

EXHIBIT 1

1 Mayela C. Montenegro-Urch (SBN: 304471)
mmontenegro@shb.com
2 SHOOK, HARDY & BACON L.L.P.
Jamboree Center
3 5 Park Plaza, Suite 1600
Irvine, California 92614
4 Telephone: (949) 475-1500
Facsimile: (949) 475-0016
5

6 Robert H. Reckers (*pro hac*)
rreckers@shb.com
7 SHOOK, HARDY & BACON L.L.P.
600 Travis, Suite 3400
8 Houston, Texas 77002
Telephone: (713) 227 – 8008
9 Facsimile: (713) 227 - 9508

10 Attorneys for Plaintiff
11 Sandpiper CDN, LLC
12

13 IN THE UNITED STATES DISTRICT COURT
14 FOR THE CENTRAL DISTRICT OF CALIFORNIA
15

16 SANDPIPER CDN, LLC,

17 *Plaintiff,*

18
19 v.

20 GOOGLE LLC,

21 *Defendant.*
22

Civil Case No. 2:24-03951-AB-RAO

**FIRST AMENDED COMPLAINT
FOR PATENT INFRINGEMENT;
JURY TRIAL DEMANDED**

23 **FIRST AMENDED COMPLAINT FOR PATENT INFRINGEMENT**

24 Plaintiff Sandpiper CDN, LLC (“Sandpiper CDN” or “Plaintiff”) hereby files
25 this First Amended Complaint for Patent Infringement against Google LLC
26 (“Google” or “Defendant”), alleging infringement of U.S. Patent Nos. 8,478,903;
27 8,645,517; 9,021,112; 10,924,573; and 10,057,322.
28

1 **JURISDICTION**

2 1. This action arises under the United States Patent Laws, Title 35 of the
3 United States Code. This Court has subject matter jurisdiction over this action under
4 28 U.S.C. §§ 1331 and 1338(a).

5 2. This Court has personal jurisdiction over Google in this action because
6 Google has committed acts within the Central District of California giving rise to this
7 action and has established minimum contacts with this forum such that the exercise
8 of jurisdiction over Google would not offend traditional notions of fair play and
9 substantial justice. Defendant Google, directly and/or through subsidiaries or
10 intermediaries, has committed and continues to commit acts of infringement in this
11 District by, among other things, using, offering to sell, and selling products and/or
12 services that infringe the Asserted Patents. Google maintains offices and facilities in
13 this District and actively directs its activities to customers located in the State of
14 California.

15 **PARTIES**

16 3. Plaintiff Sandpiper CDN is a Delaware limited liability company with
17 its principal place of business in Wilmington, Delaware.

18 4. Defendant Google LLC is a Delaware limited liability company with
19 its principal place of business at 1600 Amphitheatre Parkway, Mountain View,
20 California 94043. Google maintains a permanent physical presence within the
21 Central District of California, including at 340 Main Street, Los Angeles, California
22 90291; 12422 W. Bluff Creek Drive, Playa Vista, California 90094; and 5865 Campus
23 Center Drive, Playa Vista, California 90094.¹

24 **NATURE OF THE ACTION**

25 5. This is a civil action against Google for patent infringement arising
26 under the United States Patent Laws, 35 U.S.C. § 271, *et. seq.* for the infringement of

27 _____
28 ¹ https://about.google/intl/ALL_us/locations/?region=north-america.

1 United States Patent Nos. 8,478,903; 8,645,517; 9,021,112; 10,924,573; and
2 10,057,322 (collectively “the Asserted Patents”). A true and correct copy of each
3 Asserted Patent is attached to this Complaint as Exhibits A-E, respectively. Each of
4 the Asserted Patents is owned by Plaintiff Sandpiper CDN, and Plaintiff and/or its
5 predecessors-in-interest have satisfied all statutory obligations required to collect pre-
6 and post-filing damages for the full period allowed by law for infringement of the
7 Asserted Patents, including compliance with 35 U.S.C. § 287.

8 **VENUE**

9 6. Venue is proper in this District pursuant to 28 U.S.C. §§ 1391(b) and
10 (c) and/or 1400(b). Defendant Google maintains a regular and established place of
11 business in the Central District of California and has committed and continues to
12 commit acts of patent infringement in the Central District of California.

13 7. Google maintains regular and established places of business in this
14 District, located at 340 Main Street, Los Angeles, California 90291; 12422 W. Bluff
15 Creek Drive, Playa Vista, California 90094; and 5865 Campus Center Drive, Playa
16 Vista, California 90094.

17 8. Google has conducted and conducts business in the State of California,
18 including in this District. Google, either directly or through subsidiaries and/or
19 intermediaries, makes, uses, offers for sale, sells, and/or advertises its infringing
20 products and/or services in the Central District of California. Google further directs
21 and encourages its customers to use its infringing products and/or services in the
22 United States and the Central District of California.

23 9. Google, either directly or through subsidiaries and/or intermediaries,
24 has voluntarily placed one or more of its infringing products and/or services into the
25 stream of commerce with the expectation that those products and/or services will be
26 purchased and used by customers in the Central District of California. These
27 infringing products and/or services have been and continue to be used, sold, offered
28

1 for sale, and/or purchased by customers and/or consumers in the Central District of
2 California.

3 **FACTUAL BACKGROUND**

4 **Content Delivery Networks**

5 10. Today, content delivery networks (“CDN”) provide the critical services
6 that enable content providers to quickly deliver online content to millions of
7 consumers simultaneously over the Internet. But this has not always been the case.

8 11. In the early 1990s, the World Wide Web saw increasing adoption,
9 becoming a household staple. This mass adoption led to data congestion issues due
10 to the ever-growing number of users seeking to simultaneously access Internet
11 content. A typical computer server in the 1990s, for example, could only handle a
12 limited number of simultaneous connections before becoming overloaded. Moreover,
13 signals take time to move through physical internet cables, and consumers living far
14 from the physical server(s) hosting content experienced sluggish load times and high
15 latency due to problems such as overloaded servers, congested network segments, and
16 geographic separation.

17 **Sandpiper Networks**

18 12. In the mid-1990s, Andrew Swart and David Farber were among the
19 first individuals to develop services that allowed content providers to distribute their
20 content over the Internet, while avoiding the common congestion and performance
21 issues that plagued Internet transmission at that time. One solution was to deploy
22 CDN servers around the world, replicate appropriate content from customers’ origin
23 servers to appropriate CDN servers, transparently rendezvous end users requesting
24 that content to the “best” CDN server to deliver that content, while providing their
25 customers with control over their content and user experience. This service and its
26 architecture was quickly imitated by many others in the industry.

1 13. Using solutions developed by Mr. Swart and Mr. Farber, consumers
2 would connect to an edge server that was closer to them and that had available
3 bandwidth. Distributing content across a network of servers alleviated data
4 congestion issues and allowing consumers to connect to edge servers located near
5 them reduced latency. Messrs. Swart and Farber developed and built systems and
6 methods for propagating data from origin servers to edge servers (a process known as
7 “caching”) based on network demand and for seamlessly routing users to the optimal
8 edge server with the correct content.

9 14. In 1996, Mr. Swart and Mr. Farber founded Sandpiper Networks Inc.
10 to further develop and commercialize their novel concept for a content delivery
11 network. Sandpiper Networks was based in Thousand Oaks, California. Beginning in
12 1996, Sandpiper Networks designed and built a CDN referred to as “Footprint.” By
13 at least May 24, 1996, the Sandpiper team had conceived techniques for delivering
14 streaming resources, such as audio and video, using Sandpiper’s CDN.

15 15. Sandpiper Networks labored not only to build and implement its CDN,
16 but also to protect their groundbreaking innovation through the patent system.
17 Recognizing that its invention could revolutionize content delivery worldwide,
18 Sandpiper Networks filed numerous patent applications directed to its foundational
19 CDN technology, including U.S. Patent Application No. 09/021,506 (“the ‘506
20 application”), which was filed on February 10, 1998.

21 16. By at least May 1998, Sandpiper Networks was caching content and
22 delivering cached content to end users of content providers using its CDN. Sandpiper
23 Networks’ first paying customer was the L.A. Times, which paid Sandpiper Networks
24 to host the report of Independent Counsel Ken Starr on his investigation of President
25 Bill Clinton (“the Starr Report”) beginning on September 11, 1998. Sandpiper
26 Networks’s CDN was capable of caching, and used to cache and deliver Internet
27
28

resources including *inter alia*, pictures, text files, dynamic resources, and streaming multimedia resources.

17. By October 30, 1998, Sandpiper Networks had partnered with WebRadio to utilize Sandpiper Networks's CDN to deliver streaming audio from radio stations on behalf of WebRadio. This streaming audio was readily available to any Internet user.

18. On April 19, 1999, Sandpiper Networks used its CDN to broadcast a live concert by the band "Big Bad Voodoo Daddy."

19. In December 1999, Sandpiper Networks merged with Digital Island, Inc. ("Digital Island"), which filed additional patent applications directed to CDN technology. For example, patent application, U.S. Patent Application No. 09/612,598 ("the '598 application"), was filed as a continuation of the '506 application and eventually resulted in the '903 Patent, which issued on July 2, 2013.

20. Following a series of acquisitions, the assets of Sandpiper Networks and Digital Island, as well as their CDNs and patents, were acquired by Level 3 in January 2007.

Level 3 Enters the CDN Market

21. Following its acquisition of Sandpiper Networks's CDN, Level 3 understood the groundbreaking technologies pioneered by Sandpiper Networks. Level 3 continued building upon the foundation laid by Sandpiper, eventually becoming one of the foremost CDN operators in the United States.

22. Level 3 also continued innovating upon the CDN technologies developed by Sandpiper Networks. On March 21, 2009, Level 3 filed U.S. Patent Application No. 12/408,681 ("the '681 application").

23. A continuation of the '681 application issued as the '573 Patent on February 16, 2021. On March 14, 2013, Level 3 filed U.S. Patent Application No.

1 13/828,251 (“the ’251 application”) directed to improving the delivery of content in
2 a CDN. A division of the ’251 application issued as the ’451 Patent on October 8,
3 2018.

4 24. On August 14, 2018, Level 3 filed U.S. Patent Application 16/103,575,
5 which issued as the ’579 Patent on February 16, 2021.

6 25. On April 27, 2020, Level 3 filed U.S. Patent Application No.
7 16/859,918 (“the ’918 application”) directed to improving storage of content in
8 CDNs. The ’918 application issued as the ’520 Patent on October 4, 2022.

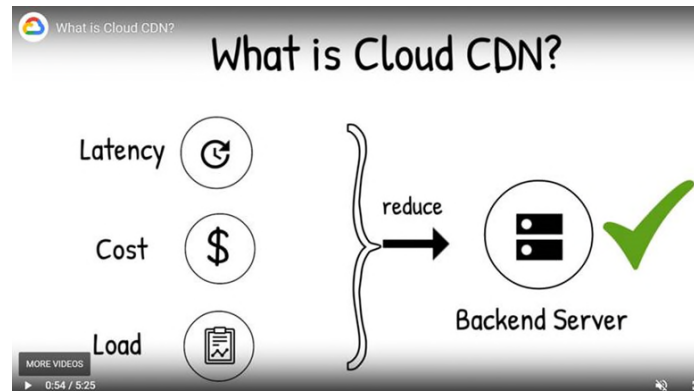
9 **Industry Infringement**

10 26. In the early-to-mid 2000s, demand for CDNs exploded. This increased
11 demand prompted a slew of companies to enter the CDN market. These companies
12 commercialized their own CDNs that incorporated the foundational CDN technology
13 pioneered and patented by Sandpiper Networks and Level 3. In doing so, these
14 companies capitalized on the investment made into CDN research and development
15 made by Level 3 and/or its predecessors, misappropriating years of research and
16 investments.

17 **Google’s Infringing Services**

18 27. One such company is Google. Google built its own in-house CDN,
19 which uses technology described and claimed by the Asserted Patents. For example,
20 YouTube, which Google purchased in 2006, relies upon CDN technology to meet its
21 enormous data streaming needs, as well as Google’s video search results (powered by
22 YouTube). Google uses and provides content delivery network(s) (“Google CDN”),
23 including Google’s internal services and services Google offers to third parties that
24 are used to provide content, such as webpages and/or video streams, over a network.
25 On its website, Google touts the benefits of its infringing CDN services, noting the
26 cost savings and improved performance achieved by using this technology.²

27 _____
28 ² <https://cloud.google.com/cdn>



28. Google CDN is used and sold by Google in connection with services such as Google Cloud CDN, Google Media CDN, Google Cloud DNS, YouTube, and YouTube TV.

29. Google did not license this patented technology from Sandpiper CDN, Level 3 and/or its predecessors. After implementing its in-house CDN using technology described and claimed by the Asserted Patents, Google began directly competing with Level 3 by offering CDN services to third parties.

Sandpiper CDN

30. Given the rampant infringement of its patents, which depressed its revenue and profit, Level 3 decided to exit the CDN market in 2023 and began selling off its CDN assets. Level 3 sold the Asserted Patents to Plaintiff Sandpiper CDN in 2024. As such, Sandpiper CDN is presently the owner in right, title, and interest in and to each of the Asserted Patents.

31. Named after, and in homage to, the company that originally pioneered and developed CDN technologies in the 1990's, Sandpiper CDN now brings this suit to address Google's longstanding infringement of the patented technology claimed by the Asserted Patents. The Asserted Patents are valid and enforceable, and the

1 inventions claimed in the Asserted Patents were novel, non-obvious, unconventional,
2 and non-routine at least as of their respective filing dates.

3 **ASSERTED PATENTS**

4 32. Patent Number 8,478,903 (“the ’903 Patent”) is entitled “Shared
5 Content Delivery Infrastructure,” and it claims priority to U.S. Patent Application No.
6 09/021,506, filed on February 10, 1998.

7 33. One issue related to using CDN technology is delivering resources
8 associated with more than one content provider. The inventors of the ’903 Patent
9 understood that website owners and other content providers want to have their own
10 internet domain shown in end users’ browsers, even when the content providers are
11 using a CDN to serve their content.

12 34. The ’903 Patent is directed to CDN technology and to solving issues
13 related to delivering resources from more than one content provider. In some
14 embodiments, this involves replicating content from a source associated with a client
15 of a CDN network onto CDN servers. End user requests are then directed to the CDN
16 servers instead of to the client’s source servers (generally referred to as “origin”
17 servers), in some cases. Embodiments of the ’903 Patent address issues such as load
18 balancing and reducing traffic to client origin servers. When an origin server receives
19 and must reply to multiple requests, delivery of content from a content provider can
20 be slow. Methods of using CDN servers to deliver content as described in the ’903
21 Patent help to solve this issue.

22 35. The inventors devised the concrete solutions recited in the ’903 Patent
23 to address unique problems related to providing content from various content
24 providers using CDN technology. For instance, the inventors conceived of approaches
25 to content delivery in a network for delivering resources associated with more than
26 one content provider, embodiments of which involve a shared CDN server and alias
27 names in order to provide resources in response to requests. The solutions recited in
28

1 the claims of the '903 Patent include specific combinations that were not conventional
2 at the time of the invention of the '903 Patent. The '903 Patent describes how such
3 solutions are implemented. As one example, the '903 Patent describes using a second
4 alias name in association with a second resource and a second content provider, and
5 a table that can be consulted to determine a content provider associated with a
6 particular resource. Specific innovations by the inventors enable CDN customers to
7 use their own name while using a provider's CDN services to serve content to end
8 users.

9 36. Patent Number 8,645,517 (the '517 Patent) is entitled "Policy-Based
10 Content Delivery Network Selection." The '517 Patent claims priority to U.S. Patent
11 Application Number 10/259,497, filed on September 20, 2002.

12 37. When using CDN technology, challenges arise related to network
13 traffic associated with requests for content. For example, companies that want to
14 provide their content to end users using a CDN face issues regarding directing
15 network traffic to serve end users' requests. As one example, issues arise when trying
16 to deliver content from multiple servers at geographically-separated locations with
17 suitable end-user experiences under high traffic loads.

18 38. The inventors of the '517 Patent understood these challenges faced by
19 CDN providers and their customers relating to directing network traffic and end-user
20 requests for content. The inventors devised solutions utilizing Domain Name Server
21 (DNS) technology and allowing for the resolution of requests to multiple CDNs based
22 on various policies. Embodiments of the disclosed invention solve network traffic
23 issues such as server failure, and controlling the distribution of requests according to
24 economic or contractual parameters, by providing a network distribution
25 infrastructure that can be configured with network traffic rules. These rules can
26 account for factors like server availability, geolocation, load, and latency. The '517
27 Patent address specific needs in the art via specific combinations, which were not
28

1 conventional at the time of the invention of the '517 Patent, and the '517 Patent
2 describes how specific technical solutions are achieved. For instance, content
3 providers must address how to handle requests from end users, in some cases in
4 accordance with policy constraints faced by the content providers, even as network
5 conditions affecting the Internet are subject to change. The inventors of the '517
6 Patent developed methods to address these issues, including methods in which a
7 server network implements policy-based traffic direction. For example, embodiments
8 of the invention in the '517 Patent provide a specific graphical user interface for
9 obtaining at least one policy for the direction of network traffic, such as using a
10 decision tree with resource and branch nodes. The various criteria that can be used
11 may relate to IP addresses or geographic zones, with answers consisting of IP
12 addresses and CNAMEs, for example.

13 39. Patent Number 9,021,112 ("the '112 Patent") is entitled "Content
14 Request Routing and Load Balancing For Content Distribution Networks." The '112
15 Patent is a divisional application of U.S. Patent Application No. 09/982,721, filed on
16 October 18, 2001.

17 40. When offering CDN services, there is a need for responding to requests
18 by providing content quickly and without unnecessary network traffic to more distant
19 sources. Providers of CDN services face challenges relating to serving content in a
20 timely manner from multiple servers while maintaining a positive user experience.

21 41. The inventors of the '112 Patent solved these issues as described in the
22 '112 Patent. For example, the '112 Patent includes solutions using DNS technology
23 and anycast addresses in a CDN where the DNS infrastructure routes the content
24 request to the server that is closest in terms of network distance to the user. The
25 described solutions improve upon anycast DNS. The solutions in the '112 Patent
26 address issues relating to bandwidth and latency, which users may experience as
27 unacceptable delays. The inventors of the '112 Patent recognized the impact on users'

1 experiences due to these problems. The inventors devised methods for delivering
2 content in a network involving using multiple servers to serve requests. For example,
3 the inventors provided methods of causing servers to respond to requests, including
4 by using certain hostnames in association with certain servers to retrieve content for
5 users. The solutions recited in the claims of the '112 Patent include specific
6 combinations that were not conventional at the time of the invention of the '112
7 Patent. The '112 Patent describes how the solutions are implemented. For instance,
8 the claimed solutions in the '112 Patent can improve localization for DNS, where
9 servers as close as possible may be desirable but may provide too many options, while
10 the use of anycast as recited in the '112 Patent can be used to leverage servers closest
11 in the network and to leverage the Border Gateway Protocol and provide for co-
12 location of servers, for example.

13 42. Patent Number 10,924,573 ("the '573 Patent") is entitled "Handling
14 Long-Tail Content in a Content Delivery Network (CDN)," and it claims priority to
15 Provisional Patent Application No. 61/042,412, filed on April 4, 2008.

16 43. CDN technology generally includes multiple servers used to serve
17 content, which can create problems relating to responding to requests for content. For
18 example, technical issues arise in the context of content delivery using a network and
19 multiple servers, relating to handling requests for resources and timely providing
20 resources to requesting devices.

21 44. The inventors of the '573 Patent described solutions to these problems
22 in the '573 Patent. The '573 Patent describes a CDN including a tiered server system.
23 In some embodiments, a first tier server attempts to respond to a user request for
24 content. If the first tier server does not have the content, and the content is popular,
25 that server will request the content from higher tier server and subsequently cache the
26 content for future delivery, for example. The disclosed invention addresses the reality
27 of storage limitations by setting forth a framework that automatically caches only
28

1 popular content, thus both speeding up content delivery and preserving memory space
2 for popular content, in some cases. The inventors devised solutions as shown in the
3 embodiments claimed by the '573 Patent, for example methods for using tiers of
4 servers and specific processes for obtaining content for a requesting device. The
5 solutions recited in the claims of the '573 Patent include particular combinations that
6 were not conventional at the time of the invention of the '573 Patent, such as specific
7 technical improvements to content delivery networks, and the '573 Patent describes
8 how the solutions are implemented.

9 45. Patent Number 10,057,322 ("the '322 Patent") is entitled "Network
10 Address Resolution," and it claims priority to Provisional Patent Application No.
11 62/098,930 ("the '930 Application"), filed on December 31, 2014.

