inspect what is happening in Wireshark (after starting Capturing).

```
vagrant@nethack:~$ ping www.wustl.edu
PING wordpress-prod.g.wustl.edu (128.252.114.30) 56(84) bytes of
   data
64 bytes from 128.252.114.30: icmp_seq=1 ttl=235 time=122 ms
64 bytes from 128.252.114.30: icmp_seq=2 ttl=235 time=122 ms
64 bytes from 128.252.114.30: icmp_seq=3 ttl=235 time=122 ms
64 bytes from 128.252.114.30: icmp_seq=4 ttl=235 time=122 ms
```

Use `man` command for `ping` to know better the command options and usage. For example, the flag `-t ttl` is used to set the IP Time to Live (TTL), that represents the number of *hops* that the IP packet is allowed to traverse. Try to find how many *hops* are needed for `ping` reach the host with the IP address of $193.136.128.169$.

```
vagrant@nethack:~$ ping -t 2 193.136.128.169
```

3.3 traceroute

This tool `traceroute`, or equivalent ones such as `tracepath`, are used as a network diagnostic tool to display the route (path) up to the destination, and measures the transit delays of packets across the network (the Internet).

NOTE: While the command is working, inspect what is happening in Wireshark.

Use `man` command to learn more about `traceroute` and use this tool to trace the path to `www.wustl.edu`, as exemplified:

```
vagrant@nethack:~$ man traceroute

vagrant@nethack:~$ traceroute www.wustl.edu
1  *  *  *
2  core22.fsn1.hetzner.com (213.239.245.241)  5.043 ms  5.052 ms
2  core21.fsn1.hetzner.com (213.239.245.237)  0.206 ms
3  core1.fra.hetzner.com (213.239.245.177)  4.876 ms
3  core1.fra.hetzner.com (213.239.245.218)  5.134 ms
3  core11.nbg1.hetzner.com (213.239.245.221)  2.796 ms
4  juniper4.dc2.nbg1.hetzner.com (213.239.203.138)  2.887 ms  2.891 ms  2.900 ms
5  ae1-710.fra20.core-backbone.com (80.255.15.121)  10.392 ms
5  ae51.bar2.Munich1.Level3.net (62.140.25.101)  5.428 ms
ae1-710.fra20.core-backbone.com (80.255.15.121)  10.392 ms
6  ffm-b4-link.telia.net (213.248.81.209)  5.705 ms  *
7  xe-0-0-2-sliac-core.nts.wustl.edu (4.28.92.178)  us  117.329 ms
7  be3031.agr41.fra03.atlas.cogentco.com (130.117.14.197)  8.127 ms  8.135 ms
8  be3187.ccr42.fra03.atlas.cogentco.com (130.117.1.118)  8.112 ms
8  xe-0-0-5-bih-1017-wu-vrt-0.nts.wustl.edu (128.252.1.253)  116.163 ms
8  be3186.ccr41.fra03.atlas.cogentco.com (130.117.0.1)  8.375 ms
9  eth5-23-eps-core.nts.wustl.edu (128.252.1.63)  121.056 ms  120.912 ms  120.930 ms
10 be2182.ccr21.lpl01.atlas.cogentco.com (154.54.77.246)  24.827 ms
10 be2183.ccr22.lpl01.atlas.cogentco.com (154.54.58.69)  24.828 ms
10 po52-wcdc-core-127-0.nts.wustl.edu (128.252.254.141)  122.970 ms
11 eth4-5-wcdc-agg-126-0.nts.wustl.edu (128.252.100.225)  123.690 ms
11 be3042.ccr21.ymq01.atlas.cogentco.com (154.54.44.162)  93.942 ms
11 eth4-5-wcdc-agg-126-0.nts.wustl.edu (128.252.100.225)  123.690 ms
```

We can emulate the `traceroute` tool using `ping` with incrementing values for TTL parameter.

There is a Web implementation of this tool (Figure 1). Using the browser in your host system go to `http://ping.eu/traceroute` and try to trace the route to the same host. Compare the outputs of both implementations of this tool.

3.4 whois

Another interesting tool is `whois`, a tool that allows us to get informative content about entities on the Internet.

