Identifying Webbrowsers in Encrypted Communications

Jiangmin Yu
Oklahoma State University
Computer Science Department
jiangmy@cs.okstate.edu

Eric Chan-Tin
Oklahoma State University
Computer Science Department
chantin@cs.okstate.edu

ABSTRACT

Webbrowser fingerprinting is a powerful tool to identify an Internet end-user. Previous research has shown that the information extracted from webbrowsers can uniquely identify an end-user. To collect webbrowser specific information, intentional JavaScript codes are embedded in web pages. In this paper, we show that fingerprinting characteristics of a webbrowser can also be collected by solely checking the network traffic data generated when browsing a website. We collect network traffic data generated by browsing the homepage of the most popular websites. Based on this data, we show that the browser fingerprinting characteristics can be inferred with high accuracy. Among these characteristics, type of webbrowser can be identified with over 70% accuracy rate. Usage status of popular plug-ins like JavaScript and flash can also be accurately identified.

Categories and Subject Descriptors

K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*; C.2.0 [**Computer-Communication Network**]: General—*Security and Protection*

Keywords

Privacy, Webbrowsers, Anonymity

1. INTRODUCTION

Webbrowser fingerprinting is widely used in nowadays Internet environment. Many Web services employ webbrowser fingerprinting techniques to track end users. The identification of an end user can import great benefits to web services providers. For example, an advertisement service provider can export new car ad to a user if the user once access auto-vender websites and his/her browser's fingerprinting is recorded by the website. Among these webbrowser fingerprinting techniques, browser plugins, cookies and Javascripts codes are used to collect fingerprinting characteristics. These tools can collect webbrowser specific information such as webbrowser's producer, minor version, plug-in usages and font usages etc. Combining these information, an end user can be accurately identified.

The goal of this research is to determine whether the webbrowser fingerprinting characteristics can be obtained based solely on network traffic. The adversary is not the server, but any entity between the client and the server. The adversary thus can only eavesdrop on the network traffic. Relying only on the network traffic, the goal is then to identify the webbrowser being used. For the websites used in our experiments, the identification accuracy rate of browser type ranges from 72% to 85%. Popular webbrowser plug-ins like JavaScript and flash usage status can also be accurately identified.

All work on webbrowser identification such as [12,17] have assumed that the identifier entity is located at the server's side. That entity has the ability to set cookies, look at user agent strings, and so on. Using this method, a lot of information can be collected about the webbrowser such as the type of browser, the major and minor version of the browser, plugins installed, the fonts installed, and the web history. As shown in [4, 12], users can be almost uniquely identified from their webbrowser information.

For network traffic identification, the closest work is website fingerprinting [10, 13, 18, 20]. Looking at anonymized and encrypted network traffic, an adversary can identify the website being visited. This is useful for censoring particular websites. Our work focuses on network traffic analysis and uses similar techniques to identify webbrowsers.

Threat Model: The threat model considered in this paper is that the client machine, the webbrowser used, and the server are all honest and free of any malware or bugs. The adversary can see the traffic between the client and the server. The adversary could be the client's ISP or any router on the path from the client to the server. The network traffic could be encrypted or in plain-text or through an anonymizing network such as Tor. The adversary can only eavesdrop on the network traffic; it cannot modify, drop, or inject any network packet.

2. RELATED WORK

It has been shown [8, 10, 13, 14, 18, 20] that analyzing network traffic can reveal the websites and webpages being visited. The authors looked mostly at packet sizes and packet direction. They collected a training dataset from specific websites and clustered the dataset. Given a network traffic trace, the authors used machine learning techniques to predict the website visited. Website fingerprinting can be used when users want to hide who they are communicating

WPES 2014
Copyright 2014 ACM 978-1-4503-3148-7/14/11 ...$15.00.
http://dx.doi.org/10.1145/2665943.2665968.

with, for example, by using Tor. These users might want to bypass any censorship mechanism blocking certain websites. Using website fingerprinting, the censor can identify users accessing censored web content. The latest paper [18] showed that specific websites can be accurately identified, only by examining the network trace. Our work differs in that although we are analyzing network traffic, the traffic analysis is used to identify and track webbrowsers, not to fingerprint websites visited.

