Herbivore: A Scalable and Efficient Protocol for Anonymous Communication

Sharad Goel, Mark Robson, Milo Polte, and Emin Gün Sirer

Cornell University, Ithaca NY 14850, USA

Abstract. Anonymity is increasingly important for networked applications amidst concerns over censorship and privacy. In this paper, we describe Herbivore, a peer-to-peer, scalable, tamper-resilient communication system that provides provable anonymity and privacy. Building on dining cryptographer networks, Herbivore scales by partitioning the network into anonymizing cliques. Adversaries able to monitor all network traffic cannot deduce the identity of a sender or receiver beyond an anonymizing clique. In addition to strong anonymity, Herbivore simultaneously provides high efficiency and scalability, distinguishing it from other anonymous communication protocols. Performance measurements from a prototype implementation show that the system can achieve high bandwidths and low latencies when deployed over the Internet.

1 Introduction

Even though strong anonymity and privacy guarantees are critical for many applications, current Internet networking protocols provide no support for masking the identity of communication endpoints. An adversary that monitors Internet routers can determine which IP addresses have contacted which services. Tracking software installed at the ISPs can map IP addresses back to individuals. While encryption protocols, such as SSL, make it computationally difficult for attackers to decipher *what* was sent, they cannot hide *who* sent it. The situation is particularly problematic when governments and large corporations monitor online activity, track user behavior or censor communications.

In this paper, we describe a peer-to-peer, tamper-resistant, scalable anonymous communication protocol, called Herbivore. Herbivore has three critical properties: (1) it hides the identity of communication endpoints, even from attackers with unrestricted wiretapping capabilities, (2) it scales well with large numbers of users, and (3) it is efficient in terms of bandwidth and latency.

Herbivore is structured as an overlay network on top of an untrusted substrate, like the Internet. It provides strong anonymity by building on the information theoretic guarantees of dining cryptographer networks (DC-nets) [2]. Specifically, Herbivore guarantees sender and receiver anonymity; that is, it is computationally intractable for observers that monitor all network traffic, as well as participate in the protocol, to deduce the origin or destination of a message beyond an anonymizing clique. Herbivore's scalability stems from a divide-and-conquer approach that partitions the network into anonymizing cliques. Herbivore uses a decentralized, peer-to-peer approach to organize the global network securely into smaller cliques in which anonymous communication can occur efficiently. Actual measurements from a prototype deployed over the Internet demonstrate that the system can achieve high bandwidths and low latencies in practice.

Overall, the Herbivore system provides a provably anonymous communication channel that can scale to large numbers of participants and run efficiently over existing networks. Herbivore enables participants to contact legacy services on the Internet through proxy nodes, though in this mode of usage, it guarantees only the anonymity of the endpoint that is part of the Herbivore network.

The rest of this paper describes the design of Herbivore, examines its behavior analytically and presents measurements from a prototype implementation. Section 2 describes previous work on anonymous communication protocols. Section 3 provides the necessary background on DC-nets. Section 4 describes the

operation of Herbivore, including the local and global network topology algorithms. Section 5 addresses potential attacks on the network. In Section 6 we describe the structure of representative applications built on top of Herbivore. Section 7 presents results from our prototype implementation. Finally, Section 8 summarizes our contributions.

2 Related Work

Three critical properties for anonymous communication protocols are *strong anonymity*, *scalability*, and *efficiency*. By strong anonymity, we mean a system's ability to protect the participants' identities from an adversary that is both capable of unlimited wiretapping (unlimited passive attacks), and compromising a significant fraction of participating nodes (active attacks). By scalability, we mean the ability of the system to accommodate large numbers of participants and to decouple its performance from the number of total nodes. By efficiency, we are referring to the effective anonymous bandwidth of the system, that is, the number of bits sent per anonymous bit transferred, as well as its latency.

Previous work on anonymous communication protocols can achieve any two, but not all three of these critical properties. Past work can be grouped into three categories: source-rewriting systems, broadcast protocols, and DC-nets.

Source-rewriting systems Source-rewriting systems provide anonymity via packet reshuffling. Mixes [1] work by removing the timing correlation between arriving packets and outgoing packets. A mix is a forwarder node that queues incoming packets for a period of time $0 \leq t \leq T$ such that, when finally sent, the packets appear as if they may have originated from any one of the nodes that contacted that mix in the last T seconds. Mixes work best when arranged in series, and need a constant amount of traffic to retain anonymity while avoiding long packet delays. Tarzan [7] uses a decentralized, peer-to-peer approach to construct such a series of mixes, called tunnels. Neither system is immune to statistical analysis, and their latency is proportional to the degree of anonymity they provide.

In other source-rewriting protocols, including Onion Routing [19], Crowds [14], and Hordes [17], messages are sent through the network via random paths that deter packet traces. Each node that forwards the message rewrites the source field of the packet with its own id. Consequently, attackers with limited wiretapping abilities cannot easily track packets back to their originators. In order to make it difficult to track a packet's propagation through the network, the packets are encrypted and potentially reordered at each node, though the systems differ on how these operations are performed. In Onion Routing, the sender picks a full path through the network and encrypts the packets in layers. Each node along the path reorders the packets, strips off a layer of encryption and forwards them. In Crowds, the paths are determined locally and on-the-fly by the forwarders instead of *a priori* by the senders, and encryption is done via pair-wise key exchange and symmetric encryption between forwarders. In both of these systems, returned packets follow the reverse route from the sender. Hordes improves upon these two source-rewriting systems by replacing the return path with a low latency multicast operation. Source rewriting is also used in peer-to-peer content distribution networks such as Freenet [3] and the Freedom Network [8] to obfuscate the identity of request originators. While these systems are efficient and scale well, they do not provide strong anonymity guarantees. A passive adversary can perform traffic analysis to trace the packet through the intermediary nodes to a sender and receiver. Further, source-rewriting incurs high latencies, as the round-trip times grow linearly with the number of anonymizing forwarders traversed.

Broadcast protocols Broadcast protocols, such as $\mathcal{P}^5$ [16], provide sender and receiver anonymity by transmitting encrypted packets at a constant rate to all participants. When a node has no data packets to send, it sends noise, which is then propagated throughout the network in the same manner as data packets. This approach provides strong anonymity, as a passive eavesdropper cannot tell which packets contain data and which packets are noise. Alone, a constant bit rate broadcast protocol would scale poorly with increasing network sizes. $\mathcal{P}^5$ scales by partitioning the network into anonymizing broadcast groups.

With this protocol, a node can send messages at a rate of $1/S$ with low packet loss and $\log S/S$ with high packet loss, where S is the number of nodes in its anonymizing broadcast group. $\mathcal{P}^5$ achieves anonymity and scalability, but at a cost of efficiency. It can waste bandwidth and place large loads on the underlying network, since accommodating high peak data bandwidths with this approach requires that the network constantly run at the highest possible load. Overall, the constant bit rate limitation is not well-suited for bursty data traffic, and efficiency is elusive under a scheme whose anonymity depends on propagating noise.

DC-Nets Introduced in 1988 [2], DC-nets are an elegant mechanism for strongly anonymous communication. Since they provide the basis of Herbivore, we describe their operation in detail in the next section.

