Tor に対する時間特性を利用する指紋攻撃とその対策

施　屹　　松浦　幹太

東京大学生産技術研究所
東京都目黒区駒場 4-6-1 153-8505
{shiyi, kanta}@iis.u-tokyo.ac.jp

Extended Fingerprinting Attack on Tor with Time Characteristics and Defense Mechanism

Yi Shi　　Kanta Matsuura

Institute of Industrial Science, The University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan
{shiyi, kanta}@iis.u-tokyo.ac.jp

Abstract By using interval classifications, we had proposed a novel way to implement a fingerprinting attack against Onion Routing anonymity systems such as Tor. Our motivation of previous work was to provide a realistic threat against existing popular anonymity systems. Through the simple threat model where an adversary could be mounted by nothing but controller of an entrance router, we had achieved our object. The attack we had proposed had a very good robustness against greatly-varied Internet conditions but the resolution is not so satisfactory to us. By employing time characteristics, we present an extended fingerprinting attack on anonymity systems here. Our new method has better performance, but still keeps the fingerprinting attack's advantage of being realistic in terms of the required small resource. We give experimental evaluation by comparing the extended attack with our earlier work. Also, we discuss defense mechanisms against fingerprinting attacks.

1 Introduction

Internet is one of the greatest inventions in the world. With Internet, we could keep touch with others around the world, get the newest information, do shopping freely, etc. Our life is really benefited from it. But on the contrary, with the emerge of Internet, our privacy seemed to be threatened and both the risk and the damage of leaking private information are greater than any other times before. Encryption helped us a lot in protecting privacy, but not everything. It can hide the communication contents such as data payloads, but it can do nothing with the packet headers, which leaks the identity of communication parties. Anonymity system tries to provide the foundation for users to share information over public networks without compromising their privacy.

Researches on anonymity systems are generally classified into two types: how to protect privacy and how to break it. To the first type, they are mainly focused on the introducing of new anonymity systems, such as [4, 3, 14, 6, 7, 5, 15]. On the contrary, many attacks towards existed anonymity systems are also presented like [1, 9, 13, 10, 11, 8, 12, 2] and so on. These works could stimulate researchers considering new defense mechanisms.

In [16],[17], we had proposed a novel way to implement a fingerprinting attack against Onion Routing anonymity systems such as Tor. Our motivation of previous work was to provide a realistic threat against existing popular anonymity systems. In this paper, we call the formula presented as similarity score calculation formula with interval method, to distinguish with the following time window method.

S_I[1] is calculated with the following formula:

$$S_I = \frac{\vec{V} \cdot \vec{F}}{\|\vec{V}\|\|\vec{F}\|} \cdot w_I \quad (1)$$

By employing time characteristics, we present an extended fingerprinting attack on anonymity systems here. The following sections are arranged like this: Section 2 talks about the extended fingerprinting attack with time window method, then a short discussion about the combination of two methods in Section 3. In Section 4, we will see the experimental results to show the efficiency of our works. Section 5 will have a quick word about the dummy packets. Finally we will give the conclusion in Section 6.

2 Fingerprinting Attack with Time Windows

With the definition and application of *interval* into traffic pattern, we could gain some advantage towards anonymity system users, but the result is still not so satisfied to us. The interval vector method omitted the information of relative positions between intervals, so we could get a robust result, which will not change greatly by some abnormal events (e.g. retransmission, lag, etc.) that may occur quite often in practical network environment. Also, we will not get very good resolution for using so limited information.

So we want to introduce some other factors to get better resolution and success rate for our attack plan. Time is a good candidate for us, it is widely used in all kinds of passive attacks. The problem is: how could we introduce the time into our attack?

First, we tried to make the assumption that all the packets remained the same positions; the time between packets are kept relatively constant (remain same or with same proportion). Then we may want to use a long vector to describe the time between each packet and calculate the similarity by use correlation or other method. Unfortunately, the result is not as good as we expected.

Then we want to use a slightly more rough way to measure that: we tried to make the assumption that the time between intervals (as we claimed above) are kept relatively constant. This is because that the several packets in an interval are transferred in a very short period but the waiting-for-response time is mainly related to the network environment.

Although the result is better than the first one, it is still not a good method. In these two ways, we treated the whole traffic pattern as if it was a "spring". When the network lag is high, the "spring" is stretched and vice versa. But the thing is: practical network is not so stable as we thought, the relative position of intervals also not remained same all the time. We want to find a better way to solve that.

