On the Effectiveness of Traffic Analysis Against Anonymity Networks Using Flow Records

Sambuddho Chakravarty[1], Marco V. Barbera[2], Georgios Portokalidis[3], Michalis Polychronakis[1], and Angelos D. Keromytis[1]

[1] Columbia University, NY, USA
{sc2516, mikepo, angelos}@cs.columbia.edu
[2] Sapienza Universita Di Roma, Rome, Italy
barbera@di.uniroma1.it
[3] Stevens Institute of Technology, NJ, USA
gportoka@stevens.edu

Abstract. We investigate the feasibility of mounting a de-anonymization attack against Tor and similar low-latency anonymous communication systems by using NetFlow records. Previous research has shown that adversaries with the ability to eavesdrop in real time at a few internet exchange points can effectively monitor a significant part of the network paths from Tor nodes to destination servers. However, the capacity of current networks makes packet-level monitoring at such a scale quite challenging. We hypothesize that adversaries could use less accurate but readily available monitoring facilities, such as Cisco's NetFlow, to mount large-scale traffic analysis attacks. In this paper, we assess the feasibility and effectiveness of traffic analysis attacks against Tor using NetFlow data. We present an active traffic analysis technique based on perturbing the characteristics of user traffic at the server side, and observing a similar perturbation at the client side through statistical correlation. We evaluate the accuracy of our method using both in-lab testing and data gathered from a public Tor relay serving hundreds of users. Our method revealed the actual sources of anonymous traffic with 100% accuracy for the in-lab tests, and achieved an overall accuracy of 81.6% for the real-world experiments with a false positive rate of 5.5%.

1 Introduction

Anonymous communication networks hide the actual source (or destination) address of internet traffic, preventing the server (or client) and other entities along the network from determining the actual identities of the communicating parties. Among others [2, 3], Tor [8] is probably the most widely used low-latency anonymity network. To offer acceptable quality of service, Tor and similar systems try to preserve packet interarrival times. Unfortunately, this makes them vulnerable to traffic analysis attacks [5, 11, 13, 17, 20, 21], whereby an adversary with access to traffic from/to entry and exit nodes, can correlate seemingly unrelated traffic flows and reveal the actual endpoints.

As Tor nodes are scattered around the globe and the nodes of circuits are selected at random, mounting a traffic analysis attack, in practice, would require a powerful adversary with the ability to monitor traffic at a multitude of autonomous systems (AS).

Murdoch and Zieliński, however, showed that monitoring traffic at a few major internet exchange (IX) points could enable traffic analysis attacks against a significant part of the Tor network [18]. Furthermore, Feamster et al. [12], and later Edman et al. [10], showed that even a single AS may observe a large fraction of entry and exit-node traffic—a single AS could monitor over 22% of randomly generated Tor circuits. Recently, Johnson et al. [15], extended this study and observed, through simulation, that compromised high bandwidth Tor relays and IX operators, observing both entry and exit traffic, could de-anonymize 80% of random Tor circuits.

Packet-level traffic monitoring at this scale requires the installation of passive monitoring sensors capable of processing tens or hundreds of Gbit/s traffic. Although not impossible, setting up a passive monitoring infrastructure at such a scale is challenging in terms of cost, logistics, and effort. An attractive alternative for adversaries would be to use the readily available, albeit less accurate, traffic monitoring functionality built into the routers of major IXs and ASs, such as Cisco's NetFlow. Murdoch and Zieliński showed through simulation that traffic analysis using sampled NetFlow data is possible, provided there are adequate samples. Still, there have been no prior efforts to explore the various practical aspects of mounting traffic analysis attacks using NetFlow data.

As a step towards filling this gap, in this paper we study the feasibility and effectiveness of traffic analysis attacks using NetFlow data, and present a practical active traffic analysis attack against Tor. Our approach is based on identifying pattern similarities in the traffic flows entering and leaving the Tor network, using statistical correlation. To alleviate the uncertainty due to the coarse-grained nature of NetFlow data, our attack relies on a server under the control of the adversary that introduces deterministic perturbations to the traffic of anonymous visitors. Among all client-to-entry-node flows, the actual victim flow can be distinguished due to its high correlation with the respective exit-node-to-server flow, as both carry the induced traffic perturbation pattern.

