Herd: A Scalable, Traffic Analysis Resistant Anonymity Network for VoIP Systems

Stevens Le Blond
MPI-SWS

David Choffnes
Northeastern University

William Caldwell
MPI-SWS

Peter Druschel
MPI-SWS

Nicholas Merritt
MPI-SWS

ABSTRACT

Effectively anonymizing Voice-over-IP (VoIP) calls requires a scalable anonymity network that is resilient to traffic analysis and has sufficiently low delay for high-quality voice calls. The popular Tor anonymity network, for instance, is not designed for the former and cannot typically achieve the latter. In this paper, we present the design, implementation, and experimental evaluation of Herd, an anonymity network where a set of dedicated, fully interconnected cloud-based proxies yield suitably low-delay circuits, while untrusted superpeers add scalability. Herd provides caller/callee anonymity among the clients within a trust zone (e.g., jurisdiction) and under a strong adversarial model. Simulations based on a trace of 370 million mobile phone calls among 10.8 million users indicate that Herd achieves anonymity among millions of clients with low bandwidth requirements, and that superpeers decrease the bandwidth and CPU requirements of the trusted infrastructure by an order of magnitude. Finally, experiments using a prototype deployment on Amazon EC2 show that Herd has a delay low enough for high-quality calls in most cases.

CCS Concepts

•**Networks** → **Network privacy and anonymity;**

Keywords

Anonymity networks; Voice-over-IP; Intersection attacks; Strong anonymity

SIGCOMM '15 August 17-21, 2015, London, United Kingdom

ACM ISBN 978-1-4503-3542-3/15/08.

DOI: http://dx.doi.org/10.1145/2785956.2787491

1. INTRODUCTION

Voice-over-IP (VoIP) systems are popular and their support for pseudonymous identities and strong encryption makes them appealing to users concerned about communication privacy. However, many nation states use Internet surveillance to monitor and repress critics [18, 14], and even western democracies conduct mass surveillance [9, 10, 8, 4]. In this environment, VoIP systems are not sufficient to ensure the anonymity of Internet users, nor the safety of political activists and whistleblowers, as illustrated, for example, by the NSA's ability to intercept Skype conversations [13].

Further, existing anonymity networks are not designed for latency sufficiently low to support VoIP, or scalability to millions of users under a strong adversarial model. For example, Tor [28] scales to large numbers of users but is vulnerable to an adversary who can eavesdrop on the ingress and egress paths of Tor circuits. Recent revelations indicate that the UK's GCHQ is pursuing this to deanonymize Tor users [1].

We demonstrate in Section 4 that a simple attack using only the start and end times would trace 98.3% of the calls of a large, real voice workload, if they had been made over Tor. (A more sophisticated attack that also considers the time series of encrypted packets would likely trace even more calls.) In addition, Tor typically incurs round trip delays between 2–4 seconds on established, sender-anonymous circuits because of random proxy selection and high-latency connections [43, 15], which is prohibitive for VoIP.

Other anonymity systems resist traffic analysis at the cost of performance or scalability [37, 47, 26, 30, 45]. For example, Dissent, the state-of-the-art Dining Cryptographers network (DC-Net), is resilient under a strong threat model but requires one *broadcast channel* per anonymity set, impacting anonymity and scalability, as well as *several round-trip times*, yielding delays significantly higher than Tor [47].

In this paper, we present the design, implementation, and evaluation of *Herd*, a scalable anonymity network that combines delays suitable for VoIP with anonymity under a strong adversarial model. *Herd* achieves these properties using a novel design tailored to the specific requirements and traffic characteristics of VoIP. In particular, (1) *Herd* leverages the multitude of jurisdictions in the Internet with a design that ensures a user's anonymity depends only on mixes located in jurisdictions that (s)he trusts, independent of her communication partners' choices; (2) it exploits the constant-rate, low-bandwidth nature of VoIP traffic to resist traffic analysis while achieving low delay; (3) it employs a novel design based on network coding to hide clients' activity patterns, which enables *Herd* to offload its trusted infrastructure with the help of untrusted peers.

At its core, *Herd* relies on a set of dedicated mixes (proxies that relay VoIP data packets) operated by independent organizations in different jurisdictions around the world. The underlying insight is that no adversary has access to mix locations everywhere in the world. *Herd* exploits this by allowing clients to choose a mix within a *trust zone*, i.e., a jurisdiction/provider that is friendly to their cause, and by guaranteeing that client anonymity depends only on this choice.

Because the set of mixes is fully connected, a *Herd* circuit involves at most three intercontinental links (usually one), which keeps delays low enough for acceptable VoIP quality. To increase scalability, *Herd* can optionally use untrusted superpeers that forward traffic between clients and a mix. Superpeers reduce the CPU and bandwidth demands on trusted infrastructure without sacrificing anonymity. Bitwise unlinkablity (i.e., decorrelating the content of packets arriving and departing a mix) and traffic-analysis resistance is achieved via encrypted connections padded using artificial traffic (*chaff*), where rate adjustments are correlated within large anonymity sets.

This paper makes the following contributions:

- We present the design of *Herd*, a scalable anonymity network with latency sufficiently low to carry VoIP calls, and anonymity under a strong adversarial model.
- A trace-based evaluation of *Herd*'s overheads and performance using a trace of 370 million mobile phone calls among 10.8 million users, and a comparison with Drac [26], an existing low-delay anonymity network.
- An open-source implementation of *Herd*.
- An experimental evaluation of *Herd*'s perceived call quality based on VoIP calls made by volunteers using a *Herd* deployment on Amazon EC2.

Our key results are as follows. First, using trace-based simulations we show that *Herd* achieves anonymity with low bandwidth requirements for millions of clients. Second, we find that superpeers decrease the bandwidth and CPU demands on trusted infrastructure by an order of magnitude, because they allow *Herd* to take advantage of resource contributions by untrusted parties without affecting anonymity. We estimate that untrusted superpeers that contribute resources for free or in exchange for free calls would reduce *Herd*'s operational costs from $10-100 to $0.10-1.14 per user/month for this workload. Third, experiments with an implementation and a prototype deployment on 4 Amazon EC2 data centers show that *Herd* incurs modest additional latency as compared to direct communication. The *Herd* source code is available under a BSD license [19].

We discuss background and related work on private voice communication in Section 2. We then present the design and implementation of *Herd* in Section 3 and compare its performance with alternative low-latency anonymity networks in Section 4. We conclude in Section 5.

2. BACKGROUND & RELATED WORK

2.1 Context

Providing anonymous communication is challenging from a technical, legal and political standpoint. Recent revelations about state-sponsored mass surveillance suggest that government agencies analyze large-scale data collected at Internet exchange points [1, 4] and request customer data from service providers [10, 9]. In this environment, achieving effective and practical anonymity is difficult.

Even in the presence of such powerful attackers, however, we posit that there exist free havens supporting private communication, even if those free havens differ depending on the communication partners and their topic.

We believe this is the case for two reasons. First, *no single adversary has access to all parts of the Internet and jurisdiction over all service providers*. This is true despite the fact that countries like the US and the UK have access to a disproportionate amount of Internet traffic today. Moreover, in the wake of the recent revelations, there is some indication that more countries may seek to establish national Cloud and Web service providers and independent Internet routes that do not needlessly cross other countries or continents [17].

Second, *for any worthy cause that a group of users pursue under the cloak of anonymity, there is likely a country that is friendly or at least indifferent to that cause*. Combined with the first point, we believe that users can find a jurisdiction that avoids legal and political challenges to anonymous communication, leaving only technical challenges.

The design of *Herd* seeks to exploit these circumstances, by allowing callers and callees to independently choose a provider within a jurisdiction they deem friendly to their cause, and through a trust model where a user's anonymity depends only on their own choice of provider. Before we describe the design of *Herd* in the next section, we review existing work in anonymous voice services.

2.2 Existing techniques

Burner phones. A simple method for anonymous voice calling is to use pre-paid SIM cards and cheap cellular phones purchased with cash and discarded after a period of use. To receive calls, a user must disseminate her current phone number to callers using an out-of-band mechanism. Using burner phones may not be appropriate in all cases due to its monetary cost, relative inconvenience, and the unavailability of unregistered, pre-paid SIM cards in some countries. Further, using a burner phone does not guarantee that calls made using the phone will not be tapped or otherwise logged.

VoIP services. In principle, VoIP services are an alternative for anonymous voice calling. While VoIP signaling protocols such as SIP and P2PSIP were not designed for anonymous call setup, it is possible, for instance, to create a Skype or Google Voice account under a pseudonym and contact the service through a VPN, which obscures the user's IP address. However, VPNs can be compelled by authorities to release the IP addresses of their customers. For instance, a UK-based VPN recently complied with a US subpoena to trace one of its users [3]. Also, it is widely assumed that Skype complies with requests for call metadata by government authorities [7], and that the NSA has the ability to intercept the contents of Skype calls [13].

