Toward a More Meaningful Neutral Traffic Matrix

Joseph Choi

CIS 6930 Course Term Paper
Final Draft
December 11, 2015

Abstract

The Neutral Traffic Matrix (TM) approach uses dummy-message padding, rerouting, and delaying of network traffic to prevent traffic analysis by a Global Passive Adversary. The proposed study is an attempt to identify and test new means of realizing a neutral TM, one in which all messages are meaningful and contribute to network activity. Three alternatives are proposed: (1) splitting of messages, (2) repackaging of messages according to original size, and (3) periodic control message exchange. Of the three, the repackaging option seems to be the most promising. The proposed alternatives are expected to go one step beyond hiding traffic information from an adversary, as they enable better utilization of a network through optimized message exchange..

1 Introduction

The Neutral Traffic Matrix approach is an attempt at smoothing the traffic information available to a network adversary. The adversary in question is a Global Passive Adversary, who is able to see (1) the source and destination of all messages, and (2) the number of messages crossing each link. Using this observed information, the adversary pieces together a traffic matrix in order to model network activity, hoping to extract some new knowledge of the network dynamics.

1.1 The Traffic Matrix

In a traffic matrix, each row corresponds to a sender, and each column corresponds to a receiver. An entry (i, j) in the traffic matrix at the intersection of a particular sender i and a particular receiver j gives the number of messages sent from i to j. Both sender and receiver roles are assigned to each node in the network, meaning the traffic matrix takes a square shape. Thus, if there are n nodes in the network, the matrix dimensions are necessarily n * n. Figure 1 presents an illustration of the traffic matrix format.

The network topology this study deals with is the complete graph. As such, all nodes in the network are connected to all other nodes. A node will not generate nor receive self-traffic, so matrix entries along the top-left to bottom-right diagonal will be zeroes. That is, for any given node i, the entry (i, i) = 0. The entries at the intersection of any other sender-receiver pair may occasionally take zero values as well, if no messages were transferred over the respective link during the measured window. However, in the average case, all nodes are expected to exchange some traffic with all other nodes, resulting in nonzero entry values.

		Receiver				
		Node 1	Node 2	Node 3	...	Node k
Sender	Node 1	(1, 1)	(1, 2)	(1, 3)	...	(1, k)
	Node 2	(2, 1)	(2, 2)	(2, 3)	...	(2, k)
	Node 3	(3, 1)	(3, 2)	(3, 3)	...	(3, k)
	...	...	...	...	...	...
	Node k	(k, 1)	(k, 2)	(k, 3)	...	(k, k)

Figure 1: Traffic Matrix Format

1.2 Unmodified Traffic

If network traffic is not altered in any way, observed traffic matrixes leak a lot of information. Though messages would be encrypted, an adversary would easily be able to associate every message with its sender and receiver. The volume of messages exchanged over a link might suggest the identities of the communicating parties. An attacker is also capable of collecting traffic matrix observations over time in order to gain an understanding of normal operating behavior, any departure from which might be of interest. So as to prevent a Global Passive Adversary from mapping network activity with absolute certainty, traffic should be altered.

1.3 Modified Traffic

Previous work by Newman and Venkatraman on traffic analysis prevention via the Neutral Traffic Matrix approach proposed (a) the rerouting of messages, (b) the addition of dummy messages, and (c) the delaying of messages [1]. If nodes are able to reroute messages through another intermediate node, an attacker can no longer claim with absolute confidence the initial sender or ultimate receiver of any particular message. If nodes are able to send out dummy messages, it becomes possible to make less active nodes appear just as involved in network traffic as more active nodes. If nodes are able to introduce delay, larger groups of messages can be sent in appropriate-sized chunks, avoiding sudden spikes in message flow that would raise adversarial suspicion.

2 Perceived Problem

Message rerouting promotes even distribution of link loads, but some links must be revisited afterwards for padding via dummy messages. The addition of dummy messages, however, only ever increases load cost on a link, and thus on the network. Aside from its inflating of traffic, a dummy message is not very useful to the underlying network communication. No meaning is to be derived from it, and no message is reliant on it for purposes of delivery. Furthermore, dummy

messages must be artificially created and sufficiently random to prevent their nature from being exposed.

Is there some other method of altering traffic that would better utilize the real messages being exchanged? An alternative method should (a) not inconvenience a node with the fabrication of throwaway messages, (b) not perform any worse than dummy message insertion, and (c) sufficiently distance an observed traffic matrix from the actual traffic matrix.

