Large Scale Simulation of Tor: Modelling a Global Passive Adversary

Gavin O' Gorman and Stephen Blott

Dublin City University
Glasnevin, D9, Dublin, Ireland
{gogorman,sblott}@computing.dcu.ie

Abstract. Implementing global passive adversary attacks on currently deployed low latency anonymous networks is not feasible. This paper describes the implementation of a large scale, discrete event based simulation of Tor, using the SSFNet simulator. Several global passive adversary attacks are implemented on a simulated Tor network comprised of approximately 6000 nodes. The attacks prove to be highly accurate (80 percent stream correlation rate) for low traffic conditions but significantly less effective on denser, multiplexed links (18 percent success rate).

1 Introduction

The Internet, specifically TCP/IP, was not designed to provide anonymity. One solution to this problem is to create an overlay network, that is, a network which runs on top of an existing TCP/IP network. By abstracting away from IP addresses of hosts, the overlay network allows for the explicit control of the routing of its messages. This control enables the obfuscation of sender and receiver addresses, and thus offers a degree of anonymity.

An accurate, scalable testbed for implementing new features and measuring anonymity is required to test theorized attacks against these overlay networks. Establishing test networks in a lab is a limited option, such a test network could not scale to current and future deployment sizes. Creating a comprehensive analytical model is difficult given the level of complexity in current overlay networks.

One solution is to use discrete event-based simulation. This is the approach described in this paper.

– We describe a discrete event-based simulation of a popular low latency anonymising network, Tor, using the SSFNet simulator. Our simulation models the Tor routing of HTTP data with circuits, stream multiplexing, proxies, routers and exit routers.
– We have implemented several preliminary global passive adversary attacks using approximately 4500 HTTP clients, 100 HTTP servers and over 950 Tor routers.

Tor. Several anonymous network designs have been developed which attempt to apply mixes to low latency traffic. The most widely-used of these tools is Tor [1], the second generation onion router. The current Tor network contains approximately 900 router nodes with hundreds of thousands of streams transiting the network [2].

The Tor network consists of proxies, onion routers and exit routers. A user runs a Tor proxy on their local machine which offers a SOCKS interface to TCP applications. This Tor proxy begins the process of establishing a circuit through the Tor network of onion routers, to a suitable exit router and finally to the target TCP server. Circuits are established in a telescoping manner. On circuit establishment, the incoming TCP stream is routed over the circuit.

SSFNet. Discrete event-based simulation is used to create an abstract representation of the important elements of a system. Event based simulation allows one to build models of systems and investigate how the system might work under different conditions.

The Scalable Simulation Framework (SSF) [3], has been designed to model large scale simulations. The framework describes an interface for a simulation kernel. This generic kernel can then be built upon to implement varying simulators, of which network simulators are one type.

2 Our Implementation

Apart from some simulation specific techniques for ensuring a linear time execution of events, the simulation code itself is very similar to that of a real application. Three distinct network elements were created, on top of those already provided by the simulation libraries. These network elements are a proxy, a router and an exit router.

2.1 Protocol

The circuit establishment protocol described in the Tor design document [1] is simulated exactly. Encryption is not simulated; for our purposes, there is no need.

Traffic routed through the modeled Tor network is provided by a HTTP traffic generator, `SSF.OS.WWW` distributed with the SSFNet protocols. Further details of the traffic generator can be found in the appendix of [4].

Data received from the HTTP client is broken into 512-byte cells, labelled with the correct stream ID and sent to the router associated with that stream. Each router passes the data on, until the exit router receives it, recreates the original data, and sends this to the target server.

Several traffic streams may be multiplexed over circuit connections. For example, if the proxy receives a new incoming client connection and chooses as the first router on its path one to which a connection is already available, then that socket is reused. A new stream ID is allocated to the stream, the circuit establishment procedure is followed, and the stream is correctly routed.

2.2 Topology

The network topology used is taken from the SSFNet website. It is an simplification of a US ISP. The topology consists of 24 interconnected autonomous systems (AS), where each AS is composed of a number of sub networks. One proxy, two onion routers and one exit router were added to each of the subnetworks, evenly distributing the nodes throughout the network. This results in 325 proxies, 650 router nodes and 325 exit router nodes in the whole network, approximating the number of onion routers in the currently deployed Tor network. The number of clients per LAN was then set to 5, resulting in a total of 5760 clients.

