Synchronous Batching: From Cascades to Free Routes

Roger Dingledine[1], Vitaly Shmatikov[2], and Paul Syverson[3]

[1] The Free Haven Project (arma@freehaven.net)
[2] SRI International (shmat@csl.sri.com)
[3] Naval Research Lab (syverson@itd.nrl.navy.mil)

Abstract. The variety of possible anonymity network topologies has spurred much debate in recent years. In a synchronous batching design, each batch of messages enters the mix network together, and the messages proceed in lockstep through the network. We show that a synchronous batching strategy can be used in various topologies, including a free-route network, in which senders choose paths freely, and a cascade network, in which senders choose from a set of fixed paths. We show that free-route topologies can provide better anonymity as well as better message reliability in the event of partial network failure.

1 Introduction

Modern deployed mix networks, including Mixmaster [21] and its successor Mixminion [8], are subject to partitioning attacks: a passive adversary can observe the network until a target message happens to stand out from the others [3], and an active adversary can manipulate the network to separate one message from the others via blending attacks [24]. Berthold et al. argue [3] that partitioning opportunities arise because the networks use a *free-route* topology—one where the sender can choose the mixes that make up her message's path. They suggest instead a *cascade network* topology, where all senders choose from a set of fixed paths through the mix network.

In this paper we argue that the cascade design resolves these attacks because it uses a *synchronous batching* strategy, not because it uses a particular network topology. We show that synchronous batching prevents these attacks even when free routes are used. Further, we explore three topologies with synchronous batching—cascades, stratified (a restricted-route hybrid topology), and free-route—and find that the free-route network provides the highest expected anonymity as well as the best robustness to node failure.

In Section 2 we describe the synchronous batching model. Section 3 relates previous work to synchronous batching, including a response to each of the arguments from [3]. Section 4 presents the three topologies, and Section 5 describes their entropy (average anonymity the sender expects from the network). We use a model checker to compute entropy for networks with 16 nodes: we present our results and assess the assumptions behind them in Section 6. Section 7 considers other metrics such as bandwidth requirements, latency, and robustness.

2 Synchronous batching

Chaum proposed hiding the correspondence between sender and recipient by wrapping messages in layers of public-key cryptography, and relaying them through a path composed of *mixes* [4]. Each mix in turn decrypts, delays, and re-orders messages, before relaying them toward their destinations.

A mixnet design groups messages into batches and chooses paths; its design choices affect the degree of anonymity it can provide [24]. We might define ideal anonymity for a mixnet to be when an attacker can gain no information (beyond prior knowledge) about the linkage between messages entering and leaving the network, other than that the maximum time between them is equal to the maximum network latency.

This ideal is not achieved by protocols like Mixminion that use locally computed random delays: if the maximum latency of such a network is t, the probability that an output message corresponds to a particular input message might be considerably higher than for other messages that have entered over that time. (In principle, because of its pool mode, a message's maximum latency could be infinite, but that's not a significant improvement in practice: if the probability of a given latency t drops off exponentially with t, then so does the probability that a message leaving the network could have been sent that long ago [23].) Also, because Mixminion is both *asynchronous* (messages can enter and leave the network at any time) and uses free routes, it is subject to the attacks from [3] described in Section 3.2 below.

A network that uses *synchronous batching* has a fixed *batch period*, t_{batch}, which is related to the maximum desired latency, for example 3 hours. Messages entering the network in each batch period are queued until the beginning of the next period. They are then sent through the mixnet synchronously, at a rate of one hop per *hop period*. All paths are a fixed length ℓ hops, so that if no messages are dropped, the messages introduced in a given batch will progress through their routes in lockstep, and will all be transmitted to their final destinations ℓ hop periods later. Each layer of a message, once decrypted, specifies the hop period in which it must be received, so that it cannot be delayed by an attacker.

The *width* w of a mixnet using synchronous batching is the number of nodes that simultaneously process messages from a given batch in each hop period. (If this is not constant, we can still talk about the maximum, minimum, and mean width.) When $w = 1$, we have a cascade. The latency is between ℓt_{hop} and $t_{\text{batch}} + \ell t_{\text{hop}}$, depending on when the message is submitted. We might set $t_{\text{hop}} < t_{\text{batch}}/\ell$, so the latency is at most $2t_{\text{batch}}$, independent of the path length. Thus the entire batch is processed and delivered before the next batch enters the network. Under this constraint, we can give nodes the maximum opportunity to make use of the available bandwidth, and the best chance at delivery robustness, by setting $t_{\text{hop}} \simeq t_{\text{batch}}/\ell$.

3 Related work

3.1 Synchronous batching timing model and protocol

Dingledine et al. present in [11] a mix network that uses synchronous batching. We refer to that paper for a detailed discussion of the timing model, how to handle loosely synchronized clocks, and the step-by-step instructions for senders and mixes to use the network and judge whether messages have arrived on time.

That paper also describes a receipt and witness system by which senders and mixes can prove that a given mix failed to pass on or accept a given message. These receipts allow a reputation system: senders can recognize which nodes tend to drop messages, and avoid them in the future.

3.2 The Disadvantages of Free Mix Routes

Berthold et al. argue [3] that cascades are safer than free-route mix networks against a strong adversary who watches all links and controls many of the mixes. We consider each of their attacks below and find in each case that the arguments of [3] do not apply if the free-route network is synchronous. Indeed, against some of the attacks a free-route network is much stronger than the cascade network.

Position in mix route: This attack partitions messages that go through a given honest node based on how many hops each message has travelled so far. If the adversary owns all other nodes in the network, he can distinguish messages at different positions in their path (say, one has traversed two mixes already, and another has traversed three), and thus learn the sender and recipient of each message. The authors note: *Eventually, a message is only unobservable in that group of messages which have this mix on the same routing position.* But in the synchronous design, that's not a problem because this group is large (if only one mix is trustworthy, $1/w$ of all messages in the batch). They conclude: *If only one mix of a route is trustworthy, then the achievable anonymity is distinctly lower in a mix network compared to a synchronously working mix cascade.* The actual conclusion should be: If only one mix of a route is trustworthy, then the achievable anonymity for a given topology is distinctly lower in an asynchronous mixnet than in a synchronous mixnet.