Private VoIP services like RedPhone [16] and Silent Circle [12] offer end-to-end encrypted voice calls (strong encryption and at least partially open source), and advertise that no call metadata is stored. However, the services rely on rendezvous and relaying services that can be compelled by local authorities to capture and release call metadata. A subpoena against the operator of a server involved in a call may reveal the IP addresses of caller and callee.

None of the existing VoIP services are designed to resist *traffic analysis*, where a passive network observer can determine the caller and callee by matching the time series of encrypted traffic at different points in the network.

Anonymity networks. Users can improve protection against de-anonymization by accessing a VoIP service via an anonymity network like Tor [15]. Unfortunately, the typical delays incurred by Tor circuits are too high for adequate VoIP call quality. More generally, a measurement study indicates that achieving acceptable VoIP call quality using an Tor-like (onion routing) anonymity network based on volunteered resources (PlanetLab) is challenging [39].

LAP [33] is a light-weight anonymity network with near-optimal delay and low routing overhead. However, it assumes a weaker threat model than Tor and requires network support that is not deployed today.

Traffic analysis resistant anonymity. Traffic analysis consists of statistical attacks that reveal communication partners, by considering the time series of packets and user activity. Tor, like most other medium-delay anonymity networks [33, 21, 20], is not designed to withstand traffic analysis, and we empirically show the effectiveness of such attacks against Tor in Section 4.1.4.

Anonymity networks can batch messages to hide the correlation between input and output messages, as originally done in Chaumian mix-nets [24]. Batching prevents an attacker from tracing messages based on their arrival and departure times, but without chaff it requires delaying messages for substantial periods. As a result, mix-nets are suitable only for delay-tolerant communications such as e-mails [32, 40, 27].

ISDN-MIXes [44, 35] combine mixing of time-sliced channels with chaff traffic and broadcast of incoming calls on subscriber links to achieve anonymous telephony on an ISDN network. *Herd* instead provides anonymous VoIP calls over the Internet, under a threat model that considers today's powerful adversaries.

Finding the right tradeoff between delaying messages and adding chaff traffic is an open research challenge. For example, both the Java Anonymous Proxy (JAP) [21] and the Freedom Network [46] abandoned chaffing due to its high bandwidth overhead. Below, we discuss designs that combine batching with chaffing to reduce mix-net delays, as well as Dissent, a system based on DC-nets.

Aqua [37] provides k-anonymity for BitTorrent-like traffic in the presence of traffic analysis. While the designs of *Herd* and Aqua share several ideas, they support different applications and traffic types, and differ in other ways: First, *Herd* uses a hybrid architecture with trusted mixes and untrusted superpeers (SPs), where SPs improve scalability using a novel design based on network coding. Second, *Herd* leverages (a) the constant-rate nature of VoIP traffic and (b) the jurisdictional diversity of the Internet to achieve paths that require less than half the number of hops as Aqua (at most 3 versus 7 intercontinental hops) without loss of resistance against traffic analysis. Whereas Aqua requires multipath between two pairs of mixes to disperse hotspots in file-sharing traffic, constant-rate traffic does not have this constraint, enabling *Herd* to achieve latencies appropriate for VoIP calls. By organizing its infrastructure into disjoint trust zones and coupling rate adjustments both within and across them, *Herd* decouples the anonymity of users of different zones, while Aqua does not consider mixes' jurisdiction.

P^5[45] and Tarzan [30] are P2P anonymity networks that use broadcast channels and chaffing, respectively, to defend against traffic analysis. Because they use a P2P model, their security depends on forwarding traffic via a series of proxies hosted on endpoints, affecting their latency and reliability.

Dissent [47] is an infrastructure-based anonymity service with a very strong adversarial model, where a single honest proxy is sufficient to ensure anonymity, even in the face of traffic analysis attacks. The system relies on DC-nets and verifiable shuffles, and its infrastructure-based architecture scales to hundreds of clients with modest delay and bandwidth sufficient for web browsing. However, the system's capacity, scalability, and latency are subject to DC-net scaling limits with respect to the number of proxies, overhead per payload bit for cryptographic processing, and delay.

Drac [26] is a traffic-analysis resistant anonymity network for VoIP and IM that relies on a social network to provide anonymous communication among strangers and unobservable communication among friends. An important difference with *Herd* is that Drac exposes users' social network to an adversary. Drac also routes calls in a peer-to-peer fashion over the social network, which can cause impractically long latency for calls between users who are several hops apart in the social graph. Whereas *Herd* clients maintain a handful of chaffing links (e.g., 2 or 3), Drac may require substantially more (e.g., Twitter and Facebook users have on average 45 and 190 contacts, respectively [31, 29]). We provide a quantitative comparison of Drac and *Herd*, in terms of anonymity, scalability, and perceived call quality in Section 4.

2.3 Our system: Herd

Herd is designed specifically for VoIP traffic and the anonymity needs of voice callers. *Herd* takes advantage of VoIP calling patterns and traffic characteristics to provide anonymity under a strong adversarial model with reasonable overhead and delay low enough for high-quality voice calls.

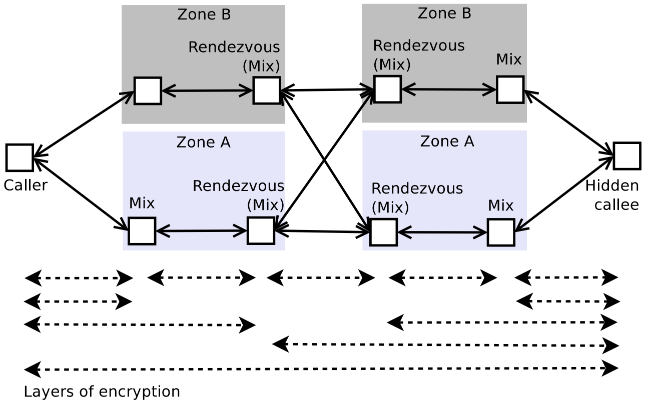

Figure 1: *Herd*'s architecture comprises clients, mixes located in trust zones, and optionally, SPs (not shown). *Herd* employs both hop-by-hop and layered encryption to hide content, routing information, and circuit IDs. Mixes can also act as rendezvous enabling clients to receive incoming VoIP calls with zone anonymity. Optionally, a caller and callee can use an additional zone (e.g., *A* and *B*) to avoid depending on the same jurisdiction.

Moreover, in *Herd*'s trust model, the anonymity of a caller or callee depends only on her own choice of a *trust zone*, i.e., a provider within a jurisdiction. Callers and callees pick a trust zone independently and do not depend on the choices made by their call partner.

3. DESIGN AND IMPLEMENTATION

System model. *Herd* comprises clients, which originate and terminate traffic; mixes, which relay traffic; and optional untrusted superpeers (SPs), which can originate, terminate, and relay traffic. We call a client initiating a call *caller* and one accepting a call *callee*. All participants are connected by links that carry encrypted and padded traffic. A client connects to at least one mix, either directly (shown in Figure 1), or via one or more SPs attached to the same mix. SPs connect to one mix. Links among the mixes form a fully connected network. *Herd* mixes are further partitioned into *trust zones*. All mixes within a trust zone are operated by a single provider under a single jurisdiction. Typically, the mixes of a trust zone are hosted in the same data center. (Because a data center can serve a large user population, small countries may have a single provider/data center, while large countries/jurisdictions would be served by multiple providers/data centers.) To bootstrap, *Herd* needs at least one zone with one mix. A new zone requires a minimum set of clients before establishing calls, which can be achieved by requiring a minimal number of initial subscribers or by splitting an existing, large zone into several partitions.

3.1 Overview

Herd adopts a hybrid of P2P and infrastructure-based architecture for performance and scalability reasons. First, a modest number of well-provisioned, dedicated mixes are connected to the core of the Internet around the world, and maintain a fully connected network of links between each other. Second, the clients and SPs served by a particular mix form a two-level, hierarchical network, with SPs as the interior nodes. SPs are highly available clients with low-latency access to the Internet and a public IP address.

All links connecting mixes, SPs and clients carry bi-directional encrypted traffic, which is padded to a uniform rate. The links connecting clients to an SP or a mix carry traffic at a small multiple of the unit rate u (the payload rate of a single voice call), irrespective of whether the client actively makes a call. All other links carry traffic at a multiple of u. The links connecting SPs to mixes within a given zone have identical rates at any given time; this rate can be changed simultaneously on all such links. Likewise, the rate on links connecting mixes within a zone, and the rate on links connecting mixes between a given pair of zones, respectively, is identical at any time and may be changed only simultaneously. Mixes periodically report their link utilization to a local, zone *directory* and rate adjustments are decided unilaterally for intrazone links, or by the two corresponding directories for interzone links.

Trust and incentive model. We assume that each zone is bound by its local laws and authorities. In particular, zone operators can be compelled by local authorities to release the information available to the mixes in their zone.

Clients and SPs trust the mix they attach to, and thus the authorities within the mix's jurisdiction, with their partial call metadata (source IP address and destination mix, duration of call). Therefore, clients and SPs should select a mix operated by a provider of their choice within a zone whose jurisdiction meets their privacy needs.