Finally, we have found that by dividing into several windows, calculate the correlation between packets number in each window is a good way to make the resolution better. We also made some assumptions that are:

- Each page is consist by several files with different sizes. (Same as ordinary fingerprinting attack)
- In network transfer, (especially with good network environment), time is largely consumed by the waiting-for-response time than the time which is using for packet transport.

With these two assumptions, even the similar webpages (in the number of files, file sizes) with different sequences by using this method could be distinguished.

In this method, the basic concept is divide a given traffic pattern by relative time (e.g. 25%, 50%, etc.) That is because the time itself varies greatly due to the different path. Under the given assumption, we could treat the packet transfer time as "very short" and see the waiting-for-response time as the main part of a traffic pattern's time line. Then if the path is slow, the total time is long and vice versa, but the packets in each time window will not change greatly in normal cases. Then by calculate the correlation between two time window series, we could make the guess.

Let us discuss it in more detail way: First, decide how many windows should be divided - the total window number n. So the length of each part would be the $(total\ time/n)$. Then we will get a time window divided vector as $(v_1, v_2, ..., v_i, ..., v_n)$. v_i refers to the number of packets in the i-th time window. Here we

[1] I stands for Interval

could treat the inflow and outflow packets separately, but I believe that the inflow could describe the feature of object better. After that, we could calculate the similarity score S_{TW}[2] with two time window vectors by getting the correlation coefficient of them. That is:

$$\begin{aligned} S_{TW} &= w_{TW} * \frac{Cov(V', F')}{StdDev'_V * StdDev'_F} \\ &= w_{TW} * Corr(V', F') \end{aligned} \quad (2)$$

Two vectors represent as V' and F', also as $(v'_1, v'_2, ..., v'_n)$ and $(f'_1, f'_2, ..., f'_n)$. $Cov(V', F')$ stands for the covariance of two vectors, which is $E[(V' - E[V'])(F' - E[F'])]$. $StdDev$ stands for standard deviation, calculated by $\sqrt{E[(V' - E[V'])^2]}$. And $Corr(V', F')$ means correlation coefficient, the same as covariance divided by the multiplier of two standard deviations.

We used w_{TW} here again and that is slightly different with w_I used above. It also ranges from 0 to 1, calculated by divide the smaller number of inflow packets of the two vector with the bigger number of inflow packets. Weight is useful to filter out obviously irrelevant samples, and almost without any side-effect. The correlation gives us the information of the trends between variations of packets but not the absolute number of packets. Then weight could help us to introduce absolute number of packets into calculation. Actually, either weight calculated by number of intervals or by number of packets does not differ greatly. So they are somewhat interchangeable.

The time window divided attack results better than the interval method; we shall see that in the following section. But the interval method is much more robust than time window divided method. An abnormal long lag will make this sample completely worthless in time divided method, but one or two retransmission does not hurt seriously in interval method.

[2]TW stands for Time Window

3 Combination of Two Methods

We have presented two methods before, and both of the two methods have their own suitable cases. It is hard for attacker to analysis each case and determine which method to use, so the combination of two methods are recommended to introduce as many factors as possible.

From intuition, there are equations like this:

$$S_C = S_I * S_{TW} \ / \ w_I \quad (3)$$

$$S_{C'} = S_I + S_{TW} \quad (4)$$

Besides these two basic formulas, we could also adjust each item's weight. Towards different samples, there may be different effective formulas, but we want to discuss in a more general case.

Compare Formula 3 and Formula 4, I will tend to use the first one for two reasons. First, the Formula 3 will give us a result between -1 and 1, which is more formal way and could still introduce other factors in future without change the range of result. Second, in my opinion, I think extreme case should be considered seriously. Compare to the similarity score of 1 and 0, the score of 0.5 and 0.5 maybe the better choice. (Although seem both of them are not the right choice.)

4 Evaluation

In this section, we shall see the experiment result when we using time window attack, and also the combination of these two attack methods. Also we could see the improvement of success rate by employing pity hit. We use Alexa Ranking and choose top 20 sites to implement the experiments.

In the experiment, we choose top $n = 5, 10, 15, 20$ sites, and build fingerprint of the site. Then we surfed webpages and recorded the user activity vector, compared with the fingerprint, and guessed which website user is surfing by time window and combination methods. The success rate represents in the Figure 1.