Determining the next mix: An adversary owning most nodes in the network can attack the honest mixes: he can link senders to messages entering honest mixes, and he can link receivers to messages exiting honest mixes. Thus the target messages will only be mixing with other messages that enter the mix node at that time, and not with other messages elsewhere in the network. Even if senders use the same path for multiple messages, the authors point out that *the batches always generate different anonymity groups.* Again, the important property is whether the network uses synchronous batching, not whether it uses free routes. In a synchronous batching design, all messages in a batch exit the network together after the last hop, so messages cannot be partitioned based on when they enter or exit the network.

Probability of Unobservability: The authors explain that the cascade topology optimizes for the case that only one mix node is honest. They compare a 4-node cascade (with 3 compromised nodes) to a 20-node free-route mix network (with 75% compromised nodes), and find that whereas the cascade provides complete protection, a user choosing four nodes in the free-route network has a non-trivial chance of picking an entirely compromised path. But this is a false comparison. A better comparison would consider either a free-route mix network with 4 nodes, or a network of five $\ell = 4$ cascades—so the cascade network also has a chance of fully-compromised paths. In Section 6 we show that while each cascade in a cascade network of width w only mixes $1/w$ of the messages from the batch, a free-route network can mix all the messages from the batch and thus achieves significantly stronger anonymity even with 75% compromised nodes.

Active Attacks: The authors discuss an active attack called a trickle attack [17], wherein the adversary prevents legitimate messages from entering the batch, or removes some messages from the batch, so he can more easily trace Alice's message. To make the attack less overt, he can send his own messages into the batch, or replace the messages already in the batch with his own messages. These attacks where the adversary *blends* his messages with Alice's message threaten both synchronous-batching and asynchronous-batching networks in all topologies, and a complete solution that is practical is not known [24]. The authors of [3] present some approaches to mitigating this attack in a cascade environment, but a variety of other approaches have been developed that also work in a free-route environment. We discuss them next. Other active attacks are described in Section 7.3.

3.3 Blending attacks

Active attacks where the adversary targets a message by manipulating the other messages in the system are a widespread problem in mix-based systems. Solutions fall into three categories: attempts to *prevent* the attack, attempts to *slow* the attack, and attempts to *detect and punish* the attacker.

One prevention technique requires each sender to acquire a *ticket* for each mix in his path before joining a given batch (the senders receive blinded tickets [5] so the mixes cannot trivially link them to their messages). Mixes ensure their messages come from distinct senders, so Alice can expect good mixing at each honest node in her path [1]. For cascades this approach is clearly efficient because Alice only needs tickets for her chosen cascade [3], but her anonymity set is still limited to that one cascade. We conjecture that other topologies can give equivalent anonymity while only obtaining tickets from a fraction of the mixes, but we leave that analysis to future work. A bigger problem with the ticket scheme, however, is the feasibility of requiring all users to register with the mixes: it is hard to imagine that attackers can be excluded from being registered in an open network [13]. Other prevention techniques use complex cryptography to provide *robustness* [18] — messages are only delivered if a threshold of the mixes agree that the batch has been properly processed.

Techniques to slow the blending attack are generally designed for asynchronous mix networks. In Mixmaster and Mixminion, the goal of the batching algorithm is to hide from the adversary when an outgoing message entered the mix. Mixes 'pool' some messages from previous batches, to try to mix them as far back as possible. These approaches force the adversary to spend more time and messages on the attack [24]. Some designs allow a pool mix to commit to its choice of randomness to allow verifying its behavior [15]. Link encryption, as well as Babel's *inter-mix detours* [17] and early Onion Routing's *loose routing* [16], aim to block a limited adversary from knowing when his message has exited a mix. This also complicates blending because even the sender cannot always recognize a message he created. In stop-and-go mixes [20], each sender specifies a time window for each mix in his path: as with synchronous batching designs, messages arriving outside the time window are dropped, so the attacker cannot arbitrarily delay messages without destroying them.

Other approaches aim to detect and deter misbehavior. Chaum suggests allowing each sender to examine the output of each mix [4], but this approach scales poorly. Danezis and Sassaman propose a 'heartbeat' dummy scheme [9] for asynchronous pool mix networks: dummies are sent from a node in the network back to itself, creating an early warning system to detect if the adversary is launching a blending attack. *Reliability* mechanisms aim to improve a sender's long-term odds of choosing a mix path with well-behaving nodes. The witness-and-receipt system in [11] provides such a reputation system for synchronous-batching networks. Another reputation system for cascades [12] allows mixes to send test messages into the network to detect misbehavior. Finally, randomized partial checking [19] allows each mix to show evidence of its correctness by revealing a pseudo-randomly selected subset of its input-output relationships, while the mix network as a whole still protects linkability with high probability.

Clearly much work has been done to address blending attacks. Each topology seems to have some plausible partial solutions.

4 Threat model and mixnet topologies

Our analysis considers a slight variant on the traditional powerful adversary who observes globally and controls a fraction of the nodes [3]. We assume the adversary compromises nodes at random rather than in a targeted fashion (see Section 6.1 for more discussion on this point). Along with being able to control some of the nodes, our adversary can observe messages from senders to the mixnet and from the mixnet to receivers, but our initial analysis assumes he cannot observe the links between honest nodes in the mixnet (in Section 6.4 we argue that with high probability, observing these links will not yield much new information anyway). This paper only examines sender anonymity, though many of the advantages of synchronous batching may carry over to receiver anonymity.

We assume that selective forwarding will be discovered, and either the attack will be prevented or the malfunctioning node will be removed from the network (see Section 3.3). We address the attack of intersecting individual batches in

Section 3.2 (under "determining the next mix"), but unsurprisingly, we leave the long-term intersection attack [2, 7] unsolved. Further active attacks to degrade anonymity are described in Section 7.3.

We analyze a 16 node mixnet where all messages follow a four node path. Besides being a tractable size for analysis, 16 nodes also approximates deployed mixnets. (Mixminion currently has between 20 and 30 active nodes.) One might argue that a 4 node mixnet gives better security, because all messages are mixed together in any topology. We assume a larger network is needed because 1) the bandwidth of a single node may be insufficient to handle all the traffic; 2) a single path may not include as many choices for jurisdiction as some users want; and 3) a single path is not very robust, either to network attacks or to nature.

