Traffic Analysis Attack on HTTPS traffic

Abstract

Traffic Analysis is a technique in which the intruder intercepts and examines the messages in the traffic, in order to deduce information from patterns in communication, i.e., the meta-data, rather than the content of the messages. Meta data can be used to reveal the time, duration, frequency of communication, identities of the communicating parties, link between senders and receivers, and their locations. Website fingerprinting enables the attacker to infer which page a client is browsing through an encrypted or anonymized network connections [1]. Hence, I perform traffic analysis attacks using website fingerprinting.
Using the tcp dump of the https traffic over a set of URLs, I extracted the fingerprints from the raw packet dump (tcp dump) of each URL. The dataset was built by repeatedly collecting the packet traces of the URLs over the period of 10 days. Given the unknown packet traces, I matched them with the given set of URLs. The Traffic analysis attack in this case is a classification problem and was performed using the k-Nearest Neighbour classifier with input as the features extracted from the given unknown packet traces and packet dump of the given set of URLs. Hence, as an attacker, in a closed world setting such as this, I can know which urls the user visited.

1 Introduction

Performing Traffic analysis attack makes use of the website fingerprinting. The meta data that is collected from the tcp dump over the HTTPS connection, gives information about the incoming and outgoing packets. Using this information the attacker can know the identities and the location of the communicating parties using the IP addresses, time and duration, and the link between the senders and receivers.

For the attack, the attacker has listened to the traffic of a user and collected it - this makes the set of unknown packet traces. And the attacker has an auxillary knowledge of the set of URLs that the user may visit - this forms the set of known URLs. Here, I have considered a closed world setting, where I have a set of URLs and a set of packet traces. I need to match the URLs with the given unknown packet traces. This project illustrates that the attacker can know which URLs the user visited in a closed world setting.

2 Literature

Website fingerprinting enables an attacker to infer which web page a client is browsing through encrypted or anonymized network connections.

Website fingerprinting is commonly formulated as a classification problem. An attacker wishes to know whether a client browses one of n web pages. The attacker first collects many examples of traffic traces from each of the n web pages by performing web-requests through the protection mechanism under attack; features are extracted and a machine learning algorithm is trained to classify the website using those features. When a client browses a web page, the attacker passively collects the traffic, passes it in to their classifier and checks if the client visited one of the n web pages. In the literature this is referred to as the closed-world setting a client is restricted to browse a limited number of web pages, monitored by the attacker.

According to the k-fingerprinting method proposed by Hayes and Danezis, there are a list of features that are important in determining the clustering of the fingerprints. They defined a distance metric between two traces based on the output of the forest: given a feature vector each tree in the forest associates a leaf identifier with it, forming a vector of leaf identifiers for the item, which they refer to as the *fingerprint*.

Feature Importance

According to work by Hayes and Danezis, from each packet sequence, the following features were extracted:

- **Number of packets statistics**. The total number of packets, along with the number of incoming and outgoing packets [12, 27, 39]. The number of incoming packets during transmission is the most important feature, and together with the number of outgoing packets during transmission are always two of the five most important features. The total number of packets in transmission has rank 10.

- **Incoming and outgoing packets as fraction of total packets**. The number of incoming and outgoing packets as a fraction of the total number of packets. Always two of the five most important features.

- **Packet ordering statistics**. For each successive incoming and outgoing packet, the total number of packets seen before it in the sequence [7, 27, 39]. The standard deviation of the outgoing packet ordering list has rank 4, the average of the outgoing packet ordering list has rank 7. The standard deviation of the incoming packet ordering list has rank 12 and the average of the incoming packet ordering list has rank 13.

- **Concentration of outgoing packets**. The packet sequence split into non-overlapping chunks of 20 packets. Count the number of outgoing packets in each of the chunks. Along with the entire chunk sequence, they extract the standard deviation (rank

16), mean (rank 11), median (rank 64) and max (rank 65) of the sequence of chunks. This provides a snapshot of where outgoing packets are concentrated. The features that make up the concentration list are between the 15th and 30th most important features, but also make up the bulk of the 75 least important features.

- **Concentration of incoming and outgoing packets in first and last 30 packets**. The number of incoming and outgoing packets in the first and last 30 packets. The number of incoming and outgoing packets in the first thirty packets has rank 19 and 20, respectively. The number of incoming and outgoing packets in the last thirty packets has rank 50 and 55, respectively.