A client may select a mix in a location and jurisdiction different from her own, e.g., when she does not trust the local authorities with her partial call metadata. This presumes, however, that local authorities do not systematically block network access to such mixes. To circumvent censorship, *Herd* could rely on SPs with unpublished IP addresses (like Tor bridges) and obfuscate client traffic. Applying obfuscation mechanisms like Tor's obfsproxy [11] to *Herd* is the subject of future work. A key challenge is that appropriate cover traffic must sustain a minimum rate of one VoIP call at all times to provide obfuscation.

A client may either connect to an SP or directly to a mix. We assume there is a business model to cover the cost of operating mixes and possibly to incentivize SP operators. For instance, mixes may charge their clients a subscription fee, possibly using a digital currency such as Bitcoin [42]. Recall that clients trust their mixes, so the payment transaction need not be anonymous (which Bitcoin is not), unless the client wishes to keep her identity hidden from the mix. The mix may in turn grant the operators of SPs free calls or pay them a fee, as an incentive to relay calls from clients.

SPs learn the IP addresses of their connected clients; however, they cannot detect when a client makes a call or with whom. We assume SPs act rationally, i.e., they perform in a way that will allow them to remain in good standing as an SP for their mix. Specifically, mixes monitor and reject SPs with insufficient availability or significant packet loss/jitter.

Clients connect to *Herd* continuously (modulo power or network outages), regardless of call activity. Given the low bitrate of VoIP relative to typical flat-rate broadband capacity, chaffing at a small multiple of that rate seems reasonable. Further, VoIP users tend to remain available to receive incoming calls. For example, prior work showed that half of Skype users are available more than 80% of the time [38].

Threat model. We assume an adversary who seeks to infer the IP addresses of the caller and callee of calls made via *Herd*, their time and duration, as well as their content. The adversary is able to observe the time series of encrypted traffic on all *Herd* links as part of a *global, passive traffic analysis attack.* Within a portion of the Internet controlled by the adversary, he can additionally compromise mixes and network components, change their behavior, and modify the time series of encrypted traffic as part of a *local, active traffic analysis attack.* These assumptions are reasonable because an active subversion with global reach is harder to achieve and maintain than a passive one. An active subversion is easier to detect (and thus counteract by local authorities) and typically requires a synchronous control loop. We make the common assumption that the adversaries cannot break the cryptographic primitives or compromise the keys used by clients, SPs, or mixes they do not control.

We assume a PKI that provides a root of trust to authenticate legitimate mixes and zone directories. For instance, the root certificate can be embedded in the *Herd* client software, and a hash of the client software can be published periodically in several trusted outlets.

We assume that legitimate clients and SPs attach to mixes in a trust zone that is friendly to their cause. Clients and SPs can make their choice of a trust zone based on the zone's location, jurisdiction, history and operator.

Anonymity property. Under the threat model described above, *Herd* ensures *zone anonymity* for the caller and callee. That is, assuming a call is known to originate or terminate in a given zone, the communicating party is equally likely to be any of the users attached to the zone.

In the case of an inter-zone call (i.e., caller and callee attach to mixes in different zones), zone anonymity is independent for caller and callee. That is, even if zone anonymity is violated for one participant due to a compromised mix, it continues to hold for the other. However, in case of an intra-zone call (i.e., caller and callee attach to different mixes in the same zone), anonymity is lost for both caller and callee if the zone is compromised. If this is undesirable, users can attach to mixes in two different zones; in this case, *Herd* ensures that only inter-zone calls are established.

Roadmap. In the rest of this section, we discuss the components of *Herd*'s design. First, *layered encryption* ensures bitwise unlinkability and end-to-end confidentiality of VoIP packets. Second, *Herd* uses a *rendezvous* mechanism to establish low-delay circuits while maintaining anonymity for both caller and callee. Third, *traffic obfuscation* is used to thwart traffic analysis attacks, using appropriate mechanisms at clients, SPs, and mixes. Jointly, these components ensure that an adversary can learn only upper bounds on the volume of active calls between any given pair of zones, the volume of calls within a zone, and the maximum number of concurrently active calls among the clients attached to an SP.

Finally, *superpeers* can reduce a mix's client-side bandwidth requirements from linear in the number of online clients to linear in the number of active callers. *Herd* relies on network coding to hide from the untrusted SPs which clients are making calls.

3.2 Layered encryption

Herd uses hop-by-hop and layered encryption over a sender-receiver mutually anonymous circuit as shown in Figure 1. Mixes maintain a Datagram TLS (DTLS) link to all other mixes, SPs maintain a DTLS link to the mix they are attached to, and clients maintain either one such link to a mix, or a small number of links to SPs. All *Herd* traffic is transferred over these links.

As in Tor, mixes, SPs, and clients maintain a long-term identity key pair l used to sign DTLS certificates and their descriptors, and a short-term key pair s used to set up circuits and negotiate symmetric, ephemeral session keys e. Descriptors containing public keys l and s of the zone participants are published in their directory, where they can be queried. Mixes and users communicate via DTLS links encrypted with ephemeral key e, sealing the traffic with perfect forward secrecy. Finally, clients build circuits incrementally, negotiating a symmetric key with each mix on the circuit, one hop at the time, using s over DTLS links. We refer to the *Herd* specifications for more details [19].

Layered encryption provides bitwise unlinkability, and hides content and routing information from both individual mixes and eavesdroppers, while hop-by-hop encryption hides circuit IDs from the latter and provides perfect forward secrecy. (Circuit IDs are needed to lookup the key used to peel off a layer of encryption, so they are transmitted outside of layered encryption.) *Herd* VoIP content is encrypted end-to-end between the caller and callee using a symmetric key negotiated over two circuits concatenated at rendezvous mixes (described below). A complete circuit has five hops (up to seven if optional SPs are used), with typically at least two intra-zone (and thus intra-data center) hops, and for interzone calls, at most three intercontinental hops (at most one if clients select a zone in their respective continent).

For security, simplicity, and future interoperability *Herd* borrows most of its signaling and cryptographic protocol from Tor. This design decision enables *Herd* to build upon years of research and engineering from the Tor community.

Security. Layered and hop-by-hop encryption maintain the following security invariants: **I1:** *The encrypted content on successive links of a circuit is uncorrelated.* **I2:** *The interior mixes on a circuit know only the previous and next hop on the circuit.* **I3:** *The caller's mix knows only the caller and the next mix on the circuit; the callee's mix knows only the callee and the previous mix on the circuit.* **I4:** *A circuit includes two (not necessarily distinct) mixes in each of the caller and callee's zone. Given I2, this implies that, from the perspective of a mix in the caller's zone, the callee is equally likely to attach to any of the mixes in the callee's zone, and vice versa.*

3.3 Rendezvous

The *Herd* rendezvous mechanism connects clients anonymously, independent of their trust zones. This mechanism comprises a per-zone directory server, at least one mix for each of the caller and callee, and for each such mix, a random and not necessarily distinct rendezvous mix in the same zone. The rendezvous mix is selected and trusted by the caller (resp. callee) to initiate/accept calls on her behalf. Each zone directory server stores the rendezvous mixes of all the clients attached to that zone (client's public key and rendezvous mix IP address).

The *Herd* client software comes with a default list of zone directories and public keys used to locate and authenticate zones. Upon joining the system, a client obtains a signed certificate from a zone directory that contains a client ID and the zone's signature. We assume that users exchange certificates prior to making calls using an out-of-band mechanism. We will discuss the join protocol in detail in Section 3.5.

A call is established using the rendezvous mechanism as follows. First, a hidden callee builds a circuit comprising a mix and rendezvous mix in her trust zone and uses it to publish her rendezvous mix in the zone directory. The caller follows the same procedure to select its mix and rendezvous mix. To make a call, a caller looks up the callee's rendezvous mix in the directory of the zone contained in the callee's certificate and initiates a handshake with the hidden callee. If the call is accepted, the two clients communicate via the rendezvous mixes, hence hiding the mixes to which they attach from each other, thus maintaining zone anonymity.

As we discussed in Section 3.1 (anonymity property), zone anonymity is lost for the caller and callee if they both attach to the same, compromised zone. If they want to ensure that their anonymity does not depend on the same trust zone, they may use an alternative, pre-established circuit to a different zone to contact each other as shown in Figure 1. To do so, a caller whose certificate bears the same zone as the callee may chose a different trust zone to establish the call.

Rendezvous enables clients to receive VoIP calls while retaining zone anonymity by interposing a trusted local mix between the client's mix and an untrusted zone. This is because the interposed mix hides the actual entry mix to which a client is attached (either directly or via an SP). Although our rendezvous mechanism is similar to Tor's, *Herd* is the first system to combine it with traffic obfuscation (as we will describe below) in order to achieve zone anonymity.