3 Initially Proposed Alternative: Simple Splitting

Let us consider the splitting and recombination of message parts, appropriately rerouting these parts separately in order to confuse the adversary. Real messages are being split to form the new messages for this purpose, as opposed to being generated completely from scratch like a dummy message would be. This alternative is also expected to provide more flexibility when rerouting.

3.1 Assumptions

The initial assumptions are as follows. Messages exchanged across the network are initially padded out to a length matching that of the longest message. This is done to make messages indistinguishable from one another. If the encryption scheme produces the same output for each instance of the same message, the same message should not be re-sent. Otherwise, the no-duplicate restraint may be safely ignored.

3.2 Methodology

Two splitting transform schemes are proposed. For the sake of simplicity, messages are only split into two halves. It is not immediately clear whether messages split into more than two pieces would provide any added benefit; that is left for future work.

To isolate the benefit of the splitting scheme, rerouting of full messages is not considered. As no rerouting of full messages is being done, the only way to construct a Neutral Traffic Matrix is by reaching the maximum message count on all links.

3.2.1 Splitting Transform Scheme 1

Let there be two nodes, A and B, where A wishes to send a message m to B. Node A splits message m into two halves: m_1 and m_2. Each of these halves is padded to reach full message length, meaning each half of the split message behaves like a full message. Figure 2 illustrates how this is done.

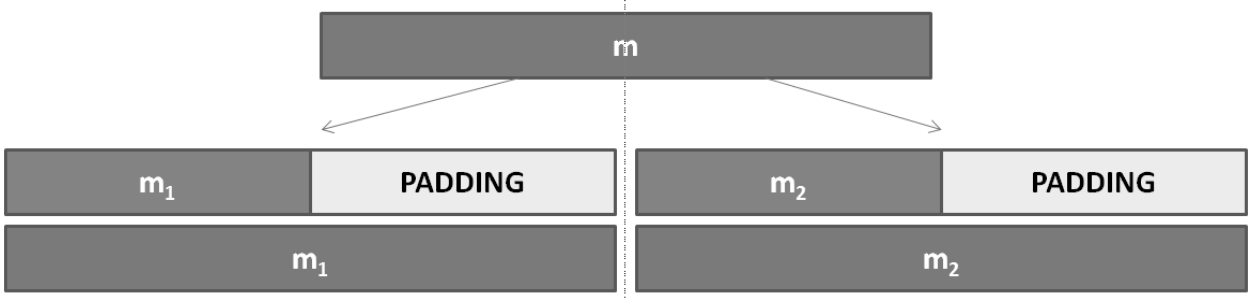

Figure 2: Splitting by Scheme 1

Let's take a look at Splitting Transform Scheme 1 in action. A simple example is provided in Figure 3, which looks at a network of three nodes.

	Node 1	Node 2	Node 3
Node 1	0	msg a, msg b	msg c, msg d
Node 2	msg i, msg j, msg k	0	msg e
Node 3	msg g, msg h	msg f	0

	Node 1	Node 2	Node 3
Node 1	0	msg a.1, msg b, msg c.2	msg c.1, msg d, msg a.2
Node 2	msg i, msg j, msg k	0	msg e, msg c.2
Node 3	msg g, msg h	msg f, msg a.2	0

Figure 3: Splitting by Scheme 1 in Action

In Figure 3, message a being sent Node 1 → 2 is split into a.1 and a.2, with a.2 rerouted through node 3. At the same time, message c being sent Node 1 → 3 is split into c.1 and c.2, with c.2 rerouted through node 2. Now, half of the links are at the maximum of 3 messages, whereas only one link was at the maximum initially. We might also simply split messages without rerouting their parts: splitting messages e, f, and g into two messages each would put their respective links at 3 messages apiece, fulfilling matrix neutrality.

3.2.2 Splitting Transform Scheme 2

Let there be two nodes, A and B, where A wishes to send a message m to B. Node A splits message m into two halves: m_1 and m_2. Node A also splits another message n into two halves: n_1 and n_2. Message n may or may not be destined for B. One of message m's halves would then be exchanged with one of message n's halves. Figure 3 below illustrates how this might be done.