Panopticlick [4,12] showed how browser information, such as user agent string, fonts installed, languages supported, and plugins enabled, can be used to uniquely identify users. Every user has an almost unique fingerprint and this can be used to track users visiting different websites or across different visits to the same website. The main goal of identifying users is to track them. This is especially useful for advertising. The EverCookie [3] is a persistent storage that is not cleared when the browsers cleans its cache and cookies. This permanent cookie can be used to track users. [17] showed how users can still be tracked without setting any cookies. FPDetective [7] did a wide survey of popular Internet websites and determined which website was performining browser fingerprinting. Our work is different because the fingerprinting is not at the server side; we assume the server is honest. Instead, the adversary can only eavesdrop on network traffic between the client and the server.

In an attempt to hide their Internet activities, remain anonymous, or bypass censors, anonymity-providing schemes such as Tor [9] and Anonymizer [2] have been deployed. These schemes only hide who is communicating with whom. Network traffic analysis can still be performed to identify any website visited. Server-side fingerprinting can still be performed to identify users and webbrowsers. There are tools [5] that attempt to remove identifying or sensitive information from web headers. However, they do not provide complete protection as shown in [17,18].

Network traffic analysis have been used for some time [11, 16]. They can be used to detect the application being used (such as Skype), or more commonly, to deanonymize users. Various countermeasures such as traffic morphing [19] have been proposed. A survey of network traffic analysis attacks and countermeasures is given in [15]. In our work, we leverage network traffic analysis to identify and track users from their webbrowser information.

3. EXPERIMENTAL SETUP

3.1 Collecting Data

Our assumption is that we know which web site the user is visiting, and we need to identify who is visiting the web site, that is, to get the fingerprinting information of the webbrowser which the user is using. The information we can use to reach this goal is only the network traffic generated when browsing a web page. In this paper, network traffic data are collected for four types of webbrowsers and homepages of 9 top Alexa [1] websites.

Our experiments are done on Windows 7 platform. The browsers we run experiments upon are Google Chrome 29.0, Internet Explorer 10.0, Firefox 24.0 and Tor Browser Bundle 3.6.1. The first three are widely used web browsers. Tor Browser Bundle is a pre-configured Firefox browser, it integrates Tor software to the webbrowser and takes advantage of Tor to protect users' anonymity. Wireshark 1.10.1 is used to record TCP/IP traffic data.

We use top 9 most popular web sites (downloaded from Alexa web site [1]), open them in web browsers and record TCP/IP traffic data. Some web sites support localization, they will automatically switch to different versions of web pages according to the region where we launch the browser. When using Tor Browser bundle, the region of our browser is determined by the exit node of the Tor circuit. In our experiments, we specify the region for google.com and yahoo.com as `https://www.google.com/ncr` and `https://us.yahoo.com/?p=us` respectively. In this way, we can ensure that each time we are visiting the same website. 9 web sites used in our experiments are: google.com, yahoo.com, facebook.com, amazon.com, qq.com, taobao.com, live.com, ebay.com and youtube.com.

To run experiments automatically, we wrote a Java program. This program will iterate all these 9 websites and 4 browsers. For each iteration, the following 5 steps are taken:

1. Launch Wireshark to record traffic data.
2. Wait 30 seconds then launch one web browser and connect to a website.
3. Wait 300 seconds then kill web browser.
4. Save traffic data to file, wait 30 seconds then Quit Wireshark.
5. Wait 300 seconds then go to step 1 and start another experiment.

For each experiment, we open a website with a specific web browser and then record all the TCP/IP packets between the web browser and the web server. Javascript and flash are two popular plug-ins. With JavaScript, web pages are more user friendly. With flash, we can watch videos on web pages. However, JavaScript and flash can also put users at risk. To avoid this, some users tend to disable them. Taking usages of JavaScript and flash into consideration, we define a browser to be in four different status. Except for Tor Browser Bundle, we perform 10 experiments for each browser, web page and plugin usages status combination. In total we have 1440 experiments ($4\times9\times40$). Our experiments were performed from February 2, 2014 to April 18, 2014.

