Appeared in: Proceedings of the 5th Workshop on Privacy
Enhancing Technologies (PET 2005), Cavtat, CROATIA,
May 2005.

Unmixing Mix Traffic

Ye Zhu and Riccardo Bettati

Department of Computer Science
Texas A&M University
College Station TX 77843-3112, USA
zhuye@tamu.edu, bettati@cs.tamu.edu

Abstract. We apply blind source separation techniques from statistical signal processing to separate the traffic in a mix network into either individual flows or groups of flows. This separation requires no a priori information about the individual flows. As a result, unlinkability can be compromised without ever observing individual flows. Our experiments show that this attack is effective and scalable. By correlating separated groups of flows across nodes, a passive attacker can get an accurate traffic map of the mix network. We use a non-trivial network to show that the combined attack works. The experiments also show that multicast traffic can be dangerous for anonymity networks.

1 Introduction

In this paper, we describe a class of attacks on low-latency anonymity networks. These attacks, which we will call *flow separation* attacks, aim at *separating* (as opposed to *identifying*) flows inside a network, based on *aggregate* traffic information only. This is in contrast to many previously proposed attacks against mix networks, where the adversary has at least some information about one or more flows in the system.

Since Chaum [1] pioneered the basic idea of the anonymous communication systems, researchers have developed various mix-based anonymity systems for different applications. One of the main functions of the mix network is to mix the traffic flows and so render senders or receivers anonymous. Mix networks typically achieve this by perturbing the traffic in (a) the payload domain (through encryption), (b) in the route domain (through re-routing) and (c) in the timing domain (through batching and link padding). By using the flow separation attack, an attacker can separate the flows based on passively collected

This work is supported in part by the Texas Information Technology and Telecommunication Task Force.

traffic data. Further attacks by frequency spectrum matching or time domain cross-correlation [2] can then easily determine the path of a flow in the mix network if additional knowledge about the flow is available or determine the traffic directions in the mix network.

The flow separation attack employs the *blind source separation* model [3], which was originally defined to solve *cocktail party problem*: The blind source separation algorithms can extract one person's voice signal given the mixtures of voices in a cocktail party. Blind source separation algorithms solve the problem based on the independence between voices from different persons. Similarly, in a mix network, we can use blind source separation algorithms to separate independent flows.

The contributions of this paper can be summarized as follows:

- We propose a new class of anonymity attacks, which can *separate* the flows through a mix. Further attacks can make use of the information about the separated flows and so be very effective in reducing anonymity.
- We use experiments to show that flow separation attacks are effective for both single mixes and mix networks.
- We analyze the effect of multicast/broadcast traffic on the flow separation attack. In contrast to intuition, our analysis and experiments show that the presence of multicast/broadcast traffic significantly helps the attacker to more precisely separate the flows.
- We discuss the possible use of flow separation attack in other anonymity network settings and pros and cons of counter-measures.

The remainder of this paper is organized as follows: Section 2 reviews the related work. Section 3 outlines our mix network model and the threat model. In Section 4, we introduce the flow separation attack. We will also describe the frequency spectrum matching that we will use to evaluate the quality of flow separation. The same method is used in the Flow Correlation Attack described in [4]. In Section 5 and 6, we use ns-2 simulation experiments to show the effectiveness of the flow separation attack. We evaluate the flow separation attack against a non-trivial mix network in Section 7. Section 8 discusses the application of flow separation attacks in different network settings and countermeasures against the flow separation attack. We conclude this paper in Section 9 and identify extensions of this work.

2 Related Work

Chaum [1] pioneered the idea of anonymity in 1981. Since then, researchers have applied the idea to different applications, such as message-based email

and flow-based low-latency communications, and they have invented new defense techniques as more attacks have been proposed. For anonymous email applications, Chaum proposed to use relay servers, called mixes, that re-route messages. Messages are encrypted to prevent their tracking by simple payload inspection.

Low-latency anonymity systems have been developed recently for the dominant flow-based traffic in the Internet. A typical example is Tor [5], the second-generation onion router, developed for circuit-based low-latency anonymous communication. It can provide perfect forward secrecy.

In addition to the traditional message-based anonymity attacks [6], a large number of flow-based anonymity attacks have been proposed. Examples are intersection attacks [7], timing attacks [2], Danezis' attack on continuous mixes [8], and the flow correlation attack [4]. The timing attack [2] uses time domain cross-correlation to match flows given the packet timestamps of the flow. Danezis' attack on the continuous mix [8] relies on the per-packet independent delay in the continuous mix in order to use convolution a a comparison measure and likelihood ratios to detect a flow in aggregate traffic. The flow correlation attack [4] employs statistical methods to detect TCP flows in aggregate traffic.

The flow separation attack proposed in this paper belongs to the class of flow-based anonymity attacks.

3 Models

3.1 Mix and Mix Network

A mix is a relay device for anonymous communication. A single-mix network can achieve a certain level of communication anonymity: The sender of a message attaches the receiver address to a packet and encrypts it using the mix's public key. Upon receiving a packet, the mix decrypts the packet using its private key. Different from an ordinary router, a mix usually will not relay the received packet immediately. Rather, it will attempt to perturb the flows through the Mix in order to foil an attacker's effort to link incoming and outgoing packets or flows. It does this, typically, in three ways: First, it re-encrypts the packet to foil attacks that attempt to match packets in the payload data domain. Then, it *re-routes* the packet to foil correlation attacks that rely on route traceback. Finally, it perturbs the flows in the time domain through *batching*, *reordering*, and *link padding*. Batching collects several packets and then sends them out in a *batch*. The order of packets may be altered as well. Both these batching techniques are important in order to prevent timing-based attacks. Different batching and reordering strategies are summarized in [4] and [6].

Most practical systems employ only a subset of these strategies, for different reasons. For example, Onion Router [9], Crowds [10], Morphmix [11], P5 [12], and Tor [5] do not use any batching and reordering techniques.