Security. Rendezvous maintains security invariant **I5:** *Any mix in a client's zone is equally likely to be the client's rendez-vous mix. Thus, knowledge of a client's rendez-vous mix does not reveal any information about which mix the client attaches to.*

3.4 Traffic-analysis resistance

Despite bitwise unlinkability, an adversary can observe and correlate the time series of encrypted packets on different links. Changes in call volume, payload rate (e.g., due to congestion) or manipulation of traffic by an active adversary can cause correlated changes in the time series of encrypted packets along a flow's path. To defeat such *traffic analysis* the anonymity network must craft the time series of packets on each (per-hop) link such that the adversary is unable to infer which clients are communicating. *Herd* does this by adding chaff traffic as needed to pad flows to a constant rate.

Herd uses different padding mechanisms on the clients, SPs and mixes' links. On the clients' links, *Herd* simply maintains *constant chaffing* at a rate sufficient to carry a small number (possibly one) VoIP call(s). Chaff is then replaced with payload traffic when a call is made. On the links between SPs and mixes and among mixes, we use a dynamic chaffing strategy for traffic obfuscation, which can take advantage of temporal and spatial correlation among the payload traffic of groups of clients, in order to resist both short and long-term traffic analysis with low overhead. We describe both methods in more detail below.

3.4.1 Traffic obfuscation on client links

Each client sends and receives at a constant rate equivalent to a small number of voice calls on their SP link(s). To do so, a constant number of encrypted packets equal to the size and rate of the VoIP codec's packets is sent and received per time unit. Chaff packets, which include a sequence number, are substituted when no VoIP packets are to be sent. Both chaff and VoIP packets are encrypted with a symmetric key established between client and mix when the client joined.

Security. Consider the security properties of clients exchanging bidirectional encrypted traffic at a fixed target rate with their SP. Due to congestion and packet loss, the achieved rate on a given link may differ from the target. However, the actual link rate reflects only the capacity and congestion state of the underlying network, and reveals nothing about the payload it carries. Thus, the constant rate traffic is perfectly resistant to passive traffic analysis. Further, actively delaying or dropping traffic on a link does not leak useful information to a network observer, because the downstream node's outgoing stream rate will not be affected (it simply adds more chaff).

3.4.2 Traffic obfuscation on SP links

The links connecting SPs to their mixes carry bidirectional, encrypted and padded traffic at a rate that is a multiple of the VoIP base rate. All links between the mixes of a zone and their SPs carry traffic at the same rate at any given time, but the rate can change over time to accommodate long-term changes in call volume (e.g., based on historical call volume information). Such changes take place at time scales of hours, e.g., to accommodate diurnal load patterns, but do not reveal individual call activity. Rate changes on SP links are orchestrated by the zone directory; mixes periodically report statistics about link utilization to their directory, which then signals them to ramp up/down the rate based on a utilization metric (e.g., average link utilization).

Security. The chaffing reveals no information about individual clients' call activity. However, the sum of the rates on the mix-SP links of a zone reveals an upper bound on the maximal number of active calls originating or terminating in the zone at any given time. This is a deliberate design decision that significantly reduces the client-side bandwidth overhead of mixes, at the cost of revealing an upper bound on the per-zone call volume.

3.4.3 Traffic obfuscation on mix links

The mixes are fully connected by a set of encrypted and padded links. The links connecting mixes within a given zone, as well as the links connecting mixes in a given pair of zones, respectively, carry traffic at the same rate at any given time. However, the rates within a zone and between any pair of zones may change over time to accommodate changes in the aggregate call rate. As above, these rate changes take place on time scales of hours and do not reveal individual call activity. Although rate changes on links spanning a single zone are orchestrated unilaterally by their directory (as for SPs), rate changes on links crossing zones require coordination between the directories of the two zones.

Security. No information is leaked about individual call activity. However, a global observer can determine an upper bound on the volume of calls within a zone and between any given pair of zones. Again, this is a deliberate design decision that significantly reduces inter-mix bandwidth at the cost of revealing some upper bounds on aggregated call volume among large client populations, which can likely be estimated anyhow based on the number of clients and the known call volumes between different parts of the world.

Traffic obfuscation on client, SP, and mix links maintains security invariants **I6:** *The time series of encrypted packets on the links of a circuit is uncorrelated with the VoIP payload carried by the circuit* and **I7:** *Manipulating the time series of encrypted traffic on a network link does not affect the time series of encrypted traffic on downstream links.*

3.5 Join protocol

When a client wishes to join the system, it chooses a zone and is redirected by that zone's directory (hardcoded in the client software) to a mix within the zone. The client then establishes a symmetric key s with the mix, which is used for all subsequent communication with the mix, and forms the outer layer of any onion circuit subsequently established by the client. Finally, the mix either adopts the client with a direct link, or redirects the client to one or more of the superpeers (SPs) connected to the mix. Finally, the client establishes either a single padded connection to the mix, or k padded connections to the SPs, in each case encrypted with symmetric key s. In the case of a mix or superpeer failure, a client contacts another mix in the same zone and re-joins.

3.6 Superpeer architecture

Next, we describe *Herd*'s superpeer (SP) architecture. SPs are an optional component of *Herd*—the system is fully functional if all clients attach directly to mixes. However, SPs can increase *Herd*'s scalability by reducing the client-side bandwidth load of mixes by a factor of up to n/a, where a is the maximum number of active clients (i.e., those making calls) in a zone, and n is the number of online clients attached to a zone. In *Herd*, n/a is likely to be large (above 10), because clients have an incentive to remain online independent of their call activity, both to receive calls at unexpected times, and to prevent long-term intersection attacks.

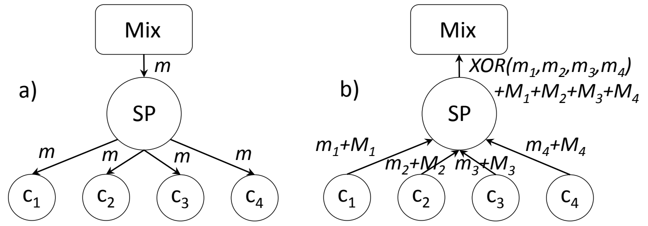

Figure 2: (a) Downstream channel operation: The SP forwards packet m received from the mix to each client. Only the active client can decrypt the packet, others discard the packet as chaff. (b) Upstream channel operation: The SP forwards to the mix the XOR of packets received from clients, concatenated with the list of packet manifests M.

SPs are well-connected, highly-available nodes with a public IP address, who invest resources to support the system altruistically or in exchange for free calls or payment. Like clients, SPs are assumed to be continuously available (modulo power/network outages). SPs are expected to perform in a way that keeps them in good standing with their mix (else they will be blacklisted), but are not otherwise trusted.

Note that an SP cannot decrypt the packets it forwards between client and mix. Moreover, it cannot distinguish chaff packets from VoIP payload packets, and thus cannot determine whether a client is active or not.

3.6.1 Routing and network coding

An SP with c attached clients must route information from its mix link with a rate sufficient to carry r calls to its c client links, and vice versa, where $c \geq r$. Our goal is to do this without leaking information about clients' activity to the untrusted SPs.

Downstream. Let us first consider the downstream direction. The SP receives incoming packets in rounds of r packets, where each packet contains VoIP payload for one of the r active clients. The mix must forward these packets so that each active client receives the packet destined for it. For this purpose, the clients attached to an SP are partitioned into r *channels*, and each packet in a round of r is forwarded to a different channel. Each packet includes the initialization vector (IV) used to encrypt it.

Each channel supports at most one active call. The active client can decode the packet it receives, while the remaining clients in the channel cannot—they discard the packet as chaff. To reduce the risk of call blocking when the channel to which a client attaches is occupied, clients attach to $k > 1$ channels (possibly at multiple SPs). Figure 2(a) illustrates the operation of a channel in the downstream direction.

Upstream. In the upstream direction, in each round, the SP receives a packet from each client attached to a channel. Because at most one client can be active in each channel, we can use a simple form of network coding. The SP simply forwards to the mix the XOR of the client packets received in each of the r channels, of which at most one is a VoIP packets and the rest are chaff. Because the ciphertext of the chaff packets from the idle clients is predictable to the mix (the cleartext contains a sequence number and the packets include the IVs), the mix can trivially recover the r payload packets from the r XORs it receives.

Along with each XOR packet, the client forwards the packet manifests that were attached to each client packet included in the XOR. Each of these manifests is 4 bytes long, encrypted with s, and includes the client's id within the channel, packet sequence number, and a signaling bit (described below). The information included in the manifest enables the mix to quickly decode the XOR even in the case of lost or delayed packets, at the expense of slightly more bandwidth on the mix–SP links for the manifests. Figure 2(b) illustrates the operation of a channel in the upstream direction.

Note that a malicious SP or client could deny service by sending an incorrect manifest or an unexpected chaff packet. In this case, the mix asks the SP to send the full packets (which the SP is expected to buffer for a couple of rounds) from which the packets were computed, enabling the mix to identify, drop, and blacklist the culprit's *Herd* account.

