EAVESDROPPING ON WEBSITES

CYBER SECURITY PROJECT

Ishan Desai – ijd140030
Karthik Vesireddy – kxv141230
Praful Bhosale – pbb140230
Praveen Kumar Anagi Prabhakar – pxa140230
Raja Shekar Subbaraj – rxs148130

***Abstract*—Eavesdropping is a problem for online secure communications today. Although encryption does hide the contents of the communications and makes it secure, it does has its own pitfall. However user online interactions can be concealed if passed through security protocols or anonymity networks like Tor. Nowadays Tor has been widely used to conceal website address, website page contents, user actions and user anonymity. Nonetheless website traffic analysis and fingerprinting techniques are used to break privacy of users by revealing user actions and anonymity. So in this paper we test the feasibility of Website fingerprinting attack by actually implementing the attack. We illustrate how the fingerprinting techniques can be used as cybercrime investigation tools and discuss the challenges of such applications.**

Keywords—cybercrime, investigation models, Eavesdropping, Forensics, Website fingerprinting.

I. INTRODUCTION

Privacy leakage is a hot topic on the web right now. As web applications are increasingly used to exchange sensitive information such as social security numbers, credit card numbers, health reports etc., so are the concerns about the safety of these data. The encryption protocols do their best to conceal user privacy by encrypting data sent from or received by them. This is achieved by encrypting packets into hard-to-reveal ones. Some of the information about traffic behavior cannot be encrypted even if the contents of the file are encrypted. This information can be used for k-anonymity attacks on the user, whereby n sets of identity are mapped back to user, revealing a partial picture of the user. When the data is sent over the HTTP protocol instead of HTTPS, data is sent in clear text, which can be easily snooped over the wireless range network. The types of attacks that can be carried on the network are:

- Direct Anonymity attacks
- Indirect or k-anonymity attacks
- Eavesdropping, spoofing and wormhole attacks.

The broadcast nature of the wireless networks makes it easier for a hacker to spoof and inject data into user's network. This data can be used to retrieve the user's personal info without their consent. We also need to take into consideration the phishing attacks which is when an attacker tries to get user information posing as a legitimate source. The simplest form will be cloning a website and making the URL available to the user.

Man in the Middle is another spoofing technique. Not even the encrypted traffic will protect the user from MitM attack as the hacker can strip the packets from the certificates that made the connection secure. Attackers were also till recently able to force downgrade the secure connections to less secure one. The types of architecture predominantly used in online communication networks are Client-server and peer to peer. The attacks are particularly not serious in client server architecture when protected by HTTPS. Historically HTTPS was used in credit card payment processing industry, but later on extended to be used in web browsers and currently supported by all major web browsers such as Google Chrome and Mozilla Firefox.

Apart from these security threats we have other ones such as:

- SQL Injection
- Session Hijacking
- Keylogging
- Malware/Trojans
- Hack website server's directly.

II. FEATURE DEFINATION

Pre-processing of the probed traffic is performed following the approach in [5]. Pre-processing removes the non-TCP and pure ACK packets that might affect the results.

A. Feature Analysis

In this section, we demonstrate the dynamism of the dataset we collected by presenting sample probed traffic trace for web transactions corresponding to different destination sites. Fig. 1 shows data rate over time during the progression of a transaction. Traffic in both directions is considered in the graphs.

It can be observed that the data rate signatures for different destination sites vary significantly in terms their burstiness, number of bursts, and the aggregated average rate. These variations in the traffic patterns translate into distinguishable units features across various target websites, thus yielding good classification performance as reported later in this section. To

depict dynamism of the pages to the same website, in what follows, we present intra-site feature variation by showing samples of burst sizes and surge period features from various target sites.

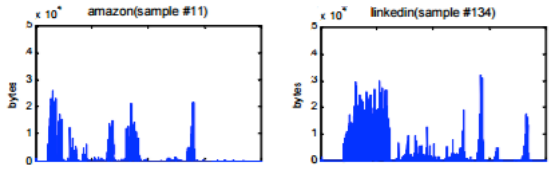

Fig. 1: Traffic rate-traces for web transaction to different sites

B. *Burst Size:*

A burst is defined as the interval during which there are only packets in one direction, but preceded and followed immediately by packets in the opposite direction. There are upstream and downstream bursts. Upstream bursts are shown on the negative half of the graph while downstream packets on the positive half. Each time the stream switches direction from the negative half to positive half or vise-versa, a burst boundary is recorded. Each burst is marked by 2 boundaries and the direction of packets inside that burst becomes the direction of the burst itself. Burst is also used in [5], albeit named differently. It is used as part of the feature set in two different ways. 1) Number of bytes transmitted in each burst is summed up into a histogram and contained in the feature. 2) Number of packets transmitted in each burst is summed up into a histogram and contained in the feature.

C. *Surge Period*

We propose a new feature, termed as Surge Period, which unlike the packet direction based bursts, is defined based on the timing of surges in traffic density, or high bandwidth utilization. A Surge Period marks the parts of traffic trace where the channel is busy transmitting packets upstream or downstream with back to back packets. Any packet except for the first one and the last one within the surge period should be separated from its predecessor and subsequent packets by a time period no larger than a pre-defined time window size. The window size depends on the network condition and typically ranges from 10 milliseconds to several hundred milliseconds.