A network may consist of multiple mixes that are inter-connected by a network such as the Internet. A mix network may provide enhanced anonymity, as payload packets may go through several mixes so that if one mix is compromised, anonymity can still be maintained.

3.2 Threat Model

We assume a passive adversary, whose capabilities are summarized as follows:

1. The adversary observes a number of input and output links of a mix, collects the packet arrival and departure times, and analyzes them. This type of attack is *passive*, since traffic is not actively altered (by, say, dropping, inserting, and/or modifying packets during a communication session), and is therefore often difficult to detect. This type of attack can be easily staged on wired and wireless links [13] by a variety of agents, such as governments or malicious ISPs [14].
2. For simplicity of discussion, we assume a *global* adversary, i.e. an adversary that has observation points on all links between mixes in the mix network. While this assumption seems overly strong, it is not, as the attacker will naturally aggregate mixes for which it has no observation points into *super-mixes*.
3. We don't assume that the adversary has access to any per-flow information. That is, it may only observe aggregates of flows. This is in contrast to most anonymity attacks (e.g., [2, 4, 6],) which start from *a priori* flow information and attempt to identify the flow in an aggregate.
4. The adversary cannot correlate (based on packet timing, content, or size) an individual packet on an input link to another packet on an output link based on content and packet size. This is prevented by encryption and packet padding, respectively.
5. We focus on mixes operating as simple proxy. In other words, no batching or reordering is used. Link padding (with dummy packets) is not used either. This follows the practice of some existing mix networks, such as Tor [5].
6. Finally, we assume that the specific objective of the adversary is to identify the path of a flow in a mix network if there is some knowledge about the flow, or to determine a map of traffic directions in the mix network.

4 Flow Separation in Mix Networks

In this section, we will first define the problem in the context of blind source separation and then describe how to apply the flow separation method in a mix network.

4.1 Blind Source Separation

Blind source separation is a methodology in statistical signal processing to recover unobserved "source" signals from a set of observed mixtures of the signals. The separation is called "blind" to emphasize that the source signals are not observed and that the mixture is a black box to the observer. While no knowledge is available about the mixture, in many cases it can be safely assumed that source signals are independent. In its simplest form [15], the blind source separation model assumes n independent signals $F_1(t), \cdots, F_n(t)$ and n observations of mixtures $O_1(t), \cdots, O_n(t)$ where $O_i(t) = \sum_{j=1}^{n} a_{ij} F_j(t)$. The goal of blind source separation is to reconstruct the source signals $F_j(t)$ using only the observed data $O_i(t)$ and the assumption of independence among the signals $F_j(t)$. A very nice introduction to the statistical principles behind blind source separation is given in [15]. The common methods employed in blind source separation are minimization of mutual information [16, 17], maximization of nongaussianity [18, 19] and maximization of likelihood [20, 21].

4.2 Flow Separation as a Blind Source Separation Problem

In this paper, we define a *flow* as a series of packets that are exchanged between a pair of hosts. Typically, such a flow is identified by a tuple of source/destination addresses and port numbers. Similarly, we define an *aggregate flow* at the *link-level* to be the sum of the packets (belonging to different flows) on the link. We define the aggregate flow at *mix-level* as sum of packets through the same input and output port of a mix. Unless specified, otherwise the word "flow" in the remaining of this paper means "mix-level aggregate flow" for brevity.

We will show in this paper that, for the attacker who tries to break the anonymity of a mix, it is very helpful to *separate* the flows through the mix based on the observation of the link traffic. The separation of the flows through the mix can recover the traffic pattern of flows, which can be used in further attacks, such as the frequency spectrum matching attack described in Section 4.3 or the time domain cross-correlation attack [2].

In this paper, we are interested in the traffic pattern carried in the time series of packet counts during each sample interval T. For example, in Figure 1, the

attacker can get a time series $O_1 = [o_1^1, o_2^1, \cdots, o_n^1]$ of packet counts by observing the link between Sender S_1 and the mix. We use n to denote the *sample size* in this paper. The attacker's objective is to recover the packet count time series $F_i = [f_1^i, f_2^i, \cdots, f_n^i]$ *for each flow*. For the simplest case, we assume that (a) there is no congestion in mix and that (b) the time series can be synchronized. (We will relax both assumptions in later sections.) In the example of Figure 1, the time series F_1 is contained in both time series O_1 and O_3 i.e. $O_1 = F_1 + F_2$, $O_3 = F_1 + F_3$. For a mix with j input ports, k output ports and m mix-level aggregate flows, we can rewrite the problem in vector-matrix notation,

$$\begin{pmatrix} O_1 \\ O_2 \\ \vdots \\ O_{j+k} \end{pmatrix} = \mathbf{A}_{(j+k)\times m} \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_m \end{pmatrix} \tag{1}$$

where $\mathbf{A}_{(\mathbf{j+k})\times \mathbf{m}}$ is called *mixing matrix* in the blind source separation problem [3].

The flow separation can be solved using a number of blind source separation techniques. The rationale for blind source separation relies on the fact that the aggregate flows through a mix are independent from each other, since the aggregate flows are from different sources. Even the flows from a same host, such as F_1 and F_2, can be regarded as independent as they follow different paths and controlled by different sockets. This independence assumption is of course only valid as long as Sender S_1 is not heavily overloaded, since otherwise one flow would influence the other. Given the observations $O_1, O_2, \cdots, O_{j+k}$, blind source separation techniques estimate the independent aggregate flows $F_1, F_2, \cdots, F_m$ by maximizing the independence between estimated aggregate flows. In the following, we need to keep in mind that flow separation often is not able to separate individual flows. Rather, mix-level aggregates flows that share the links at the observation points form the minimum separable unit.

