An Architecture for Content Routing Support in the Internet

Mark Gritter, David R. Cheriton
Computer Science Department
Stanford University
{mgritter,cheriton}@dsg.stanford.edu

Abstract

The primary use of the Internet is content distribution — the delivery of web pages, audio, and video to client applications — yet the Internet was never architected for scalable content delivery. The result has been a proliferation of proprietary protocols and ad hoc mechanisms to meet growing content demand.

In this paper, we describe a content routing design based on *name-based routing* as part of an explicit Internet content layer. We claim that this content routing is a natural extension of current Internet directory and routing systems, allows efficient content location, and can be implemented to scale with the Internet.

1 Introduction

With the emergence of the World Wide Web, the primary use of the Internet is content distribution, primarily in the form of web pages, but increasingly audio and video streams as well. Some measurements indicate that 70 to 80 percent of wide-area Internet traffic is HTTP traffic. Much of the remainder consists of RealAudio streams and DNS[4, 12]. That is, almost all of the traffic in the wide area is delivery of content, and ancillary traffic to locate it. Today, millions of clients are accessing thousands of web sites on a daily basis, with relatively few popular sites supplying a large proportion of the traffic. Moreover, new popular web sites and temporarily attractive web sites can prompt the sudden arrival of a "flash crowd" of clients, often overwhelming the resources of the associated servers.

To scale content delivery to support these demands, more and more content providers and content hosting sites are replicated at multiple geographically dispersed sites around the world, either on-demand (i.e. caching) or by explicit replication. The problem is then to route client requests to a nearby replica of the content, the *content routing* problem.

In this paper, we argue that current content routing designs are unsuitable due to their closed nature and scalability limits. Section 3 describes a content routing system that forms part of an explicit Internet *content layer*, and we claim that this system provides better latency and scalability than current approaches. We support these arguments by an analysis of the scalability of name-based routing in section 4, and close with a description of related work, future goals, and conclusions.

2 The Content Routing Problem

The goal of content routing is to reduce the time needed to access content. This is accomplished by directing a client to one of many possible content servers; the particular server for each client is chosen to reduce round-trip latency, avoid congested points in the network, and prevent servers from becoming overloaded. These content servers may be complete replicas of a popular web site or web caches which retrieve content on demand.

Currently, a variety of *ad hoc* and, in some cases, proprietary mechanisms and protocols have been deployed for content routing. In the basic approach, the domain name of the desired web site or content volume is handled by a specialized name server. When the client initiates a name lookup on the DNS portion of the content URL, the request goes to this specialized name server which then returns the address of a server "near" the client, based on specialized routing, load monitoring and Internet "mapping" mechanisms. There may be multiple levels of redirection, so that the initial name lookup returns the address of a local name server which returns the actual server to be used, and the client must send out additional DNS requests.

As shown in figure 1, a client which misses in the DNS cache first incurs the round-trip time to access the DNS root server, to obtain the address of the authoritative name server for a site, e.g. `microsoft.com.`. Next the client must query this

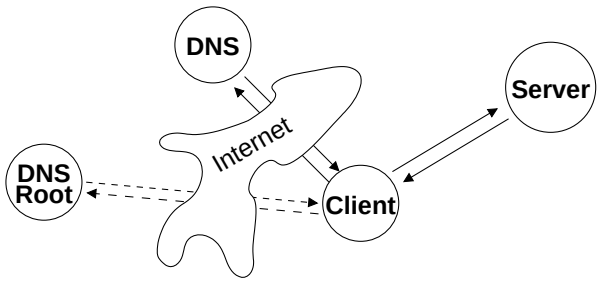

Figure 1: Conventional Content Routing

name server to receive the address of a nearby content server, incurring another round-trip time. Finally, it incurs the round-trip time to access the content on the designated server. If in this example the client is located in Turkey, the first round-trip is likely to go to Norway or London. The second round-trip may have to travel as far as Redmond, Washington, and the final might be to a content distribution site in Germany.

Thus, the conventional content routing design does not scale well because it requires the world-wide clients of a site, on a cache miss, to incur the long round-trip time to a centralized name server as part of accessing that site, from wherever the client is located. This round-trip times are purely overhead, and are potentially far higher than the round-trip times to the content server itself; this long latency becomes the dominant performance issue for clients as Internet data rates move to multiple gigabits, reducing the transfer time for content to insignificance. These name requests may also use congested portions of the network that the content delivery system is otherwise designed to avoid.

DNS-based content routing systems typically use short time-to-lives on the address records they return to a client, in order to respond quickly to changes in network conditions. This places additional demands on the DNS system, since name requests must be sent more frequently, increasing load on DNS servers. This can lead to increased latency due to server loads, as well as increased probability of a dropped packet and costly DNS timeout. As shown in [5], DNS lookup can be a significant portion of web transaction latency.