We evaluated the effectiveness of our traffic analysis attack in a controlled lab environment, as well as using public Tor relays. In the in-lab environment, our method revealed the actual sources of anonymous traffic with 100% accuracy. When evaluating our attack with traffic going through public Tor relays, our method detected the actual source in 81.6% of the cases, with a a false positive rate of 5.5% and false negative rate of 12.7%. Due to the sensitivity of the correlation process, especially for flows with sparse samples, we couple correlation with heuristics to filter out flows that are unlikely to correspond to a victim, thus reducing false positives.

2 Related Work

Tor [8] safeguards the anonymity of internet users by relaying TCP streams through a network of overlay nodes, run by volunteers. It typically hides the identity (IP address) of the initiator of a connection, although the opposite is also possible through the use of *hidden services*. Murdoch and Danezis [17] developed the first practical traffic analysis attack against Tor. Their technique involved a corrupt server and a client that buildt one-hop circuits via candidate relays to determine relays participating in a circuit. Hopper et al. [13] used this method, along with one-way circuit latency and the Vivaldi network coordinate system, to determine the possible source of anonymous traffic. In

2009, however, Evans et al. [11] demonstrated that Murdoch and Danezis' method was not accurate, due to an increase in the number of relays and the large volume of Tor traffic. They proposed a modification to amplify the traffic by loops in circuits.

Previously, we proposed a method for performing traffic analysis using remote network bandwidth estimation tools to identify the Tor relays and network routers involved in Tor circuits [7]. Our method assumed that the adversaries were in a position to perturb the victim traffic by colluding with the server, and are in control of various network vantage points, from where they can remotely observe variations in network bandwidth. Mittal et al. [16] demonstrated a modified version of the Murdoch and Danezis' attack that relies on path bandwidth variation.

In 2007, Murdoch et al. [18] proposed the use of NetFlow data from routers in IXes to perform traffic analysis attacks against traffic entering and leaving the Tor network. They discovered that there is a small number of IXes that can potentially observe a large part of Tor traffic, and allow the use of existing facilities, such as Cisco NetFlow, to mount traffic analysis attacks. They proposed a traffic and attack model that receives as input NetFlow traffic gathered from monitoring a Tor relay. They described, through simulations, how varying the number of flows, bandwidth, and end-to-end delay, affects the accuracy of determining the source of anonymous traffic. In a follow-up work, Johnson et al. [15] recently showed that a small number of compromised Tor relays that advertise high bandwidth and IXes observing both entry end exit traffic, can de-anonymize 80% of various types of Tor circuits within six months.

Previous efforts did not explore the feasibility and effectiveness of using a facility such as NetFlow to determine the source of anonymous traffic from a practical perspective. Our work attempts to assess the possibilities of accurately de-anonymizing Tor users using NetFlow data by implementing and experimentally evaluating a traffic analysis attack in realistic settings.

3 Approach

Threat Model and Attack Methodology: The goal of the attacker is to determine the network identity (i.e., public IP address) of a client using Tor to access a server. We assume the attacker can observe NetFlow traffic records on routers at or near Tor relays. In our model, the attacker deliberately injects a traffic variation pattern on one side of a victim Tor connection, which travels via the relays to the peer. The easiest way for the attacker to achieve this is by controlling the server; the attacker would then serve sufficient content volume (e.g., a large volume of "invisible" HTML content) and inject traffic perturbation patterns in the connection between the Tor exit node and the server. We also assume that attackers can select specific anonymous connections they are interested in (e.g., those that correspond to a particular user identity in the server). Alternatively, attackers could de-anonymize all clients accessing the server; our current work demonstrates de-anonymization of a single client at a time. Simultaneous anonymization of multiple clients (with or without correlation between client identities and anonymous sessions) is left for future work. A powerful adversary could monitor a large part of the relays participating in the Tor network, one of which with high probability would correspond to the entry node of the targeted user. Alternatively, an at-

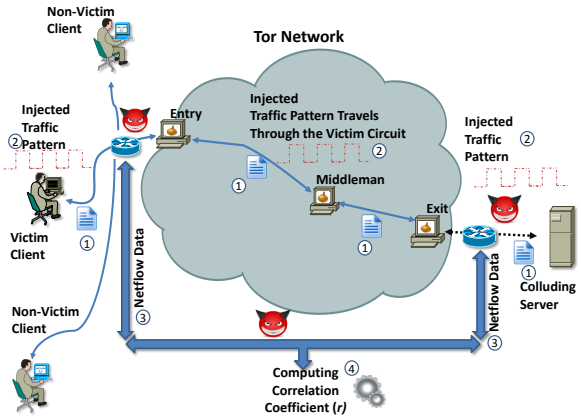


Fig. 1. NetFlow-based traffic analysis against Tor: The client is forced to download a file from the server ①, while the server induces a characteristic traffic pattern ②. After the connection is terminated, the adversary obtains flow data corresponding to the server-to-exit and entry-to-client traffic ③, and computes their correlation coefficient ④.

