

Improving stream correlation attacks on anonymous networks

Gavin O’Gorman
Dublin City University
Glasnevin, D9
Dublin, Ireland

gavin.ogorman@computing.dcu.ie

Stephen Blott
Dublin City University
Glasnevin, D9
Dublin, Ireland

stephen.blott@computing.dcu.ie

ABSTRACT

The level of anonymity offered by low latency, interactive, anonymous networks is unknown. This paper implements correlation attacks on the deployed Tor network and a simulated Tor network under defined network conditions. The accuracy of the attacks act as a metric for the networks anonymity in the face of a passive adversary. From observation of the deployed Tor network, several techniques were developed to compensate for some of the modifications the Tor protocol induces in traffic. These techniques increase correlation accuracy by 10% to 40% for differing correlation functions. Almost 50% of traffic streams on the simulated network are identified immediately with 10% of experimental traffic on the real Tor network identified.

Categories and Subject Descriptors

C.2.2 [Computer-communication networks]: Network Protocols—*Applications*; K.6.5 [Security and Protection]

General Terms

Security, Anonymity, Simulation, Latency

1. INTRODUCTION

Remaining anonymous on the Internet is a difficult task. Current solutions to this problem are based on the Mix [4] concept, that of routing traffic via a number of intermediary nodes. These intermediary nodes obfuscate the traffic route, thus increasing the difficulty of an observer attempting to trace the traffic stream. One solution for an attacker is to not trace the traffic route, but rather treat the network as a black box and observe traffic entering and exiting it. Statistical similarities in the traffic may allow the attacker to correctly determine with whom one is communicating using correlation functions.

Such correlation attacks require an attacker to compare their target stream, observed entering the network, with every stream exiting the network. The correlation function

used to compare the streams returns a value describing the similarity of two streams. The attacker then judges the stream with the highest degree of similarity to be the same stream. As this is the technique an attacker can deploy to compromise anonymity, by implementing it and measuring the accuracy it allows one to measure the level of anonymity offered by the network.

A number of factors affect a stream’s anonymity. If there are two or more independent, identical streams transiting the network simultaneously, the attacker has no way of distinguishing between the two. If the anonymous network is heavily congested, packets may be intermittently delayed or even re-sent, no longer having the same interarrival times. Finally, the anonymous protocol itself may, either purposefully or as a side-effect, modify a stream such that it becomes unidentifiable.

By implementing a given anonymous network, controlling the traffic distribution and network conditions, it is possible to measure the accuracy of the correlation functions. This provides a means of measuring the anonymity offered by the anonymous network under these conditions. It is also possible to take into account the modifications to streams the anonymous network protocol may induce, thus giving a more accurate measure of stream anonymity.

This paper describes such a system. A simulation of the Tor[6] anonymous network was created. Three correlation functions were tested, modified packet counting, the cross correlation co-efficient and mutual information. Two techniques were developed to compensate for modifications the Tor protocol induces in traffic. Several thousand traffic streams were sent through the simulated network, correlated and the subsequent anonymity measured. To validate the simulation, traffic from the same HTTP generator was sent through the real Tor network, the anonymity measured and compared with the simulation results. The compensation techniques developed were applied and the subsequent drop in anonymity measured.

An overview of anonymous networks and correlation techniques is presented in Section 2. A description of the test environment and system used is discussed in Section 3. Section 4 describes the compensation techniques developed. Results are presented in Section 5, related work in Section 6 and Section 7 concludes.

2. BACKGROUND

2.1 Anonymous networks

Anonymous networks were initially proposed by Chaum

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC’09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.
Copyright 2009 ACM 978-1-60558-166-8/09/03 ...\$5.00.

[4] in the early 80s and dealt with high latency, message based communications. These networks operate on the concept of message routers, called mixes, that delay and batch messages. The introduction of delays and batching prevents timing and size analysis on incoming and outgoing messages. However, when applied to low-latency, interactive, communications, such batching and delaying techniques can introduce debilitating latency. For example, Voice over IP applications cannot tolerate a latency much larger than about 150ms between start and endpoints. Several anonymous network designs have been developed which attempt to solve this problem, Tor[6] being the most widely deployed.

2.1.1 Tor

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. A connection request is sent to the first onion router. When this connection has been established, the request to connect to the next router is sent via the already established connection. This repeats for each node on the route. Once the circuit is established, the incoming TCP stream is routed over the circuit. The Tor proxy handles receiving TCP packets from the client, extracting the payload, breaking it into 512-byte cells and forwarding these onto the next Tor router. Each router forwards the cells onto the next router. An exit router receives the cells, recreates the initial payload and sends it onto the server.