3.6.2 Signaling

Next, we describe how mix and client signal incoming and outgoing calls, in such a way that the SP does not learn when a call starts or ends, or which client is calling or being called. In the case of an incoming call, the mix simply chooses an available channel to which the callee attaches (if any), and encrypts downstream packets in the channel with the key s shared with the callee. The callee, which like every client, tries to decrypt every incoming packet on each channel, is able to decrypt the information signaling an incoming call, responds on the same channel, and once it has accepted the call, receives VoIP packets for the duration of the call.

In the case of an outgoing call, the caller sets the signaling bit in the manifest of the chaff packets it sends to the mix via the SPs. Note that the caller does not know which, if any, channel is available. The mix will respond on an available channel to which the caller attaches (if any), as in the case of an incoming call. The caller then provides signaling information for the desired outgoing call on the available channel.

3.6.3 Channel allocation

Next, we describe how the mixes allocate joining clients to channels, and incoming/outgoing calls to channels. Without loss of generality, we consider a single mix. (The set of clients and SPS attached to different mixes are disjoint.)

Each client attaches to k channels. As we will show, $k = 3$ provides a good balance between blocking probability and client bandwidth overhead. The number of channels C per zone is chosen to exceed the expected number of active calls a within the zone during the busiest period, i.e., $C \geq a$.

One might consider changing C dynamically to accommodate diurnal load variations efficiently. However, such reconfigurations have significant overheads due to clients changing channels, and care must be taken not to leak information that could aid in long-term intersection attacks. Therefore, in our design, configuration changes occur infrequently and under administrator control, as necessary to accommodate changes in the client population and call volume.

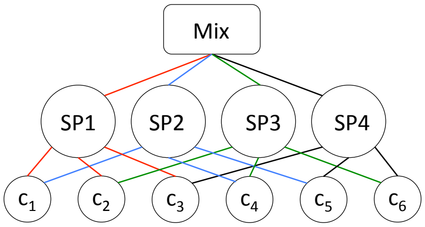

Figure 3: SP configuration with k=2, N=6, C=4, where each of the four SPs supports one channel.

The mix allocates a new client to k distinct channels. We use a greedy algorithm that picks k distinct channels randomly from the least occupied channels. Figure 3 shows an example configuration with 4 SPs, 6 clients, and $k = 2$.

An ideal configuration should enable *any* subset of C clients to make concurrent calls, thus imposing no constraints on the set of active clients. The toy configuration in Figure 3 has this property, but unfortunately, it is infeasible in large configurations for reasonable choices of k and C. In practical configurations, the maximal number of concurrent calls varies among different subsets of C clients, depending on how many channels the subset collectively attaches to.

Note that these connectivity limitation could be overcome trivially by dynamically changing the association of clients to channels, for instance by changing the "routing tables" at the SPs. Unfortunately, dynamic routing inevitably leaks information related to call activity, which could be used as part of an intersection attacks. Therefore, *Herd* uses static allocations of clients to channels. We will show experimentally in Section 4.1.6 that, despite the resulting connectivity constraints, the blocking probability remains good in practice.

Dynamic channel allocation: When an outgoing/incoming call starts, the mix must dynamically allocate to the call an available channel (if any) among the k channels to which the caller/callee attaches. This is an instance of the online bipartite matching problem. A simple, optimal algorithm exists for this problem [36]. It initially ranks all channels randomly, and then allocates the available channel with the highest rank in each step.

3.6.4 Superpeer security

With the routing and coding scheme described above, a set of colluding SPs can learn the static assignment of clients to channels. The channel assignment reveals some constraints on the possible sets of active clients. Clearly, at most C clients can be active within the zone at any one time. Due to connectivity constraints, the maximal number of active calls for a specific set of C clients may be lower than C and known to an adversary. However, without information about call activity, this information is of little use to an adversary. Call activity cannot be observed directly or indirectly either by the SPs or by an adversary with access to the network.

A compromised SP can drop or delay encrypted packets it forwards. However, mixes blacklist SPs that fail to meet a high standard of packet loss rate and jitter. Legitimate SPs that fail to meet the standard due to an unreliable network may require their clients to use error-correcting codes on their encrypted channels to the mix, thus reducing the effective loss rate to acceptable levels.

Superpeers maintain security invariant **I8:** *Superpeers do not know which of its clients are active at any time. Moreover, an SP cannot manipulate the encrypted traffic of its clients without being banned.*

3.7 Security

Next, we argue informally that *Herd*'s security invariants (summarized below) provide zone anonymity.

I1: The encrypted content on successive links of a circuit is uncorrelated.

I2: The interior mixes on a circuit know only the previous and next hop on the circuit.

I3: The caller's mix knows only the caller and the next mix on the circuit; the callee's mix knows only the callee and the previous mix on the circuit.

I4: A circuit includes two (not necessarily distinct) mixes in each of the caller and callee's zone. Given I2, this implies that, from the perspective of a mix in the caller's zone, the callee is equally likely to attach to any of the mixes in the callee's zone, and vice versa.

I5: Any mix in a client's zone is equally likely to be the client's rendez-vous mix. Thus, knowledge of a client's rendez-vous mix does not reveal any information about which mix the client attaches to.

I6: The time series of encrypted packets on successive links of a circuit is uncorrelated with the VoIP payload carried by the circuit.

I7: Manipulating the time series of encrypted traffic on a network link does not affect the time series of encrypted traffic on downstream links.

I8: Superpeers do not know which of its clients are active at any time. Moreover, an SP cannot manipulate the encrypted traffic of its clients without being banned.

I2–I5 jointly maintain zone anonymity for a caller even if the callee's zone is compromised, and vice versa. I1 and I6 maintain zone anonymity in the presence of a global, passive eavesdropper. I7 maintains zone anonymity despite an active network attacker. And invariant I8 maintains zone anonymity despite compromised SPs. A formal analysis of *Herd*'s security properties remains as future work.

Next, we discuss specific attacks within our threat model and how *Herd* defends against them.

Passive traffic analysis attack. The traffic rate on links does not depend on individual payload flows. Short-term traffic analysis is therefore unproductive. Furthermore, the overlay topology is independent of the underlying workload. Therefore, the adversary cannot tell which clients are communicating, even over long periods.

Active traffic analysis attack. Here, the adversary manipulates the flow of encrypted, chaffed *Herd* traffic by delaying, dropping, or replaying packets, or by shutting down or otherwise causing the intermittent failure of clients, SPs, or mixes. Doing so has no impact on the rate of the downstream chaffed traffic, so it does not help the adversary to trace payload flows. For that, the adversary would require access to a client's payload flow at the other end of a *Herd* circuit. However, this case is ruled out by our threat model.

Moreover, *Herd*'s design makes it hard to exploit SPs for active attacks even if the attacker has access to the payload flow on the other end, because mixes ban SPs that manifest significant jitter or packet loss.

Compromised caller/callee. The rendezvous mechanism ensures that compromised callers or callees cannot learn anything other than the zone of their communication partner from the system. However, a compromised participant could conceivably try to estimate the distance to their peer by measuring the round-trip latency on the audio channel. As part of future work, we are planning to investigate if such an attack is feasible. If so, it should be possible to add artificial delay up to the boundaries of the call's MOS quality level, as predicted by the E-Model [34].

Long-term intersection attacks. In Section 4.1.5, we will show experimentally the power of long-term intersection attacks against unanonymized voice calls. *Herd* makes such attacks unproductive, because it makes it impossible to observe when a user makes a call. Since users are online virtually all the time, an adversary cannot even observe significant periods during which a client could not make a call.

Sybil attacks. Like all open anonymous communication systems, *Herd* is susceptible to Sybil attacks, where an adversary controls a large number of clients or superpeers. SPs cannot observe which of their clients are active. Therefore, an adversary who controls SPs can learn only static, weak constraints on the possible sets of concurrently active clients, which are not useful without information about call activity.

If the adversary manages to control a large fraction of the clients attached to a zone, he is able to reduce the anonymity of the remaining legitimate clients proportionally. Since zones serve a large geographic area and user community, however, such an attack would be difficult even with large amounts of resources. Another approach for an adversary is to control all but one of the clients within an SP channel, leaving the remaining legitimate client as the only possible active user. However, such an attack would be difficult because the mix controls which SPs a client attaches to.

An adversary can also use Sybils to deny service. As described above, a misbehaving client or SP can be detected and blacklisted by the mix. To circumvent the blacklist, the adversary has to register a new account, from a new IP address and using a different payment channel, making such an attack expensive. By charging a one-time sign-up fee, the system can further increase the cost of such an attack, although it is impossible to eliminate it completely.

3.8 Herd prototype

Our prototype implementation consists of a core *Herd* module written in ClojureScript ($3,289$ lines) and a network module written in C ($3,159$ lines). The core executes on the node.js JavaScript runtime. In addition, the implementation relies on the OpenSSL and curve25519 libraries for TLS and cryptography support.

We implemented the core in a language compatible with JavaScript to enable future distribution of the *Herd* client via standard, encrypted web channels, and execution within a browser. The network component ensures consistently timed packet transmissions, despite garbage collection or other events in the managed JavaScript runtime. The mix, SP and client software currently run as standalone programs on Linux. Porting the software to MacOS or Windows is a matter of preparing appropriate installers. Finally, our *Herd* client supports the Jitsi VoIP client [6] and should support other SIP-compliant clients with minimal effort.

