Improving Web Availability for Clients with MONET

David G. Andersen
Carnegie Mellon University
dga@cs.cmu.edu

Hari Balakrishnan, M. Frans Kaashoek, Rohit N. Rao
MIT Computer Science and Artificial Intelligence Lab
{hari,kaashoek,rohit}@csail.mit.edu

http://nms.csail.mit.edu/ron/ronweb

Abstract

Despite the increasing degree of multi-homing, path and data redundancy, and capacity available in the Internet, today's clients experience outage rates of a few percent when accessing Web sites. MONET ("Multi-homed Overlay NETwork)*, is a new system that improves client availability to Web sites using a combination of link multi-homing and a cooperative overlay network of peer proxies to obtain a diverse collection of paths between clients and Web sites. This approach creates many potential paths between clients and Web sites, requiring a scalable way to selecting a good path. MONET solves this problem using a* waypoint selection *algorithm, which picks a good small subset of all available paths to actively probe.*

MONET runs on FreeBSD, Linux, and Mac OS X, and is deployed at six different sites. These installations have been running MONET for over one year, serving about fifty users on a daily basis. Our analysis of proxy traces shows that the proxy network avoids between 60% and 94% of observed failures, including access link failures, Internet routing problems, persistent path congestion, and DNS failures. The proxy avoids nearly 100% of failures due to client and wide-area network failures, with negligible overhead.

1 Introduction

Web clients experience failure rates as high as a few percent when attempting to connect to Web sites today. To improve this situation, many techniques have been proposed: client-side *multi-homing*, in which the client's access to the Internet uses multiple links, deploying and using redundant paths in the Internet, server-side multi-homing, and server replication. These methods do help, but previous work [10, 7] and our results (Section 4) demonstrate that the resulting availability, defined as the fraction of time that a service is reachable and working, is between 95% and a little over 99%. To put these numbers in perspective, consider the availability figures for the U.S. public telephone system (over 99.99% [26, 19, 13]) and the emergency telephone service (99.994% in 1993 [25]).

The "1.5-2 nines" of availability of current Internet-based systems makes them unattractive for important applications such as medical collaborations and certain financial transactions, both of which often use expensive, dedicated networks today in order to provide the required availability. The desire for high availability is not limited to so-called critical applications—any downtime is expensive for businesses that conduct transactions over the Internet [41]. Even brief interruptions lasting more than a few seconds can degrade user perception of a site's performance and lead to substantial revenue losses.

We seek to improve the availability of client accesses to Web sites by an order of magnitude (one more "nine") or better. We restrict our attention to the Web to make the problem focused and tractable. Despite this narrowed focus, the problem remains challenging, because there are many components whose failure can prevent a client from reaching a Web site. The client's access link may be down; the Domain Name System (DNS) may not respond or may have incorrect information [17]; misconfigurations [22], congestion, and routing pathologies [29] might make the network path between client and server unavailable; or the server itself or its access network may be down. Many of these failures are unpredictable, silent, and have complex root causes.

We propose *MONET (Multi-homed Overlay Network)*, a system that improves Web site availability for clients. Web clients use MONET as a standard Web proxy. MONET attempts to mask failures by obtaining and exploring multiple different end-to-end paths for each HyperText Transfer Protocol (HTTP) request. To help mask failures at different locations in the Internet, MONET finds these paths in three ways: (a) link multi-homing; (b) forwarding requests and responses via a small overlay network of peer MONET proxies; and (c) contacting multiple server replicas. MONET explores paths using probes that check the availability of multiple underlying components.

MONET's end-to-end approach recovers from a variety of failures of the individual components involved in an HTTP request. MONET's protocols and algorithms detect and respond to failures within a small number of round-trips, and with low overhead, sending only a few additional packets. It detects failures regardless of their root cause, providing a measure of resilience against not only network-layer faults, but also persistent congestion, active attacks, misconfiguration, DNS outages, and server-side failures.

MONET uses a *waypoint selection algorithm* that dynam-

ically decides the order in which the many possible paths between client and server should be used, and at what time to use any given path. The algorithm determines this ordering by maintaining statistics about path success rates and connection times through different interfaces and peers. By pruning the large space of possible paths to a handful of the most promising ones, this algorithm reduces MONET's overhead on the network and on Web sites to tolerable levels.

This paper describes a version of MONET that has been in daily use by over fifty people (a conservative estimate; the MONET logs anonymize user activity) at MIT CSAIL since Sept. 2003. The CSAIL proxy is multi-homed to three different ISPs and uses five other peer proxies at different Internet locations.

Our analysis of trace data collected from the MONET installations shows that MONET overcomes at least 60% of all outages (Table 3) and nearly *all* non-server failures (Figure 10), while imposing little overhead . We found that access link failures, wide-area failures, and server-side failures all contributed to the lack of availability and had to be masked. While multi-homing a service alone does not increase its availability (Figure 11), using MONET *in conjunction* with server multi-homing greatly increases availability. This increase arises because MONET reduces client and wide-area failures, *and* because MONET actively seeks out multiple paths to multi-homed sites. MONET achieves significant ("one to two nines") availability improvements at modest cost; for instance, MONET can use a cheap DSL line to greatly increase the availability of a site that already uses BGP multi-homing.

These benefits are tempered by some limitations of the current system. If the different paths available between a proxy and server all share a single point of failure (*e.g.*, a particular network link, a misconfigured DNS database, etc.), MONET will not mask the failure of that element. The current MONET implementation does not mask mid-stream failures that might occur in the middle of a TCP connection; such failures may be recovered from by issuing appropriate HTTP range requests or using transport-layer techniques.

2 MONET Design

MONET consists of a set of Web proxies deployed across the Internet, which serve as conduits for client connections to Web sites. One site might have one or a few proxies, and the entire system a handful to a few hundred proxies.

The goal of MONET is to reduce periods of downtime and exceptional delays that lead to a poor user experience. The idea is to take advantage of several redundant client to server paths, whose failure modes are expected to be mostly independent. MONET must therefore address two questions:

1. How to obtain multiple paths from a client to a site?
2. Which path(s) to use, and at what times?

The answers are shaped by three requirements:

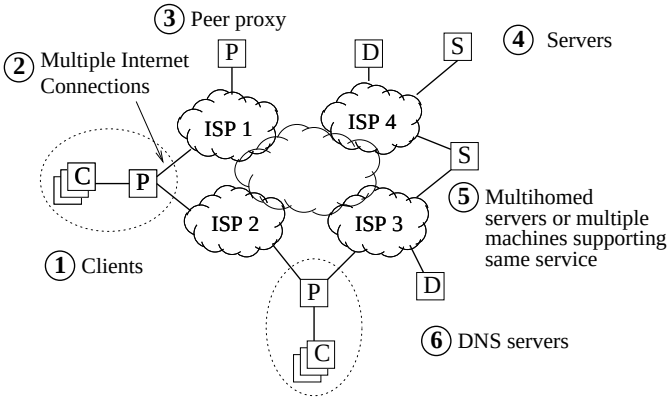

Figure 1. The MONET environment. Clients (1) contact Web sites via a local MONET proxy. That local proxy may be multi-homed with multiple local interfaces (2), and may also route requests through remote peer proxies (3). Clients wish to communicate with web sites (4), which may be themselves multi-homed or spread over multiple machines (5). Web sites must be located using DNS (6); DNS servers are typically replicated over multiple machines.

- **R1** The network overhead introduced by MONET in terms of the number of extra packets and bytes must be low.
- **R2** The overhead imposed on Web servers in terms of TCP connections and data download requests must be low.
- **R3** When possible, MONET should improve user-perceived latency, by reducing the tail of the latency distribution and balancing load on multi-homed links.

The first two requirements preclude an approach that simply attempts concurrent connection requests along all paths between a proxy and Web site.

2.1 Obtaining Multiple Paths

Each proxy has some of the following paths to a Web site at its disposal, as shown in Figure 1. The term *path* refers either to a direct Internet path from one IP address to another, or to an indirect path that goes through an intermediate node.

2.1.1 Multi-homed Local Interfaces

A MONET proxy can obtain Internet access via multiple Internet Service Providers (ISPs), ideally at least two, and perhaps three or four. The proxy can then use a subset of these *local interfaces*, either concurrently or serially, to resolve DNS names and to connect to Web sites. The MONET proxy is assigned one IP address from each upstream ISP, allowing it to direct requests through any chosen provider. This "host-based" multi-homing approach works particularly well for MONET proxies in smaller organizations, providing them the benefits of multi-homing without the complexity of BGP configuration and management.

MONET initiates several TCP connections (sending TCP SYNs) to the server both to probe and to establish a connection over which to request data. The proxy then directs requests only along a link over which a connection succeeded.