Both of these problems can be ameliorated by introducing multiple levels of redirection. Higher-level names (e.g., `m.contentdistribution.net`) specify a particular network or group of networks and have a relatively long time-to-live (30 minutes to an hour). The records identifying individual servers (such as `s12.m.contentdistribution.net`) expire in just seconds. However, this increases the amount of work a client (or a client's name server) must perform on a "higher-level" cache miss, and requires additional infrastructure. Also, such a design conflicts with the desire for high availability, since alternate location choices are unavailable to a client with cached DNS records in the event of network failure.

Conventional content routing systems may also suffer from other availability problems. A system which uses only network-level metrics does not respond to application-level failure, so a client may be continually redirected to an unresponsive web server. Designs which rely upon measurements taken from server locations may also choose servers which are not suitable from the client's perspective, due to asymmetric routing. A smaller, related, problem is that DNS requests go through intermediate name servers, so that the actual location of the client may be hidden.

Finally, content routing systems may have difficulty scaling to support multiple content provider networks and large numbers of content providers. Some content providers (such as CNN.com) serve HTML pages from a central web site but provide graphics and other high-bandwidth objects from a content delivery network; the URLs of these objects are located under the content delivery network's domain name. This has the advantage of increasing the probability of DNS cache hits, since the same server location information (`akamai.net`, for example) can be used for other sites. However, it does not help increase availability of the site's HTML content or improve latency to access it. Performing content routing on a larger set of domain names in order to improve web latency may result in lower DNS hit ratios, in addition to the costs of a larger database at a content delivery network's name servers.

Obtaining access to the network routing information needed to perform content routing may also be problematic. Content provider networks must either obtain routing information from routers near to their servers (via BGP peering or a proprietary mechanism) or else make direct network measurements. Both these schemes require aggregating network information for scalability, duplicating the existing routing functions of the network. It may also be politically infeasible to obtain the necessary information from the ISPs hosting content servers.

There is also no clear path for integrating access to multiple content delivery networks. In order to do

so, a content provider would have to include an additional level of indirection to decide which CDN to direct clients to. This may be infeasible in practice (for example, if a URL-rewriting scheme is used to indicate the CDN in use), or at the very best difficult due to conflicting mechanisms and metrics. The proprietary approaches to content routing violate the basic philosophy of the Internet of using open, community-based standard protocols, imposing a closed overlay on top of the current Internet that duplicates many of the existing functions in the Internet, particularly the routing mechanisms.

3 Network-Integrated Content Routing

Our approach to the content routing problem is to view it as, literally, a *routing* problem. Clients (and users) desire connectivity not to a particular server or IP address but to some piece of content, specified by name (typically a URL). Replicated servers can be viewed as offering alternate routes to access that content, as depicted in Figure 2. That is, the client can select the path through server 1, server 2 or server 3 to reach the content, assuming each server is hosting the desired content. Thus, it is the same multi-path routing problem addressed in the current Internet in routing to a host.

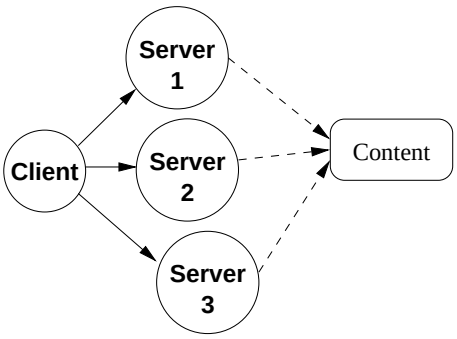

Figure 2: Content-Layer Routing

Network-integrated content routing provides support in the core of the Internet to distribute, maintain, and make use of information about content reachability. This is performed by routers which are extended to support naming. These *content routers* (CRs) act as both conventional IP routers and name servers, and participate in both IP routing and *name-based routing*. This integration forms the basis of the *content layer*. Not every router need be a content router; instead, we expect firewalls, gateways, and BGP-level routers to be augmented while the vast majority of routers are oblivious to the content layer.

3.1 Content Lookup

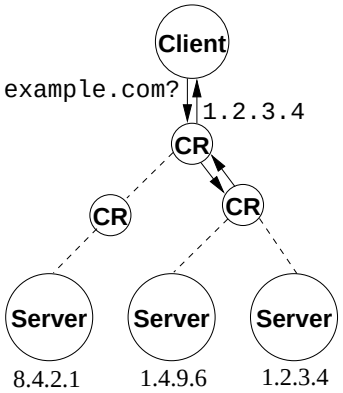

Figure 3: Internet Name Resolution Protocol

Name lookup is supported by the *Internet Name Resolution Protocol* (INRP); this protocol is reverse-compatible with DNS, using the same record types and packet format, but with different underlying semantics. Clients initiate a content request by contacting a local content router, just as they would contact a preconfigured DNS server. Their requests may include just the "server" portion of a URL, although in the long run it would be advantageous to include the entire URL.

Each content router maintains a set of name-to-next-hop mappings, just as an IP router maps address prefixes to next hops. (This name routing information is maintained using a dynamic routing protocol detailed below.) When an INRP request arrives, the desired name is looked up in the name routing table, and the next hop is chosen based on information associated with the known routes. The content router forwards the request to the next content router, and in this way the request proceeds toward the "best" content server, as shown in Figure 3. The routing information kept for a name is typically just the path of content routers to the content server, although it may be augmented with load information or metrics directly measured by a content router.