3.9 Summary of contributions

In summary, the key contributions of *Herd*'s design are as follows. 1) *Herd* introduces *zone anonymity*, which guarantees that a user's anonymity depends only on the their own choice of a zone provider, independent of the choice of their communication partners and other users. 2) *Herd* employs a novel hybrid architecture, which shifts resource requirements from its trusted infrastructure to untrusted superpeers without affecting anonymity. Thus, *Herd* can take advantage of untrusted peers willing to contribute resources (for free or in exchange for free calls) to increase scalability and reduce operational cost. 3) *Herd* exploits the specific workload characteristics of VoIP traffic to provide traffic-analysis resistant anonymity with low bandwidth cost.

4. EVALUATION OF HERD

To begin our evaluation, we show that when a traffic-analysis resistant anonymity network is not used, a powerful adversary could trace 98.3% of voice calls performed among millions of users in our mobile phone call trace.

We then conduct trace-driven simulations and experiments with our *Herd* implementation to compare the anonymity, scalability, and call quality of *Herd* and Drac. We find that, in the median case, *Herd* supports anonymity sets comparable to Drac and has comparable call quality, but reduces client bandwidth by up to two orders of magnitude. *Herd* requires some dedicated infrastructure. However, we show that untrusted superpeers that contribute resources for free or in exchange for free calls can reduce *Herd*'s operational costs from $10-100 to $0.1-1.14 per user and month.

4.1 Anonymity and scalability

We rely on a real trace of phone calls among millions of mobile subscribers to compare the anonymity provided by Tor, Drac, and *Herd*, and the scalability of Drac and *Herd*. Here, we use trace-driven simulations instead of testbed experiments because we are interested in comparing these systems under a large, real voice workload.

4.1.1 System models

We use simulation models of Drac, *Herd* and Tor, where the first two use a chaffing strategy as described below, and the latter uses none.

Drac. Drac maintains one chaffing connection for each link within a social network, thus hiding the call patterns within the social network. As a result, *Drac's bandwidth requirements are proportional to the degree of nodes in the social network, i.e., the size of users' contact lists.*

Herd. *Herd* relies on a small, constant number of connections at the clients (3 in our simulations), and it also employs SPs and carefully orchestrates rate changes on the links among mixes to reduce the load on its trusted infrastructure. By doing so, *Herd achieves zone anonymity while minimizing bandwidth requirements.*

Tor. *Tor does not employ chaffing and so does not offer any resistance to traffic analysis.*

4.1.2 Datasets and methodology

Mobile dataset. To determine the anonymity and scalability of the different system models, we use a month of data from August 2010, comprising 370 million calls among 10.8 million users of a large cellular provider (IRB approved). The dataset contains call times, durations, and salted hashes of caller/callee telephone numbers, and indicates which hashes are subscribers of the cell provider.

We use only the calls between subscribers of the carrier. This gives us bidirectional information for each subscriber call, and consists of subscribers who would reasonably use a single zone in *Herd*.

Social network datasets. The call partners recorded over a one month period may significantly underestimate the actual size of contact lists (which determines Drac's bandwidth overhead). Thus, we complement the mobile dataset with social network data to explore Drac's properties under a range of workloads. We use Twitter and Facebook datasets with 54 million, and $1,165$ users, respectively [23, 41].

Methodology. We evaluate our system models using trace-based simulations. When evaluating anonymity for Tor (Section 4.1.4), we conduct an intersection attack using 1-second intervals for call start/end times. For the cost analysis in Section 4.1.6, we aggregate the call start and end times into one-minute bins to improve the runtime of our simulations. We determine the peak number of calls and statically provision the *Herd* topology of mixes and SPs accordingly so the network has enough capacity to handle the peak call rate.

4.1.3 Goals

Anonymity. We characterize the anonymity provided by each system as the number of clients who could be the corresponding party, given one known party in a call (anonymity set), under the threat model described in Section 3. In Drac, the size of the anonymity sets corresponds to the number of clients reachable via H hops over the social network. Al-

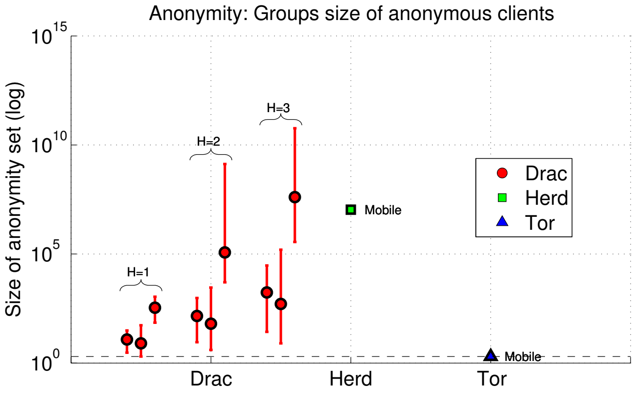

Figure 4: Size of anonymity sets for *Drac*, *Herd*, and *Tor*: median, 10th, and 90th percentiles (left to right). For Drac, we show the results for the *Mobile*, *Twitter*, and *Facebook* datasets and $H = 1, 2, 3$, and for Tor, the results for the Mobile dataset. *The size of anonymity sets in Herd is independent of the node degree and workload.*

though this size can be as large as the entire social network for a sufficiently large H, only clients relatively nearby in the social graph can expect latencies low enough for VoIP (unlike in *Herd*, all hops in Drac typically involve last-mile links). $H = 0$ means that the caller and callee communicate directly and so have an anonymity set of 1. Therefore, we show results only for $H = 1, 2, 3$. In *Herd*, the size of the anonymity set corresponds to the number of subscribers in the mobile dataset, who are assumed to be in a single zone. In Tor, the anonymity set is the set of potential communication partners identified by an intersection attack [22].

Client bandwidth requirement. The bandwidth requirement is the number of chaffing connections multiplied by the rate of a VoIP call using the G.711 codec (8KB/s). As chaff is substituted with payload during a call, the bandwidth requirement accounts for both chaff and payload traffic.

Operational Costs. We evaluate *Herd*'s operational costs in dollars per user/month using EC2 pricing [2].

4.1.4 Results: Tor's Anonymity

In the absence of chaffing, a passive attacker can correlate call start and end times to identify which partners are communicating via an intersection attack [22]. That is, the attacker sees that sets of users start and end calls simultaneously, and attempts to identify pairs of communicating clients from this set. To confirm whether a single pair of users, (u, v), is communicating, the attacker takes the intersection of the sets of users with the same call start/end times as (u, v). When the intersection set is size 2, the attacker has confirmed these communication partners.

We find that, despite the large number of subscribers in our dataset, an intersection attack is highly successful for real call patterns. In particular, after one month, an attacker can trace 98.3% of all calls when using 1-second granularity for tracking call start and end times.

Summary. Real voice communication patterns are highly susceptible to traffic analysis. Anonymity systems that do not defend against such analysis cannot provide practical anonymity. For this reason, we focus the rest of our evaluation on Drac and *Herd*.

4.1.5 Results: Anonymity/Scalability of Drac and Herd

Drac. The effective size of the anonymity sets in Drac correspond to the number of clients that can be reached within H hops in the social network, where H is the maximal number of hops that still yield latencies tolerable for VoIP. We obtain the size of the anonymity sets for $H = 1$ empirically from the different social network data sets, and estimate the sizes for $H = 2, 3$ using the median node degrees.

The median anonymity set sizes for the Mobile, Twitter, and Facebook datasets shown in Figure 4 are 12, 8, and 343 for $H = 1$, and 1728, 512, and 40 million for $H = 3$, respectively.

Drac's bandwidth demand for its clients is proportional to the median node degree. As shown in Figure 5, the median bandwidth required with the Mobile, Twitter and Facebook datasets are 96KB/s, 64KB/s, and 2.6MB/s, respectively, and the maxima are 12MB/s, 39MB/s, and 6.2GB/s.

Herd. *Herd*'s anonymity set sizes depend only on the number of clients in a zone. Consequently, the size of *Herd*'s anonymity set with the mobile workload corresponds to 10.8 millions (the number of subscribers), as we see in Figure 4. Based on our trace-based simulations, we find that 1,000 mixes is sufficient to handle the load from these users. Thus, we expect to be able to support on the order of 10M users in a large cloud provider like EC2.

Herd clients keep a small, constant number of chaffing connections k to the SPs (or only one connection if a client connects directly to a mix). As we have seen in Section 3 (and will confirm below) three connections are sufficient to achieve a very low probability for a call to block due to all its channels being busy. Even with three channels, a client's bandwidth requirement is only 24KB/s (3*8KB/s).

Summary. We showed that *Herd* has significantly lower bandwidth requirements than Drac. To achieve this, *Herd* relays VoIP traffic through a dedicated infrastructure, incurring additional costs and delays. In the rest of this section, we evaluate *Herd*'s operational costs and call quality.