2.2 Correlation techniques

In the attack scenario measured in this paper, the Tor network is treated as a black box. Traffic entering and exiting the network is recorded. Given two traffic streams, one entering and one exiting the network, their similarity is measured using a correlation function. Several correlation techniques as used in the literature are described here.

2.2.1 Packet Counting

Packet counting [1, 9] is a simple technique whereby the number of packets in a traffic stream are counted. Other streams with a similar number of packets are potentially the same stream. An alternate to this is to count the amount of bytes, rather than packets. This proves to be more effective, as the number of packets can change due to varying network MTU values.

Byte counts will vary between same streams entering and exiting the network. This can be as a result of streams not being fully recorded, overhead anonymous network management packets and padding inserted by the anonymous network. As such, simple equality checks are inaccurate. Instead, to measure the similarity between byte count x from stream X and count y from stream Y , use a distance measure.

$$\sqrt{(x - y)^2} \quad (1)$$

The smaller the difference, the greater the similarity of the streams.

2.2.2 Correlation Co-efficient

The cross-correlation co-efficient [7, 3], measures the average slope obtained from plotting three or more (x, y) points on a graph. For each pair, if both the x and y components are the same value, the angle of the line is 45° , or a slope of 1. As the values become random and there is no correlation, the slope approaches 0. A slope of -45° indicates the pairwise values are the inverse of each other.

To apply this correlation technique to streams of network traffic one must extract a sequence of values from the streams. The technique used here is to set a windows size W and count the number of packets received 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.

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

The two streams being compared are x and x' with delay value d . This delay value is the time required for the stream to transit the network. x_i is the i th packet count of stream x and x'_i is the i th 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' .

2.2.3 Mutual Information

Mutual Information [13] is formally defined as:

$$I(X; Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \quad (3)$$

where X and Y are two random variables. It is a measure of the similarity of the two variables based on the difference between the actual joint distribution of X and Y and the joint distribution of X and Y if X and Y were independent. $p(x, y)$ is the value from the joint probability distribution for x and y . $p(x)$ and $p(y)$ are the marginal probability distributions for x and y . \log_2 measures in bits the difference. This bit measure is then weighted. The weighting variable in the above equation is the joint distribution value.

The X and Y sequences are generated using the same window technique as in the Pearson product-moment correlation coefficient function.

3. SYSTEM

Performing experimental black box attacks against the currently deployed Tor network is relatively straightforward. A client and server can be monitored. Streams are recorded as they enter and exit the network. These streams are then correlated, and the accuracy of this correlation gives a metric for anonymity. However, performing attacks on the deployed Tor network is time consuming. Controlling Tor network conditions such as latency, traffic rates, numbers of clients, routers, exit routers and servers is not possible. A compromise between the accuracy of attacks on the real network versus the time cost and lack of control is to use a simulation. Comparing results from the simulation against results obtained from the real network allow for a validation of the simulation. This section describes the simulation implemented, the system used to obtain results from the deployed Tor network and the anonymity measure used.

3.1 Simulation implementation

The Tor simulation is written using the SSFNet simulator. Apart from some simulation specific techniques for ensuring a linear time execution of events, the 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.

The circuit establishment protocol described in the Tor design document [6] is simulated exactly. Encryption however, is not simulated as it serves no purpose within a simulation.

Traffic routed through the modeled Tor network is provided by a HTTP traffic generator, `SSF.OS.WWW` distributed with the SSFNet protocols.

On the real Tor network, cells are not transmitted directly. They are instead transmitted over a TLS connection between network elements. This in turn modifies the stream as TLS effectively buffers packets before transmission. The simulation does not implement TLS encryption, however it does approximate the batching effect that TLS creates as well as overhead traffic.

3.2 Real Tor network

In order to evaluate the accuracy of results produced by the simulation, 10,000 streams of traffic were routed through the deployed Tor network over the Internet. The traffic was created using the same HTTP generator as the simulation, with a combination of `tcpdump`, `wget` and the Apache web server. Guard nodes were disabled. A single client connected to a single HTTP server via the Tor network, requesting each stream consecutively. Streams were extracted from the `tcpdump` output and correlated using the correlation function described. Results are presented in Section 5.

