Passive Attack Analysis for Connection-Based Anonymity Systems

Andrei Serjantov and Peter Sewell

University of Cambridge Computer Laboratory
William Gates Building, JJ Thomson Avenue
Cambridge CB3 0FD, United Kingdom
First.Last@cl.cam.ac.uk

Abstract. In this paper we consider low latency connection-based anonymity systems which can be used for applications like web browsing or SSH. Although several such systems have been designed and built, their anonymity has so far not been adequately evaluated.
We analyse the anonymity of connection-based systems against passive adversaries. We give a precise description of two attacks, evaluate their effectiveness, and calculate the amount of traffic necessary to provide a minimum degree of protection against them.

1 Introduction

Systems for anonymous interaction are a basic building block for application-level privacy. The anonymity properties these systems aim to provide are subtle: in contrast to most security protocols, they must cover statistical traffic analysis attacks. A number of anonymity systems have been designed, starting from [Cha81]. They can be divided into two classes:

- Message-based (mix) systems, for asynchronous (email) messages. They provide anonymity by delaying and mixing messages; email can tolerate substantial delay. There is a significant body of work on their design [Cot94,GT96,DDM03] and implementation [MC00,DDM02].
- Connection-based systems, for low-latency bidirectional communication (e.g. SSH connections and web browsing). There are several implemented designs [GRS99,RP02,FM02]. Although these are also sometimes called mix systems, current designs do not do any mixing as such, so we choose not to use this term in the paper.

Analysis of these systems is crucial: users need more than a "warm fuzzy feeling" that they are anonymous. For message-based systems, we have well-understood threat models and both qualitative [BPS00,Ray00] and (some) quantitative [SD02,SDS02] analysis. For connection-based systems, on the other hand, the threats are harder to characterise – the low-latency constraint makes these systems vulnerable to powerful timing attacks. Qualitative analyses include [BMS01]. Quantitative analysis has so far been limited to evaluating the impact of compromised nodes on the anonymity provided. [STRL00,WALS02].

E. Snekkenes and D. Gollmann (Eds.): ESORICS 2003, LNCS 2808, pp. 116–131, 2003.

In this paper we examine the protection such systems can provide against a passive attacker (i.e. one who watches the network rather than tries to compromise nodes). Some systems, MorphMix [RP02] for example, explicitly state that they are vulnerable to such an adversary; our work may provide insight into ways of strengthening them. Others, notably Tarzan [FM02], employ an expensive dummy traffic policy in an effort to protect against this adversary. We show that it is possible to avoid this.

It is important to note that in this paper we consider connection-based anonymity systems which are running on top of standard Internet protocols, i.e. packet-switched networks. We do not consider schemes over circuit-switched networks like [PPW91].

2 Systems and Usage

We begin by outlining the application scenario, high-level anonymity goals, and system architecture that we consider in our analysis. The latter is a distillation of the key choices of Onion Routing, Tarzan and MorphMix [GRS99,FM02,RP02].

Scenario. We are primarily considering systems for anonymous web browsing. A number of *users*, running anonymity clients, connect through the system to some *web servers* (not running special software). HTTP requests and responses both travel through the system.

System goals. Such a system should:

1. provide usable web browsing, with no more than a few seconds additional latency; and
2. make it hard for an attacker to determine what any given user is browsing[1]. In particular, as we discuss below, it should protect a user against an attacker who can observe all traffic on the path of their connection. Note that to do this, the adversary does not need to observe all traffic in the anonymity system.

The goals clearly involve a trade-off: the more delay, the higher the (potential) anonymity.

Architecture. We make some basic assumptions about the system structure:

– The system consists of a number of *nodes*. Some designs have a 'classic' architecture, with relatively few nodes, whereas others have a 'P2P' architecture, with nodes run by each user. Each node has logical *links* to some (not necessarily all) other nodes, along which it forwards traffic. Links are implemented above inter-node TCP connections between IP/port addresses, link-encrypted. To protect against node compromise, each connection passes through several nodes. Nodes also accept connections from end-user clients.

[1] The system need not protect against the attacker determining *that* the user is browsing, or which web servers are being accessed through the anonymity system. The system should, of course, protect against the webserver determining who is browsing the website, unless it is compromised by an application-level feature (e.g. cookies).

- To protect against the simple passive observer, who can bitwise compare traffic entering and leaving a node, traffic is onion-encrypted (as first suggested in [Cha81]). This also protects against some node compromise attacks.
- The length of messages remains observable, so the data is divided into fixed-length *cells*. Typically these are small (in the Onion Routing design each cell carries 128 bytes of data).
- "Onion connection" setup is expensive, so each client/server communication is routed via the same sequence of nodes. (Application proxying may reduce the number of communications, e.g. fetching all objects of a webpage in one communication.)
- Routes may be chosen either by the end-user or by the network.