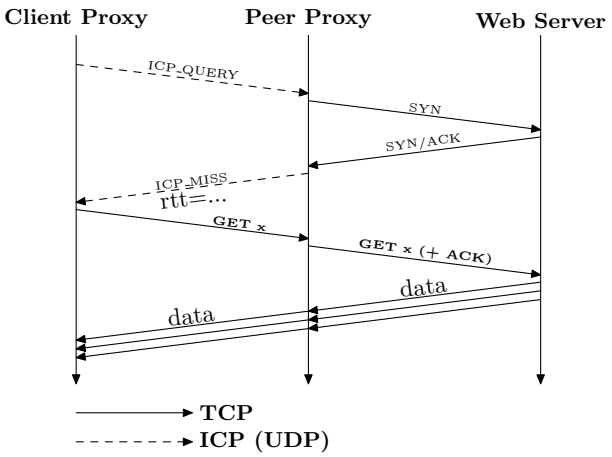

Figure 2. The ICP+ protocol probing a path through a peer. The peer proxy uses a cached DNS response for the site. After the ICP+ probe, the client proxy sends via TCP a request to the peer proxy to fetch the object; the peer retrieves this data over the TCP connection that it used to probe the site during the first exchange.

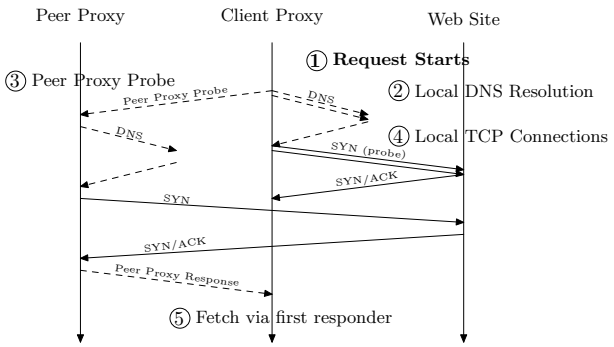

Figure 3. The client proxy performs several queries in parallel. When the request begins (1), it simultaneously begins DNS resolution (2), and contacts peer proxies for the object (3). After DNS resolution has completed, the MONET proxy attempts TCP connections (delayed by the output of the waypoint selection step) to the remote server via multiple local interfaces (4). The remote proxy performs the same operations and returns a reply to the client proxy. The MONET proxy retrieves the data via the local or indirect path that responded first.

This dual use of the TCP SYN packets reduces network overhead, and is an effective tactic for choosing between a set of replicated servers [11]. Only one of these connections will be used to retrieve data.

2.1.2 Paths Through Peer Proxies

An overlay network is a convenient way of obtaining access to multiple paths between two end points, allowing many Internet path failures to be masked [7]. Building upon this observation, MONET attempts to find additional paths using an overlay network of peer proxies. To let MONET probe the availability of these paths, we designed ICP+, a backward-compatible extension to the Inter-Cache Protocol (ICP) [38].

ICP checks whether an object is in a peer's cache. ICP+ extends this check by optionally asking the peer to probe the origin server using a TCP SYN, as described earlier, and return the round-trip connection establishment time. The client proxy can then request the object via a TCP connection to the peer proxy. Figure 2 depicts the operation of ICP+.

An ICP+ query includes the URL of the object that the proxy wants to retrieve through a peer proxy. A peer proxy handles ICP+ queries just like requests from its clients, but the proxy does not contact other peer proxies in turn. MONET proxies handle ICP+ queries as follows:

1. If the object is cached, then reply immediately.
2. If an open connection to the server exists, then reply with that connection's establishment RTT.
3. Else, resolve DNS, perform waypoint selection, ignore other peer paths;
   Open a connection to the server;
   Reply with RTT when TCP established.

The operation of a proxy with one peer proxy is illustrated in Figure 3. This diagram shows one additional benefit of performing the ICP+ queries in parallel with sending TCP SYNs to the origin server: it eliminates delays that the proxy would ordinarily experience waiting for ICP replies. If the ICP replies for a cached object are delayed, the client proxy might fetch the object directly, which is the correct behavior if the origin server is closer than the peer proxy.

2.1.3 Multi-homed Web Sites

Web sites are sometimes replicated on distinct hosts, or are multi-homed using different links to the Internet. The DNS name for a replicated site is often bound to multiple IP addresses. MONET considers each address as corresponding to a different server machine or Internet path, although portions of the paths may be shared (we believe that configurations that deliberately violate this assumption are rare).

Today's Web clients typically contact only one address for a Web site, or they wait between 30 seconds and 13 minutes before contacting subsequent addresses. Because they cannot count on clients to quickly fail over, Web site administrators rely on one of two mechanisms to direct clients to a working server. Many sites use front-end load distributors to direct clients to a host in a cluster. Others answer DNS queries with responses that have very low TTL (time to live) values, forcing clients to frequently refresh the name-to-address mapping for the site. If a server fails, the DNS server stops announcing the failed address. MONET masks failures on shorter timescales without requiring Web sites to set low TTLs in their DNS records.

2.1.4 Multi-path DNS Resolution

A MONET proxy performs at least two concurrent DNS requests (on different local interfaces) to mask DNS failures for two reasons. First, DNS servers are—or should be—

replicated, so finding multiple paths is easy. Second, sending multiple DNS requests does not cause high network overhead because DNS lookups are much less frequent than TCP connections: in our Web traces, 86% of the connections from the deployed MONET proxy to remote servers used a cached DNS entry. This number is consistent with other studies of DNS and TCP workloads [17], which estimated that overall cache hit rates were between 70 and 80%.

Because some server-side content distribution services return DNS responses tailored to a client's location in the network, a MONET proxy performs DNS resolution using only its local interfaces. Each peer proxy performs its own DNS resolution. This localized resolution helps the MONET proxy fetch data from a replica near it.

2.2 Choosing Paths: Waypoint Selection

If a MONET proxy has ℓ local links and r single-homed peer proxies it can use, and if the site has s IP addresses, then the total number of potential paths to the Web site at the proxy's disposal is $\ell \cdot s$ direct paths plus $\ell \cdot r \cdot s$ indirect paths. If each peer proxy has ℓ local interfaces of its own, then the number of paths increases to $\ell \cdot s$ direct paths plus $\ell^3 \cdot r \cdot s$ indirect paths. For even moderate values of ℓ, r, and s, this number is considerable; *e.g.*, when $\ell = 3$, $r = 10$, and $s = 2$, the number of possible paths is 546. When the peer proxies are single-homed, this number is 66, still quite large.

Of course, not all of these paths are truly independent of each other, and pairs of paths may actually share significant common portions. Each path, however, has something different from all the other paths in the set. MONET uses waypoint selection to pick subsets of its paths to probe at different times.

The waypoint selection algorithm takes the available local interfaces, peer-proxy paths, and target Web site IP addresses, and produces an ordered list of these interfaces and paths. Each element of this list is preceded by an optional delay that specifies the time that should elapse before the corresponding path is probed. The proxy attempts to connect to the server(s) in the specified order. The waypoint selection algorithm seeks to order paths according to their likelihood of success, but it must also occasionally attempt to use paths that are not the best to determine whether their quality has changed. MONET attempts these secondary paths in parallel with the first path returned by waypoint selection. If the measured path connects first, MONET uses it as it would any other connection.

Waypoint selection is superficially similar to classical server selection in which a client attempts to pick the best server according to some metric. Under waypoint selection, however, a client can use its history of connections to a variety of servers along different paths to infer whether or not those paths are likely to be functioning, and what the path loss probabilities are. Then, when confronted with a request involving a new server, the client can decide which of its paths are best suited to retrieve data.

2.2.1 Which Paths to Probe

MONET ranks its local links and local link-remote proxy pairs using an exponential weighted moving average (EWMA) of the success rate (fraction of probes that received a response within a timeout period) along each of these paths. It breaks ties using average response time. The algorithm updates the success rate for a local link a short time after sending a TCP SYN or DNS request using that link. Similarly, ICP+ queries update the statistics for the particular local link-proxy pair through which the query was sent.

The proxy sends all DNS requests both on the local link with the highest success rate and also via a randomly selected second local link. The proxy also attempts an additional TCP SYN to the site or sends an ICP+ query to a random peer via a random link between 1% and 10% of the time to measure infrequently used paths.

In designing MONET's waypoint selection algorithm, we considered only schemes that rank the local links and peer proxy paths, regardless of which servers were previously accessed along the various paths. Grouping the success rates by remote site name or IP prefix might yield additional benefit.

2.2.2 When to Probe Paths

To keep overhead small, a MONET proxy should perform the next request attempt only when it is likely that each prior attempt has failed. The delay between requests on different paths must be long enough to ensure this behavior, but short enough so that requests are fulfilled without undue delay. This delay should adapt to changing network conditions.

Measurements of round-trip connect times from the operational MONET proxy at MIT show that their distribution is multi-peaked (the "knee" on the CDF, and the peaks in the histogram in Figure 4), suggesting that the best delay threshold is just after one of the peaks. For example, in this figure, very few arrivals occur between 0.6 and 3.1 seconds; increasing the threshold past 0.6 seconds increases delay without significantly reducing the chances of a spurious probe.