3.3 Measuring Anonymity

The measure in this paper is based on the K-anonymity [11] measure. For a given stream, its K-anonymity value describes the number of streams of either greater or the same level of similarity. An attacker then has a probability of less than or equal to $\frac{1}{K}$ of identifying the stream correctly.

To calculate the K-anonymity for a stream the results for each correlation method are ranked from highest (most similar) to lowest (least similar). The correlation value for the correct stream is also recorded. By counting the number of streams with a larger, or same, correlation value as the actual correlation value, we can obtain a K-anonymity measure. For example, with 5 traffic streams, A to E, and one correlation method, there are 25 comparison values, each stream compared with itself and each other stream. Taking stream A, the 5 comparison values might be:

A	B	C	D	E
1	.9	.2	1	.4

The results are ranked. In this instance, stream A has a K-anonymity value of 2, meaning an attacker has a 50% chance of guessing the correct stream.

3.4 Validation

Figure 1 shows the results from correlating traffic on the real Tor network. Figure 2 is the simulated network. Mutual

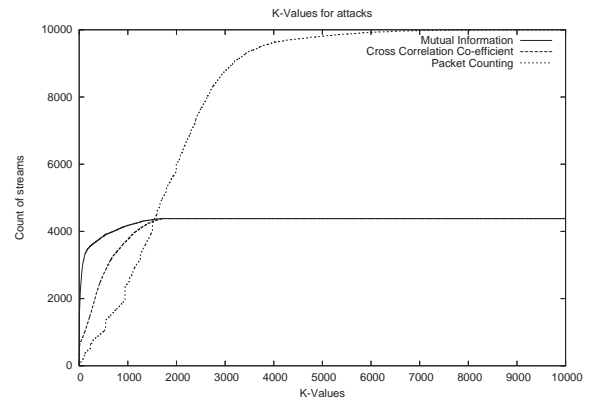


Figure 1: Real Tor network

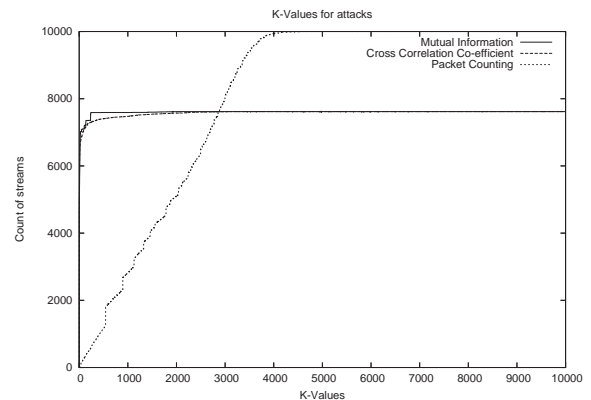


Figure 2: Simulated Tor network

Information correlation on the simulated traffic is approximately four times more accurate than real network traffic. Cross-correlation is almost almost 20 times more accurate. The greater accuracy in the simulation is because of the homogeneous nature of nodes and traffic. The simulation enters a state of equilibrium, whereas the real Tor traffic fluctuates a great deal in terms of delay. Additionally, there is more noise on the Tor network, command packets from directory servers for example. These affect the byte counting attack. The results from the simulation in comparison to the real network are reasonable. Given the homogenous nature of the simulation and lack of noise, the correlation attacks will be more accurate than when used on the real Tor network.

4. NETWORK COMPENSATION

As a stream of packets transits any reasonably sized network the stream is modified. Congestion on routers may cause packets to be delayed or even lost, thus requiring retransmission. These delays change the inter-arrival times of stream packets. Predicting such modifications without constantly measuring the congestion at each router is extremely difficult. However, in the case of an anonymous network, certain modifications to a stream can be predicted. Two modifications and corresponding compensatory techniques are described below. By applying these techniques to a HTTP

stream entering the network, it allows one to predict the shape of the stream exiting the Tor network. Comparing the predicted stream with the actual Tor output gives more accurate results.

Traffic coming from the HTTP server is described as a sequence of tuples, $TS = ((t_1, s_1), \dots, (t_n, s_n))$ where t is the time of the packet arrival and s is the size of the packet.

4.1 Cell conversion

The sequence of tuples from the HTTP generator are sent as TCP packets. These packets are then converted to homogeneous cells in the anonymous network which is designed to mitigate simple traffic size correlation attacks. The compensation function below calculates the number of cells created, depending on the cell size of the anonymous network and overhead traffic. From observation of the Tor network the first cell contains a payload of 470 bytes and subsequent cells 500 bytes.