This architecture broadly follows the design of 2nd generation Onion Routing [ord], Tarzan [FM02], and MorphMix [RP02]. Some of our results are also applicable to JAP [JAP] and the Freedom Network [BGS00].

Adding *dummy traffic* is a standard technique used in message-based anonymity systems e.g. Mixmaster. For connection-based systems, however, practical experience shows the bandwidth requirements of nodes are large; the additional cost of dummies must be minimised. Accordingly, in this paper we assume that inter-node links do not involve dummies (though it turns out to be beneficial to apply some padding to the links between the client and the first node). We leave for future work the question of how a given quantity of dummy traffic can be most effectively used.

3 Threat Models

Prior work on threat models for connection-based systems has focused on the threat of malicious nodes, looking at how anonymity is affected by the fraction of attacker-controlled nodes [Shm02,STRL00].

In this paper we focus on the threat of traffic analysis by a *passive* observer. Earlier notions of "global passive" attacker, as used in analysis of message-based systems, are too vague for connection-based systems. The threats must be stated more precisely: the quality (time accuracy) of the traffic data available to different global passive attackers may vary considerably, making different traffic analyses possible. We leave analysis of *active* attacks to future work.

There are several different low-level mechanisms an attacker might use to obtain traffic data, differing in the quality of data they make available, and in the effort required.

- Attacker-controlled nodes. Outside our scope.
- By applying legal (or sublegal) pressure to an ISP, a high-resolution traffic monitor can be installed on a machine on the same collision domain as a node. This could capture all IP packets travelling to and from other nodes, with precise (sub-millisecond) timestamps; that data could be forwarded online to an analysis machine. Note that if nodes are distributed among judicial domains, it is hard to attack a substantial proportion of them.

- By compromising a machine in the same collision domain as a node the same data could be captured, though here there may be difficulties in surreptitiously forwarding it to the analyser.
- By installing non-intrusive fibre taps 'in the field', on the fibres that carry traffic between nodes [Hod91], one can capture similar data, but here, as there are typically routers between a node and an external fibre, some timing accuracy will be lost (several router delay variances). How many such attackers are required to intercept all node-to-node communications depends on the topology, but typically examining just backbone fibres will not suffice.
- Traffic data can also be obtained by compromising a router on each of the inter-node links and placing traffic monitoring code there. However, here the attacker is more likely to get per link packet counts (over large fractions of a second) rather than per-packet data with timestamps. These can be retrieved via the standard SNMP protocol. More accuracy can be obtained by compromising routers closer to each node.

Broadly, all these attackers gain access to the same class of data – the number of packets that travel between pairs of nodes (on anonymity-system logical links) during particular time intervals. The packet counting interval determines what kinds of traffic analysis the attacker can perform: taking long intervals amounts to low-pass filtering of the data, erasing informative high-frequency components.

A further distinction is between *per-interval* and *waveform* analysis. In the former, each packet-counting interval is treated separately – the attacker can forget the data for each interval after it has been analysed – whereas in the latter a substantial region of the traffic waveform must be considered. The latter may obviously be more expensive to mount.

4 Analysis: Lone Connection Tracking

Our first analysis is based on packet counting. We recall that traffic travels down a connection in small cells. Consider a node in the system. During a particular time interval the number of packets on each of the connections travelling through it is highly likely to be different. This attack requires the delay introduced by the node to be small, compared to the size of the time interval, so the number of incoming and outgoing packets of the node on each connection will be very similar. They will not be identical as some packets will have been in the node before the interval started and some will remain after the interval ends.

A passive attacker can observe the number of packets of a connection which arrive at the node and leave the node only if this connection is *lone* – it is the only one travelling down that link – on its incoming and outgoing links during the time interval. This scenario is illustrated on Figure 1. It is clear that the numbers of packets on links from D to the node X and from X to T are very similar, so the attacker can be confident that the connection(s) from D have been forwarded to T. Naturally, there is a possibility that he is mistaken: one of the connections from A, B or C carried 1079 packets and was forwarded to T, while the connection(s) from D was forwarded to Q, R or S. However, the

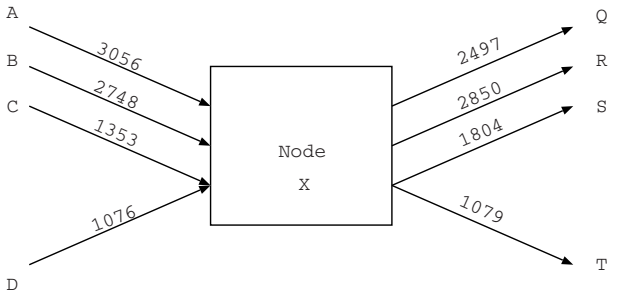

Fig. 1. Each arrow represents an incoming or an outgoing link. The number on the arrow represents the number of packets observed by an attacker on the link over one time period.

probability of this is very small (we do not calculate it here) as we assume that the number of packets on each of the incoming connections is highly likely to be different. We can further reduce this probability by doing the same observations during a different time interval and checking the results are consistent.

