Eluding Carnivores: File Sharing with Strong Anonymity

Emin Gün Sirer, Sharad Goel, Mark Robson, Doğan Engin
Dept. of Computer Science, Cornell University

August 1, 2004

Abstract

Anonymity is increasingly important for networked applications amidst concerns over censorship and privacy. This paper outlines the design of HerbivoreFS, a scalable and efficient file sharing system that provides strong anonymity. HerbivoreFS provides computational guarantees that even adversaries able to monitor all network traffic cannot deduce the identity of a sender or receiver beyond an anonymizing clique of k peers. HerbivoreFS achieves scalability by partitioning the global network into smaller anonymizing cliques. Measurements on PlanetLab indicate that the system achieves high anonymous bandwidth when deployed on 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. As a result, entities with access to network links, such as governments and ISPs, can monitor online activity, track user behavior or censor communications.

In this paper, we outline HerbivoreFS, a scalable and efficient peer-to-peer filesharing system that provides strong anonymity properties. HerbivoreFS is designed to hide the identity of communication endpoints from adversaries with unlimited wiretapping powers, scale well with large numbers of users, and operate efficiently on the Internet.

HerbivoreFS is a file transfer system, structured as a peer-to-peer overlay network built on a lower-layer protocol for anonymous communication. The lower layer Herbivore protocol derives its strong anonymity guarantees from dining cryptographer networks (DC-nets) [3]. It guarantees that an adversary with unrestricted wiretapping capabilities cannot deduce the provider or requester of a file beyond an anonymizing clique without breaking RSA or reversing a one-way hash function. Herbivore scales to large networks through a divide-and-conquer approach that partitions the network into anonymizing cliques. It organizes the global network securely into smaller groups in which anonymous communication can occur efficiently. The HerbivoreFS application performs anonymous file lookup and transfer, spreading culpability across a large number of nodes to make it intractable to mount blind legal attacks against groups of users. Measurements from a prototype implementation on PlanetLab demonstrate that the system can achieve high bandwidths and low latencies in practice. Overall, the HerbivoreFS system provides a strong anonymity for file sharers while scaling up to large networks and running efficiently over existing networks.

2 Background

Three critical properties for anonymous communication protocols are *strong anonymity*, *scalability*, and *efficiency*. Previous work on anonymous filesharing achieves any two, but not all three of these properties.

In source-rewriting systems [2, 12, 8, 11, 5, 4, 6], messages are sent through the network via random paths to obfuscate their origin. Each node that for-

wards the message rewrites the source field of the packet with its own ID. In order to make it difficult to track a packet's propagation through the network, the packets are potentially delayed, encrypted and reordered at each node. Source rewriting systems scale well, impose a low aggregate load on the network, and are thus frequently deployed [4, 6, 7]. However, source rewriting systems are vulnerable to statistical correlation attacks, where a passive adversary with a sufficient packet trace can observe correlations to link file transfers back to their origin. In general, source rewriting provides better anonymity when performed with high traffic and many chained proxies, though bandwidth and latency are reduced.

Broadcast protocols [10] provide 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. Partitioning the network into smaller groups at the cost of incurring high packet loss enables such systems to scale. While broadcast protocols achieve strong anonymity and scalability, they are inefficient. Accommodating high peak data bandwidths requires that the network constantly run at the highest possible load.

DC-nets [3] are an elegant mechanism for anonymous communication that form the basis of Herbivore. DC-nets propagate a bit of information in the following way: Suppose there are three participants, Alice and Bob and Charlie, one of whom (Alice) wants to communicate a one-bit message to Charlie with the aid of a mediator. Each pair of users tosses a coin in secret; call these AB, AC and BC. Bob and Charlie report the XOR of their two coin tosses to the mediator (Bob reports $B = AB \oplus BC$, Charlie reports $C = AC \oplus BC$). Alice, on the other hand, reports the XOR of her coin tosses along with her message; that is, she reports $A = AB \oplus AC \oplus m$. The mediator can then take the XOR of all packets sent in the network, i.e. $A \oplus B \oplus C = m$ to yield Alice's message, since the doubly-reported coin tosses cancel out to reveal m. Neither a wiretapper nor a participant can tell that Alice originated the message since it is composed with the participation of all parties.