tacker could follow a more focused approach by employing existing techniques [7, 16] to identify the actual relays used by the victim's circuit, and only monitor those.

In a second, related scenario, the attacker is a malicious Tor client seeking to identify a Tor hidden server. In this case, the attacker injects a traffic perturbation pattern and observes it between the hidden server and its entry node, against using the NetFlow records to perform the correlation. Note that the attacker need not actually control one end of a Tor circuit. For example, the attacker could inject a pattern in a chosen anonymous connection between the server and an exit node, without the server knowing about it. This scenario introduces additional complexity in terms of victim selection, especially when the connections between the Tor exit node and the server are encrypted. We defer further study of this scenario to future work.

As shown in Figure 1, after the transfer ends, the adversary obtains the flow records of all the client-to-entry-node connections that were monitored (from one or more entry nodes), and computes their correlation with the given exit-node-to-server flow. Various factors, such as flow cache eviction timeout values and the inherently bursty nature of traffic (especially web traffic), commonly result in an inadequate number of flow samples than what is ideally required for computing the correlation coefficient. The longer the duration of the fingerprinted transfer, the higher the chances that enough flow samples will be gathered. In our experiments, we assume that the victim downloads a large file (in the order of tens of megabytes), generating sustained traffic for a duration of about 5–7 minutes. Depending on the capabilities of the involved routers, the same accuracy could be achieved using shorter data transfers.

Implementation: In our prototype, the server fluctuates a client's traffic using Linux Traffic Controller [14]. We explored two different kinds of traffic perturbation patterns. The first was a simple "square wave" pattern, achieved by repeatedly fluctuating the

victim's transfer rate between two values. The second a was more complex "step" pattern, achieved by repeatedly switching between several predetermined bandwidth values. These different perturbations help evaluate our attack accuracy through both simple and complex injected traffic patterns.

For our initial in-lab experiments, flow records were generated and captured using the open source tools `ipt_netflow` [4] and `flow-tools` [1], respectively. In such a controlled environment, free of congestion and external interference, our approach achieved 100% success in determining the source of anonymous connections (more details for this experiment are included in our technical report [6]).

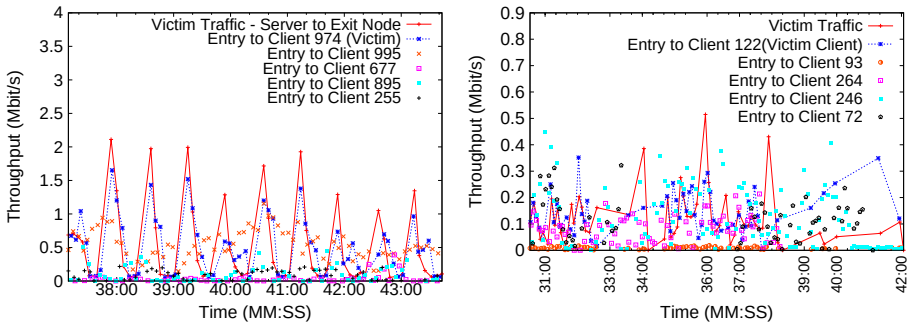
In the experiments presented in this paper, we obtained data from a public Tor relay serving hundreds of Tor clients. The flow records for the server-to-exit traffic were generated and captured using the aforementioned flow tools. The flow records for the entry-to-client traffic were generated first using the flow tools, running on the same host as the entry node, and later by our institutional edge router. For the latter, the flow data from the router was often sparse due to aggressive sampling. Multiple intervals were typically aggregated into a single flow record. This generally happens due to the combination of flow expiration timeout values and the router's network load. As such aggregation is not deterministic, it is difficult to divide a large interval into smaller ones without knowing the ordinate values of the aggregated intervals. Since correlation analysis requires the two time series to have the throughput values taken at the same points, we devised the following strategy to align the time points.