Note that this attack does not require a global packet counting attacker, merely one who observes all the links a connection travels on.

This attack is based on assumptions, some of which we have touched on already:

- The packet counting interval is much larger than the mix delay. This is necessary as otherwise the packets inside the mix at the starts and ends of packet counting intervals will make the incoming and outgoing packet counts dissimilar.
- The packet counting interval is much smaller than the mean time between new connections being set up on this pair of links. The longer the time interval, the more likely there is to be a new connection initiated which will traverse an incoming or an outgoing link, thereby ruining the packet counts and thus the attack. Note that if the adversary is unable to obtain packet counts for short enough time periods (e.g. due to extracting packet counts via SNMP), he loses some opportunity for attacks.

It may seem that the attacker can just as easily count packets coming in from the users to the first node of the connection and try to correlate this with the packet counts coming out of the anonymity system to the webservers[2]. Such an attack is less likely to succeed (and easier to protect against) for the following reasons:

- In the description of the attack on a single node, we required that the packet counting interval should be much larger than the node delay. Thus, in the case of mounting the attack on the anonymity system as a whole, the packet

[2] This is the extreme version of packet counting across several nodes.

counting interval will have to be made much larger than the delay on all the nodes of a connection together plus all the link delays. This increases the chances of the user initiating another connection to the same first node, thereby confusing the packet counts. Implementors of connection based systems should note that this is a good and cheap defence strategy (though it relies on the first node being uncompromised).
- A small amount of padding between the user and the first node protects against the attack. This requires much less bandwidth than padding each link in the anonymity system as suggested by [Ren03]. Of course, it would be desirable to also pad the link from the last node to the webserver, but this is impossible as the webserver is not running anonymity software.
- It may be possible to arrange the first link not to be observable by the attacker (e.g. run a node at the edge of a corporate network and ensure that everyone inside the company uses it as their first node).

Having shown that lone connections allow the attacker to compromise anonymity, we now calculate how many such connections a system is likely to have under different conditions (and thus how likely a user's connection is to be compromised). First, we derive an approximation, and then examine the subject in more detail using a simulator. Finally we suggest ways of defending against this attack.

4.1 Mean-Based Analysis

Assume the users initiate on average c connections per second, each forwarded along ℓ links (inside the network) and that there are n nodes in the anonymity system. Furthermore assume each connection has duration dur.

Thus on average at any instant there are $c \times dur$ connections. Each connection exists on ℓ links, so on average there are $c \times dur \times \ell$ link-occupancies. If there is a link between each pair of nodes, there are roughly $n \times n$ links[3]. On average there are $c \times dur \times \ell/(n \times n)$ connections per link. It is clear that the absolute lower bound of the number of connections per link is 1, and for a good anonymity system this number should be much greater.

Let us illustrate this with an example. Suppose we have a system with $n = 30$ nodes, the users initiate connections through $\ell = 3$ network links (or 4 nodes), each lasting $dur = 2$ seconds. If each node can talk to every other, then around 150 connection initiations per second are necessary for this system to provide at least some anonymity.

4.2 Definitions

It is clear that the approximations calculated in the previous section are rather crude. We now proceed to define lone connections formally and show how to work out the fraction of lone connections of a particular system.

[3] It is debatable whether routes with the same node occurring twice consecutively should be allowed.

First, define the anonymity system graph as a set of nodes G with $|G| = n$ and a set of edges (links) E, with each edge being a pair of nodes. A path (a connection), then, is a sequence of edges. Take all connections c_i (of length l_i) which are open during a particular packet counting interval and let $g = \{\!|[e_{1,1} \dots e_{1,l_1}]; [e_{2,1} \dots e_{2,l_2}]; \dots |\!\}$ be the multiset of paths of these connections[4]. We can easily express the number of connections on each link resulting from such a configuration.

$$f_g(e) = \sum_{p \in g} \text{occurrences of } e \text{ in } p$$

A connection is lone when it is lone on all the links it is going through. Now calculating the set of lone connections in a configuration is straightforward:

$$\mathsf{lone} = |\{\!|p | p \in g \wedge \forall e \in p. f_g(e) = 1|\!\}|$$

We can also find the fraction of lone connections: $\frac{\mathsf{lone}}{|g|}$.

We now go on to define the probability of a connection going through the anonymity system being lone.

First, let us assume some parameters of the anonymity system.