Messages proceed through the network in *layers*; all the nodes in a layer process messages of one mixnet batch at the same time. In general we describe networks as $w\text{x}\ell$, where w is the number of nodes at each layer and ℓ is the number of nodes in a path. We consider three basic topologies: a 4x4 cascade mixnet in which all messages pass through four cascades of length four; a 4x4 *stratified* mixnet, in which all messages pass through four layers of disjoint nodes such that messages may pass from any node at one layer to any node at the next layer; and a 16x4 free-route mixnet, in which all nodes may receive messages at all layers. Note that because free-route nodes are reused, '16x4' does not mean 64 nodes. Examples of the three topologies are illustrated below.

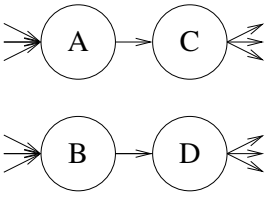

Fig. 1. A 2x2 cascade mix network (4 nodes)

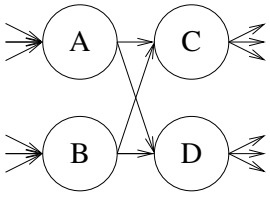

Fig. 2. A 2x2 stratified network (4 nodes)

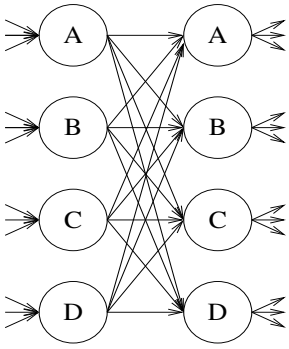

Fig. 3. A 4x2 free-route mix network (4 nodes)

5 Modeling Methodology

The basic model underlying our comparative study of mix network topologies is *mixing as probabilistic permutation*. At the cost of a few simplifying but reasonable assumptions about distribution of message traffic in the network, we obtain a tractable Markov chain model, and use a fully automated probabilistic model checking technique to compute probability distributions for different net-

work topologies and configurations. We use *entropy* of each topology's respective distribution as our comparison metric, in the spirit of [10, 23].

5.1 Mixing as permutation

Consider a single batch of N messages entering the mix network together. We can view each message $m_1, \ldots, m_N$ as occupying a certain position in a (virtual) input array of length N. Suppose the adversary targets a particular message m in position i. Without loss of generality, assume that $i = 1$ (we can always re-number the input array so that the targeted message is in the first slot).

Having passed the network, all N messages re-appear and may be observed by the adversary again. Of course, if some of the network nodes have been compromised by the adversary, the adversary will have access to their observations, too. Let $m'_1, \ldots, m'_N$ be the (virtual) output array. Due to the mixing performed by the network, it may or may not be the case that $m'_i = m_i$, *i.e.*, the messages have been probabilistically permuted by the network. We will refer to the discrete probability distribution $\mathsf{p}_1 \ldots \mathsf{p}_N$, where $\mathsf{p}_i = \mathit{Prob}(m'_i = m)$, as the *mixing distribution* of the network. Informally, each p_i is the probability that the targeted message m re-appears in the ith position of the output buffer.

In our basic model, we assume that the network doesn't lose messages (this restriction is not critical and may be relaxed, if necessary). Therefore, $\sum_{1 \leq i \leq N} \mathsf{p}_i = 1$, and p_i form a proper discrete probability distribution. Following [23], we calculate *entropy* of this distribution as

$$\mathcal{E} = - \sum_{1 \leq i \leq N} \mathsf{p}_i \log_2(\mathsf{p}_i)$$

Very informally, entropy is a measure of "randomness" in a distribution. Other things being equal, network topologies that provide mixing distributions associated with higher entropy values are considered preferable.

5.2 Overview of the model

We use the standard techniques of probabilistic verification and model the mix network as a discrete-time Markov chain. Formally, a *Markov chain* consists of a finite set of states S, the initial state s_0, the transition relation $T : S \times S \to [0, 1]$ such that $\forall s \in S \ \sum_{s' \in S} T(s, s') = 1$, and a labeling function.

In our model, the states of the Markov chain will represent the position of the targeted message m in the (virtual) buffer of N messages as m moves through the network. The initial state s_0 corresponds to the message being in the first slot of the input array prior to entering the mix network. Every probabilistic state transition $s \to s'$ is associated with m passing through a single mix within the network. Intuitively, s can be interpreted as m's position before passing through the mix, and s' as its position afterwards.

For the purposes of computing the mixing distribution p_i, we are interested in deadlock states, *i.e.*, those corresponding to the situation in which m has

passed through all mixes in its path and exited the mix network with no further transitions possible. Suppose a special predicate *done* is true in such states. Then p_i is simply $Prob[\mathcal{U}(s = i \ \wedge done)]$ evaluated in the initial state s_0. (Informally, formula $\mathcal{U}\varphi$ holds if φ eventually becomes true.)

We use a probabilistic model checker called PRISM [14] to compute these probabilities automatically. We omit the details of the underlying model checking algorithms; a detailed explanation of how probabilistic model checking is used to analyze randomized routing protocols can be found in [25].

5.3 Single-mix model

Consider a single mix receiving a batch of K messages, including the targeted message m. Assume an uncompromised mix that collects all K messages before distributing them to their respective destinations. In this case, the mixing performed by the mix can be interpreted as permutation in a virtual buffer of size K. In particular, the targeted message m appears in any of the K output positions with equal probability after passing through the mix. Therefore, each honest mix can be modeled by a simple Markov chain as below (recall that state s represents the current position of message m, and let t be the sequential number of the current hop). However, the compromised mix performs no mixing at all, and thus does not change the position of any message it processes.

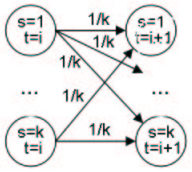

Fig. 4. Model of a good mix

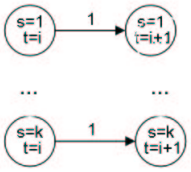

Fig. 5. Model of a bad mix

5.4 Network model

We consider several mix network topologies, and compare them under various assumptions about the density of hostile mixes in the network. Instead of assuming a fixed number of hostile mixes, in each scenario we will assume a fixed *probability* that a randomly selected mix is hostile.

For each topology, the behavior of a single node is modeled as in Section 5.3. The main difference between topologies is how the targeted message moves through the network, resulting in different mixing distributions $\mathsf{p}_1 \ldots \mathsf{p}_N$.