3 DC-nets Background

In this section, we informally describe the operation of DC-nets, and highlight the areas where Herbivore makes contributions related to their basic operation. A formal specification of DC-nets can be found in Appendix A.

DC-nets propagate a bit of information in the following way: Suppose there are two participants, Alice and Bob, one of whom (Bob) wants to communicate a one-bit message to Charlie. Alice and Bob first toss a coin in secret. Alice reports the truthful result of the coin toss to Charlie. Bob, on the other hand, reports the true result of the coin toss only if he wants to transmit a 0. If he wants to transmit a 1, Bob lies about the coin toss. Charlie deciphers the message by taking the XOR of the values sent by Alice and Bob. If they both report 0 or 1, they are both telling the truth and the one-bit message is a zero; otherwise, one of them is lying, and the one-bit message is a one. Since Charlie does not know if it was Alice or Bob who lied about the coin toss, he can never determine who sent the message. This security guarantee is strong and information-theoretic: No amount of computational power can help Charlie determine that it was Bob who sent the message.

Turning this basic idea for one bit transmission into a general scheme for communication between arbitrary numbers of hosts requires some modifications, originally outlined in [2]. First, for all participants to be able to send and receive messages, the nodes have to be arranged in a connected key graph; that is, Charlie needs to have shared secret keys with Alice and Bob. In addition to this key graph, the nodes need to communicate with each other through a physical transmission network. Previous work considered deploying DC-nets in a variety of topologies [11, 5]. Here, we prove that a star topology is optimal for the communication requirements of DC-nets, and use it as part of the Herbivore intra-clique topology.

Second, a protocol is necessary to ensure that only one node sends a message at a time. If multiple nodes try to simultaneously transmit in a DC-net, the network will in effect transmit the XOR of the messages, leading to collisions and corrupted messages. Previous work on DC-nets devoted additional bandwidth to reservations, thereby reducing the likelihood of such collisions. These reservation schemes targeted a low packet collision rate, and derived the amount of bandwidth to devote to reservations from this rate and the birthday paradox. Depending on the essentially arbitrary choices of packet size and collision rate, this reservation scheme can be quite inefficient and entail large space overheads. Here, we relate the bandwidth to be used for the reservation phase to the choice of the packet size and network load, and systematically derive an equation for the size of the reservation block.

Third, self-organization typically requires some signaling among the participants, and in an anonymous network, this signaling may have to be done anonymously. This can be inefficient if nodes simply reserve a slot and then transmit their message, as transmissions in traditional DC-nets require bandwidth proportional to the number of participants. Here, we present a scheme for probabilistic, anonymous and unanimous agreement in a DC-net that requires each node send a constant, and in practice modest, number of bits independent of clique size. Herbivore uses this signaling mechanism to control the local topology of each anonymizing clique.

The preceding modifications turn the core idea for one-bit transmission from two hosts into a general idea for sending anonymous packets between N participants. While this approach provides anonymity, it is vulnerable to a range of denial-of-service attacks and does not scale well with increasing numbers of participants. A malicious node can trivially disrupt the network by sending messages without a reservation. Worse, the anonymity guarantees provided by DC-nets make it difficult to locate such disruptive participants. Past work [20] has developed techniques for identifying and cleaving such malicious nodes out of a DC-net. It relies on setting up "traps," that is, random data sent for the express purpose of catching a malicious disrupter, and can not only detect disruption but also identify the disrupters. In a model where there is a single, global DC-net, the ability to specifically identify such malicious nodes and drop them from the network is critical, as they can render the entire network unusable. However, the ability to cleave out disrupters comes at a price: The network protocol is considerably more complex as it needs to make sure that the trap mechanism itself is not abused by malicious nodes to cleave out legitimate participants, and a substantial ($O(N)$) amount of the anonymous bandwidth is dedicated to traps. Further, the anonymous bandwidth of a DC-net is limited by the slowest participant. Overall, DC-nets elegantly provide strong anonymity, but suffer from efficiency and scalability problems.

4 Protocol

Herbivore simultaneously provides scalability, efficiency and strong anonymity. There are two components to the Herbivore system. At the lowest level, a *round protocol* governs how bits are sent among the participating nodes. This protocol achieves strong anonymity by building on DC-nets at the wire level. It extends the basic DC-net scheme in several ways to utilize the network efficiently, detect tampering and facilitate long-lived transactions. The round protocol provides efficient and anonymous communication; however, its performance is inversely proportional to the number of simultaneously active sessions, and malicious nodes can affect all participants. To scale well in the face of planetary scale networks and malicious participants, Herbivore employs a *global topology control* algorithm to divide the network into smaller anonymizing cliques. Herbivore guarantees that each clique will have at least k nodes, where k is a predetermined constant that describes the degree of anonymity offered by the system. New cliques are created automatically when existing cliques grow too large to communicate efficiently. When the number of nodes in a clique falls below k, the nodes in that clique are redistributed throughout the network. A secure entry control protocol deters attempts by malicious nodes to subvert the global topology control algorithm.

In the next two sections, we describe the operation of the system from the top down. We first treat the round protocol as a black box, and describe how Herbivore's global topology control algorithm organizes a large network into a collection of small cliques. These cliques enable the algorithm to scale and to contain malicious nodes. We then describe the round protocol used within a clique for anonymous data transmission in detail, and describe how the system achieves efficiency.

4.1 Global Topology Control

The global topology of the Herbivore network is determined by an entry control protocol, which assigns newly arriving nodes into cliques, and by a clique maintenance protocol, which rearranges existing nodes when the number of participants within a clique is too many or too few.

Entry Control Protocol A node joins the Herbivore network by contacting any one of the participants and following the entry control protocol. This protocol serves three purposes. First, it deters attacks targeted at compromising the anonymity of specific nodes. If nodes could select which clique to join, conspiring attackers could take over all but one member of a selected clique, and consequently trace all

transactions emanating from that clique to the remaining node. Second, it ensures that cliques are of approximately equal size by randomizing the point of entry for newly arriving nodes. Finally, through a challenge-based protocol, it limits the rate at which nodes can join the network, curbing the impact of malicious nodes who attempt to launch denial of service attacks by rapidly joining the network in several locations. The entry control protocol operates by assigning physical nodes uniformly distributed identifiers in a virtual space, and subsequently grouping them into cliques of size k or more.

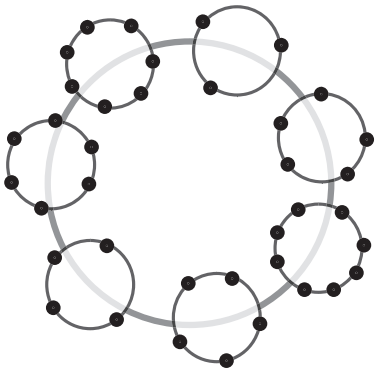

Fig. 1. The global topology of Herbivore is structured as a ring, with each clique assigned a unique key. While communication in Herbivore is primarily done at the clique level, nodes can leverage this global backbone to communicate with other cliques

A simple and intuitively appealing scheme for entry control would be to group nodes together by the hash of their IP addresses. Indeed, many peer-to-peer systems use the hash of an IP address and a small salt (between zero and eight bits) to map the physical participants onto a virtual ring [15, 18, 21, 10, 9]. Some others allow the nodes to directly pick their location [13]. All such systems are vulnerable to attack by an adversary which owns a large number of IP addresses, and can thus pick a particular IP address whose hash happens to map to a point of interest in the overlay network. The salt makes these systems even less secure, as it enables malicious nodes to pick their salt value to match the desired location, in addition to their IP address. For instance, attackers with a class-A IP address and an 8-bit salt have 2^{25+8} bits at their disposal to pick their point of entry into the network. These schemes are unsuitable for use in Herbivore, as the freedom to judiciously select the point of entry would enable an attacker to crowd out all but one legitimate node in a targeted clique, exposing the transmissions of the remaining node. We call such an attack a topology attack. Herbivore's secure random entry protocol is designed to prohibit topology attacks.