- $\Phi(c)$, the probability that c connections go through the anonymity system during the same interval.
- $\Psi(j)$, the probability that a route going through it is chosen to have length j.
- The graph of the anonymity system is given by G, E.
- The maximum number of connections which can go through a system is $\mathsf{max\_c}$ and the maximum route length is $\mathsf{max\_rt}$.

Now define $\mathcal{G}([l_1, \dots, l_c])$, to be the set of all multisets of paths of lengths $[l_1, \dots, l_c]$ in the graph G, E.

$$\mathcal{G}([l_1, \dots, l_c]) = \{m | m = \{\!|[e_{1,1} \dots e_{1,l_1}]; [e_{2,1} \dots e_{2,l_2}]; \dots; [e_{c,1} \dots e_{c,l_c}]|\!\} \wedge e_{x,y} \in m \Rightarrow e_{x,y} \in E\}$$

Now, the probability P of a particular connection being unmixed is:

$$P = \sum_{c \in 0 \dots \mathsf{max\_c}} \Phi(c) \times \sum_{\substack{L = [l_1, \dots, l_c] \\ \forall i. l_i \leq \mathsf{max\_rt}}} \prod_{l \in L} \Psi(l) \times \sum_{g \in \mathcal{G}(L)} \frac{|\{p | p \in g \wedge \forall e \in p. f_g(e) = 1\}|}{|g|}$$

There are several assumptions which we have made implicitly in the above formula. Firstly, the probability distribution for path lengths of all connections is the same, $\Psi(l)$. Secondly, paths are chosen uniformly at random from all the possible paths (of length l) through the graph G, E [5]. Finally, all the path lengths

[4] Paths can be identical, so we tag them with a unique integer.

[5] Hence the number of connections a node will handle is determined by the number of links it has to other nodes.

and all the paths themselves are chosen by the connection initiators independently.

Although the above formula defines the probability of a connection going through an anonymity system unmixed, it is hard to see the quantitative implications of it directly. We therefore make a simulation of the anonymity system.

4.3 Simulator Results

We have constructed a simulator which uses the definitions above to calculate the fraction of lone connections. Given a graph of nodes connected by links, and the number of connections we wish to simulate, it picks a route length for each connection ($\Psi(j)$ is assumed to be a uniform distribution between a minimum and a maximum value) and then generates the routes themselves. Then it calculates the fraction of lone connections (using the definitions above). Clearly, the fraction of lone connections going through the network is also the probability that a particular user's connection is going to be observed by the global packet counting attacker.

For example, let us take a peer to peer anonymity system with 100 nodes (each user running a node) all connected to each other. Suppose each of the 100 users initiates a connection ($\Phi(100) = 1$ during the packet counting interval we are considering) through a minimum of 2 and a maximum of 4 network links. This system, provides very low anonymity – around 92% of the connections going through it are lone.

A graph of the number of nodes vs the probability of connection compromise is shown in Figure 2. There are 60 connections going through the network and each connection is going through 2 network links.

It is worth noting that the fraction of lone connections is not the only measure of anonymity we could have used. Indeed, although it conveys a very clear message to the user (the probability of the connection they are about to establish being observable), it also suffers from some disadvantages. First, it does not indicate how many other connections a particular connections has been mixed with as an anonymity set (or the information theoretic metric of [SD02]) does. It is worth pointing out that if a connection is lone on some, but not all of its links, its anonymity set is very much reduced (which is not reflected in the probability of it being compromised). Secondly, the designers of the anonymity system might like to know the probability of any one or more connections being compromised – a much stronger property. We leave calculating these metrics and analysing the attack in detail to (rather tedious) future work.

4.4 Protection

As we saw in the previous section, the packet counting attack on the lone connections is quite powerful against some systems. Here we examine ways of protecting against it.

Firstly, more traffic (and/or fewer nodes) makes the system much less vulnerable to the attack. Modifying the system from the example above to one with 20