3 Attacks and Results

We have implemented a number of attacks as discussed in detail below. The results from these attacks allow us to demonstrate the correctness of the Tor simulation, in that the results we observe are very similar to those previously published.

The attacks are performed with an increasing number of clients to model an increase in density of traffic across multiplexed connections. Also, with more traffic, there will be greater delay across the network.

The simulation is run for 1120 seconds. The initial 1000 seconds is to allow for the BGP and OSPF routing to settle. After this 1000 seconds, the HTTP clients begin connecting to the Tor network and to their target server. After 60 seconds, at time 1060, tcpdump output is recorded for another 60 seconds until time 1120, at which point the simulation terminates. The initial 60 seconds is to allow the Tor routers to settle into equilibrium.

3.1 Connection Start Tracking Attack

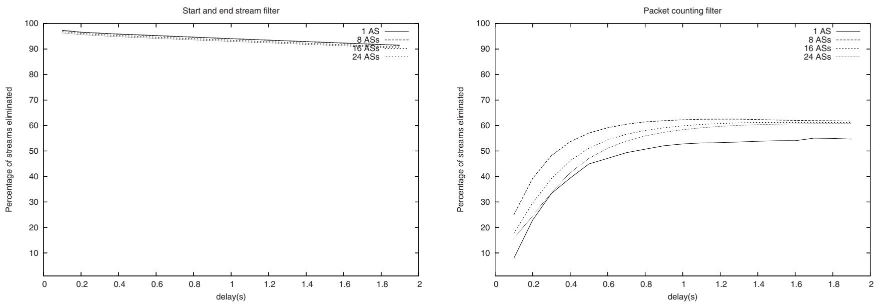
Connection start tracking was described by Serjantov et. al[5]. It works by tracking the initialization time of a connection as it spans the network. If a stream is seen to enter and then emerge from the network in a certain timeframe, it is possible to associate the two events. As demonstrated by Serjantov et al., the attack requires lone connections to successfully link streams. However, on a busy multiplexed network, connection start and end tracking serves as an effective filter to reduce the number of potential streams.

In our implementation of this attack, we take the time of the first HTTP response packet, add a variable delay (d) to it and compare this time with all recorded Tor streams. As the traffic is multiplexed, it is not possible to determine exactly when traffic streams start and end. Also with the extra delay introduced by more traffic, there is the need for a variable delay value. We perform the attack using values of d ranging from .1 to 2 seconds, increasing in increments of .1s, and with increasing numbers of clients as described above.

As seen in Fig. 1(a), the initial start and end tracking filter eliminates a high percentage of streams, up to 98% on the sparse traffic network and 96% on the densest network.

3.2 Packet Counting Attack

A packet counting attack introduced in [6] and further expounded in [5] consists of counting the number of packets entering a node and subsequently leaving a node for a given time interval. By comparing the number of packets for a particular stream entering a node with the number of packets leaving the node it can be possible to determine to which node/link the packets from that stream are being sent.



(a) (b)

Fig. 1. Start connection tracking and packet counting filters

The same method of varying the d value is used as above, but the streams being analyzed are those that have previously been filtered by the start and end time attack. Figure 1(b) shows that the 2% to 4% of streams left by the start and end time attack are further reduced by approximately 5% to 15% with the packet counting attack.