12 46. The '322 Patent is directed to novel techniques for improving the
13 performance of content delivery networks (CDNs), in some cases using an application
14 programming interface (API) "to embed link resolution information within an HTML
15 document or other content returned to a requesting client." See '322 Patent, 1:16-18.
16 Aspects of the '322 Patent "involve an alternative to using conventional domain name
17 system (DNS) servers for edge selection by instead calling an API or otherwise
18 querying the CDN when constructing a uniform resource locator (URL) for the
19 resource to be delivered from the CDN." See *id.* at 3:10-14. The solutions described
20 in the '322 Patent were unconventional at the time of the invention of the '322 Patent
21 and continue to offer numerous technical benefits.

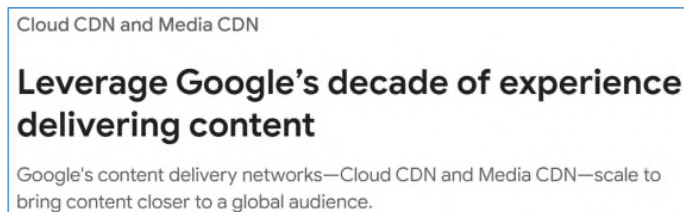
22 47. For example, as users were able to access more content (e.g., television,
23 movies, news, software updates) on networks, the network providers sought
24 alternatives to routing requests through DNS to select an edge server. See *id.* at 1:39-
25 50. The inventors of the '322 Patent devised techniques for bypassing DNS, thereby
26 reducing the number of components involved in servicing requests while improving
27 response speeds while delivering content.

1 48. As stated during prosecution of the application that resulted in the '322
2 Patent, the claims provide “specific, discrete implementation[s] . . . and do[] not
3 preempt all uses of such an idea(s).” See Amendment and Response to Non-Final
4 Office Action dated March 15, 2018, p. 9 (quotes omitted). For example, the inventors
5 developed a novel technique to “providing, at a content delivery network, an
6 application programming interface (API); at the API, receiving, from an origin system
7 that is separate from the content delivery network, an embedded resource and at least
8 one parameter associated with the embedded resource; generating a modified
9 embedded resource providing a direct link to a node within the content delivery
10 network for obtaining content associated with the embedded resource; delivering, by
11 the content delivery network, the modified embedded resource to the origin system;
12 receiving, at the node, a request from a client device to obtain the content; and
13 providing the content to the client device.” See *id.* at claim 15.

14 49. The solutions described and claimed in the '322 Patent improve the
15 functioning of a CDN, including, for example, offloading computational resource
16 consumption away from a client-side DNS infrastructure, and instead having content
17 retrieved from an optimal node instead of another suboptimal node that would cause
18 errors in the network or a longer request service time.

19 **THE ACCUSED GOOGLE CDN FUNCTIONALITIES**

20 50. The accused Google functionalities comprise Google CDN. For
21 example, Google uses and sells CDN services via its Cloud CDN and Media CDN
22 offerings, as shown by the below excerpts from Google’s website.



<https://cloud.google.com/cdn?hl=en>

Cache hits and cache misses

A cache is a group of servers that stores and manages content so that future requests for that content can be served faster. The cached content is a copy of cacheable content that is stored on origin servers.

<https://cloud.google.com/cdn/docs/overview>

IP addressing

Each Edge Cache service that you configure has dedicated, anycast IPv4 and IPv6 addresses, which are associated with each Edge Cache service that you create and are not shared with other customers.

<https://cloud.google.com/media-cdn/docs/client-connectivity>

Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure.

<https://cloud.google.com/media-cdn/docs/overview#certificate-support>

- Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads.
- Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates.

<https://cloud.google.com/media-cdn/docs/origins#origin-requirements>

Advanced routing features

Media CDN provides advanced HTTP routing capabilities that let you map traffic to specific edge configurations and origins at a fine-grained level.

For more information, see [Advanced routing](#).

<https://cloud.google.com/media-cdn/docs/overview#certificate-support>

1
2 51. Google CDN leverages Google’s “Cloud DNS,” which provides a
3 plurality of Domain Name System (DNS) servers (*e.g.*, via providing “anycast name
4 servers”).

5
6 Use Google’s infrastructure for
7 production quality and high-
8 volume authoritative DNS serving.
9 Your users will have reliable, low-
10 latency access from anywhere in
11 the world using our anycast name
12 servers.

Cloud DNS can scale to large
numbers of DNS zones and
records. You can reliably create
and update millions of DNS
records. Our name servers
automatically scale to handle
query volume.

13 <https://cloud.google.com/dns?hl=en>

14 **Fast anycast name servers**

15 Cloud DNS uses our global network of anycast name servers
16 to serve your DNS zones from redundant locations around the
17 world, providing high availability and lower latency for your
18 users.

19 <https://cloud.google.com/dns?hl=en>

20 Google Cloud DNS is a scalable, reliable and managed authoritative Domain
21 Name System (DNS) service running on the same infrastructure as Google. It
22 has low latency, high availability and is a cost-effective way to make your
23 applications and services available to your users.

24 [https://console.cloud.google.com/marketplace/product/google-cloud-
platform/cloud-dns](https://console.cloud.google.com/marketplace/product/google-cloud-platform/cloud-dns)

25 **YouTube**

26 52. Google CDN is also used in connection with Google’s YouTube
27 platform, enabling delivery of YouTube’s video content across the globe for YouTube
28

viewers and advertisers. YouTube provides embedded resources including links to nodes in a CDN for delivering content, for example, such as a link to specific cache server for serving video content and/or advertisements.

YouTube Live is an easy way for Creators to reach their community in real time. Whether streaming an event, teaching a class, or hosting a workshop, YouTube has tools that will help manage live streams and interact with viewers in real time.

Creators can live stream on YouTube via [webcam](#), [mobile](#), and [encoder streaming](#). Webcam

<https://www.youtube.com/howyoutubeworks/product-features/live/#youtube-live>

For Creators to monetize their live stream or Premiere, their channel needs to be part of the [YouTube Partner Program \(YPP\)](#). Creators have several ways to earn money from live streams and Premieres: Ads, Super Chat and Super Stickers, and Channel Memberships.

<https://www.youtube.com/howyoutubeworks/product-features/live/#monetization>

Monetize your live stream

Latest updates

- More flexibility added to the mid-roll ad settings for live streams:
 - 'Let YouTube insert mid-rolls' during natural ad breaks, but choose between 3 frequency options for how often ads surface.

<https://support.google.com/youtube/answer/7385599?sjid=2366353019137992062-NC>

Ad personalization

Google makes your ads more useful on Google services such as Search and YouTube (including portions of YouTube TV). You can control what info we use to show you ads by [visiting your ad settings](#) .

<https://support.google.com/youtubetv/answer/7126139?hl=en#zippy=%2Clive-tv%2Cad-personalization>

Delay a mid-roll ad

If you choose to let YouTube insert mid-roll ads for you, you can delay mid-roll ads during important moments when you don't want viewers interrupted.

In the Live Control Room, go to the top right and click Delay ads . Mid-roll ads are delayed from displaying for viewers for 10 minutes. A 5-second countdown will surface in the LCR for you before ads resume.

<https://support.google.com/youtube/answer/7385599?sjid=2366353019137992062-NC#YTinsertmid>

Channel level settings for mid-roll ads

Monetized channels without rights management can set mid-roll ad defaults for future live streams at the channel level. Channel level settings let you choose monetization settings that will be the default for any newly created live streams.

To choose your channel level settings: Open the Live Control Room > In the bottom-left corner, click Settings .

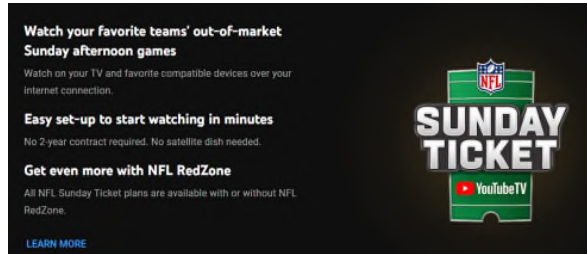
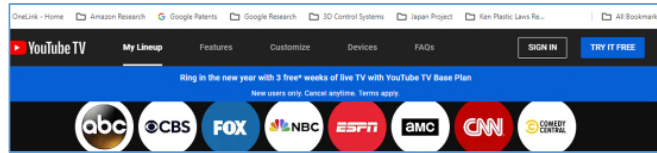
Live stream ad revenue

You can view a breakout of your ad revenue from live streams and live replays in YouTube Analytics. To view your ad revenue breakout for live streams, select the **Live** filter.

https://support.google.com/youtube/answer/7385599?sjid=2366353019137992062-NC#channel_settings_midroll

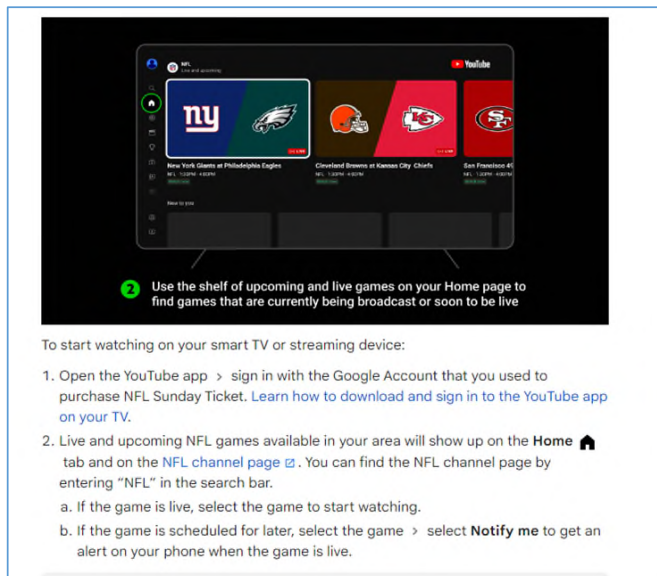
YouTube TV

53. Google CDN also is used to deliver Google's YouTube TV service. YouTube TV provides resources including links for delivering video content such as Ads. To use YouTube TV, users enter credential information to access video streams, and some content requires users to subscribe to access an associated video stream.



<https://tv.youtube.com/welcome/>

54. YouTube TV receives a request from an end user for delivery of a video stream across a network and provides controls for said end user to request delivery of a video stream by, for example, selecting a specific event to stream.



<https://support.google.com/youtube/answer/13821595?hl=en>

snippet.resourceId	object The id object contains information about the channel that the user subscribed to.
snippet.resourceId.kind	string The type of the API resource.
snippet.resourceId.channelId	string The value that YouTube uses to uniquely identify the channel that the user subscribed to.
snippet.channelId	string The ID that YouTube uses to uniquely identify the subscriber's channel. The resource_id object identifies the channel that the user subscribed to.

<https://developers.google.com/youtube/v3/docs/subscriptions>

Google Cloud provides an intelligent, open, and unified [data and AI cloud](#) built from the same underlying architecture that powers Google's most popular, global products, like YouTube, Search, and Maps. Revolutionize customer experiences with operational databases you know and love, in virtually any environment whether in the cloud or on-premises.

<https://cloud.google.com/solutions/databases?hl=en>

COUNT I: INFRINGEMENT OF THE '903 PATENT

55. Plaintiff hereby incorporates by reference each of the allegations in the foregoing paragraphs as though fully set forth herein and further alleges as follows:

56. Google directly infringed at least claim 28 of the '903 Patent by making, using, selling, offering for sale, importing, exporting, suppling, or distributing within the United States its Google CDN offering, both for the distribution of content via its own platforms and to provide CDN services to Google customers.

57. As further set forth in Exhibit F, Google provided and continued to provide a method in a content delivery system operative in a computer network for delivering content to client machines, the computer network comprising a plurality of origin servers, each of said origin servers having resources associated therewith, and the content delivery system comprising at least one shared repeater server operable to replicate resources associated with the plurality of origin servers. Google associates at least one repeater server with a first alias name, wherein requests for a first

1 resource, located on a first origin server, are directed based at least in part on the first
2 alias name. Google also associates the at least one repeater server with a second alias
3 name, wherein requests for a second resource located on a second origin server are
4 directed to the at least one repeater server for delivery of the second resource. Google
5 provides a table listing origin servers having content located thereon, wherein said
6 content is authorized for delivery to client machines via the at least one shared
7 repeater server wherein the origin servers comprise a first and second origin server.
8 Finally the at least one repeater server utilized by Google is constructed and adapted
9 to analyze, using the table and the alias name received with a client request to
10 determine the origin server associated with the requested resource.

11 **COUNT II: INFRINGEMENT OF THE '778 PATENT**

12 **(DISMISSED BY THE COURT'S SEPTEMBER 16, 2024 ORDER)**

13 58. The Court's Order of September 16, 2024 granted Google's motion to
14 dismiss Sandpiper CDN's claim Google infringed U.S. Patent No. 8,595,778 . See
15 Dkt. No. 28 at 6-11. Sandpiper CDN reserves its right to appeal the Court's Order.

16 **COUNT III: INFRINGEMENT OF THE '517 PATENT**

17 59. Plaintiff hereby incorporates by reference each of the allegations in the
18 foregoing paragraphs as though fully set forth herein and further alleges as follows:

19 60. Google directly infringed at least claim 1 of the '517 Patent by making,
20 using, selling, offering for sale, importing, exporting, suppling, or distributing within
21 the United States its Google CDN, both for the distribution of content via its own
22 platforms and to provide CDN services to Google customers. As further set forth in
23 Exhibit G, Google provided and continues to provide a method that is operable in a
24 framework in which an adaptive traffic control name server network implements
25 policy-based traffic direction, the name server network comprising at least one
26 domain name server comprising hardware in combination with software and
27 constructed and adapted to provide adaptive policy-based domain name service.

1 Google further provides a graphical user interface (GUI). Google also uses said GUI
2 to obtain at least one policy for direction of network traffic, wherein the GUI supports
3 the setting of said at least one policy using a decision tree representing rules. Google
4 provides this at least one policy to a name server network, wherein the decision tree
5 comprises one or more resource nodes, and one or more branch nodes, wherein the
6 one or more resource nodes specifies one or more answers to be provided in response
7 to a DNS request, and wherein the one or more branch nodes specify one or more
8 decision criteria to be applies, and wherein the GUI supports the specification of one
9 or more answers for the one or more resource nodes and one or more decision criteria
10 for the one or more branch nodes, wherein the one or more criteria are selected from
11 criteria related to world zones, countries, states, time zones, etc.

12 **COUNT IV: INFRINGEMENT OF THE '886 PATENT**

13 **(DISMISSED BY THE COURT'S SEPTEMBER 16, 2024 ORDER)**

14 61. The Court's Order of September 16, 2024 granted Google's motion to
15 dismiss Sandpiper CDN's claim Google infringed U.S. Patent No. 8,719,886. See
16 Dkt. No. 28 at 12–15. Sandpiper CDN reserves its right to appeal the Court's Order.

17 **COUNT V: INFRINGEMENT OF THE '112 PATENT**

18 62. Plaintiff hereby incorporates by reference each of the allegations in the
19 foregoing paragraphs as though fully set forth herein and further alleges as follows:

20 63. Google directly infringed at least claim 1 of the '112 Patent by making,
21 using, selling, offering for sale, importing, exporting, suppling, or distributing within
22 the United States its Google CDN, both for the distribution of content via its own
23 platforms and to provide CDN services to Google customers. As further set forth in
24 Exhibit H, Google provided and continues to provide a method of content delivery in
25 a network. Google provides a plurality of DNS servers associated with a CDN, said
26 plurality of CDN DNS servers sharing a common anycast address wherein each CDN
27 DNS server is associated with a respective plurality of content servers. Google also
28

1 causes said plurality of CDN DNS servers to be authoritative for a hostname
2 associated with a content provider by causing said common anycast address to be
3 associated with said hostname. In response to a request for content associated with
4 the content provider and issued by a client, said request including at least a hostname,
5 Google causes the hostname to be resolved to a common anycast address by an ISP
6 DNS server and then one of Google's CDN DNS servers resolves the hostname to
7 identify an IP address for use by the client to retrieve the content from a content
8 server.

9 **COUNT VI: INFRINGEMENT OF THE '573 PATENT**

10 64. Plaintiff hereby incorporates by reference each of the allegations in the
11 foregoing paragraphs as though fully set forth herein and further alleges as follows:

12 65. Google directly infringes at least claim 1 of the '573 Patent by making,
13 using, selling, offering for sale, importing, exporting, supplying, or distributing within
14 the United States its Google CDN, both for the distribution of content via its own
15 platforms and to provide CDN services to Google customers. As further set forth in
16 Exhibit I, Google provides a method of content delivery in a content delivery network.
17 Google receives, at a first server of a first tier of servers, a request from a requesting
18 device for a resource available from Google's CDN, Next, Google accesses a
19 popularity service associated with its CDN to determine a popularity designation
20 associated with the requested resource and requests the resource from a second server
21 of the CDN. Google then processes, at a first server of the first tier servers, a redirect
22 instruction from the second server of the CDN to obtain the resource from a content
23 server of the CDN. Google's CDN receives an instruction to not cache a portion of
24 the resource at the first server when that portion of the resource is obtained from the
25 content server of the CDN.

COUNT VII: INFRINGEMENT OF THE '322 PATENT

66. Plaintiff hereby incorporates by reference each of the allegations in the foregoing paragraphs as though fully set forth herein and further alleges as follows:

67. Google directly infringed at least claim 15 of the '322 Patent by making, using, selling, offering for sale, importing, exporting, supplying, or distributing within the United States its Google CDN, both for the distribution of content via its own platforms and to provide CDN services to Google customers. As further set forth in Exhibit J, Google provided and continues to provide a method of content delivery in a network with an application programming interface (API). At the API, Google receives, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource. Google generates a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource, and Google then delivers, with the content delivery network, the modified embedded resource to the origin system. At the node, Google receives a request from a client device to obtain the content and provides the content to the client device.

PRAYER FOR RELIEF

Wherefore, Plaintiff requests entry of judgment in its favor and against Google as follows:

A. Judgment that Google has directly infringed one or more claims of the Asserted Patents;

B. An award of damages to compensate Plaintiff for Google's infringement, including damages pursuant to 35 U.S.C. § 284, as well as prejudgment and post-judgment interest;

C. An award of costs and expenses in this action, including an award of Plaintiff's reasonable attorneys' fees pursuant to 35 U.S.C. § 285;

1 D. A permanent injunction restraining and enjoining Google, and its
2 respective officers, agents, servants, employees, attorneys, and those persons in active
3 concert or participation with Google who receive actual notice of the order by
4 personal service or otherwise, from any further sales or use of their infringing
5 products and/or services and any other infringement of the Asserted Patents;

6 E. A finding that this is an exceptional case and ordering Google to pay
7 Plaintiff's costs of suit and attorneys' fees; and

8 F. Any such other and further relief as the Court may deem just, proper,
9 and equitable under the circumstances.

10 **JURY DEMAND**

11 Plaintiff respectfully demands a trial by jury on all claims and issues so triable.
12
13
14

15 Dated: January 10, 2025

Respectfully submitted,

17 /s/Mayela C. Montenegro-Urch
18

19 Mayela C. Montenegro-Urch
20 Robert H. Reckers
Attorneys for Plaintiff
Sandpiper CDN, LLC
21
22
23
24
25
26
27
28

EXHIBIT A



US008478903B2

(12) **United States Patent**
Farber et al.

(10) **Patent No.:** **US 8,478,903 B2**
(45) **Date of Patent:** **Jul. 2, 2013**

(54) **SHARED CONTENT DELIVERY
INFRASTRUCTURE**

(75) Inventors: **David A. Farber**, Oak View, CA (US);
Richard E. Greer, Red Lodge, MT
(US); **Andrew D. Swart**, Westlake
Village, CA (US); **James A. Balter**,
Santa Barbara, CA (US)

4,726,017 A 2/1988 Krum et al.
4,803,641 A 2/1989 Hardy et al.
4,839,798 A 6/1989 Eguchi et al.
4,847,784 A 7/1989 Clancey
4,920,432 A 4/1990 Eggers et al.
4,922,417 A 5/1990 Churm et al.

(Continued)

FOREIGN PATENT DOCUMENTS

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

CA 2202572 10/1998
EP 0801487 A2 10/1997

(Continued)

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 2052 days.

OTHER PUBLICATIONS

“Cisco DistributedDirector,” Cisco Systems, Inc., White Paper, cited
in Expert Report of Kevin Almeroth with date of 1996 (month
unknown). [9 pages].

(21) Appl. No.: **11/065,412**

(22) Filed: **Feb. 23, 2005**

(Continued)

(65) **Prior Publication Data**

US 2005/0198334 A1 Sep. 8, 2005

Primary Examiner — Douglas Blair

Related U.S. Application Data

(60) Continuation of application No. 09/612,598, filed on
Jul. 7, 2000, now abandoned, which is a division of
application No. 09/021,506, filed on Feb. 10, 1998,
now Pat. No. 6,185,598.