We explored two ways of estimating this delay threshold:

1. **k-means clustering.** This method identifies the peaks in the connect time PDF by clustering connect time samples into k clusters, and finding a percentile cutoff just outside one of the peaks (clusters). The centroids found by k-means with $k = 16$ are shown as horizontal lines in Figure 4. The clustering is relatively insensitive to the value of k.

This method is computationally expensive, particularly if the clustering is recomputed each time a connection attempt succeeds or fails. Even when the threshold is only recomputed periodically, the computational load and memory requirements may exceed what is acceptable for a busy proxy: the k-means clustering requires that the proxy maintain a large history of previous probes.

2. ***rttvar*-based scheme.** To avoid the cost of the k-means scheme, we considered an *rttvar* scheme inspired by TCP retransmission timers. Each delay sample, independent of

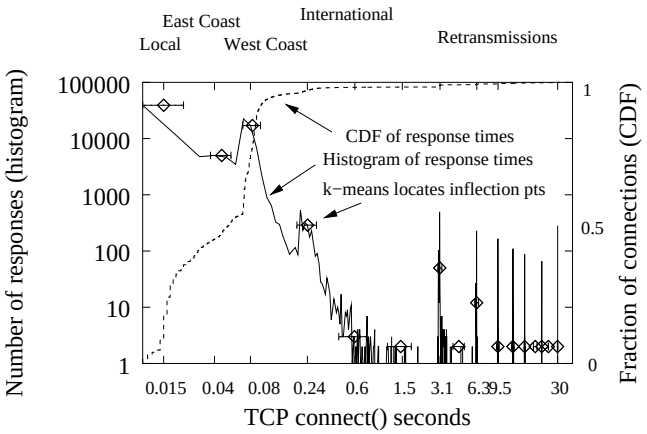

Figure 4. k-means clustering applied to TCP connect times for 137,000 connections from one access link on the east coast to sites around the world. The CDF shows the cumulative fraction of requests amassed by the histogram.

the server contacted or path used, updates an EWMA estimate of the average delay (*rtt*) and another EWMA estimate of the average linear deviation of the delay (*rttvar*). The delay threshold between subsequent requests is set to $rtt + 4 \cdot rttvar$.

The *rttvar* scheme is substantially simpler to calculate than k-means clustering, but it may pick a threshold in the middle of a "valley" between two peaks in the delay sample distribution. In practice, measurements from MONET (*e.g.*, the data illustrated in Figure 4) show that *rttvar* estimates an 800 ms delay threshold, while k-means estimates thresholds of 295 ms (2% false transmission probability), 750 ms (1.6%), and 3.2s (1%). A MONET using the 2% k-means estimator would decide that its first connection had failed after 300 ms instead of 800 ms, reducing the fail-over time for the failed connection. We do not believe that this modest latency improvement justifies the complexity and increased computational and storage requirements of the k-means estimation, and so we chose the rttvar scheme for MONET.

2.3 The Client-MONET Interface

Clients specify a set of MONET nodes, preferably nodes that are close to them in the network, as their Web proxies (one proxy is the primary and the rest are backups). The proxy-based approach allows MONET to be easily and incrementally deployed within an organization, and has been essential to attracting users and gathering data using live user traffic. In addition to ease of deployment, we chose the proxy approach because it provides two other significant benefits:

1. Path information: Because a MONET proxy observes what *site* clients want to contact (such as `www.example.com`), instead of merely seeing a destination IP address, it has access to several more paths for the waypoint selection algorithm to consider when the site is replicated across multiple IP addresses. Moreover, by operating at the application layer and resolving the DNS name of a site to its IP addresses, MONET is able to mask DNS errors; such errors are a non-negligible source of client-perceived site outages and long delays [17, 9].

2. Access control: Many sites control access to content based upon the originating IP address, which is changed by using a different local link or by transiting through a remote proxy. Early users of MONET were occasionally unable to access material in licensed scientific journals, because the proxy had redirected their access through a non-licensed IP address. The deployed MONET proxy is now configured to direct access to 180 licensed web sites through a local interface. As with the CoDeeN proxies [27], this approach also ensures that clients cannot gain unauthorized access to licensed content via MONET.

2.4 Putting it All Together

When presented with a client's request for a URL, MONET follows the procedure shown in Figure 5. The MONET proxy first determines whether the requested object is cached locally. If not, then the proxy checks to see whether the site has successfully been contacted recently, and if so, uses an open TCP connection to it, if one already exists.[1]

Otherwise, the proxy uses MONET's *waypoint selection* algorithm to obtain an ordered list of the available paths to the site. This list is in priority order, with each element optionally preceded by a delay. The proxy attempts to retrieve the data in the order suggested by this list, probing each path after the suggested delay.

If waypoint selection lists a peer proxy first, the request is issued immediately. MONET concurrently resolves the site's DNS name to its corresponding IP addresses to determine which paths are available for local interfaces. To mask DNS failures, the proxy attempts this resolution using all of its local interfaces.

After resolving the domain name, the proxy sends TCP SYN probes from the selected local interfaces. The proxy retrieves data from the first probe (SYN or peer-proxy request) that responds. The results of the DNS lookups and path probes update information about path quality maintained by the waypoint selection algorithm.

The MONET approach to masking failures operates on three different time-scales to balance the need to adapt rapidly with the desire for low overhead. The slowest adaptation (days or weeks) involves the deployment of multi-homed local links and peer proxies in different routing domains. Currently, this configuration is updated manually; automating it is an important future task.

The intermediate time scale adaptation, waypoint selection, maintains a history of success rates on the different paths, allowing MONET to adapt the order of path exploration on a time-scale of several seconds.

To respond to failures within a few round-trip times, the proxy generally attempts the first two paths returned by waypoint selection within a few hundred milliseconds, probing

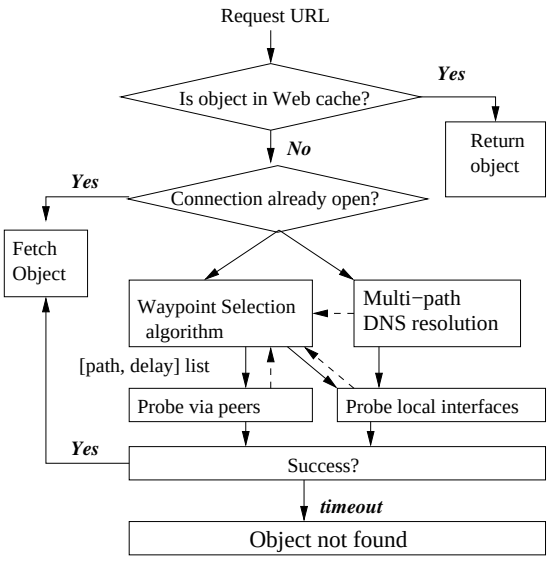

Figure 5. Request processing in MONET.

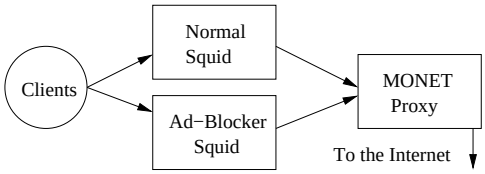

Figure 6. The squid configuration

the rest of the paths within a few seconds. If this order is good, the chances of a successful download via one of the probed paths is high, since the probe includes setting up the connection to the destination site.

Once the proxy has established the connection for a request, it uses the same path. MONET could mask mid-stream failures during large transfers by, for example, issuing an HTTP range request to fetch the remaining content, but the current implementation does not do so. Typical Web workloads consist of many smaller objects, so mid-stream failover will not make much difference for most connections.

3 Implementation

The MONET proxy is implemented as a set of changes to the Squid Web proxy [33] and the `pdnsd` parallel DNS resolver [24], along with a set of host policy routing configurations to support explicit multi-homing. MONET runs under FreeBSD, Linux, and Mac OS X, and should run any POSIX-compliant system that provides a way to support explicit multi-homing.

In the deployed system, Web client configurations are specified with Javascript that arranges for a suitable backup proxy from the specified set to be used if the primary proxy fails. As an extra incentive for users to use the MONET proxy, one front-end blocks common banner ads and pop-up advertisements. Figure 6 shows the Squid configuration.

Because we wanted to evaluate multiple waypoint selection algorithms, the deployed proxy probes *all* of its paths in parallel without performing waypoint selection. We then used subsets of this all-paths data to determine the performance of the waypoint selection algorithms. The currently deployed waypoint selection algorithm returns a static list of (path, delay) pairs that it chooses based upon the name of the destination Web site, to address the access control problems mentioned in Section 2.3.