3.2 Processing Data

To analyze these traffic data, for each experiment, we transfer the packets information to a data series based on the direction of packets (outgoing and incoming) and time series they are recorded. We sum up the packet size for a specific time frame interval based on the direction of packets. For outgoing packets, we use negative numbers to represent them by multiplying the sum of packets size with -1. We put the browser type at the end of the data series as class attribute. For each web site, we have 160 ($10 \times 4 \times 4$) experiments. Correspondingly we have 160 data series.

In this paper, Weka [6] 3.6.10 is used as the classification tool. To use Weka, we put all the 160 data series to a file with .arff file format (the standard Weka dataset file format). To get the best classification model, we tried ZeroR, OneR, C4.5, NaiveBayes and SVM (support vector machine) algorithms. These algorithms are all implemented in Weka. ZeroR is the base line of classification, basically, it is randomly guessing based on probability. oneR is trying to find one attribute to classify instances. The test model we used is 10-fold cross-validation. In this test model, the dataset is randomly reordered and then split into 10 folds of equal size. In each iteration, one fold is used for testing and the other 9 folds are used for training the classifier. The test results are collected and averaged over all folds. Cross validation is quite useful in dealing with small datasets since it utilizes the greatest amount of training data from the dataset [6].

4. RESULTS

4.1 Webbrowser type identification

The type of the webbrowser is one of the most common webbrowser fingerprinting characteristics. Webbrowser type and version information can be obtained by checking the http user-agent header. Our assumption is that we can not see the plain-text of the packets content. All we have is the time series of packet size. Classification models are built upon these time series and classification algorithms.

Figure 1 (a) shows the average accuracy rate of identifying browser type for all nine web sites with C4.5 classification algorithm. In this paper, we use accuracy rate to represent the performance of a classification model. Suppose 40 experiments with 10 experiments for each one of the four type of browsers, 8 experiments are correctly classified as using Chrome, 7 experiments are correctly classified as using Firefox, then the accuracy rate of identifying Chrome and Firefox is 80% and 70% correspondingly.

The accuracy rate of identifying browser type is slightly different for specific browsers. The best one is for Google Chrome which is identified with 87.27% average accuracy rate. That means that for all 360 experiments which use Chrome, 317 (360×0.88) of them are correctly identified, and 43 of them are identified as others. Tor browser bundle has lowest identification average accuracy at 69.99%. In our dataset, we have 4 different types of browsers, so the base line probability of identifying the correct browser type is 25%. Our worst accuracy rate, which is 70% for Tor bundler, is still 2.8 times the base line accuracy.

For different websites, the accuracy rate of identifying browser type also changes. Table 1 shows the exact accuracy rate for nine websites. Among these 9 websites, facebook.com has the best average accuracy which is about 85% and qq.com has the worst average accuracy which is about 73%. Though 70% accuracy rate is pretty good comparing to the 25% base line accuracy, there are still some improvements we can make to enhance it. Like Cai's work [20], we can round packets size to a multiple of 600 and remove noises of traffic data. In future work, we will try these modifications to obtain higher accuracy.

4.2 Webbrowser plug-ins identification

Besides the browser type, plug-in usage status is also important characteristic to fingerprint a browser. Our assumption is that we have 13 combinations of browser type and plug-in usages. For browser Chrome, IE and Firefox, each of them can have 4 possible plug-in usage combination regarding the enabling or disabling of flash and JavaScript plug-in. For Tor bundle, we use the default plug-in setting.