(57) **ABSTRACT**

A system in which a plurality of content providers provide
multiple resources to multiple clients, wherein each content
provider provides at least some resources via one or more
content sources associated with that content provider. A client
requests a resource from a content provider, where the
resources includes references to other resources of the con-
tent provider. Referenced resources are served to requesting
clients from a shared content delivery network (CDN) formed
from a plurality of servers distinct from content sources asso-
ciated with said content providers. The content sources may
be origin servers associated with the respective content pro-
viders. The referenced resources may be video content, audio
content, text, image content, web pages, HTML files, XML
files, files in a markup language, documents, hypertext docu-
ments, data files, and embedded resources.

(51) **Int. Cl.**
G06F 15/16 (2006.01)
G06F 15/173 (2006.01)

(52) **U.S. Cl.**
USPC **709/245**; 709/219; 709/226

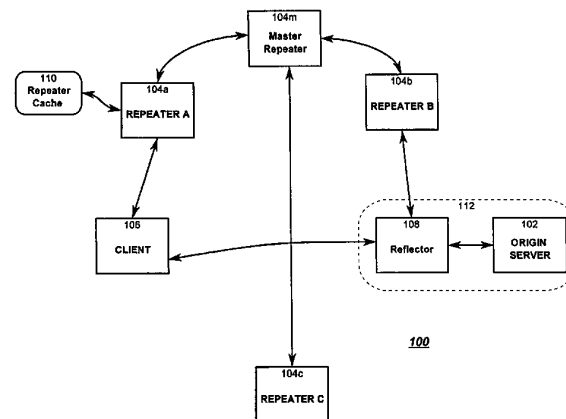
(58) **Field of Classification Search**
USPC 709/217, 219, 245
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,495,570 A 1/1985 Kitajima et al.
4,591,983 A 5/1986 Bennett et al.
4,594,704 A 6/1986 Ollivier

46 Claims, 6 Drawing Sheets



US 8,478,903 B2

Page 2

U.S. PATENT DOCUMENTS			
4,943,932 A	7/1990	Lark et al.	
4,949,187 A	8/1990	Cohen	
4,949,248 A	8/1990	Caro	
5,029,232 A	7/1991	Nall	
5,130,792 A	7/1992	Tindell et al.	
5,132,992 A	7/1992	Yurt et al.	
5,136,716 A	8/1992	Harvey et al.	
5,172,413 A	12/1992	Bradley et al.	
5,191,573 A	3/1993	Hair	
5,253,275 A	10/1993	Yurt et al.	
5,253,341 A	10/1993	Rozmanith et al.	
5,287,499 A	2/1994	Nemes	
5,287,537 A	2/1994	Newmark et al.	
5,291,554 A	3/1994	Morales	
5,341,477 A	8/1994	Pitken et al.	
5,371,532 A	12/1994	Gelman et al.	
5,410,343 A	4/1995	Coddington et al.	
5,414,455 A	5/1995	Hooper et al.	
5,442,389 A	8/1995	Blahut et al.	
5,442,390 A	8/1995	Hooper et al.	
5,442,749 A	8/1995	Northcutt et al.	
5,471,622 A	11/1995	Eadline	
5,475,615 A	12/1995	Lin	
5,508,732 A	4/1996	Bottomley et al.	
5,515,511 A	5/1996	Nguyen et al.	
5,519,435 A	5/1996	Anderson	
5,522,070 A	5/1996	Sumimoto	
5,528,281 A	6/1996	Grady et al.	
5,539,621 A	7/1996	Kikinis	
5,542,087 A	7/1996	Neimat et al.	
5,544,313 A	8/1996	Shachnai et al.	
5,544,327 A	8/1996	Dan et al.	
5,550,577 A	8/1996	Verbiest et al.	
5,550,863 A	8/1996	Yurt et al.	
5,550,982 A	8/1996	Long et al.	
5,557,317 A	9/1996	Nishio et al.	
5,572,643 A	11/1996	Judson	
5,590,288 A	12/1996	Castor et al.	
5,592,611 A	1/1997	Midgely et al.	
5,594,910 A	1/1997	Filepp et al.	
5,603,026 A	2/1997	Demers et al.	
5,619,648 A	4/1997	Canale	
5,623,656 A	4/1997	Lyons	
5,625,781 A	4/1997	Cline et al.	
5,627,829 A	5/1997	Gleeson et al.	
5,630,067 A	5/1997	Kindell et al.	
5,633,999 A	5/1997	Clowes et al.	
5,634,006 A	5/1997	Baugher et al.	
5,638,443 A	6/1997	Stefik et al.	
5,644,714 A	7/1997	Kikinis	
5,646,676 A	7/1997	Dewkett et al.	
5,649,186 A	7/1997	Ferguson	
5,659,729 A	8/1997	Nielsen	
5,666,362 A	9/1997	Chen et al.	
5,671,279 A	9/1997	Elgamal	
5,675,734 A	10/1997	Hair	
5,682,512 A	10/1997	Tetrick	
5,699,513 A	12/1997	Feigen et al.	
5,708,780 A	1/1998	Levergood et al.	
5,712,979 A	1/1998	Graber et al.	
5,715,453 A	2/1998	Stuart	
5,721,914 A	2/1998	DeVries	
5,734,831 A	3/1998	Sanders	
5,740,423 A	4/1998	Logan et al.	
5,742,762 A	4/1998	Scholl et al.	
5,751,961 A *	5/1998	Smyk	709/217
5,761,507 A	6/1998	Govett	
5,761,663 A	6/1998	Lagarde et al.	
5,764,906 A	6/1998	Edelstein et al.	
5,774,660 A	6/1998	Brendel et al.	
5,774,668 A	6/1998	Choquier et al.	
5,777,988 A	7/1998	Cisneros	
5,777,989 A	7/1998	McGarvey	
5,778,187 A	7/1998	Monteiro et al.	
5,784,058 A	7/1998	LaStrange et al.	
5,794,253 A *	8/1998	Norin et al.	1/1
5,796,952 A	8/1998	Davis et al.	
5,799,141 A	8/1998	Galipeau et al.	
5,802,106 A	9/1998	Packer	
5,802,291 A	9/1998	Balick et al.	
5,812,769 A	9/1998	Graber et al.	
5,815,664 A	9/1998	Asano	
5,819,092 A *	10/1998	Ferguson et al.	717/113
5,826,031 A	10/1998	Nielsen	
5,828,847 A	10/1998	Gehr et al.	
5,832,506 A	11/1998	Kuzma	
5,832,514 A	11/1998	Norin et al.	
5,835,718 A	11/1998	Blewett	
5,838,906 A	11/1998	Doyle et al.	
5,845,303 A	12/1998	Templeman	
5,856,974 A	1/1999	Gervais et al.	
5,862,339 A	1/1999	Bonnaure et al.	
5,867,706 A *	2/1999	Martin et al.	718/105
5,867,799 A	2/1999	Lang et al.	
5,870,546 A	2/1999	Kirsch	
5,870,559 A	2/1999	Leshem et al.	
5,878,212 A	3/1999	Civanlar et al.	
5,884,038 A	3/1999	Kapoor	
5,890,171 A	3/1999	Blumer et al.	
5,893,116 A	4/1999	Simmonds et al.	
5,894,554 A	4/1999	Lowery et al.	
5,896,533 A *	4/1999	Ramos et al.	709/217
5,903,723 A	5/1999	Beck et al.	
5,907,704 A	5/1999	Gudmundson et al.	
5,913,028 A	6/1999	Wang et al.	
5,913,033 A	6/1999	Grout	
5,918,010 A	6/1999	Appleman et al.	
5,919,247 A	7/1999	Van Hoff et al.	
5,920,701 A	7/1999	Miller et al.	
5,925,106 A *	7/1999	Nielsen	709/247
5,931,904 A	8/1999	Banga	
5,933,832 A	8/1999	Suzuoka et al.	
5,935,207 A	8/1999	Logue et al.	
5,940,831 A *	8/1999	Takano	1/1
5,944,780 A	8/1999	Chase	
5,945,989 A	8/1999	Freisztat et al.	
5,956,489 A	9/1999	San Andres et al.	
5,956,716 A	9/1999	Kenner et al.	
5,958,008 A	9/1999	Pogrebisky et al.	
5,961,596 A	10/1999	Takubo et al.	
5,966,440 A	10/1999	Hair	
5,968,121 A	10/1999	Logan et al.	
5,973,696 A	10/1999	Agranat et al.	
5,978,791 A *	11/1999	Farber et al.	707/640
5,983,005 A	11/1999	Monteiro et al.	
5,983,214 A	11/1999	Lang et al.	
5,983,227 A	11/1999	Nazem et al.	
5,987,606 A	11/1999	Cirasole et al.	
5,991,809 A	11/1999	Kriegsman	
5,996,025 A	11/1999	Day	
6,002,720 A	12/1999	Yurt et al.	
6,003,030 A	12/1999	Kenner et al.	
6,006,264 A	12/1999	Colby et al.	
6,012,090 A *	1/2000	Chung et al.	709/219
6,014,686 A	1/2000	Elnozahy et al.	
6,014,698 A	1/2000	Griffiths	
6,018,516 A	1/2000	Packer	
6,021,426 A	2/2000	Douglis	
6,026,440 A	2/2000	Shrader et al.	
6,029,175 A	2/2000	Chow et al.	
6,029,176 A	2/2000	Cannon	
6,035,332 A	3/2000	Ingrassia, Jr. et al.	
6,038,216 A	3/2000	Packer	
6,038,310 A	3/2000	Hollywood et al.	
6,038,610 A	3/2000	Belfiore et al.	
6,041,307 A	3/2000	Ahuja et al.	
6,041,324 A	3/2000	Earl et al.	
6,044,405 A	3/2000	Driscoll, III et al.	
6,046,980 A	4/2000	Packer	
6,049,831 A	4/2000	Gardell et al.	
6,052,718 A *	4/2000	Gifford	709/219
6,052,730 A	4/2000	Felciano et al.	
6,061,738 A *	5/2000	Osaku et al.	709/245
6,065,051 A	5/2000	Steele et al.	
6,065,062 A	5/2000	Periasamy et al.	
6,070,191 A	5/2000	Narendran et al.	
6,078,943 A *	6/2000	Yu	718/105

Page 3

FOREIGN PATENT DOCUMENTS

US 8,478,903 B2

Page 4

JP	10-027148	1/1998
WO	WO-96/42041	12/1996
WO	WO-97/11429	3/1997
WO	WO-97/29423	8/1997
WO	WO-98/04985	2/1998
WO	WO-98/06033	2/1998

OTHER PUBLICATIONS

Almeroth, K., et al. "Scalable Delivery of Web Pages Using Cyclic Best-Effort (UDP) Multicast", IEEE Infocom, San Francisco, California, USA, Jun. 1998.

Almeroth, K., Exhibit A to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008, [31 pages].

Almeroth, K., Exhibit B to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [131 pages].

Almeroth, K., Exhibit C to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [157 pages].

Almeroth, K., Exhibit D to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [128 pages].

Almeroth, K., Exhibit E Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [286 pages].

Almeroth, K., Exhibit F to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [221 pages].

Almeroth, K., Exhibit G to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [369 pages].

Almeroth, K., Exhibit H to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [59 pages].

Almeroth, K., Exhibit I to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [95 pages].

Almeroth, K., Exhibit J to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008, [105 pages].

Almeroth, K., Exhibit K-1 to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008, [152 pages].

Almeroth, K., Exhibit K-2 to Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008 [25 pages].

Almeroth, K., Expert Report of Dr. Kevin C. Almeroth filed in Civil Action No. 2:07cv589 (RGD-FBS), U.S. Dist. Ct. E.D. Va., dated Jul. 25, 2008. [64 pages and Exhibits A-J, K1, K2].

Baentsch, M., et al., "Introducing Application-Level Replication and Naming into Today's Web," Computer Networks and ISDN Systems, vol. 28, No. 7-11, pp. 921-930, May 1996.

Berners-Lee, T. & Connolly, D., "Hypertext Markup Language—2.0," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1866, Nov. 1995.

Berners-Lee, T., et al., "Hypertext Transfer Protocol—HTTP/1.0," Internet Engineering Task Force (IETF) Internet Draft, draft-ietf-http-v10-spec-00.ps, Mar. 1995.

Berners-Lee, T., et al., "Hypertext Transfer Protocol—HTTP/1.0," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1945, May 1996.

Berners-Lee, T., "Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1630, Jun. 1994.

Bestavros, A. Demand-based document dissemination to reduce traffic and balance load in distributed information systems. In Proc. IEEE Symp. on Parallel and Distributed Processing, San Antonio, TX, Oct. 1995.

Callon, R., "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1195, Dec. 1990.

Chandra, R. et al., "BGP Communities Attribute," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1997, Aug. 1996.

Chen, E., et al., "An Application of the BGP Community Attribute in Multi-home Routing," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1998, Aug. 1996.

Colajanni, M., et al., "Scheduling Algorithms for Distributed Web Servers," International Conf. on Distributed Computing Systems (ICDCS), Baltimore, Maryland, USA, May 1997.

Communication from European Patent Office dated May 30, 2008 in patent application EP 99906680 (12 pages).

Eriksson, H., "MBONE: The Multicast Backbone," Communications of the ACM, vol. 37, No. 8, p. 54-60, Aug. 1994.

Fielding, R., et al., "Hypertext Transfer Protocol—HTTP/1.1," Internet Engineering Task Force (IETF) Internet Draft, draft-ietf-http-v11-spec-00.txt, Nov. 1995.

Fielding, R., et al., "Hypertext Transfer Protocol—HTTP/1.1," Internet Engineering Task Force (IETF) Request for Comments (RFC) 2068, Jan. 1997.

Malkin, G., "RIP Version 2 Carrying Additional Information," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1388, Jan. 1993.

Moy, J., "OSPF Version 2," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1583, Mar. 1994.

Parris C., et al., "A Dynamic Connection Management Scheme for Guaranteed Performance Services in Packet-Switching Integrated Services Networks," UC Berkeley Computer Science Division Tech. Report TR-93-005, 1993.

Parris C., et al., "The Dynamic Management of Guaranteed Performance Connections in Packet Switched Integrated Service Networks," UC Berkeley Computer Science Division and International Computer Science Institute Tech. Report CSD-94-859, 1994.

Partridge, C., et al., "Host Anycasting Service," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1546, Nov. 1993.

Povey, D. et al., "A distributed internet cache," in Proc. of the 20th Australasian Computer Science Conf., Feb. 1997.

Rekhter Y., et al., "A Border Gateway Protocol 4 (BGP-4)," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1771, Mar. 1995.

Rekhter Y., et al., "Application of the Border Gateway Protocol in the Internet," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1772, Mar. 1995.

Topolcic, C., "Experimental Internet Stream Protocol, Version 2 (ST-II)," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1190, Oct. 1990.

Touch, J. et al., "The X-Bone," Third Global Internet Mini-Conference at Globecom '98, Sydney, Australia, Nov. 1998, pp. 59-68 (pp. 44-52 of the mini-conference).

Traina, P., "BGP-4 Protocol Analysis," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1774, Mar. 1995.

Traina, P., "Experience with the BGP-4 protocol," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1773, Mar. 1995.

Wessels, D. et al., RFC 2187, "Application of Internet Cache Protocol (ICP), Version 2", Sep. 1997. Memo; Internet RFC/STD/FYI/BCP Archives; Network Working Group.

Complaint for Injunctive and Other Relief, filed Dec. 17, 2007 in Civil Action No. 2:07CV589.

Report on the Filing or Determination of an Action Regarding a Patent or Trademark, dated Dec. 28, 2007, in connection with Civil Action No. 2:07CV589 filed Dec. 17, 2007.

Rule 7.1 Financial Interest Disclosure Statement filed Dec. 17, 2007 in Civil Action No. 2:07CV589.

"Cache Director System", <http://www.cisco.com/warp/public/751/cache/cdsov.htm>, Cisco Systems, Inc., (1997), 1-5.

"Cache Director Systems Technology Overview", http://www.cisco.com/warp/public/751/cache/cds_ds.htm, Cisco System, Inc., (1997), 1-3.

"Cisco Cache Engine", <http://www.cisco.com/warp/public/751/cache/index.html>, Cisco Systems, Inc.

US 8,478,903 B2

Page 5

- "Cisco Introduces Web Cache Product for Scaling the Internet", <http://www.cisco.com/warp/public/146/1947.html>, Cisco Systems, Inc., (1997).
- "Cisco Systems Ships Cisco LocalDirector to Meet Demands of High-volume Web Traffic", <http://www.cisco.com/wrap/public/146/323.html>, Cisco Systems, Inc., (1996), 1-2.
- "Cisco Take Global Route", *PC Week News*, (Feb. 17, 1997), 23.
- "Communication (1 page) and European Search Report for", Application No. EP 00 12 8346 (3 pages) Aug. 24, 2001.
- "Excite Chooses Cisco Local Director to Support its Growing Service", <https://www.cisco.com/wrap/public/146/1857.html> Cisco Systems, Inc., (1997), 1-2.
- "Global IP/PX Service Should Keep Network Delays Down", *Infoworld*, (Jan. 20, 1997), 1 page.
- "How to Cost-Effectively Scale Web Servers", <http://www.cisco.com/wrap/public/784/5.html>, Cisco Systems, Inc., (1996), 1-5.
- "http://www.cisco.com/warp/public/751/distdir/dd_wp.htm" Cisco Systems Inc., (1997), 1-15.
- "http://www.cisco.com/wrap/public/751/lodir/lodir_ds.htm", Cisco System Inc., (1997), 1-4.
- "International Search Report for PCT/US99/01477", (Jun. 8, 1999).
- "Local Area Network Server Replacement Procedure", *IBM Technical Disclosure Bulletin*, vol. 38, No. 1, (Jan. 1995), 235-236.
- "Local Director", <http://www.cisco.com/wrap/public/751/lodir/index.html>, 2 pages.
- "Local Director", Cisco Systems, Inc., 1-3.
- "Patent Abstracts of Japan, Electronic Mail Multiplexing System and Communication Control Method in the System", JP 05162529, (Jun. 30, 1993).
- "Patent Abstracts of Japan, Method and Device for Repeating and Converting Information", JP 08328583, (Sep. 1996).
- "Scaling the Internet News Service", http://www.cisco.com/wrap/public/751/lodir/news_wp.htm, Cisco Systems, Inc., (1997), 1-11.
- "Scaling the World Wide Web", http://www.cisco.com/wrap/public/751/lodir/www_wp.htm, Cisco Systems, Inc., (1996), 1-7.
- Adler, "Distributed Coordination Models for Client/Server Computing", *Computer*, vol. 28, No. 4, (Apr. 1, 1995), 14-22.
- Andresen, et al., "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", *Proceedings of IPPS*, Apr. 15, 1996, 850-856.
- Bestavros, Azer, "Speculative Data Dissemination and Service to Reduce Server Load, Network Traffic and Service Time in Distributed Information Systems", *In Proceedings of ICDE '96: The 1996 International Conference on Data Engineering*, (Mar. 1996), 4 pages.
- Carter, J. Lawrence, et al., "Universal Classes of Hash Functions", *Journal of Computer and System Sciences*, vol. 18, No. 2, (Apr. 1979), 143-154.
- Chankhunthod, Anawat, et al., "A Hierarchical Internet Object Cache", *USENIX Proceedings*, (Jan. 1996), 153-163.
- Cormen, Thomas H., et al., "Introduction Algorithms", *The MIT Press*, Cambridge, Massachusetts, (1994), 219-243, 991-993.
- De Bra, P.M.E., et al., "Information Retrieval in the World Wide Web: Making client-based searching feasible", *Computer Networks and ISDN Systems, NL, North Hollan Publishing, Amsterdam*, vol. 27, No. 2, ISSN: 0169-7552, (Nov. 1, 1994), 183-192.
- Deering, Stephen, et al., "Multicast Routing in Datagram Internetworks and Extended LANs", *ACM Transactions on Computer Systems*, vol. 8, No. 2, (May 1990), 85-110.
- Devine, Robert, "Design and Implementation of DDH: A Distributed Dynamic Hashing Algorithm", *In Proceedings of 4th International Conference on Foundations of Data Organizations and Algorithms*, (1993), 101-114.
- Feeley, Michael, et al., "Implementing Global Memory Management in a Workstation Cluster", *In Proceedings of the 15th ACM Symposium on Operating Systems Principles*, (1995), 201-212.
- Floyd, Salley, et al., "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing", *In Proceeding of ACM SIGCOMM '95*, 342-356.
- Fredman, Michael, et al., "Storing a Sparse Table with $O(1)$ Worst Case Access Time", *Journal of the Association for Computing Machinery*, vol. 31, No. 3 (Jul. 1984), 538-544.
- Grigni, Michelangelo, et al., "Tight Bounds on Minimum Broadcasts Networks", *SIAM Journal of Discrete Mathematics*, vol. 4, No. 2, (May 1991), 207-222.
- Gwertzman, James, et al., "The Case for Geographical Push-Caching", *Technical Report HU TR 34-94 (excerpt)*, Harvard University, DAS, Cambridge, MA 02138, (1994), 2 pages.
- Gwertzman, J.S., et al., "The case for Geographical push-caching", *Workshop on Hot Topics in Operating Systems*, (May 4, 1995), 51-55.
- Gwertzman, James, et al., "World-Wide Web Cache Consistency", *In Proceedings of the 1996 USENIX Technical Conference*, (Jan. 1996), 8 pages.
- Karger, David, et al., "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web", *In Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, (May 1997), 654-663.
- Litwin, Wilthold, et al., "LH-A Scalable, Distributed Data Structure", *ACM Transactions on Database Systems*, vol. 21, No. 4, (Dec. 1996), 480-525.
- Malpani, Radhika, et al., "Making World Wide Web Caching Servers Cooperate", *In Proceedings of World Wide Web Conference*, (1996), 6 pages.
- Mourad, et al., "Scalable Web Server Architectures", *Proceedings IEEE Symposium on Computers and Communications*, (Jul. 1, 1997), 12-16.
- Naor, Moni, et al., "The Load, Capacity and Availability of Quorum Systems", *In Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, (Nov. 1994), 214-225.
- Nisan, Noam, "Pseudorandom Generations for Space-Bounded Computation", *In Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, (May 1990), 204-212.
- Palmer, Mark, et al., "Fido: A Cache that Learns to Fetch", *In Proceedings of the 17th International Conference on Very Large Data Bases*, (Sep. 1991), 255-264.
- Panigrahy, Rina, "Relieving Hot Spots on the World Wide Web", *Massachusetts Institute of Technology*, (Jun. 1997), 1-66.
- Peleg, David, et al., "The Availability of Quorum Systes", *Information and Computation*, 123, (1995), 210-223.
- Plaxton, C. Greg, et al., "Fast Fault-Tolerant Concurrent Access to Shared Objects", *In Proceedings of the 37th IEEE Symposium of Foundations of Computer Science*, (1996), 570-579.
- Rabin, Michael, "Efficient Dispersal of Information for Security, Load Balancing and Fault Tolerance", *Journal of the ACM*, vol. 36, No. 2, (Apr. 1989), 335-348.
- Ravi, R., "Rapid Rumor Ramification: Approximating the Minimum Broadcast Time", *In Proceedings of the 35th IEEE Symposium on Foundation of Computer Science*, (Nov. 1994), 202-213.
- Schmidt, Jeanette, et al., "Chernoff-Hoeffding Bounds for Applications with Limited Independence", *In Proceedings of the 4th ACS-SIAM Symposium on Discrete Algorithms*, (1993), 331-340.
- Tarjan, Robert E., et al., "Storing a Sparse Table", *Communications of the ACM*, vol. 22, No. 11, (Nov. 1979), 606-611.
- Vitter, Jeffrey S., et al., "Optimal Prefetching via Data Compression", *In Proceedings of 32nd IEEE Symposium on Foundations of Computer Science*, (Nov. 1991), 121-130.
- Wegman, Mark, et al., "New Hash Functions and Their Use in Authentication and Set Equality", *Journal of Computer and System Sciences*, vol. 22, (Jun. 1981), 265-279.
- Yao, Andrew C., "Should Tables be Sorted?", *Journal of Association for Computing Machinery*, vol. 28, No. 3, (Jul. 1981), 615-628.
- Awerbuch, B. et al., Distributed Paging for General Networks. In Proc. of the 7th ACM-SIAM Symposium on Discrete Algorithms, pp. 574-583, Jan. 1996.
- Baentsch, M. et al. "Enhancing the Web's Infrastructure: From Caching to Replication." *IEEE Internet Computing*, 1(2):Mar. 18-27, 1997.
- Carter, J. Lawrence et al., "Universal Classes of Hash Functions", *Journal of Computer and System Sciences*, vol. 18, No. 2, (Apr. 1979), 143-154.
- Cisco Systems, Inc., "Cache Director System", <http://www.cisco.com/warp/public/751/cache/cdsov.htm>, (1997), 1-5.
- Cisco Systems, Inc., "Cisco Cache Engine", <http://www.cisco.com/warp/public/751/cache/index.html>.