The entry protocol takes the decision on which clique to join out of the hands of the joining node. Each node and each clique in Herbivore have a unique *node key* and *clique key*, respectively. Let f and g be one-way functions. To enter the network, a node first generates a public/private key pair $K_{public}/K_{private}$. It then randomly generates vectors until it finds a $y \neq K_{public}$ such that the lower m_k bits of $f(K_{public})$ equal those of $f(y)$. The value $g(K_{public}, y)$ forms the node key for the entering node, which then joins the clique whose key is closest numerically to its node key. Any available method can be used for locating this clique; however, distributed hash tables are a natural choice as they provide a decentralized, peer-to-peer way of locating close objects given a key. In Herbivore, we use the off-the-shelf Chord protocol [18, 9]. Having located the closest clique, the node first presents the pair (K_{public}, y). Each node computes $f(K_{public})$ and $f(y)$, confirms that that the lower m_k bits match, and checks that the key $g(K_{public}, y)$ corresponds to its clique. If (K_{public}, y) has been previously presented by another member of the clique, then the node is denied admission – this prevents a malicious node from 'inviting' colluders into the clique. Finally, the clique collectively generates a random vector v_{chal}, and sends it to the entering node who should then return $v_{resp} = K_{private}(v_{chal})$. After the clique confirms that $K_{public}(v_{resp}) = v_{chal}$, the node is granted admission into the clique.

The computation involved for a node to legitimately enter the network is essentially determined by m_k, as generating the public/private key pair and performing the encryptions is relatively cheap. Let X be the number of attempts before finding y with the lower m_k bits of $f(K_{public})$ equal to those of $f(y)$. Then $EX = 2^{m_k}$. Furthermore, $P(X > lEX) = \left(1 - \frac{1}{2^{m_k}}\right)^{lEX} < e^{-l}$.

A malicious node may attempt to sidestep random entry in several ways. It is impractical to generate a specific key $g(K_{public}, y)$, since this involves computing the inverse of a one-way function. However, a node may attempt to gain admission into a particular clique by repeatedly generating keys until it finds one

which is sufficiently close to the clique's key K_C. We can deter this method of attack by picking a large m_k. This safeguard has little impact on legitimate nodes since they only need to generate a key once, and can use the same key whenever entering the network. The public/private key pair deters malicious nodes from reusing the keys presented by other nodes. For added security, we time stamp the keys: Instead of choosing random y, we require nodes to choose y with the lower bits encoded with the date. While this forces honest nodes to generate new keys periodically, it makes it significantly more difficult for malicious nodes to generate a large number of keys and then attack the network using this dictionary.

Overall, our secure random entry mechanism ensures that it is impractical for coordinated attackers to take over a clique. For instance, an attacker that has compromised 90% of the total nodes participating in Herbivore has only a $0.9^{127} \approx 1.5 \times 10^{-6}$ chance of taking over a clique of size 128.

Cliques can grow and shrink over time as nodes join and leave the network. To ensure anonymity, Herbivore guarantees that each clique has at least k participants at all times. Since each participant is equally culpable for all packets in a DC-net, even modest numbers for k suffice to provide anonymity. Large cliques, however, can degrade performance. Herbivore maintains clique sizes between k and approximately $3k$ by automatically fusing existing cliques and splitting them when necessary.

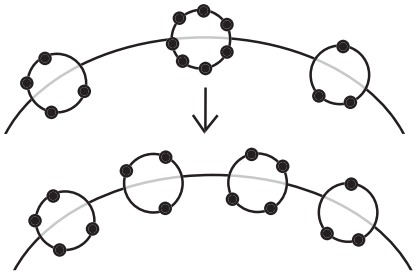

Fig. 2. When a clique's size hampers efficient communication, it splits into two new cliques that are equally spaced in the gap left behind by the original clique.

New cliques are created when there are $3k$ or more nodes in a clique and all nodes in the clique agree to disband. (Section 4.2 discusses how the nodes come to this consensus.) Cliques in Herbivore are arranged in lexicographic order by their clique keys. The Chord protocol is used to maintain pointers to the cliques immediately preceding (C_{prev}) and succeeding (C_{succ}) a given clique C. During clique formation, the old clique key is eliminated, and two new cliques, C_1 and C_2, are formed with keys $K_{C_1} = K_{C_{prev}} + \frac{1}{3}\left(K_{C_{succ}} - K_{C_{prev}}\right)$ and $K_{C_2} = K_{C_{prev}} + \frac{2}{3}\left(K_{C_{succ}} - K_{C_{prev}}\right)$. To finalize the clique formation, C_{prev} and C_{succ} must link to the new cliques. To ensure that malicious nodes cannot insert a new clique into the network, nodes in C_{succ} and C_{prev} examine the node keys for the newly created clique before linking to them. Lastly, the nodes in the cleaved clique re-enter the network using their original keys. Since the entry protocol provides a probabilistic guarantee that node keys will be uniformly spaced, we can expect half of the nodes to go to C_1 and half to go to C_2.

Cliques are disbanded when the number of participants drops below k. The clique is destroyed entirely, and the nodes independently re-enter the network. While there is a tradeoff between anonymity and bandwidth, the choice of k, which in our current implementation is set to 64, is essentially arbitrary. In line with the philosophy of $\mathcal{P}^5$, we could have several independent Herbivore networks, each enforcing different minimum clique sizes. This would allow nodes to select their own anonymity to bandwidth ratio.

4.2 Round Protocol

The round protocol governs the behavior of nodes within a given clique. It ensures that nodes can transmit data anonymously, reserve bandwidth and detect tampering. In each round, fixed amounts of data, corresponding to packets, are anonymously transferred in consecutively numbered *slots*. As described before, bit-level transmissions in DC-nets require a random number stream. Herbivore ensures during clique entry that each pair of nodes in a clique has secretly exchanged [4] a vector that is then used to seed a random number generator. A round then proceeds in three phases:

Reservation Phase This phase assigns transmission slots to nodes in order to reduce collisions and improve bandwidth utilization. Each node that would like to transmit during this round uniformly at random picks a number i from $\{1, \ldots, m_r\}$, and then anonymously transmits the m_r-bit vector with a 1 in the i^{th} bit and zeros everywhere else. All others anonymously transmit the 0 vector. The vector broadcast by the clique indicates the order of sending in the next phase. Since multiple nodes broadcast the reservation block simultaneously, it is possible for their transmissions to collide. If an even number of nodes attempt to reserve a given slot, the collision will be evident in the reservation phase, and they will simply wait until the next round to transmit. If an odd number of nodes collide, the collision will occur during the transmission phase.

