

MANAGEMENT PROJECT

2019

**David Carrascal Adrián Guerrero**

**Pablo Collado Artem Strilets**

INDEX

**INTRODUCTION**3

**MININET AND CONFIGURATION SCENARIO**4

1. Vagrant file installation4

1.1. Troubleshooting problems regarding SSH 5

2. Native File Installation6

**GRAFANA CONFIGURATION**7

**INFLUXDB CONFIGURATION** 7

**TELEGRAF CONFIGURATION** 7

# INTRODUCTION

A network scenario is planned with Mininet. For this, we think of a simple triangle-shaped network in which 2 virtual machines will be used in which this scenario will be implemented, with "Test" and "Controller" being the chosen names.

Therefore, we will need a virtual environment to work. In this way, we will use Mininet at all times to carry out this project.

In addition, we will carry out a monitoring infrastructure that will be based on Grafana, together with InfluxDB and Telegraf, so that we can run the telegraph on the "Test" machine and InfluxDB together with Grafafa on the "Controller" machine. In this way, we will have total control over our network.

Also, we will choose two tools to recreate a DDoS attack (denial of service attack). Being a simple and virtual network, we will generate these attacks ourselves and verify through Grafana how this data is recorded. To do this, we will use "ping" and "hping3", which are two tools whose purpose is the analysis and assembly of TCP / IP packets

On the one hand, using the InfluxDB (Python API) interface, we will create a script that implements an artificial intelligence algorithm that determines whether or not we are under a DDoS attack or in a normal traffic situation through a classification.

On the other hand, we will see how we can import the output of the script that decides if a DDoS attack is being executed in the Grafana panel, to verify it and generate an alarm if it is in this situation.

We have worked using GitHub for the development of the project since we can have a version control and be able to work each without problems, each carrying out a part of the project and sharing it with the rest of the group.

**MININET AND CONFIGURATION SCENARIO**

For the correct operation of our project, we need a scenario that, to mount it, we have previously created a Vagrantfile through which all the scripts necessary for the installation and configuration of each of the machines are provisioned.

When working in a virtualized environment, we ensure that all machines have the same specifications and configuration to help us in the event of an error. Thus, it helps us to have the most localized tracking and correction of errors and we can more easily trace the traceability of their cause.

In addition, in case you do not want to use the Vagrant installation method as a provider, you can follow the native installation method that we present below

### Vagrant file installation

For the installation of Vagrant, we will need to have this program installed in our local machine in order to work. To do this, we will download the executable for Windows from the official website

https://www.vagrantup.com/downloads.html

Once downloaded and installed, we will proceed to install VirtualBox, version 6.0. Although we can currently have higher versions, as it is today 6.1, Vagrant is not implemented for this new version, so we will proceed to download an "old builds" from VirtualBox.

https://www.virtualbox.org/wiki/Download\_Old\_Builds\_6\_0

To have a simpler display, we can use either the console that Windows offers us or download a console emulator, such as Cmder.

After having the main files downloaded, for the configuration and installation of our project, we make a clone of our repository

git clone https://github.com/GAR-Project/project

cd project

In this way, we will have in our local machine the necessary files for the realization of the project.

We enter the “Project” folder and raise the virtual machine

vagrant up

What it does is download, install, configure and start the virtual machine. Once the command is finished, we will have the virtual machine started and ready to work. Now we only have access to these. To do this, we connect to virtual machines using ssh

vagrant ssh test

vagrant ssh controller

We should already have all the virtual machines configured with all the necessary tools to lift our network scenario using Mininet in the virtual machine "test" and Ryu in the virtual machine "controller".

### Troubleshooting problems regarding SSH

If the connection by SSH to the virtual machine is a problem, either "test" or "controller", it should be checked that the keys that are in the path .vagrant / machines / test / virtualbox / have the user as owner, and read-only permissions for the owner of the key. To do this, we simply change the permissions

cd .vagrant/machines/test/virtualbox/

chmod 400 private\_key

# We could also use this instead of "chmod 400" (u,g,o -> user, group,others)

# chmod u=r,go= private\_key

Instead of using the Vagrant administrator to make the SSH connection, we can choose to do it manually by passing the path to the private key using SSH

ssh -i .vagrant/machines/test/virtualbox/private\_key [vagrant@10.0.123.2](mailto:vagrant@10.0.123.2)

### Native File Installation

This method assumes that you already have all the virtual machines running, configured correctly and with the dependencies installed

Ideally you should have 2 virtual machines, where one of them will be the "controller", responsible for running Ryu and the other will be the emulated network of Mininet where we will have our network topology already created

To do this, we will make a clone of the repository:

git clone https://github.com/GAR-Project/project

cd project

For later, to be able to launch the scripts in each machine:

# To install Mininet and Mininet's dependencies. Run it on the "mininet" VM

sudo ./util/provisioning.sh

# To install Ryu. Run it on the "controller" VM

sudo ./util/ryu.sh

**GRAFANA CONFIGURATION**

Grafana es una herramienta que nos permite monitorizar, controlar y analizar cualquier recurso de nuestra red virtualizada. Está basada en código abierto por lo que podemos trabajar con ella. Además, es una herramienta que se puede instalar en cualquier plataforma, muy útil si queremos trabajar en Ubuntu o Windows como es nuestro caso.