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 [3]. First, we can make standard cryptographic assumptions to eliminate the coin tosses and use a computationally secure PRNG-generated bit stream instead. Nodes simply exchange a seed value with other nodes when they join the network that they use to generate coin tosses until exhaustion. Second, we accommodate more participants by simply generalizing the triangle into a fully connected key graph. Third, we can eliminate the mediator and replace it with a broadcast network. Finally, we extend this basic transmission mechanism with a concept of transmission slots and a reservation protocol to ensure that only a single party transmits at any given time (Note that simultaneous transmissions will result in collisions, garbling the messages). The result is a shared broadcast channel, like an Ethernet carrier, except that the transmissions cannot be traced back to senders even if the entire network is tapped.

3 Herbivore

The mechanism described above provides a strong anonymity guarantee, namely, $N-1$ colluding nodes are required to identify a packet's origin. However, this approach is not practical because it scales with N. HerbivoreFS provides scalability through divide-and-conquer, and improves efficiency through modifications to the clique-level DC-net protocol.

Herbivore scales by partitioning the global network into smaller anonymizing cliques (Figure 1). Small cliques enable the protocol to operate efficiently within a small group and decouple the performance from the total number of participants in the network.

Herbivore relies on the Pastry [9] ring for its global organization. Each Herbivore node has a unique node ID that determines its virtual position in relation to other nodes. In order to avoid $N - 1$ malicious nodes from crowding into a clique around a targeted user and compromising her anonymity, Herbivore forces every node to solve a computational puzzle to compute a random ID as well as compute a public, private key pair, as in the distributed scheme

described in [1]. Note that that using a central point of entry and money in exchange for a node ID, as advocated in [1], would not impact the system except to simplify its implementation – exchanging money for a place on the ring is not fundamentally different from exchanging computational power, as it is also subject to Sybil attacks by attackers with deep pockets. Cryptopuzzles avoid the centralization, bookkeeping, and secure processing that would otherwise be required, while limiting the rate of Sybil attacks. Overall, the random entry mechanism ensures that it is impractical for coordinated attackers to take over a targeted 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 particular, targeted clique of size 128.

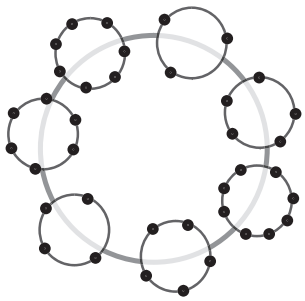

Figure 1: Global structure of Herbivore.

Once a node arrives on the Pastry ring, it finds the closest preexisting clique in the virtual identifier space and initiates a join protocol. The join protocol authenticates the client to the clique, authenticates the clique members to the client, and then picks pairwise PRNG seeds to be used for transmission. In our current implementation, authentication is performed by a simple challenge-response that ensures that a node successfully solved the cryptopuzzle corresponding to its virtual identifier, which in turn is tied to its public key. A node that has exchanged PRNG keys with all preexisting clique members notifies a randomly selected clique participant of its intent to join the anonymous communication. This request is broadcast to all clique members on the anonymous channel for atomicity, and the node is included in the next round of communication.

Cliques can grow and shrink over time as nodes join and leave the network. 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. New cliques are created when there are $3k$ or more nodes in a clique. The old clique is simply eliminated, and two new cliques are formed such that they are equally spaced in the gap left behind by the original clique. To ensure that malicious nodes cannot insert a new clique into the network, nodes in successor and predecessor cliques examine the node keys for the newly created clique before linking to them. The nodes in the cleaved clique re-enter the network using their original IDs.

Each node monitors the behavior of other nodes in its clique to detect failures as well as straggler hosts that are reducing the performance of the clique by not sending their packets in a timely fashion. Every node maintains a *strike table* that it uses to determine which nodes to eliminate from the clique. When a node expects a transmission from another participant, but does not receive it within a predetermined amount of time, it issues a strike against that node. Nodes whose strikes exceed a threshold are cleaved from the clique. The node issuing a strike incurs a fractional strike itself; hence, it is infeasible for less than $k/2$ malicious nodes to cleave good users out of the clique by issuing fake strikes.

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 is essentially arbitrary. 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.