Transmission Phase This is the phase in which data transmission occurs. Each node that has reserved a slot anonymously transmits its packet in the appropriate slot; all other nodes anonymously transmit 0. Each node is both transmitting and receiving in this phase. Specifically, the nodes who have reserved a slot monitor the anonymous channel and monitor the packet sent anonymously. They can thus detect bona-fide collisions and tampering by malicious nodes: The packet received over the anonymous channel will simply not match the data they intended to send in that slot. This is akin to collisions in an Ethernet. A node that detects a collision waits until the next round to try to retransmit. After a fixed number of unsuccessful retries, it joins the network in a different location. The integrity of the transmitted data is protected by an MD5 checksum attached to each packet.

Exit Phase The purpose of the exit phase is to ensure that long-running network transactions are protected from traffic analysis. This phase consists of a vote to check if the current round is a suitable time for changes in clique membership. A node may use this phase to anonymously signal to other nodes that it is in the midst of a long-running transaction, and that they should delay their departure from the clique if possible. This process is quite efficient, requiring a constant number (m_v) of bits independent of clique size, but it is not binding, as nodes may leave or crash at any time.

Below, we describe and analyze the round protocol in more detail, and determine specific values for system parameters.

Local Network Topology Logically, anonymizing cliques form a fully-connected graph, where every node has a shared secret, and a corresponding random bit stream, with every other participant. The selection of seeds for pseudo-random number generators is done securely [4], pairwise among participants when a node joins a clique. This arrangement avoids having to trust any single coordinator node, while providing full anonymity. Deciphering the identity of a message originator requires collusion by $k-1$ members of the clique.

Physically, anonymizing cliques in Herbivore are arranged in a star topology. Each node transmits the XOR of its key streams and (if present) its message to the center of the star, which then transmits the deciphered message to each of the $k-1$ nodes. A star-shaped intra-clique network topology requires that $2(k-1)$ bits be sent by the network in order for a node to transmit 1 bit anonymously.

The following lemma shows that the local star topology used by Herbivore is optimal for DC-nets.

Lemma 1. *Consider a DC-net with k nodes and assume that a bit transmitted by one node is received by exactly one other node. Let b_i be the number of bits transmitted by node i while facilitating the clique to anonymously broadcast a 1-bit message m. Then, $\sum_{i=1}^{k} b_i \geq 2(k-1)$. That is, the network sends at least $2(k-1)$ bits to propagate 1 bit anonymously.*

Proof: Label the bits sent $\{m_j\}_{j=1}^{N}$, and let $\{t_j\}_{j=1}^{N}$ be the times at which the corresponding bit is received by a node. Without loss of generality, we can assume $t_j \neq t_k$ for $j \neq k$. Then there is a time t'_1 such that

exactly one node is able to deduce the message m. Call this node k_1. But in order for k_1 to have deduced m, each of the other $k-1$ nodes must have already transmitted the XOR of their keys and their individual message (although not necessarily to k_1). So by time t'_1, at least $k-1$ bits have been transmitted. Finally, since at time t'_1 none of the other $k-1$ nodes are able to deduce the message, each of them requires at least one additional bit of information. Because a bit transmitted by one node can only be received by one other node, we have the result. □

Since the center node incurs a disproportionate network load, Herbivore selects a new center for each round by cycling through the clique members. As a center, a malicious node can corrupt messages, inject fake packets or slow down the network, but cannot compromise anonymity. We note that leaf nodes can also launch these attacks, and treat them in more detail in Section 5.

Overall, the fully-connected key graph preserves anonymity, while the physical star topology assures that the latency of transmission is independent of clique size, and bandwidth is used efficiently.

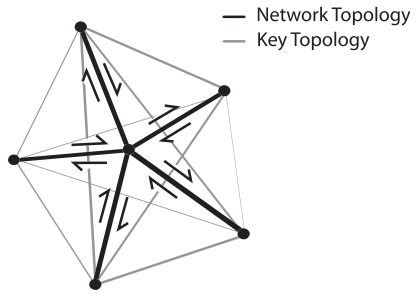

Fig. 3. A Six Node Clique. While the network topology is star-shaped to ensure high bandwidth utilization and low latency, the key topology is a complete graph to protect anonymity. Since the center of the star has a disproportionate network load, Herbivore selects a new center for each round by cycling through the clique members.

Reservation Phase In this section, we discuss how we determine the size of the reservation vector used in the first phase of the round protocol and show that Herbivore is bandwidth efficient. Previous work [2, 20] also posited a reservation phase but differed from Herbivore in two important respects. First, to detect disrupters, it advocated that all participants reserve and use at least one slot in each round regardless of whether or not they had data to send. Second, it used a loose analysis to determine the ideal size for the reservation vector. In contrast, Herbivore requires nodes to reserve an anonymous transmission slot only when they have data to transmit, and proceeds systematically to determine the size of the reservation vector, increasing bandwidth efficiency.

Here, we compute the size of the reservation vector by first deriving a bound on the expected number of bits sent by a node in the Herbivore network before it is able to anonymously transmit a packet. We then minimize this expression as a function of the size of the reservation vector in order to efficiently use bandwidth.

Assume packets of fixed size p are transmitted in each round. Observe that packets collide only when two or more nodes transmit simultaneously. Assume node n_t is trying to send a packet. Let Y be the number of bits n_t must send until the packet is sent successfully. We are interested in bounding EY. Let X_i be the number of bits sent by n_t in round i and let A_i be the event that the packet was not sent successfully in rounds $\{1, \ldots, i-1\}$. Also, let N_i be the number of nodes other than n_t that try to transmit in round i, and let C_i be the event that n_t is the center of the clique's star topology for round i. We make the reasonable assumptions that for each i, we have independent A_i, N_i and C_i, and $\{N_i\}$ are iid. Finally, let m_r be the number of bits used in the reservation phase and let c be the additional overhead that is incurred in each round. Note that the number of nodes that actually transmit in a given round may be less than the number of nodes that try to reserve a slot (due to collisions in the reservation phase). So,

$$\begin{aligned}
EY &= E\left[\sum_{i=1}^{\infty} X_i 1_{A_i}\right] \\
&\leq E\left[\sum_{i=1}^{\infty} \left[(k-2)1_{Z_i}+1\right]\left[(N_i+1)p+m_r+c\right]1_{A_i}\right] \\
&\leq 2\left[(EN_i+1)p+m_r+c\right]\sum_{i=1}^{\infty} P(A_i) \\
&\leq 2\left[(EN_i+1)p+m_r+c\right]\sum_{i=1}^{\infty} P(A)^{i-1} \\
&= 2\frac{(EN_i+1)p+m_r+c}{1-P(A)}
\end{aligned}$$

where A is the event the packet is not sent successfully in a given round. Now,

$$1-P(A) = \sum_k \left(\frac{m_r-1}{m_r}\right)^x P(N_i=x) = Ef(N_i)$$

where $f(x) = \left(\frac{m_r-1}{m_r}\right)^x$. Since f is a convex function, by Jensen's Inequality we have

$$1-P(A) \geq f(EN_i) = \left(\frac{m_r-1}{m_r}\right)^{EN_i}.$$

So,

$$EY \leq 2\frac{(EN_i+1)p+m_r+c}{\left(\frac{m_r-1}{m_r}\right)^{EN_i}} \approx \left[(EN_i+1)p+m_r+c\right]e^{EN_i/m_r}.$$