From the Figure 1, we could see that if we choose the webpage whose fingerprints have the highest similarity score, time window

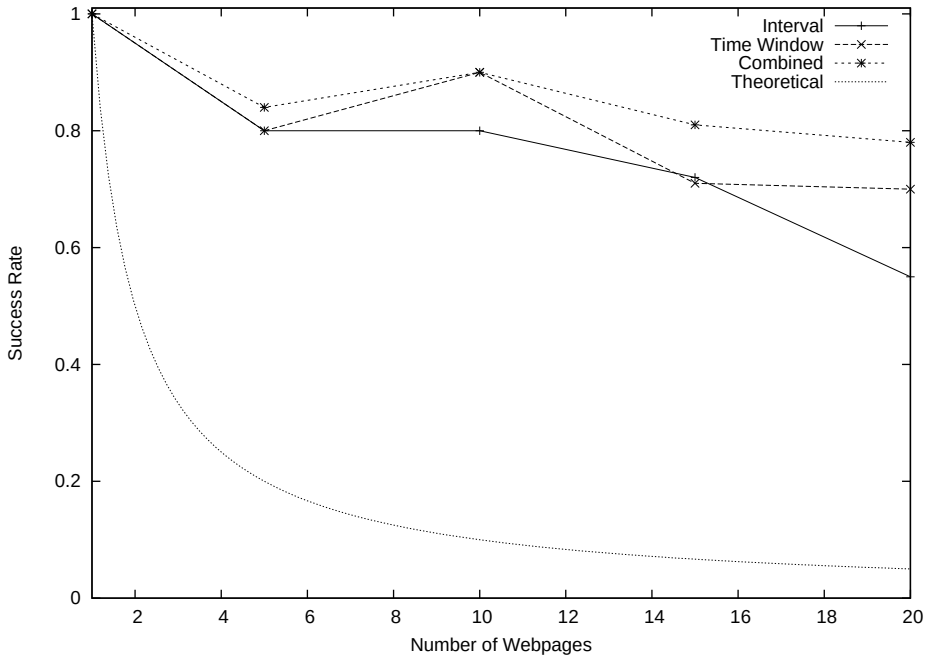

Figure 1: Success rate with 3 different methods

shows better results than the interval method. And combination of two methods performed best in this situation. There are some points we shall notice here: First, time window do not always outperform the interval method, we could see that from the graph. Actually, both of them have their own suitable cases as we have discussed earlier. Second, the success rate does not always drop as the number of webpages increases. Consider in the 5 webpages case, there are two pages are very close to each other. And when the number of webpages increases to 10, maybe the new pages are all easy to be distinguished. Then the success rate would be increased in the total.

5 Countermeasure

Although there exists some trivial but sometimes effective countermeasures like making odd requests, building own nodes, etc, we must aware that these countermeasures are depended on users' own activities, which are not so reliable in most of times. So we want to employ a method which does not rely on the users' activities. In this section, we will introduce the dummy packet as our recommended defense mechanism.

It is hard to consider all the factors in the dummy packet employing here, so we want to do some experiments and just illustrate the efficiency of this idea. We will use data captured from Tor to make the experiments. In the experiment, we will randomly select one traffic pattern. If there is no dummy packet in it, either interval or time window method could have the answer of 1. Then we want to add some dummy packets into this traffic and calculate again with S_I and S_{TW}. Of course, the lower S, we have the better protection.

In the experiments, we have discovered two parameters directly lead to the efficiency of the defense mechanism. The first one is *number of dummy packets generated every second*, or we could call it density. It is very trivial that as this number increases, the protection effect will also increase. But with the traffic emerging into the dummy packets, the marginal utility will also become weaker and weaker. And no doubt higher density will increase the cost of the anonymity system, and then the usability will also be hurt.

Another factor is the *coverage ratio*, here we define it as for the whole traffic pattern, how many parts in it could be inserted with dummy packets. Increase it will lead to higher cost

and vice versa. But what makes this factor really interesting is that the higher coverage ratio will not always lead to the better protection.

In our attack framework, we have discussed mainly two calculation methods of similarity score, one is interval and another is time window. These two different attack methods have different sensitivity towards different coverage ratios. To the interval method, if the coverage ratio is low, that means many dummy packets are focused in a short period of time. The result is the length of a few intervals will be increased. But since we have limited the maximum element in an interval vector, the affected number of intervals is small, that will not change the result dramatically. For example, the change with coverage ratio which is 0.1 may only cause v_1 decreased by 2 and v_5 increased by 2. And most of the intervals still remains the same. On the contrary, if the coverage ratio is high, then more intervals' length are changed so the interval vector will be transformed greatly, so the S_I.

