Digital Forensics - Project Report

Project 1 - Packet classification for mobile applications

Eugen Saraci - 1171697
Università degli Studi di Padova

eugen.saraci@studenti.unipd.it

January 13, 2019

In the last years the large expansion qualche altro termine non sarebbe male of SSL/TLS has made it harder for attackers to collect clear text information through packet sniffing or, more in general, through network traffic analysis. The main reason for this is that SSL/TLS provides encrypts the traffic between two endpoints, which means that even though packets can still be easily captured, no useful information can be inferred from the packet's content without having the encryption keys.[1]

The authors of [1] and [2] showed that by training a machine learning algorithm with encrypted traffic data, one could correctly classify the user actions performed on the most common Android applications such as Facebook, Gmail, or Twitter, which could easily lead, through a correlation attack, to the full deanonimization of fake, privacy preserving identities.

In this work I try to reproduce the results achieved in [1] and [2] by implementing the classification model as described in the papers.

1 Introduction and Notation

1.1 Actions and Flows

Follows some useful and necessary terminology to better understand how the whole framework works.

Action and Action Label

An action is simply the action performed by a user while using one of the aforementioned Android apps. Examples of actions are: clicking on a profile page,

[1]It is worth mentioning that the endpoints of the communication (i.e. source and destination IP addresses) are transmitted in clear text for routing purposes; by performing a DNS lookup of the addresses and attacker could easily infer what site a user is visiting.

tweeting a message, sending an email etc. Please note that “clicking on a profile page” is what I refer to as the *action label*, in many cases I use the words “action” or “action flow” to refer to the set of flows that represent that action. spiegare meglio sta roba

Flows

When a user performs an action some encrypted packets are exchanged with the destination server. A flow consists of the sequence of the byte sizes of the exchanged packets. If the packet is going from the user’s phone to the server it is said to be *outgoing*; if the packet is coming from the server to the user’s phone it is said to be *incoming* and it is marked with a “-” sign before the integer number representing its size.[2] An example of a 5 packet flow is: `[-12, 80, 90, -111, 30]`. Please note that a single action performed by the user usually generates multiple flows of different dimensions, by that follows that an action actually consists of multiple flows. The techniques used by the authors to determine which flows belong to which action, the ordering of the packets, the packet capturing system, the packet filtering system, and the statistical analysis on the flows will not be treated in this report since the starting point for this work comes when the dataset is already constructed.

1.2 Notation

- A: action, it represents a sequence of flows;
- a: action label;
- F: a flow, it represents a sequence of packets;
- p: a single packet, it is an integer number representing the size in bytes of that packet.

  Please note that all of the above can be subscripted by indexes; a subscripted element means that that element is the i-th element of a sequence, e.g. F_i is the i-th flow of a sequence of flows (possibly an action A). notazione [1,2,3,4]_i da spiegare

  By this follows that $A = [F_1, \dots, F_n]$ and $F = [p_1, \dots, p_m]$. Note that n and m are possibly (and probably) different for each flow F and for each action A, even for two actions A_i, A_j where $a_i == a_j$.

[2]The “-” sign is just notation, packets cannot have a negative size.

2 Machine Learning

2.1 Dynamic Time Warping - INGLOBARE NEL CAPITOLO MACHINE LEARNING

Dynamic Time Warping or **DTW**, is an algorithm used to measure similarity between two time series even if they are of different lengths and/or have repetitions or deletions in them. If we view every single flow F as a time series of packets $p_i, \dots p_m$, we can use **DTW** to measure how similar two flows are. The reason we are interested in this will become clear later.

2.2 Machine Learning

Ideal goal: we want to develop a machine learning algorithm that outputs the *action label* a_i given the *action's flows* $A_i = [F_1, \dots, F_n]$.

Given the fact that each action generates multiple flows of different lengths, we know that standard supervised learning approaches are hard to apply. To see why standard approaches would not work we need to think about the structure of the input and output spaces. Our output space i.e. what we want to predict would be the *action label* a_i, while our input space, i.e. the predictors, would be the flows generated by a_i which we denote as $[F_1, \dots, F_n]_i$. One way we could represent flows as features would be to have a feature for each flow F_j generated by the action a_i, and the value of a feature would be the sequence of byte sizes of the flow $[p_1, \dots, p_m]_j$. Because of the different number of flows of each action (n) we would immediately see that each row could possibly have a different number of features. The main problem of this approach is that we are artificially defining features with no real justification; in other words, we have no reason to associate the first flow of an action with the first flow of another action by marking them as the first feature of the respective samples.

To overcome this obstacle the original authors applied a two stage process. In the first stage, by using an unsupervised clustering algorithm they addressed their missing knowledge about the number of flows and their features. The aim of the first stage is to identify some features for our actions. The second stage exploits the just found features to perform a the canonic classification.

2.2.1 Hierachical Agglomerative Clustering

In the typical clustering scenario the goal of the algorithm is to find k groups called clusters, where the *inter-cluster* similarity is very high while the *intra-cluster* similarity is minimal, meaning that the samples belonging to one clusters are very similar to each other while still being very different from samples of other clusters. Notice that the number of clusters k and the similarity function between samples $f\_dist()$ have to be explicitly defined by the teacher. The output of a clustering algorith finire spiegazione

In my specific implementation $k = 50$ and $f_d ist = dtw$. In my specific implementation I use as samples all the flows by themselves, ignoring temporarily the idea of action. The reason for this is that w

2.2.2 Random Forest

3 Evaluation

3.1 Experimental Setup

3.2 Experimental Results

References

[1] Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2015. Can't You Hear Me Knocking: Identification of User Actions on Android Apps via Traffic Analysis. In Proceedings of the 5th ACM Conference on Data and Application Security and Privacy (CODASPY '15). ACM, New York, NY, USA, 297-304. DOI: https://doi.org/10.1145/2699026.2699119

[2] Conti, M., Mancini, L. V., Spolaor, R., & Verde, N. V. (2016). Analyzing android encrypted network traffic to identify user actions. IEEE Transactions on Information Forensics and Security, 11(1), 114-125.