US 8,478,903 B2

Page 6

- Gadde, S., et al., "Reduce, reuse, recycle: An approach to building large Internet caches," in Workshop on Hot Topics in Operating Systems (HotOS), Apr. 1997.
- Naor, Moni, et al., "The Load, Capacity and Availability of Quorum Systems", In Proceedings of the 35th IEEE Symposium on Foundations of Computer Science, (Nov. 1994), 214-225.
- Pulgar, Giray, "USENIX, Automatically Selecting a Close Mirror Based on Network Topology," Proceedings of the Twelfth Systems Administration Conference; Boston, Massachusetts, Dec. 1998, p. 159-166.
- Ravi, R., "Rapid Rumor Ramification: Approximating the Minimum Broadcast Time", In Proceedings of the 35th IEEE Symposium on Foundations of Computer Science, (Nov. 1994), 202-213.
- Schmidt, J., et al., "Chernoff-Hoeffding Bounds for Applications with Limited Independence", in Proc. 4th ACM-SIAM Symp. on Discrete Algorithms, (1993), 331-340.
- Wegman, Mark, et al., "New Hash Functions and Their Use in Authentication and Set Equality", Journal of Computer and System Sciences, vol. 22, (Jun. 1981), 265-279.
- Korkea-aho, M. (1995). Scalability in Distributed Multimedia Systems, Technical report TKO-B128, Helsinki University of Technology.
- "Cache Director System", <http://www.cisco.com/warp/public/751/cache/cdsover.htm> Cisco Systems, Inc., (1997), 1-5.
- "Cisco Introduces Web Cache Product for Scaling the Internet", <http://www.cisco.com/warp/public/146/1947.html> Cisco Systems, Inc., (1997).
- "Cisco Systems Ships Cisco LocalDirector to Meet Demands of High-volume Web Traffic", <http://www.cisco.com/wrap/public/146/323.html>, Cisco Systems, Inc., (1996), 1-2.
- "Communication (1 page) and European Search Report for ", Application No. EP 00 12 8346 (3 pages), (Aug. 24, 2001).
- "Excite Chooses Cisco Local Director to Support its Growing Service", <http://www.cisco.com/wrap/public/146/1857.html>, Cisco Systems, Inc., (1997), 1-2.
- "http://www.cisco.com/warp/public/751/distdir/dd_wp.htm", Cisco Systems Inc., (1997), 1-15.
- "International Search Report for PCT/US99/01477", (Jun. 8, 1999).
- "Local Director", Cisco Systems Inc., 1-3.
- "Patent Abstracts of Japan, Electronic Mail Multiplexing System and Communication Control Method in the System", JP 05162529, Jun. 30, 1993.
- "Scaling the Internet News Service", http://www.cisco.com/warp/public/751/ldir/news_wp.htm, Cisco Systems Inc., (1997), 1-11.
- Adler, "Distributed Coordination Models for Client/Server Computing", *Comptuer*, vol. 28, No. 4, Apr. 1, 1995, 14-22.
- Andresen, et al., "SWEb: Towards a Scalable World Wide Web Server on Multicomputers", Proceedings of IPPS, (Apr. 15, 1996), 850-856.
- Bestavros, Azer, "Speculative Data Dissemination and Service to Reduce Server Load, Network Traffic and Service Time in Distributed Information Systems", In Proceedings of ICDE '96: The 1996 International Conference Conference on Data Engineering, (Mar. 1996), 4 pages.
- Carter, J. Lawrence, et al., "Universal Classes of Hash Functions", *Journal of Computer and System Sciences*, vol. 18, No. 2, (Apr. 1979), 143-154.
- Cormen, Thomas H., et al., "Introduction to Algorithms", *The MIT Press*, Cambridge, Massachusetts, (1994), 219-243, 991-993.
- De Bra, P.M.E., et al., "Information Retrieval in the World Wide Web: Making client-based searching feasible", *Computer Networks and ISDN Systems, NL, North Holland Publishing, Amsterdam*, vol. 27, No. 2, ISSN: 0169-7552, (Nov. 1, 1994), 183-192.
- Floyd, Salley, et al., "A Reliable Multicast Framework for Lightweight Sessions and Application Level Framing", In Proceedings of ACM SIGCOMM '95, 342-356.
- Fredman, Michael, et al., "Storing a Sparse Table with $O(1)$ Worst Case Access Time", *Journal of the Association for Computing Machinery*, vol. 31, No. 3, (Jul. 1984), 538-544.
- Karger, David, et al., "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web", In Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, (May 1997), 654-663.
- Litwin, Wlthold, et al., "LH-A Scaleable, Distributed Data Structure", *ACM Transactions on Database Systems*, vol. 21, No. 4, (Dec. 1996), 480-525.
- Nisan, Noam, "Pseudorandom Generations for Space-Bounded Computation", In Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, (May 1990), 204-212.
- Panigrahy, Rina, "Relieving Hot Spots on the World Wide Web", *Massachusetts Institute of Technology*, (Jun. 1997), 1-66.
- Peleg, David, et al., "The Availability of Quorum Systems", *Information and Computation*, 123, (1995), 210-223.
- Ravi, R., "Rapid Rumor Ramification: Approximating the Minimum Broadcast Time", In Proceedings of the 35th IEEE Symposium on Foundations of Computer Science, (Nov. 1994), 202-213.
- Tarjan, Robert E., et al., "Storing a Sparse Table", *Communications of the ACM*, vol. 22, No. 11, (Nov. 1979), 606-611.
- "Exporting Web Server Final Report," http://www.cs.technion.ac.il/Labs/Lccn/projects/spring97/project4/final_report.html, Spring 1997 (downloaded Jul. 7, 2007).
- Andresen, D., et al., Multiprocessor scheduling with client resources to improve the response time of WWW applications, Proc. 11th Int'l Conf. on Supercomputing (Austria, Jul. 1997). ICS '97. ACM Press, NY, NY, 92-99.
- Basturk, E., et al., "Using network layer anycast for load distribution in the Internet," Tech. Rep., IBM T.J. Watson Research Center, 1997 (21 pgs.).
- Berners-Lee et al., RFC 1738—Uniform Resource Locators, Dec. 1994.
- Bestavros, et al., "Server-Initiated Document Dissemination for the WWW," IEEE Data Engineering Bulletin 19(3):Sep. 3-11, 1996.
- Bhattacharjee et al., "Application-layer anycasting," in Proc. IEEE INFOCOM '97, 1997.
- Braun, H., et al., "Web traffic characterization: an assessment of the impact of caching documents from NCSA's web server," Comput. Netw. ISDN Syst. 28, 1-2 (Dec. 1995), 37-51.
- Brisco, T. P. RFC 1794: DNS support for load balancing, Apr. 1995.
- Carter et al., "Dynamic server selection using bandwidth probing in wide-area networks," Tech. Rep. BU-CS-96-007, Comp. Sci. Dept., Boston University, Mar. 1996.
- Carter et al., Server selection using dynamic path characterization in Wide-Area Networks, IEEE INFOCOM '97, 1997 (pp. 1014-1021).
- Cisco Systems, Inc., Cisco DistributedDirector 2500 Series Installation and Configuration Guide, pp. xix-xxii; 1-1 to 1-12; 6-1 to 6-18; 7-1 to 7-18; 8-1 to 8-24, pub. date unknown, downloaded Apr. 2007: <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distdir/dd2501/>.
- Cisco Systems, Inc., Cisco DistributedDirector 4700-M Installation and Configuration Guide, pp. xix-xxii; 1-1 to 1-14; 7-1 to 7-18, 8-1 to 8-20; pub. date unknown, [downloaded Apr. 2007 from <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distdir/dd4700m/>].
- Cohen, J., et al., "Cache Array Routing Protocol v1.1", Sep. 29, 1997; <http://tools.ietf.org/id/draft-vinod-carp-v1-01.txt> (Last-Modified: Wed, Oct. 1, 1997).
- Colajanni, M. and Yu, P. S. 1997. Adaptive TTL schemes for load balancing of distributed Web servers. SIGMETRICS Perform. Eval. Rev. 25, 2 (Sep. 1997), 36-42.
- Crovella et al., Dynamic server selection in the Internet, 3rd IEEE Workshop on the Arch. and Implementation of High Performance Computer Sys. '95, pp. 158-162, Aug. 1995.
- Danzig, P. B., et al., "An analysis of wide-area name server traffic: a study of the Internet Domain Name System," Conf. Proc. Communications Architectures & Protocols (Aug. 1992). D. Oran, Ed. SIGCOMM '92. ACM Press, New York, NY, 281-292.
- Doi, K. "Super Proxy Script—How to make distributed proxy servers by URL hashing," Sharp Corp., <http://naragw.sharp.co.jp/sps/>, dates unknown (1996-2000), download Jul. 7, 2007.
- Fox, A., "A Framework for Separating Server Scalability and Availability from Internet Application Functionality," PhD thesis, University of California, Berkeley, 1998.
- Fox, A., et al "Cluster-based scalable network services", Proc. 16th ACM Symp. on Operating Systems Principles (Saint Malo, France, Oct. 5-8, 1997). W. M. Waite, Ed. SOSP '97. ACM Press, New York, NY, 78-91.

US 8,478,903 B2

Page 7

- Goldszmidt, et al., "Load Distribution for Scalable Web Servers: Summer Olympics 1996—A Case Study," In Proc. 8th IFIP/IEEE Int'l Workshop on Distributed Systems: Operations and Management, Sydney, Australia, Oct. 1997.
- Gulbrandsen, A., et al., "A DNS RR for specifying the location of services (DNS SRV)," Network Working Group, RFC 2052, Oct. 1996.
- Guyton et al., "Locating nearby copies of replicated Internet servers," Proc. ACM SIGCOMM '95, pp. 288-298, 1995.
- Jeffrey et al., Proxy-Sharing Proxy Servers, IEEE, pp. 116-119, 1996.
- Kwan et al., NCSA's World Wide Web Server: Design and Performance, IEEE, pp. 68-74, Nov. 1995.
- Luotonen et al., World-Wide Web Proxies, CERN, Apr. 1994 (modified May 24, 1994).
- Mockapetris et al., "Development of the Domain Name System," Proc. SIGCOMM '88 Computer Communications Review, vol. 18, No. 4, Aug. 1988.
- Mockapetris, P., RFC 1034: Domain Names—Concepts and Facilities, Nov. 1987.
- Mockapetris, P., RFC 1035: Domain Names—Implementation and Specification, Nov. 1987.
- Oguchi et al., A Study of Caching Proxy Mechanisms Realized on Wide Area Distributed Networks, High Performance Distributed Computing, 5th Int'l Symposium, pp. 443-449, 1996.
- Peleg D., et al., "The Availability of Quorum Systems," Tech. Rep. CS93-17, The Weizmann Institute of Science, Rehovot, Israel, 1993.
- Petri S., et al., "Load Balancing and Fault Tolerance in Workstation Clusters. Migrating Groups of Communicating Processes," Operating Systems Review, vol. 29, No. 4, Oct. 1995, pp. 25-36.
- Postel, RFC 1591—Domain Name System Structure and Delegation, Mar. 1994.
- Ross, K.W., "Hash-Routing for Collections of Shared Web Caches", IEEE Network Magazine, 11, 7:37-44, Nov.-Dec. 1997.
- Schemers, R., "Ibnamed—A load balancing name server written in Perl," 1995 LISA IX—Sep. 17-22, 1995—Monterey, CA.
- Schuba, Christoph; "Addressing Weaknesses in the Domain Name System Protocol," COAST Laboratory, Dept. of Computer Sciences, Purdue University, West Lafayette, IN; Aug. 1993, p. 1-87.
- Smith, "What can Archives offer the World Wide Web?", Technical Report 11, University of Kent, Computing Laboratory, University of Kent, Canterbury, UK, 1994.
- Thaler, D. G. And Ravishankar, C. V. 1998. Using name-based mappings to increase hit rates. IEEE/ACM Trans. Netw. 6, 1 (Feb. 1998), 1-14.
- Vixie, Paul; "Name Server Operations Guide for BIND," Internet Software Consortium; La Honda, CA; p. SMM:10-2-SMM:10-30 (undated, 1996).
- Wessels, Intelligent Caching for World-Wide Web Objects, Masters Thesis, University of Colorado, 1995.
- Claims of European Patent No. 1 143 337 B allowed/maintained in oral proceedings, Apr. 24, 2008 [7 pages].
- Communication from European Patent Office dated Feb. 8, 2007 regarding opposition to European Patent No. 1 143 337 B [11 pages].
- Notice of Opposition to a European Patent filed against European Patent No. 1 143 337 B, by Akamai Technologies, Inc., Apr. 23, 2004 [5 pages].
- Owner's Response of Apr. 10, 2007 to summons to attend oral proceedings dated Feb. 8, 2007 in Opposition Procedure of European Patent No. 1 143 337 B, including auxiliary requests 1 and 2 and attachment discussing claim features [26 pages].
- Owner's Response of Mar. 20, 2008 to summons to attend oral proceedings dated Jan. 24, 2008 in Opposition Procedure of European Patent No. 1 143 337 B, including auxiliary requests 3 to 7 [21 pages].
- Reply of Dec. 21, 2004 of the Patent Proprietor to the Notice of Opposition of European Patent No. 1 143 337 B, Responsive to the Notice of Opposition of Akamai Technologies, Inc. dated Apr. 23, 2004 [11 pages].
- Statement of Opposition to European Patent No. 1 143 337 B, filed by Akamai Technologies, Inc. Apr. 23, 2004 [43 pages].
- Written Submissions in Opposition to European Patent No. 1 143 337 B, by Akamai Technologies, Inc., Apr. 5, 2007 [5 pages].
- Defendant Limelight Networks, Inc.'s Memorandum in Support of its Motion for Summary Judgment of Non-Infringement and Invalidity of all Patents-In-Suit, for Civil Action No. 2:07cv589 (RGD-FBS), Aug. 28, 2008.
- Interlocutory Decision in Opposition proceedings for EPO Appln. No. 00 128 346.4-2211, Patent No. 1143337, Sep. 22, 2008.
- Level 3 Communications, LLC's Non-Confidential Version of its Opposition Brief to Defendant's Motion for Summary Judgment of Non-Infringement and Invalidity of all Patents-In-Suit in Civil Action No. 2:07cv589 (JBF-FBS), Sep. 17, 2008.
- Level 3's Memorandum of Law in Reply to Limelight's Claim Construction Brief for U.S. Patent Nos. 6,654,807; 7,054,935; and 6,473,405, in Civil File No. 2:07CV589, Jun. 27, 2008.
- Level 3's Memorandum of Law in Reply to Limelight's Claim Construction Brief, in Civil File No. 2:07CV589, Jul. 3, 2008.
- Memorandum of Law in Support of Level 3's Proposed Claim Constructions, from Case No. 2:07cv589 RGD FBS, Jun. 13, 2008.
- Minutes of Oral Proceedings Appln. No. 00 128 346.4-2211, Patent No. 1143337, Sep. 22, 2008.
- Amended Answer of Defendant Limelight Networks, Inc., Exhibit 1 to Defendant Limelight Network Inc.'s Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses, in Civil Action No. 2:07cv589, Sep. 9, 2008 [18 pgs].
- Claim construction Opinion and Order in Civil Action No. 2:07cv589, Dec. 10, 2008 [51 pages].
- Communications from European Patent Office in European Patent Application No. 99 906 680.6: Minutes of Oral Hearing of Nov. 20, 2008, including auxiliary requests; and Invitation under Art. 94(3) and Rule 71(1) EPC (dated Dec. 11, 2008) to amended dependent claims. [32 pgs.].
- Declaration Kevin C. Almeroth in Support of Limelight Networks, Inc.'s Motion for Summary Judgment of Non-Infringement and Invalidity of All Patents-In-Suit, filed Aug. 28, 2008 in Civil Action No. 2:07cv589 (RGD-FBS) [39 pgs.].
- Defendant Limelight Network Inc.'s Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses, in Civil Action No. 2:07cv589, Sep. 9, 2008 [4 pgs].
- Defendant Limelight Network Inc.'s Memorandum of Points and Authorities in Support of its Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses (PUBLIC VERSION), in Civil Action No. 2:07cv589, Sep. 9, 2008 [18 pgs.].
- Defendant Limelight Network Inc.'s Memorandum of Points and Authorities in Support of its Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses, (Public Version, redacted) filed Sep. 9, 2008, in civil action No. 2:07-cv-00589-RGD-FBS.
- Defendant Limelight Network Inc.'s Rebuttal Brief in Support of its Combined Motion for Leave to Amend its Answer and Compel Related Interrogatory Responses (Public version), filed Oct. 7, 2008 in civil action No. 2:07-cv-00589-RGD-FBS.
- Defendant Limelight Network Inc.'s Supplemental Responses to Plaintiff's First Set of Interrogatories (Redacted), Apr. 21, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [46 pgs.].
- Defendant Limelight Networks, Inc.'s Claim Construction Brief for U.S. Patent Nos. 6,654,807; 7,054,935; and 6,473,405, filed in Civil Action No. 2:07CV589 (RGD-FBS), Jun. 27, 2008 [38 pgs.].
- Defendant Limelight Networks, Inc.'s Reply Brief in Support of its Motion for Summary Judgment of Non-Infringement and Invalidity of All Patents-In-Suit, filed Oct. 1, 2008 in Civil Action No. 2:07cv589 (JBF-FBS) [21 pgs.].
- Defendant Limelight Networks, Inc.'s Supplemental Responses to Plaintiff's First Set of Interrogatories (Nos. 2-4, 12-13, and 19) [Redacted], May 16, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [70 pgs.].
- Defendant Limelight Networks, Inc.'s Supplemental Responses to Plaintiff's First Set of Interrogatories (Nos. 1-11, 13-14, 17 and 19-20, [redacted] Feb. 9, 2008, in civil action No. 2:07-cv-00589-RGD-FBS [101 pgs.].
- Exhibit A to Level 3 Communications, LLC's Objections and Responses to Limelight Networks, Inc.'s First Set of Interrogatories, Apr. 21, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [68 pgs.].