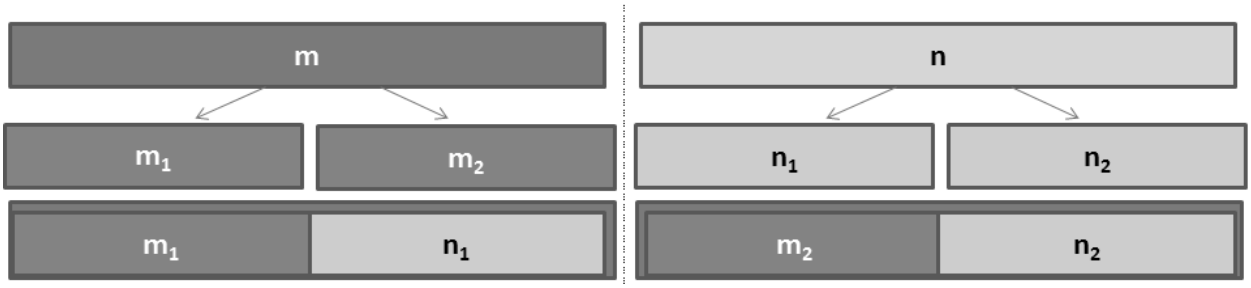

Figure 4: Splitting by Scheme 3

Let's take a look at Splitting Transform Scheme 2 in action. A simple example is provided in Figure 5, which looks at a network of three nodes.

	Node 1	Node 2	Node 3
Node 1	0	msg a, msg b	msg c, msg d
Node 2	msg i, msg j, msg k	0	msg e
Node 3	msg g, msg h	msg f	0

	Node 1	Node 2	Node 3
Node 1	0	msg a, msg b	msg c, msg d, msg i/(?)
Node 2	msg i/j msg k	0	msg e, msg j/i
Node 3	msg g, msg h msg j/(?)	msg f	0

Figure 5: Splitting by Scheme 2 in Action

In Figure 5, Node 2 wishes to send message i to Node 3 and message j to node 1. These two messages are split and their halves interchanged, producing message i/j and j/i; one is directed to node 1 and the other to node 3. Node 1 must now find a way to get the first half of message i to Node 3, whereas Node 3 must then find a way to get the first half of message j to Node 1. Unfortunately, at this point we get stuck, since both Node 1 and Node 3 now have two-and-a-half messages to send to one another, and Splitting Scheme 2 does not allow the standalone sending of half-messages. The only way to get unstuck is to pad the messages at this point in the communication, but doing so results in a total change of +2 in the message transfer count.

3.3 Evaluation

Both splitting schemes require an optimal order for splitting of messages to be chosen, which is no easy problem to be solved. Furthermore, less flexibility is provided, since each part must ultimately be received by its destination. Dummy messages, on the other hand, could be mindlessly sent just about anywhere.

Splitting by Scheme 1, in effect, acts no differently than the addition of dummy messages. Instead of one full dummy message being created, two half-dummy messages are. Certainly every message is now involved in the communication, but there is no added benefit to efficiency. Link cost is only ever increased as single message become two.

Splitting by Scheme 2, on the other hand, would theoretically act no differently than rerouting of whole messages, at best. Otherwise, the change in message transfer count is +2. While this +2 change might be useful in certain cases, it is unclear whether it ought to be used instead of two unit reroutes. Furthermore, additional complexity is incurred, as rerouting nodes must un-package “messages” with disparate halves, actively repackaging them to better approach their respective destinations.

4 Revised Alternative: Repackaging

In the initially proposed alternative, messages would be padded out to a length matching the longest one. That itself adds overhead to the communication, since a very short message may be expanded many-fold, the expansion being mostly filler material. In this revised alternative, let us consider messages as they are.

4.1 Assumptions

Messages being exchanged over a network, in their unmodified state, are of varying lengths. Additional header information needed to indicate split message order or combined message divisions would also require some message space; this space requirement is mostly overlooked for purposes of simplification.

4.2 Methodology

We take a look at all messages to be sent in a given time window to determine a target message size, T, that all messages should satisfy. Messages which are of length less than T would be packed together in a super-message. For example, if the target size T is 9, and we have two messages of size 4, we would attach the two smaller messages. Messages which are exactly of length T would remain untouched, whereas messages of length greater than T would be split into several pieces. These fragments may then be combined in new ways with smaller messages. Carrying out this procedure is expected to decrease link cost, as most of the filler padding is weeded out of the system.