- **Number of packets per second**. The number of packets per second, along with the mean (rank 44), standard deviation (rank 38), min (rank 117), max (42), median (rank 50).

- **Alternative concentration features**. This subset of features is based on the concentration of outgoing packets feature list. The outgoing packets feature list split into 20 evenly sized subsets and sum each subset. This creates a new list of features. Similarly to the concentration feature list, the alternative concentration feature list are regularly in the top 20 most important features and bottom 50 features. Note though concentration features are never seen in the top 15 most important features whereas alternative concentration features are, at rank 14 and 15, so information is gained by summing the concentration subsets.

- **Packet inter-arrival time statistics**. For the total, incoming and outgoing packet streams extract the lists of inter-arrival times between packets. For each list extract the max, mean, standard deviation, and third quartile. These features have rank between 40 and 70.

- **Transmission time statistics**. For the total, incoming and outgoing packet sequences they extract the first, second, third quartile and total transmission time. These features have rank between 30 and 50. The total transmission time for incoming and outgoing packet streams are the most important out of this subset of features.

- **Alternative number of packets per second features**. For the number of packets per second feature list they created 20 even sized subsets and sum each subset. The sum of all subsets is the 9 th most important feature. The features produced by each subset are in the bottom 50 features - with rank 101 and below. The important features in this subset are the first few features with rank between 66 and 78, that are calculated from the first few seconds of a packet sequence.

They conclude that the total number of incoming packets is the most informative feature. This is expected as different web pages have different resource sizes, that are poorly

hidden by encryption or anonymization. The number of incoming and outgoing packets as a fraction of the total number of packets are also informative for the same reason.

3 Building the Dataset

For performing the Traffic Analysis attack, we have a set of URLs. I need to construct fingerprints of each URL. The packet dump of the URLs over a period of 10 days was collected, using the 'tcpdump' command in linux. Using the packet dump, I extracted the fingerprints, similar to the unknown packet traces. I listened for 10 seconds to the packet exchange between the source and the destination, using the port as *https* and destination as the domain of the URL.

A sample **packet dump** for one URL looks like :
09:22:40.598070 IP 151.101.40.193.https > toshal-Aspire-E5-571G.lan.60724: Flags [.], seq 271095:272543, ack 1013, win 65, options [nop,nop,TS val 1219125024 ecr 585738118], length 1448
09:22:40.598093 IP 151.101.40.193.https > toshal-Aspire-E5-571G.lan.60724: Flags [P.], seq 272543:272718, ack 1013, win 65, options [nop,nop,TS val 1219125024 ecr 585738118], length 175
09:22:40.598105 IP toshal-Aspire-E5-571G.lan.60724 > 151.101.40.193.https: Flags [.], ack 272718, win 2355, options [nop,nop,TS val 585738288 ecr 1219125024], length 0

The incoming packets are the packets from the destination IP address to source IP address, and vice versa for the out-going packets. I extracted the fingerprints in the format (*time-instance*, *packet-length*, *label*), where *time-instance* begins from 00.00.00.000000, *packet-length* is the length of the packet representing the number of packets, and *label* is the label of the packet whether it is incoming or outgoing. For example -
00:00:00.000070 1448 in
00:00:00.000093 175 in
00:00:00.000105 0 out

4 Extracting Features

The matching of the unknown packet traces with the URLs is a classification problem. Various features are extracted to classify the fingerprints into different classes (labels as URLs number).
Using the list of important features (rank-wise) for website fingerprinting used in the k-fingerprinting method, proposed by Jamie Hayes and George Danezis, I extracted the following features for the attack -

- Number of incoming packets.

- Number of outgoing packets as a fraction of the total number of packets.
- Number of incoming packets as a fraction of the total number of packets.
- Standard deviation of the outgoing packet ordering list.
- Number of outgoing packets.
- Average of the outgoing packet ordering list.
- Sum of incoming, outgoing, and total number of packet.
- Standard deviation of the incoming packet ordering list.
- Average of incoming packet ordering list.