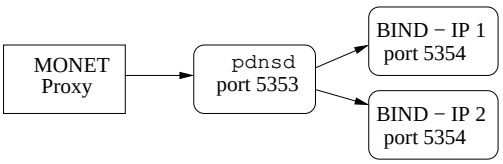

Figure 7. The DNS configuration. `pdnsd` sends queries in parallel to each BIND server, which resolves the query independently.

3.1 Explicit Multi-homing

The MONET proxy and DNS server explicitly bind to the IP address of each physical interface on the machine. MONET uses FreeBSD's `ipfw` firewall rules or Linux's policy routing to direct packets originating from a particular address through the correct upstream link for that interface.

The MONET proxy communicates with a front-end DNS server, `pdnsd`, running on a non-standard high port. `pdnsd` is a DNS server that does not recursively resolve requests on its own, but instead forwards client requests to several recursive DNS servers in parallel—in our case, to BIND, the Berkeley Internet Name Daemon [5]. An instance of BIND runs on each local interface, as shown in Figure 7. This configuration resolves each DNS query independently on each of the outbound interfaces, so that we can confirm during analysis whether the query would have succeeded or failed if sent on that interface alone. Each BIND resolves the query independently, and rotates through the list of available name servers. Because most domains have at least two name servers [12], MONET usually copes with the failure of one of its links or of a remote DNS server without delay.

3.2 ICP+ with Connection Setup

ICP+ adds two new flags to the `ICP_QUERY` message: `ICP_FLAG_SETUP` and `ICP_FLAG_SETUP_PCONN`. A query with `ICP_FLAG_SETUP` requests that the remote proxy attempt a TCP connection to the origin server before returning an `ICP_MISS`. Peer caches that do not support ICP+—or do not wish to provide ICP+ to that client—simply ignore the flag and reply with standard ICP semantics. Squid supports a mechanism for occasionally sending ICMP ping packets to origin servers, using ICP's option data field to return that ping time in response to an ICP query. ICP+ piggybacks upon this mechanism to return the measured RTT from connection initiation.

Because it is used for probing network conditions, ICP+ uses unreliable UDP datagrams to communicate between

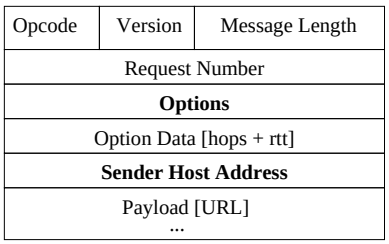

Figure 8. The ICP Packet Format. Bold indicates the fields extended to support ICP+. Brackets show the contents of the fields for Web proxy communication.

peer proxies. Using UDP avoids mistaking temporary failures and packet loss for increased latency, as would happen if a reliable transport protocol like TCP were used for the probes. To treat local interfaces and peer-proxy paths consistently, MONET retransmits lost ICP+ messages with the same 3-second timer that TCP uses for its initial SYN packets. Once a peer has confirmed access to a Web site, the proxies use TCP to transmit objects between them.

MONET uses Squid's persistent connection cache to reduce connection setup overhead. If the originating proxy has a persistent connection open to a Web site, it bypasses peer selection and directly uses the persistent connection, on the assumption that in one of the previous selection attempts, its own connection seemed best. When a remote proxy has a persistent connection to the origin server, it responds immediately to ICP queries, setting the `ICP_FLAG_SETUP_PCONN` flag, and supplying the RTT from when it initially opened the connection.

Figure 8 shows the ICP packet header with the MONET additions in bold. RFC 2187 notes that the sender host address is normally zero-filled. ICP+ uses this field and the request number to suppress duplicates. A multi-homed MONET proxy can transmit multiple ICP+ probes to a peer, from each of its local interfaces to each of the peer's interfaces. On startup, each MONET proxy picks a 32-bit number as its sender ID (*e.g.*, a random number or a local interface address), and uses the same ID when sending via any of its interfaces. The $(sender\ ID, request\ \#)$ tuple uniquely identifies each request and allows a peer proxy to not send multiple identical requests to a Web server. This mechanism provides additional redundancy between proxies without imposing additional server overhead.

Finally, we note that ICP's lack of authentication causes several known security flaws. The newer UDP-based HyperText Caching Protocol (HTCP) [37] supports strong authentication of requests. HTCP requests also carry request attributes such as cookies that may affect whether an object can be served from cache or not. Our HTCP-based MONET is functionally identical to the ICP-based version. The deployed system uses the more mature ICP+ implementation.

3.3 Reducing Server Overhead

Waypoint selection greatly reduces the number of wasteful connection attempts. MONET must also ensure that the few remaining connection attempts do not unnecessarily create server state. Because modern servers minimize processing of SYN packets (to thwart denial-of-service attacks) using techniques like SYN Cookies [8] and SYN caches [21], MONET can send multiple SYN packets without incurring serious overhead, as long as *exactly one TCP three-way handshake completes*, since a connection consumes significant server resources once the server receives the final ACK in the three-way TCP handshake. After opening one connection successfully, MONET closes the remaining probe connections. If this close occurs before the kernel sent an ACK for the connection, the overhead is avoided. We have proposed a simple kernel modification that reduce the overhead even further, and enables applications to change servers at earlier points in the connection attempt [6]; we omit a detailed discussion because of space constraints.

4 Evaluation

Our experimental evaluation focuses on the number of "nines" of availability achieved with and without MONET. The number of nines does not capture all aspects of availability (such as the rate at which failures occurred and how long they lasted), but it does give a good idea of overall availability (and downtime) with and without MONET.

We address the following questions:

1. To what extent do subsystems such as DNS, access links, *etc.* contribute to failures incurred while attempting to access Web sites?
2. How well does MONET mask failures, what is its overhead, and how does it compare against an idealized (but high-overhead) scheme that explores all available paths concurrently?
3. What aspects (physical multi-homing, peer proxies, *etc.*) of MONET's design contribute to MONET's observed improvement in availability? Is MONET useful if BGP multi-homing is already used at the client?
4. How much more of an availability improvement does MONET provide if the Web site is replicated?

4.1 MONET Testbed and Data Collection

We deployed the MONET proxy at six sites from the RON testbed, listed in Table 1. This analysis examines requests sourced from two of these proxies, CSAIL and Mazu , both of which are physically multi-homed. The CSAIL proxy has three peers and uses three local links:

1. **MIT:** A 100 Mbits/s link to MIT's network. MIT's network is itself BGP multi-homed to three different upstream ISPs.
2. **Cog:** A 100 Mbits/s link from Cogent.
3. **DSL:** A 1.5 Mbits/s (downstream), 384 Kbits/s (upstream) DSL link from Speakeasy.

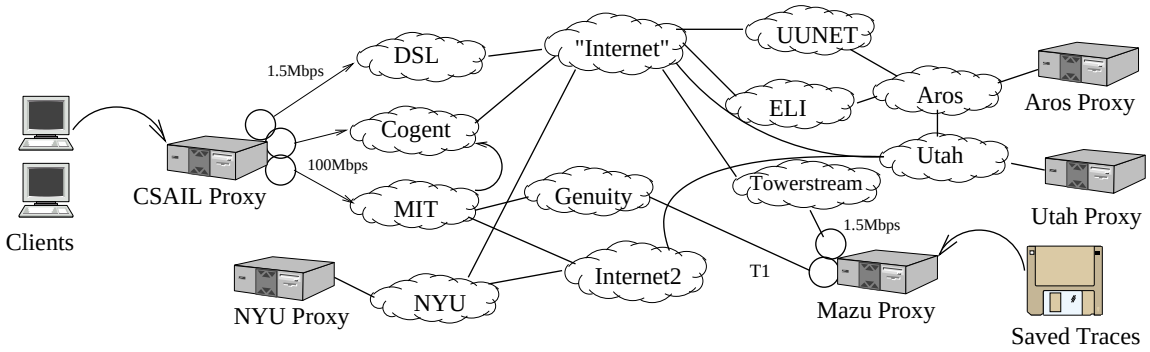

Figure 9. A partial AS-level view of the network connections between five of the deployed MONET proxies (the mediaone and CMU proxies are not shown). The CSAIL proxy peers with NYU, Utah, and Aros; the Mazu proxy peers with CSAIL, Aros, and NYU. The other sites are not directly multi-homed and do not have a significant number of local users; their traces are omitted from the analysis.

Site	Connectivity	Times
CSAIL	3: 2x100Mb, 1.5Mb DSL	6 Dec - 27 Jan 2004
Mazu	2: T1, 1.5Mb wireless	24 Jan - 4 Feb 2004
Utah	1 (US university - West)	proxy-only
Aros	1 (US local ISP - West)	proxy only
NYU	1 (US university - East)	proxy-only
Cable	2: DSL, Cable	22 Sep - 14 Oct 2004

Table 1. The sites at which the MONET proxy was deployed. Mazu's wireless connection uses a commercial wireless provider. The cable modem site was operational for one year, but was monitored only briefly.

