POLITECNICO
MILANO 1863

WI-FI ENCRYPTED TRAFFIC CLASSIFICATION

Wireless Internet Project

7TH SEPTEMBER 2020
TERESA COSTA CAÑONES AND ELEONORA ZUPI
Politecnico di Milano

Wi-Fi encrypted traffic classification

Introduction

In the beginnings of Wireless Communications there was no need of encrypting such communications because only a lucky few people were able to use it. The years have passed, and technology has evolved, and wireless communication has not been an isolated case. Nowadays there is the need to encrypt to provide a more secure browsing and a more secure experience with the use of wireless communications in all its aspects. The goal of encryption protocols is to provide security guarantees for data confidentiality and integrity, but unfortunately there are many attackers as well that use encryption and cryptographic techniques in their attacks. For that reason, it is necessary to know which type of traffic is passing through the network, being then able to be aware of the existence of malicious applications. The problem is that it is difficult to know which flows of data are traversing the network when they are encrypted, due to that it is needed to use other techniques.

The aim of the current project consists of classifying the traffic flows of Wi-Fi encrypted traffic between different types of applications. Specifically, classify traffic coming from a smart phone and distinguish traffic between *Youtube*, *Chrome* and *Whatsapp*.

The project has been divided into two main parts. One part is done in an Offline state and the other part is done in an Online state. In the first part it is extracted or computed different features of each type of traffic flow. More accurately it is computed: the length of the packets, and the data rate. The second part consists of classifying the traffic flows while they are captured. The classification is done using a *python* program and *Whireshark*.

Technical Background

Encryption protocols provide privacy and security to network communication. Without encryption an eavesdropper (an attacker, or even a network administrator) would be able to read network packets and view their contents. Traffic or packet encryption methods, such as TLS, IPSec etc., ensure that while an eavesdropper can still record packets, he can no longer decipher their content or modify them without detection. The most used network traffic encryption protocol is the Transport Layer Security (TLS). This protocol has three main features:

- Confidentiality: a connection is prevented from anyone reading its contents.
- Authentication: verifies the identity of the communicating parties.
- Message Integrity: provides a way to verify that a message has not been modified on the way from the sender to the receiver.

With modern encrypted traffic analysis (ETA) methods, an eavesdropper can read the following information from encrypted network traffic (that should be private thanks to TLS):

- They can get information about the presence or absence of applications installed on a user's smartphone.
- Find out which type of websites a user is surfing to.
- Find out which files a user downloads and shares over an encrypted channel.
- Identify user actions in mobile applications and build a user profile.

This means that our internet traffic (through computers or smartphones) spills more information than we assume possible. The main challenge today is to find a balance between end-to-end security and at the same time assembly information from the traffic to detect possible threats and better allocate and protect resources. It is a difficult problem and it is important to identify alternative ways of detecting malicious behaviour that do not decrease the confidentiality of users.

In order to identify traffic flows and being able to fulfil the current project, it is needed to know the different features of each application taken in consideration. *YouTube*, *Whatsapp* and *Google Chrome* can be distinguished using feature-based machine learning with reasonable accuracy, while observing certain properties of the encrypted data. It is possible to create data records which map these properties to the corresponding application. Those properties can be data rate, length of the packet, packet payload, frame interarrival time, etc... Using the SVM, which stands for Support Vector Machines, the classification of the different applications can be made. SVMs are very useful methods in machine learning, they can work in high dimensional spaces, with a number of dimensions greater than number of samples, are memory efficient and are also versatile because of the different Kernel functions provided.

Project development

The project has been developed with a computer having a Linux operating system. The previous statement is worth mentioning because having a computer with another operating system, as for example Windows, would mean that the computer prerequisites and the execution of the project would be different.

Computer prerequisites

For the purpose of being able to execute the project it is necessary to have a computer with a wireless network card which is able to have monitor mode. The previous mode of the wireless network card is able to sniff the traffic. It is also needed to take into account that it is needed to know which is the interface name where it is going to be sniffed.

Changing the network card into a monitor mode is done by following the steps:

- Turn off th wireless network interface
- Open a terminal and write the commands:
    - *sudo ifconfig wlo1 down*
    - *sudo ifconfig mode monitor*
    - *sudo ifconfig wlo1 up*
    - *sudo iw dev wlo1 set freq 2462*

“wlo1” is the wireless interface of the computer used in the current project, in order to know the wireless interfaces of a specific computer type on the terminal: *sudo iw dev*. In the current case the frequency is 2462 because it is used a wireless network using 2.4Ghz in the channel 11, but each network uses their own frequency. To discover which is the channel used it is necessary to enter inside the configuration site of the router used for testing the project.

Moreover, python is also needed with the *pyshark* library correctly installed. The library *numpy* and *sklearn* are also added. *Numpy* library was implemented for doing the statistics for the three application results, in uplink and downlink. *Sklearn* is added in order to do the classification of the traffic by using machine learning. The SVM *sklearn* supports the use of the *numpy arrays* in order to classify the data sets.