Packet ordering statistics: For each successive incoming and outgoing packet, the total number of packets seen before it in the sequence [7, 27, 39]. According to the most important features' list, packet ordering features have rank 4, 7, 12 and 13, indicating these features are a good criterion for classification. [1]

5 Packet Matching

I have the set of unknown packet traces. The unknown packet traces (fingerprints) are classified into (matched with) the URLs using the extracted features and k-Nearest Neighbour classifier. K-nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). The distance formula used for calculating the distance between the two points, is the *minkowski distance* with p=2, which is equivalent to euclidian distance.

$$dist\_Minkowski = \left(\Sigma_{i=1}^{m} |\ x_i\ -\ y_i\ |^p \right)^{1/p}$$

The weight points by the inverse of their distance - closer neighbours of the query point will have a greater influence than the neighbours which are farther away.

6 Results

The attack was run using the packet trace as the test dataset, and the dataset collected as the training data. The features extracted from both the sets were used as inputs to the k-NN classifier. The classifier got upto 65.71% accuracy.

7 Conclusion

This this project I performed Traffic analysis attack in a closed world system using website fingerprinting and kNN classifier and matched most of the urls with an unknown set of packet traces. There are various other classifiers that can be implemented to perform the Traffic analysis attacks, such as the random forests classifier, which can help in better feature extraction and classification for website fingerprinting.

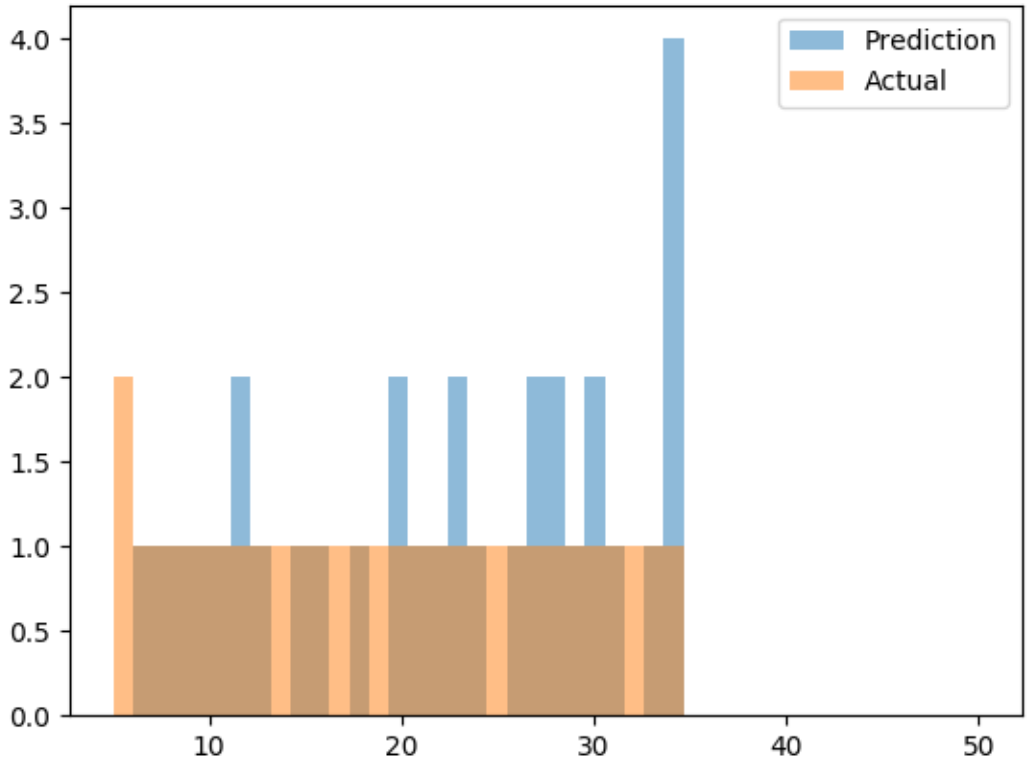

Figure 1: Comparision of actual values and predicted values

References

[1] Jamie Hayes, George Danezis. *k-fingerprinting: a Robust Scalable Website Fingerprinting Technique*. USENIX Security Symposium, 1187-1203, 2006.

[2] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. *Effective Attacks and Provable Defenses for Website Fingerprinting*. Proceedings of the 23rd USENIX Security Symposium, pp. 143157, 2014.