The number of cells, nc for an object of size o , with a cell size of cs can be calculated as with the function *cellcount()*:

Algorithm 1 Determine number of cells, nc from object of size o

```

if  $o > 470$  then
     $nc ++$ 
     $o = o - 470$ 
end if
while  $o > 500$  do
     $nc ++$ 
     $o = o - 500$ 
end while
if  $o > 0$  then
     $nc ++$ 
end if

```

For each object, the data is converted to cells using the *cellcount()* function, given a TLS header of 74 bytes and then sent out as a batch message, each packet size determined by the Maximum Segment Size (MSS).

The overall effect is that despite cell conversion, the entry and exit streams are very similar, taking into account overhead traffic from cell padding and TLS header.

4.2 Inter-arrival times

The cells on an anonymous network are routed over a number of routers. Each router has a queue of cells to be sent, and a network delay between links. This combination of queues and network delay modify packet inter-arrival times. For those routers with a large amount of traffic, the queues can backup, resulting in cells being delayed until the queue ahead is transmitted. This can lead to large bursts of data being ‘stretched’ over a period of time.

There are two elements which can be accounted for. These are the delay across the network and the delay resulting from queuing of packets. To determine the network delay value when comparing a HTTP stream with its potential equivalent Tor stream, the delay between packets on the Tor stream is measured. Taking the average of these delays gives an approximation of the delay across the network. This is the $r(t)$ value used below.

To calculate the delay each packet suffers as a result of being queued, the following algorithm is used. From the packet counting conversion, there is a sequence of times for

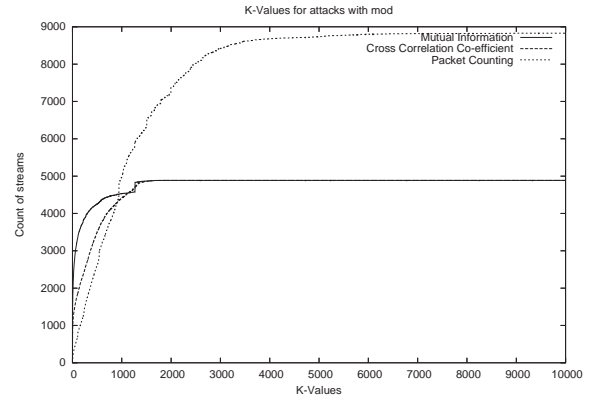


Figure 3: Real Tor network with compensation

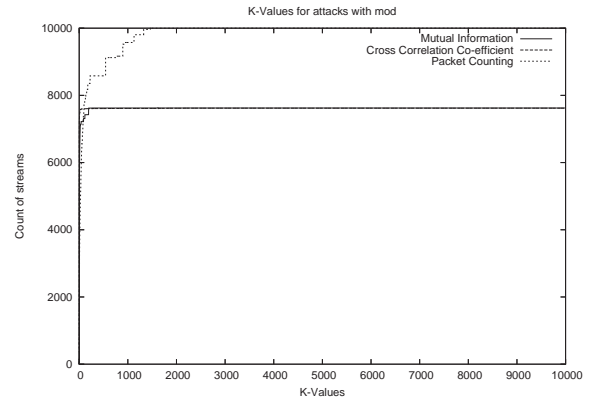


Figure 4: Simulated Tor network with compensation

each stream, $T = (t_1, t_2, \dots, t_n)$

The algorithm used to modify T is:

$$\forall i(i < n)t_i = \begin{cases} t_{i-1} + r(t) & \text{if } t_i \leq t_{i-1} \\ t_i + r(t) & \text{otherwise} \end{cases}$$

This modified sequence of times should approximate the traffic seen emerging from the anonymous network. The effectiveness of this modification is shown in the results.

5. RESULTS

The simulation is run for 60 seconds (after an initial period of time to allow for equilibrium). Approximately 10,000 streams of traffic are generated. These streams are extracted from the tcpdump data. Each incoming HTTP stream is then compared with every other outgoing Tor stream using correlation functions as described. The streams are compared with and without the modifications applied.

5.1 Applying compensation techniques

Figures 3 and 4 show results of the compensation techniques applied to the real Tor network and simulation network, respectively. The accuracy of the attacks increases by approximately 10% for mutual information and approximately 40% for cross correlation on the real network at $K = 1$. The byte counting attack also shows an improve-

ment, from less than 5 streams being identified at $K = 1$ to over 25.