Within a clique, the DC-net protocol outlined in Section 2 is used to communicate bits anonymously. As in traditional DC-nets, communication in Herbivore takes place in rounds, with a reservation and transmission phase. The reservation phase enables nodes to anonymously sign up for a slot in which to transmit their data. During the transmission phase, nodes transmit their data in the slots they reserved, or else simply broadcast the XOR of the keys they derived from their keys in order to assist other nodes'

transmissions.

To reduce network load, Herbivore designates a center node in each round to act as the mediator for collecting the packets from clique members and disseminating them. A flow checksum prohibits the center from injecting packets into ongoing communications. The strike mechanism prohibits centers from consistently corrupting or dropping packets. Note that the center, like an omnipotent wiretapper, cannot determine the origin of any packet.

The Herbivore implementation achieves high anonymous communication bandwidth by running many instances of the protocol concurrently. For batch transfers, many rounds can be executed concurrently, each with a different mediator. This permits Herbivore to mask the latency of the slower nodes and achieve relatively high throughputs for an anonymous communication system.

4 Filesharing with Herbivore

The Herbivore system described above provides a general-purpose anonymous communication channel on top of which we layer the HerbivoreFS application. The interface to the application is straightforward: users provide a list of files they would like to make available, as well as a list of files they would like to download. There are two components to the implementation. The *query routing* component propagates queries throughout the network and identifies cliques from which a file can be obtained. The *file transfer* component performs the transfers anonymously. Intersection attacks, described below, form the central challenge to the design of both components.

Intersection attacks can be mounted whenever a network is decomposed into anonymizing groups, and there is some way to learn the presence of a node within a group. For instance, a naive approach to filesharing would simply port Gnutella to Herbivore; that is, it would have publishers answer queries if they happen to store the requested file. This implementation is subject to an intersection attack: initially, there is a single copy of the file, and if the sole publisher answers a query from within more than one clique, his identity can be compromised. Namely, it can be narrowed down to the intersection of all cliques from which that file originated. Query routing and file transfer are designed to mitigate such attacks.

Files in our current implementation are published under user-supplied names, carry a content hash for differentiation, and can be queried textually. Herbivore is a self-sufficient system; an external mechanism for publishing filename to file ID mappings is not necessary as it is in Freenet since files can be queried textually. Each node reserves space for two sets of files: the A-list is a set of files to be introduced into the network by that node, and the B list is a set of files that this node overheard in its clique and cached locally. Files are placed on the A-list by the user and are transferred to the B list after they are first broadcast in response to a query. The B list is managed as an LRU cache on disk.

A node seeking a file initially broadcasts an unencrypted Query packet to its own clique. Query packets as well as all response packets are rebroadcast in case there are any collisions at the Herbivore link layer. A node that receives a Query for a file on its A or B lists responds by packetizing and transmitting the file over the anonymous channel. All non-malicious clique members receive the transmissions and place the file on their B list when the transfer is complete. If the file was on the file provider's A-list, it is moved to the B-list. This anonymous replication of the document defeats intersection attacks as long as the same file is introduced into the network no more than once by the same node. Since anyone in the clique could have distributed the document, and since everyone in the clique now 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. Note that this HerbivoreFS does not guarantee file availability in perpetuity - files that are not requested frequently may get dropped from the network because of the way the B-list is managed. We see this as a reasonable tradeoff against protection from intersection attacks.

Usually, however, files cannot be located within

the initial clique and the search has to be expanded to other cliques. While the fileseeker can locate other cliques and their members through the Pastry ring, she cannot simply ask them for a file as this would betray her identity. Consequently, she employs local proxies, chosen at random from her own clique members, to contact other cliques on her behalf. The proxies contact a single node on another clique (a remote proxy) with the request, which performs a local query and transfers the request back to the local proxy, which in turn broadcasts the file locally in the clique so the originator can capture it.

5 Performance

We have implemented Herbivore as described in the preceding sections. The system consists of approximately 27,000 lines of Java and C code, 2,000 of which comprise the GUI for anonymous filesharing and a helper application for k-anonymous chat while the rest form the core system. While a full evaluation of the system's performance is out of the scope of this paper, we present the results from a bulk transfer experiment, consisting of 45 averaged runs, performed on geographically distributed nodes on Planetlab.