When an INRP request reaches the content router adjacent to the "best" content server, that router sends back a response message containing the address of the preferred server. This response is sent back along the same path of content routers. If no response appears, intermediate content routers can select alternate routes and retry the name lookup. A client application which is INRP-aware can also

request exclusion of a non-responsive server in an INRP request.

In this fashion, client requests are routed over the best path to the desired content in the normal case, yet can recover from a failing server or out-of-date routing information. INRP thus provides an "anycast" capability at the content level, with network and client control to re-select alternatives based on direct experience with the chosen server.

Routing is done at the granularity of server names rather than full URLs (although the latter could be useful for proxies or content transformers). This decision does limit some possible caching applications, but provides little practical obstacle to web designers, since directory names from the file portion of a URL can be moved into the front of the server name (i.e., `http://foo.com/bar/index.html` can become `http://bar.foo.com/index.html`). Routing is longest-suffix match, since this can be much more efficiently performed than other possible matches on URLs.

Relaying the name lookup request across the same path as the packets are to flow ensures that naming is as available as endpoint connectivity— and that the replica selected is actually reachable. Moreover, the trust in name lookup matches the trust in delivery because both depend on the same set of network nodes. Also, the name lookup load for a path is imposed just on the routers on that path, so upgrading a router on that path for increased data capacity can also upgrade the name lookup capacity on that path. Additionally, we are exploring piggybacking connection setup these name lookups, in which case the name lookup would progress all the way to the content server itself.

3.2 Name-Based Routing

The *Name-Based Routing Protocol* (NBRP) performs routing by name with a structure similar to BGP [15]. Just as BGP distributes address prefix reachability information among autonomous systems, NBRP distributes *name suffix* reachability to content routers. Like BGP, NBRP is a distance-vector routing algorithm with path information; an NBRP routing advertisement contains the path of content routers toward a content server.

At its most basic, a BGP routing advertisement consists of an address range, a next-hop router address, and a list of the autonomous system (AS) numbers through which the advertised route will direct traffic. For example, an advertisement for Stanford's IP address range might specify `171.64/255.192` as the range, `192.41.177.8` as the next hop router, and `7170, 1` as the AS-path.

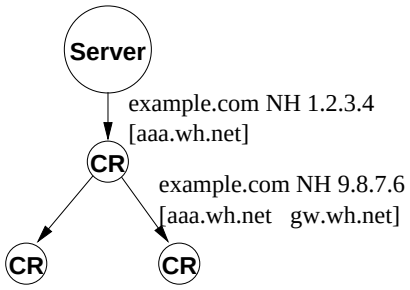

Figure 4: Name-Based Routing

As shown in figure 4, a name-based routing advertisement contains essentially the same information. The advertised content is named `example.com`, the next hop toward that content is the address of the content server or content router, and the path of routers through which the content is accessed.

Routing advertisements from content servers may also include a measure of the load at that server, specified in terms of the expected response latency. This extra attribute indicates that content which takes longer to access appears "further away" from a routing perspective, and may be treated internally by a content router as extra hops in the routing path. The distance this load information is propagated is limited to keep the number of routing updates manageable.

NBRP updates can be authenticated by cryptographic signatures, in a manner similar to Secure BGP [10]. A content server's authenticity is verified by the signature on its initial routing update; content routers receive explicit permission from this content server to advertise routes with their name added to the path list.

Content routers should apply information learned from IP routing to the content routing; if a content peer becomes unreachable then all the content available through that peer is unreachable as well. IP routing information can also be used to select among routes that appear identical at the content routing level. Finally and most importantly, IP routing policies must be consistent with content routing policies so that the decisions made at the content level are faithfully carried out by the IP forwarding level. (It is possible that existing traffic engineering schemes can be used to ensure this behavior; however, we provide some additional ideas on how the two layers can be integrated in the future work section.) Content

routers may also make routing decisions based upon information obtained via measurement and mapping techniques.

3.3 Benefits

Using INRP and NBRP as described above, a client request is mapped to a nearby content server within one round-trip time to a content router near to the client, without the need to contact off-path name servers. Latency is typically dominated by round-trip time to the content server, not by content routing; cache misses require only one RTT to do a new name lookup. Moreover, by increasing the number of content routers, this property is retained even as the Internet scales to ever larger size and increasing number of clients.

By making name lookup low-latency, INRP eliminates the need to perform multiple levels of redirection in DNS. Instead, low-TTL address records can be returned at the first layer of naming, to preserve sensitivity to network conditions. (Assuming, of course, that the content routers can handle the name lookup load required, which will be addressed below.)

By making INRP and NBRP open standard Internet protocols, all ISPs, router manufacturers and content providers can participate in this content routing layer, further enhancing the cost-effective scalability to the clients.