Considering the equation above as a function of m_r and taking derivatives, we see that it is minimized for $m_r \approx EN_i\sqrt{p}$. As the amount of data sent in each round increases, it becomes more important to avoid collisions. With this choice of m_r we have

$$EY \leq 2\frac{(EN_i+1)p+EN_i\sqrt{p}+c}{\left(1-\frac{1}{EN_i\sqrt{p}}\right)^{EN_i}}.$$

Furthermore, $\frac{p}{EY}$, the ratio of anonymous bits sent by the node to total bits sent by that node satisfies

$$\begin{aligned}
\frac{p}{EY} &\geq \frac{p\left(1-\frac{1}{EN_i\sqrt{p}}\right)^{EN_i}}{2(EN_i+1)p+EN_i\sqrt{p}+c} \\
&\overset{p\to\infty}{\longrightarrow} \frac{1}{2(EN_i+1)}.
\end{aligned}$$

The above analysis suggested that the reservation vector should be of size $m_r = k_r\sqrt{p}$, where p is the packet size, and k_r is the expected number of nodes that will transmit in a given round. In practice, we

can approximate k_r by a running exponentially-weighted average of the number of nodes that transmit in a given round. With this choice of m_r, we showed that a node will, on average, transmit no more than approximately $2k_rp$ bits before successfully sending a packet anonymously. Note that this bound scales according to the expected number of nodes that will actually send data anonymously, not the total number of nodes in the clique. Consequently, nodes can achieve high anonymous bandwidth on a lightly loaded Herbivore network, and still maintain high anonymity as determined by clique size.

Exit Phase Often, nodes would like to anonymously transmit multiple packets. For instance, a node may want to send a message whose length exceeds the fixed slot size in Herbivore, or a user may engage in a long-running transaction with a web service. If the time frame for these transactions significantly exceeds average node lifetimes, an attacker may be able to analyze network traffic to compromise a sender's identity. Recall that Herbivore provides anonymity among the members who were in the clique at the time a message was sent. If an identifiable stream of packets is sent from a clique at times $t_0, t_1, ..., t_n$, the possible originator must be one of the nodes that was in the clique at all of those times. Consequently, an *intersection attack* might be feasible against long transactions and cliques that rapidly replace member nodes.

Herbivore has two protection measures against intersection attacks. The first is for each node to monitor its long-running transactions and determine whether it could be targeted for such an attack. That is, each node continually launches this attack against itself, and terminates its long-running transaction whenever its anonymity drops below a critical threshold. In our implementation, we chose this threshold to be the minimum clique size k. Consequently, each Herbivore node hides its identity among at least $k = 64$ nodes. While effective, this approach abruptly terminates communication when this threshold is reached. To ensure that this is rare, Herbivore provides a third phase to the round protocol that limits changes to clique membership during long-running transactions.

This new phase consists of an anonymous poll to discover whether any node would be adversely impacted by a node departure. Nodes in the middle of sending an anonymous stream, uniformly at random generate and anonymously transmit an m_v-bit vector $\{v_i\}$. All others anonymously transmit the 0 vector. To leave the clique, nodes wait until the message broadcasted during this phase is identically 0. For example, if exactly two nodes attempt to veto a node's departure and they randomly generate the same vector, then the broadcasted vector, which is the XOR of the two vectors, will be zero. The following lemma shows that with high probability, nodes will successfully signal a veto.

Lemma 2. *Assume we have k nodes in a clique, and $k_v \geq 0$ nodes attempt to veto. Let $\{X_i\}_1^{m_v}$ be the m_v-bit message broadcasted by the DC-net. That is $\{X_i\}$ is the XOR of the randomly generated veto vectors. If $k_v = 0$, $P_0(X_i = 0) = 1$, and for $k_v > 0$,*

$$P_{k_v}(\{X_i\} = \{v_i\}) = \frac{1}{2^{m_v}}$$

where $\{v_i\}$ is any m_v-bit vector. In particular,

$$P_{k_v}(\{X_i\} \neq 0) = 1 - \frac{1}{2^{m_v}}.$$

Proof: The first statement is clear. For $k_v > 0$, an easy combinatorial argument yields that half of the subsets of $\{1 \dots k_v\}$ have even order. Furthermore, since the 0 bit will be broadcast by the DC-net iff an even number of nodes transmit a 0, $P_{k_v}(X_i = 0) = P_{k_v}(X_i = 1) = \frac{1}{2}$. The result follows immediately. □

Note that for $k_v > 0$ the distribution of $\{X_i\}$ is independent of k_v. Consequently, the above protocol only indicates whether or not at least one veto was issued, but does not reveal specific information regarding the number of nodes that veto.

Node Failures While the exit phase delays the departure of cooperating nodes, malicious nodes may leave despite a veto. Further, even non-malicious nodes may crash at any time. In this section, we examine the resilience of Herbivore to random node failures. We assume a fail-stop failure model in this section, and treat Byzantine failures as part of the active attacks in section 5.

Assume we have k nodes in a clique and let $X_i(t)$ denote the number of times node i has failed up to time t. We model $X_i(t)$ as a parameter 1 Poisson process. Then, $P(X_i(t) = 0) = e^{-t}$. Let $X(t)$ be the number of nodes that have failed at least once up to time t. Then,

$$P(X(t) \leq m) = \sum_{i=0}^{m} \binom{n}{i} \left(1 - e^{-t}\right)^i \left(e^{-t}\right)^{n-i}.$$

Since the exit phase ensures that well-behaved clients will remain connected during long-running anonymous transactions, we are primarily concerned with nodes either crashing or users abruptly disconnecting from the network. Consequently, we can expect relatively long mean lifetimes for nodes in the network. Figure 4 shows how the number of node failures scales for various numbers of nodes and times. For example, if the mean lifetime for a node is 10 hours, a member of a 128 node clique can transmit an anonymous stream for 1 hour and be very confident that no more than 25 nodes will fail during that time, leaving its identity concealed to 1 of 103 nodes. We find this to be a conservative, and acceptable, level of anonymity.

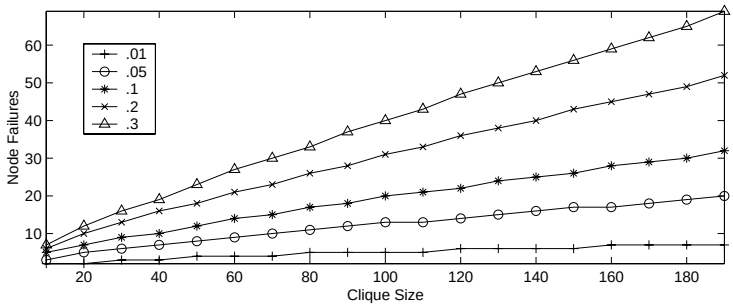

Fig. 4. Expected node failures as a function of clique size and failure rate. The Y-axis displays $\min_m \left(P\{\text{at most } m \text{ node failures by time } t\}\right) > .999)$, where t is the time on the network relative to the mean time between failures for a given node.

5 Attacks

Herbivore is resilient to a wide array of attacks aimed at both compromising a node's anonymity and degrading the network's performance. Here we describe possible attacks against the Herbivore network and detail their impact.