We assume the adversary observes the edge of the network and thus knows the first mix chosen by the targeted message—so the randomness of mix selection is ignored for the first hop. Formally, we make probability p_i conditional on selection of a particular first mix. Instead of computing $Prob[\mathcal{U}(s = i \ \wedge done)]$, we compute $Prob[\mathcal{U}(s = i \ \wedge done \mid \text{mix } x \text{ was selected as entry point})]$.

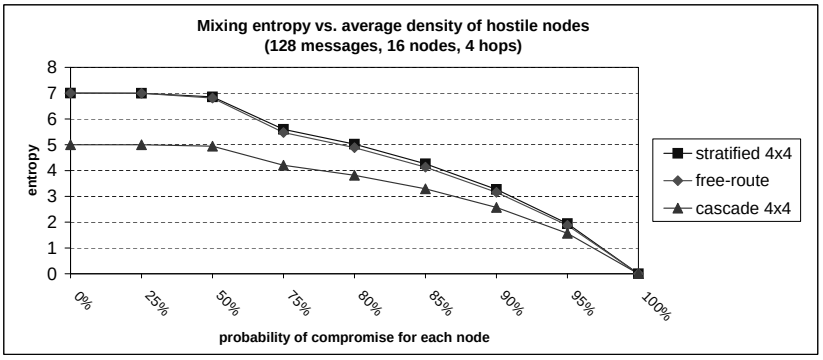

Fig. 6. Entropy vs probability of compromise for each node (16 nodes)

Note that we must consider two sources of uncertainty. The first is the distribution of compromised nodes in the network, which we address by assuming a fixed probability that any given node is bad. Thus we are calculating *prior* distributions—effectively the average of all possible occurrences of compromised nodes in the network. (In contrast, [10, 23] consider *posterior* distributions, where certain nodes are known to be bad). The second uncertainty is the users' selection of message routes, which we address by treating the message load on each internal link within the network as exactly equal to the statistically expected load given a particular network topology. This assumption is approximated with very high probability when the number of messages in a single batch is significantly higher than the number of network nodes (see Section 6.4 for discussion).

Intuitively, suppose there are four mixes in the first layer of the network, and batch size is 128. We will analyze the average-case behavior of the network, *i.e.*, we will assume that each of the mixes receives exactly 32 messages, even though it is possible (albeit highly improbable) that in some batch all 128 senders will randomly choose the same entry mix.

Under the equal loading assumption, we treat the size of the input/output buffer for each mix (see Section 5.3) as a constant which is determined only by batch size and network topology, and is independent of the actual random distribution of a given batch through the network.

Appendix B provides a walk-through of calculating entropy for each topology, to help the unfamiliar reader build intuition about our assumptions and results.

6 Graphs and Analysis

Figure 6 shows the entropy Alice can expect from each of the three topologies. The cascade network immediately divides the incoming batch by the number of cascades, so it provides substantially less protection even with many compromised nodes. The stratified topology provides about the same expected entropy as the free-route topology. In this section and the next we will examine other

metrics for deciding which is best. Further graphs in Appendix A indicate how much entropy is achieved after a given number of steps through each network.

6.1 Is the adversary really randomly distributed?

To keep our model tractable, we have assumed that each node has an equal chance of being controlled by the adversary. A real adversary might prefer to control certain key nodes in the topology. To justify our assumption, we might assume that secure nodes (or equivalently, vulnerable nodes) are randomly distributed. That is, rather than letting the adversary have his pick of nodes, we instead let the adversary control all the machines that have some security vulnerability. A related approach would be to place particularly secure and trusted (or at least jurisdictionally separate) nodes in key places in the topology: if such nodes are discouragingly secure, they are no longer an appealing target.

Alternatively, the mixes can periodically generate a communally random seed to reorganize the network [12]. Thus, being able to control or sign up a node does not allow the adversary to dictate its position in the topology. This may be a satisfactory solution, though it is not a complete solution because not all nodes are equal: e.g. nodes that refuse to deliver messages to the final recipients shouldn't be chosen as exit nodes, so they may be less appealing targets.

6.2 Choosing the same node twice in a row

Conventional wisdom (see e.g. [8]) suggests that in a free-route network, Alice should never pick the same node twice in a row: it increases her risk of picking only bad nodes. We find that for a sufficiently large network, this increased complexity in path selection has little impact on Alice's entropy.

Intuitively, when the adversary density is low, entropy will be high in either case; whereas when most nodes are owned by the adversary, the difference between picking between B and $B-1$ bad nodes is slight.

More formally, for G good nodes and B bad nodes, the chance of selecting a bad node next is $\frac{B-1}{G+B}$ if the current node is bad and $\frac{B}{G+B}$ otherwise. The difference is only $\frac{1}{G+B}$: it does not depend on what fraction of the nodes are bad. Specifically, for a 16x4 free-route mixnet (8 bad nodes), it's a 5.1% chance of an all bad path if a node cannot be picked twice in a row, and 6.3% chance if it can. With 32x4, it's 5.7% vs. 6.3%.

6.3 Reputations and Node Preferences

Most deployed systems let users choose a preferred entry or exit hop, e.g. based on trust. A skewed distribution of messages only at the entry or exit of the network should not impact entropy too much—we see from Figures 7-9 that much of each network's entropy is achieved from just a few hops.

Reputation systems, on the other hand, encourage users to prefer certain nodes at *each layer* of the network. Further, reputation information can be exploited by an adversary to reduce anonymity, for example by predicting the

user's behavior based on reputation statistics, or by attracting more traffic by building a strong reputation or degrading the reputation of others. Placing nodes with similar reputation in the same layer of a stratified network, or placing them in the same cascade, might complicate these attacks, but employed naively, this can facilitate other attacks [12]. This topic merits further investigation.

6.4 Average Entropy vs Actual Entropy

The graphs and analysis above are for average entropy—the network's behavior for very large batches. But in reality the batch size may be quite small, and each sender chooses paths independently from the others. We must consider the possible variance in entropy depending on the actual path choices.