Flow records are arranged as time intervals with the bytes transferred in each of them [6]. To correctly align the time points, we first take the intervals of all server-to-exit records and divide them into steps of one second. We then consider the starting and ending times of every entry-to-client flow record and attempt to align them with the one-second steps of the server-to-exit flow. For every successfully aligned time point, we assume the corresponding entry-to-client (and respective server-to-exit) throughput value to be the average throughput of the entry-to-client (and respective server-to-exit) interval that covers this time point, obtained by dividing the total bytes transferred in the corresponding interval by the length of that interval. Unaligned time points are ignored. Finally, we compute the correlation of throughput values of the two aligned sets.

4 Experimental Evaluation

To evaluate our attack using data obtained from public Tor relays we used the set-up shown in Figure 1. Victim clients were hosted on three different PlanetLab locations: Texas (US), Leuven (Belgium) and Corfu (Greece). The clients communicated via Tor circuits through our relay to a server under our control in Spain.

Flow Collection using NetFlow Tools: In our first set of experiments, the flow records were obtained from the server and the entry node using the open source flow generation and capture tools mentioned in the previous section. We configured the active and inactive timers to 5 seconds each. This resulted in a uniform view of the traffic with an adequate number of samples for accurately computing correlation. The first experiment involved the server injecting a "square-wave" like traffic pattern having an amplitude



(a) Square-wave pattern (b) Step pattern

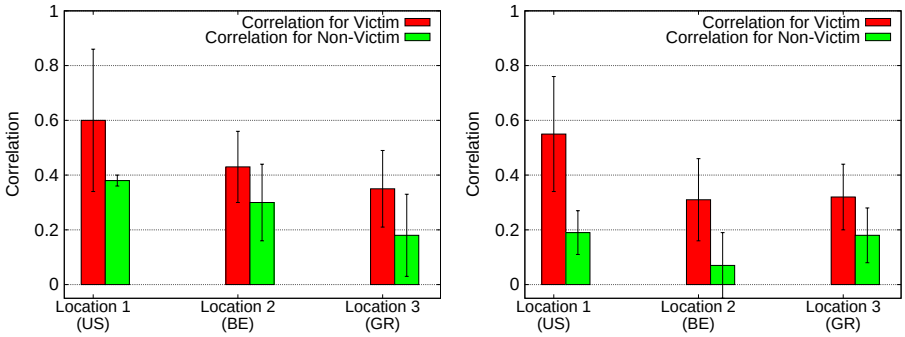
Fig. 2. A victim flow with a server-induced "square-wave" (a) and "step" (b) pattern. The remaining points correspond to the non-victim flows with the four highest correlation coefficients.

of roughly 2 Mbit/s, achieved by switching the server-to-exit traffic bandwidth between 2 Mbit/s and 30 Kbit/s, every 20 seconds. Figure 2(a) shows sample traffic throughput variations for five flows, from one such experiment. These five flows are the ones with the highest correlation to the server-to-exit flow (solid line) that carries the injected traffic pattern. The victim flow had the highest correlation coefficient of 0.83 (among 1100 other clients), while the second-highest correlation, for to a non-victim client, was 0.17.

We repeated similar experiments with the server injecting a more complex "step" like pattern, achieved by switching the server-to-exit traffic throughput between roughly 1 Mbit/s, 50 Kbit/s, 300 Kbit/s and 100 Kbit/s, every 20 seconds. This pattern was again repeated several times. Figure 2(b) shows one such sample where the server injected the "step" like pattern. The victim flow had the highest correlation coefficient of 0.84 (among 874 other clients), while the second-highest correlation, corresponding to a non-victim client, was 0.25. In general, we observe higher correlation of the server-to-exit and the victim traffic, when the server injects the "step" like pattern.

These experiments were repeated 90 times (15 times for each traffic pattern, for each of the three client location). The average correlation between the server-to-exit and entry-to-victim traffic statistics (corresponding to the flows that were most correlated to the flow carrying the traffic pattern) was higher than the average correlation to the non-victim client statistics, as shown in Figures 3(a) and 3(b). We were able to correctly identify the victim in 76 out of the 90 tests. The average correlation of the injected pattern for the victim traffic was lower than those for in-lab tests. This happens because the traffic pattern is distorted when it leaves the Tor entry node and proceeds towards the victim, reducing the victim's correlation coefficient.