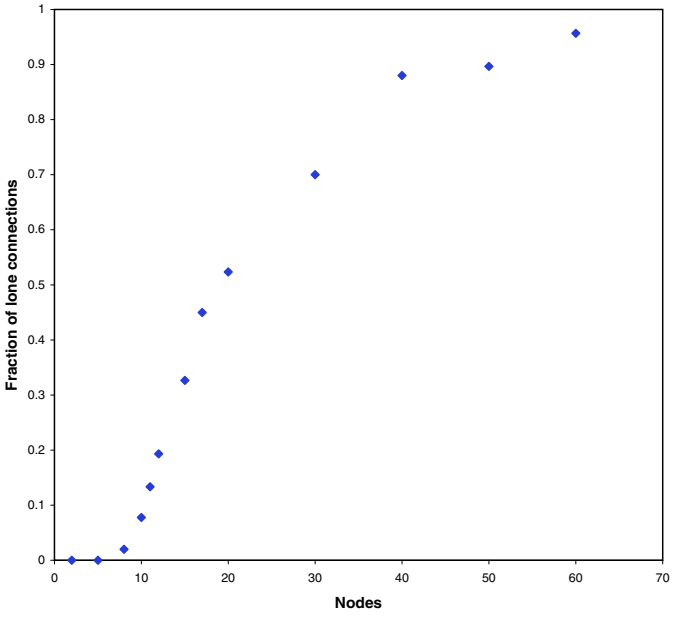

Fig. 2. Graph of the number of nodes in an anonymity system vs the fraction of lone connections.

nodes with 200 connections going through it (and keeping the route length the same at between 2 and 4 links) reduces the fraction of compromised connections from 92% to 2.5%.

Secondly, increasing the route length helps increase the total volume of traffic in the network, but also has the undesirable effect of increasing latency. For example, doubling the route length in our example above (100 nodes, 100 connections, route length of 4 to 8 network links) reduces the probability of a connection being compromised from around 92% to 72%. The graph showing how route length affect the fraction of lone connections is show in Figure 3.

Thirdly, and most importantly, we can design the architecture of the system to suit the amount of traffic we expect to flow through it. If there is very little traffic, a cascade ought to be used. If there is slightly more, a restricted route architecture (see [Dan03]) can be employed to dramatically decrease the fraction of lone connections. For instance, for an anonymity system of 100 nodes, each able to forward traffic to 5 others (with route length of between 2 and 4 links and 100 connections), the fraction of lone connections is reduced to around 17%. This is still, however, unacceptable and suggests that making every client run a node is not a good choice for a strong anonymity system.

The fact that peer to peer anonymity systems do not provide much anonymity against the global passive attacker is an important conclusion which may seem counterintuitive. However, it can be easily explained when we think of the fact that the connections have to "fill up" the network graph at least twice. If we

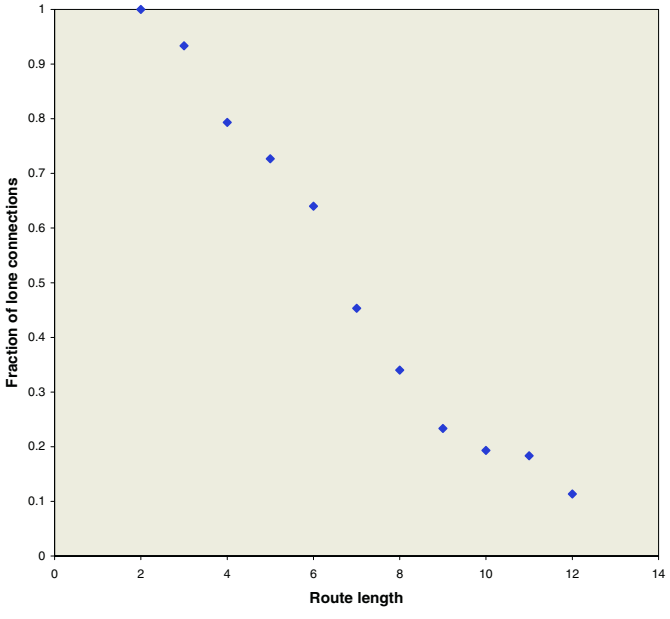

Fig. 3. Graph of the route length vs the fraction of lone connections.

run a node at every client, the number of connections will be within a constant factor of number of nodes, kn. The number of links in a fully connected network roughly n^2, so the anonymity of such a network will be low. This is clear even from the crude calculations in Section 4.1.

As well as designing the system in a way which suits the expected level of traffic, we need to be able to handle daily or weekly variations in the number of connections established through the system. This may be possible by dynamically reconfiguring the network topology from cascade to restricted routes to full network. Of course, short term (daily) variations in traffic could not be handled in this way; it should be handled by suggesting to the users the number of nodes their connections should go through to stay anonymous.

5 Analysis: Connection-Start Tracking

Our second attack is based on tracking the increase in the volume of traffic from an incoming to an outgoing link of a node which results from data starting to flow on a new connection. This increase happens when the webserver is starting to send the webpage data in response to a request made by a client. We call such an increase a "connection start". We note that the propagation of such a connection start through a node is observable to a packet counting attacker, even if the connection is not lone. If the node does not delay traffic significantly (as current systems do not), the attacker will observe a node in a steady state;

a start of connection arriving followed by a start of connection leaving, and will deduce where the new connection has come from and been forwarded to.