Collusion and Occupancy Attacks If malicious nodes gather in a clique, they may share information to compromise the anonymity of an honest node. Since we employ a complete key topology, colluding nodes can only determine that they did not send a given message. Consequently, a node's anonymity is determined by the number of honest nodes in a clique. Herbivore's random entry protocol, which uniformly distributes malicious nodes across all cliques, deters attackers from targeting a specific clique. For instance, in a 128-node clique, an attacker that has compromised 90% of the total nodes participating in Herbivore has only a $0.9^{127} \approx 1.5 \times 10^{-6}$ chance of taking over a given clique.

Sybil Attacks A small number of malicious nodes may attempt to overwhelm the system by joining a large number of cliques under many false identities. While such Sybil attacks are theoretically impossible to prohibit without external measures [6], Herbivore addresses the problem by rate limiting node entry into the network. To enter the Herbivore network, nodes complete a computational challenge, which we have shown is resilient to subversion.

Topology Attacks We rely on a variant of the Chord distributed hash function [18, 9] to provide the mapping from clique keys to clique members. We treat the security of Chord as a separable problem and do not analyze its resilience against attacks, though we note that the high level of redundant routing information in Chord makes it difficult for malicious nodes to subvert the system if each lookup is repeated from different starting points on the ring. Herbivore's secure random entry protocol prohibits topology attacks in which nodes target specific cliques in the network.

Intersection Attack By identifying long-running transactions and monitoring clique membership, an attacker may try to subvert a node's anonymity with an intersection attack. As discussed, we minimize the impact of such attacks by introducing an exit phase in the round protocol. Consequently, we can expect cooperating nodes to remain in their cliques for the duration of these transactions. Furthermore, we have shown in section 4.2 that node failures do not significantly affect anonymity. For example, if the mean lifetime of a node is 10 hours, a member of 128 node clique can transmit an anonymous stream for 1 hour and be very confident that no more than 25 nodes will fail during that time, leaving its identity concealed to 1 of 103 nodes. Finally, nodes in Herbivore continually launch an intersection attack on themselves, and cease transmitting when their anonymity is sufficiently compromised.

Statistical Analysis Very long-lived transactions in Herbivore are susceptible to statistical analysis: If an attacker observes that node n_v is disproportionately often in a clique that contacts a certain network service (e.g. a specific web site), then the attacker has statistical reason to believe that n_v is in fact the node contacting that service. While every packet transmitted in Herbivore is anonymized among members of a clique, Herbivore does not protect transactions that are significantly longer than clique lifetimes. This is a fundamental limitation of any scheme that provides anonymity in a crowd; participants need to monitor crowd sizes to gauge if they are sufficient for their application.

Coordinator Attack Since we use a star topology for intraclique transmissions, the central node represents a point of vulnerability. If the central node is compromised, it can tamper with transmissions in that round by dropping, inserting or modifying packets. Herbivore relies on higher-level protocols, such as SSL and TCP/IP, to prohibit or to recover from such attacks. Overall, the impact of a malicious center node is limited to one out of every k rounds, when it acts as the center.

Exit Attack A malicious node could attempt to nullify the votes cast during the exit phase to incite participating nodes to leave the clique during a long-lived transaction. Since center nodes can easily nullify a veto, we employ multiple coordinators in the exit phase to reduce the likelihood of such an attack. If a node could observe all broadcast vectors, that is, the XOR of all key streams and the message, transmitted by all other $k - 1$ nodes, it could arbitrarily influence the outcome of the round by judiciously choosing its own broadcast vector. Herbivore's star topology prohibits this attack, as the center node does not transmit its broadcast vector over the network.

Denial of Service Malicious nodes can decrease bandwidth by reserving too many slots, transmitting without a reservation, or slowing down the round frequency. Similar problems are frequently encountered in any domain where there is a common shared channel, like the Ethernet. The final line of defense against such attacks is to simply join a different clique. The randomized entry algorithm ensures that an attacker cannot feasibly follow a specific node through clique changes. We further note that denial of service attacks require substantial bandwidth to launch. For example, if we have 10^6 cliques transmitting at 100 Kb/s, an attacker transmitting 10 Gb/s will only shut down 10% of the network. Since round frequency is determined by the slowest node in a clique, a malicious node can decrease bandwidth by slowing round frequency. However, we can eliminate nodes from a clique that transmit too slowly. Since a malicious node at the center of the clique topology can make it appear as if other nodes are transmitting too slowly, we require other members of the clique to confirm a node's transmission speed before eliminating it from the clique.

6 Applications

Herbivore provides a general purpose anonymous communication channel that can accomodate many diverse applications. Herbivore functions as a virtual network-layer protocol and carries encapsulated IP traffic at the anonymous layer. This approach has the benefit of supporting many existing applications with few or no modifications to current networking protocols or application binaries. In this section, we examine some common application scenarios in order of increasing anonymity requirements and discuss how Herbivore can be used to meet their needs.

Herbivore enables clients to contact traditional network services, such as existing web sites, anonymously. Since these services are not expected to participate directly in the Herbivore network, we rely on proxies to ferry packets between the anonymous and the public network. Each Herbivore node wishing to communicate with such a service uniformly at random selects one of the nodes in its clique, other than itself, and anonymously asks for it to serve as a proxy. Selecting a proxy reduces the anonymity of the sender by just one node, which the affected node treats analogously to a single node failure in the clique. Since Herbivore provides anonymity, not privacy, applications need to ensure that the data they send do not betray their user's identities. For instance, anonymous web browsers should not send cookies containing usernames over the anonymous channel. An end-to-end encryption protocol will typically be necessary to ensure that other nodes in the clique do not eavesdrop on the contents of such communication. In this mode of deployment, Herbivore provides anonymity for clients, but the identities of the web services that reside outside Herbivore are exposed.

When both communication endpoints are part of the Herbivore network, Herbivore can hide the identities of both communication endpoints from third parties. A good example of such an application is messaging and two-way teleconferencing. These applications require a scheme to locate the receiver within the greater Herbivore network, and to establish an anonymous channel between the sender and receiver. The former task of locating users can be performed by flooding the network with a connection request destined for the receiver (e.g. the vector $(x, K^{n_s}_{public}, K^{n_s}_C), MD5(x, K^{n_s}_{public}, K^{n_s}_C)$ encrypted with the recipient's public key, where $K^{n_s}_C$ is the sender n_s's clique key, $K^{n_s}_{public}$ is its public key, and x is a nonce; we assume that keys are distributed through external, secure channels). Since Chord lends Herbivore a connected network topology, several methods are available for efficient broadcast, ranging from cycling through the ring to flooding to all known entries in the routing table. The receiving node, n_r, responds by sending the message $(x, K^{n_r}_C)$, encrypted with $K^{n_s}_{public}$, directly to clique $K^{n_s}_C$. All further communication occurs directly between n_s and n_r's cliques. The random vector x deters replay attacks and provides each message stream an ID so that nodes can maintain multiple messaging channels.