For m messages to u buckets (nodes in a layer), we find the chance that any bucket will have less than p messages based on Maxwell-Boltzmann statistics and inclusion-exclusion:

$$\binom{u}{1}\sum_{i=0}^{p}(\frac{1}{u})^i(1-\frac{1}{u})^{m-i}\binom{m}{i} - \binom{u}{2}\sum_{i=0}^{p}\sum_{j=0}^{p}(\frac{1}{u})^i(\frac{1}{u})^j(1-\frac{2}{u})^{m-i-j}\binom{m}{i,j}$$
$$+\binom{u}{3}\sum_{i=0}^{p}\sum_{j=0}^{p}\sum_{k=0}^{p}(\frac{1}{u})^i(\frac{1}{u})^j(\frac{1}{u})^k(1-\frac{3}{u})^{m-i-j-k}\binom{m}{i,j,k} - \ldots$$

For $m = 128$ messages and $u = 4$ nodes (i.e. cascade or stratified network), the chance of any node getting less than 16 messages (compared to the 32 we expect each to get) is $6 \cdot 10^{-4}$—meaning with very high probability the average entropy represents the behavior we will see in reality. However, for $u = 16$ nodes (free-route), 48% of the time some node will get less than half the expected number; and it is not until a batch size of 480 that this metric reaches 1%.

This result makes sense: each link on a free-route network has a smaller expected number of messages, so variations have a bigger impact. Whether it is acceptable depends on a number of factors. First, how large do we expect batches to be in reality? The Mixmaster network receives more than 1000 messages an hour, which seems plenty sufficient. Second, how bad is it when a link varies by half the expected volume? If we change our metric to require at least 2 messages on each link, then for $m = 128$ we find that only 1% of the cases fall outside this value. Another significant question is how the number of layers affects the results: the more layers, the greater the chance that some of them are well balanced. The exact relation and its effect on entropy are open questions.

Danezis also considers this issue of variance from average entropy for his mixnet design based on sparse expander graphs [6]. He argues that having at least one message on each link is sufficient for basic protection, and he uses a similar approach to show that his design achieves this distribution with high probability. He further raises the idea of padding unused links to guarantee one message on each link, with the aim of preventing trivial traffic analysis attacks. Is it worthwhile to prevent this simple attack? Are all other attacks significantly harder? Clearly more research remains.

6.5 Flooding attacks to degrade anonymity or service

In Section 3.3 we talk about techniques to discourage a mix from dropping or substituting messages in the batch. But what if the adversary simply submits *more* messages to the batch?

It turns out that as long as k of the n input messages come from honest senders, Alice will still be assured that she can expect entropy based on a batch of k messages. That is, assuming uniform distribution of messages over mixes, the entropy of a baseline network (all-honest senders) plus hostile messages is at least the entropy of the baseline network by itself. This is different from the pooling batching strategy [24], where messages from the adversary will influence the behavior (and thus entropy) of Alice's message.

On the other hand, directed floods can overflow node capacity. We might use techniques where mixes can prove that any output message was derived from an input message, which reduces the problem to detecting or stopping floods at the beginning of the batch. We might also argue that the fraction of adversary messages in the batch limits the maximum size of the flooding attack—honest messages will still be randomly distributed. In general, this flooding issue is an unsolved problem for all mixnet designs; more research remains.

7 Other metrics for comparison

7.1 Throughput, delay, capacity, bandwidth

One parameter we cannot control is the rate that messages arrive to the mixnet. Similarly, we cannot control the latency that users will be willing to accept. To make the analysis more concrete, assume we choose $\ell = 4$, that users deliver 128 messages every 3 hours, and that users will tolerate a latency of 3–6 hours (which is on par with the latency experienced by a typical Mixmaster message, though it could be much longer in theory).

We can compute the maximum flow rate (traffic in unit time) through any given node. Assume that sending a message over a single hop consumes a fixed amount of network traffic; we can then use that as the unit for traffic. Let T_{batch} be the expected throughput in a single batch period, i.e. the number of messages that go through the network in a batch. If the available nodes are used optimally (see Section 6.4), the flow rate required through each node is $\frac{T_{\text{batch}}}{w \cdot t_{\text{hop}}} = \frac{\ell \cdot T_{\text{batch}}}{w \cdot t_{\text{batch}}}$.

If we choose $t_{\text{batch}} \simeq \ell t_{\text{hop}}$, all messages clear the mixnet before the next batch enters: we introduce a batch of 128 messages every 3 hours. We get 42.7 messages/hour for all three topologies. Latency is between 3 hours and 6 hours, depending on when Alice's message arrives. By accepting messages over a large amount of time, we get better expected entropy; make the actual behavior of the network closer to the expected behavior of the network (as in Section 6.4); and smooth spikes and troughs in the rate of incoming messages.

In the free-route network, each node needs to process 8 messages at a time and is active at each layer. The cascade and stratified networks require a larger capacity from each node: they must handle 32 messages at once ($128/w$), but

they are idle for all but one hop in the batch. One could imagine a *systolic* or *pipelined* network where $t_{\text{batch}} = t_{\text{hop}}$ and 32 messages are let in every 45 minutes. In this case the capacity of nodes in cascade and stratified networks would also be 8, and indeed the latency could be cut to between 3 hours and 3 hours 45 minutes—but the expected entropy would be cut by a factor of ℓ.

Bandwidth is acceptable. Assuming a higher load of 5000 messages per batch, and 32KB per message (as in Mixminion), nodes in the free-route system use less than 4KB/s (nodes in the other topologies use 16KB/s but only 1/4 as often). That's well within the capabilities of current Mixmaster nodes.

7.2 Robustness of Message Delivery

Better entropy can be achieved by longer routes: e.g., if we form our 16 nodes into a 1x16 cascade or a 16x16 free-route, there is almost no falloff in entropy until each node has a ninety percent chance of being compromised. But this ignores robustness of message delivery. For the free-route 16x16 mixnet with only a single node failure, nearly two thirds of messages will be undelivered (because they will need to pass through it at some point). The 1x16 cascade is even worse: a single node crash blocks all message delivery. (We might take advantage of schemes to bypass a single failed node [22], but it's not clear how this works with the synchronous approach in all topologies.) Parallel cascades can be added to the network, but unlike the free-route, they will *a priori* reduce the entropy of an input message for a given size mixnet batch. We must be sure to consider robustness when comparing topologies.