Issues about Blind Source Separation Basic blind source separation algorithms require the number of observations to be larger than or equal to the number of independent components. For flow separation, this means that $j+k \geq m$, where j and k denote the number of observations at the input and output of the mix, respectively, and m denotes the number of flows. Advanced blind source separation algorithms [22, 23] target over-complete bases problems and can be used for the case where $m > j + k$. But they usually require that m, the number of independent flows, be known. Since all the mix traffic is encrypted and padded, it is hard for the attacker to estimate m. In this paper, we assume that

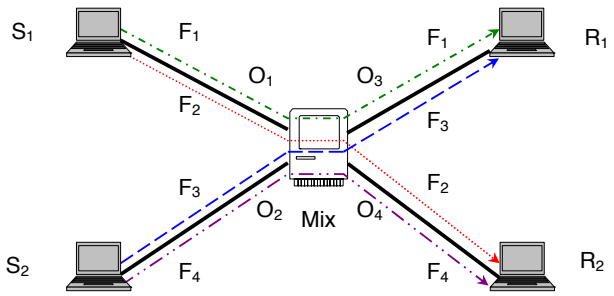


Fig. 1. An Example for Flow Model

$m = j + k$. The cost of this assumption is that some independent flows can not be separated, that is, they remain mixed. We will see that this is not a severe constraint, in particular not in mix networks where flows that remain mixed in some separations can be separated using separation results from neighboring mixes.

Unless there is multicast or broadcast traffic through the mix, the $j + k$ observations will have some redundancy, because the summation of all the observations on the input ports are equal to the summation of all the observations on the output ports. In other words, the row vectors of the mixing matrix are linearly dependent. Again, the cost of the redundancy is that some independent flows are not separated.

The flow estimation generated by blind source separation algorithms is usually a lifted, scaled version of the actual flow (of its time series, actually). Sometimes, the estimated flow may be of different sign than the actual flow. Both lifting and scaling do not affect the frequency components of the time series, and so frequency matching can be used to further analyze the generated data.

Furthermore, since the elements of the estimated mixing matrix are not binary, it is not straightforward to tell the direction of each aggregate flow. Some heuristic approach can be used, but we leave this to further research.

In the rest of this paper, we will show that the issues identified above can be largely solved with the use of appropriate frequency matching.

4.3 Frequency Matching

After the flows have been separated, a number of flow aggregates (some of them consisting of a single flow) have been determined to traverse the mix, each with a given time series of packet counts. It is not known, however, without appropriate post-processing, *on which links* each of these flows enters and leaves the mix.

Frequency spectrum matching has shown to be particularly effective to further analyze the traffic. The rationale for the use of frequency matching is fourfold: First, the dynamics of a flow, especially a TCP flow [24], is characterized by its periodicities. By matching the frequency spectrum of a known flow with that of the estimated flows obtained by blind source separation, we can identify the known flow with high accuracy. Second, frequency matching can easily remove the ambiguities introduced by the lifting and scaling in the estimated time series by removing the zero-frequency component. Third, frequency spectrum matching can also be applied on the mix-level aggregate flows, since the different frequency components in each individual flow can characterize the aggregate flow. Fourth, the low frequency components of traffic are often not affected by congestion as they traverse multiple switches and mixes. This is particularly the case for TCP traffic, where the frequency components are largely defined by the behavior at the end hosts. In summary, frequency spectrum analysis has excellent prerequisites to be highly effective.

Even if no information is available about individual flows, the attacker can easily determine if there is communication between two neighboring mixes. Matching the estimated aggregate flows through the neighboring mixes can give attackers more information, such as how many aggregate flows are going through the next mix. In a mix network, an aggregate flow through a mix may split into aggregate flows of smaller size, multiplex with other aggregate flows, or do both. By matching the estimated aggregate flows through neighboring mixes, the attacker can detect the split and multiplex. Based on the information gathered, the attacker can eventually get a detailed map of traffic in a mix network. In Section 7, we show a traffic map obtained from the aggregate flow matching.

The sample interval T is important to the frequency spectrum matching. The averaging effect of sample interval T on frequency spectrum matching results can be modeled as low-pass filtering. If we are matching TCP flows, it is important to select a proper sample interval to avoid filtering out interesting TCP frequency components such as round trip time (RTT), time-out frequencies. More details on selecting T and modeling of the effect of T can be found in [24].

In the following, we will be using frequency matching of the *separated* flows against the *actual* flows in the network to measure the accuracy of the flow separation. The rationale for this method is that a highly accurate flow separation will result in good matching with the component flows, whereas a poor separation will generate separated flows that can not be matched with the actual ones.

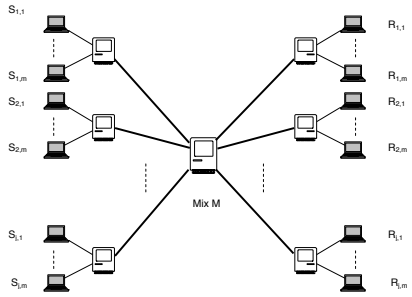


Fig. 2. Experiment Setup for Single Mix

5 Evaluation on Single Mix with Different Combinations of Traffic

In this section, we will evaluate the performance of flow separation for a single mix. We use the blind source separation algorithm proposed in [25] to separate the flows. The accuracy of separation will be measured using frequency matching with actual flows.

5.1 Experiment Setup

Figure 2 shows the experimental network setup for the single mix. We use ns-2 to simulate the network. The links in the figure are all of $10Mbit/s$ bandwidth and $10ms$ delay if not specifically mentioned otherwise. (Senders and receivers can be at large distances from the mix, potentially connecting through several routers and switches.) In the series of experiments in this section, the mix under study has two input ports and two output ports and four aggregate flows passing through the mix, as shown in Figure 1. We will study mixes with more than two ports in Section 6. Unless specified otherwise, we will use time observation intervals of $32second$ length and sample interval of $10ms$ length, resulting in time series of size $n = 3200$. Similar results were obtained for shorter observations as well.

5.2 Metrics

In the following, we will adopt two metrics to evaluate the accuracy of the flow separation. Both metrics are based on a comparison of the separated flows with the actual flows in the mix.