We also found four instances where the correlation of the injected traffic pattern with the victim client traffic was lower compared to some other non-victim clients' traffic. Such false positives are primarily a combined effect of the background network congestion and routing in Tor relays, which attempts to equally distribute the available bandwidth among all circuits. To deal with such inaccuracies, we also computed the average throughput of the clients' traffic (over the duration of the experiment), and subtracted it from the average throughput of the server-to-exit traffic. For the victim traffic,



(a) Average correlation: Square-wave pattern

(b) Average correlation: Step pattern

Fig. 3. (a) Average Pearson's Correlation between server injected "square-wave" like pattern and the victim and non-victim flows for the different planetlab client locations. (b) Average Pearson's Correlation between server injected "step" pattern and the victim and non-victim flows for the different planetlab client locations.

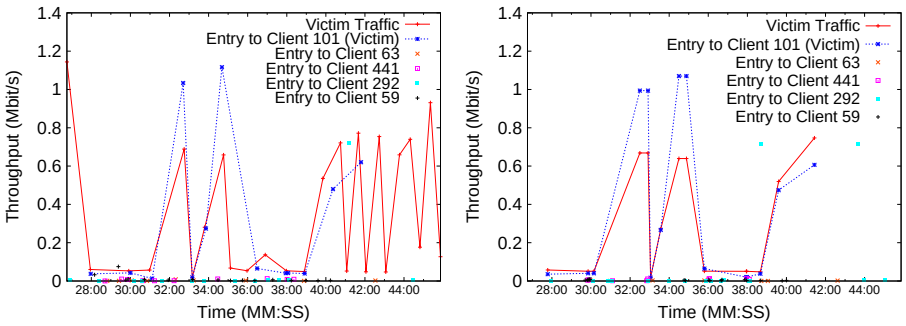
this difference is often amongst the smallest. This difference between the victim traffic and server-to-exit traffic can be used to filter out flows that could lead to inaccurate correlation coefficients arising from an inadequate number of flow samples. We used this observation in the experiments that involved sparse data from Cisco routers, to remove flows where the average throughput was not comparable to that of the victim's.

Flow Collection from Cisco Router: To evaluate the attack effectiveness when using data from our institutional edge router, we used the same experimental set-up that we used to test our attack using data obtained from open source packages. However, the entry-node-to-client traffic statistics were gathered from our institutional router. The router was configured with an active and inactive timeouts of 60 and 15 seconds respectively. We configured the NetFlow packages on the server with the same values. But, from our initial experience, we realized that the data obtained from the router was sparse and non-uniformly aligned, compared to the flow records from server-to-exit. We thus applied our rectification strategy (described previously) to align the flows. The rectified flow values were then directly used as input to the correlation coefficient formula.

These experiments were essentially the same as those described in the previous subsection. The first experiment involved the server injecting a "square-wave" like traffic pattern with an amplitude of about 1 Mbit/s. However, here the server switched the throughput every 30 seconds, instead of 20 seconds, enabling us to capture adequate ($\geq$ 10) samples for computing the correlation coefficient[4]. Figure 4(a) presents a sample bandwidth variation pattern for the server-to-exit traffic and the entry-node-to-client traffic. It shows server-to-exit traffic with more data points and fewer entry-to-client points. Figure 4(b) presents the same data pattern after it has been rectified.

As mentioned previously, we eliminated flows whose average of the traffic throughput was not comparable to that of the server-to-exit throughput variation. We computed

[4] This was done solely to compensate for the lack of samples obtained when the experiments ran for a shorter duration of 20 seconds (as previously)



(a) Square-wave pattern (before rectification)

(b) Square-wave pattern (after rectification)

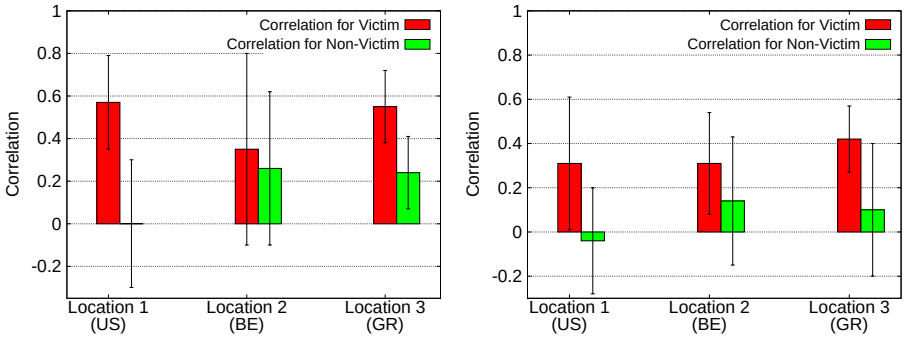
Fig. 4. (a) Server induced "square-wave" pattern of amplitude 1 Mbit/s along with other non-victim flows from the entry-to-victim and non-victim hosts having the four highest correlation co-efficient. Victim location: Texas, US. (b) Flows in Figure 4(a) adjusted and corrected using our rectification strategy.