Request type	Count
Client objects fetched	2.1M
Cache misses	1.3M
Client bytes fetched	28.5 GBytes
Cache bytes missed	27.5 GBytes
TCP Connections	616,536
Web Sessions	137,341
DNS lookups	82,957

Table 2. CSAIL proxy traffic statistics.

The Mazu proxy uses two different physical access links: a 1.5 Mbits/s T1 link from Genuity, and a 1.5 Mbits/s wireless link from Towerstream. Figure 9 shows the Autonomous Systems (AS) that interconnect our deployed proxies.

The CSAIL proxy has the largest client base, serving about fifty different IP addresses every day. It has been running since April 2003; this evaluation focuses on data collected during a six-week period from December 6, 2003 until January 27, 2004. Analysis of a second one-month period from Sep-Oct 2004 showed results similar to those presented here. Table 2 shows the traffic statistics for the CSAIL proxy.

The MONET proxies record the following events:

1. **Request time:** The time at which the client (or peer) request arrived at the proxy, and, if the request was served, the time at which the HTTP response was sent to the requester. For uncached objects, the proxy also maintains records of the following three events.
2. **DNS resolution duration:** The time at which the proxy made a request to `pdnsd`. For uncached DNS responses, the time at which DNS requests were sent on each local link, and the times at which the corresponding responses were received (if at all).
3. **TCP connect duration:** The time at which TCP SYN packets were sent on each local link and the times at which either the TCP `connect()` call completed, or a TCP connection reset (RST) packet was received.
4. **ICP+ duration:** The time at which the proxy sent an ICP+ message to a peer proxy, the time at which it was received by the peer proxy, and the time at which the ICP+ response returned.

In our experiments, when the proxy receives a request for an object from a Web site, it attempts to contact the Web site using *all* of its local interfaces and all of its peer proxies. The proxy records the time at which the original request was received and the times at which the connection establishment steps occurred using each of the local interfaces and peer proxies. Because the proxy uses all of its interfaces concurrently, the later analysis can examine the performance of a proxy that used only a subset of the interfaces. The analysis then simulates the effects of different waypoint selection algorithms by introducing various delays before additional interfaces are used.

We make a few observations about the data collected from the MONET proxies:

Caching effects: 37% of valid objects were served from the cache, saving about 3.5% of the requested bytes. As in previous studies, a few large transfers dominated the proxy's byte-count, while the majority of the sessions consisted of smaller requests. These cache hits reduce user-perceived delays, but do not mask many outages: numerous pages either required server re-validation, or included uncached objects.

Sessions: We primarily examine the success or failure of a *session*, defined as the first request to a particular server

for a Web site after 60 seconds or more of inactivity.[2]). Analyzing failures in terms of sessions rather than connections avoids a significant bias—an unreachable server generates only a *single* failed request, but a successful connection generates a stream of subsequent requests, which would give a false sense of higher availability. The proxy also uses persistent connections to fetch multiple objects from the same Web server, which reduces the total number of TCP connections. The proxy attempted 616,437 connections to external Web sites over 137,341 sessions.

Excluded objects: The following requests were excluded from analysis: Web sites within MIT, cached objects, accesses to unqualified hostnames or non-existent domain names (NXDOMAIN), access to subscription-based Web sites for which the proxy performs non-standard handling, and accesses to ten Web sites that consistently exhibited anomalous DNS or other behavior.[3] Excluding NXDOMAIN requests ignores some classes of misconfiguration-based DNS failures. Because internal network failures at the proxy site prevent users' requests from reaching the proxy, the analysis missed network failures that coincided with client failures (*e.g.*, power failures).

We do not claim that the performance of these five Internet links at MIT and Mazu represents that of a "typical" Internet-connected site. In fact, MONET would likely be used in much worse situations than those we studied to group a set of affordable low-quality links into a highly reliable system. These measurements do, however, represent an interesting range of link reliability, quality, and bandwidth, and suggest that MONET would likely benefit many common network configurations.

4.2 Characterizing Failures

The failures observed by MONET fall into five categories, listed below. We were able to precisely determine the category for each of the 5,201 failures listed in Table 3 because the links connecting the CSAIL proxy (from which the bulk of our traces are gathered) never all failed at the same time. The categories of observed failures are:

1. **DNS**: The DNS servers for the domain were unreachable or down. The originating proxy contacted multiple peer proxies, and no local links or peers could resolve the domain.
2. **Site RST**: The site was reachable because a proxy saw at least one TCP RST from a server for the site being contacted, but no connection succeeded on any local interface, and no peer proxy was able to retrieve the data. TCP RST packets indicate that the server was unable to accept the TCP connection.
3. **Site unreachable**: The site was unreachable from multiple vantage points. The originating proxy contacted at least two peer proxies with at least two packets each, but none elicited a response from the site.
4. **Client Access**: One or more of the originating proxy's access links did not work for resolving DNS names, establishing a TCP session to a server for the Web site, or for contacting any of the peer proxies.
5. **Wide-area**: A link at the originating proxy was working, but the proxy could not use that link either to perform DNS resolution or to contact a server for the desired Web site. Other links and proxies *could* resolve and contact the site, suggesting that the failure was not at either the client access link or the server.

	CSAIL			Mazu	
	137,612 sessions			9,945 sessions	
Failure Type	**MIT**	**Cog**	**DSL**	**T1**	**Wi**
DNS	1	1	1	1	1
Site RST	50	50	50	2	2
Site Unreach	173	173	173	21	21
Client Access	152	14	2016	0	5
Wide-area	201	238	1828	14	13
Availability	99.6%	99.7%	97%	99.7%	99.6%

Table 3. Observed failures on five Internet links at two sites. The DNS, RST and Unreach rows represent per-site characteristics and are therefore the same for each link at a given proxy.

4.2.1 DNS and Site Failures

After filtering out ten sites with persistent DNS misconfigurations, each proxy observed only one total DNS failure. In both failures, all servers for the domain were on the same LAN. Because DNS resolvers already fail-over after a timeout, MONET's primary benefit is reducing long DNS-related delays.

The 173 site failures in Table 3 show times when no proxy could reach the site but could reach other proxies and other sites. If the proxy received TCP RSTs from the failed site, the server host or program was at fault, not the network. Roughly 20% of the identified site failures sent RSTs to the CSAIL proxy, and 10% sent RSTs to Mazu .

Because of peer proxy restarts and crashes, 8.2% of sessions at the CSAIL proxy never contacted a peer proxy. This analysis thus underestimates the benefits from the overlay, and undercounts the number of site failures by a small margin. We expect to miss about 8.2% (18) of the 223 site failures. In 6.3% (14) instances, MONET could not reach any peers or the site. In our later analysis, most of these instances are probably incorrectly identified as MONET failures instead of unreachable sites. Supporting this conclusion, the proxies observed RSTs from three of the servers in these instances of "MONET failures," similar to the 20% RST rate with the identified server failures. We believe, therefore, there were no instances in which the proxies were unable to reach a functioning site—not surprising, given the number and quality of links involved.

To determine whether this analysis correctly identified failed sites, we re-checked the availability of the unavailable sites two weeks after the first data collection period. 40% of failed sites were *still* unreachable after two weeks. Many of the observed failures were probably attempts to contact per-

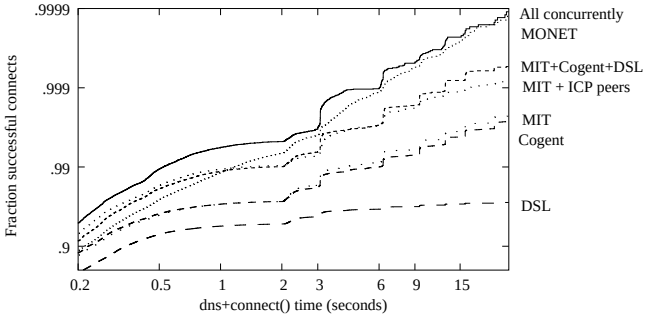

Figure 10. MONET performance at CSAIL. MONET with waypoint selection is nearly as effective as using all paths concurrently, but with only 10% overhead. The MIT+Cogent+DSL and MIT+ICP peers lines use the paths concurrently without waypoint selection delays.

manently failed or non-existent sites.

To better understand how MONET could overcome failures that prevent a client from reaching properly functioning sites, the rest of this analysis excludes positively identified server-side failures. To put these numbers in perspective, Section 4.2.1 examines the (site failure-included) performance of MONET to both all sites, and to a more reliable subset of replicated sites.

4.2.2 Client access link failures

Most links other than the DSL line displayed good availability—near 99.9%. Such high link availability is expected in the environments we measured; for example, MIT (one of the CSAIL proxy's upstream links) is itself connected to the Internet to three upstream ISPs. The remaining unavailability occurred despite the relatively high availability of the links themselves; BGP multi-homing does not provide an end-to-end solution to failures or problems that occur in the middle of the network or close to the server.