4.1.6 Results: Herd's operational costs

We now evaluate the traffic volume on the *Herd* mixes with and without SPs, with interzone traffic ranging from 10-100%, and the resulting monetary cost per user. We use one week of the phone call data to drive these simulations; results for other weeks were similar.

Impact of using SPs instead of direct connections. We ran SP simulations with 100 SPs per mix and 100 clients per SP, and varied the number of clients per channel (between 5 and 50) and the number of channels each client attaches to (2 and 3). A call is blocked if there are no available channels at the caller or callee's end. In our simulations, the blocking rate for 2 channels varied between 5% and 0.1% with 50 and 5 clients per channel, respectively. We observed that the

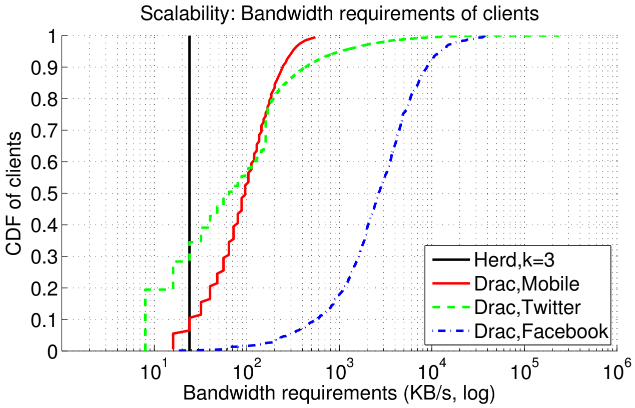

Figure 5: Bandwidth demands of Drac and Herd clients. Herd *clients keep a small, constant number of chaffing connections, leading to uniformly low bandwidth demand.*

average blocking rate decreased by an order of magnitude when clients attached to 3 channels instead of 2.

SPs have the potential to greatly offload mixes. In our simulations, these savings varied between 80% and 98% with 5 and 50 clients per channel, respectively. This low blocking rate and high savings are explained by low instantaneous system utilization for voice workloads—in the day-long trace we considered, the peak duty cycle (i.e., peak fraction of users making calls simultaneously) was 1.6%.

Dollar cost per user. We now use trace-based simulation results to estimate the cost of running *Herd* for a large set of clients with Amazon EC2 mixes. We use the AWS calculator to explore the cost of several points in the design space.

The cost ranges from \$0.10 to \$1.14 per month per subscriber. The low end of the range corresponds to a call volume of 1% of users simultaneously making calls at any time and only 10% interzone calls; while the high end of range corresponds to 2% of the users making calls at any time (larger than the peak value in our traces described above) and 100% interzone calls. The reason for the relatively low cost is that intrazone traffic in EC2 (i.e., data traffic that stays within the same data center) does not incur charges, interzone traffic (i.e., between EC2 data centers) incurs low charges, and traffic to SPs and clients costs the most. Because most traffic is between mixes in the same zone, and traffic leaving the cloud is relatively small with SPs, costs are generally low. Thus, we believe that *Herd*'s design is sufficiently affordable when scaled to millions of users.

The above estimates rely on SPs that provide their services for free, and that we do not charge SPs for their use of *Herd*. As an additional incentive, we can pay SPs for their contribution to the system: in this case, the cost per paying subscriber is an additional \$0.14 per dollar we pay SPs.

Note that choosing not to include SPs in a deployment can have a significant impact on costs. Our estimates show that it will cost two orders of magnitude more per user to run *Herd* (\$10-100 per month per user).

Summary. Providing strong anonymity for VoIP in a purely infrastructure-based system is relatively expensive, but a set of well-connected, altruistic, but otherwise untrusted hosts can substantially reduce costs.

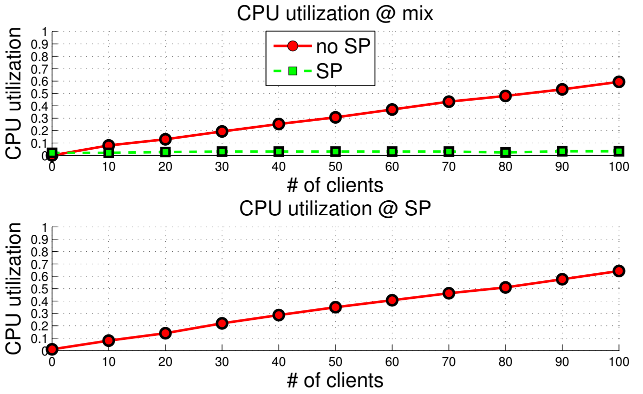

Figure 6: CPU utilization of a mix (top) and SP (bottom), as a function of the number of clients supported.

4.2 Prototype performance

Next, we present an experimental evaluation of *Herd*'s CPU, memory, and bandwidth consumption.

Goal and methodology. We evaluate the performance of the *Herd* implementation described in Section 3.8 on a mix with and without a superpeer, while varying the number of clients between 0 and 100. The mix and SP were each hosted on a Dell OptiPlex 980 (2.8Ghz Intel Core i5-760, 8GB RAM) and the clients were co-hosted on another machine of the same model (3.2GHz Intel Core i5-650, 8GB RAM). We launched three, 60-second runs for each configuration and used the Linux *time* command to measure the CPU utilization of the *Herd* network process, which performs all data-plane processing. The CPU utilization is the sum of the user and system CPU times divided by the elapsed real time.

Results. SPs reduce resource demand on the mixes. As expected, using an SP reduces the bandwidth required at the mix to support n clients by a factor of nearly n/a, where a is the number of concurrent active calls (one in our experiment). Moreover, as shown in Figure 6, without an SP, the mix's network process has a CPU utilization of 59% for 100 clients, while an SP with one chaffed connection between mix and SP reduces that utilization to only 3%. The marginal CPU utilization for supporting an additional client is .01% and .6% with and without the SP, respectively. The reason is that the network coding for an SP requires far fewer CPU cycles than maintaining a chaffed connection with multiple clients. Finally, the memory utilization on the mix is low regardless of the configuration. For example, the mix without an SP uses 3.4MB of virtual memory for 100 clients.

Summary. SPs reduce the mixes' bandwidth and CPU requirements by an order of magnitude.

4.3 Perceived call quality

We now present an experimental comparison of Drac and *Herd*'s perceived call quality.

4.3.1 Goal: Perceived call quality

We use E-Model to evaluate the quality of actual VoIP calls performed by the authors. E-Model is an analytic model of call quality defined by the International Telecommunication Union (ITU), which calculates the Rating factor (R-factor), a simple measure of voice quality [34]. The R-factor ranges from 0 to 100 and directly determines the Mean Opinion Score (MOS), which ranges from 1 (poor) to 5 (perfect). For VoIP environments, the R-factor is defined in terms of mouth-to-ear delay and packet loss. We refer to Cole et al. for more details [25].

4.3.2 Systems, deployment, and methodology

Drac. Since a Drac implementation is not available, we used *ping* to measure the direct latency between contacts ($H = 0$). *Drac calls between contacts are established directly.*
***Herd*.** For *Herd*, we used the implementation presented in Section 3.8 in our deployment. *Herd routes traffic via sender-receiver, mutually anonymous circuits with SPs.*
Deployment. To evaluate *Herd*'s call quality in practice, we deployed a complete *Herd* system with 4 zones, and requested that volunteers make one-way calls between every zone pairs. Our deployment currently comprises of 8 mixes and rendezvous, 2 directories, and 4 SPs hosted on Amazon EC2. We used ClojureScript 0.0-2644, node.js 0.10.12, Jitsi 2.4, and EC2 small and medium instances with Core OS 522.6.0.
Methodology. For this call quality experiment, each of our 4 volunteers, who connected from university networks, called one other volunteer located in each of the 4 zones (12 calls in total), and listened to a pre-recorded voicemail hosted at the callee. Although VoIP calls are full-duplex, transmissions in each direction are logically independent and so our experiments transmit voice in only one direction. We borrowed this methodology from the industry standard ITU [5].

The *Herd* clients in this experiment conduct the following tests. First, upon receiving an incoming call, each client automatically plays a pre-recorded voicemail. (We host the voicemails at the volunteers to ensure that calls experience realistic end-to-end loss and latency.) Second, an active client periodically (i.e., every second) measures the end-to-end latency and packet loss rate between caller and callee.

4.3.3 Results: Perceived call quality

We show the perceived call quality for Drac with $H = 0$ and *Herd* in Figure 7. In the absence of packet loss, latencies between Europe, North America, and South America were of high or perfect quality, and latencies between Australia and the rest of the world were of medium quality. In our experiments, the packet loss never exceeded a few percents which, according to E-Model, would result in the loss of at most one MOS level for both systems.
Summary. *Herd* achieves modest additional latency while imposing significantly less bandwidth requirements on clients, and much larger anonymity sets for direct calls than in Drac (i.e., 10.8M vs. 1).