the difference between average throughput for the server-to-exit and the entry-to-client traffic (for all clients). From our experience, for the victim traffic, the difference was within 120 Kbit/s. We removed flows where this difference was over 120 Kbit/s.

These experiments were also repeated with the server injecting a "step" like pattern, achieved by switching the traffic between 1 Mbit/s, 50 Kbit/s, 300 Kbit/s and 100 Kbit/s, every 30 seconds. The average correlation between the server-to-exit and entry-to-client traffic statistics was higher than the average correlation to the non-victim client statistics. These can be seen in Figures 5(a) and 5(b). We correctly identified the victim flow in 71 out of the 90 trials (success rate of 78.9%). There were six false positives in our measurements, where non-victim clients showed highest correlation to the server-to-exit traffic. In these false positive, the number of sample intervals for the entry-to-client traffic were less than half the number of sample intervals corresponding to the server-to-exit traffic. These fewer sample intervals resulted in correlation representing an inaccurate relationship. In 13 of the remaining cases we were not able to correctly select the victim either because the correlation coefficient was statistically not significant (< 0.2), or the victim flow was filtered out as its average throughput varied from the the average server-to-exit throughput by more than 120 Kbit/s.

Monitoring multiple Tor relays: Finally, we evaluated our attack in a scenario involving an additional relay. We launched a second relay in our institution. The purpose of this second Tor relay was to judge the effectiveness of our attack in the presence more clients. The two relays together served about 1500 clients. This scenario indicates what to expect when an adversary monitors multiple relays.

Our experiments involved injecting the "step" like pattern, described above. These experiments were repeated 24 times, 8 times for to each of the victim client location. We observed higher average correlation between server-to-exit and entry to victim client traffic, compared to non-victim clients' traffic. We were able to correctly identify the victim client in 14 out of the 24 trials (success rate 58.3%). There were three false positives, where the correlation of the server-to-exit traffic was higher to a non-victim than



(a) Average correlation: Square-wave pattern

(b) Average correlation: Step pattern

Fig. 5. (a)Average Pearson's Correlation between server injected "square-wave" pattern and the victim and non-victim flows, for the different planetlab client locations. (b) Average Pearson's Correlation between server injected "step" like pattern and the victim and non-victim flows, for the different planetlab client locations.

to the victim. The remaining seven were false negatives, where the correlation coefficient was not statistically significant (< 0.2). The false negatives were primarily a result of the few sample points obtained during the experiment, which were further reduced by our flow alignment method. This loss of information decreases the correlation of the server-to-exit and entry-to-victim client traffic.

5 Limitations

Our attack is very accurate in an in-lab set-up with symmetric network paths and capacities (having low congestion and no uncontrolled disturbances). However, in tests with public Tor relays, the overall correlation between server-to-exit and entry-to-victim traffic is decreased due to congestion and Tor's traffic scheduling, which distort the injected traffic pattern. In experiments involving data from the institutional Cisco router, such effects were quite pronounced. Moreover, the were fewer sample intervals compared to the data obtained from Linux NetFlow packages. This was due to flow aggregation, and lead to to flow records with unequal lengths, not evenly spaced. To counter such effects, we devised an approximation strategy, described in Section 3. Such approximations decrease the overall correlation of server-to-exit with entry-to-victim traffic, since the process eliminates data points from flow intervals that cannot be correctly rectified. This resulted in false positives in our measurements. Although not very precise, these results are indicative of the capabilities of more powerful adversaries. A powerful adversary could launch a *sybil* attack [9] by running many high-bandwidth Tor nodes to attract a large fraction of Tor traffic. Such relay operators, equipped with flow capture tools, would not require access to network routers for flow records.

