Setup

NETMET Lab Exercises 1

Prelude

Hi everyone! Welcome to Internet Measurements Labs.

These labs will cover three main topics related to internet measurements: Topology (8 courses), IP Geolocation (4 courses) and Security (2 courses).

There is multiple goals for you to follow the labs:

- Get your hands on practical measurements on the internet. Learn more about the network concepts that allow us to collect information about the internet.
- Learn how to code simple programs (in Python) to create your own measurements
- Because Internet measurements are an active research field, learn how to get state-of-the-art information from research papers efficiently.

Don't worry about the code, you will not be evaluated on your coding skills. Nonetheless, I think coding is a fundamental tool of computer scientists / IT experts, even for network-oriented works. These labs are a great opportunity for you to improve your programming skills!

We will sometimes need to read some papers in order to understand state-of-the-art techniques or analysis related to network measurements. This may not be something you are used to doing, thus it can be a bit complicated at first. I suggest you to read this little document that give you tips to read a paper more easily and more effectively : https://web.stanford.edu/class/ee384m/Handouts/HowtoReadPaper.pdf

Introduction

In this lab we will get our hands on the measurement platform that we will use throughout the labs. This platform is called EdgeNet and you can find more information about it here: https://www.edge-net.org/.

The main thing to know about this platform is that it is based on **Kubernetes** technology. Thus, we will use the Kubernetes CLI (kubectl) to interact with it, i.e., create our vantage points, connect into it, etc …

The first thing to do is to install kubectl. You will find a tutorial here that depends on your operating system (Windows, MacOs, Linux). Please go ahead and install it on your personal computer. If you don't bring one, you can use the lab's computers.

Register yourself to the platform

Now we are able to interact with EdgeNet, let's register into it.
You simply have to follow this procedure.
The tenant name should be "lip6-lab" (the same as in the procedure).

Deploy a vantage point

A vantage point is a server (bare-metal, VM, container, …) with an internet connection where we will be able to do measurements. When doing Internet Measurements research, it's often the most complicated resource to get.

For these labs we won't have the problem of finding available vantage points thanks to the EdgeNet platform.

First let's list the available serveurs available at the moment on the platform. In the Kuberbernets terminology, a server is called a "node".

```
kubectl get nodes --kubeconfig edgenet-kubeconfig.cfg
```

You should see many nodes, with some with the status "Ready" while others "NotReady". One of them is called "edgenet.planet-lab.eu" and has the role "control-plane,master"; we mustn't use it to create our nodes.

On the other nodes, we can deploy multiple containers. For simplicity, let's call these containers "pods" or "vantage-point".
Let's create a pod on a node that is in a ready state. Let's pretend this node is called "**geni-us-ca-551d.edge-net.io**" (take another one for your test).

```
kubectl run <YOUR_NAME> --image=praqma/network-multitool:extra --overrides='{
"apiVersion": "v1", "spec": { "nodeSelector": { "kubernetes.io/hostname":
"geni-us-ca-551d.edge-net.io" }}}' --requests='cpu=100m,memory=512Mi' --kubeconfig
./edgenet-kubeconfig.cfg
```

You should see your pod with this command:

```
kubectl get pods --kubeconfig edgenet-kubeconfig.cfg
```

Now you have create a pod, you can enter into it with this command:

```
kubectl exec -it <YOUR_NAME> --kubeconfig ./edgenet-kubeconfig.cfg -- /bin/bash
```

Ping

The ping program can measure the round trip time (RTT) between a computer and another computer on the Internet.

You should see an output like this:

```
$ ping google.com
PING google.com (216.58.208.206) 56(84) bytes of data.
64 bytes from par10s21-in-f206.1e100.net (216.58.208.206): icmp_req=1 ttl=52 time=1.88 ms
64 bytes from par10s21-in-f206.1e100.net (216.58.208.206): icmp_req=2 ttl=52 time=2.43 ms
64 bytes from par10s21-in-f206.1e100.net (216.58.208.206): icmp_req=3 ttl=52 time=2.44 ms
64 bytes from par10s21-in-f206.1e100.net (216.58.208.206): icmp_req=4 ttl=52 time=1.99 ms
64 bytes from par10s21-in-f206.1e100.net (216.58.208.206): icmp_req=5 ttl=52 time=2.45 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 1.884/2.241/2.455/0.258 ms
```

In this example we saw that 5 ping packets were sent, and ping shows the RTT for each packet (for example `time=1.88 ms`).
In the last two lines there are statistical averages for all 5 ping packets. It shows the packet loss that occurred, and the minimum, average and maximum RTT. `mdev` is the average standard deviation.

Traceroute

Traceroute is a program that can show us which way or path a packet takes from our computer to another computer. For example, it can show us how a packet goes from our computer to the servers of google.

You should see an output like this:

```
$ traceroute google.com
traceroute to google.com (216.58.208.206), 30 hops max, 60 byte packets
 1  132.227.62.65 (132.227.62.65)  0.542 ms  0.203 ms  0.643 ms
 2  10.1.1.1 (10.1.1.1)  1.163 ms  1.154 ms  0.855 ms
 3  r-interco1-lab-upmc.reseau.jussieu.fr (134.157.167.125)  37.032 ms  36.733 ms  36.429
ms
 4  r-jusrap-reel.reseau.jussieu.fr (134.157.254.124)  1.148 ms  1.590 ms  1.298 ms
 5  interco-6.01-jussieu.rap.prd.fr (195.221.127.181)  0.609 ms  0.663 ms  1.077 ms
 6  vl165-te3-2-jussieu-rtr-021.noc.renater.fr (193.51.181.102)  1.551 ms  1.045 ms  1.108
ms
 7  te0-1-0-3-paris2-rtr-001.noc.renater.fr (193.51.177.114)  63.681 ms  72.526 ms  72.589
ms
 8  * * *
 9  72.14.214.160 (72.14.214.160)  1.652 ms  2.115 ms  1.752 ms
10  72.14.238.228 (72.14.238.228)  2.125 ms  1.747 ms  2.150 ms
11  209.85.143.241 (209.85.143.241)  2.271 ms  1.927 ms  1.948 ms
12  par10s21-in-f206.1e100.net (216.58.208.206)  1.603 ms  1.624 ms  2.351 ms
```

In this example we can see that there are 12 computers (also called “hops”) between our computer and the server of google. The three numbers in each line (for example `0.542 ms`

`0.203 ms  0.643 ms`) show that traceroute sends 3 probes to each computer on the way. The `*` means that the hop did not respond to the probe.

Playing with ping and traceroute

- Use ping and write down the obtained values (average round trip time, standard deviation, packet loss rate etc.). Also use traceroute and write down the number of hops to the target. Use different targets as well as different vantage points.
- Make a plot with the data you got (for example using Google Sheets, Excel). What can you see?

Clean-up and conclusion

Before forget it, let's remove our vantage point:

kubectl delete pod **<YOUR_NAME>** --kubeconfig ./edgenet-kubeconfig.cfg

Perfect.
This closes up the first lab. You now have all the keys to be able to do measurements of the next labs in perfect autonomy.
Furthemore, you manipulated the most fundamental tools of internet measurement and had a glimpse of what we can already infer with them.