	Topology	1 crash	2 crash	3 crash	4 crash
Worst possible adversary distribution	16x16 free	36	12	04	01
	4x4 cascade	75	50	25	00
	4x4 stratif.	75	50	25	00
	16x4 free	77	59	44	32
Best possible adversary distribution	16x16 free	36	12	04	01
	4x4 cascade	75	75	75	75
	4x4 stratif.	75	56	42	32
	16x4 free	77	59	44	32
Expected percentage: rand. adversary dist.	16x16 free	36	12	04	01
	4x4 cascade	75	55	39	27
	4x4 stratif.	75	55	39	27
	16x4 free	77	59	44	32

Table 1. Percent of messages delivered vs number of crashed nodes

Table 1 shows that 4x4 cascades and 4x4 stratified networks do roughly the same on average, but this is for very different reasons. The chance that the

configuration will block all messages increases much more quickly for cascades, but the maximum possible delivery of messages remains much higher. This can be seen in the table reflecting the most favorable adversary distribution for up to four node crashes. To further illustrate, if half of the nodes are bad in the 4x4 cascade topology, then in about 1 in 6 cases a quarter of the messages get through, and in exactly 6 cases of 12870, half of the messages get through the cascades. For all other distributions, no messages get through. If half of the nodes are bad in the 4x4 stratified network, then the highest percentage of messages that can pass through is 6.25. However, some messages will be passed in the majority of adversary distributions.

Of the scenarios we have considered, a 16x4 free route has the best expected chance of message delivery for random adversary distribution. It outperforms the others, unless the adversary has a particularly innocuous distribution. Cascades do better under favorable distributions, which are also much rarer for cascades than other topologies. Also note that the expected fraction of passed messages is the same for free routes regardless of which nodes fail: it is the most robust with respect to adversary distribution as well as adversary size.

7.3 Robustness of Anonymity

Robustness of anonymity against active attacks is harder to determine, as such attacks can take on a variety of forms. In the simplest case though, we can consider the effect on anonymity of simple node crash, since this is the most straightforward way to actively shrink anonymity. Also, as discussed in Section 3.3, there are techniques to detect and/or deter more selective attacks.

The threat model we consider here is an extension of the one in Section 4. As before, the adversary can watch senders and receivers. But now, besides failing to mix, hostile nodes may also crash—failing to deliver any of their input messages. A combination of active attacks and observations (including some internal observations) should prove the most devastating to anonymity. However, we leave full examination of this for future work. Here we concentrate on the effect of such intentional crash failures on entropy for a mixnet periphery observer.

Anonymity of cascades is unaffected by this threat model. Since each cascade batch is independent of the others, any node that crashes will wipe out all the messages in that anonymity set. Anonymity robustness of stratified and free-route topologies is more complex.

For a stratified network, if any entry node fails, the number of messages drops by one quarter, causing a reduction in entropy of .42. If two entry nodes fail, the entropy drops by 1. If 3 entry nodes fail, entropy drops by 2. If all fail, the adversary learns nothing more than if none fail. If a second layer node fails, assuming a balanced layer-two distribution, anonymity of all messages is unaffected since there is no change to the probability that an exiting message was any incoming message. Note this is so even if the distribution of messages across entry nodes is highly skewed. If the layer-two distribution is skewed, then a node may fail with some effect on entropy. However, the ability to affect anonymity in this way should be very small for randomly chosen routes. Ignoring the small

effect of such non-entry-layer failures, we see that the anonymity of a stratified network given node crashes is usually better and at worst equal to that of the cascade topology.

Free routes are even more complex. For entry layer nodes, the initial effect of each crash is clearly smaller. However, since nodes are used at multiple layers, a message that reaches a crashed node at a given layer could not have been routed through that node at any earlier layer. Further, the attacker may gain additional information by crashing nodes only at certain layers! Even worse, as the ratio of input messages to width of a layer shrinks, it becomes more likely that nodes at a given layer will only receive messages from a subset of nodes at the previous layer or, in any case, that the layer distribution will be unbalanced between nodes to a significant degree.

On the other hand, because nodes are recycled for use at multiple layers, it is much harder to plan an attack. If nodes can't crash and then come back in a single batch (perhaps it's hard to do undetectably), crashing an entry node to reduce the anonymity of a message at another node may cause that message to be blocked when it must traverse the crashed node at a later layer. But it will generally be hard to predict when to come up beyond a few layers, because the targeted message will likely be coming from any of the remaining nodes after that much mixing.

To get some handle on this complex situation, we will consider a very lucky adversary. The adversary controls a quarter of the nodes in a 16x4 recycling free-route. Suppose a message enters the mixnet at a node not under adversary control, and the adversary crashes all of its nodes. Messages drop by a quarter. If the layer-2 distribution is such that the layer-1 node that received the target does not send any messages to the four adversary nodes, they remain crashed. Assuming that a quarter of the remaining messages are addressed to them at layer-2, remaining messages are now .56 of the original batch. Repeat for layer-3 and layer-4. Remaining messages are down to .32 of the original mixnet batch. In this case, despite all the luck of the adversary, the anonymity is thus still better than that of a message sent into a cascade processing a quarter of the original mixnet batch.

We have not considered all possible active attacks. But for those we have considered, the best choice for anonymity robustness appears to be the free route, and worst is the cascade. We invite further research.

7.4 Comparison with Asynchronous Batching Designs

We have shown synchronous free-routes can provide good anonymity, but we must also begin comparing this design to more traditional asynchronous free-route designs like Mixminion. Synchronous batching needs no replay cache (each message is labeled with its batch), weakens partitioning attacks from blending and key rotation, and generally provides clearer anonymity guarantees.

On the other hand, because Mixminion's pool batching strategy spreads out message distributions between batches, our design may fall more quickly to long-term statistical disclosure attacks [7]. Our design is also less robust to transient

failures: a late Mixminion message still arrives, whereas in our system a node that is down throughout t_{hop} loses all messages going through it. (Stratified and cascade networks have the lowest chance of being down in a hop period they are needed, but free-route networks lose proportionally fewer messages from a single down node.) But our design can tell the user for sure whether his mail was delivered in the batch (and he can resend if not), whereas Mixminion's unpredictability always leaves the user wondering if it will come out sometime.