Let us see how the coverage ratio works in the time window method. When the coverage ratio is high, that means, almost in every time window, there would be approximately the same (at least the estimation would be same) number of dummy packets. And due to the calculation of correlation, if a series of numbers changed in the almost same amount of value, it then has really small effect on the correlation coefficient. But when the coverage ratio is low, the thing comes completely different. We will see that in this case, dummy packets flow into one time window and if the number of packets in that time window is fewest in the beginning, it may become the most one in the end. The correlation coefficient would change dramatically, it even may turn into minus. And just as a result of average, low coverage ratio is still quite good in the time window method.

We will treat the cost as the multiplication of these two factors. That is:

$$\begin{aligned} \textit{Avg Cost} &= \textit{Dummy Packets per Second} \\ &\quad * \textit{Coverage Ratio} \end{aligned} \tag{5}$$

For example, if we have a density of 15 dummy packets per second and a coverage ratio of 0.6. That means the average cost would be 9 dummy packets per second. And assume all the dummy packets are 1.5 KB, and then the additional cost for one Tor connection is around 13.5 KB/s.

We have tried an experiment by using a traffic pattern captured by Tor of the Yahoo's main page, and using two different similarity calculation methods without weight. The two parameters are adjusted to show us the effect of dummy packets under this circumstance. All the slots are calculated with 30 times of sampling and take the average value.

We could also refers to the Equation 5 and see these combinations: density 50, coverage 0.1; density 25, coverage 0.2; density 10, coverage 0.5; density 5, coverage 1. The result in interval method and time window method give us completely different tendency. The results from interval method are 0.987, 0.967, 0.923 and 0.889. Meanwhile, the results from time window method are 0.626, 0.678, 0.841 and 0.984. The two tables about the standard deviations tell us the result is basically stable, especially with the interval method. We have no more space to discuss about the how to select threshold and parameters. In conclusion, we think coverage around 50%, approximately 20 dummy packets per second could be recommended. And by the Equation 5, we could estimate the cost is approximately 15KB/s on average for a Tor connection.

6 Conclusion

In this paper, By using time window classifications, we extended our previous work and implemented a new fingerprinting attack against Onion Routing anonymity systems such as Tor. We preserved our motivation of previous work, which was to provide a realistic threat against existing popular anonymity systems. In the simple threat model used in our extension, an adversary could be mounted by nothing but controller of an entrance router. We can trade off the robustness against greatly varied Internet conditions to receive higher resolution, and by employing combination, we proposed a better way in general cases. Finally, we discussed possible defense mechanisms against fingerprinting attacks. The mechanism of employing dummy packets is recommended to be introduced in future anonymity systems.

References

[1] A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. *Lecture Notes in Computer Science 2137*, pp. 245–257, Springer, April 2001.

[2] S. Chakravarty, A. Stavrou, and A. D. Keromytis. Identifying proxy nodes in a tor anonymization circuit. In *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pp. 633–639, 2008.

[3] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, vol. 1, pp. 65–75, 1988.

[4] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[5] W. Dai. Pipenet 1.0. Post to Cypherpunks mailing list, January 1998.

[6] M. J. Freedman and R. Morris. Tarzan: a peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM conference on Computer and Communications Security*, pp. 193–206, 2002.

[7] C. Gülcü and G. Tsudik. Mixing E-mail with Babel. In *Proceedings of the Network and Distributed Security Symposium - NDSS '96*, pp. 2–16, February 1996.

[8] A. Hintz. Fingerprinting websites using traffic analysis. *Lecture Notes in Computer Science 2482*, pp. 171–178, Springer, 2003.

[9] B. N. Levine, M. K. Reiter, C. Wang, and M. K. Wright. Timing attacks in low-latency mix-based systems. *Lecture Notes in Computer Science 3110*, pp. 251–265, Springer, February 2004.

[10] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia. A new cell counter based attack against Tor. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pp. 578–589, 2009.

[11] P. Mittal and N. Borisov. Information leaks in structured peer-to-peer anonymous communication systems. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pp. 267–278, October 2008.

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

[13] R. Pries, W. Yu, X. Fu, and W. Zhao. A new replay attack against anonymous communication networks. *Proceedings of the IEEE International Conference on Communications 2008*, pp. 1578–1582, May 2008.

[14] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, vol. 1, no. 1, June 1998.

[15] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based anonymous Internet usage with collusion detection. In *Proceedings of the Workshop on Privacy in the Electronic Society*, Washington, DC, USA, November 2002.

[16] Y. Shi and K. Matsuura. Fingerprinting attack on the Tor anonymity system. In *Proc. Computer Security Symposium 2009*, pp. 877–882, October 2009.

[17] Y. Shi and K. Matsuura. Fingerprinting attack on the Tor anonymity system. *Lecture Notes in Computer Science 5927*, pp. 425–438, Springer, December 2009.