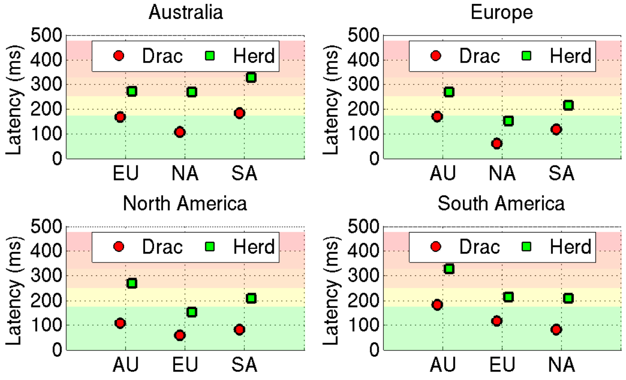

Figure 7: One-way latencies for *Drac* with $H = 0$ and *Herd*, among locations in Australia (AU), Europe (*EU*), North America (*NA*), and South America (*SA*). The colored horizontal bands from top to bottom correspond to the MOS bounds for *poor*, *low*, *medium*, *high*, and *perfect*, perceived quality. *Herd incurs a small, additional one-way latency of approximately 100ms over Drac.*

5. CONCLUSION & FUTURE WORK

Herd is a novel anonymity network for VoIP that provides strong anonymity in the face of powerful adversaries, with good call quality and modest cost. *Herd*'s architecture relies on trusted infrastructure mixes to achieve zone anonymity, where a user's anonymity among the set of clients attached to a zone depends only on that user's choice of a provider. *Herd* uses network coding in a novel way to shift resource requirements from its trusted infrastructure to untrusted superpeers, allowing it to take advantage of resource contributions by untrusted parties to increase its scalability and reduce cost. *Herd* relies on layered encryption and chaffed connections to conceal call activity from powerful adversaries with global network access, while exploiting VoIP traffic characteristics to reduce bandwidth demand. We demonstrated that *Herd* is a practical solution with good performance using an implementation deployed in Amazon EC2 and trace-based simulation with empirical call data. Currently, we are migrating mixes to diverse hosting centers in preparation for a public release. Future work includes supporting group and video calls as well as porting *Herd* to browsers and adapting it to mobile platforms.

Acknowledgements. We thank the anonymous reviewers, our shepherd Stefan Savage, and Bobby Bhattacharjee for their helpful feedback. We are grateful to Saptarshi Ghosh, Krishna Gummadi, Mainack Mondal, and Bimal Viswanath for sharing their datasets with us. Finally, we thank Zheng Leong Chua, Italo Cunha, and Vincent Gramoli for helping us test our prototype *Herd* implementation. This work was supported by the Max Planck Society, the European Research Council (ERC) under the imPACT Synergy Grant No. 610150, and the NSF under Grant No. CNS-1318396.

6. REFERENCES

[1] A potential technique to deanonymise users of the TOR network. http://www.spiegel.de/media/media-35538.pdf.

[2] Amazon Web Services Simple Monthly Calculator. http://calculator.s3.amazonaws.com/index.html.

[3] HideMyAss. http://blog.hidemyass.com/2011/09/23/lulzsec-fiasco/.

[4] Inside TAO: Documents Reveal Top NSA Hacking Unit. http://www.spiegel.de/international/world/the-nsa-uses-powerful-toolbox-in-effort-to-spy-on-global-networks-a-940969.html.

[5] International Telecommunication Union (ITU). http://www.itu.int/.

[6] Jitsi. http://www.jitsi.org.

[7] Microsoft releases 2013 Law Enforcement Requests Report. http://www.microsoft.com/about/corporatecitizenship/en-us/reporting/transparency/.

[8] NSA and GCHQ target Tor network that protects anonymity of web users. http://www.theguardian.com/world/2013/oct/04/nsa-gchq-attack-tor-network-encryption.

[9] NSA collecting phone records of millions of Verizon customers daily. http://www.theguardian.com/world/2013/jun/06/nsa-phone-records-verizon-court-order.

[10] NSA Prism program taps in to user data of Apple, Google and others. http://www.theguardian.com/world/2013/jun/06/us-tech-giants-nsa-data.

[11] obfsproxy. https://www.torproject.org/projects/obfsproxy.html.en.

[12] Silent Circle. https://silentcircle.com/.

[13] User's guide for PRISM Skype collection. http://www.spiegel.de/media/media-35530.pdf.

[14] For Syria's Rebel Movement, Skype Is a Useful and Increasingly Dangerous Tool, 2012. http://www.nytimes.com/2012/12/01/world/middleeast/syrian-rebels-turn-to-skype-for-\\communications.html.

[15] TOR Fone, 2012. http://www.torfone.org/.

[16] Creating a low-latency calling network, 2013. http://www.whispersystems.org/blog/low-latency-switching/.

[17] The future of the internet: Balkanization and borders, 2013. http://ideas.time.com/2013/10/11/the-future-of-the-internet-\\balkanization-and-borders/.

[18] Freedom on the Net 2014, 2014. https://freedomhouse.org/report/freedom-net/freedom-net-2014.

[19] Herd source code and specifications, 2015. https://aqua.mpi-sws.org.

[20] A. Back, I. Goldberg, and A. Shostack. Freedom systems 2.1 security issues and analysis. White paper, Zero Knowledge Systems, Inc., May 2001.

[21] O. Berthold, H. Federrath, and S. Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In *PET*, 2000.

[22] O. Berthold, A. Pfitzmann, and R. Standtke. The disadvantages of free MIX routes and how to overcome them. In *Workshop on Designing Privacy Enhancing Technologies*, 2001.

[23] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM*, 2010.

[24] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *CACM*, 24(2), Feb. 1981.

[25] R. G. Cole and J. H. Rosenbluth. Voice over ip performance monitoring. In *Proc. of SIGCOMM CCR*, Apr. 2001.

[26] G. Danezis, C. Diaz, C. Troncoso, and B. Laurie. Drac: An architecture for anonymous low-volume communications. In *PET*, 2010.

[27] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proc. of IEEE S&P*, May 2003.

[28] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proc. of USENIX Security Symposium*, August 2004.

[29] Facebook. Anatomy of Facebook. https://www.facebook.com/notes/facebook-data-team/anatomy-of-facebook/10150388519243859.

[30] M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *CCS*, 2002.

[31] M. Gabielkov, A. Rao, and A. Legout. Studying Social Networks at Scale: Macroscopic Anatomy of the Twitter Social Graph. In *Proc. of SIGMETRICS*, 2014.

[32] C. Gülcü and G. Tsudik. Mixing E-mail with Babel. In *Proc. of NDSS*, 1996.

[33] H.-C. Hsiao, T. H.-J. Kim, A. Perrig, A. Yamada, S. C. Nelson, M. Gruteser, and W. Meng. LAP: Lightweight anonymity and privacy. In *IEEE Security and Privacy*, 2012.

[34] ITU-T. Recommendation G.107, the e-model, a computational model for use in transmission planning, December 1998.

[35] A. Jerichow, J. Müller, A. Pfitzmann, B. Pfitzmann, and M. Waidner. Real-Time MIXes: A Bandwidth-Efficient Anonymity Protocol. *IEEE Journal on Selected Areas in Communications*, 16(4), 1998.

[36] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. of STOC*, 1990.

[37] S. Le Blond, D. Choffnes, W. Zhou, P. Druschel, H. Ballani, and P. Francis. Towards efficient traffic-analysis resistant anonymity networks. In *SIGCOMM*, 2013.

[38] S. Le Blond, C. Zhang, A. Legout, K. Ross, and W. Dabbous. I know where you are and what you are sharing: exploiting P2P communications to invade users' privacy. In *Proc. of IMC*, November 2011.

[39] M. Liberatore, B. Gurung, B. N. Levine, and M. Wright. Empirical tests of anonymous voice over IP. *J. Netw. Comput. Appl.*, 34(1):341–350, Jan. 2011.

[40] U. Möller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster Protocol — Version 2. IETF Internet Draft, 2003.

[41] M. Mondal, Y. Liu, B. Viswanath, K. P. Gummadi, and A. Mislove. Understanding and Specifying Social Access Control Lists. In *Proc. of SOUPS*, 2014.

[42] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008.

[43] A. Panchenko, L. Pimenidis, and J. Renner. Performance analysis of anonymous communication channels provided by tor. In *Proc. of Availability, Reliability and Security*, 2008.

[44] A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-MIXes: Untraceable communication with very small bandwidth overhead. In *Proc. of the GI/ITG Conference on Communication in Distributed Systems*, 1991.

[45] R. Sherwood, B. Bhattacharjee, and A. Srinivasan. P5: A protocol for scalable anonymous communication. In *Proc. of Security and Privacy*, 2002.

[46] A. Shostack and I. Goldberg. Freedom systems 1.0 security issues and analysis. White paper, Zero Knowledge Systems, Inc., October 2001.

[47] D. I. Wolinsky, H. Corrigan-Gibbs, B. Ford, and A. Johnson. Dissent in Numbers: Making Strong Anonymity Scale. In *Proc. of OSDI*, 2012.