As first performance metric, we use *mean square error (MSE)*, a widely used performance criterion in blind source separation research. Let $F_A = [f_1^A, f_2^A, \cdots, f_n^A]$ represent the time series of the actual flow and $F_B =$

$[f_1^B, f_2^B, \cdots, f_n^B]$ represent the time series estimated by the blind source separation algorithm. To match the time series F_A with F_B, we first need to scale and lift F_B so that they have the same mean and variance.

$$F'_B = \frac{std(F_A)}{std(F_B)} \cdot (F_B - mean(F_B) \cdot [1, 1, \cdots, 1]) + mean(F_A) \cdot [1, 1, \cdots, 1] \ , \tag{2}$$

where $std(F)$ and $mean(F)$ denote the standard deviation and average of the time series F, respectively. The *mean square error* is defined as follows:

$$\varepsilon_{A,B} = \frac{\|F_A - F'_B\|^2}{n} \ . \tag{3}$$

Since the times series F_B can be a flipped version of F_A, we need to match F_A with $-F_B$ as well.

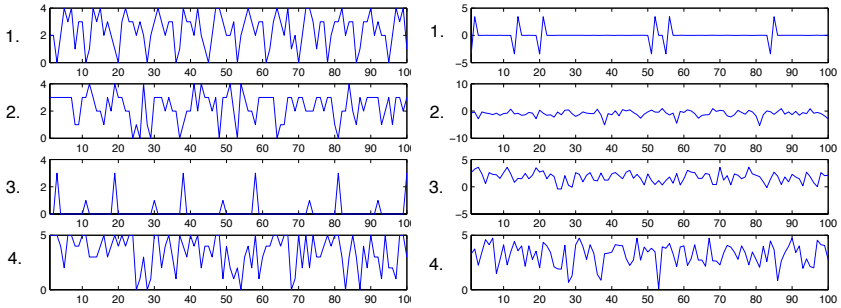
As the second metric, we use what we call *frequency spectrum matching rate*. We define the matching rate to be the probability that the separated flow F_B has the highest frequency spectrum cross-correlation with the actual flow F_A.

We note that, while the mean square error captures the accuracy of the separation in the time domain, the matching rate captures the effectiveness of the separation in the frequency domain.

5.3 Different Types of Traffic

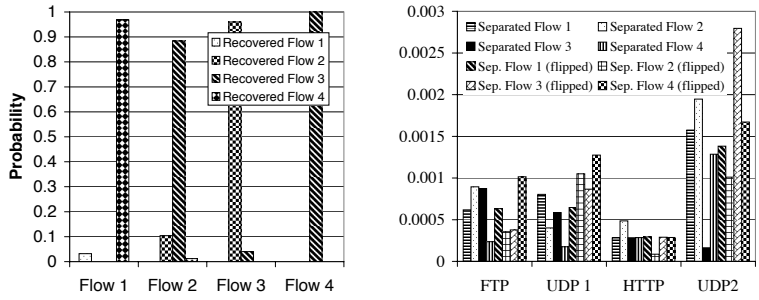
In this experiment the mix carries four aggregate flows: one FTP flow, one sequence of HTTP requests, and two on/off UDP flows. The parameters for the flows are as follows: Flow 1: FTP flow, with round trip time around $80ms$. Flow 2: UDP-1 flow, on/off traffic, with burst rate $2500kbit/s$, average burst time $13ms$ and average idle time $6ms$. Flow 3: HTTP flows, with average page size 2048 byte. Flow 4: UDP-2, on/off traffic with burst rate $4000kbit/s$, average burst time $12ms$ and average idle time $5ms$. All the random parameters for the flows are exponentially distributed. The flows are passing through the mix as shown in Figure 1.

Figure 3(a) shows portions of the actual time series and of 3(b) of the estimated time series. From these figures, it is apparent that the flipped version of the actual flow 3 (HTTP flows) is contained in the estimated flow 2. We also observe the resemblance between actual flow 1 (FTP flow) and estimated flow 4. Estimated flow 1 is clearly not close to any actual flows. This inability to fully separate the flows is caused by the redundancy contained in the observations, as described in Section 4.2.



(a) Actual Flows **(b) Separated Flows**

Fig. 3. Example of Flow Separation for Different Types of Traffic



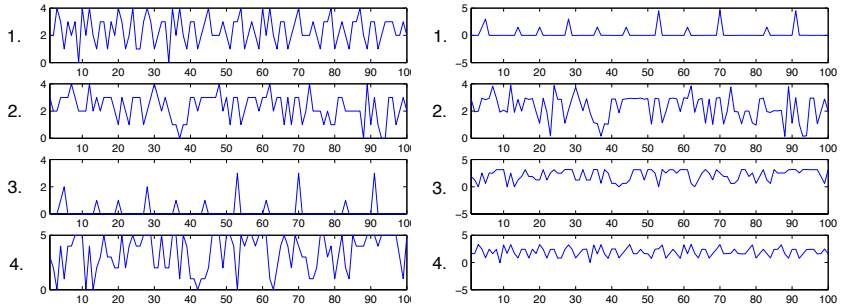
(a) Frequency Spectrum Matching Rate **(b) MSE**

Fig. 4. Performance of Flow Separation for Different Types of Traffic

Figure 4 shows the separation accuracy using the two metrics defined earlier. We note in Figure 4(b) that both the separated flow and its flipped time series is compared against the actual flows. Both metrics can identify the FTP flow, HTTP flows and one UDP flow. But the two metrics disagree on the other UDP flow. This is because of the redundancy in the observations, and the two UDP flows can not be separated. MSE fails for this case since it is designed for one-to-one flow matching while frequency spectrum matching is more suitable for matching of flows against aggregates. The latter case is more common in the context of flow separation.

