Computer Networks

HY335 - Spring Semester 2019

Project Report

Dimitris Karathanasis - csd3547@csd.uoc.gr

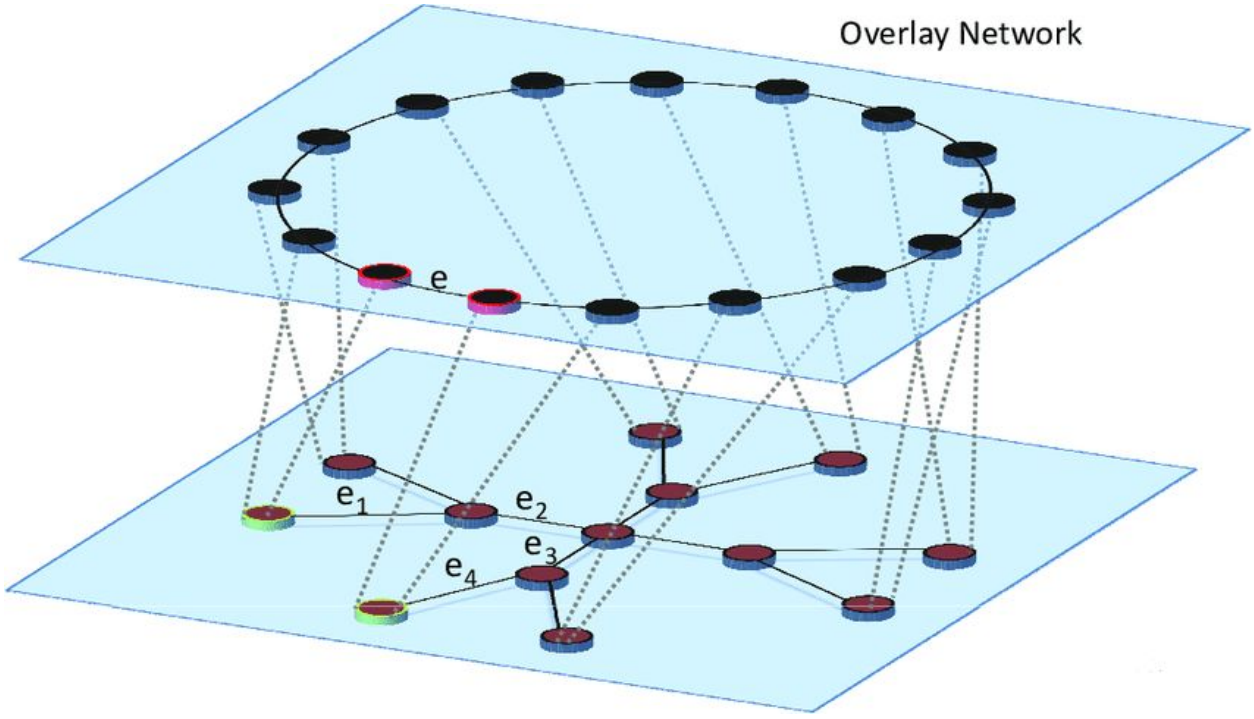
Emmanouil Sylligardos - csd3849@csd.uoc.gr

May 2019

1. Introduction

In this project we got familiar with Overlay Networks(ON). An Overlay Network is a network that is build on top of another network. In this work, ON provides two major advantages:

1) Security, by hidding client's IP address from server using relay nodes
2) Speed, by choosing the best path based on criteria such as latency



2. Workflow Analysis

In this section we are going to present the workflow of our model, configuration details and explain the different modes that are integrated into the application.
Our model's main components are two:

1) **Client**
2) **Relay**

Before jumping into further details let's see some configuration properties.

To begin with, our application is compatible with both Windows and Linux OS. That means that client's script could be executed in both environments. Git was used during the whole development process.

The main concept is that you run relay's script on all Debian machines that university provides us and then you run client's script locally. Some libraries are required to be installed in the computer running client's script(Crypto, panda, numpy, matplotlib, sklearn). After that, you ask relays to measure latency and number of hops that are needed in order to reach the end host provided by the user. At the end, client chooses the best path to download an image from, based on criteria that user defines.