During web transaction, the timing of a Surge Period corresponds to one object, or a group of objects being downloaded together. We have speculated that, on a coarse scale, the timing of a Surge Period corresponds to the location of those references that resulted in it. This is based on the fact that modern browsers want to render the contents with the highest speed possible, and that they should send requests for resources quickly after the references are discovered by the parser.

As demonstrated in Fig. 4, while the probed data can be decomposed into individual TCP connections, the Surge Period feature is extracted from the combined traffic pattern (the far right column in Fig. 4). This is done so that the feature should be prepared for situations, such as Tor, in which the traffic cannot always be separated into individual connections.

As done in [1] [5], we assume that all traffic from or to a browser client are from the same transaction. Meaning, we ignore interleaving of multiple web transactions.

The Surge Period feature is extracted as follows. First, a probed trace is converted into a time-stamped series of (time, packet-size) pairs. Second, the time-stamped series is converted into a bit rate time series computed over 100ms non-overlapping windows.

Third, the most significant Surge Periods are then extracted from the bitrate time series. To do this, we apply the following adaptive method. A continuous period of bitrate higher than a certain threshold I_{th} is counted as a Surge Period. Multiple thresholds are experiment with, starting from the highest value possible, and gradually descending towards until a threshold where more than 80% of all the bytes transferred are covered by in burst periods.

Finally, the number of bytes transmitted in a Surge Period is summed up as the 'size' of that period, and all Surge Periods are lined up according to their timing order. Top N periods in size are then chosen, where N is an arbitrary number. The resulting vector of period sizes are used as the feature that represents the sample. If there is not enough number of Surge Periods even after $P_{th} = 0$ is tried, the vector is padded with zeroes at the end.

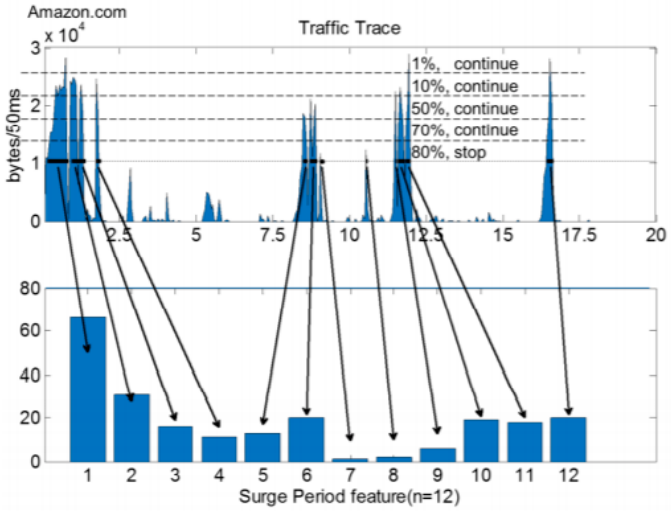

Fig. 2: Example of Surge Period for amazon.com traffic

III. PROPOSED METHOD

Fingerprinting Procedure: To mount a fingerprinting process, a set of sub-processes has to be accomplished. At first, Internet traffic should be monitored synchronously using a sniffing tool (Like Wireshark). After collecting a set of traces (or website visits), packet's features are chosen to represent the fingerprint for the given Websites or Applications.

These features are then classified through a particular Machine Learning classifier. Indeed, classifier is an effective part of any fingerprinting technique, and it differs from one technique to others (DT, NB, etc.). After performing a set of

training processes, another set of test cases is implemented to examine whether the classifier is learned to identify websites identity.

In our project, we are reading data through KafkaConsumer. Next step is to split data into 3:2(i.e. training : test) ratio. Then feeding those traing data points to generate model using three different classification techniques described below. Finally, read test data from kafka stream of messages and run test data through each model to classify test data points. User can select any trained model for classification.

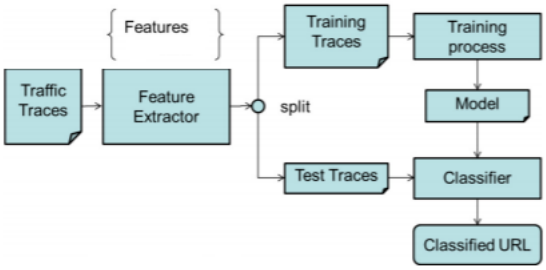

Fig. 3: Website fingerprinting classification workflow

In above block diagram Training Process block involves multiple steps which are based on selection of a classifier. We are using three classifiers that way we can compare performance of different classification methods on same test data.