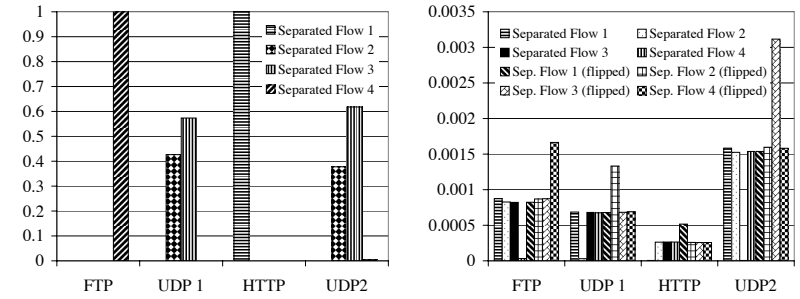
5.4 Different Types of Traffic with Multicast Flow

In this experiment, the flow UDP-1 in the previous experiment is multicast to both output ports.



(a) Actual Flows **(b) Separated Flows**

Fig. 5. Example of Flow Separation for Different Types of Traffic (with Multicast Traffic)

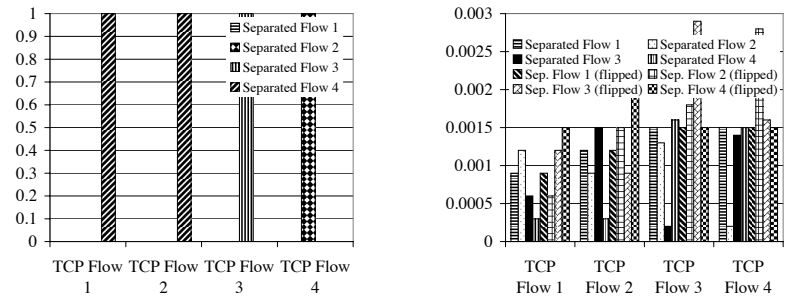


(a) Frequency Spectrum Matching Rate **(b) MSE**

Fig. 6. Performance of Flow Separation for Different Types of Traffic (with Multicast Traffic)

Portions of the actual flows and the estimated flows are shown in Figure 5. We observe the correspondence between the actual flows and estimated flows easily. In comparison with the previous experiment, we can conclude that multicast flows can help the flow separation. The reason is that in this experiment there is no redundant observation when the multicast flow is passing through the mix.

MSE performance metrics in Figure 6 identify the flows successfully. Frequency spectrum matching successfully determines the FTP and HTTP flows, but does not perform well on the UDP flows. This is because the two UDP flows have similar periods, and the periodical behavior is not strong for exponential on/off traffic.



(a) Frequency Spectrum Matching Rate **(b) MSE**

Fig. 7. Performance of Flow Separation for TCP-Only Traffic (without Multicast Traffic)

5.5 TCP-Only Traffic

Since most of the traffic in today's networks is TCP traffic, we focus on TCP traffic in the next series of experiments. All the flows in this experiment are FTP flows. To distinguish the flows, we vary the link delays between the sender and mix, with S_1 having $10ms$ link delay to the mix, and S_2 having $15ms$ delay.

Figure 7 shows the flow separation performance. Since there is no multicast traffic, the redundancy in observations results that TCP Flow 1 and TCP Flow2 are still mixed. But the flows are identified successfully, especially by the frequency spectrum matching method.

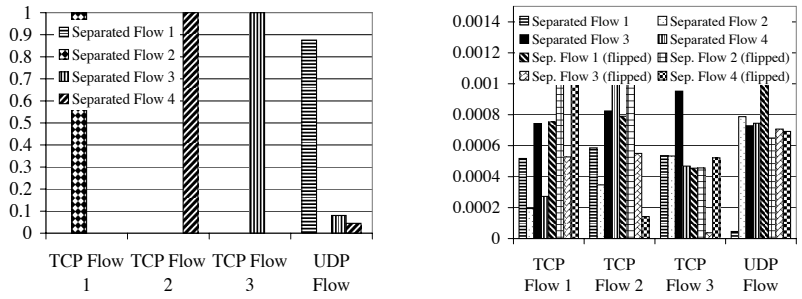
5.6 TCP-Only Traffic with Multicast Flow

In this experiment, we change one FTP flow in the previous experiment to a multicast UDP flow. The UDP flow is exponential on/off traffic with the same parameter as UDP-1 in the experiment of Section 5.3.

Figure 8 shows the flow separation performance. Similarly to the effect of multicast flow on different types of traffic, the four flows are separated completely since there are no redundant observations. We can also observe that the frequency spectrum method identifies the FTP flows successfully. But the performance on the exponential on/off UDP flow is not as good as FTP flows because exponential traffic flow's frequency signature is very weak.

6 Evaluation of Scalability of Flow Separation

We evaluated the scalability of flow separation in a series of experiments. We first increased the number of flows in mix-level aggregate flows (the number



(a) Frequency Spectrum Matching Rate **(b) MSE**

Fig. 8. Performance of Flow Separation for TCP-Only Traffic (with Multicast Traffic)

of aggregate flows remains constant), then increased the number of mix-level aggregate flows, and finally increased the number of ports per mix.

These experiments focus on a single mix. We use frequency spectrum matching results as performance metrics. Following are the observations for this series of experiments. (Refer to [26] for more details of the experiments and results.)

The performance of flow separation remains high when we increase the number of flows per mix-level aggregate flow, increase the number of mix-level aggregate flows, and increase the number of ports per mix. For example, with 20 flows per aggregate, the lowest matching for a 100Mbit/s link and 0.05 second sample interval is still above 65%.

We note that, as we increase the size of aggregate flows, the congestion caused by TCP flows leads to less accurate flow separation. However, this can be compensated by increasing the sample interval.

Increasing the number of aggregate flows may lead to some flows not being separable due to a shortage of observations. Nevertheless, the frequency matching rate remains high. The same applies to experiments on increasing numbers of ports per mix.

In summary, it can be safely said that blind source separation has a high performance in large systems as well.