Like stop-and-go mixes [20], we may be able to get improved anonymity by allowing Alice to choose to delay her message at a given hop until the next batch. That is, the node would delay her message by t_{batch} and re-introduce it at the same point in the path. If each message is either delayed once or not delayed, that gives us a latency of 3 to 6 hours for non-delayed messages, 6 to 9 hours for delayed messages, and a 6-hour anonymity set (unless the attacker knows that someone never sends or receives delayed messages, in which case the anonymity set for those users is still 3 hours; also, if the attacker owns the node Alice chooses, he may be able to speculate about which senders would choose to delay messages). We leave further comparison to future work.

8 Summary

Previously, only cascade networks were considered secure against very powerful adversaries [3]. In this paper we show that other topologies can use the synchronous batching strategy to achieve similar protection. Further, we show that free-route topologies with synchronous batching compare favorably to cascade networks. We invite further analysis of the trade-offs between each topology.

Acknowledgments

We acknowledge David Hopwood for the ideas and arguments behind Sections 2 and 3; and we thank LiWu Chang, Camilla Fox, Rachel Greenstadt, Chris Laas, Ira Moskowitz, and Itamar Shtull-Trauring for probability discussions.

References

1. Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability*, pages 115–129. Springer-Verlag, LNCS 2009, 2000.
2. Oliver Berthold and Heinrich Langos. Dummy traffic against long term intersection attacks. In Roger Dingledine and Paul Syverson, editors, *Proc. Privacy Enhancing Technologies workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
3. Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The disadvantages of free MIX routes and how to overcome them. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.

4. David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1982.
5. David Chaum. Blind signatures for untraceable payments. In D. Chaum, R.L. Rivest, and A.T. Sherman, editors, *Advances in Cryptology:Proceedings of Crypto 82*, pages 199–203. Plenum Press, 1983.
6. George Danezis. Mix-networks with restricted routes. In Roger Dingledine, editor, *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*. Springer-Verlag, LNCS 2760, March 2003.
7. George Danezis. Statistical disclosure attacks: Traffic confirmation in open environments. In Gritzalis, Vimercati, Samarati, and Katsikas, editors, *Proceedings of Security and Privacy in the Age of Uncertainty, (SEC2003)*, pages 421–426, Athens, May 2003. IFIP TC11, Kluwer.
8. George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *2003 IEEE Symposium on Security and Privacy*, pages 2–15. IEEE CS, May 2003.
9. George Danezis and Len Sassaman. Heartbeat traffic to counter (n-1) attacks. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2003)*, Washington, DC, USA, October 2003.
10. Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Roger Dingledine and Paul Syverson, editors, *Privacy Enhancing Technologies (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
11. Roger Dingledine, Michael J. Freedman, David Hopwood, and David Molnar. A Reputation System to Increase MIX-net Reliability. In Ira S. Moskowitz, editor, *Information Hiding (IH 2001)*, pages 126–141. Springer-Verlag, LNCS 2137, 2001.
12. Roger Dingledine and Paul Syverson. Reliable MIX Cascade Networks through Reputation. In Matt Blaze, editor, *Financial Cryptography*. Springer-Verlag, LNCS 2357, 2002.
13. John Douceur. The Sybil Attack. In *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS 2002)*, March 2002.
14. M. Kwiatkowska *et al.* PRISM web page. http://www.cs.bham.ac.uk/~dxp/prism/.
15. Elke Franz, Andreas Graubner, Anja Jerichow, and Andreas Pfitzmann. Comparison of Commitment Schemes Used in Mix-Mediated Anonymous Communication for Preventing Pool-Mode Attacks. In C. Boyd and E. Dawson, editors, *3rd Australasian Conference on Information Security and Privacy (ACISP'98)*, number 1438 in LNCS. Springer-Verlag, 1998.
16. David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding routing information. In R. Anderson, editor, *Information Hiding: First International Workshop*, pages 137–150. Springer-Verlag, LNCS 1174, 1996.
17. Ceki Gülcü and Gene Tsudik. Mixing E-mail with Babel. In *Network and Distributed Security Symposium (NDSS 96)*, pages 2–16. IEEE, February 1996.
18. Markus Jakobsson. Flash Mixing. In *Principles of Distributed Computing - PODC '99*. ACM Press, 1999.
19. Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th USENIX Security Symposium*, August 2002.
20. Dogan Kesdogan, Jan Egner, and Roland Büschkes. Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Proceedings of Information Hiding Workshop (IH 1998)*. Springer-Verlag, LNCS 1525, 1998.
21. Ulf Möller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster Protocol — Version 2. Draft, July 2003. <http://www.abditum.com/mixmaster-spec.txt>.

22. Andreas Pfitzmann and Michael Waidner. Networks without user observability – design options. In *Proc. of EUROCRYPT 1985*. Springer-Verlag, LNCS 219, 1985.
23. Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Roger Dingledine and Paul Syverson, editors, *Privacy Enhacing Technologies (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.
24. Andrei Serjantov, Roger Dingledine, and Paul Syverson. From a trickle to a flood: Active attacks on several mix types. In F. Petitcolas, editor, *Proceedings of Information Hiding Workshop (IH 2002)*. Springer-Verlag, LNCS 2578, October 2002.
25. V. Shmatikov. Probabilistic model checking of an anonymity system. *Journal of Computer Security (selected papers of CSFW-15)*, 2004 (to appear).

A Entropy vs number of hops, for each topology

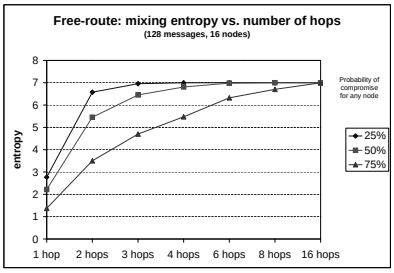

Fig. 7. Entropy vs number of hops, for cascade network (16 nodes)

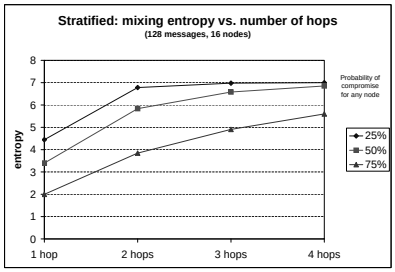

Fig. 8. Entropy vs number of hops, for stratified network (16 nodes)

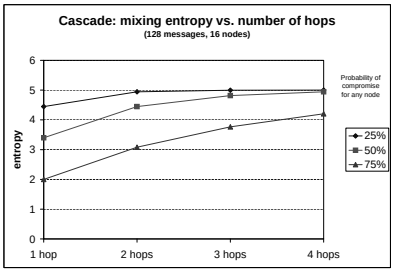

Fig. 9. Entropy vs number of hops, for free-route network (16 nodes)

B Entropy examples for each topology

B.1 Cascade.