A. Naïve Bayes

In Naive Bayes classification method which is based on Bayes theorem, it is assumed that all features are strong (naive) independent from each other. Because naive Bayes classification can be implemented easily without any complicated iterative parameters, it is useful for very large datasets. Although this method is simple, but it usually outperforms other sophisticated methods. Bayes theorem provides a way of calculating the posterior probability, P(c|x) (Posterior probability), from P(c), P(x), and P(x|c) (Likelihood). Naive Bayes classifier assumes that the effect of the value of a feature (x) on a given class (c) is totally independent of the values of other features. This assumption is called class conditional independence.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (2)$$

$$P(c|x) = P(c)\prod_i P(x_i|c) \quad (3)$$

Considering the naive Bayes equation, the likelihood probability $(\prod_i P(x_i|c))$ could be used as a score of class C. This score can be used as a threshold to separate attack from normal traffic.

B. Random Forest

We assume that the user knows about the construction of single classification trees. Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

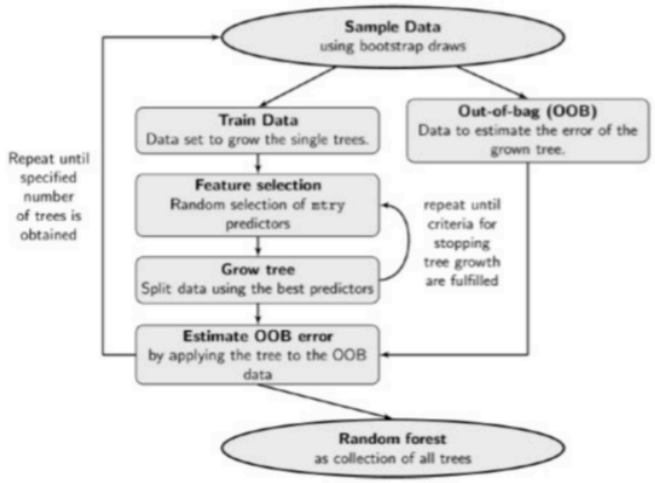

Fig. 4: Random Forest Classification Model

The approch takes a random sample of data and identifies key set of features such as burst size & surge period to grow each decision tree. These decision trees have their Out-Of-Bag error determined (error rate of the model) and then the collection of decision trees are compared to find the joint set of variables that produce the strongest classification model.

C. Decision Trees

A Decision Tree creates a type of flowchart which consists of nodes and set of decisions to be made based off of node (referred to as “branches”). Decision Tree learning is one of the most widely used and practical methods for inductive inference and is an important tool in machine learning and predictive analytics.

Finally, we present performance comparisons of classifying websites using different classifiers.

D. Spark Streaming

An extension of the core Spark API that enables processing of live data streams. Data can be ingested from many sources like Kafka, Flume, Twitter, ZeroMQ, Kinesis, or TCP sockets. We are specifically using Apache Kafka.

Data can be processed using high-level functional programming and can be applied Spark's machine learning algorithms on data streams. We are using MLlib provided with spark-streaming for Naïve Bayes classification task.

In our Project, Packet sniffer program puts live data of encrypted web traffic into Kafka's feed of messages, also known as 'topic'. Other program known as KafkaConsumer reads messages from this topic continuously, in the window of 100 seconds (which is configurable), and predicts the website domain of the traffic using Naïve Bayes, Decision Tree and Random Forest Classifier. At the end we can compare the accuracy of each classification method.

IV. Future Scope

We believe that further research on evaluating the common assumptions of the WF literature is important for assessing the practicality and the efficienc of the WF attacks. Considering the different attributes of the data packets such as average packet length, total trace time, upstream/downstream total bytes and implementing all three classification models, will result into a better accuracy for the classification.

Considering the attributes including website variance over time, multitab browsing behaviour, TBB version, Internet connection, and the open world. When each of these assumptions are violated, the accuracy of the system drops significantly, and we have not examined in depth how the accuracy is impacted when multiple assumptions are violated

V. Challenges

In this paper, we endeavor to present website fingerprinting techniques as being used as an investigation technique that can play a role against cybercrimes. In order to support detecting and preventing cybercrimes activities via traffic analysis and fingerprinting, several challenges should be taken into account in order to accomplish the process model effectively and efficiently.

In the literature, each fingerprinting techniques aimed to address some of these challenges, but affecting others. For instance, a technique may filter the gathered data in a better way in order to assist in reducing the space overhead and the processing overhead later. Although this is useful from that perspective, the filtering itself might consume more time than expected, which, consequently, delays the upcoming processes.

References

[1] D. Herrmann, R. Wendolsky and H. Federrath, "Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial Naive-bayes Classifier," in Proceedings of the 2009 ACM Workshop on Cloud Computing Security, New York, NY, USA, 2009. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] J.-F. Raymond, "Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems," in PROCEEDINGS OF INTERNATIONAL WORKSHOP ON DESIGN ISSUES IN ANONYMITY AND UNOBSERVABILITY, 2001. K. Elissa, "Title of paper if known," unpublished.

[3] A. Panchenko, L. Niessen, A. Zinnen and T. Engel, "Website Fingerprinting in Onion Routing Based Anonymization Networks," in Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society, New York, NY, USA, 2011. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[4] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7036866

[5] A. Panchenko, L. Niessen, A. Zinnen and T. Engel, "Website Fingerprinting in Onion Routing Based Anonymization Networks," in Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society, New York, NY, USA, 2011.