6 Conclusion

We have demonstrated the practical feasibility of carrying out traffic analysis attacks using statistical correlation of traffic measurements obtained from NetFlow, a popular

network monitoring framework installed in various router platforms. Our work verifies the results of previous simulation results for traffic de-anonymization using NetFlow data [18]. We focused on practically evaluating such an attack to identify the source of anonymous traffic. We relied on correlation to identify the source of anonymous traffic amidst various flows. In a controlled lab environment, free from external network congestion, our attack was 100% accurate in identifying the targeted client. In experiments involving data from public Tor relays, our approach identified correctly the source of anonymous traffic in 81.6% of the cases, with a false positive rate of 5.5%. Currently, we are working on methods for defending against such attacks, using ideas related to selective dummy traffic transmissions schemes [19].

Acknowledgements: This material is based upon work supported by (while author Keromytis was serving at) the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Flow Tools Package. http://freecode.com/projects/flow-tools.
[2] I2P Anonymous Network. http://www.i2p2.de/. http://www.i2p2.de/.
[3] Java Anonymization Proxy. http://anon.inf.tu-dresden.de/. http://anon.inf.tu-dresden.de/.
[4] Netflow iptables module. http://sourceforge.net/projects/ipt-netflow/.
[5] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker. Low-resource routing attacks against tor. In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society (WPES)*, pages 11–20, 2007.
[6] S. Chakravarty, M. V. Barbera, G. Portokalidis, M. Polychronakis, and A. D.Keromytis. On the Effectiveness of Traffic Analysis Against Anonymity Networks Using Flow Records. Computer Science Department Technical Report (CUCS Tech Report) CUCS-019-13, Columbia University, July 2013.
[7] S. Chakravarty, A. Stavrou, and A. D. Keromytis. Traffic analysis against low-latency anonymity networks using available bandwidth estimation. In *Proceedings of the 15th European conference on Research in computer security*, ESORICS'10, pages 249–267, Berlin, Heidelberg, 2010. Springer-Verlag.
[8] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium)*, pages 303–319, August 2004.
[9] J. R. Douceur. The sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 251–260, London, UK, UK, 2002. Springer-Verlag.
[10] M. Edman and P. F. Syverson. AS-awareness in Tor path selection. In E. Al-Shaer, S. Jha, and A. D. Keromytis, editors, *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009*, pages 380–389. ACM, November 2009.
[11] N. Evans, R. Dingledine, and C. Grothoff. A Practical Congestion Attack on Tor Using Long Paths. In *Proceedings of the* 18^{th} *USENIX Security Symposium (USENIX Security)*, pages 33–50, August 2009.
[12] N. Feamster and R. Dingledine. Location Diversity in Anonymity Networks. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 66–76, October 2004.
[13] N. Hopper, E. Y. Vasserman, and E. Chan-Tin. How Much Anonymity does Network Latency Leak? In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, pages 82–91, October 2007.
[14] B. Hubert, T. Graf, G. Maxwell, R. Mook, M.Oosterhout, P.Schroeder, J. Spaans, and P. Larroy. Linux Advanced Routing and Traffic Control HOWTO.
[15] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson. Users get routed: Traffic correlation on tor by realisitic adversaries. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS 2013)*, November 2013.
[16] P. Mittal, A. Khurshid, J. Juen, M. Caesar, and N. Borisov. Stealthy traffic analysis of low-latency anonymous communication using throughput fingerprinting. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 215–226, New York, NY, USA, 2011. ACM.
[17] S. J. Murdoch and G. Danezis. Low-Cost Traffic Analysis of Tor. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 183–195, May 2005.
[18] S. J. Murdoch and P. Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In *In Privacy Enhancing Technologies (PET), LNCS*. Springer, 2007.
[19] V. Shmatikov and H. M. Wang. Timing analysis in low-latency mix networks: attacks and defenses. In *Proceedings of the 11th European conference on Research in Computer Security*, ESORICS'06, pages 18–33, Berlin, Heidelberg, 2006. Springer-Verlag.
[20] M. K. Wright, M. Adler, B. N. Levine, and C. Shields. An analysis of the degradation of anonymous protocols. In *Proceedings of the Network and Distributed Security Symposium (NDSS)*, 2002.
[21] X.Fu and Z.Ling. One cell is enough to break tor's anonymity. In *Proceedings of Black Hat Technical Security Conference*, pages 578–589, February 2009.