The key issue raised by our solution is the scalability of NBRP, given it is distributing naming and load information, not just aggregatable addressing information. Ideally, we would like to completely replace the current Domain Name System by INRP and NBRP, to remove dependence on root name servers— themselves a large source of connection setup latency and scalability concerns.

4 Scaling Mechanisms for Name-Based Routing

At the global top-level domain name (GTLD) level, the domain name system is essentially flat. There is little aggregation possible with the domain name space beyond that performed at the organization level. For example, most names of the form `*.stanford.edu` appear in the same part of the network, but `stanford.edu` is not aggregatable as part of an `edu` route. So, "default-free" content routers have to know essentially all second-level domain names.

4.1 Explicit Aggregation

To handle large numbers of names which appear globally in name-based routing tables, NBRP supports combining collections of name suffixes that map to the same routing information into *routing aggregates*. For instance, we expect an ISP content router to group all of the names from its customers into a small number of aggregates. Routing updates then consist of a small number of aggregates rather than the large number of individual name entries contained in each aggregate. Load on an entire data center or network may be advertised as load on the aggregates advertised by that data center or network.

Routing aggregate advertisements contain a version number, so that a content router can detect a change in the contents of an aggregate. Aggregate contents are discovered by sending an INRP request back to the router advertising the aggregate; this request is for a "diff" between the last known version and the advertised version, so that large aggregates do not have to be resent in their entirety. Aggregate membership is relatively long-lived, compared to dynamic routing state, so that content routers can amortize the cost of learning the names in an aggregate over many routing updates.

All names in a routing aggregate are treated identically in routing calculation, thus reducing load at content routers. This is accomplished in our implementation by mirroring the indirection provided by aggregates in the routing table, as shown in figure 5. A routing table entry for a name appearing in an aggregate contains a special entry pointing to the entry for the aggregate itself. This indirect pointer is treated as the preferred route for the aggregate. Thus, when the routing information for the aggregate (`calren.net`) changes, the routing information for its constituent names (`stanford.edu` and `bekerley.edu`) is automatically updated.

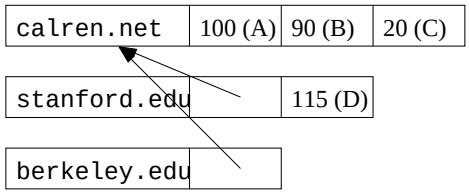

Figure 5: Routing Table with Aggregates

The number of aggregates a name belongs to is likely to be relatively small for all but the largest content providers— which can be advertised unaggre-

site_threshold	Affixes (1000s)	*aggregate_threshold*			
		3	5	10	20
2	1727	19.5 (6.7)	20.1 (5.6)	25.7 (4.4)	37.0 (3.4)
3	1692	14.9 (5.9)	16.1 (5.0)	20.6 (4.0)	30.1 (3.2)
10	1679	14.8 (5.9)	16.0 (5.0)	20.6 (4.0)	30.3 (3.2)

Table 1: Number of routes (and aggregates) in thousands for different site and aggregate threshold values.

gated. So, even though this design requires a linear search through the entire list of routes for aggregated names, we expect that this cost will be relatively small. (It would eliminate much of the benefit of aggregation to re-sort all such lists on a routing change.)

Fine-grain information such as server load is hidden by the aggregate; regions of the network where the aggregate is advertised but individual members are not must base their decision on just the aggregate. This is similar to the way BGP routing provides coarse-grained address range information and does not indicate whether particular hosts or subnets are up or down. In fact, the situation is improved by INRP's request-response nature: unlike datagram delivery, a name lookup can pursue alternate routes if a route proves inaccurate— at the cost of additional latency.

To evaluate the expected performance of aggregation if applied in the current Internet, we processed a comprehensive list of address-to-name mappings in the Domain Name System[6] and BGP table dumps from the MAE-East exchange point[7] by the following algorithm, making the assumption that name-based routing structure will roughly correspond to current BGP autonomous system boundaries:

1. Each address range from the BGP table is matched with the DNS zones represented. (If fewer than *site_threshold* hosts in a range belong to an existing zone, they are removed from the table completely and assumed to be handled with the redirection mechanism described below in section 4.2)
2. Names whose associated routing information is made redundant by a superzone are also removed.
3. Aggregates are created for any set of names larger than *aggregate_threshold* that have identical routing information (i.e., all known routes were identical, not just the preferred route.)

The resulting aggregates, although incomplete, provide an estimate of those expected to be generated by name-based routers.

One representative set of results is shown in Table 1. The original BGP table had 68,200 routing table entries. Aggressive aggregation (site threshold of 10 and aggregate threshold of 3) results in a table with 1,679,000 entries, but only 14,800 advertised names (including 5,900 aggregates). Thus, the number of routes is actually smaller than in the original IP routing table. Even relaxed values of the model parameters result in a routing "back-end" size comparable with the original BGP table. Higher-level aggregation may be able to reduce this yet further without resorting to renumbering or renaming.