Herbivore also supports strong anonymity guarantees for demanding applications such as anonymous publishing. The central challenge in anonymous, censorship-resistant publishing is to hide the origin of information from all other parties, including parties that may store or actively query for a particular file. We use an approach inspired by Freenet [3] to hide the identity of a file originator, while enabling anonymous queries, caching and retrievals of stored information. We assume that files are published under descriptive names, such as "Marx & Engels/The Communist Manifesto," and that participants query using these names. A naive implementation would simply port Freenet to Herbivore, and have publishers answer queries if they happen to store the file. This implementation is subject to an intersection attack: initially, there is a single copy of the file, and if the sole publisher ever answers a query from within more than one clique, his identity can be compromised. We address this problem by anonymously replicating the document to all nodes within the original clique in response to a query. Since anyone in the clique could have distributed the document, and since everyone in the clique has a copy, the identity of the original publisher is concealed even if she were to move across multiple cliques and continue to distribute the same file. The replication of the file across the clique can be performed efficiently, as the data packets are broadcast to all clique members during the transfer to the original requester. This approach offers stronger

anonymity guarantees than Freenet; namely, the identity of the original publisher is protected from all parties, including passive adversaries as well as requesters and clique members.

7 Performance Evaluation

In this section, we present results from a prototype Herbivore implementation to show that the system is efficient and practical. To our knowledge, this is the first implementation of a provably anonymous communication system.

Our measurements were performed over the Internet using the PlanetLab infrastructure [12]. PlanetLab is a collection of approximately 100 nodes scattered across 50 academic and industrial research organizations around the globe. Each PlanetLab node has a 2GHz Pentium IV with 128KB cache and 2GB of RAM, and is connected to other PlanetLab nodes via the public Internet. Consequently, the measurements shown below directly measure actual performance in a shared wide-area network. Due to the restrictions on the size of PlanetLab, we restrict our clique sizes to 10 to 40.

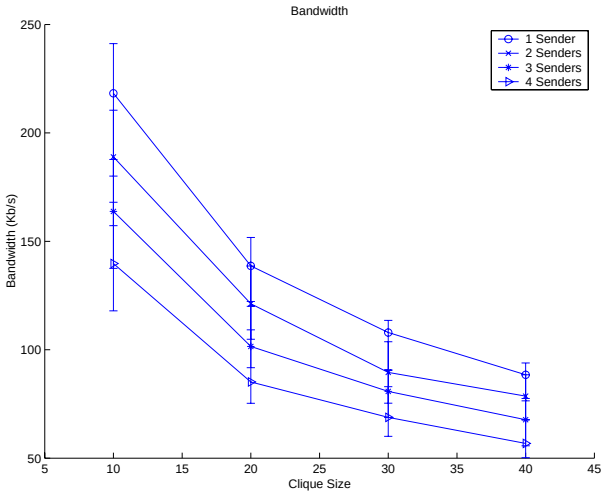

Fig. 5. Anonymous communication bandwidth in Herbivore between geographically separated nodes in the Internet. Error bars indicate interquartile ranges.

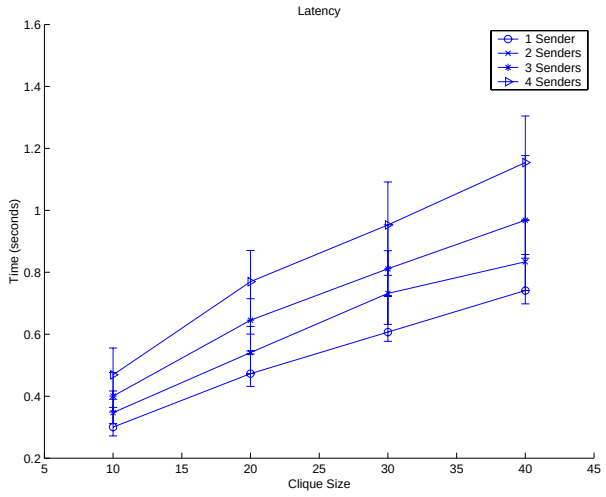

Fig. 6. Anonymous communication latency in Herbivore between geographically separated nodes in the Internet. Error bars indicate interquartile ranges.

Herbivore achieves high anonymous bandwidth and low messaging latency. Figures 5 and 6, respectively, plot the median bandwidth and latency achieved by Herbivore over the Internet over 45 runs in November 2002. Due to transient loads and congestion in the Internet, the plots exhibit some variability, though the interquartile ranges show that the performance of the system is robust across load variations during different times of the day. Overall, the bandwidth is high enough for web browsing, audio stream delivery and highly compressed video applications, while the latency is low enough for messaging applications, when the number of simultaneously active senders is small.

Note that the bandwidth and latency reported in Figures 5 and 6 will not degrade with increasing numbers of hosts in the overall Herbivore network. These graphs plot performance as a function of clique size. Since anonymous bandwidth and latency are sensitive to the slowest node within a clique, we see some reduction in bandwidth and an increase in round latency as clique sizes grow. However, Herbivore's

global topology control algorithm ensures that the performance will remain high by splitting the overall collection of participants into smaller cliques. Consequently, the right hand side of these figures represents the worst-case behavior one would expect from a Herbivore network of any size, assuming a well-provisioned Internet backbone.

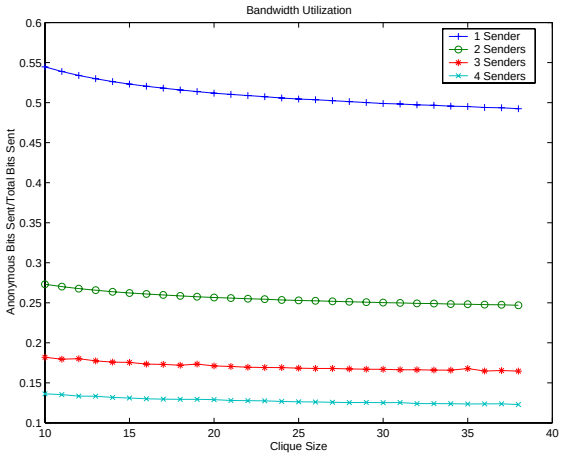

Fig. 7. Effective bandwidth utilization ratio for varying numbers of simultaneously active senders.

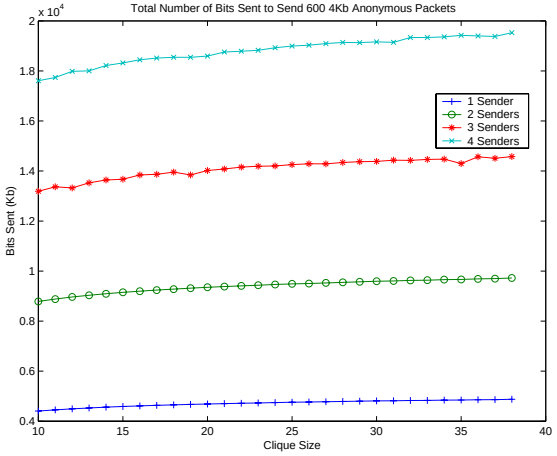

Fig. 8. Number of bits sent to transmit a constant amount of data.