On the simulated network, the increase in accuracy is similar with the same percentage improvements in mutual information and cross correlation. However, byte counting is significantly more accurate, going from less than 5 streams at $K = 1$ to over 1400 streams at $K = 1$. The increase is again because of the lack of noise on the simulated network. Extra packets do not affect the mutual information or cross correlation attacks significantly, however the amount of extra bytes in these packets does affect the byte counting attack.

6. RELATED WORK

Initial analytical work [12], using traffic matrices, provided metrics for measuring the effort required to thwart stream correlation attacks. This work was extended, using entropy to measure anonymity [8]. Real traffic measurements are taken from a campus network, however no attacks are implemented.

Later work by Levine et al. [7] describes global passive adversary attacks for stream correlation. The technique used is as described in this paper. The cross correlation coefficient was later utilized by Bissias et al. [3] to correlate encrypted HTTP streams.

Shmatikov & Wang [10] extend the 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.

Daginawala and Wright [5] use Internet distributed nodes to evaluate the attack and implement a new attack based on the largest observed values in inter-packet arrival times.

Zhu et al. [13] use mutual information and frequency analysis (wavelets/FFT) to correlate TCP traffic streams. The traffic used is generic TCP and FTP connections however, no anonymous network protocol is used.

In terms of scale, Daginawala and Wright [5] used 300 clients distributed across the PlanetLab network. Bauer et al. [2] have used approximately 60 nodes on the same network to test route compromise probabilities.

7. CONCLUSIONS AND FUTURE WORK

Tor was not designed to prevent black box attacks implemented by a global passive adversary. It is generally accepted that protecting against such an attack requires impractical resources. The results in this paper confirm the speculation that such attacks are feasible and offer a reasonable degree of accuracy. Some of the obfuscation introduced by the network and the anonymous network protocol can be compensated for to increase correlation accuracy. Even a trivial attack such as byte counting can be quite accurate, given the correct conditions. However in the experiments in this paper, the most accurate attacks in ideal conditions of the simulation still did not identify approximately 23% of the streams. Even a small amount of extra noise on the network considerably increased the difficulty of the attacks, particularly in the case of byte counting. The judicious addition of extra noise on the network in the form of padding and delay will therefore increase the difficulty of an attacker. Measuring the cost of such noise to the network as a whole

and the individual user is now practical with the simulation developed. Such measurements is part of future work and should be useful for anonymous network designers.

8. REFERENCES

- [1] A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In I. S. Moskowitz, editor, *Proceedings of Information Hiding Workshop (IH 2001)*, pages 245–257. Springer-Verlag, LNCS 2137, April 2001.
- [2] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker. Low-Resource Routing Attacks Against Anonymous Systems. *Technical Report CU-CS-1025-07, University of Colorado at Boulder (2007)*.
- [3] G. D. Bissias, M. Liberatore, and B. N. Levine. Privacy vulnerabilities in encrypted http streams. In *Proceedings of Privacy Enhancing Technologies workshop (PET 2005)*, May 2005.
- [4] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.
- [5] H. Daginawala and M. Wright. Studying Timing Analysis on the Internet with SubRosa. *Lecture Notes in Computer Science*, 5134:133–150, 2008.
- [6] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.
- [7] B. N. Levine, M. K. Reiter, C. Wang, and M. K. Wright. Timing attacks in low-latency mix-based systems. In A. Juels, editor, *Proceedings of Financial Cryptography (FC '04)*. Springer-Verlag, LNCS 3110, February 2004.
- [8] R. E. Newman, I. S. Moskowitz, P. Syverson, and A. Serjantov. Metrics for traffic analysis prevention. In R. Dingledine, editor, *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*. Springer-Verlag, LNCS 2760, March 2003.
- [9] A. Serjantov and P. Sewell. Passive attack analysis for connection-based anonymity systems. In *Proceedings of ESORICS 2003*, October 2003.
- [10] V. Shmatikov and M.-H. Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *Computer Security ESORICS 2006*, 2006.
- [11] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
- [12] B. Venkatraman and R. Newman-Wolfe. Performance analysis of a method for high level prevention of traffic analysis using measurements from a campus network. *Computer Security Applications Conference, 1994. Proceedings., 10th Annual*, pages 288–297, 1994.
- [13] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao. On flow correlation attacks and countermeasures in mix networks. In *Proceedings of Privacy Enhancing Technologies workshop (PET 2004)*, volume 3424 of LNCS, May 2004.