We run C4.5 classification algorithm against the packet size series, the test model we used is 10 fold cross verification, similar for identifying browser type. Figure 1 (b) shows that for all 9 websites, the average identification accuracy of combinations of browser type and plug-ins usages. Combination 1 to 4 are for Chrome, 5 to 8 are for Firefox, 9 to 12 are for IE and combination 13 is for Tor bundle. Besides Tor bundle, which we use default plug-in setting, the highest identification accuracy is for combination number 4 which is about 70.62% and the lowest one is for combination number 11, which is about 31.11%. Since there are 13 classes of browser type and plug-ins usage combination, the baseline probability of identifying one combination is less than 7.7% ($1/13 = 0.0769$) . Our worst case of accuracy, which is 31.11% is more then 4 times of the baseline probability. When the time frame of the packets size is 2 seconds, the average accuracy of all 13 combination is about 52.0%, which is 6.7 times higher than the baseline probability.

Table 1: Identification Accuracy for Specific Website

	Identification accuracy (%)			
Websites	Chrome	Firefox	IE	Tor bundle
amazon	89.74	82.05	64.10	75.68
ebay	84.62	87.18	82.05	70.15
facebook	92.50	80.00	87.50	81.08
google	95.00	70.00	77.50	64.86
live	84.62	82.05	89.74	63.64
qq	89.74	61.54	74.36	64.86
taobao	76.92	76.92	84.62	69.35
yahoo	85.00	72.50	75.00	70.27
youtube	90.00	67.50	85.00	73.30
Average	87.27	76.53	79.36	69.99

For a specific website, the average identification accuracy of all 13 combinations changes slightly. For websites like YouTube, eBay and Amazon, they have relatively higher average identification accuracy. The reason is that their web pages contains more videos/images, and they use JavaScript to control how these videos/images are displayed on web pages. Enabling or disabling JavaScript and flash have much more influence to the network traffic than other websites.

4.3 Varying time frames and classification algorithms

To get the best result, we tried time frames from 1 second to 10 seconds to generate the different data series. The basic idea is that when time frame is smaller, the data series we get contains finer information about the traffic. On the other hand, when time frame is smaller, it will take more time to run the classifier and model is also easier to suffer overfitting. Besides the better accuracy rate of classification, we also need some kind of trade-off between the accuracy rate and model complexity. Figure 1 (c) shows that for algorithm C4.5, the time frames we take has slight influence to the accuracy rate of identifying browser type.

When classifying a dataset, the classification algorithm chosen usually has great influence to the performance of the final classification model. Among all these 5 algorithms used in this paper, the C4.5 algorithm has best performance. It has around 80% accuracy and the accuracy is relatively stable for different time frames. Accuracy of NaiveBayes algorithm varies between 60.28% to 72.47% and it seems to increase as time frames become smaller. But as we take smaller time frames, say 0.5, its accuracy does not increase.

SVM works badly in our case, there are two reasons: 1) data series used in this paper are all based on raw data, we have no data rounding or normalization, and 2) to ensure all traffic data are recorded, for each experiment, we collect all traffic data in 5 minutes starting from the launch of accessing