Hence, nodes must delay traffic (but still provide low latency communications). The most appropriate mixing strategy here is the SG-Mix of Kesdogan (see [KEB98]). It is easy to analyse, handles each packet separately and does not rely on batching. We proceed to describe this mix and examine how it can help us protect against the above attack.

The SG-Mix mix treats each packet (cell) independently. When a cell arrives, the mix draws a random value from an exponential distribution with parameter μ and delays the cell by that time. The mean delay is of the mix is thus $1/\mu$.

Assume the users initiate (on average) c connections per second, each going through ℓ nodes. The system consists of n nodes. Write λ for the mean rate of arrival of starts of connections (per second) to a particular node. We have $\lambda = c\ell/n$.

Assume further that the arrivals of the starts of connections to the node are Poisson distributed (with parameter λ).

Now, the attacker tracks a connection through a mix iff:

1. When the start of the connection arrives, the mix is "empty of starts of connections". This means that there has not been an incoming start of connection not followed by an outgoing one.
2. Having arrived on an incoming link, the start of the connection leaves the mix whilst no other start of a connection has arrived.

This is essentially the $n-1$ attack scenario described in [KEB98], though performed here for starts of connections instead of individual asynchronous messages. We want to choose the parameters λ and μ such that the probability of the attacker tracking a start of connections through all the mixes is small.

First, consider the probability that a connection is tracked through one node.

$$\frac{e^{-\frac{\lambda}{\mu}}}{1+\frac{\lambda}{\mu}}$$

The probability of the attacker tracking a particular connection which is going through ℓ mixes is:

$$\left(\frac{e^{-\frac{\lambda}{\mu}}}{1+\frac{\lambda}{\mu}}\right)^{\ell}$$

substituting in the expression for λ from above gives:

$$\left(\frac{e^{-\frac{c\ell}{n\mu}}}{1+\frac{c\ell}{n\mu}}\right)^{\ell}$$

A user of this system would incur a delay of roughly $2\ell/\mu$ seconds for onion connection setup, ℓ/μ for a request and ℓ/μ for a response, or a total *connection delay* of: $4\ell/\mu$.

We will now do some order-of-magnitude calculations to see how much traffic needs to go through the anonymity system to get acceptable delay and anonymity values.

Clearly, as long as $\ell \geq 3$ and $cl/nm \geq 1$, the probability of tracking a connection is low (< 0.006). Hence, $c\ell/n\mu \geq 1$ or $c\ell \geq n\mu$.

Suppose $\ell = 3$ and the maximum acceptable delay is 2 seconds, hence $4\ell/\mu = 2$, hence $\mu = 2\ell = 6$. Substituting in, we get $c \geq 2n$. This implies that the users of the system have to initiate twice as many connections per second as there are nodes.

Suppose we have U users browsing every day. If each browses 100 pages a day, $c = 100/(3600 \times 24)$

Now suppose we have an anonymity system of 30 nodes. We find the number of users U needed for the system to provide anonymity. $U \times 100/(3600 \times 24) \geq 2 \times 30$, or $U \geq 2 \times 30 \times 36 \times 24 = 51000$. This is a realistic target for a small to medium size anonymity system.

Naturally, these calculations are rather crude as they involve the mean amount of traffic (and suppose that the traffic is evenly distributed throughout the day). More traffic is required to to protect against traffic troughs (e.g. at night).

It is worth considering the quality of traffic data the adversary has to have access to to mount such an attack. If a timestamp for every packet is available, the attack can be mounted with maximum effectiveness. However, if the adversary can only do packet counting over some time intervals, then the time interval must be much longer than the node delay and much smaller than the interarrival times of starts of connections *to a node*. Note that this is more precision than was required for the lone connections attack (the time interval there had to be much less than the interarrival times of connections *on a single link*).

5.1 Working with Richer Traffic Features

Before we considered starts of connections and showed that if these are allowed to propagate through the network, then a certain level of traffic is required to maintain anonymity. Now we consider how the attacker could use more general traffic features (spikes) to track individual connections. This is an example of a waveform analysis – data from several intervals will be required.

Let us consider a simple case of a node with 2 incoming and 2 outgoing links. The adversary sees a spike on one of the incoming links (say from A) and one of the outgoing links (to Q) some time later. He knows that both the links which exhibited spikes have lone connections on them[6], but the other links (from B and to R) contain many connections, so some spikes may be hidden. The smart adversary does not jump to conclusions about a correlation between the links with spikes, but instead calculates the probability of it.

There are two possibilities: Either the attacker is correct and the spike from A really went to Q, or the attacker is mistaken and there was a hidden spike

[6] This constraint is easily relaxed, we include it for clarity of exposition

which came in from B and went to Q, while the spike from A got forwarded to R and hidden in the traffic.