We observed one ten-hour failure of the Mazu wireless link during two weeks of monitoring, but it occurred from 9:45pm until 7:45am when little traffic was being replayed through the proxy. The DSL link experienced one 14-hour failure and numerous smaller failures over several months.

We also measured the "global" availability of each link by constantly probing whether or not the link could reach any of the 13 root nameservers. The availability of the links when measured in this fashion is very close to the availability measured through MONET (see [6] for details).

4.3 How Well does MONET Work?

The CSAIL proxy has provided uninterrupted service through 20 major network outages over a 12-month period.[4] One of our most notable results was the ability of a cheap DSL line to improve the availability of the MIT network connection by over an order of magnitude, which we discuss below.

Much of the following analysis concentrates on the effect that MONET has on long delays and failures. To see the overall effects of the proxy, we examine the cumulative distribution of requests whose DNS resolution and `SYN ACK` were received in a certain amount of time, omitting positively identified server failures. Figure 10 shows the "availability" CDFs for MONET and its constituent links at the CSAIL proxy, produced by calculating the fraction of sessions that successfully connected within the time specified by the x-coordinate. This graph and those that follow are in log-scale. The y-axis for the graphs starts near the 90th percentile of connections. The top line, "All concurrently," shows availability when using all paths concurrently, which the proxy performed to gather trace data. A waypoint algorithm simulator picks the order in which MONET's waypoint selection algorithm uses these links, and examines the performance of combinations of the constituent links and peer proxies. MONET's waypoint selection algorithm (Section 2.2) rapidly approaches the "All concurrently" line, and outperforms all of the individual links.

MONET has two effects on availability. First, it reduces exceptional delays. For example, on the Cogent link in Figure 10, 2% of the HTTP sessions require more than 3 seconds to complete DNS resolution and a TCP `connect()`. Combining the MIT link with the Cogent link (which is already one of MIT's upstream ISPs) provides only a small improvement, because packets leaving MIT for many destinations already travel via Cogent. When these links are augmented with a DSL line, however, only 1% of sessions fail to connect within three seconds. The improvements in the 1-3 seconds range are primarily gained by avoiding transient congestion and brief glitches.

The second effect MONET has is improving availability in the face of more persistent failures. Overall, MONET improves availability due to to non-server failures by at least an order of magnitude (*i.e.*, by at least one "nine"). The "MIT+ICP peers" curve in Figure 10 shows that adding remote proxies to a high-uptime link (MIT) can create a more robust system by allowing application-level path selection using existing path diversity. A proxy can realize similar availability benefits by augmenting its primary link with a slower and less reliable DSL line ("MIT+Cogent+DSL"). If a site's primary link is already extremely good, the peer proxy solution increases availability without requiring additional network connectivity, and without periodically directing requests via a much slower DSL line. The benefits of using MONET without local link redundancy will, of course, be limited by the overall availability of the local link. For example, "MIT+ICP peers" achieves 99.92% availability, nearly three times better than the MIT link alone.

4.3.1 Overhead

MONET's waypoint selection algorithm nearly matches the performance of "All concurrently," but adds only 10% more SYNs and 5% more ICP+ packets than a client without MONET. The average Web request (retrieving a single object) handled by our proxy required about 18 packets, so this

additional overhead comes to about 7% of the total packet load, and is a negligible addition to the byte count. The added packets are small—TCP SYN packets are 40 bytes, and the ICP+ query packets are on average around 100 bytes. The mean Web object downloaded through the MIT proxy was 13 kilobytes. The extra SYN and ICP+ packets added by MONET therefore amount to an extra nine bytes per object on average.

The simulation of the waypoint selection algorithm chose a random link to use either 5% or 10% of the time. The benefit of more frequent link probes was at most a 100-200 ms savings in the amount of time it took to find an alternate path when the first path MONET attempted to use had failed—and oftentimes, there was little benefit at all. These latency reductions do not appear to justify the corresponding increase in overhead. A better algorithm might find a better ordering of links for fail-over (*e.g.*, by discovering links whose behavior appears uncorrelated), but because failures are relatively unpredictable, we believe that overcoming transient failures is best done by attempting several alternate links. Waypoint selection avoids links and peers that fail for longer than a few seconds, but does not improve latency in the shorter ranges.

Because of remote proxy failures, random path selection performed poorly. We also simulated MONET using a static retransmit timer instead of using the `rttvar`-derived value. With careful tuning for each proxy, the static value could provide good performance with low overhead, but could not adapt to changing conditions over time.

MONET also introduces overhead from additional DNS lookups. As noted in Section 2.2.2, we believe a MONET with multiple local Internet connections should always send at least two DNS queries. Because DNS queries are frequently cached, the overhead is small—the MONET proxy performed 82,957 DNS lookups to serve 2.1 million objects. The mean packet size for the proxy's DNS queries was 334 bytes. Assuming that the average DNS lookup requires 1.3 packets in each direction [17], duplicating all DNS requests would have added 34 megabytes of traffic over 1.5 months, or 0.1% of the 27.5 gigabytes served by the proxy. Given that between 15 and 27% of queries to the root nameservers are junk queries [17], it is unlikely that the wide deployment of MONET-like techniques would have a negative impact on the DNS infrastructure, particularly since a shared MONET proxy helps aggregate individual lookups through caching.

4.3.2 How well could MONET do?

The top two lines in Figure 10 show the performance of all paths ("All concurrently") and MONET's waypoint selection, respectively. At timescales of 1-2 seconds, the scheme that uses all paths out-performs MONET, because a transient loss or delay forces MONET to wait a few round-trip times before attempting a second connection. Before this time, MONET approximates the performance of its best link; by 1 second, MONET approaches the performance of using two links concurrently.

At longer durations of two to three seconds, MONET comes very close to the performance of all-paths. A part of the difference between these algorithms arises from mispredictions by the waypoint algorithm, and a part probably arises from a conservative choice in our waypoint prediction simulator. The simulator takes the "all paths" data as input, knowing, for instance, that a particular connection attempt took three seconds to complete. The simulator conservatively assumes that the same connection attempt one second later would *also* take three seconds to complete, when in reality it would probably be shorter if the problem were transient.

4.4 Server Failures and Replicated Sites

MONET still improves availability, though less dramatically, in the face of site failures. MONET is more effective at improving availability to replicated or multi-homed sites than to single-homed sites. The leftmost graph in Figure 11 shows the performance of the "all paths" testing with server failures included. This graph includes requests to non-existent servers that could never succeed—the 40% of servers that were *still* unreachable after two weeks—and represents a lower bound on MONET's benefits.

Replicated Web sites, in contrast, generally represent a sample of more available, and presumably well-managed, sites. This category of sites is an imperfect approximation of highly available sites—at least one of the popular multi-homed sites in our trace exhibited recurring server failures—but as the data illustrated in Figure 11 shows, these sites do exhibit generally higher availability than the average site. The replicated services we measured typically used combinations of clustering, BGP multi-homing, and low-TTL DNS redirection to direct clients to functioning servers.[5]

23,092 (17%) of the sessions we observed went to Web sites that advertised multiple IP addresses. Web sessions to these multiple-address sites are dominated by Content Delivery Networks (CDNs) and large content providers. For example, Akamai, Speedera, the New York Times, and CNN account for 53% of the sessions.

Intriguingly, a single link's access to the multiply announced subset of sites is *not* appreciably more reliable than accesses to all sites (Figure 11, right). The MIT connection achieved 99.4% reachability to all sites, and 99.5% to the multi-homed site subset. When augmented with peer proxies, the MIT connection achieved 99.8% availability. Using all local interfaces, MONET achieved 99.92%, and reached 99.93% after just *six seconds* when using both its local interfaces and peer proxies.

MONET's reduction of network failures is more apparent when communicating with replicated sites. The improved performance in accessing these sites shows that MONET's use of multiple server addresses is effective, and that MONET's techniques complement CDN-like replication to further improve availability.