Secure communication is provided during all steps of this procedure using Encryption techniques. At the beginning, client and relay exchange their public keys in order to set up a secure communication channel. Then, client generates a symmetric key for each relay that both sides use for easier and more flexible communication.

In the next few paragraphs we are going to demonstrate the various modes that are application integrates. More precise, those modes are 4 and each one contributes in a different way to project's functionality.

When you execute client.py with the appropriate files as arguments, the first thing that you are instructed to do is to choose between the four modes.

❖ **Default Mode**

This is the default mode that is described in project's handout. User is asked for alias of end-server, the number of pings that will be executed and the criteria on which to choose the appropriate route between client-server. At this point, client sets up secure encrypted communication with each relay available and sends them the information that they need to run measurements towards the end-server. Meanwhile, client pings and traceroutes each relay. These procedures all run in parallel using threading mechanisms of Python. When all measurements are finished, client aggregates the results and chooses the best path. After the completion of this step, user is instructed to enter which image he would like to download. Based on the best path, client either contacts the appropriate relay to download image for him and send it back (encrypted all the way) or downloads it directly, if direct has been selected as best path.

❖ **Offline Mode**

One thing worth mentioning is that measurements over the Internet could be expensive in both time and money cost. We shouldn't take for granted the fact that now we can perform measurements for free and in a small time window. If for example you would like to scale-out your measurements and/or add different types of measurements, this could lead to financial limitations or time consuming approaches(time is money!).

For this reason, we have implemented offline mode which chooses best path, based on logged history of already measured paths. This is achieved using Machine Learning techniques.

More specific, when you select offline mode, you are then asked to choose which image you would like to download. Using some information that are known to the user without the need of executing measurements in advance, such as file size, previous best path choices and end hosts,time of the day, our model decides which is the best path to download the file from.

As it refers to the ML model selection, we trained several models using ML algorithms and ended up with the best model for our situation.

First, we started with a Logistic Regression Model and achieved an overall performance of 72%.. This result is misleading because LR only predicts the most frequent path, in our case the direct path.. The same goes for Random Forest which is the next model we trained giving us a total performance of 73%. Due to it's hierarchical structure combined with the small number of features, the algorithm could not split appropriately the small number of features to the decision trees, leading to bad generalisation.

By training an SVM model, we are achieving a performance around **76%**. This happens because SVM uses internally a technique called the kernel trick to transform data and then based on these transformations it finds an optimal boundary between the possible outputs. This allows the model to extract the most out of the logged information of the features available and thus, it performs well in our situation.

The model selection requires further investigation and testing and it could be added to the future work.

- **Train Mode**

As we were testing our ML algorithms, we observed that the more real data we had in the log file the better the performance of our model was. On the other hand, it is time consuming to run each of the 11 relays manually every time you want to perform measurements.

To deal with these limitations, we automated the procedure of enriching our current dataset with new measurements and in that way, adapt to possible changes and keeping a continuous learning approach in our application. User is asked to specify the number of times that he wants to execute the whole measuring procedure. We randomly pick criteria and the end server on each iteration and perform all the required measurements, without downloading any image at all.

❖ **Analysis Mode**

After running multiple times our training algorithm, we thought that it would be crucial to provide to the user a complete Analysis overview based on the log file. So, by running the application on Analysis Mode we are informing the user about several things such as how many times each path was selected as best path based on each criteria separately. Furthermore, we generate some plots representing some of the analysis results.

3. Contributions

In the final section we would like to provide you with the overall contribution this projects is designed to deliver to the Computer Networks community.

We know that measurements play an important role in the Internet's evolution and optimisation. Sometimes though, due to limitations, we don't have the privilege of deploying and measuring all aspects of a particular problem in a way that we would like to.

For this reason, we believe that our offline approach based on previous logged measurements is crucial and could play a significant role in many real-word projects that face limitations, either from financial or limited time aspect. This requires further designing and development of our ML approach and could be considered as future work.