Given a **IP address** or a **Domain Name** we can get some information by running the following (example):

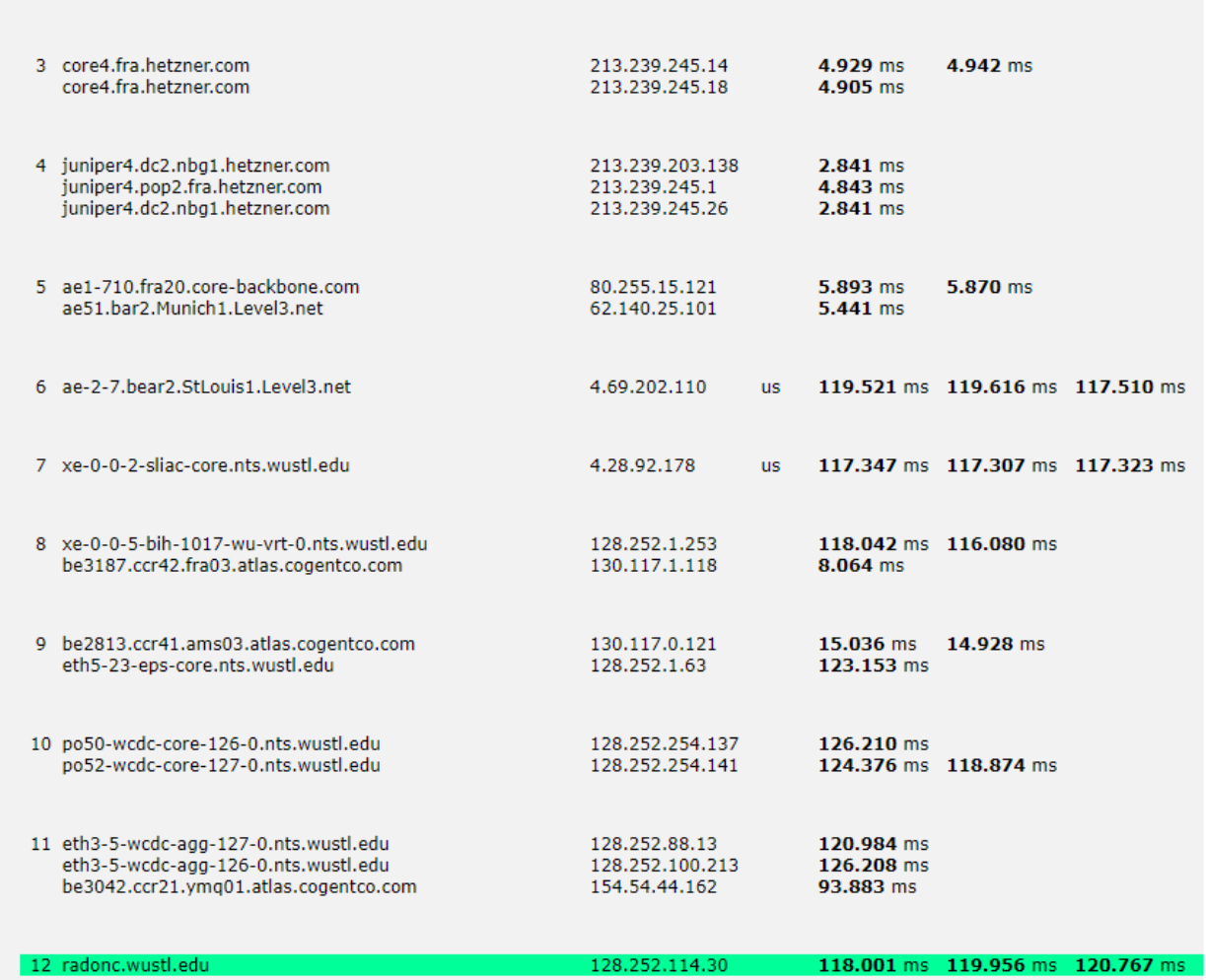
```
vagrant@nethack:~$ whois 193.136.128.169
```

NOTE: Could you identify the entity who owns a server with that IP Address?

You can also find a web implementation of `whois` in `http://ping.eu/ns-whois/`.

4 nmap

`nmap` ("Network Mapper") is a tool for network discovery and security auditing (or hacking). `nmap` uses raw IP packets in peculiar ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, the type of packet filter/-firewall in use, and dozens of other characteristics. `nmap` is the basis for most security enumeration during the initial stages of a penetration test (ethical hacking).