The number of routing entries is comparable with the best possible achieved under IP routing. BGP does have a limited mechanism for aggregation: a single route update may include several address prefixes. It is not clear the extent to which BGP software makes use of this to optimize update calculations: there is no requirement that advertisements keep these address prefixes together, and the address ranges must appear separately in the IP routing table. Aggregating all all identical address prefixes would result in 11,800 routing table entries for the original MAE-East table.

And additional challenge NBRP addresses is addition of a new names, which is much more common than addition of new BGP prefixes; this name information must propagate to all default-free content routers. This is far smaller than the rate of normal routing updates, since addition of new names is done on human time scales; even the addition of 10 million new globally routed suffixes per year results in just 19 updates per minute. Some new names are added to multiple locations or duplicated to handle flash crowds, increasing the rate of routing table change. However, the number of routing updates seen by an individual router on name addition is limited by the number of direct peers. To put this in perspective, a backbone router may receive more than 2,000 routing updates per minute; new naming information is dwarfed by the normal rate of topological change.

Also, the actual level of routing updates necessary for new names can be lower in some cases because changes to aggregates can be "batched" to reflect many new names with one update.

The cost of distributing the contents of routing aggregates is acceptable as well, even at Internet scales. The aggregates obtained from the analysis described above show a heavy-tailed distribution; the mean number of names per aggregate is 304, while the median is 24. The average size of the domain names in these aggregates is 16 bytes, although these statistics could be somewhat different if the content routing system was used more aggressively to do redirection on finer granularity. Even an aggregate of 50,000 entries requires only 782 kilobytes to be sent initially, estimating 16 bytes for each name suffix. Later updates can be sent as deltas to the known aggregate, since aggregates can be kept in permanent storage. The long-term bandwidth consumed by aggregate updates (across as single peering connection) can be estimated as

$$\text{number of names} \cdot \frac{2 \cdot (3 \cdot \text{average name size} + 170)}{\text{average name lifetime}}$$

The estimate represents the cost of sending a query (including IP and TCP headers) for a change in aggregate membership and getting the response; this transaction occurs twice since it happens when the name enters and leaves an aggregate. The packet sizes were measured by our implementation and assume one TCP packet per query. Even assuming a relatively short time of 1 day for the average time a name suffix appears in an aggregate, a database of 30 million names (with average size 16) requires 1.2 Mbit/second data transfer. When compared to the 10-gigabit expected capacity of backbone links, this number represents only 0.01 percent of the available bandwidth. Moreover, all aggregate routing update traffic takes place only between two immediate NBRP peers.

4.2 Redirection

Not all hosts with names in a given suffix are connected to the network globally advertising that name suffix. For example, there may be hosts with `stanford.edu` names scattered throughout the Internet, even though most Stanford names are located together. It does not seem feasible to advertise all of these names globally. Such hosts could simply be assigned fixed addresses by the content router operated by Stanford. However, we see some benefit in allowing local flexibility in address assignment without updating a remote server— particularly in situations where network address translation is being used, or in mobile networks.

INRP provides a redirection mechanism for finding isolated names not advertised in the the name-based routing system. Such names have records indicating their actual topological location in the Internet in terms of a more well-known name. When a request is answered with a redirection record, the client (or the first-hop content router acting on its behalf) restarts the query using the proper name. For example, if the host `gritter.stanford.edu` is located at Berkeley, a name lookup might return a redirection to `guest32.berkeley.edu`. This redirection mechanism trades fate-sharing and name lookup latency for decreased routing state; economic factors may well determine what names appear in routing tables and which are found through redirection. That is, an ISP may charge per name it places in the routing table, so an organization weighs the cost-benefit of having a name handled by the ISP versus incurring the redirection cost on a name.

This secondary mechanism is really only needed when using NBRP to replace all DNS usage; for content routing, name-based routing tables contain only site and content volume names rather than host and network interface names.

5 Implementation and Analysis

Our prototype content router has been implemented in C++. The name routing table is implemented as a hash trie, allowing longest-suffix matching to be performed in time linear with name length. (For most names in our sample and experiments this is simply two hash table lookups.) The following measurements were taken on a 600 MHz Pentium III system running Linux 2.2.12.

5.1 Content-Layer Overhead

We measured an overhead of 0.5 milliseconds (total for both request and response) for going through a single hop of the content routing layer, on a name routing table of 5 million entries stored entirely in memory. The 5 million names were randomly generated second-level domain names, with 80% in `.com` and 10% in each of `.org` and `.net`; a uniform distribution of name lengths between 3 and 17 was used. These names were divided into aggregates of 15,000.

Measurements done on the 1.7 million-name database from our aggregation experiment show no significant difference in overhead. Profiling information shows that most of this time is spent doing

packet processing; measurements on the actual routing table show that route lookup takes as little as 6 microseconds. Our implementation can easily sustain a throughput of 650 requests/second without any degradation of response time, and peak throughput of 1600 requests/second.