US 8,478,903 B2

Page 8

Exhibit B to Level 3 Communications, LLC's Objections and Responses to Limelight Networks, Inc.'s First Set of Interrogatories, Apr. 21, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [20 pgs.].

Exhibits A-G of Plaintiff Level 3 Communications, LLC's Memorandum of Law in Opposition to Defendant Limelight Networks, Inc.'s Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses, Sep. 30, 2008, Civil Action No. 2:07cv589, (Exhibit A [7 pgs.], B [5 pgs.], C [47 pgs.], D [45 pgs.], E [7 pgs.], F [8 pgs.], G [5 pgs.]).

Exhibits A-V filed in support of Limelight Networks, Inc.'s Motion for Summary Judgment of Non-Infringement and Invalidity of All Patents-In-Suit, Aug. 28, 2008 in Civil Action No. 2:07cv589 (JBF-FBS) [277 pgs.].

Level 3 Communications Supplemental Exhibit A in Civil Action No. 2:07cv589 (JBF-FBS) [69 pgs.].

Level 3 Communications, LLC's Confidential Version of its Opposition Brief to Defendant's Motion for Summary Judgment of Non-Infringement and Invalidity of All Patents-In-Suit, Sep. 17, 2008, filed in Civil Action No. 2:07cv589 (JBF-FBS) [33 pgs.].

Level 3 Communications, LLC's Objections and Answers to Defendant Limelight Networks, Inc.'s Sixth Set of Interrogatories, Feb. 9, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [9 pgs.].

Level 3 Communications, LLC's Objections and Answers to Defendant Limelight Networks, Inc.'s Third Set of Interrogatories, [redacted], Aug. 18, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [10 pgs.].

Level 3 Communications, LLC's Objections and Responses to Limelight Networks, Inc.'s First Set of Interrogatories, [redacted] Apr. 21, 2008, in civil action No. 2:07-cv-00589-WDK-FBS [28 pgs.].

Level 3 Communications, LLC's Second Supplemental Response to Limelight Networks, Inc.'s First Set of Interrogatories (No. 15), Nov. 2, 2008 in civil action No. 2:07-cv-00589-RGD-FBS [9 pgs.].

Level 3 Communications, LLC's Second Supplemental Response to Limelight Networks, Inc.'s First Set of Interrogatories (No. 4) [Redacted], Nov. 3, 2008 in civil action No. 2:07-cv-00589-RGD-FBS [8 pgs.].

Level 3 Communications, LLC's Supplemental Responses to Limelight Networks, Inc.'s First Set of Interrogatories (No. 10) [Redacted], Jun. 11, 2008 in civil action No. 2:07-cv-00589-RGD-FBS [6 pgs.].

Level 3 Communications, LLC's Supplemental Responses to Limelight Networks, Inc.'s First Set of Interrogatories (Nos. 14 and 15) [Redacted], May 30, 2008 in civil action No. 2:07-cv-00589-RGD-FBS [10 pgs.].

Memorandum on Markman terms to: Judge Robert G. Doumar, Counsel for Level 3 Communications, Counsel for Limelight from: Court appointed technical expert, Professor Ellen W. Zegura Re: Definition of terms, in civil action No. 2:07-cv-00589-MSD-FBS, Level 3 Communications, LLC v. Limelight Networks, Inc. (Aug. 2008).

Order from Judge Doumar in civil action No. 2:07-cv-00589-MSD-FBS, Level 3 Communications, LLC v. Limelight Networks, Inc. instructing Court-Appointed Expert, Professor Ellen W. Zegura Re: Definition of claim terms, Jul. 25, 2008, [2 pgs.].

Plaintiff Level 3 Communications, LLC's Memorandum of Law in Opposition to Defendant Limelight Networks, Inc.'s Combined Motion for Leave to Amend its Answer and to Compel Related Interrogatory Responses, Sep. 30, 2008, Civil Action No. 2:07cv589, [31 pages].

Response to Summons of May 30, 2008 filed in European Patent Office for European Application No. 99 906 680.6 on Oct. 20, 2008. Transcript of 30(b)(6) Deposition of Kevin F. Delgadillo, Tuesday, Sep. 9, 2008 in Civil Action No. 2:07cv589 [39 pages].

Transcript of Deposition of Maurice Andrew Collins, Sep. 9, 2008 in Civil Action No. 2:07CV589 [136 pgs.].

Opinion and Order from U.S. District Judge Mark, S. Davis in Civil Action No. 2:07cv589, denying Defendant Limelight Networks Inc.'s Motion for Summary Judgment, Norfolk, VA, Dec. 29, 2008.

"Opinion and Order," (Markman—claim construction Order), U.S. Dist. Judge Mark S. Davis, Dec. 10, 2008 in Civil Action No. 2:07cv589 (WDK-FBS), in US District Court, Norfolk, VA. [51 pgs.].

Jury Verdict Form in Civil Action No. 2:07cv589 (MSD-FBS), Jan. 23, 2009 [3 pgs.].

Non-Final Office Action in U.S. Appl. No. 11/980,689, May 20, 2009.

Cisco Systems, Inc., <http://www.cisco.com/warp/public/751/disttdir/dd wp.htm>, (1997), 1-4.

Cisco Systems, Inc <http://www.cisco.com/warp/public/751/lodir/lodir ds.htm>, (1997), 1-4.

U.S. P.T.O. Office Action in U.S. Appl. No. 11/980,684, May 19, 2009.

Answer of Defendant Limelight Networks, Inc. in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) Feb. 1, 2008 [14 pgs.].

Answering Report of Scott Bradner Pursuant to Fed. R. Civ. 26 (a)(2)(B) Concerning the Validity of U.S. Patent Nos. 7,054,935 and 6,654,807, in Civil Action No. 2:07CV589 (RGD-FBS) (E.D. Va.), Aug. 25, 2008 [83 pgs.].

Defendant Limelight Networks, Inc.'s Claim Construction Brief for U.S. Patent Nos. 6,654,807; 7,054,935; and 6,473,405, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.), Jun. 27, 2008 [38 pgs. + exhibits].

Defendant Limelight Networks, Inc.'s Reply Brief in Support of its Motion for Summary Judgment of Non-Infringement and Invalidity of All Patents-In-Suit, in Civil File No. 2:07CV589 (E.D. Va.), Oct. 1, 2008 [21 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 205, doc. titled "Leapfrog Project Overview," Andrew Swart, Aug. 29, 1996 [2 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 206, doc. titled "Leapfrog Management Meeting Agenda," Oct. 1, 1996 [1 pg.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 207, letter from Andrew D. Swart, Sandpiper Networks to Jean-Claude Asscher, Feb. 25, 1997 [2 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 209, email from David A. Farber sent Apr. 19, 1999 [5 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 210, email from Andrew Swart sent Apr. 24, 1999 [2 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 211, email from Will Crowder sent May 6, 1999 [10 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 212, email from Rick Greer sent Sep. 20, 1999 [11 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 213, email from Mike Warren sent Feb. 25, 2000 [3 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 216, email from "Andrew" sent Nov. 30, 1999 [7 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 219, email from "Andrew" sent Sep. 24, 1996 [3 pgs.].

Deposition of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 220, document titled "Statement of Opposition, European Patent No. 1 143 337 B", Apr. 23, 2004 [41 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 154, email [4 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 155, document titled "WebRepeater™ Functional Requirements," Dave Farber, Sandpiper Software Consulting, LLC, Jun. 18, 1996. [20 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 156 doc. titled "Project Status Meeting Reports," Aug. 4, 1997 [3 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 157, parts of U.S. Appl. No. 09/021,506 from PTO [68 pgs.].

US 8,478,903 B2

Page 9

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 158, documents from PTO file for U.S. Appl. No. 09/612,598 [59 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 159, email from Dave Farber sent Mar. 14, 1997 [12 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 160, email from "Andrew" sent Jan. 15, 1997 [3 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 161, email from "Will", sent Oct. 17, 1997 [24 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 162, email from Rick Greer sent Jan. 20, 1999 [7 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 163, email from "Andrew" sent Jul. 22, 1996 [2 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 164, European Patent No. 1 143 337 (WO 99/40514) [64 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 165, European Patent Specification No. 1 143 337 [31 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 166, email from Will Crowder sent Aug. 24, 1997 [1 pg.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 167, email from Chris Newton sent Oct. 26, 2000 [7 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 168, document titled "Sandpiper Raises \$6.6 Million to Boost Content Distribution for Web Publishers," May 19, 1998 [2 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 169 document titled "Sandpiper Launches Footprint Distribution Service to Improve Web Site Performance; Cost-Effective Service Manages Content Delivery for Web Publishers," Sep. 28, 1998 [3 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 170 document titled "Sandpiper Completes New Executive Management Team; Sandpiper Poised for Growth with Strong Executive Team," Sep. 28, 1998 [2 pgs.].

Deposition of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Exhibit 171 document titled "Sandpiper Addresses Web Content Management Problems At Herring on Hollywood;—New Technology Company Offers Solutions to Entertainment Industry's Internet Woes," Sep. 28, 1998.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 10, email from David A. Farber, Jan. 11, 1999 [4 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 11, U.S. Pat. No. 6,185,598 [23 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 12, email from David A. Farber, Jan. 18, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 13, email from Will Crowder, Feb. 17, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 155 document titled "WebRepeater™ Functional Requirements," Dave Farber, Sandpiper Software Consulting, LLC, Jun. 18, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 16 email from Will Crowder, May 6, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 166, email from Will Crowder, Aug. 24, 1997.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 19, email from David A. Farber, May 12, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 2, email from Dave Farber dated Oct. 15, 1997 [2 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 22, document titled "Patent Claim: Resource Request Redirector," Dave Farber, Sandpiper Software Consulting, LLC, Draft 3, May 24, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 226, email from Peggy Balesteri, Aug. 7, 2008.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 227, email from Dave Farber, May 29, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 228, email from "Dave" dated Jul. 12, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 229, email from "Andrew" dated Jul. 22, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 230, email from "Dave" dated Jul. 10, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 231, document titled "Patent Claim: Network Reflector/Repeater," Dave Farber, Sandpiper Software Consulting, LLC, Draft 5, Jul. 2, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 232, document titled "WebRepeater Features Comparison", Sandpiper Software Consulting, Sep. 17, 1996.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 233, email from "Hussain" sent Sep. 19, 1998.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 234, email from "Scott" sent May 30, 1998.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 235, email from "Andrew" sent Feb. 12, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 236, email from David A. Farber sent Apr. 19, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 237, email from David A. Farber sent May 12, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 238, email from David A. Farber sent May 6, 1999 [1 pg.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 239, document titled "Footprint DNS Server Mini-Specification," Will Crowder, Apr. 23, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 240, email from "David" sent Jun. 4, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 241, email from David A. Farber sent Dec. 6, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 242, email from Chris Newton, sent Oct. 26, 2000.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 243, email from "Andrew" sent Jul. 24, 2000.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 244, email from Andrew Swart sent Nov. 12, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 245, email from Andrew Swart sent Aug. 23, 2002.

US 8,478,903 B2

Page 10

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 246, email from "Jim" sent Dec. 16, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 247, email from "Jim" sent Nov. 17, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 248, email from "Jim" sent Nov. 17, 1999.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 249, email from Will Crowder sent Dec. 21, 1998.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 250, email from Chris Newton sent Feb. 15, 2000.

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 5, U.S. Pat. No. 6,654,807 [28 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 6, U.S. Pat. No. 7,054,935 [14 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 7, email from David A. Farber, Dec. 21, 1998 [2 pgs.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 8, email from David A. Farber, Jan. 11, 1999 [1 pg.].

Deposition of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 9, email from Jim Baiter, Apr. 19, 1999 [4 pgs.].

Deposition of James A. Balter in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Deposition Exhibit 1, Subpoena in Civil Case [10 pgs.].

Deposition of James A. Balter in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Deposition Exhibit 2, email from Dave Farber dated Oct. 15, 1997 [2 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 10, email from David A. Farber dated Jan. 11, 1999 [4 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 11, U.S. Pat. No. 6,185,598 [23 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 12, email from David A. Farber dated Jan. 18, 1999 [4 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 13, email from Will Crowder dated Feb. 17, 1999 [9 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 14, email from Jim Baiter dated Sep. 08, 2000 [3 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 15, email from Will Crowder dated May 12, 1999 [2 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 16, email from Will Crowder dated May 6, 1999 [2 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 17, email from Will Crowder dated Apr. 22, 1999 [9 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 18, email from Jim Baiter dated Apr. 22, 1999 [1 pg.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 19, email from David A. Farber dated May 12, 1999 [1 pg.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 20, email from If-tech@sandpiper.com dated Nov. 25, 1997 [3 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 21, document titled "Patent Claim: HTTP Redirector" Draft 2, May 20, 1996, Dave Farber, Sandpiper Software Consulting, LLC [11 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 22, document titled "Patent Claim: Resource Request Redirector," Dave Farber, Sandpiper Software Consulting, LLC, Draft 3, May 24, 1996 [17 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 23, document titled "Patent Claim: Network Reflector/Repeater," Dave Farber, Sandpiper Software Consulting, LLC, Draft 5, Jul. 2, 1996 [21 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 24, document titled "Software Design Specification: Advanced Rewrite Tool," J. Baiter, Dec. 3, 1999, Sandpiper Networks, Inc.

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 3, email from Andrew Swart dated Apr. 1, 1998 [1 pg.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 4, email from Rick [Richard] Greer dated Dec. 14, 1998 [6 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 5, U.S. Patent No. 6,654,807 [28 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 6, U.S. Patent No. 7,054,935 [24 pgs.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 7, email from David A. Farber dated Dec. 21, 1998 [2 pg.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 8, email from David A. Farber dated Jan. 11, 1999 [1 pg.].

Deposition of James A. Balter, Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Aug. 8, 2008, Depo. Exhibit 9, email from Jim [James] Baiter dated Apr. 19, 1999 [4 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 11, U.S. Pat. No. 6,185,598 [23 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 12, email from David A. Farber sent Jan. 18, 1999 [4 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 195, document titled "Web Repeater" (Interactive Systems Corporation), Sandpiper Software, Nov. 23, 1996, [80 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 196, Subpoena in Civil Case 2:07cv589 RGD/FBS (E.D. Va.), 6Jun. 19, 2008 [10 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 197, email from Rick Greer sent Feb. 18, 1999 [3 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 198, email from Rick Greer sent Jan. 20, 1999 [7 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 199, email from Rick@sandpiper.com sent Oct. 9, 1998.

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 200 email from Rick@sandpiper.com sent Oct. 2, 1998 [2 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 201, email from Will Crowder sent Dec. 21, 1998 [2 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 202, email from rick@digisle.com sent Nov. 20, 2000 [2 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 203, mail from David A. Farber sent Nov. 3, 1999 [1 pg.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 4, email from Rick Greer sent Dec. 14, 1998 [6 pgs.].

US 8,478,903 B2

Page 11

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 5, U.S. Pat. No. 6,654,807 [28 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 6, U.S. Pat. No. 7,054,935 [24 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 7, email from David A. Farber sent Dec. 21, 1998 [2 pgs.].

Deposition of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.), Depo. Exhibit 8, email from David A. Farber sent Jan. 11, 1999 [4 pgs.].

Deposition Transcript of Andrew D. Swart, Aug. 21, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) [redacted, 98 pgs.].

Deposition Transcript of Andrew D. Swart, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) [redacted, 78 pgs.].

Deposition transcript of David A. Farber, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) [87 pgs.].

Deposition Transcript of James A. Balter, August 8, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) [69 pgs.].

Deposition transcript of Richard E. Greer, Sep. 3, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) (E.D. Va.) [79 pgs.].

Exhibit A to Reply and Supplemental Report of Dr. Kevin C. Almeroth to the Answering Report of Scott Bradner, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Sep. 5, 2008 [14 pgs.].

Exhibit B to Reply and Supplemental Report of Dr. Kevin C. Almeroth to the Answering Report of Scott Bradner, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Sep. 5, 2008 [12 pgs.].

Exhibit C to Reply and Supplemental Report of Dr. Kevin C. Almeroth to the Answering Report of Scott Bradner, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Sep. 5, 2008 [12 pgs.].

Exhibit D to Reply and Supplemental Report of Dr. Kevin C. Almeroth to the Answering Report of Scott Bradner, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Sep. 5, 2008 [13 pgs.].

Joint Statement Regarding Claim Construction with Exhibit "The '807 and '935 Footprint Patents," in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.), filed Jul. 9, 2008 [27 pgs.].

Level 3's Memorandum of Law in Reply to Limelight's Claim Construction Brief, in Civil File No. 2:07CV589 (E.D. Va.), Jul. 3, 2008 [24 pgs. + exhibits].

Level 3's Motion for Judgment As a Matter of Law or, in the Alternative, for a New Trial Pursuant to Rules 50(B) and 59(A), in Civil File No. 2:07CV589 (E.D. Va.), Feb. 9, 2009 [3 pgs.].

Level 3's Reply in Further Support of Its Motion for Judgment As a Matter of Law or, in the Alternative, for a New Trial Pursuant to Rules 50(B) and 59(A), Mar. 12, 2009, in Civil Action No. 2:07cv589 (MSD-FBS) (E.D. Va.) [22 pgs.].

Limelight Networks, Inc.'s Claim Construction Hearing Presentation, Jul. 14, 2008, in Civil File No. 2:07CV589 (E.D. Va.) [123 pgs.].

Limelight's Opposition to Level 3's Motion for Judgment As a Matter of Law or, in the Alternative, for a New Trial Pursuant to Rules 50(B) and 59(A), Mar. 2, 2009, in Civil Action No. 2:07cv589 (MSD-FBS) (E.D. Va.) [31 pgs.].

Markman PowerPoint Presentation, Jul. 9, 2008 in Civil File No. 2:07CV589 (E.D. Va.) [16 pgs.].

Memorandum of Law in Support of Level 3's Proposed Claim Constructions in Case No. 2:07cv589 RDG FBS, (E.D. Va.) Jun. 14, 2008 [32 pgs. + exhibits].

Reply and Supplemental Report of Dr. Kevin C. Almeroth to the Answering Report of Scott Bradner, in Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Sep. 5, 2008 [62 pgs.].

Second Supplemental Expert Report of Dr. Kevin C. Almeroth Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.) Dec. 24, 2008 [9 pgs.].

Supplemental Answering Report of Scott Bradner Pursuant to Fed. R. Civ. 26(a)(2)(B) Concerning the Validity of U.S. Patent No. 6,654,807 Civil Action No. 2:07CV589 (RGD-FBS) (E.D. Va.) Dec. 10, 2008 [28 pgs.].

Transcript of Proceedings for Jury Trial, Norfolk, VA, Jan. 7-23, 2009, in Civil Action No. 2:07cv589 (E.D. Va.) [1052 pgs.].

Transcript of Proceedings for Markman Follow-up Hearing, Norfolk, VA, Oct. 23, 2008, in Civil Action No. 2:07cv589 (E.D. Va.) [58 pgs.].

Transcript of Proceedings for Markman Hearing, Norfolk, VA, Jul. 14, 2008, in Civil Action No. 2:07cv589 (E.D. Va.) [69 pgs.].

European Patent Office (EPO), European Search Report mailed Aug. 19, 2010 in European Appln. No. 08005349.9 [10 pgs.].

Akamai Tech., Inc. v. Cable & Wireless Internet Svs., Inc., 344 F.3d 1186, 68 USPQ2d 1186 (Fed. Cir. 2003) [Sep. 15, 2003].