3	core4.fra.hetzner.com core4.fra.hetzner.com	213.239.245.14 213.239.245.18		**4.929** ms **4.905** ms	**4.942** ms	
4	juniper4.dc2.nbg1.hetzner.com juniper4.pop2.fra.hetzner.com juniper4.dc2.nbg1.hetzner.com	213.239.203.138 213.239.245.1 213.239.245.26		**2.841** ms **4.843** ms **2.841** ms		
5	ae1-710.fra20.core-backbone.com ae51.bar2.Munich1.Level3.net	80.255.15.121 62.140.25.101		**5.893** ms **5.441** ms	**5.870** ms	
6	ae-2-7.bear2.StLouis1.Level3.net	4.69.202.110	us	**119.521** ms	**119.616** ms	**117.510** ms
7	xe-0-0-2-sliac-core.nts.wustl.edu	4.28.92.178	us	**117.347** ms	**117.307** ms	**117.323** ms
8	xe-0-0-5-bih-1017-wu-vrt-0.nts.wustl.edu be3187.ccr42.fra03.atlas.cogentco.com	128.252.1.253 130.117.1.118		**118.042** ms **8.064** ms	**116.080** ms	
9	be2813.ccr41.ams03.atlas.cogentco.com eth5-23-eps-core.nts.wustl.edu	130.117.0.121 128.252.1.63		**15.036** ms **123.153** ms	**14.928** ms	
10	po50-wcdc-core-126-0.nts.wustl.edu po52-wcdc-core-127-0.nts.wustl.edu	128.252.254.137 128.252.254.141		**126.210** ms **124.376** ms	**118.874** ms	
11	eth3-5-wcdc-agg-127-0.nts.wustl.edu eth3-5-wcdc-agg-126-0.nts.wustl.edu be3042.ccr21.ymq01.atlas.cogentco.com	128.252.88.13 128.252.100.213 154.54.44.162		**120.984** ms **126.208** ms **93.883** ms		
12	radonc.wustl.edu	128.252.114.30		**118.001** ms	**119.956** ms	**120.767** ms

Figure 1: Partial output of web version of traceroute

NOTE: While the command is working, inspect what is happening in Wireshark.

For example, the following command scans the network, listing machines that respond to ping:

```
vagrant@nethack:~$ nmap -sP 192.168.1.0/24
```

The following command scans all reserved TCP ports on the machine `scanme.nmap.org`. The -v option enables verbose mode:

```
vagrant@nethack:~$ nmap -v scanme.nmap.org
```

Now you know new tools to explore networks, more specifically, the Internet.

5 e-mail Experiments

In these experiments we will play with e-mail protocols Simple Mail Transfer Protocol (SMTP) and Post Office Protocol (POP3), illustrated in Figure 2.

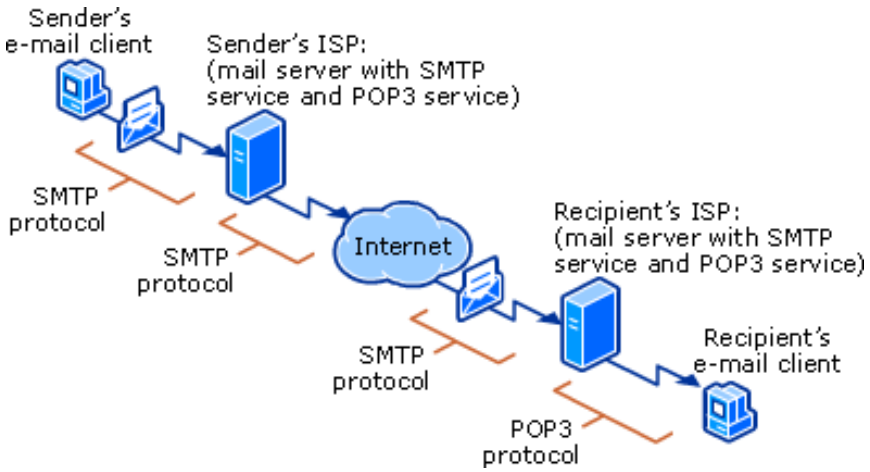


Figure 2: Typical working of e-mail protocols SMTP and POP3

NOTE: While the commands are working, inspect what is happening in Wireshark.