The total amount of memory used by the content router for a 5 million entry table was 344MB, while that on a similarly generated 100,000 entry (but unaggregated) table was 20MB. This leads to an estimate of 69 bytes per routing table entry. We can extrapolate that a 30-million entry database would require nearly 2GB of memory; while large, this is not an infeasible amount of DRAM, costing only about $4000. (It is worth noting that name lookups which must go to the DNS root *already* encounter a database lookup of approximately this size.)

5.2 Improved Performance with INRP

The improved performance provided by INRP is illustrated by considering access times to content servers through Akamai versus our proposed content routing. An additional example shows the benefit of using INRP rather than contacting root name servers.

A conventional name lookup of `a388.g.akamai.net` from Stanford returns the addresses of two content servers which are located 6.6 ms round-trip-time away.

At the next level, the name servers for `akamai.net` are located throughout the Internet, with an average round-trip times ranging from 12 ms to 93 ms. Overall, this set of name servers has a mean response time of 65 ms and median of 83 ms, ignoring dropped requests.

Using INRP, the same request would go through about 5 content servers (at least one per intervening network), so we will estimate 3 ms extra round-trip time. The direct path to the content servers would then require approximately 10 ms for the name request. A similar example for a miss at the root name servers is carried out in Table 2 for `www.cisco.com`.

As the latency measurements in Table 2 indicate, INRP reduces average request latency in these examples by 86 to 95 percent and also eliminates the variability in latency, providing more predictable performance.

Site	Server Prefix	request latency minimum	mean
Akamai	`akamai.net`	12 ms	65 ms
	`g.akamai.net`	7 ms	7 ms
	Total	19 ms	72 ms
	INRP (5 hops)	10 ms	10 ms (-86%)
Cisco	`com`	9 ms	101 ms
	`cisco.com`	4 ms	40 ms
	Total	13 ms	141 ms
	INRP (5 hops)	7 ms	7 ms (-95%)

Table 2: Example name request round-trip times on cache miss (measured from Stanford) for `a388.g.akamai.net` and `www.cisco.com`.

5.3 Name-Based Routing Performance

We measured the routing throughput of our prototype implementation on a local network using a random routing update traffic generator. A single machine was the source of all routing traffic; an instrumented content router was configured to advertise its preferred routes to a variable number of peered content routers, all connected by a 100Mbit LAN. To maximally exercise the content router, the generated traffic consisted of previously unknown routes and changes to aggregate membership, so that all routing updates were propagated to all configured peers. The routing preference function used was minimal; with more complicated routing polices, the cost of calculating route preferences dominates, so the results presented below should be considered upper bounds on the performance of this particular implementation.

Figure 6 shows the routing throughput for each number of peers. The "no aggregates" data set represents routing updates advertising individual names. Throughput gracefully declines with the number of routing peers, from a maximum of 1050 updates per second with one peer, to 370 updates per second with six peers.

For "1 percent update", 1% of the routing advertisements consisted of a one-name change to an aggregate. As the figure shows, this reduces throughput by approximately 5%, due to the extra queries needed to obtain the new name and the file system accesses to store the new aggregate contents. Increasing the update size to 20 names showed only a 1-2% additional reduction in throughput, indicating there is some benefit to batching aggregate membership changes. Higher proportions of aggregate changes to normal routing updates result in further reductions in throughput; an experiment where all

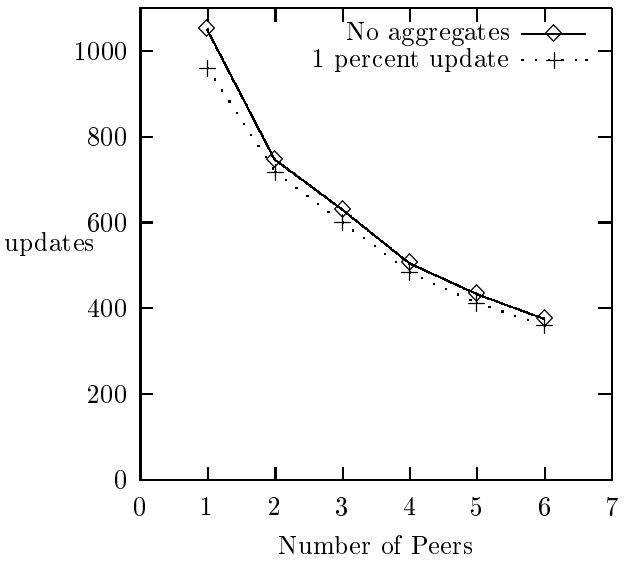

Figure 6: Name-based Routing Performance (updates/sec)

routing updates were aggregate changes resulted in only 87 updates/second.

6 Deployment

The content layer has a simple deployment path, based on user need and on an ISP's motivation to provide a better web experience to customers and superior service to colocated service providers. INRP and NBRP can initially be implemented in ISP name servers, which fail over to normal DNS behavior for unrecognized names. INRP provides a way for ISPs to quickly direct their customers to colocated servers, eliminating any need for name requests to leave their network. NBRP is not strictly needed, but may prove a convenient way to advertise new content to an ISP's name servers.