The probability of the former is $1/2$ (assuming the connection from A was equally likely to be forwarded to Q and R). The probability of the latter is $P(\text{spike}) \times 1/2$. Hence, the attacker is correct with probability $\frac{1}{1+P(\text{spike})}$. The probability of a spike occurring on a link is low, so the attacker of the attacker being correct is high.

A complete analysis is not presented here – we would need to examine real connection traffic to determine the probability of spikes occurring and determine what other kinds of traffic features we might make use of. It is notable that interactive applications like SSH are much more vulnerable to this kind of attack than web browsing as the traffic is much less uniform. In general one would expect to use signal-processing techniques – filtering the signal to frequency ranges known to include identifiable features and/or calculating running correlations between signals and common feature patterns. We leave this for future work.

6 Discussion and Solutions

In the previous sections we looked at two powerful attacks which can be mounted on connection-based anonymity systems by passive attackers, quantitatively evaluated their effectiveness and assessed potential protection measures.

Unlike all previous analyses, we stayed clear of using vague and costly proposals of adding dummy traffic to the system, instead calculating the amount of user connections required to maintain anonymity. This approach is crucial for building efficient, fast and therefore deployable connection based anonymity system, whilst still providing anonymity to the users.

However, we have not examined all the attacks which the adversaries can potentially mount against connection-based anonymity systems. In particular, in this paper we have not considered the "first and last node" attack or any active attacks. We comment upon them briefly here.

The "first and last node" attack involves the attacker compromising the first and the last node of a particular connection. He can now filter padding from the client to the first node (if there was any) and modify traffic travelling in both directions. In particular, he can insert a signal into the inter-arrival times of cells of a particular connection and then look for it (low-pass filtered to account for the variances in network and mix delays) on the other side. As the packets are small, the signal is likely to carry a substantial amount of information and help the attacker succeed. Note that an active attacker who can modify traffic on links (but has not compromised any nodes has the same capability).

There are several potential countermeasures which will help make this attack less powerful. First, longer routes will help reduce the amount of signal which propagates from the first to the last node. Secondly, increasing the packet size (and thus decreasing the number of packets) will help reduce the size of the signal which can be inserted into the connection. In the limit, if all webpages fit into one packet, active attacks become ineffective (though this comes with a massive efficiency loss).

We also briefly mentioned traffic shaping as a countermeasure to the "lone connections" attack. It is worth noting that such a traffic shaping policy would have to make all the connections in the anonymity system have the same profile, which is likely to be expensive in terms of introducing delays or bandwidth (dummy traffic). We have not investigated this mostly because protection against the attacks outlined could be achieved by cheaper means.

One of the implications of the results presented here is that (peer to peer) anonymity systems which involve all the users running nodes are impractical simply because there is not enough traffic to fill all the links. Therefore, it is evident that adding nodes provides *less* anonymity (contrary to popular belief) against the global passive attacker. Whether this statement is true for the case of partial attackers remains the subject of future work.

Another interesting line of research is to see whether an anonymity system could be built using an architecture which elevates a subset of the P2P nodes to a "supernode" status. The role of such supernodes would be to perform mixing, while the other, "lesser" nodes would simply forward traffic. This may yield a good compromise between the performance and scalability of current P2P architectures and resistance against passive attackers.

7 Related Work

As mentioned before, there is relatively little quantitative analysis of connection-based anonymity systems. The notable exception is [STRL00] which gives a detailed account of the security of the first generation of the Onion Routing system against compromised nodes.

To the best of our knowledge, the first work which describes the packet counting attack is the analysis by Back, Möller and Stiglic [BMS01], however, they fail to point out the crucial requirement of the connection being lone on its link. (Although under some conditions we may be able to decompose the sums of numbers of packets on different links uniquely into the number of packets belonging to each connection, this is unlikely to be important in practice). Another recent work [Ren03] analyses packet counting attacks but remains vague about the assumptions on node delay and details of connections travelling on links, and proposes a constant dummy traffic policy which turns out to be costly.

The connection-start tracking problem was observed in [PPW91]; there it is solved by dividing the connection into a number of "time slice channels". This scheme requires extra overhead, and its effectiveness remains to be evaluated.

There are also systems which provide anonymous connections for web browsing [SBS02] which do not follow the "mix" architecture of Chaum, but they also lack quantitative analyses of the anonymity provided.

8 Conclusion

We examined in some detail two attacks which can be mounted by passive adversaries on connection-based anonymity systems. These compromise existing

anonymity systems completely. However, the threats can be analysed and can be protected against without resorting to dummy traffic and keeping the delay to users' connections acceptable.

We note that these threats to connection-based anonymity systems (some of which are currently in the process of being implemented and deployed) are practical and realistic, and the designers should take them into account, especially as the methods of protection need not be costly.