5.1 Connecting to a SMTP server

SMTP is the universally used protocol for email transfer. For our SMTP experiment we will use an external SMTP test server provided for study purposes, the "smtp.mailtrap.io". Try the following interaction (note that the user credentials to be used are those in the example), where your inputs are represented with color orange, and use Wireshark to inspect the packets exchanged.:

```
vagrant@nethack:~$ telnet smtp.mailtrap.io 2525
Trying 54.85.222.127...
Connected to mailtrap.io.
Escape character is '^]'.
220 mailtrap.io ESMTP ready
EHLO smtp.mailtrap.io
250-mailtrap.io
250-SIZE 5242880
250-PIPELINING
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-DSN
```

```
250-AUTH PLAIN LOGIN CRAM-MD5
250 STARTTLS
AUTH LOGIN
334 VXNlcm5hbWU6
ZTM5OTNkZDg1ZWVhNDU=
334 UGFzc3dvcmQ6
MDU1ODgxNTAxOTQ4YWY=
235 2.0.0 OK
MAIL FROM: <from@smtp.mailtrap.io>
250 2.1.0 Ok
RCPT TO: <to@smtp.mailtrap.io>
250 2.1.0 Ok
DATA
354 Go ahead
To:  to@smtp.mailtrap.io
From:  from@smtp.mailtrap.io
Subject:  Hello world!
This is the test message...
.
250 2.0.0 Ok: queued
quit
221 2.0.0 Bye
Connection closed by foreign host.
```

5.2 POP3 Experiments

POP3 is an email access client application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection. Despite this protocol being less and less used, it is simple and is one of the most adequate ones to illustrate the mechanisms used in the email service. Try the following interaction (note that the user credentials to be used are those in the example), where your inputs are represented with color orange, and use Wireshark to inspect the packets exchanged.

```
vagrant@nethack:~$ telnet smtp.mailtrap.io 1100
Trying 54.85.222.127...
Connected to mailtrap.io.
Escape character is '^]'.
+OK POP3 ready
USER e3993dd85eea45
+OK
PASS 055881501948af
+OK maildrop locked and ready
STAT
+OK 0 0
```

```
LIST
+OK 0 messages (0 octets)
.
QUIT
+OK Bye
Connection closed by foreign host.
```

6 Finishing your Experiments

In order to stop the Virtual Machine and to verify the global state of all active Vagrant environments on the system, we can issue the following commands:

```
:~$ vagrant halt
==> nethack: Attempting graceful shutdown of VM...
:~$ vagrant global-status
```

Confirm that the statuses of the VMs is 'powered off'. In order to prevent those instantiated machines to use resources, you can destroy them, as they can now be recreated with that simple command `vagrant up`. Confirm that there are no VMs listed.

```
:~$ vagrant destroy
nethack: Are you sure you want to destroy the 'nethack' VM? [y/N]
==> nethack: Destroying VM and associated drives...
:~$ vagrant global-status
```

7 Submitting the Results of the Experiments

The experiments that you execute in this LAB will either produce results that you need to report or from which you will be asked questions about the execution. In order to report the results you achieved, proceed as follows:

7.1 General Procedure

The procedure for submission is quite simple:

1. In your Google Classroom for this course, you will find an Assignment for reporting this specific Lab experiment (with a Due Date);

2. The Assignment provides a Link to the TSUGI CLOUD Web Form with Exercise Questions;

3. When you are prepared with the requested materials (screen-shots, command line outputs, developed code, etc.) you will submit the items into the respective Exercise Form questions of the Assignment;

4. In the same Exercise Form of the Assignment you may be asked to comment your submissions;

5. Please note that in some types of Exercises you may also be asked to provide feedback (anonymously) to the answers from some of your classmates, classifying with "points" the level of accomplishment of your classmate answers.

6. When finished answering the Exercise Form, click the button **Done** in the top left of the Form; You will be returned to the Google Classroom Assignment;

7. In the Assignment, do not forget to confirm (click the button) **MARK AS DONE** or **TURN IN** when the assignment is completed;

7.2 Specific Procedure

For this LAB Assignment you will provide results from using the Network Tools, as well as from interacting with SMTP amd POP3. Additionally some screenshots are also needed, as follows:

1. Capture a screenshot of the command `ifconfig` with information about the ethernet interface with the private addressing, in the **nethack** machine, to submit in the TSUGI Form where asked;

2. Capture the result from the traceroute command to `www.wustl.edu`, to submit in the TSUGI Form where asked;

3. Capture the result of the whois command to IP address $193.136.128.169$, to submit in the TSUGI Form where asked;

4. Capture a screenshot of the results of the second `nmap` command, to submit in the TSUGI Form where asked;

5. Copy from the Terminal window the text resulting from the interaction with the SMTP server, to submit in the TSUGI Form where asked;

6. Capture a screenshot of the Wireshark window, showing the packets exchanged with the interaction via POP3 with the e-mail server, to submit in the TSUGI Form where asked;

WARNING. Submissions MUST BE MADE in the TSUGI CLOUD Web Form through Classroom. No other type of submission will be considered.