European Patent Office (EPO) Summons to attend oral proceedings pursuant to Rule 115(1) EPC, dated Aug. 4, 2011, for European Patent No. 99906680.6, including accompanying communication (EPO Form 2906), "Facts & Submissions" [20 pgs.].

EPO, Annex to Minutes of the Oral Proceedings Before the Opposition Division, Claims of Main Request and new first (noted as IV) and second Auxiliary Request (noted as II), Apr. 24, 2012, European Application No. 99 906 680.6, patent No. EP-B-1053524 [22 pgs.]. EPO, Interlocutory Decision in Opposition Proceedings, Allowed text for re-granted patent, Apr. 24, 2012, European Application No. 99 906 680.6, patent No. EP-B-1053524, European patent No. EP-B-1 053 524 [29 pgs.].

EPO, Interlocutory Decision in Opposition Proceedings, Apr. 24, 2012, European Application No. 99 906 680.6, patent No. EP-B-1053524 [2 pgs.] and enclosures.

EPO, Interlocutory Decision in Opposition Proceedings, Grounds for Decision, Apr. 24, 2012, European Application No. 99 906 680.6, patent No. EP-B-1053524 [8 pgs.].

EPO, Minutes of the Oral Proceedings Before the Opposition Division, published Apr. 24, 2012, proceedings held Mar. 6, 2012, European Application No. 99 906 680.6, patent No. EP-B-1053524 [12 pgs.].

"Memorandum in Support of Level 3's Motion in Limine to Preclude Limelight Witnesses from Referring to Prosecution History in Rendering Non-Infringement Opinions," filed in Civil Action No. 2:07cv589 (MSD-FBS), U.S. Dist. Ct. E.D. Va., Jan. 15, 2009 [7 pgs.].

"Memorandum in Support of Limelight's Motion in Limine No. 7 to Exclude Testimony From Level 3's Experts Not Disclosed in Their Expert Reports Regarding the Disclosures of the Patents-In-Suit," filed in Civil Action No. 2:07cv589 (MSD-FBS), U.S. Dist. Ct. E.D. Va., Dec. 4, 2008 [6 pgs.].

Combined Petition for Panel Rehearing or Rehearing En Banc, U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight Networks, Inc., 2009-1480, Jun. 4, 2010 [22 pgs.].

Communication dated Apr. 27, 2010 from European Patent Office re Opposition to European Patent application No. (99906680.6-2211) Patent No. 1 053 524 B [1 pg.].

Defendant Limelight Networks, Inc.'s Claim Construction Brief for U.S. Patent Nos. 6,654,807; 7,054,935 and 6,473,405, in Civil Action No. 2:07cv589, Jun. 27, 2008, with Exhibits 1-24 [333 pgs.].

Exhibit 56 to Deposition of Maurice Andrew Collins in Civil Action No. 2:07cv589 (WDK-FBS), "Defendant Limelight Networks, Inc.'s Notice of Deposition of Andrew Collins," Aug. 29, 2008 [28 pgs.].

Exhibit 57 to Deposition of Maurice Andrew Collins in Civil Action No. 2:07cv589 (WDK-FBS), Document titled "Detour: Informed Internet Routing and Transport," Savage, et al., IEEE Micro, vol. 19, issue 1, pp. 50-59, Jan./Feb. 1999 [10 pgs.].

Exhibit 58 to Deposition of Maurice Andrew Collins in Civil Action No. 2:07cv589 (WDK-FBS), document titled Internet Archive Wayback Machine, sample search Sep. 9, 2008 [1 pg.].

Exhibit 59 to Deposition of Maurice Andrew Collins in Civil Action No. 2:07cv589 (WDK-FBS), Document titled "Detour," deponent Collins, Sep. 9, 2008, [1 pg.].

Exhibit 65 to Deposition of Kevin F. Delgadillo, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS), Document titled Subpoena in a Civil Case [12 pgs.].

Exhibit 66 to Deposition of Kevin F. Delgadillo, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS), Document titled, "Cisco DistributedDirector," [9 pgs.].

Exhibit 67 to Deposition of Kevin F. Delgadillo, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS), Document titled, "Electronic Publishing Form," dated Nov. 27, 1996 [5 pgs.].

Exhibit 68 to Deposition of Kevin F. Delgadillo, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS), Email from Kevin Delgadillo, dated Jul. 19, 1996 [1 pg.].

Exhibit 69 to Deposition of Kevin F. Delgadillo, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS), Document titled, "Cisco Advantage: Leveraging Intranet and Internet Productivity," [9 pgs.].

US 8,478,903 B2

Page 12

Letter from Chris Hirsz dated Apr. 22, 2010 to European Patent Office forwarding Statement of Opposition to European Patent No. 1 053 524 B in the name of Level 3 CDN International, Inc. filed in European Patent Office by Akamai Technologies, Inc., Apr. 22, 2004 [2 pgs.].

Level 3 Communications, LLC's Supplemental Responses to Limelight Networks, Inc.'s First Set of Interrogatories (No. 4), in civil action No. 2:07-cv-00589-RGD-FBS, in U.S. Dist. Ct. E.D. Va., Sep. 3, 2008 [8 pgs.].

Limelight Networks Inc.'s Response to the Report of Professor Ellen W. Zegura, Sep. 5, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) [5 pgs.].

Notice of Abandonment from Canadian Intellectual Property Office for Canadian Patent Application No. 2,320,261 [1 pg.].

Office Action (Notice of Reasons for Rejection) in Japanese Patent Appln. No. 2007-212908, Mar. 30, 2010, JPO [3 pgs.] with summary English translation [3 pgs.].

Office Action dated Sep. 24, 2007 from Canadian Intellectual Property Office for Canadian Patent Application No. 2,320,261 [5 pgs.].

Office Action dated Sep. 28, 2006 from Canadian Intellectual Property Office for Canadian Patent Application No. 2,320,261 [4 pgs.].

Opening Brief of Appellant Level 3 Communications, LLC, U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight Networks, Inc., 2009-1480, Oct. 5, 2009 [106 pgs.].

Opinion and Order from U.S. District Judge Mark. S. Davis in Civil Action No. 2:07cv589, ruling on Level 3 Communications, LLC's post-trial motions, Norfolk, Va, Jun. 23, 2009 [26 pgs.].

Order denying Appellant's Petition for Panel Rehearing or Rehearing EN Banc, U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight Networks, Inc., 2009-1480, Jun. 29, 2010 [2 pgs.].

Plaintiff Level 3 Communications, LLC'S Response to Defendant Limelight Networks, Inc.'S Response to Report of Professor Ellen Zegura, Sep. 9, 2008, in Civil Action No. 2:07cv589 (WDK-FBS) [5 pgs.].

Reinstatement and Response, filed Apr. 24, 2009 in Canadian Intellectual Property Office for Canadian Patent Application No. 2,320,261 [18 pgs.].

Reply Brief of Appellant Level 3 Communications, LLC, U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight Networks, Inc., 2009-1480, Feb. 2, 2010 [41 pgs.].

Response Brief of Defendant-Appellee Limelight Networks, Inc., U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight Networks, Inc., 2009-1480, Jan. 19, 2010 [72 pgs.].

Response to Office Action, filed Mar. 28, 2007 in Canadian Intellectual Property Office for Canadian Patent Application No. 2,320,261 [24 pgs.].

Statement of Opposition to European Patent No. 1 053 524 B in the name of Level 3 CDN International, Inc. filed in European Patent Office by Akamai Technologies, Inc., Apr. 22, 2004 [40 pgs.].

Transcript of Deposition of Kevin C. Almeroth, Civil Action No. 2:07cv589 (RGD-FBS) (E.D. Va.), Sep. 11, 2008 (redacted) [98 pgs.].

Transcript of Federal Circuit Oral Argument, Level 3 Communications LLC vs. Limelight, Monday, May 3, 2010 [14 pgs.].

Transcript of Oral Argument, U.S. Ct. Appeals for Federal Circuit, Level 3 Communications LLC vs. Limelight, Monday, May 3, 2010 [14 pgs.].

Transcript of Proceedings (Motion for Summary Judgment), in Civil Action No. 2:07cv589 in U.S. Dist. Ct. E.D. Va., Norfolk, Va., Dec. 18, 2008, [145 pgs.].

Transcript of Proceedings, Hearing before Hon. Mark S. Davis, U.S. District Judge, in Civil Action No. 2:07cv589 (E.D. Va.), Oct. 23, 2008 [62 pgs.].

U.S. Ct. Appeals for Fed. Circuit, 2009-1490, Level 3 Communications, LLC v. Limelight Networks, Inc., Judgment Affirmed, Fed. Cir. R. 36, May 5, 2010 [2 pgs.].

Yokohashi, Takao, System Architectonic Solaris2.x, 26th, Unix User, Japan, Softbank Corp., Jan. 1, 1998, vol. 7, No. 1, p. 89-96.

European Patent Office (EPO) Communication (examination report) dated Jun. 15, 2012 from EP08005349.9 [6 pgs.].

European Patent Office (EPO) Communication (search report) dated Jul. 4, 2012 from EP07024629.3 [4 pgs.].

Partial European Search Report and Section 6.2, EPO communication dated Oct. 18, 2012 in EP Application No. 07024629.3.