Project execution

As it is said before, in the project there are two files: one for the online mode and another for the offline mode. The difference between the two files is minimal. In the offline file it is possible to test the accuracy of the program by testing captures that the traffic is already classified. In order to execute the project, it is only needed to execute the file online.py or offline.py by executing the command *python3 online.py.* For the proper execution it is necessary to have also the files for the training (in the current case: *Youtube_2p.pcapng*, *Whatsup_p.pcapng* and *Chrome_1p.pcapng*). It is recommended to use a user interface program as for example *Pycharm* because it is easier to run and work with the project. In case the user wants to execute the file and save in another file the output of it, it is necessary to write in the terminal: *python3 online.py > output_online.txt.*

Code development

In the previous sections has been said that the project has two different parts: the offline part and the online part. In the current section will be explained how the code and the program is organized.

At the beginning, it is necessary to have different captures with only one traffic in it. In order to do that, a firewall was installed in the smartphone. The firewall blocks the traffic flow of the applications desired. The recordings have been done by blocking all the applications except from the wanted one. Each capture has the name of the application run during the capture. Once the data of the three applications *Youtube*, *Whatsapp* and *Chrome* browsing has been captured, it is necessary to extract the information of them in order to train the SVM. In the current case, the features are the packet length and the data rate. The data is extracted by averaging the features every 30 packets. It is needed to take into account that the number of samples that we have from each type of traffic is different. Due to that, it is necessary to weigh the sample in order to have well trained the SVM and to have a good result.

All the aforementioned is part of the two files: offline and online. In the second part is when it is decided to test the SVM with known traffic or with unknown traffic, this is the difference between online and offline files.

In the testing part, the features are also averaged every 10 packets and then it is predicted which type of traffic it is.

Results

Offline part

WhatsUpp

Whatsupp is an application which aims to send messages. The user sends and receives usually small messages, and sometimes it receives voice messages or videos. The test has been done by only using small messages. Due to that it is expected to have small length on the packets and data rate small, the user does not need a high speed because can receive the message with delay, so small data rate in uplink and in downlink.

The results show that in the file used for testing, there are no errors.

In case of taking into account voice messages, video messages or other characteristics of the applications for example voice calls or videocalls, the analysis and the results should be totally different (heterogeneous traffic).

Youtube

Youtube is an application of video streaming, and due to that it needs a high speed of downlink traffic and large packet lengths. Talking about uplink traffic it could be similar to Chrome. The previous reasoning is because the user is able to search other videos while is also playing the video.

The results show that there are some errors. The errors occurred when the packet (uplink or downlink) is very small. That the system predicts that is from the *Whatsupp* traffic. Moreover, when the downlink traffic has the packets of medium lengths, it predicts that this is chrome browsing. The previous happens when it is searching the information of the other videos (heterogeneous traffic).

Chrome

Chrome is an application that needs a little more data rate for the uplink than for the downlink. The previous statement is because it is needed that the user has without delay what is reading (for example of a google search).

It is worth to mention that the test has been done by only searching for information in the google and the seeing what is found. In case of entering in a website, depending on the type of website the features of the traffic changes and due to that it is more difficult to classify the traffic flows. Chrome is also an application which has a heterogeneous traffic that makes difficult to classify the traffic flows.

The result show that there are some errors. When the traffic increases, it is to say when the search have images or others, the packet length is greater and the data rate higher which causes errors on the prediction.

Online part

The results of the online part are similar than the results of the offline part. The difference is that in this case it is not always known the error. Due to that is more difficult to predict the traffic. It is worth to mention that has less error in case that the averaging of packets is greater (more than 10), but in this case it is probable that the short traffic flows are not well recognised.

Moreover, as all the applications have a heterogeneous traffic it is difficult to classify the traffic flows accurately.

Conclusions

The students have been able to develop a system capable of classifying three types of encrypted traffic flows, but not with a very high accuracy. The previous has been achieved because of the differentiation by two features of the types of traffics, which in fact should not be enough in order to have a high accuracy. The reason of the previous statement is because the traffic of the applications is heterogeneous which makes the differentiation more difficult.

Further improvements

The current project could be improved by different manners:

- Using more features. One of them could be inter arrival packet time. The previous feature has not been implemented due to problems of being able to have the measure accurately. It has been decided to not be used in order to not have more errors.
- Taking into account the different sequence number to differentiate the application traffic flow.
- Classifying the traffic of other applications.
- Being able to classify the applications using a heterogeneous traffic flow. In the current project it has only been classified using a more or less homogenous traffic, which actually this is not real.

Bibliography

SVM

https://scikit-learn.org/stable/modules/svm.html

[Last time entered 07/09/2020]

ENISA-Encrypted traffic analysis

https://www.enisa.europa.eu/publications/encrypted-traffic-analysis

[Last time entered 07/09/2020]

Github repository

https://github.com/terecosta/Wi-fi-encrypted-traffic-classification