4.3 Sample Scenario

Suppose we have a network of six nodes interconnected within a complete graph topology. For purposes of illustration, each node is assigned a random number of packets to send to all other nodes. Further assume all links in the network can support a maximum of 5 messages during each time-window. The actual traffic matrix is represented in Figure 4.

		Receiver					
		Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
Sender	Node 1	0	5	0	0	5	1
	Node 2	3	0	5	4	5	1
	Node 3	1	1	0	3	1	2
	Node 4	5	3	0	0	2	1
	Node 5	2	1	0	1	0	1
	Node 6	0	2	2	3	0	0

Figure 6: Actual Traffic Matrix View of the Sample Scenario

Not enough information is made available in the original traffic matrix design to implement the message repackaging alternative. A new message magnitude matrix is proposed. In this new matrix, each entry will contain size information of all messages being exchanged by each node. Figure 5 presents an example of what this might look like. Assume that links can only support messages with a maximum size of 50 (units omitted for generality). For purposes of simplification, message sizes occur in multiples of 10, and additional header requirements are not addressed.

	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
Node 1		{40, 20, 60, 10, 100}	{ }	{ }	{20, 20, 50, 90, 10}	{100}
Node 2	{10, 90, 10}		{50, 20, 60, 80, 90}	{30, 60, 50, 30}	{60, 30, 80, 50, 50}	{80}
Node 3	{30}	{20}		{80, 10, 40}	{70}	{80, 100}
Node 4	{30, 40, 50, 80, 50}	{20, 30, 80}			{10}	{10}
Node 5	{90, 30}	{60}	{ }	{70}		{90}
Node 6	{ }	{100, 20}	{80, 20}	{10, 80, 60}	{ }	

Figure 7: Message Magnitude Matrix based on Figure 4

In reality, such a Message Magnitude Matrix would never be constructed, but it helps provide an at-a-glance view of the network. Individual nodes would be expected to keep track of the sizes of just their own to-be-sent messages, and not those of neighboring nodes. Nodes would also be aware of the message size the network and its links can support.

Let us take a look at the Node 1 → 2 relationship. Node 1 might combine the message of size 40 with that of size 10, while splitting the message of size 100 into two messages of size 50. The net change in message count is 0.

Another interesting relationship is that of Node 6 → 4. Node 6 might split the message of size 80 into one of 30 and another of 50. Furthermore, the message of size 60 might be split into one of 50 and another of 10. Now, we have newly split off messages of size 30 and 10 which we can combine with the unmodified message of size 10. Again, the net change is 0.

Now, Let us take a look at the Node 1 → 5 relationship. Node 1 might combine 20 + 20 + 10 into one message, and split 90 into two messages. The net change in message count is -1.

4.4 Evaluation

This repackaging alternative is promising, since it can provide a negative net change in message count, where the simple splitting schemes initially proposed were unable to do so. Where the original goal was to cut the number of dummy messages being transmitted across the network, this alternative additionally cuts the amount of padding applied to individual messages.

Of course, this idea shares the limitation of splitting scheme 2 in that added intermediate node processing is necessary. To facilitate this, appropriate headers should be tacked on to messages being sent through the network and intermediate nodes configured to handle the added functionality. Headers may include fields for: packaged message count, division indices, split flag, split message id, and order within split message.

5 Another Alternative: A New Message Type

The new alternatives presented up to this point dealt with real messages being exchanged. Instead, let us now consider a new message type: control messages. As opposed to dummy messages, which are artificially constructed and hold no intrinsic purpose, control messages would enable inter-node configuration communications.

5.1 Implementation Details

Every so often, nodes will negotiate the number of messages to be sent out in subsequent time windows. This will be done through 2 control messages. One message will be sent by each node to all other nodes, containing the expected # of messages it intends to send in the future. Each node will aggregate all the expected counts it receives, and determine the minimum. If it agrees with that number, it will broadcast that value in the 2^{nd} message. Otherwise, the node will broadcast the higher value it requires. Each node once again aggregates all the counts it receives, and this time chooses the maximum. This value will be the number of messages to be sent.

5.2 Evaluation

Control messages enable the elimination of dummy messages, but control messages are only really useful if introduced to networks that experience large volumes of traffic between each pair of nodes. As control messages introduce one or two additional messages per link (and possibly one additional time-window), a network in which node activity is usually low will be burdened with considerable cost. On the other hand, if nodes regularly send many messages to every other node, then a couple more should be tolerable. Additionally, control messages would be best employed in networks that do not handle urgent/time-sensitive messages.