The foregoing analysis counted as replicated *all* sites that

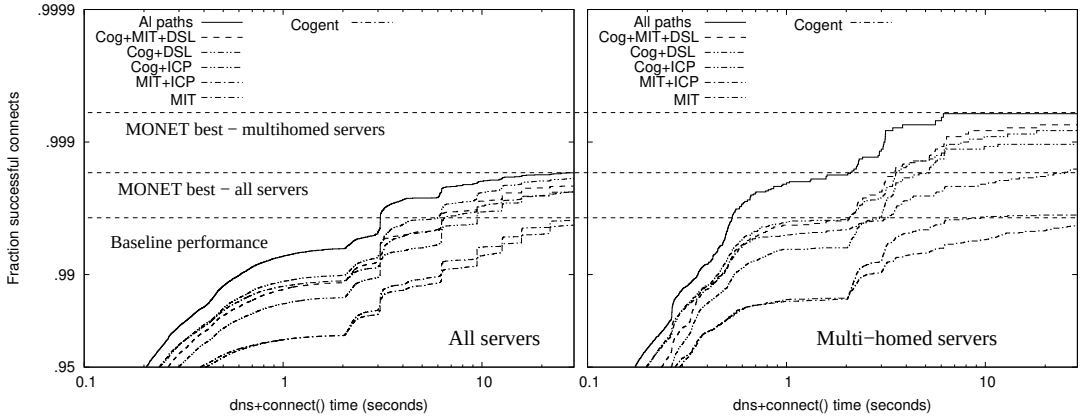

Figure 11. HTTP session success, including server failures, through the CSAIL proxy to all servers (left) and only multi-homed servers (right). The success rate of the base links is unchanged, but MONETs effectiveness is enhanced when contacting multi-homed services.

advertised multiple IP addresses. We assigned sites to a content provider by a breadth-first traversal of the graph linking hostnames to IP addresses, creating clusters of hosted sites. We manually identified the content provider for the largest clusters and created regular expressions to match other likely hosts within the same provider. These heuristics identified 1,649 distinct IP addresses belonging to 38 different providers. While this method will not wrongly assign a request to a content provider, it is not guaranteed to find *all* of the requests sent to a particular provider.

4.5 Discussion and Limitations

MONET masked numerous major failures at the borders of its host networks and in the wide-area. In the cable modem deployment, its ability to balance load between multiple access links provided appreciable performance gains. MONET's benefits are, however, subject to several limitations, some fundamental and some tied to the current implementation:

Site failures: Two power failures at the CSAIL proxy created failures that MONET could not overcome.[6] Improvements provided by the proxy are bounded by the limitations of its environment, which may represent a more significant obstacle than the network in some deployments.

Probes do not always determine success: A failed Internet2 router near MIT's border began dropping most packets larger than 400 bytes. Because MONET uses small ($\sim$ 60 byte) SYN packets to probe paths, the proxy was ineffective against this bizarre failure. While MONET's probes are more "end-to-end" than the checks provided by other systems, there are failures that could be specifically crafted to defeat a MONET-like system. A higher-level progress check that monitored whether or not data was still flowing on an HTTP connection could provide resilience to some of these failures and to mid-stream failures by re-issuing the HTTP request if necessary. Such solutions must avoid undesirable side-effects such as re-issuing a credit card purchase.

Software failures. Several Web sites could never be reached directly, but could always be contacted through a remote proxy. These sites sent invalid DNS responses that were accepted by the BIND 8 name server running on the remote proxies, but that were discarded by the BIND 9 nameserver on the multi-homed proxies. While these anomalies were rare, affecting only two of the Web sites accessed through the MIT proxy, they show some benefits from having diversity in software implementations in addition to diversity in physical and network paths.

Download times. Initial connection latency is a critical factor for interactive Web sessions. Total download time, however, is more important for large transfers. Earlier studies suggest that connection latency is effective in server selection [11], but there is no guarantee that a successful connection indicates a low-loss path. We briefly tested whether this held true for the MONET proxies. A client on the CSAIL proxy fetched one of 12,181 URLs through two randomly chosen paths at a time to compare their download times, repeating this process 240,000 times over 36 days. The SYN response time correctly predicted the full HTTP transfer 83.5% of the time. The objects fetched were a random sample from the static objects downloaded by users of our proxy.

5 Related Work

Benefits from path choice. The RON [7], Detour [31] and Akarouting [23] studies demonstrated that providing clients with a choice of paths to the server increases both peformance and reliability. The RON study found that single-hop overlay routing provided most of the benefits achievable by overlay routing. The recent SOSR work expanded upon these findings, showing that selecting just four random intermediaries provided excellent reliability with low overhead [15]. The SOSR study focused on failures lasting between 30 seconds to six minutes; the MONET results suggest that the SOSR results also apply at shorter time-scales.

Akella *et al.* found that multi-homing two local links us-

ing route control can improve latency by about 25% [2]. The improvements are insensitive to the exact route control mechanism and measurement algorithms [4]. These results complement our findings: MONET focuses primarily on strategies for achieving the reliability benefits of multi-homing (the worst 5 percent of responses), while these studies focus on latency improvements.

Their more recent study of five days of pings between 68 Internet nodes found that most paths have an availability of around 99.9% [3]. These numbers are consistent with our estimates of link failure rates; the remainder of our breakdown analyzes the contribution of other sources of failure and extends this analysis to a much wider set of hosts.

Commercial products like Stonesoft's "Multi-Link Technology" send multiple TCP SYNs to servers to multi-home clients without BGP [36]. RadWare's "LinkProof" pings a small set of external addresses to monitor connectivity on each link, failing over if a link appears down [14]. These systems, and others, can help balance load across multiple links [16]

The Smart Clients approach downloads mobile code to clients, providing flexible and effective server selection. [40]. MONET achieves many of the same reliability benefits without changes to name resolution and without mobile code.

Content Delivery Networks (CDNs) such as Akamai [1] and CoDeen [27] use DNS, server redirects, and client proxy configuration to redirect clients to intermediate nodes, which cache content for quicker access. CDNs deliver replicated popular content and are particularly effective in the face of flash crowds [34, 35], but, without additional reliability mechanisms like those discussed in this paper, are not as effective against network disruptions to un-cached content and access link failures. In fact, our results showed that MONET can improve the performance of CDN-hosted sites.

CoDNS [28] masks DNS lookup delays by proxying DNS requests through peers. When CoDNS does not hear a DNS response from its local nameserver within a short static timeout (200 to 500ms, typically), the CoDNS resolver forwards the query to a peer node. When a majority of recent requests get resolved through a peer node, CoDNS instead immediately sends all queries both locally and through the peer.

Multi-homing Techniques. BGP-based techniques recover only from link failures, and require a few minutes to do so [20]. BGP's route aggregation suppresses the announcement of failures within an aggregate, and financial and technical requirements preclude many small clients from using BGP. These limitations are partly addressed by traffic control systems and higher-layer multi-homing techniques.

RouteScience [30] and SockEye [32] use end-to-end measurements to select outbound routes for networks with multiple BGP-speaking Internet links. To control the inbound link, the following systems change the IP address from which traffic originates, forcing traffic to return to one machine augmented with multiple Internet connections or to a specific overlay node. SOSR, Detour, and NATRON [39] all interpose a NAT on outbound traffic; MONET uses an application-layer proxy. While NAT is more general, the MONET proxy provides more information and is easier to partially deploy (Section 2.3). All of these approaches change the outbound IP address in some fairly intrusive way.

6 Conclusion

This paper presented MONET, a Web proxy system to improve the end-to-end client-perceived availability of accesses to Web sites. MONET masks several kinds of failures that prevent clients from connecting to Web sites, including access link failures, Internet routing failures, DNS failures, and a subset of server-side failures. MONET masks these failures by obtaining and exploring multiple paths between the proxy and Web sites, considering paths via its multi-homed local links, via peer MONET proxies, and to multiple server IP addresses. MONET incorporates a waypoint selection algorithm that allows a proxy to explore these different paths with little overhead, while also achieving quick failure recovery, usually within a few round-trip times.

In contrast to approaches that improve a specific component of the end-to-end path from Web client to server, MONET incorporates simple, reusable failure-masking techniques that overcome failures in many different components.

We deployed a single-proxy multi-homed MONET two years ago. The version of the system described in this paper using multiple proxies has been operational for over 18 months, and has been in daily use by a user community of at least fifty users. The MONET code is publicly available.

Our experimental analysis of traces from a real-world MONET deployment show that MONET corrected nearly *all* observed failures where the server (or the server access network) itself had not failed. MONET's simple waypoint selection algorithm performs almost as well as an "omniscient" scheme that sends requests on all available interfaces. In practice, for a modest overhead of 0.1% (bytes) and 6% (packets), we find that between 60% and 94% of all observed failures can be eliminated (on the different measured physical links), and the "number of nines" of non-server-failed availability can be improved by one to two nines.

Our experience with MONET suggests that Web access availability can be improved by an order of magnitude or more using an inexpensive and relatively low speed link (*e.g.*, a DSL link), or using a few other peer proxies. The techniques incorporated in MONET demonstrate that the cost of high Web access availability (three to four "nines") need not be daunting.

We believe that MONET's end-to-end approach addresses all the reasons for service unavailability and our experimental results show that these failures are maskable, except

for server failures themselves. With MONET in place, the main remaining barrier to "five nines" or better availability is server-side failure resilience.

Acknowledgments

The NSDI reviewers and several of our colleagues provided great feedback: Nick Feamster, Mary Lou Godbe, John Guttag, Max Krohn, Stan Rost, Jon Salz, Mythili Vutukuru, and Michael Walfish. Special thanks to our "shepherd," David Wetherall, who helped us look at the paper with fresh eyes after too much editing. We appreciate the (necessarily anonymous) people at MIT who used our proxies for so long to provide us with measurement data, and the administrators who hosted our proxies. This work was funded by the NSF under Cooperative Agreement No. ANI-0225660; David Andersen was funded by a Microsoft Research fellowship.