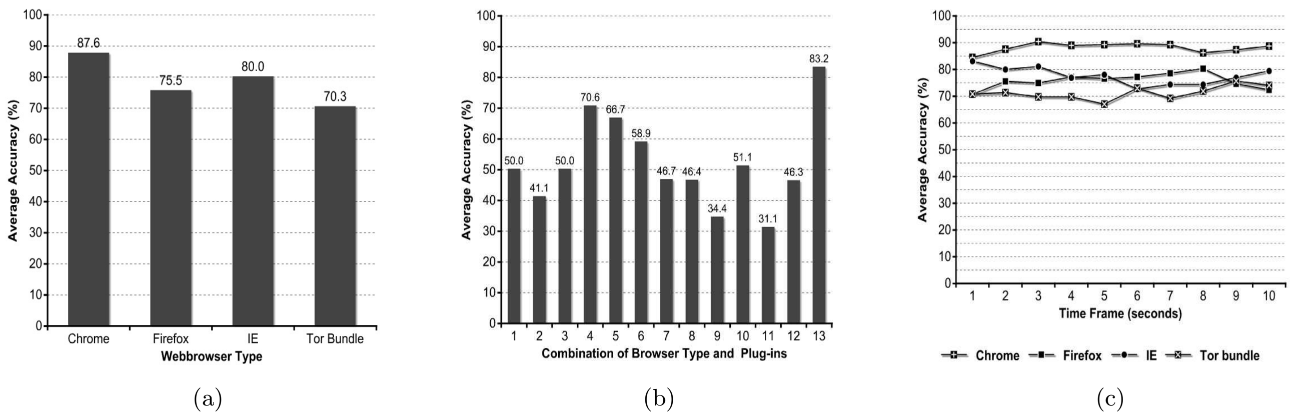

Figure 1: Average identification accuracy for: (a) different type of browsers when analysis time frame is 2 seconds; (b) 13 different combinations of browser and plug-ins when time frame is 2 seconds; (c) C4.5 algorithm for different type of browsers and different time frames.

a web page. This results in a lot of zeros and small random numbers in the data series because a web page usually can be loaded in less than 1 minute. These zeros and small random numbers in data series decrease the classification accuracy of SVM greatly.

5. DISCUSSION AND CONCLUSION

We have shown that it is possible to identify webbrowsers characteristics through network traffic analysis only, even if the network traffic is encrypted and through an anonymized service such as Tor. The consequences of this research are numerous. The webbrowsers and plugins used can be identified through network traffic analysis only. Users can potentially be identified and tracked across sessions and visits to different websites. The identification entity does not have to be part of the server. It can be any router along the path from the user to the server, for example, the ISP of the server or the user.

Our research also shows that the traffic data is seriously affected by webbrowser specific characteristics. This suggests that when we perform website fingerprinting based on traffic data, the webbrowser plays an important role and need to be considered in future work. There remains more work to be done in this research area. More experiments need to be performed. The number of experiments per website need to be increased. The number of websites tested need to be increased. The reason why the difference of traffic data exists is also an interesting topic.

6. REFERENCES

[1] Alexa. `http://www.alexa.com/topsites`.

[2] Anonymizer. https://www.anonymizer.com/.

[3] Evercookie. http://samy.pl/evercookie/.

[4] Panopticlick. https://panopticlick.eff.org/.

[5] Privoxy. http://www.privoxy.org/.

[6] Weka. `http://www.cs.waikato.ac.nz/ml/weka/`.

[7] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel. Fpdetective: Dusting the web for fingerprinters. In *Proceedings of the ACM Conference on Computer; Communications Security*, CCS '13.

[8] A. Z. Andriy Panchenko, Lukas Niessen and T. Engel. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of the ACM workshop on Privacy in the electronic society*, WPES '11.

[9] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium*, 2004.

[10] R. W. Dominik Herrmann 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*, CCSW '09. ACM, 2009.

[11] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton. Peek-a-boo, I still see you: Why efficient traffic analysis countermeasures fail. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, May 2012.

[12] P. Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10.

[13] X. Gong, N. Borisov, N. Kiyavash, and N. Schear. Website detection using remote traffic analysis. In *Proceedings of the 12th International Conference on Privacy Enhancing Technologies*, PETS'12. Springer-Verlag, 2012.

[14] A. Hintz. Fingerprinting websites using traffic analysis. In R. Dingledine and P. Syverson, editors, *Proceedings of Privacy Enhancing Technologies workshop (PET), 2002.*

[15] S. Le Blond, D. Choffnes, W. Zhou, P. Druschel, H. Ballani, and P. Francis. Towards efficient traffic-analysis resistant anonymity networks. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM.*

[16] S. J. Murdoch and G. Danezis. Low-cost traffic analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy.* IEEE CS, May 2005.

[17] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP '13. IEEE Computer Society, 2013.

[18] T. Wang and I. Goldberg. Improved website fingerprinting on tor. In *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, WPES '13.

[19] C. Wright, S. Coull, and F. Monrose. Traffic morphing: An efficient defense against statistical traffic analysis. In *Proceedings of the Network and Distributed Security Symposium - NDSS '09*. IEEE, February 2009.

[20] B. J. Xiang Cai, Xin Cheng Zhang and R. Johnson. Touching from a distance: website fingerprinting attacks and defenses. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12.