This initial deployment requires no changes to end hosts and no change to the basic IPv4 routers and switches constituting the infrastructure of the leaf and backbone networks. It only requires the deployment of content routers, which can be implemented on top of existing hardware using packet filtering and redirection techniques. In particular, hosts still use conventional DNS lookup to get an address, but benefit from reduced dependence on distant root name servers and lower latency to access local content servers. However, some customers may be running their own name servers and avoiding the use of ISP DNS servers, and thus see no benefit until they reconfigure.

The overhead of doing content routing in this manner is very small, since the ISP's name server already does DNS packet handling. Only the cost of accessing the name-based routing table would be wasted on names not in the content routing system— much less than the 0.5 ms described above.

ISPs who already peer at the IP routing level are motivated to peer at the content routing level to provide their customers faster access to nearby content servers— and increase the benefit of placing content servers in their network. As demand grows, additional content routers can be placed to handle the increased usage without user-visible changes. As the content routing topology evolves to more closely matches IP routing topology, the content routing system can make more accurate decisions.

7 Related Work

The original Internet directory service was supplied by a "hosts.txt" file that listed all hosts in the Internet. As the Internet grew, this approach was replaced by DNS [13] in 1985. Subsequent work on "network directories" such as X.500 attempts to support naming of other types of objects such as mailboxes and users, and providing more flexible ways of specifying identification, such as lists of attributes. We choose instead to restrict the entities being named (and the namespace) in order to improve scalability.

Content routing builds on the *decentralized naming* approach advocated from experience in the V distributed system [3], That is, as in V, names are the primary identification of objects, hosts and content volumes in this case, and each server implements the naming for the objects it implements. There is no centralized name repository.

Current wide-area content routing depends on HTTP- or DNS-level redirection, and is generally done on the "server side". For instance, Cisco's Distributed Director (DD) redirects a name lookup from the main site to a replica site closer to requesting client address, based on responses from a set of participating routers running an agent protocol, supporting DD. Unfortunately, the client incurs the response time penalty of accessing this main site DD before being directed to the closer site. Proprietary schemes by Akamai, Sightpath, Arrowpoint and others appear to work similarly. These proprietary content distribution networks can be centrally monitored and managed, unlike name-based rout-

ing. This may lead to better understanding of network performance; however, CDNs still rely upon the existing IP routing framework for content delivery, so the amount of benefit to be gained from a proprietary overlay network is limited. Additionally, we believe future work on advanced routing designs and improved network management is applicable to name-based routing as well as IP routing. (As research and experience with BGP has shown [11], wide-area routing is neither easy to understand nor easily tuned.)

Smart Server Selection [14], in contrast, is a "client-side" approach to content routing. An authoritative name server for a content volume returns *all* available addresses for replicas of the content, and the client (or the client's name server) interacts with a nearby router to obtain routing metrics for this set of addresses and chooses the nearest one. This requires cooperation from the router in the form of a request protocol, adding an extra step to the name lookup process. Smart Server Selection also does not address the problem of name lookup latency.

Similarly, some DNS servers measure round-trip times to known name servers in order to choose the lowest-latency server, especially at the root level. Although this can improve the performance of name lookups by lowering the mean lookup latency, it only helps at one level of a cache miss. Further, such name servers are ignorant of network conditions and thus may experience several timeouts before switching their preference to an alternate server.

Intentional naming [1] integrates name resolution and message delivery, offering application-layer anycast and multicast similar to our proposed content layer. The Intentional Naming System offers attribute-based naming, a much richer form of content addressing than URLs or domain names. However, INS is not designed to provide global reachability information, and the attribute-based naming is less scalable than the a hierarchal namespace provided by URLs. INS's "late binding", where every message packet contains a name, is too expensive to use for content distribution; our proposed architecture corresponds to INS's "early binding", where names are resolved to addresses before content is exchanged.

Much work has been done on distributed caching schemes; one design very similar in spirit to our content-layer routing is "adaptive web caching" [16]. In this system, caches exchange information about which web pages they currently hold (in order to eliminate the need for "cache probing") and maintain "URL routing tables". Our design does not offer routing on the granularity of URL prefixes, as adaptive web caching does, but offers a more comprehensive solution intended to replace current naming systems.

Network-level anycast designs such as GIA [8] attempt to solve server location problems at the IP level. The semantics of an anycast IP address are to deliver a packet destined for that address to the "best" of an available pool of servers. GIA does not incorporate application-level metrics, so anycast packets may be routed to an unresponsive server without providing any recourse for the client. Also, unless a client is statically configured with all needed anycast addresses, it must still use a directory to determine the address to use.

8 Future Directions

The motivation for a "content layer" approach came as part of the TRIAD[2] project. TRIAD is a new, NAT-friendly Internet architecture which seeks to reduce dependency on addresses by promoting names as transport-layer endpoints. In a TRIAD Internet, all large-scale routing would occur at the naming level. We believe this approach is ultimately more scalable and deployable than attempts to solve problems (such as mobility, multihoming, anycasting, and wide-area addressing) at the network level.