References

[1] Akamai. http://www.akamai.com, 1999.

[2] AKELLA, A., MAGGS, B., SESHAN, S., SHAIKH, A., AND SITARAMAN, R. A measurement-based analysis of multihoming. In *Proc. ACM SIGCOMM* (Karlsruhe, Germany, Aug. 2003).

[3] AKELLA, A., PANG, J., MAGGS, B., SESHAN, S., AND SHAIKH, A. A comparison of overlay routing and multihoming route control. In *Proc. ACM SIGCOMM* (Portland, OR, Aug. 2004).

[4] AKELLA, A., SESHAN, S., AND SHAIKH, A. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *Proc. USENIX Annual Technical Conference* (Boston, MA, June 2004).

[5] ALBITZ, P., AND CIU, C. *DNS and BIND in a Nutshell*. O'Reilly and Associates, 1992.

[6] ANDERSEN, D. G. *Improving End-to-End Availability Using Overlay Networks*. PhD thesis, Massachusetts Institute of Technology, Feb. 2005.

[7] ANDERSEN, D. G., BALAKRISHNAN, H., KAASHOEK, M. F., AND MORRIS, R. Resilient Overlay Networks. In *Proc. 18th ACM Symposium on Operating Systems Principles (SOSP)* (Banff, Canada, Oct. 2001), pp. 131–145.

[8] BERNSTEIN, D. J. SYN Cookies. http://cr.yp.to/syncookies.html, 1996.

[9] COHEN, E., AND KAPLAN, H. Prefetching the means for document transfer: A new approach for reducing Web latency. In *Proc. IEEE INFOCOM* (Tel Aviv, Israel, Mar. 2000), vol. 2, pp. 854–863.

[10] DAHLIN, M., CHANDRA, B., GAO, L., AND NAYATE, A. End-to-end WAN Service Availability. *IEEE/ACM Transactions on Networking 11*, 2 (Apr. 2003).

[11] DYKES, S. G., ROBBINS, K. A., AND JEFFERY, C. L. An empirical evaluation of client-side server selection algorithms. In *Proc. IEEE INFOCOM* (Tel Aviv, Israel, Mar. 2000), vol. 3, pp. 1361–1370.

[12] ELZ, R., BUSH, R., BRADNER, S., AND PATTON, M. *Selection and Operation of Secondary DNS Servers*. Internet Engineering Task Force, July 1997. RFC 2182 / BCP 16.

[13] ENRIQUEZ, P., BROWN, A., AND PATTERSON, D. A. Lessons from the PSTN for dependable computing. In *Workshop on Self-Healing, Adaptive and Self-Managed Systems* (2002).

[14] GREENFIELD, D. RadWare Linkproof. *Network Magazine* (Dec. 1999). http://www.networkmagazine.com/article/NMG20000427S0005.

[15] GUMMADI, K. P., MADHYASTHA, H. V., GRIBBLE, S. D., LEVY, H. M., AND WETHERALL, D. Improving the reliability of Internet paths with one-hop source routing. In *Proc. 6th USENIX OSDI* (San Francisco, CA, Dec. 2004).

[16] GUO, F., CHEN, J., LI, W., AND CKER CHIUEH, T. Experiences in building a multihoming load balancing system. In *Proc. IEEE INFOCOM* (Hong Kong, Mar. 2004).

[17] JUNG, J., SIT, E., BALAKRISHNAN, H., AND MORRIS, R. DNS Performance and the Effectiveness of Caching. In *Proc. ACM SIGCOMM Internet Measurement Workshop* (San Fransisco, CA, Nov. 2001).

[18] KRISHNAMURTHY, B., AND REXFORD, J. *Web Protocols and Practice*. Addison Wesley, Upper Saddle River, NJ, 2001.

[19] KUHN, D. R. Sources of failure in the public switched telephone network. *IEEE Computer 30*, 4 (1997).

[20] LABOVITZ, C., AHUJA, A., BOSE, A., AND JAHANIAN, F. Delayed Internet Routing Convergence. In *Proc. ACM SIGCOMM* (Stockholm, Sweden, Sept. 2000), pp. 175–187.

[21] LEMON, J. Resisting SYN flood DoS attacks with a SYN cache. In *BSDCon* (Feb. 2002).

[22] MAHAJAN, R., WETHERALL, D., AND ANDERSON, T. Understanding BGP Misconfiguration. In *Proc. ACM SIGCOMM* (Pittsburgh, PA, Aug. 2002), pp. 3–17.

[23] MILLER, G. Overlay routing networks (Akarouting). http://www-math.mit.edu/~steng/18.996/lecture9.ps, Apr. 2002.

[24] MOESTL, T., AND ROMBOUTS, P. pdnsd. http://www.phys.uu.nl/~rombouts/pdnsd.html, 2004.

[25] NETWORK RELIABILITY AND INTEROPERABILITY COUNCIL (NRIC). Network reliability: A report to the nation, 1993. http://www.nric.org/pubs/nric1/index.html.

[26] NETWORK RELIABILITY AND INTEROPERABILITY COUNCIL (NRIC). Network reliability—the path forward, 1996. http://www.nric.org/pubs/nric2/fg1/index.html.

[27] PAI, V. S., WANG, L., PARK, K., PANG, R., AND PETERSON, L. The dark side of the Web: An open proxy's view. In *Proc. 2nd ACM Workshop on Hot Topics in Networks (Hotnets-II)* (Cambridge, MA, Nov. 2003).

[28] PARK, K., PAI, V., PETERSON, L., AND WANG, Z. CoDNS: Improving DNS performance and reliability via cooperative lookups. In *Proc. 6th USENIX OSDI* (San Francisco, CA, Dec. 2004).

[29] PAXSON, V. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, U. C. Berkeley, May 1997.

[30] RouteScience. Whitepaper available from http://www.routescience.com/technology/tec_whitepaper.html.

[31] SAVAGE, S., COLLINS, A., HOFFMAN, E., SNELL, J., AND ANDERSON, T. The End-to-End Effects of Internet Path Selection. In *Proc. ACM SIGCOMM* (Cambridge, MA, Sept. 1999), pp. 289–299.

[32] Sockeye. http://www.sockeye.com/.

[33] Squid Web Proxy Cache. http://www.squid-cache.org/.

[34] STADING, T., MANIATIS, P., AND BAKER, M. Peer-to-peer caching schemes to address flash crowds. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)* (Cambridge, MA, Mar. 2002).

[35] STAVROU, A., RUBENSTEIN, D., , AND SAHU, S. A lightweight, robust P2P system to handle flash crowds. In *IEEE International Conference on Network Protocols (ICNP)* (Paris, France, Nov. 2002).

[36] STONESOFT. Multi-link technology white paper. http://www.stonesoft.com/files/products/StoneGate/SG_Multi-Link_Technology_Whitepaper.pdf, Oct. 2001.

[37] VIXIE, P., AND WESSELS, D. *Hyper Text Caching Protocol (HTCP/0.0)*. Internet Engineering Task Force, Jan. 2000. RFC 2756.

[38] WESSELS, D., AND CLAFFY, K. *Application of Internet Cache Protocol (ICP), version 2*. Internet Engineering Task Force, Sept. 1997. RFC 2187.

[39] YIP, A. NATRON: Overlay routing to oblivious destinations. Master's thesis, MIT, Aug. 2002.

[40] YOSHIKAWA, C., CHUN, B., EASTHAM, P., VAHDAT, A., ANDERSON, T., AND CULLER, D. Using smart clients to build scalable services. In *Proc. USENIX Annual Technical Conference* (Anaheim, CA, Jan. 1997).

[41] ZONA RESEARCH. The need for speed II. *Zona Market Bulletin 5* (Apr. 2001). http://www.keynote.com/downloads/Zona_Need_For_Speed.pdf.

Notes

[1] Caching complements MONET's path selection by improving the delivery of "hot" static objects, but caching alone does not improve availability [10]. In contrast, MONET's path selection also improves access to uncacheable dynamic content such as that found in Web-based commerce applications.

[2] The gaps between requests from a single user to one Web site are usually under 60 seconds ([18], pp. 394

[3] Several sites had persistent DNS lame delegations; another site always returned `0.0.0.0` as its IP address. One site returned DNS names that exceeded the 128 byte capture length used to obtain the DNS packets.

[4] During the first outage it experienced, we realized that the proxy failed to perform redundant DNS lookups; fixing this shortcoming permitted uninterrupted service during all known outages thereafter. Many of these early outages occurred before our detailed measurement period.

[5] This analysis assumes that being able to reach a replica denotes service availability, which may or may not be the case with some caching CDNs.

[6] It is likely that most of the clients also had power failures, but clients accessing the proxy from other buildings may have been affected.