It is well-understood that anonymous communication involves a tradeoff between anonymity and performance. Figure 2 shows that HerbivoreFS can achieve high transfer bandwidth while providing strong anonymity. This performance is due to two factors: small clique sizes enable the DC-net protocol to operate efficiently, and executing many rounds in parallel increases bandwidth significantly. We can expect the performance of the system to drop when home computers behind slow links are permitted to join the system, as bandwidth is proportional to the slowest link. One way to compensate for this is to have separate Herbivore networks for hosts with different limiting bandwidths. Note that Herbivore's bandwidth and latency will not degrade with increasing numbers of hosts in the global network.

6 Future Work and Conclusions

Due to their inherently limited scale, DC-Nets have long been considered theoretically interesting but infeasible in practice. Consequently, we know of no implementations of DC-Nets prior to Herbivore.

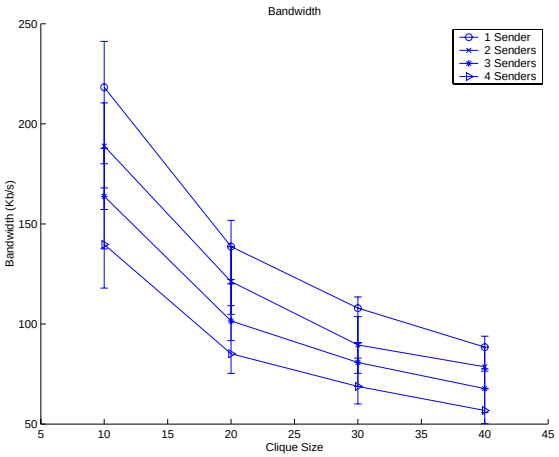

Figure 2: Anonymous communication bandwidth with Herbivore on PlanetLab. Error bars indicate interquartile ranges.

Broadcast protocols, of which P5 is a scalable variant, are the only other approach that offer comparable or stronger anonymity guarantees, but even P5 has never been implemented, partly because of the performance limitations and resource consumption of constant broadcast. The Herbivore prototype demonstrates that DC-Nets are not just theoretically interesting, but also feasible in practice, and that strong anonymity, scalability and performance are not mutually exclusive. Our divide-and-conquer approach enables Herbivore to scale without having to rely on any centralized servers or infrastructure. As we gain more experience with the deployment, we plan to address problems that might arise in practice. In particular, since anonymous systems are prone to abuse [7], a commitment and trap scheme [13] to track down link-level clique disruptors may be necessary.

Overall, HerbivoreFS is a peer-to-peer filesharing system that simultaneously provides strong anonymity, scalability and efficiency. It provides stronger anonymity guarantees than source-rewriting schemes common in P2P networks, and demonstrates that DC-nets can be adopted to support large-scale peer-to-peer applications through a divide-and-conquer approach which decouples the performance of the system from the network size. Our implementation of the system achieves high anonymous trans-

fer bandwidths on the Internet.

References

[1] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of OSDI*, Boston, MA, December 2002.

[2] D. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[3] D. Chaum. The Dining Cryptographers Problem. *Journal of Cryptology*, 1(1):65–75, 1988.

[4] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. *LCNS-2009*, 2001.

[5] M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th CCS*, Washington, D.C., November 2002.

[6] I. Goldberg and A. Shostack. Freedom Network 1.0 Architecture, November 1999. http://www.freedom.net/.

[7] D. Mazieres and M. F. Kaashoek. The design, implementation and operation of an email pseudonym server. In *Proceedings of the Conference on Computer and Communications Security*, pages 27–36, 1998.

[8] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[9] A. Rowstron and P. Druschel. Pastry: Scalable, Distributed Object Location and Routing for Large-scale Peer-to-peer Systems. In *Proceedings of Middleware 2001*, Heidelberg, Germany, Nov 2001.

[10] R. Sherwood, B. Bhattacharjee, and A. Srinivasan. P5: A Protocol for Scalable Anonymous Communication. In *Oakland Security Conference*, 2001.

[11] C. Shields and B. N. Levine. A Protocol for Anonymous Communication over the Internet. In *Proceedings of CCS*, pages 33–42, 2000.

[12] 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.

[13] M. Waidner and B. Pfitzmann. The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability. *Advances in Cryptology, LNCS-434*, 1990.