3.3 Stream Correlation Attack

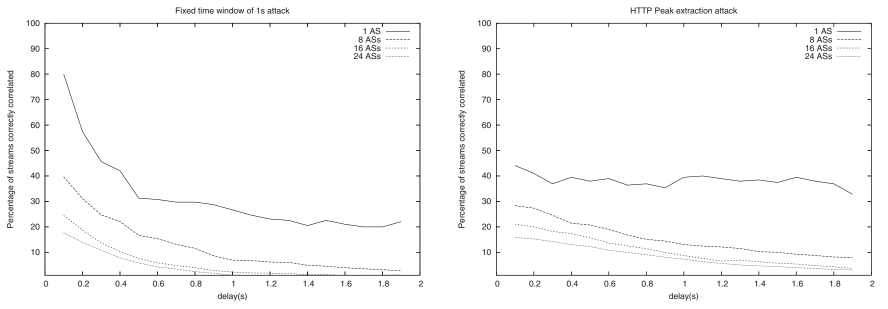
Fixed Time Window. Traffic correlation attacks were proposed and implemented in [7]. The technique is to set a windows size W and count the number of packets received, beginning at time t, during that window size. This process is repeated for the duration of the stream. The sequence of packet counts can then be compared with the sequences from other streams in the network. The cross correlation coefficient function used to compare these sequences is below [7]:

$$r(d) = \frac{\sum_i((x_i - \mu)(x'_{i+d} - \mu'))}{\sqrt{\sum_i(x_i - \mu)^2}\sqrt{\sum_i(x'_{i+d} - \mu')^2}}$$

The two streams being compared are x and x' with d being the delay value. x_i is the ith packet count of stream x and x'_i is the ith packet count of stream x'. μ is the average of packet counts in stream x and μ' is the average of packet counts in stream x'. The more the result tends towards 1, the greater the similarity of the streams. This same correlation function was also used in [8,9] for end to end traffic confirmation.

Some minor modifications to the attack are needed. The first is that the Tor protocol breaks the HTTP data into 512 byte cells. As such, the number of packets being sent from the HTTP server is not the same as the number of packets received at the Tor proxy. Allowing for this is straightforward.

For the fixed interval attack, a time window of 1s is used, as recommended by Shmatikov & Wang [8]. The fixed time interval attack is highly effective, Fig, 2, with approximately 80% of streams correctly identified on the low traffic network. Most connections were lone and so easily correlated. On the denser networks, the attack proved to be less effective with the extra noise of the multiplexed traffic. As the delay was increased, the accuracy of the attack fell rapidly. With the denser network, the most accurate attacks are still at .1s delay, demonstrating that network congestion did not prove to be an issue as anticipated. This is most likely as a result of high bandwidth of clients and servers. Realistic bandwidth values and increased traffic should demonstrate the effect of congestion on the network.



(a) (b)

Fig. 2. Success rate of correlation attacks

Peak Extraction. An alternative to using a fixed time interval is to break each stream into fixed fractions, count the number of packets observed in these fractions and correlate them with the function above. The values for these fractions can be determined by examining the HTTP stream. HTTP traffic is bursty. A certain web page will contain a number of objects, each of which is downloaded individually and can be observed as a burst of traffic, or a peak, across the connection. We obtain ratios for each peak termination. These ratios can then be applied to the Tor stream. Allowing for the delay d the corresponding packet counts should be equal.

The results of the peak extraction attack are presented in Fig. 2(b). The attack is not as effective as the fixed window of 1s attack, however it is slightly more robust when inaccurate delay values are used. The percentage of streams successfully identified decreases at a lower rate than the fixed interval attack. The success rate surpasses the fixed interval attack for larger delay values.

4 Related Work

Initial analytical work [10], using traffic matrices, provided metrics for measuring the effort required to thwart stream correlation attacks. This work was extended, using entropy to measure anonymity [11]. Real traffic measurements are taken from a campus network, however no attacks are described.

Later work by Levine et al. [7] describes global passive adversary attacks for stream correlation. The technique used is described in the results section. The Levine et al. attacks do not account for the multiplexing of traffic streams. The cross correlation coefficient was later utilized by Bissias et al. [9] to correlate encrypted HTTP streams.

Shmatikov & Wang [8] extend the original attack of Levine et al. by proposing and testing a new defense. This defense, adaptive padding, involves applying padding to ensure that streams are indistinguishable from each other. As with Levine et al. the attacks are performed on links with non-multiplexed streams. Also, as is pointed out in the paper, stream times are also not taken into account, whereas in our attacks, we filter results by checking start and end time.

Zhu et al. [12] use mutual information and frequency analysis (wavelets/FFT) to correlate TCP traffic streams. In ongoing experimental work, we have applied frequency analysis to HTTP streams generated by the simulation. As yet, the accuracy of the method appears quite low as the bursty nature of HTTP traffic does not lend itself to frequency analysis.