Consider a 2x2 cascade network as in Figure 1. Assume there are 128 messages in a batch, and that any node has $\frac{1}{4}$ chance of having been compromised. Let m be the targeted message, and suppose the adversary observed that the sender of m chose A as the first mix. Under the equal loading assumption, there are 63 other messages in addition to m that chose A as the first mix. Without loss of generality, we may assume that m occupies the first position in A's input buffer of length 64.

With probability $\frac{1}{4}$, mix A is hostile. In this case, m remains in the first position after passing through the mix. With probability $\frac{3}{4}$, mix A is honest. In this case, m appears in any of the 64 output positions of this mix with probability $\frac{1}{64}$ (note that m may not appear in the output buffer of mix B). The resulting probability distribution for m's position after passing through A is

$$\begin{array}{cccc} \underbrace{\frac{67}{256}}_{\frac{1}{4}\cdot 1+\frac{3}{4}\cdot\frac{1}{64}} & \underbrace{\frac{3}{256}}_{\frac{3}{4}\cdot\frac{1}{64}} & \ldots & 0\ 0\ \ldots\ 0 \\ \text{position 1} & \text{positions 2..64} & & \text{positions 65..128} \end{array} \tag{1}$$

Next mix C is pre-determined by network topology, and the distribution it produces on its messages is the same as (1). Combining two distributions, we obtain that m appears in the cascade's output buffer with following probabilities:

$$\begin{array}{cccc} \underbrace{\frac{5056}{65536}}_{\frac{67}{256}\cdot\frac{67}{256}+\frac{3}{256}\cdot(63\cdot\frac{3}{256})} & \underbrace{\frac{960}{65536}}_{\frac{67}{256}\cdot\frac{3}{256}+\frac{3}{256}\cdot(\frac{67}{256}+62\cdot\frac{3}{256})} & \ldots & 0\ 0\ \ldots\ 0 \\ \text{position 1} & \text{positions 2..64} & & \text{positions 65..128} \end{array} \tag{2}$$

Entropy of this distribution is approximately 5.9082. Effective anonymity set provided by a 2x2 cascade with 25% density of hostile nodes and 128 messages per batch is 60 messages.

B.2 Stratified array.

The procedure for calculating mixing distribution for a stratified array is essentially the same as for a cascade, but there is an additional probabilistic choice. After the message passes through a mix, the next mix is selected randomly among all mixes in the next layer.

Consider a 2x2 stratified array as in fig. 2. Again, assume there are 128 messages in a batch, $\frac{1}{4}$ chance that a node is hostile, and that A was selected (in a manner visible to the adversary) as the first mix. The mixing performed by any single mix is exactly the same as in the cascade case, thus mixing distribution (1) after the first hop is the same in a stratified array as in a cascade.

After the first hop, however, mix C is selected only with probability $\frac{1}{2}$, while D may also be selected with probability $\frac{1}{2}$ (by contrast, in a cascade C is selected with probability 1). Distribution (2) has to be adjusted to take into account the fact that mix D, selected with probability $\frac{1}{2}$, has a $\frac{1}{4}$ chance of being hostile and thus leaving each received message in the same position.

$$\underbrace{\frac{4672}{65536}}_{\frac{1}{2}\cdot\frac{5056}{65536}+\frac{1}{2}\cdot\frac{1}{4}\cdot\frac{67}{256}} \quad \underbrace{\frac{576}{65536}}_{\frac{1}{2}\cdot\frac{960}{65536}+\frac{1}{2}\cdot\frac{1}{4}\cdot\frac{3}{256}} \quad \cdots \quad \underbrace{\frac{3}{512}}_{\frac{1}{2}\cdot\frac{3}{4}\cdot\frac{1}{64}} \quad \cdots$$

position 1 positions 2..64 positions 65..128

Entropy of this distribution is approximately 6.8342. Effective anonymity set provided by a 2x2 stratified array with 25% density of hostile nodes and 128 messages per batch is 114 messages.

B.3 Free-route network.

Probability distribution is computed in exactly the same way for a free-route network as for a stratified array, except that the entire set of mixes is treated as a layer. Consider a 4x2 free-route network as in fig. 3. With 128 messages per batch, the buffer for each mix is 32 messages. If A is the first mix selected (and is hostile with probability $\frac{1}{4}$), the probability distribution after the message passes through A is

$$\underbrace{\frac{35}{128}}_{\frac{1}{4}\cdot 1+\frac{3}{4}\cdot\frac{1}{32}} \quad \underbrace{\frac{3}{128}}_{\frac{3}{4}\cdot\frac{1}{32}} \quad \cdots \quad 0\ 0\ \ldots\ 0$$

position 1 positions 2..32 positions 33..128

The next mix is selected from among all four mixes with equal probability. A mix other than A is selected with probability $\frac{3}{4}$, and has $\frac{1}{4}$ chance of being hostile, producing the following probability distribution:

$$\underbrace{\frac{1216}{16384}}_{\frac{1}{4}\cdot\left(\frac{35}{128}\cdot\frac{35}{128}+\frac{3}{128}\cdot\left(31\cdot\frac{3}{128}\right)\right)+\frac{3}{4}\cdot\frac{1}{4}\cdot\frac{35}{128}} \quad \underbrace{\frac{192}{16384}}_{\frac{1}{4}\cdot\left(\frac{35}{128}\cdot\frac{3}{128}+\frac{3}{128}\cdot\left(\frac{35}{128}+30\cdot\frac{3}{128}\right)\right)+\frac{3}{4}\cdot\frac{1}{4}\cdot\frac{3}{128}} \quad \cdots \quad \underbrace{\frac{3}{512}}_{\frac{1}{4}\cdot\frac{3}{4}\cdot\frac{1}{32}} \quad \cdots$$

position 1 positions 2..32 positions 33..128

Entropy of this distribution is approximately 6.7799 and effective anonymity set is 110 messages — slightly lower than in a stratified 2x2 array.