* cited by examiner

U.S. Patent

Jul. 2, 2013

Sheet 1 of 6

US 8,478,903 B2

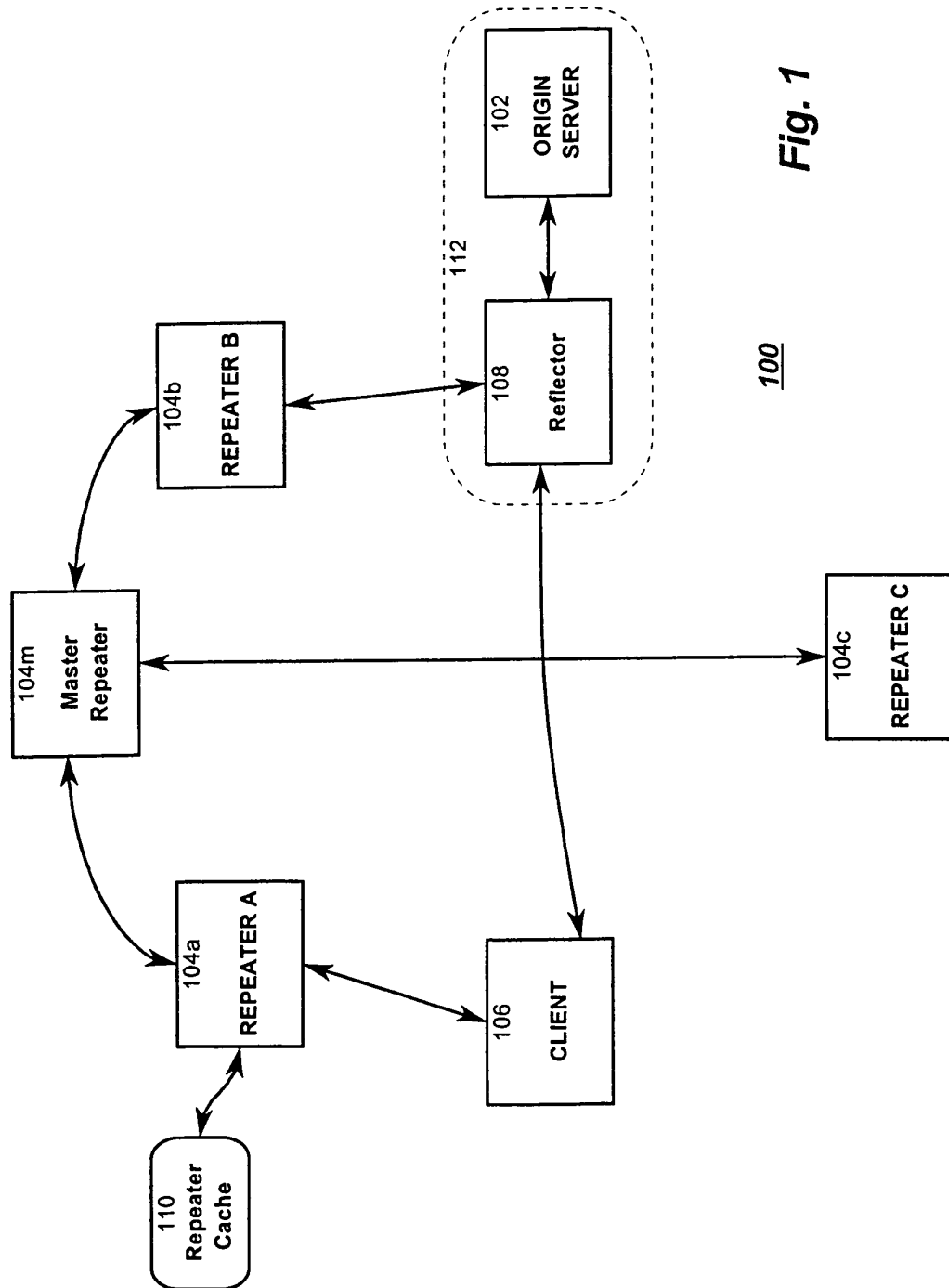


Fig. 1

U.S. Patent

Jul. 2, 2013

Sheet 2 of 6

US 8,478,903 B2

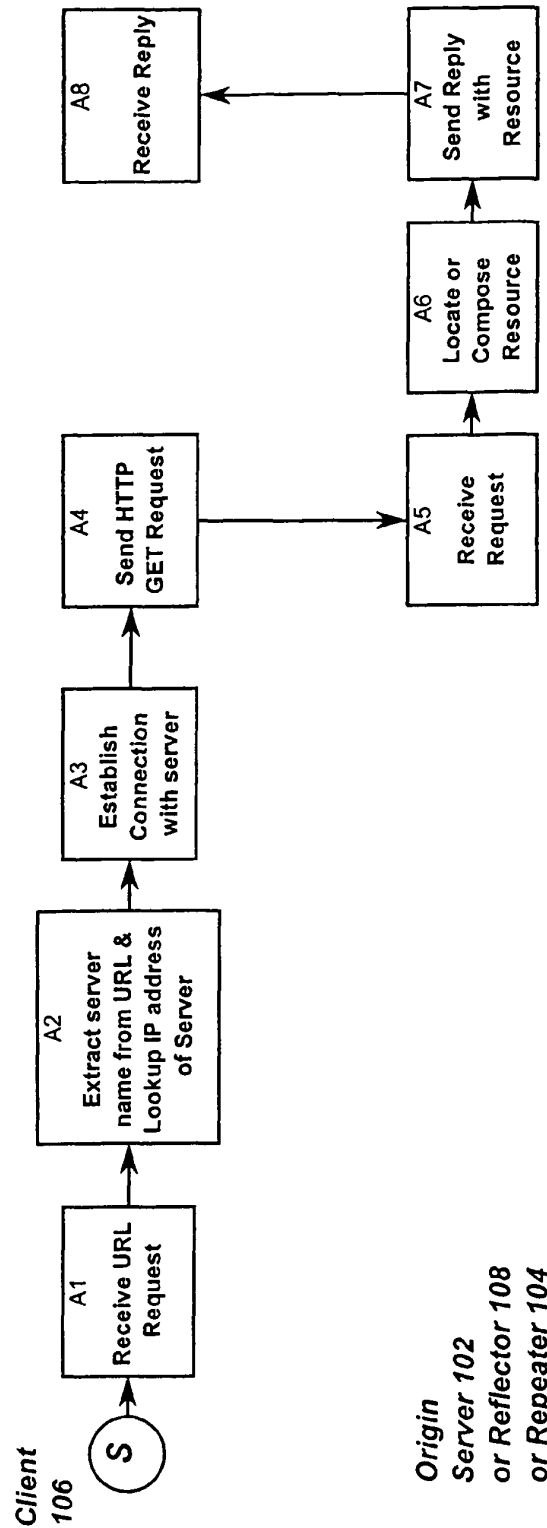


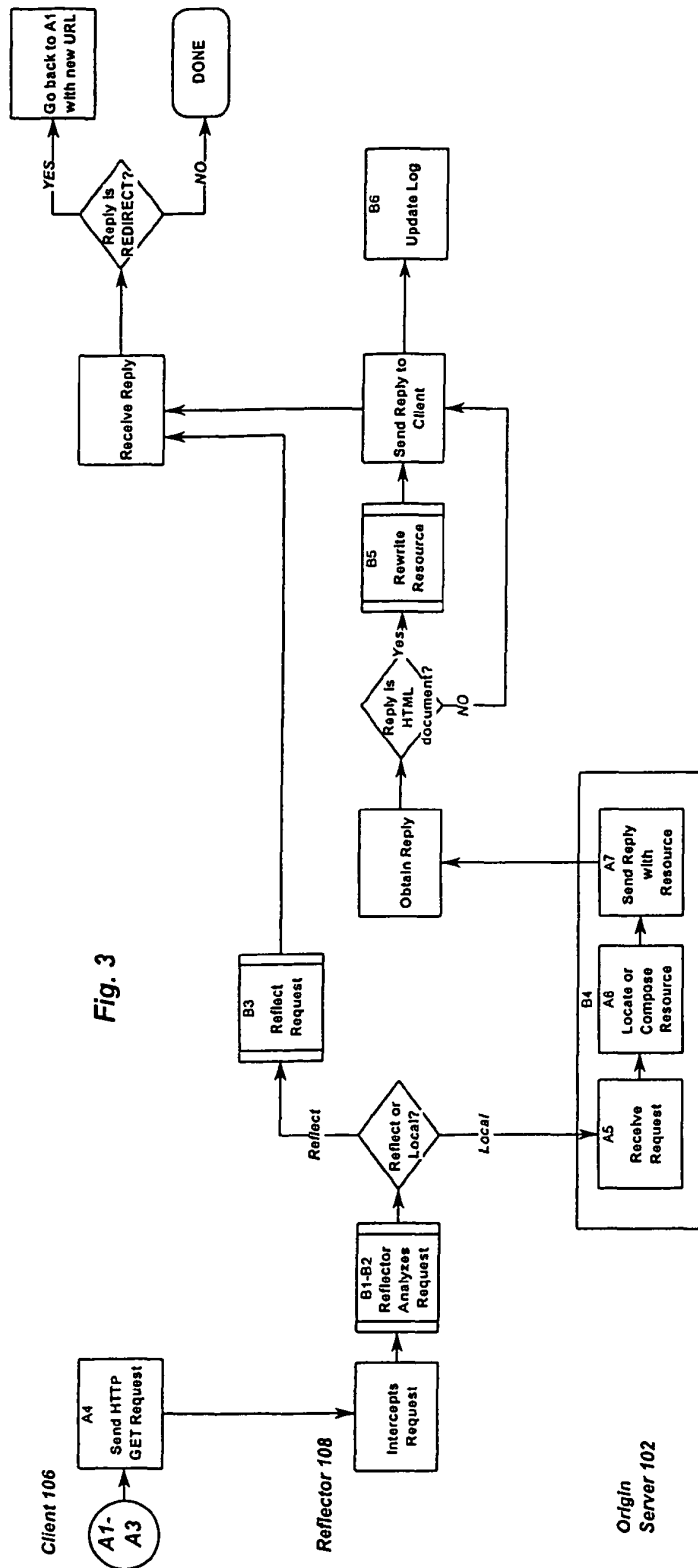
Fig. 2

U.S. Patent

Jul. 2, 2013

Sheet 3 of 6

US 8,478,903 B2



U.S. Patent

Jul. 2, 2013

Sheet 4 of 6

US 8,478,903 B2

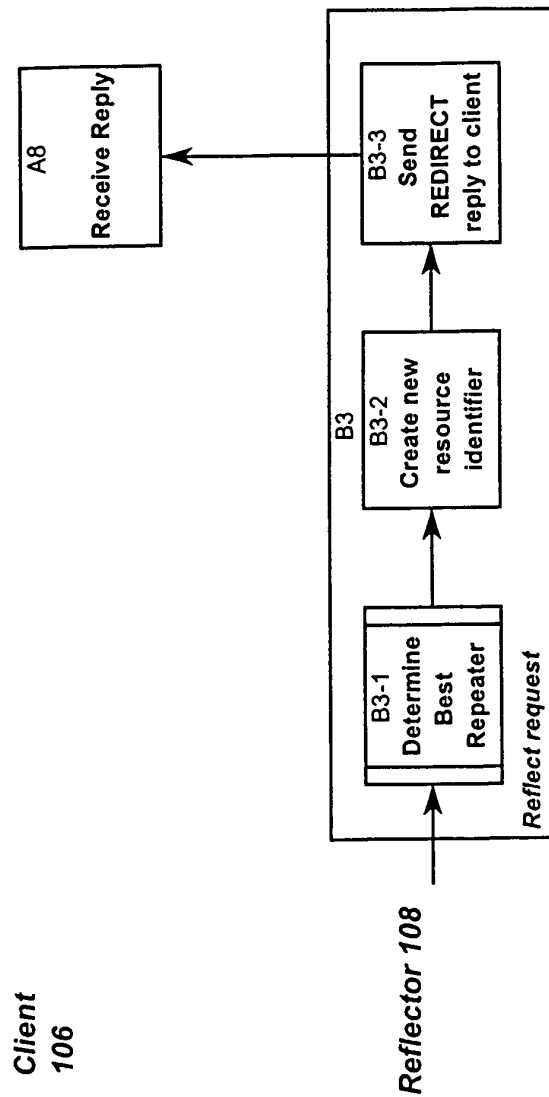


Fig. 4

U.S. Patent

Jul. 2, 2013

Sheet 5 of 6

US 8,478,903 B2

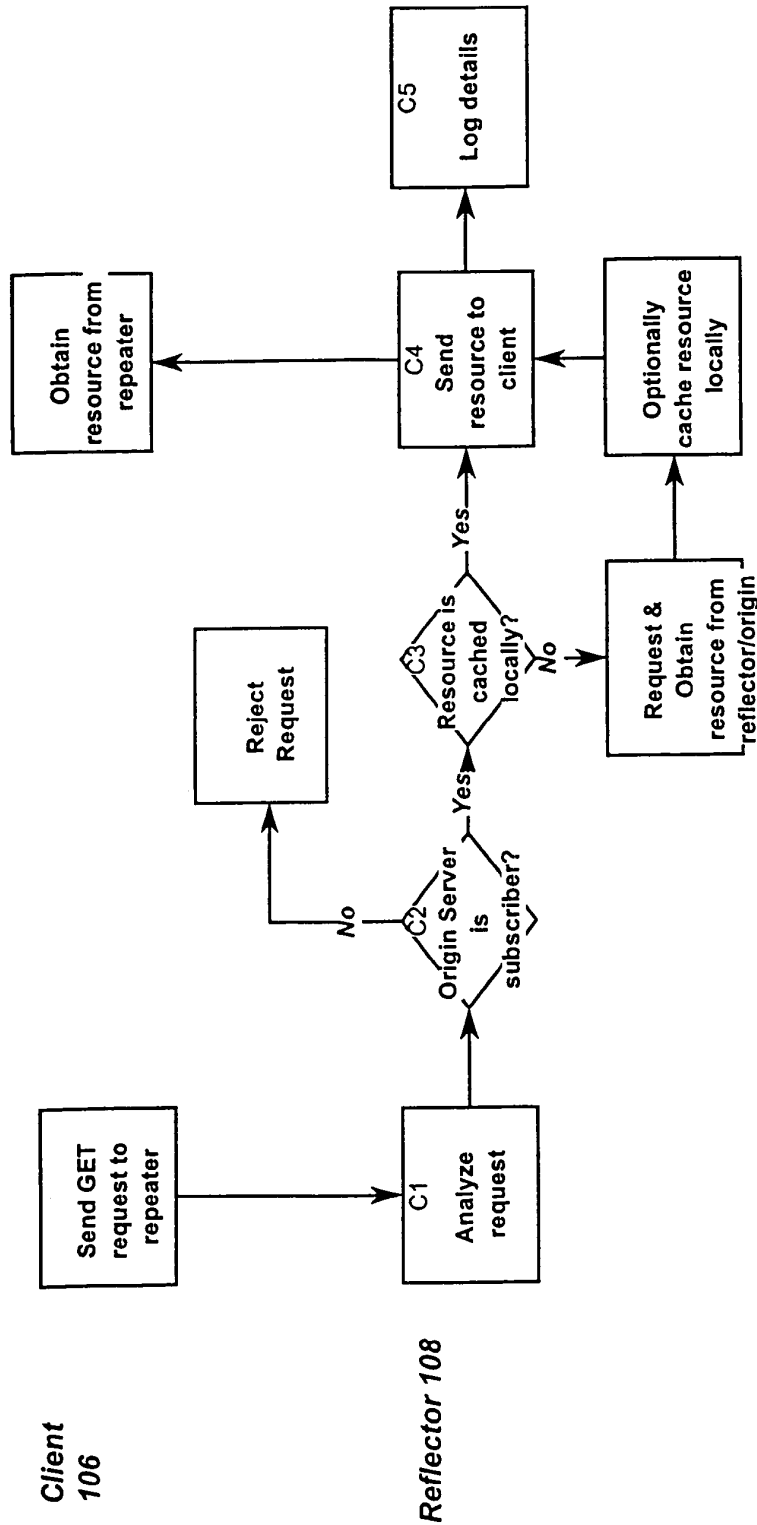


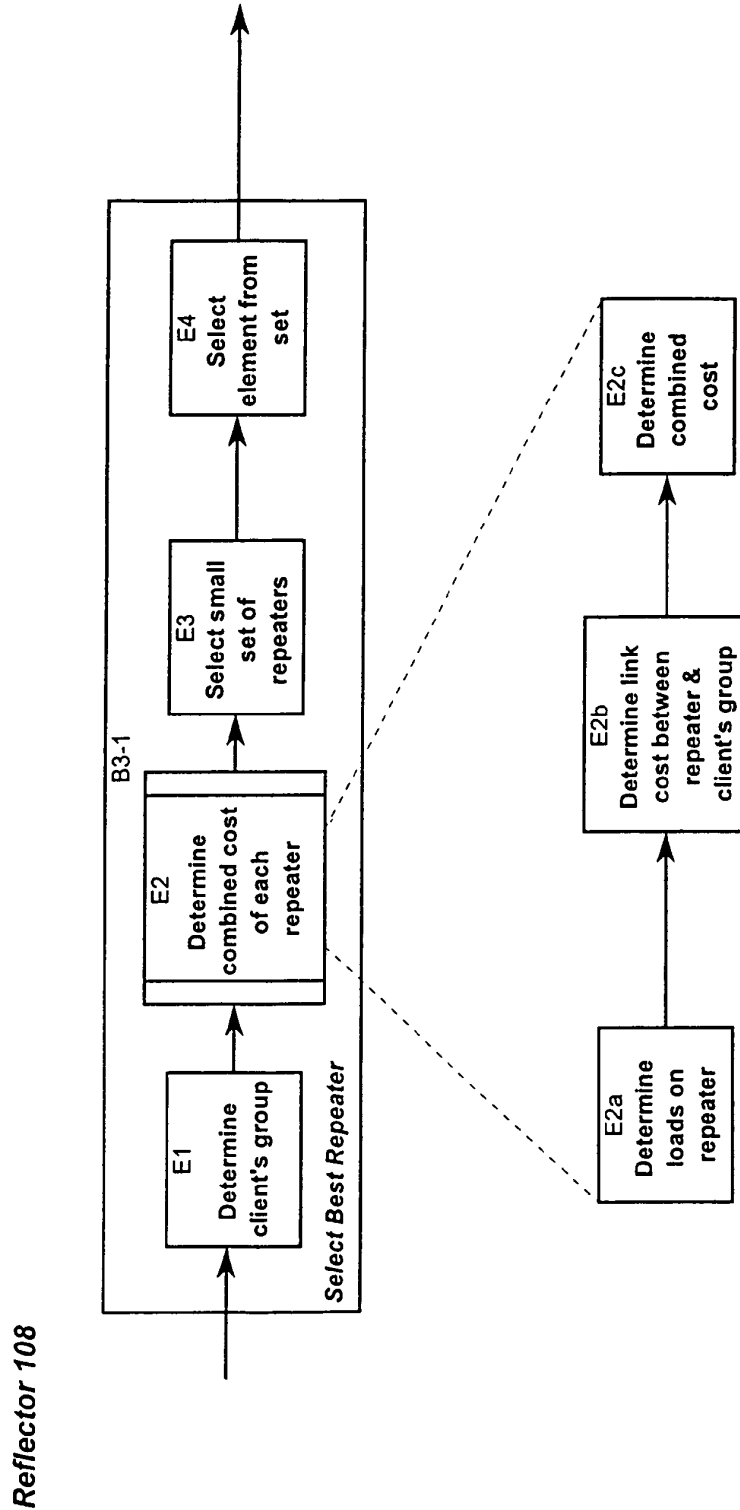
Fig. 5

U.S. Patent

Jul. 2, 2013

Sheet 6 of 6

US 8,478,903 B2



US 8,478,903 B2

1

SHARED CONTENT DELIVERY INFRASTRUCTURE

This application is a continuation of application Ser. No. 09/612,598 filed Jul. 7, 2000, now abandoned which is a divisional of application Ser. No. 09/021,506 filed Feb. 10, 1998, now U.S. Pat. No. 6,185,598.

FIELD OF THE INVENTION

This invention relates to replication of resources in computer networks.

BACKGROUND OF THE INVENTION

The advent of global computer networks, such as the Internet, have led to entirely new and different ways to obtain information. A user of the Internet can now access information from anywhere in the world, with no regard for the actual location of either the user or the information. A user can obtain information simply by knowing a network address for the information and providing that address to an appropriate application program such as a network browser.

The rapid growth in popularity of the Internet has imposed a heavy traffic burden on the entire network. Solutions to problems of demand (e.g., better accessibility and faster communication links) only increase the strain on the supply. Internet Web sites (referred to here as “publishers”) must handle ever-increasing bandwidth needs, accommodate dynamic changes in load, and improve performance for distant browsing clients, especially those overseas. The adoption of content-rich applications, such as live audio and video, has further exacerbated the problem.

To address basic bandwidth growth needs, a Web publisher typically subscribes to additional bandwidth from an Internet service provider (ISP), whether in the form of larger or additional “pipes” or channels from the ISP to the publisher’s premises, or in the form of large bandwidth commitments in an ISP’s remote hosting server collection. These increments are not always as fine-grained as the publisher needs, and quite often lead times can cause the publisher’s Web site capacity to lag behind demand.

To address more serious bandwidth growth problems, publishers may develop more complex and costly custom solutions. The solution to the most common need, increasing capacity, is generally based on replication of hardware resources and site content (known as mirroring), and duplication of bandwidth resources. These solutions, however, are difficult and expensive to deploy and operate. As a result, only the largest publishers can afford them, since only those publishers can amortize the costs over many customers (and Web site hits).

A number of solutions have been developed to advance replication and mirroring. In general, these technologies are designed for use by a single Web site and do not include features that allow their components to be shared by many Web sites simultaneously.

Some solution mechanisms offer replication software that helps keep mirrored servers up-to-date. These mechanisms generally operate by making a complete copy of a file system. One such system operates by transparently keeping multiple copies of a file system in synch. Another system provides mechanisms for explicitly and regularly copying files that have changed. Database systems are particularly difficult to replicate, as they are continually changing. Several mechanisms allow for replication of databases, although there are no standard approaches for accomplishing it. Several companies

2

offering proxy caches describe them as replication tools. However, proxy caches differ because they are operated on behalf of clients rather than publishers.

Once a Web site is served by multiple servers, a challenge is to ensure that the load is appropriately distributed or balanced among those servers. Domain name-server-based round-robin address resolution causes different clients to be directed to different mirrors.

Another solution, load balancing, takes into account the load at each server (measured in a variety of ways) to select which server should handle a particular request.

Load balancers use a variety of techniques to route the request to the appropriate server. Most of those load-balancing techniques require that each server be an exact replica of the primary Web site. Load balancers do not take into account the “network distance” between the client and candidate mirror servers.

Assuming that client protocols cannot easily change, there are two major problems in the deployment of replicated resources. The first is how to select which copy of the resource to use. That is, when a request for a resource is made to a single server, how should the choice of a replica of the server (or of that data) be made. We call this problem the “rendezvous problem”. There are a number of ways to get clients to rendezvous at distant mirror servers. These technologies, like load balancers, must route a request to an appropriate server, but unlike load balancers, they take network performance and topology into account in making the determination.

A number of companies offer products which improve network performance by prioritizing and filtering network traffic.

Proxy caches provide a way for client aggregators to reduce network resource consumption by storing copies of popular resources close to the end users. A client aggregator is an Internet service provider or other organization that brings a large number of clients operating browsers to the Internet. Client aggregators may use proxy caches to reduce the bandwidth required to serve web content to these browsers. However, traditional proxy caches are operated on behalf of Web clients rather than Web publishers.

Proxy caches store the most popular resources from all publishers, which means they must be very large to achieve reasonable cache efficiency. (The efficiency of a cache is defined as the number of requests for resources which are already cached divided by the total number of requests.)

Proxy caches depend on cache control hints delivered with resources to determine when the resources should be replaced. These hints are predictive, and are necessarily often incorrect, so proxy caches frequently serve stale data. In many cases, proxy cache operators instruct their proxy to ignore hints in order to make the cache more efficient, even though this causes it to more frequently serve stale data.

Proxy caches hide the activity of clients from publishers. Once a resource is cached, the publisher has no way of knowing how often it was accessed from the cache.

SUMMARY OF THE INVENTION

This invention provides a way for servers in a computer network to off-load their processing of requests for selected resources by determining a different server (a “repeater”) to process those requests. The selection of the repeater can be made dynamically, based on information about possible repeaters.

US 8,478,903 B2

3

If a requested resource contains references to other resources, some or all of these references can be replaced by references to repeaters.

Accordingly, in one aspect, this invention is a method of processing resource requests in a computer network. First a client makes a request for a particular resource from an origin server, the request including a resource identifier for the particular resource, the resource identifier sometimes including an indication of the origin server. Requests arriving at the origin server do not always include an indication of the origin server; since they are sent to the origin server, they do not need to name it. A mechanism referred to as a reflector, co-located with the origin server, intercepts the request from the client to the origin server and decides whether to reflect the request or to handle it locally. If the reflector decides to handle the request locally, it forwards it to the origin server, otherwise it selects a "best" repeater to process the request. If the request is reflected, the client is provided with a modified resource identifier designating the repeater.

The client gets the modified resource identifier from the reflector and makes a request for the particular resource from the repeater designated in the modified resource identifier.

When the repeater gets the client's request, it responds by returning the requested resource to the client. If the repeater has a local copy of the resource then it returns that copy, otherwise it forwards the request to the origin server to get the resource, and saves a local copy of the resource in order to serve subsequent requests.

The selection by the reflector of an appropriate repeater to handle the request can be done in a number of ways. In the preferred embodiment, it is done by first pre-partitioning the network into "cost groups" and then determining which cost group the client is in. Next, from a plurality of repeaters in the network, a set of repeaters is selected, the members of the set having a low cost relative to the cost group which the client is in. In order to determine the lowest cost, a table is maintained and regularly updated to define the cost between each group and each repeater. Then one member of the set is selected, preferably randomly, as the best repeater.

If the particular requested resource itself can contain identifiers of other resources, then the resource may be rewritten (before being provided to the client). In particular, the resource is rewritten to replace at least some of the resource identifiers contained therein with modified resource identifiers designating a repeater instead of the origin server. As a consequence of this rewriting process, when the client requests other resources based on identifiers in the particular requested resource, the client will make those requests directly to the selected repeater, bypassing the reflector and origin server entirely.

Resource rewriting must be performed by reflectors. It may also be performed by repeaters, in the situation where repeaters "peer" with one another and make copies of resources which include rewritten resource identifiers that designate a repeater.

In a preferred embodiment, the network is the Internet and the resource identifier is a uniform resource locator (URL) for designating resources on the Internet, and the modified resource identifier is a URL designating the repeater and indicating the origin server (as described in step B3 below), and the modified resource identifier is provided to the client using a REDIRECT message. Note, only when the reflector is "reflecting" a request is the modified resource identifier provided using a REDIRECT message.

4

In another aspect, this invention is a computer network comprising a plurality of origin servers, at least some of the origin servers having reflectors associated therewith, and a plurality of repeaters.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects and advantages of the invention will be apparent upon consideration of the following detailed description, taken in conjunction with the accompanying drawings, in which the reference characters refer to like parts throughout and in which:

FIG. 1 depicts a portion of a network environment according to the present invention; and

FIGS. 2-6 are flow charts of the operation of the present invention.

DETAILED DESCRIPTION OF THE PRESENTLY PREFERRED EXEMPLARY EMBODIMENTS

Overview

FIG. 1 shows a portion of a network environment **100** according to the present invention, wherein a mechanism (reflector **108**, described in detail below) at a server herein (origin server **102**) maintains and keeps track of a number of partially replicated servers or repeaters **104a**, **104b**, and **104c**. Each repeater **104a**, **104b**, and **104c** replicates some or all of the information available on the origin server **102** as well as information available on other origin servers in the network **100**. Reflector **108** is connected to a particular repeater known as its "contact" repeater ("Repeater B" **104b** in the system depicted in FIG. 1). Preferably each reflector maintains a connection with a single repeater known as its contact, and each repeater maintains a connection with a special repeater known as its master repeater (e.g., repeater **104m** for repeaters **104a**, **104b** and **104c** in FIG. 1).

Thus, a repeater can be considered as a dedicated proxy server that maintains a partial or sparse mirror of the origin server **102**, by implementing a distributed coherent cache of the origin server. A repeater may maintain a (partial) mirror of more than one origin server. In some embodiments, the network **100** is the Internet and repeaters mirror selected resources provided by origin servers in response to clients' HTTP (hypertext transfer protocol) and FTP (file transfer protocol) requests.

A client **106** connects, via the network **100**, to origin server **102** and possibly to one or more repeaters **104a** etc.

Origin server **102** is a server at which resources originate. More generally, the origin server **102** is any process or collection of processes that provide resources in response to requests from a client **106**. Origin server **102** can be any off-the-shelf Web server. In a preferred embodiment, origin server **102** is typically a Web server such as the Apache server or Netscape Communications Corporation's Enterprise™ server.

Client **106** is a processor requesting resources from origin server **102** on behalf of an end user. The client **106** is typically a user agent (e.g., a Web browser such as Netscape Communications Corporation's Navigators™) or a proxy for a user agent. Components other than the reflector **108** and the repeaters **104a**, **104b**, etc., may be implemented using commonly available software programs. In particular, this invention works with any HTTP client (e.g., a Web browser), proxy cache, and Web server. In addition, the reflector **108** might be fully integrated into the data server **112** (for instance, in a Web Server). These components might be loosely integrated based on the use of extension mechanisms (such as so-called add-in

US 8,478,903 B2

5

modules) or tightly integrated by modifying the service component specifically to support the repeaters.

Resources originating at the origin server **102** may be static or dynamic. That is, the resources may be fixed or they may be created by the origin server **102** specifically in response to a request. Note that the terms “static” and “dynamic” are relative, since a static resource may change at some regular, albeit long, interval.

Resource requests from the client **106** to the origin server **102** are intercepted by reflector **108** which for a given request either forwards the request on to the origin server **102** or conditionally reflects it to some repeater **104a**, **104b**, etc. in the network **100**. That is, depending on the nature of the request by the client **106** to the origin server **102**, the reflector **108** either serves the request locally (at the origin server **102**), or selects one of the repeaters (preferably the best repeater for the job) and reflects the request to the selected repeater. In other words, the reflector **108** causes requests for resources from origin server **102**, made by client **106**, to be either served locally by the origin server **102** or transparently reflected to the best repeater **104a**, **104b**, etc. The notion of a best repeater and the manner in which the best repeater is selected are described in detail below.

Repeaters **104a**, **104b**, etc. are intermediate processors used to service client requests thereby improving performance and reducing costs in the manner described herein. Within repeaters **104a**, **104b**, etc., are any processes or collections of processes that deliver resources to the client **106** on behalf of the origin server **102**. A repeater may include a repeater cache **110**, used to avoid unnecessary transactions with the origin server **102**.

The reflector **108** is a mechanism, preferably a software program, that intercepts requests that would normally be sent directly to the origin server **102**. While shown in the drawings as separate components, the reflector **108** and the origin server **102** are typically co-located, e.g., on a particular system such as data server **112**. (As discussed below, the reflector **108** may even be a “plug in” module that becomes part of the origin server **102**.)

FIG. 1 shows only a part of a network **100** according to this invention. A complete operating network consists of any number of clients, repeaters, reflectors, and origin servers. Reflectors communicate with the repeater network, and repeaters in the network communicate with one another.

Uniform Resource Locators

Each location in a computer network has an address which can generally be specified as a series of names or numbers. In order to access information, an address for that information must be known. For example, on the World Wide Web (“the Web”) which is a subset of the Internet, the manner in which information address locations are provided has been standardized into Uniform Resource Locators (URLs). URLs specify the location of resources (information, data files, etc.) on the network.

The notion of URLs becomes even more useful when hypertext documents are used. A hypertext document is one which includes, within the document itself, links (pointers or references) to the document itself or to other documents. For example, in an on-line legal research system, each case may be presented as a hypertext document. When other cases are cited, links to those cases can be provided. In this way, when a person is reading a case, they can follow cite links to read the appropriate parts of cited cases.