7 Evaluation for Mix Networks

Flow separation can also be used in mix networks when assuming a global passive attacker. The attacker can do flow separation at each mix according to observations obtained at that mix. Then the attacker can correlate the separated aggregate flows to derive the traffic map of the whole mix network.

Flows	Path	Parameters	Throughput (packets/s)
1	$S_1 \rightarrow M_1' \rightarrow M_1 \rightarrow M_3 \rightarrow M_5 \rightarrow M_7 \rightarrow M_9 \rightarrow M_{11} \rightarrow M_5' \rightarrow R_1$	FTP	106.125
2	$S_2 \rightarrow M_1' \rightarrow M_1 \rightarrow M_4 \rightarrow M_5 \rightarrow M_8 \rightarrow M_9 \rightarrow M_{12} \rightarrow M_7' \rightarrow R_5$	FTP	100.791
3	$S_3 \rightarrow M_2' \rightarrow M_1 \rightarrow M_3 \rightarrow M_5 \rightarrow M_7 \rightarrow M_9 \rightarrow M_{11} \rightarrow M_6' \rightarrow R_3$	FTP	95.936
4	$S_4 \rightarrow M_2' \rightarrow M_1 \rightarrow M_4 \rightarrow M_5 \rightarrow M_8 \rightarrow M_9 \rightarrow M_{12} \rightarrow M_8' \rightarrow R_7$	FTP	91.541
5	$S_5 \rightarrow M_3' \rightarrow M_2 \rightarrow M_3 \rightarrow M_6 \rightarrow M_7 \rightarrow M_{10} \rightarrow M_{11} \rightarrow M_5' \rightarrow R_2$	FTP	87.531
6	$S_6 \rightarrow M_3' \rightarrow M_2 \rightarrow M_4 \rightarrow M_6 \rightarrow M_8 \rightarrow M_{10} \rightarrow M_{12} \rightarrow M_7' \rightarrow R_6$	FTP	83.858
7	$S_7 \rightarrow M_4' \rightarrow M_2 \rightarrow M_3 \rightarrow M_6 \rightarrow M_7 \rightarrow M_{10} \rightarrow M_{11} \rightarrow M_6' \rightarrow R_4$	FTP	80.483
8	$S_8 \rightarrow M_4' \rightarrow M_2 \rightarrow M_4 \rightarrow M_6 \rightarrow M_8 \rightarrow M_{10} \rightarrow M_{12} \rightarrow M_8' \rightarrow R_8$	FTP	77.357
9	$\rightarrow M_3 \rightarrow M_5 \rightarrow M_8 \rightarrow M_{10} \rightarrow$	Pareto	319.317
10	$\rightarrow M_3 \rightarrow M_6 \rightarrow M_8 \rightarrow M_9 \rightarrow$	Pareto	318.558
11	$\rightarrow M_4 \rightarrow M_5 \rightarrow M_7 \rightarrow M_{10} \rightarrow$	Pareto	321.806
12	$\rightarrow M_4 \rightarrow M_6 \rightarrow M_7 \rightarrow M_9 \rightarrow$	Pareto	323.36

Table 1. Flow Configuration

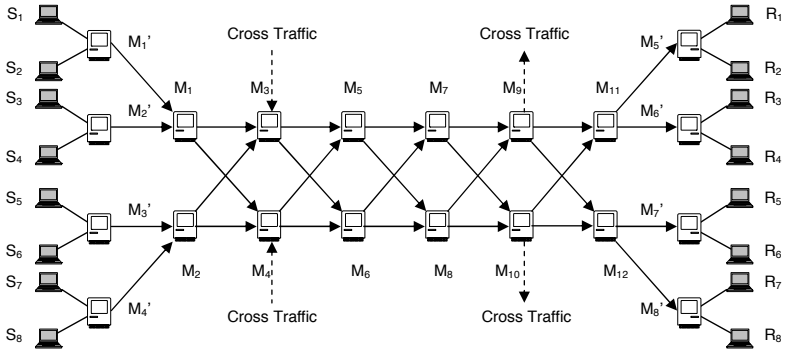


Fig. 9. Experiment Setup of Mix Network

7.1 Experiment Setup

Figure 9 shows the network setup in this experiment. Eight FTP flows from senders on the left side traverse the mix network. To distinguish these eight FTP flows, we incrementally add $5ms$ delay to the link connected to each sender. To simulate the cross traffic in the mix network, four larger aggregates of flows are added to the mix network. In order to mimic a self-similar network traffic in general [27], the high-volume cross traffic is Pareto distributed. The configuration of the flows is shown in Table 1.

In the center of the mix network, the traffic volume ratio between link-level aggregate traffic and each individual flow from senders is at least $7:1$. We assume the attacker can observe links connected to Mix $M_1, M_2, \cdots, M_{12}$. Thus, a flow originating from S_1 can take 2^6 possible paths.

7.2 Performance Metrics

To evaluate the performance of detecting a flow in the network, we introduce a network-level performance metrics, which is based on the entropy-based anonymity

degree proposed in [28, 29]. Suppose we are interested in flow F_x. The attacker can suspect the flow F_x taking a path P_i with probability p_i based on the information gathered from the anonymity attack on the mix network. Assuming there are h possible paths that can be suspected as the path taken by the flow F_x, we define the anonymity degree as

$$D = -\sum_{i=1}^{h} p_i \log_2 p_i \quad . \tag{4}$$

Suppose a flow originating from S_1 in Figure 9 is suspected to use each of the 2^6 possible paths with equal probability. Then the anonymity degree is $D = 6bit$.

7.3 Performance

Figure 10 shows the mean value of the cross correlation using frequency spectrum matching for the first four FTP flows and the separated flows recovered from Mix $1-12$. The cross-correlation values less than 0.1 are marked as white. Please note that the cross-correlation values between separated flows recovered from the same mix are also marked as white. This includes the cross-correlation (auto-correlation) for the same separated flow or FTP flow.