Finally, this paper shows that quantitative analysis of connection-based anonymity systems is just as feasible as of message-based ones. Furthermore, such analysis is required to develop and evaluate methods of protection against real threats. As a promising direction for future work, we suggest that mounting real attacks on implemented (and deployed) anonymity systems will provide further insight into the measures necessary to keep anonymity systems anonymous.

Acknowledgements

We acknowledge support from a Royal Society University Research Fellowship (Sewell), EPSRC grant GRN 24872 and EC FET-GC project IST-2001-33234 PEPITO. We also thank Tuomas Aura, Richard Clayton, George Danezis, Andreas Pfitzmann, Marc Rennhard, Paul Syverson for thoughtful comments.

References

BGS00. P. Boucher, I. Goldberg, and A. Shostack. Freedom system 2.0 architecture. `http://www.freedom.net/info/whitepapers/`, 2000. Zero-Knowledge Sytems, Inc.

BMS01. A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In *Information Hiding Workshop*, pages 245–257. LNCS 2137, 2001.

BPS00. O. Berthold, A. Pfitzmann, and R. Standtke. The disadvantages of free MIX routes and how to overcome them. In *Workshop on the Design Issues in Anonymity and Observability*, LNCS 2009, pages 10–29. 2000.

Cha81. D. Chaum. Untraceable electronic mail, return addresses and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

Cot94. L. Cottrell. Mixmaster and remailer attacks, 1994. `http://www.obscura.com/~loki/remailer/remailer-essay.html`.

Dan03. G. Danezis. Mix-networks with restricted routes. In *Privacy Enhancing Technologies*, LNCS 2760. Dresden, Germany, 2003.

DDM02. G. Danezis, R. Dingledine, and N. Mathewson. Type III (Mixminion) Mix Protocol Specifications, 2002. `http://mixminion.net/minion-spec.txt`.

DDM03. G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *IEEE Security and Privacy*. 2003.

FM02. M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Computer and Communications Security (CCS)*. 2002.

GRS99. D. Goldschlag, M. Reed, and P. Syverson. Onion Routing for anonymous and private internet connections. *Communications of the ACM*, 42(2):39–41, 1999.

GT96. C. Gülcü and G. Tsudik. Mixing Email with *Babel*. In *Internet Society Symposium on Network and Distributed Sytem Security*, pages 2–16. 1996.

Hod91. H. Hodara. Secure fiberoptic communications. In *Symposium on Electromagnetic Security for Information Protection*. Rome, Italy, 1991.

JAP. The JAP project. `http://anon.inf.tu-dresden.de/index_en.html`.

KEB98. D. Kesdogan, J. Egner, and R. Büschkes. Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Information Hiding Workshop*. LNCS 1525, 1998.

MC00. U. Moeller and L. Cottrell. *Mixmaster Protocol Version 3*, 2000. `http://www.eskimo.com/~rowdenw/crypt/Mix/draft-moeller-v3-01.txt`.

ord. Onion Routing developers mailing list. `http://archives.seul.org/or/dev/`.

PPW91. A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-mixes: Untraceable communication with very small bandwidth overhead. In *Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, pages 451–463. 1991.

Ray00. J. Raymond. Traffic analysis: Protocols, attacks, design issues, and open problems. In *Workshop on the Design Issues in Anonymity and Observability*, LNCS 2009, pages 10–29. 2000.

Ren03. M. Rennhard. Practical anonymity for the masses with mix-networks. Technical Report 157, ETH Zurich,Switzerland, 2003.

RP02. M. Rennhard and B. Plattner. Introducing morphmix: Peer-to-peer based anonymous internet usage with collusion detection. In *Workshop on Privacy in the Electronic Society (WPES)*. Washington, DC, USA, 2002.

SBS02. R. Sherwood, B. Bhattacharjee, and A. Srinivasan. P5: A protocol for scalable anonymous communication. In *IEEE Security and Privacy*. 2002.

SD02. A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Privacy Enhancing Technologies*, LNCS 2482. 2002.

SDS02. A. Serjantov, R. Dingledine, and P. Syverson. From a trickle to a flood: Active attacks on several mix types. In *Information Hiding Workshop*, LNCS 2578. 2002.

Shm02. V. Shmatikov. Probabilistic analysis of anonymity. In *15th IEEE Computer Security Foundations Workshop*, pages 119–128. 2002.

STRL00. P. F. Syverson, G. Tsudik, M. G. Reed, and C. E. Landwehr. Towards an analysis of Onion Routing security. In *Workshop on Design Issues in Anonymity and Unobservability*, LNCS 2009. 2000.

WALS02. M. Wright, M. Adler, B. Levine, and C. Shields. An analysis of the degradation of anonymous protocols. In *ISOC Symposium on Network and Distributed System Security*. 2002.