6 Related Work

The Neutral Traffic Matrix approach was outlined in detail in Newman's paper on Traffic Analysis Prevention (TAP), in which a combination of rerouting and post-rerouting dummy packet insertion was explored to achieve matrix neutrality [1]. This work was expanded in another paper by Newman et al., in which TAP metrics were proposed to judge the true effectiveness of rerouting and padding [2]. These two works inspired the question: is there something that can do better?

The NetCamo system of Texas A&M University makes use of traffic padding and rerouting to prevent traffic analysis in mission critical systems, while also providing performance guarantees [3]. The NetCamo system had a functionality to increase rerouting paths, but it was deemed ineffective against an adversary that could compromise nodes. In addition, dummy

messages used in one method of traffic padding were found to be easily identifiable by an attacker, as larger amounts of real messages could influence timer behavior of the system.

Any changes to network traffic behavior carry a risk of unearthing a covert channel; such channels can be tackled through auditing [4]. It is important to keep this in mind as different alternatives to rerouting and padding are explored, and the control message alternative proposed might help to this end.

7 Future Work

The idea for the revised alternative presented in Section 4 of this paper was first explored toward the latter stages of this study. Therefore, many details are not fully addressed in the possible implementation of the repackaging alternative, including header construction specifics and optimal number of messages to be aggregated. A closer look at details such as these is left for future work.

Moving forward, Michael Rabin's Information Dispersal Algorithm will be worthwhile to examine in detail. The algorithm enables a file to be broken into a number of pieces, but some of the data is duplicated across pieces in such a way that a subset of those pieces may be collected to reconstruct the entire file [5]. If applied to the splitting of larger messages, this may add some additional flexibility to the repackaging alternative.

Another avenue for future exploration is the combination of the repackaging alternative with the control message system introduced in Section 5. It will be interesting to see whether dynamic adjustments to node sending practices might facilitate more effective repackaging.

8 Conclusions

Several new means of realizing a Neutral Traffic Matrix are proposed: splitting, repackaging, and control message coordination. The initial splitting idea was motivated by a desire to make better use of existing messages in a network, rather than introducing dummy, throwaway messages. The proposed splitting was discovered to not add much in terms of efficiency; in essence, full dummy messages were being replaced by lesser-dummy messages, and additional overhead was incurred in production of these lesser-dummy messages.

The limitations of splitting were tackled by a new repackaging alternative. In repackaging, another dimension is taken into account: the size of messages. By not padding messages from the get-go, it becomes possible to naturally tie together and break up messages without artificial splitting practices. Padding is as much a throwaway practice as dummy messages, so it makes sense to minimize required padding while also opposing dummy messages. If successful, more overall meaning would be derivable from each message passing through the network, while still keeping a Global Passive Adversary in the dark. Of the three alternatives presented, this seems to be the most interesting and applicable. A new Message Magnitude Matrix is introduced to provide an appropriate, more detailed view of traffic flow.

Purposeful control messages are also explored as an alternative to dummy messages, although their situational nature suggests other alternatives may be preferred. Such messages would, however, enable nodes to dynamically change sending behavior, which might further help obfuscate underlying traffic patterns.

9 Acknowledgements

I thank Professor Newman and my fellow students taking the CIS 6930 course on Cryptographic Anonymity during the Fall 2015 semester for their feedback during the presentations of my work. I thank Dr. Newman, in particular, for suggesting I take a look at the repackaging alternative to make use of the naturally varying sizes of messages passing through networks.

References

1. R.E. Newman-Wolfe and B.R. Venkatraman. "High Level Prevention of Traffic Analysis," *Seventh Annual Computer Security and Applications Conference*, San Antonio, Texas, December 2-6, 1991, pp. 102-109.

2. Richard E. Newman, Ira S. Moskowitz, Paul Syverson and Andrei Serjantov. "Metrics for Traffic Analysis Prevention," In PET 2003, Dresden, March 2003.

3. X. Fu, B. Graham, Y. Guan, R. Bettati and W. Zhao. "NetCamo: Camouflaging Network Traffic for Real-Time Applications," *Texas Workshop Security of Information Systems*, April 2003.

4. B.R. Venkatraman and R.E. Wolfe. "Capacity Estimation and Auditability of Network Covert Channels," *1995 IEEE Computer Society Symp. Security and Privacy*, pp. 186-198.

5. Michael Rabin. "Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance," In ACM April 1989, pp.335-348.