From the cross-correlation map shown in Figure 10, we can easily figure out the traffic direction in the mix network.

Figure 11 shows an algorithm to detect a flow say F_x in the network based on flow separation and frequency spectrum matching. The main idea behind the algorithm is to first use the aggregate flow F_{tmp}, which is determined to be on the path previously to match the separated flows on the neighboring mixes. The threshold $threshold\_1$ is used to determine the Candidate array which includes the separated flows that have some components of the identified aggregate flow F_{tmp}. Then we match the flow F_x with the separated flows in the Candidate array to determine the most closely matching flow on the next hop. The process continues until the correlation is too weak, which is determined by the threshold $threshold\_2$. Thresholds $threshold\_1$ and $threshold\_2$ can be determined by online learning based either on data collected by attacker or on some heuristics setting. The algorithm uses dynamic programming. It can be further improved by considering more possible routes and select the ones that have have the largest overall possibilities.

We set the Thresholds $threshold\_1$ to zero and $threshold\_2$ to 0.1 heuristically. The result is based on the observations of 32 seconds of traffic. Our data indicates that similar results can be obtained with significantly smaller observation intervals. Our results indicate that the attack is very effective. In most

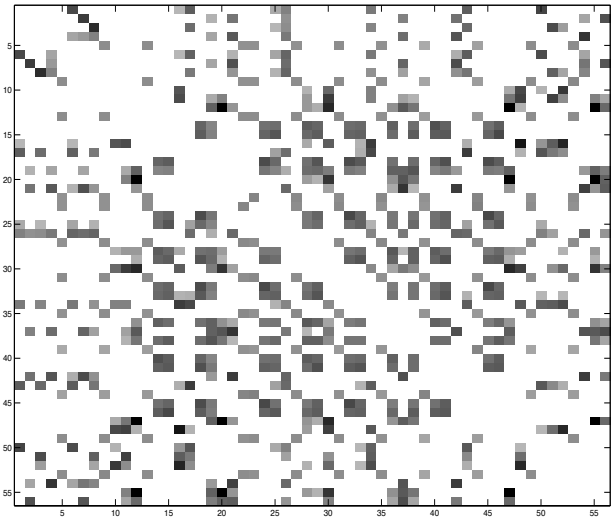


Fig. 10. Mean Value of Cross Correlation between Four FTP flow and Estimated Flows

cases, the anonymity was reduced from 6 bit to zero bit, while in one case, it was reduced form 6 bit to about 0.5 bit.

8 Discussion

In this paper, we focus on simple proxy mixes because of their popular use in practical anonymity systems. But flow separation can also be used to attack mixes that use other batching strategies, such as the timed mix. Timed mixes batch the packets and release them every t seconds. So packets arriving at a timed mix in one batch will depart in the next batch. In turn, the noise in the observation at the output ports caused by queuing delays is zero as long as the timed mix is not congested. This helps the flow separation attack.

Flow separation attacks can also be used in wireless ad-hoc anonymity networks such as ANODR [30]. ANODR assumes that the packets in wireless anonymity network are encrypted and sender uses broadcast to avoid MAC address disclosure. Flow separation is more powerful in wireless anonymity network for two reasons: First, the passive attacker can easily get a larger number of observations in a wireless setting than in wired network by simply placing more wireless receivers in the wireless anonymity network. In order to eliminate redundant observations, the locations of these wireless receivers will depend on the transmission range of the wireless transmitters in the wireless anonymity network. Second, the attacker can execute a flow separation attack not only on

```
F_tmp=F_x
M_tmp=M_x
while (mix M_tmp is not a dead-end) do {
        empty Candidate array
        for each mix M_i connected to M_tmp {
                for each flow F'_y separated by flow separation attack on M_i {
                        matching(F_tmp, F'_y)=Cross-correlation coefficient of the frequency
                        spectrums of F_tmp and F'_y
                        if  matching(F_tmp, F'_y)> threshold_1
                                record (F'_y, M_i) into array Candidate
                }
        }
        find the element (F'_max, M_max) in candidate array,  so that
        matching(F_x, F'_max) ≥matching(F_x, F'_y), for any F'_y in Candidate array
        if  matching(F_x, F'_max) <threshold_2
                break
        F_tmp=F'_max
        M_tmp=M_max
        record M_max as a mix on the flow path
}
```

Fig. 11. Flow Detection Algorithm

the basis of packet count time series but also on the physical strength of the wireless signal.

The countermeasures to flow separation attacks are intuitive.

- Padding the links so that the observations obtained by the passive attacker are identical, or at least mostly redundant.
- Use pool-mix like batching strategies. Pool mixes fire packets with a certain probability p. If the probability p is small enough, the aggregate flows at the output ports can be significantly different from aggregate flows at the input ports. Adding noise in the passive attacker's observations can degrade the performance of flow separation attacks. But the cost will be increased packet transfer latency and lower throughput, especially for TCP traffic.
- Increase the dependency among flows by adding dependent dummy traffic flows to the mix-level aggregate flows.
- Padding each aggregate flow so that the distribution of the packet count is Gaussian. Most blind source separation algorithms fail when the signals mixed are Gaussian distributed. But different classes of blind source separation algorithm that make use of the time structure of the signals can still separate the flows e.g., [31, 32].

In general, it can be said that blind source separation algorithms coping with noisy delayed signals, over-complete base problems are still active research

topics in blind source separation research. Flow separation attacks will be more powerful when more advanced algorithms become available.

9 Conclusion and Future Work

Traditional flow-based anonymity attacks often do not perform well in large systems, either because they rely on the observability of some flow data before the flow enters the mix network, or because they do not cope well with with large flow aggregates, which give raise to low signal-to-noise ratios. It would be therefore helpful to be able to *pre-condition* the traffic analysis data in order to reduce the size of flow aggregates by separating the flows.