In terms of scale, Bauer et al. [13] have implemented perhaps the most ambitious attacks to date, using approximately 60 Tor nodes distributed across the globe in a test network. Our simulation in contrast utilise approximately 6000 nodes.

5 Future Work

Our initial work on TCP stream analysis, using wavelets and Fourier transforms shows promise. We intend developing these attacks in conjunction with more realistic topologies. Additional work to verify the fidelity of the simulation to the real Tor client will consist of packet count and timing analysis on small scale networks.

Beyond that, we intend measuring the average delay for streams crossing the network and introducing delay to the Tor nodes. We can measure the impact this has on the network in terms of Quality of Service and effectiveness against stream correlation. The overall goal is to determine the optimal compromise between latency and anonymity for anonymous networks.

6 Conclusion

We have developed an initial Tor simulation. We have begun verification the simulation with the implementation of previously discussed attacks and obtained expected results. The simulation needs to be extended to replicate the traffic control techniques Tor utilises. Given that, we will be able to reliably measure quality of service across the network. This, in combination with the attacks presented will allow us to quantify the compromise between latency and annonymity for a given network configuration.

The ability to test and implement new features on the Tor simulation will, we believe, prove to be invaluable for the Tor developers and future researchers.

References

1. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (August 2004)
2. Developers, T.: Tor website (June 2007), `http://tor.eff.org`
3. Cowie, J., Liu, H.: Towards realistic million-node internet simulations. In: Proceedings of the 1999 International Conference on Parallel and Distributed Processing Techniques and Applications (1999)
4. Feldmann, A., Gilbert, A., Huang, P., Willinger, W.: Dynamics of IP traffic: A study of the role of variability and the impact of control. In: Proceedings of ACM SIGCOMM 1999, pp. 301–313. ACM Press, New York (1999)
5. Serjantov, A., Sewell, P.: Passive attack analysis for connection-based anonymity systems. In: Snekkenes, E., Gollmann, D. (eds.) ESORICS 2003. LNCS, vol. 2808, Springer, Heidelberg (2003)
6. Back, A., Möller, U., Stiglic, A.: Traffic analysis attacks and trade-offs in anonymity providing systems. In: Moskowitz, I.S. (ed.) Proceedings of Information Hiding Workshop (IH 2001). LNCS, vol. 2137, pp. 245–257. Springer, Heidelberg (2001)
7. Levine, B.N., Reiter, M.K., Wang, C., Wright, M.K.: Timing attacks in low-latency mix-based systems. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, Springer, Heidelberg (2004)
8. Shmatikov, V., Wang, M.-H.: Timing analysis in low-latency mix networks: Attacks and defenses. In: Gollmann, D., Meier, J., Sabelfeld, A. (eds.) ESORICS 2006. LNCS, vol. 4189, Springer, Heidelberg (2006)
9. Bissias, G.D., Liberatore, M., Levine, B.N.: Privacy vulnerabilities in encrypted http streams. In: Danezis, G., Martin, D. (eds.) PET 2005. LNCS, vol. 3856, Springer, Heidelberg (2006)
10. Venkatraman, B., Newman-Wolfe, R.: Performance analysis of a method for high level prevention of traffic analysis using measurements from a campus network. Computer Security Applications Conference, Proceedings, 10th Annual (1994) 288–297 (1994)
11. Newman, R.E., Moskowitz, I.S., Syverson, P., Serjantov, A.: Metrics for traffic analysis prevention. In: Dingledine, R. (ed.) PET 2003. LNCS, vol. 2760, Springer, Heidelberg (2003)
12. Zhu, Y., Fu, X., Graham, B., Bettati, R., Zhao, W.: On flow correlation attacks and countermeasures in mix networks. In: Martin, D., Serjantov, A. (eds.) PET 2004. LNCS, vol. 3424, Springer, Heidelberg (2005)
13. Bauer, K., McCoy, D., Grunwald, D., Kohno, T., Sicker, D.: Low-Resource Routing Attacks Against Anonymous Systems. Technical Report CU-CS-1025-07, University of Colorado at Boulder (2007)