Two features of TRIAD enhance the content routing architecture. TRIAD provides extended addressing via the *Wide-Area Relay Addressing Protocol* (WRAP), which provides loose-source routing among multiple address realms. WRAP addresses can be used to specify a path through the network, ensuring that the route selected by the content routing layer is the path actually used by data packets. TRIAD also integrates TCP connection setup into INRP name lookup; by sending TCP connection initiation information inside an INRP request, the latency for web transactions can be lowered yet further. Thus, the full TRIAD architecture integrates naming, routing, and connection setup into a single framework.

INRP allows proxies and web caches to intercept content requests based on URL. We have not implemented or fully explored this design, but it appears to be a promising way to provide "semi-transparent" proxies, which would require no explicit configuration at the client, but would be used by the client as a content request's TCP connection endpoint.

Finally, the integration of naming and routing al-

lows *feedback-based routing*. Conventional IP routing schemes have few ways to tell if the routes they select actually deliver packets to the intended destination. Content routers, however, can track the responses they receive to forwarded queries, allowing them to make better decisions and react more quickly to routing problems than conventional routers. For example, if content servers send back load information in INRP responses, then content routers can obtain up-to-date load information on heavily used sites without placing this load information into routing updates.

The most current version of this paper can be found at [9].

9 Conclusions

Current content routing solutions will not scale to handle increasing global demands for content. Conventional content routing distributes content delivery but does not effectively distribute content discovery. Further, the proprietary nature of most content routing designs in use today makes them undesirable for global use and are in conflict with the Internet open standard philosophy.

The content layer — integrated naming and routing — provides a mechanism for large-scale content routing that addresses these issues. By pushing naming information out into the network, content routers allow fast location of nearby content replicas; in essence, content routers provide the same service for naming that CDNs do for the content itself.

We developed NBRP to distribute names in this fashion and INRP to perform efficient lookup on this distributed integrated named-based routing system. Our results indicate that client name lookup is then faster and far less variable.

The content layer can be easily deployed to provide immediate benefits to ISPs and their customers. Our implementation, and the networking community's experience with BGP, give confidence that name-based routing can scale at least to the demands of content routing for popular content. We anticipate that additional research and experience will demonstrate the feasibility of using name-based routing for all Internet naming.

10 Acknowledgments

The TRIAD project was supported by the US Defense Advanced Research Projects Agency (DARPA) under contract number MDA972-99-C-0024. This paper greatly benefited from the guidance of our shepherd, Steven Gribble, and the comments of the USITS reviewers; thanks also to Vincent Laviano and Dan Li for their feedback.

References

[1] William Adjie-Winoto, Elliot Schwartz, Hari Balakrishnan, and Jeremy Lilley, "The design and implementation of an intentional naming system", Proc. 17th ACM SOSP, Dec. 1999.

[2] David Cheriton and Mark Gritter, "TRIAD: A New Next-Generation Internet Architecture", `http://www.dsg.stanford.edu/triad`, July 2000.

[3] D. R. Cheriton and T.P. Mann, "Decentralizing a Global Naming Service for Improved Performance and Fault Tolerance", ACM TOCS, May 1989.

[4] k claffy, G. Miller, and K. Thompson, "The nature of the beast: recent traffic measurements from an Internet backbone", INET 98.

[5] E. Cohen and H. Kaplan, "Prefetching the means for document transfer: A new approach for reducing Web latency", INFOCOM 2000.

[6] Internet Domain Survey, July 1999, `http://www.isc.org/ds/`.

[7] Internet Performance Measurement and Analysis project, `http://www.merit.edu/ipma/`.

[8] Dina Katabi and John Wroclawsi, "A Framework for Scalable Global IP-Anycast (GIA)", SIGCOMM 2000.

[9] Mark Gritter and David R. Cheriton, "An Architecture for Content Routing Support in the Internet", `http://www.dsg.stanford.edu/papers/contentrouting/`, 2001.

[10] S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway Protocol (S-BGP)", IEEE Journal on Selected Areas in Communication, 1999.

[11] Craig Labovitz, G. Robert Malan, and Farnam Jahanian, "Internet Routing Instability", ACM SIGCOMM Conference, September 1997.

[12] Sean McCreary and kc claffy, "Trends in Wide Area IP Traffic Patterns", ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management, September 2000.

[13] P. Mockapetris, "Domain Names – Concepts and Facilities", RFC 882, November 1983 (obsoleted by RFC 1034 and 1035).

[14] W. Tang, F. Du, M. W. Mutka, L. Ni, and A. Esfahanian, "Supporting Global Replicated Services by a Routing-Metric-Aware DNS", Proceedings of the 2nd International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS 2000), June 2000.

[15] Y. Rekhter, T. Li (editors), "A Border Gateway Protocol 4 (BGP-4)", RFC 1771, March 1995.

[16] Lixia Zhang, Scott Michel, Khoi Nguyen, Adam Rosenstein, Sally Floyd, and Van Jacobson, "Adaptive Web Caching: Towards a New Global Caching Architecture", 3rd International WWW Caching Workshop, June 1998.