The following graphs provide insights into Herbivore's performance. Figure 7 plots the effective bandwidth utilization within a single clique as a function of simultaneously active senders. By effective bandwidth utilization, we mean the ratio of anonymous bits transmitted to the number of total bits sent over the wire. The graph shows that bandwidth utilization in Herbivore is primarily affected by the numbers of simultaneous senders in a clique, while anonymity is determined by clique size. Consequently, we can achieve both a high degree of anonymity and high bandwidth on a lightly loaded Herbivore network. Figure 8 shows the total number of bits a client has to send to transmit a fixed size stream. Collisions are rare, and the measurement scales linearly with the number of simultaneously active participants in the network, independent of clique size. Performance improves slightly at small clique sizes due to the star topology used for transmission.

8 Conclusion

Herbivore is a peer-to-peer system that simultaneously provides strong anonymity, scalability and efficiency. It makes several key contributions to provide all three of these properties at the same time. First, Herbivore builds on the strong anonymity provided by dining cryptographer networks, and extends the basic protocol by incorporating an efficient reservation phase and voting protocol. Combined, these novel modifications to DC-nets increase effective anonymous bandwidth and facilitate long-lived anonymous transactions. Second, Herbivore scales by partitioning the overall network into smaller anonymizing cliques. This decouples the anonymization protocol from the size of the overall network, while it provides sufficient participants in an anonymizing clique to provide effective anonymity. Herbivore partitions the network into smaller cliques using decentralized entry control and clique maintenance algorithms, layered on top of a distributed hash table implementation. A secure random entry mechanism for a peer-to-peer network limits the ability of

malicious nodes to launch concerted attacks. Third, Herbivore is bandwidth and latency efficient. Measurements on the public Internet across nodes distributed around the globe demonstrate that the system achieves sufficiently high bandwidth and low latency for a wide variety of applications. Herbivore provides an optimality proof on the star topology used within a clique, presents a constant-space protocol for anonymous signaling within a DC-net, and analytically derives the optimal amount of bandwidth to devote to reservations. Finally, the simplicity of the protocol makes the system analytically tractable, facilitates a practical implementation and enables us to derive guaranteed bounds for its efficiency. Combined, these properties make Herbivore a practical system that can be layered on top of the Internet to provide strong anonymity.

Many real world applications rely critically on anonymity: whistle-blowers need secure anonymous channels, patients need access to medical information without fear of reprisal by insurance companies, and political dissenters need techniques for voicing their opinions without censorship. Current networking protocols used on the Internet expose the identity of communicating endpoints. We hope that a practical, scalable and efficient system that provides strong anonymity will serve as the building-block for supporting such applications online.

References

1. D. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
2. D. Chaum. The Dining Cryptographers Problem. *Journal of Cryptology*, 1(1):65–75, 1988.
3. I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. *Lecture Notes in Computer Science*, 2009, 2001.
4. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, Nov 1976.
5. S. Dolev and R. Ostrovsky. Xor-Trees for Efficient Anonymous Multicast and Reception. *ACM Transactions on Information and System Security*, 3(2):63–84, May 2000.
6. J. Douceur. The Sybil Attack. In *Proceedings of the First Workshop on Peer-to-Peer Systems*, Cambridge, MA, 2002.
7. M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, D.C., November 2002.
8. I. Goldberg and A. Shostack. Freedom Network 1.0 Architecture, November 1999. http://www.freedom.net/.
9. S. Hazel and B. Wiley. Achord: A Variant of the Chord Lookup Service for Use in Censorship Resistant Peer-to-Peer Publishing Systems. In *Proceedings of the First Workshop on Peer-to-Peer Systems*, Cambridge, MA, 2002.
10. P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Proceedings of the First Workshop on Peer-to-Peer Systems*, Cambridge, MA, March 2002.
11. A. Pfitzmann and M. Waidner. Networks Without User Observability - Design Options. *Computers and Security*, 6(2):158–166, 1987.
12. PlanetLab. An Open Testbed for Developing, Deploying and Accessing Planetary-Scale Services, September 2002. http://www.planet-lab.org.
13. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content Addressable Network. Technical Report TR-00-010, University of California, Berkeley, Berkeley, CA, 2000.
14. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
15. A. Rowstron and P. Druschel. Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-peer Systems. In *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, Heidelberg, Germany, Nov 2001.
16. R. Sherwood, B. Bhattacharjee, and A. Srinivasan. P5: A Protocol for Scalable Anonymous Communication. In *Oakland Security Conference*, 2001.

17. C. Shields and B. N. Levine. A Protocol for Anonymous Communication over the Internet. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 33–42, 2000.
18. I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Peer-to-Peer Lookup Service for Internet Applications. In *Proceedings of the ACM SIGCOMM Conference*, San Diego, CA, Sep 2001.
19. P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous Connections and Onion Routing. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 44–54, Oakland, California, 4–7 1997.
20. M. Waidner and B. Pfitzmann. The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability. *Advances in Cryptology EUROCRYPT 89, LNCS-434*, 1990.
21. B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An Infrastructure for Wide-area Fault-tolerant Location and Routing. In *Proceedings of the ACM SIGCOMM*, pages 161–170, August 2001.

A DC-Nets

We now give a more formal description of DC-nets. Let $G = (V, E)$ be a connected graph with vertex set $V = \{v_1, \dots v_k\}$ that represents the communicating nodes, and edge set $E = \{e_1, \dots e_m\}$. By $e_{ij} \in E$ we denote the edge between vertices v_i and v_j. For each edge e_{ij} assume nodes v_i and v_j secretly pick a key $k_{e_{ij}} \in \mathbb{Z}_2$, i.e. only v_i and v_j know $k_{e_{ij}}$. Further suppose each node v_i selects a message $m_i \in \mathbb{Z}_2$ and then broadcasts $b_i \in Z_2$ where

$$b_i \equiv m_i + \sum_{\{j | e_{ij} \in E\}} k_{e_{ij}} \pmod 2.$$

Then we have $\sum_i b_i \equiv \sum_i m_i \pmod 2$, since

$$\begin{aligned} \sum_{i=1}^{n} \sum_{\{j | e_{ij} \in E\}} k_{e_{ij}} &= 2 \sum_{e \in E} k_e \\ &\equiv 0 \pmod 2. \end{aligned}$$

In particular, if $m_1, \dots, m_{k-1} = 0$ then $\sum_i b_i \equiv m_k \pmod 2$. Now, call $A \subset V$ an anonymity set of $S \subset E$ if A is a connected component of $(V, E \setminus S)$. Stated informally, by knowing only the values of $\{b_i\}$ and $\{k_e \mid e \in S\}$, we can deduce no more than $\sum_{\{i | v_i \in A\}} m_i$. More precisely we have the following result [2]:

Theorem 1. *Using the notation above, let $K = (k_{e_1}, \dots, k_{e_m})$ be a random variable with uniform distribution over $\mathbb{Z}_2^m$, and let $B(m, K)$ be the broadcasts from the nodes when they transmit messages $m = (m_1, \dots m_k)$. Then for $(b_1, \dots, b_k)$ and $(m_1, \dots, m_k)$ with $\sum_i m_i \equiv \sum_i b_i \pmod 2$, we have*

$$P\left(B(m, K) = \{b_i\}_1^k\right) = \frac{1}{2^{k-1}}.$$

In particular, the above probability is independent of the message m.

Consequently, if G is the complete graph on n vertices, then vertices $\{v_1, \dots, v_j\}$ that reveal to each other their secret keys can only determine $\sum_{i=j+1}^{n} m_i$, not the specific messages $\{m_i\}$.