Gracias a esta herramienta, podemos personalizar los paneles y gráficos para monitorizar el tráfico de nuestra red y comprobar si es tráfico normal o por el contrario poder tratarse de un ataque DDoS como veremos más adelante y poder generar alarmas para poder actuar en caso de tratarse de este último caso.

En nuestro caso, trabajaremos con Grafana junto con InfluxDB (que hablaremos más adelante) en la máquina virtual “Controller”

Para ello, lo que haremos es conectarnos mediante SSH a nuestra máquina “Controller”

vagrant ssh controller

//Instalación de Grafana con fotos. Comprobación de que el servicio está funcionando con systemctl status grafana-server

Una vez comprobado que el servicio funciona, solo tenemos que autenticarnos en el sistema con la IP de la máquina “Controller” mediante interfaz web

<https://10.0.123.3:3000>

//Foto de la interfaz

En cuanto entramos la primera vez a la interfaz, el propio sistema nos propone que modifiquemos la contraseña que tiene por defecto grafana, siendo User: admin. Password: admin.

Tras realizar este cambio y haber hecho login, ya podremos trabajar con grafana

**INFLUXDB CONFIGURATION**

Una vez que tenemos ya nuestro servidor montado y listo para monitorización que, en nuestro caso emplearemos la herramienta de Grafana, será necesario disponer de una base de datos (BBDD) en el sistema para poder empezar a recibir parámetros y alojar los valores de los servidores o elementos que son aptos para ser monitorizados

En cuanto al ámbito de Grafana se refiere, la base de datos más utilizada y conocida es InfluxDB, ya que al ser una herramienta de código abierto nos permite poder trabajar con esta sin problemas.

Ahora bien, para poder emplear la base de datos de InfluxDB, partimos de la base de que Grafana lo tenemos instalado en la máquina virtual “controller” del apartado anterior.

//Instalación de influxDB con fotos y comprobación de que el servicio está activo

Una vez instalada la base de datos, procedemos a crear una base de datos con la cual trabajaremos en nuestro proyecto

//Pasos para crear base de datos

En este punto, ya tenemos en nuestra máquina Grafana e InfluxDB junto con una base de datos creada. A continuación, procedemos a crear un “DataSource” de Grafana que apunte a esta nueva base de datos para poder monitorizar posteriormente los parámetros que queramos y alojarlos en esta base de datos recién creada.

Esto lo podemos conseguir gracias a la propia interfaz web que nos ofrece Grafana, desde la sección

Configuración > DataSources > Add data source

//foto

Y agregamos los datos que necesitamos para el correcto funcionamiento, como son el nombre, URL (<http://localhost:8086> ya que en nuestro caso, InfluxDB se encuentra en el mismo servidor que Grafana) y escribimos la base de datos que hemos creado anteriormente

//foto

Tras acabar de poner estos datos, simplemente hacemos click en “Save & Test” y comprobamos que se ha hecho correctamente mediante el mensaje “Data source is working”

//Foto

Para finalizar esta parte y comprobar que se ha añadido correctamente, vamos a

Configuración > DataSources

Y comprobamos que ya está disponible nuestro Data Source con el nombre de la base de datos que hemos creado. De esta forma, queda configurada correctamente la BBDD y lista para recibir los elementos que queremos monitorizar dentro de nuestra red

En el siguiente apartado, veremos cómo instalar y configurar Telegraf en servidores para que pueda enviar parámetros al sistema de monitorización así como la creación de dashboards en Grafana para parametrizar las vistas a monitorizar

**TELEGRAF CONFIGURATION**

Partimos ahora de la base de tener instalado y configurado correctamente tanto Grafana como la base de datos InfluxDB en la máquina “Controller”.

A continuación, pasaremos a configurar el agente de Telegraf. Para ello, lo que haremos es conectarnos mediante SSH a nuestra máquina “test”

vagrant ssh test

De esta forma, tendremos a nuestro servidor enviando información directamente a Grafana con base de datos InfluxDB

//Pasos para ainstalar telegraf y comprobación de servicio