We proposed a new anonymity attack, called the *flow separation attack* that can be used either alone or as a pre-conditioner, in conjunctions with other attacks, to significantly reduce the effectiveness of anonymous communication systems. We base flow separation on blind source separation, which is widely used to recover individual signals from mixtures of signals. Our experiments show that the anonymity attack is effective and scalable. With the aid of further attacks, such as frequency spectrum matching, flow separation can be used to detect the path taken by a flow in a mix network. Flow separation can also be used to simply recover the traffic map of the anonymity network. We discuss the possible usage of flow separation attack in different anonymity network settings, and we elaborate on criteria for its countermeasures.

Our future work will focus on the usage of the attack in the wireless and ad-hoc anonymity networks. We also planning to analytically model the effectiveness of the attack.

References

1. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM **4** (1981)
2. Levine, B.N., Reiter, M.K., Wang, C., Wright, M.K.: Timing attacks in low-latency mix-based systems. In Juels, A., ed.: Proceedings of Financial Cryptography (FC '04), Springer-Verlag, LNCS 3110 (2004)
3. Jutten, C., Herault, J.: Blind separation of sources, part 1: an adaptive algorithm based on neuromimetic architecture. Signal Process. **24** (1991) 1–10
4. Zhu, Y., Fu, X., Graham, B., Bettati, R., Zhao, W.: On flow correlation attacks and countermeasures in mix networks. In: Proceedings of Privacy Enhancing Technologies workshop (PET 2004). (2004)
5. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium. (2004)
6. Serjantov, A., Dingledine, R., Syverson, P.: From a trickle to a flood: Active attacks on several mix types. In Petitcolas, F., ed.: Proceedings of Information Hiding Workshop (IH 2002), Springer-Verlag, LNCS 2578 (2002)

7. Danezis, G., Serjantov, A.: Statistical disclosure or intersection attacks on anonymity systems. In: Proceedings of 6th Information Hiding Workshop (IH 2004). LNCS, Toronto (2004)
8. Danezis, G.: The traffic analysis of continuous-time mixes. In: Proceedings of Privacy Enhancing Technologies workshop (PET 2004). LNCS (2004)
9. Goldschlag, D., Reed, M., Syverson, P.: Onion routing for anonymous and private internet connections. Communications of the ACM (USA) **42** (1999) 39–41
10. Reiter, M., Rubin, A.: Crowds: Anonymity for web transactions. ACM Transactions on Information and System Security **1** (1998)
11. Rennhard, M., Plattner, B.: Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In: Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2002), Washington, DC, USA (2002)
12. Sherwood, R., Bhattacharjee, B., Srinivasan, A.: P5: A protocol for scalable anonymous communication. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy. (2002)
13. Howard, J.D.: An analysis of security incidents on the internet 1989 - 1995. Technical report, Carnegie Mellon University Dissertation (1997)
14. F.B.I: Carnivore diagnostic tool. http://www.fbi.gov/hq/lab/carnivore/carnivore2.htm (2003)
15. Cardoso, J.: Blind signal separation: statistical principles. **9** (1998) 2009–2025 Special issue on blind identification and estimation.
16. Comon, P.: Independent component analysis, a new concept? Signal Process. **36** (1994) 287–314
17. He, Z., Yang, L., Liu, J., Lu, Z., He, C., Shi, Y.: Blind source separation using clustering-based multivariate density estimation algorithm. IEEE Trans. on Signal Processing **48** (2000) 575–579
18. Hyvärinen, A.: Fast and robust fixed-point algorithms for independent component analysis. IEEE Transactions on Neural Networks **10** (1999) 626–634
19. Hyvärinen, A., Oja, E.: A fast fixed-point algorithm for independent component analysis. Neural Comput. **9** (1997) 1483–1492
20. Gaeta, M., Lacoume, J.L.: Source separation without prior knowledge: the maximum likelihood solution. In: Proc. EUSIPCO'90. (1990) 621–624
21. Pham, D.T., Garrat, P., Jutten, C.: Separation of a mixture of independent sources through a maximum likelihood approach. In: Proc. EUSIPCO. (1992) 771–774
22. Hyvärinen, A., Inki, M.: Estimating overcomplete independent component bases for image windows. J. Math. Imaging Vis. **17** (2002) 139–152
23. Hyvärinen, A., Cristescu, R., Oja, E.: A fast algorithm for estimating overcomplete ICA bases for image windows. In: Proc. Int. Joint Conf. on Neural Networks, Washington, D.C. (1999) 894–899
24. Zhu, Y., Fu, X., Graham, B., Bettati, R., Zhao, W.: Correlation attacks in a mix network. Texas A&M University Computer Science Technical Report (2005)
25. Cruces-Alvarez, S.A., Cichocki, A.: Combining blind source extraction with joint approximate diagonalization: Thin algorithms for ICA. In: Proc. of the Fourth Symposium on Independent Component Analysis and Blind Signal Separation, Nara, Japan (2003) 463–468
26. Zhu, Y., Bettati, R.: Unmixing mix traffic. Texas A&M University Computer Science Technical Report (2005)
27. Park, K., Willinger, W.: Self-similar network traffic: An overview (1999)
28. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In Dingledine, R., Syverson, P., eds.: Proceedings of Privacy Enhancing Technologies Workshop (PET 2002), Springer-Verlag, LNCS 2482 (2002)

29. Diaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In Dingledine, R., Syverson, P., eds.: Proceedings of Privacy Enhancing Technologies Workshop (PET 2002), Springer-Verlag, LNCS 2482 (2002)
30. Kong, J., Hong, X.: Anodr: anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In: MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing, ACM Press (2003) 291–302
31. Tong, L., Liu, R.W., Soon, V.C., Huang, Y.F.: Indeterminacy and identifiability of blind identification. Circuits and Systems, IEEE Transactions on **38** (1991) 499–509
32. Molgedey, L., Schuster, H.G.: Separation of a mixture of independent signals using time delayed correlations. Physical Review Letters **72** (1994) 3634–3637