In the case of the Internet in general and the World Wide Web specifically, documents can be created using a standardized form known as the Hypertext Markup Language (HTML). In HTML, a document consists of data (text,

6

images, sounds, and the like), including links to other sections of the same document or to other documents. The links are generally provided as URLs, and can be in relative or absolute form. Relative URLs simply omit the parts of the URL which are the same as for the document including the link, such as the address of the document (when linking to the same document), etc. In general, a browser program will fill in missing parts of a URL using the corresponding parts from the current document, thereby forming a fully formed URL including a fully qualified domain name, etc.

A hypertext document may contain any number of links to other documents, and each of those other documents may be on a different server in a different part of the world. For example, a document may contain links to documents in Russia, Africa, China and Australia. A user viewing that document at a particular client can follow any of the links transparently (i.e., without knowing where the document being linked to actually resides). Accordingly, the cost (in terms of time or money or resource allocation) of following one link versus another may be quite significant.

URLs generally have the following form (defined in detail in T. Berners-Lee et al, *Uniform Resource Locators* (URL), Network Working Group, Request for Comments: 1738, Category: Standards Track, December 1994, located at “http://ds.internic.net/rfc/rfc1738.txt”, which is hereby incorporated herein by reference):

scheme://host[:port]/url-path

where “scheme” can be a symbol such as “file” (for a file on the local system), “ftp” (for a file on an anonymous FTP file server), “http” (for a file on a file on a Web server), and “telnet” (for a connection to a Telnet-based service). Other schemes, can also be used and new schemes are added every now and then. The port number is optional, the system substituting a default port number (depending on the scheme) if none is provided. The “host” field maps to a particular network address for a particular computer. The “url-path” is relative to the computer specified in the “host” field. A url-path is typically, but not necessarily, the pathname of a file in a web server directory.

For example, the following is a URL identifying a file “F” in the path “A/B/C” on a computer at “www.uspto.gov”:

http://www.uspto.gov/A/B/C/F

In order to access the file “F” (the resource) specified by the above URL, a program (e.g., a browser) running on a user’s computer (i.e., a client computer) would have to first locate the computer (i.e., a server computer) specified by the host name. I.e., the program would have to locate the server “www.uspto.gov”. To do this, it would access a Domain Name Server (DNS), providing the DNS with the host name (“www.uspto.gov”). The DNS acts as a kind of centralized directory for resolving addresses from names. If the DNS determines that there is a (remote server) computer corresponding to the name “www.uspto.gov”, it will provide the program with an actual computer network address for that server computer. On the Internet this is called an Internet Protocol (or IP) address and it has the form “123.345.456.678”. The program on the user’s (client) computer would then use the actual address to access the remote (server) computer.

The program opens a connection to the HTTP server. (Web server) on the remote computer “www.uspto.gov” and uses the connection to send a request message to the remote computer (using the HTTP scheme). The message is typically an HTTP GET request which includes the url-path of the requested resource, “A/B/C/F”. The HTTP server receives the

US 8,478,903 B2

7

request and uses it to access the resource specified by the url-path "A/B/C/F". The server returns the resource over the same connection.

Thus, conventionally HTTP client requests for Web resources at an origin server **102** are processed as follows (see FIG. 2) (This is a description of the process when no reflector **108** is installed.):

- A1. A browser (e.g., Netscape's Navigator) at the client receives a resource identifier (i.e., a URL) from a user.
- A2. The browser extracts the host (origin server) name from the resource identifier, and uses a domain name server (DNS) to look up the network (IP) address of the corresponding server. The browser also extracts a port number, if one is present, or uses a default port number (the default port number for http requests is 80).
- A3. The browser uses the server's network address and port number to establish a connection between the client **106** and the host or origin server **102**.
- A4. The client **106** then sends a (GET) request over the connection identifying the requested resource.
- A5. The origin server **102** receives the request and
- A6. locates or composes the corresponding resource.
- A7. The origin server **102** then sends back to the client **106** a reply containing the requested resource (or some form of error indicator if the resource is unavailable). The reply is sent to the client over the same connection as that on which the request was received from the client.
- A8. The client **106** receives the reply from the origin server **102**.

There are many variations of this basic model. For example, in one variation, instead of providing the client with the resource, the origin server can tell the client to re-request the resource by another name. To do so, in A7 the server **102** sends back to the client **106** a reply called a "REDIRECT" which contains a new URL indicating the other name. The client **106** then repeats the entire sequence, normally without any user intervention, this time requesting the resource identified by the new URL.

System Operation

In this invention reflector **108** effectively takes the place of an ordinary Web server or origin server **102**. The reflector **108** does this by taking over the origin server's IP address and port number. In this way, when a client tries to connect to the origin server **102**, it will actually connect to the reflector **108**. The original Web server (or origin server **102**) must then accept requests at a different network (IP) address, or at the same IP address but on a different port number. Thus, using this invention, the server referred to in A3-A7 above is actually a reflector **108**.

Note that it is also possible to leave the origin server's network address as it is and to let the reflector run at a different address or on a different port. In this way the reflector does not intercept requests sent to the origin server, but can still be sent requests addressed specifically to the reflector. Thus the system can be tested and configured without interrupting its normal operation.

The reflector **108** supports the processing as follows (see FIG. 3): upon receipt of a request,

- B1 The reflector **108** analyzes the request to determine whether or not to reflect the request. To do this, first the reflector determines whether the sender (client **106**) is a browser or a repeater. Requests issued by repeaters must be served locally by the origin server **102**. This determination can be made by looking up the network (IP) address of the sender in a list of known repeater network (IP) addresses. Alternatively, this determination could be made by attaching information to a request to indicate

8

that the request is from a specific repeater, or repeaters can request resources from a special port other than the one used for ordinary clients.

- B2 If the request is not from a repeater, the reflector looks up the requested resource in a table (called the "rule base") to determine whether the resource requested is "repeatable". Based on this determination, the reflector either reflects the request (B3, described below) or serves the request locally (B4, described below).

The rule base is a list of regular expressions and associated attributes. (Regular expressions are well-known in the field of computer science. A small bibliography of their use is found in Aho, et al., "Compilers, Principles, techniques and tools", Addison-Wesley, 1986, pp. 157-158.) The resource identifier (URL) for a given request is looked up in the rule base by matching it sequentially with each regular expression. The first match identifies the attributes for the resource, namely repeatable or local. If there is no match in the rule base, a default attribute is used. Each reflector has its own rule base, which is manually configured by the reflector operator.

- B3. To reflect a request, (to serve a request locally go to B4), as shown in FIG. 4, the reflector determines (B3-1) the best repeater to reflect the request to, as described in detail below. The reflector then creates (B3-2) a new resource identifier (URL) (using the requested URL and the best repeater) that identifies the same resource at the selected repeater.

It is necessary that the reflection step create a single URL containing the URL of the original resource, as well as the identity of the selected repeater. A special form of URL is created to provide this information. This is done by creating a new URL as follows:

- D1. Given a repeater name, scheme, origin server name and path, create a new URL. If the scheme is "http", the preferred embodiment uses the following format:
http://<repeater>/<server>/<path>

If the form used is other than "http", the preferred embodiment uses the following format:

```
http://<repeater>/<server>@proxy=<scheme>@/  
<path>
```

The reflector can also attach a MIME type to the request, to cause the repeater to provide that MIME type with the result. This is useful because many protocols (such as FTP) do not provide a way to attach a MIME type to a resource. The format is

```
http://<repeater>/<server>@proxy=<scheme>:  
<type>@/<Path>
```

This URL is interpreted when received by the repeater.

The reflector then sends (B3-3) a REDIRECT reply containing this new URL to the requesting client. The HTTP REDIRECT command allows the reflector to send the browser a single URL to retry the request.

- B4. To serve a request locally, the request is sent by the reflector to ("forwarded to") the origin server **102**. In this mode, the reflector acts as a reverse proxy server. The origin server **102** processes the request in the normal manner (A5-A7). The reflector then obtains the origin server's reply to the request which it inspects to determine if the requested resource is an HTML document, i.e., whether the requested resource is one which itself contains resource identifiers.

- B5. If the resource is an HTML document then the reflector rewrites the HTML document by modifying resource identifiers (URLs) within it, as described below. The resource, possibly as modified by rewriting, is then returned in a reply to the requesting client **106**.

US 8,478,903 B2

9

If the requesting client is a repeater, the reflector may temporarily disable any cache-control modifiers which the origin server attached to the reply. These disabled cache-control modifiers are later re-enabled when the content is served from the repeater. This mechanism makes it possible for the origin server to prevent resources from being cached at normal proxy caches, without affecting the behavior of the cache at the repeater.

B6. Whether the request is reflected or handled locally, details about the transaction, such as the current time, the address of the requester, the URL requested, and the type of response generated, are written by the reflector to a local log file.

By using a rule base (B2), it is possible to selectively reflect resources. There are a number of reasons that certain particular resources cannot be effectively repeated (and therefore should not be reflected), for instance:

the resource is composed uniquely for each request;
the resource relies on a so-called cookie (browsers will not send cookies to repeaters with different domain names);
the resource is actually a program (such as a Java applet) that will run on the client and that wishes to connect to a service (Java requires that the service be running on the same machine that provided the applet).

In addition, the reflector **108** can be configured so that requests from certain network addresses (e.g., requests from clients on the same local area network as the reflector itself) are never reflected. Also, the reflector may choose not to reflect requests because the reflector is exceeding its committed aggregate information rate, as described below.

A request which is reflected is automatically mirrored at the repeater when the repeater receives and processes the request.

The combination of the reflection process described here and the caching process described below effectively creates a system in which repeatable resources are migrated to and mirrored at the selected reflector, while non-repeatable resources are not mirrored.

Alternate Approach

Placing the origin server name in the reflected URL is generally a good strategy, but it may be considered undesirable for aesthetic or (in the case, e.g., of cookies) certain technical reasons.

It is possible to avoid the need for placing both the repeater name and the server name in the URL. Instead, a "family" of names may be created for a given origin server, each name identifying one of the repeaters used by that server.

For instance, if `www.example.com` is the origin server, names for three repeaters might be created:

`wr1.example.com`
`wr2.example.com`
`wr3.example.com`

The name "`wr1.example.com`" would be an alias for repeater 1, which might also be known by other names such as "`wr1.anotherExample.com`" and "`wr1.example.edu`".

If the repeater can determine by which name it was addressed, it can use this information (along with a table that associates repeater alias names with origin server names) to determine which origin server is being addressed. For instance, if repeater 1 is addressed as `wr1.example.com`, then the origin server is "`www.example.com`"; if it is addressed as "`wr1.anotherExample.com`", then the origin server is "`www.anotherExample.com`".

The repeater can use two mechanisms to determine by which alias it is addressed:

1. Each alias can be associated with a different IP address. Unfortunately, this solution does not scale well, as IP

10

addresses are currently scarce, and the number of IP addresses required grows as the product of origin servers and repeaters.

2. The repeater can attempt to determine the alias name used by inspecting the "host:" tag in the HTTP header of the request. Unfortunately, some old browsers still in use do not attach the "host:" tag to a request.

Reflectors would need to identify such browsers (the browser identity is a part of each request) and avoid this form of reflection.

How a Repeater Handles a Request

When a browser receives a REDIRECT response (as produced in B3), it reissues a request for the resource using the new resource identifier (URL) (A1-A5). Because the new identifier refers to a repeater instead of the origin server, the browser now sends a request for the resource to the repeater which processes a request as follows, with reference to FIG. 5:

- C1. First the repeater analyzes the request to determine the network address of the requesting client and the path of the resource requested. Included in the path is an origin server name (as described above with reference to B3).
- C2. The repeater uses an internal table to verify that the origin server belongs to a known "subscriber". A subscriber is an entity (e.g., a company) that publishes resources (e.g., files) via one or more origin servers. When the entity subscribes, it is permitted to utilize the repeater network. The subscriber tables described below include the information that is used to link reflectors to subscribers.

If the request is not for a resource from a known subscriber, the request is rejected. To reject a request, the repeater returns a reply indicating that the requested resource does not exist.

- C3. The repeater then determines whether the requested resource is cached locally. If the requested resource is in the repeater's cache it is retrieved.

On the other hand, if a valid copy of the requested resource is not in the repeater's cache, the repeater modifies the incoming URL, creating a request that it issues directly to the originating reflector which processes it (as in B1-B6). Because this request to the originating reflector is from a repeater, the reflector always returns the requested resource rather than reflecting the request. (Recall that reflectors always handle requests from repeaters locally.) If the repeater obtained the resource from the origin server, the repeater then caches the resource locally.

If a resource is not cached locally, the cache can query its "peer caches" to see if one of them contains the resource, before or at the same time as requesting the resource from the reflector/origin server. If a peer cache responds positively in a limited period of time (preferably a small fraction of a second), the resource will be retrieved from the peer cache.

- C4. The repeater then constructs a reply including the requested resource (which was retrieved from the cache or from the origin server) and sends that reply to the requesting client.

- C5. Details about the transaction, such as the associated reflector, the current time, the address of the requester, the URL requested, and the type of response generated, are written to a local log file at the repeater.

Note that the bottom row of FIG. 2 refers to an origin server, or a reflector, or a repeater, depending on what the URL in step A1 identifies.

Selecting the Best Repeater

If the reflector **108** determines that it will reflect the request, it must then select the best repeater to handle that

US 8,478,903 B2

11

request (as referred to in step B3-1). This selection is performed by the Best Repeater Selector (BRS) mechanism described here.

The goal of the BRS is to select, quickly and heuristically, an appropriate repeater for a given client given only the network address of the client. An appropriate repeater is one which is not too heavily loaded and which is not too far from the client in terms of some measure of network distance. The mechanism used here relies on specific, compact, pre-computed data to make a fast decision. Other, dynamic solutions can also be used to select an appropriate repeater.

The BRS relies on three pre-computed tables, namely the Group Reduction Table, the Link Cost Table, and the Load Table. These three tables (described below) are computed off-line and downloaded to each reflector by its contact in the repeater network.

The Group Reduction Table places every network address into a group, with the goal that addresses in a group share relative costs, so that they would have the same best repeater under varying conditions (i.e., the BRS is invariant over the members of the group).

The Link Cost Table is a two dimensional matrix which specifies the current cost between each repeater and each group. Initially, the link cost between a repeater and a group is defined as the "normalized link cost" between the repeater and the group, as defined below. Over time, the table will be updated with measurements which more accurately reflect the relative cost of transmitting a file between the repeater and a member of the group. The format of the Link Cost Table is <Group ID><Group ID><link cost>, where the Group ID's are given as AS numbers.

The Load Table is a one dimensional table which identifies the current load at each repeater. Because repeaters may have different capacities, the load is a value that represents the ability of a given repeater to accept additional work. Each repeater sends its current load to a central master repeater at regular intervals, preferably at least approximately once a minute. The master repeater broadcasts the Load Table to each reflector in the network, via the contact repeater.

A reflector is provided entries in the Load Table only for repeaters which it is assigned to use. The assignment of repeaters to reflectors is performed centrally by a repeater network operator at the master repeater. This assignment makes it possible to modify the service level of a given reflector. For instance, a very active reflector may use many repeaters, whereas a relatively inactive reflector may use few repeaters.

Tables may also be configured to provide selective repeater service to subscribers in other ways, e.g., for their clients in specific geographic regions, such as Europe or Asia.

Measuring Load

In the presently preferred embodiments, repeater load is measured in two dimensions, namely

1. requests received by the repeater per time interval (RRPT), and
2. bytes sent by the repeater per time interval (BSPT).

For each of these dimensions, a maximum capacity setting is set. The maximum capacity indicates the point at which the repeater is considered to be fully loaded. A higher RRPT capacity generally indicates a faster processor, whereas a higher BSPT capacity generally indicates a wider network pipe. This form of load measurement assumes that a given server is dedicated to the task of repeating.

Each repeater regularly calculates its current RRPT and BSPT, by accumulating the number of requests received and bytes sent over a short time interval. These measurements are used to determine the repeater's load in each of these dimen-

12

sions. If a repeater's load exceeds its configured capacity, an alarm message is sent to the repeater network administrator.

The two current load components are combined into a single value indicating overall current load. Similarly, the two maximum capacity components are combined into a single value indicating overall maximum capacity. The components are combined as follows:

$$\text{current-load} = B \times \text{current RRPT} + (1-B) \times \text{current BSPT}$$

$$\text{max-load} = B \times \text{max RRPT} + (1-B) \times \text{max BSPT}$$

The factor B, a value between 0 and 1, allows the relative weights of RRPT and BSPT to be adjusted, which favors consideration of either processing power or bandwidth.

The overall current load and overall maximum capacity values are periodically sent from each repeater to the master repeater, where they are aggregated in the Load Table, a table summarizing the overall load for all repeaters. Changes in the Load Table are distributed automatically to each reflector.

While the preferred embodiment uses a two-dimensional measure of repeater load, any other measure of load can be used.

Combining Link Costs and Load

The BRS computes the cost of servicing a given client from each eligible repeater. The cost is computed by combining the available capacity of the candidate repeater with the cost of the link between that repeater and the client. The link cost is computed by simply looking it up in the Link Cost table.

The cost is determined using the following formula:

$$\text{threshold} = K \times \text{max-load}$$

$$\text{capacity} = \text{max}(\text{max-load} - \text{current-load}, e)$$

$$\text{capacity} = \text{min}(\text{capacity}, \text{threshold})$$

$$\text{cost} = \text{link-cost} \times \text{threshold} / \text{capacity}$$

In this formula, e is a very small number (epsilon) and K is a tuning factor initial set to 0.5. This formula causes the cost to a given repeater to be increased, at a rate defined by K, if its capacity falls below a configurable threshold.

Given the cost of each candidate repeater, the BRS selects all repeaters within a delta factor of the best score. From this set, the result is selected at random.

The delta factor prevents the BRS from repeatedly selecting a single repeater when scores are similar. It is generally required because available information about load and link costs loses accuracy over time. This factor is tunable.

Best Repeater Selector (BRS)

The BRS operates as follows, with reference to FIG. 6:

Given a client network address and the three tables described above:

E1. Determine which group the client is in using the Group Reduction Table.

E2. For each repeater in the Link Cost Table and Load Table, determine that repeater's combined cost as follows:

E2a. Determine the maximum and current load on the repeater (using the Load Table).

E2b. Determine the link cost between the repeater and the client's group (using the Link Cost Table).

E2c. Determine the combined cost as described above.

E3. Select a small set of repeaters with the lowest cost.

E4. Select a random member from the set.

Preferably the results of the BRS processing are maintained in a local cache at the reflector 108. Thus, if the best repeater has recently been determined for a given client (i.e., for a given network address), that best repeater can be reused

US 8,478,903 B2

13

quickly without being re-determined. Since the calculation described above is based on statically, pre-computed tables, if the tables have not changed then there is no need to re-determine the best repeater.

Determining the Group Reduction and Link Cost Tables

The Group Reduction Table and Link Cost Table used in BRS processing are created and regularly updated by an independent procedure referred to herein as NetMap. The NetMap procedure is run by executing several phases (described below) as needed.

The term Group is used here to refers to an IP "address group".

The term Repeater Group refers to a Group that contains the IP address of a repeater.

The term link cost refers to a statically determined cost for transmitting data between two Groups. In a presently preferred implementation, this is the minimum of the sums of the costs of the links along each path between them. The link costs of primary concern here are link costs between a Group and a Repeater Group.

The term relative link cost refers to the link cost relative to other link costs for the same Group which is calculated by subtracting the minimum link cost from a Group to any Repeater Group from each of its link costs to a Repeater Group. The term Cost Set refers to a set of Groups that are equivalent in regard to Best Repeater Selection. That is, given the information available, the same repeater would be selected for any of them.

The NetMap procedure first processes input files to create an internal database called the Group Registry. These input files describe groups, the IP addresses within groups, and links between groups, and come a variety of sources, including publicly available Internet Routing Registry (IRR) databases, BGP router tables, and probe services that are located at various points around the Internet and use publicly available tools (such as "traceroute") to sample data paths. Once this processing is complete, the Group Registry contains essential information used for further processing, namely (1) the identity of each group, (2) the set of IP addresses in a given group, (3) the presence of links between groups indicating paths over which information may travel, and (4) the cost of sending data over a given link.

The following processes are then performed on the Group Registry file.

Calculate Repeater Group Link Costs

The NetMap procedure calculates a "link cost" for transmission of data between each Repeater Group and each Group in the Group Registry. This overall link cost is defined as the minimum cost of any path between the two groups, where the cost of a path is equal to the sum of the costs of the individual links in the path. The link cost algorithm presented below is essentially the same as algorithm #562 from ACM journal Transactions on Mathematical Software: "Shortest Path From a Specific Node to All Other Nodes in a Network" by U. Pape, ACM TOMS 6 (1980) pp. 450-455, <http://www.netlib.org/toms/562>.

In this processing, the term Repeater Group refers to a Group that contains the IP address of a repeater. A group is a neighbor of another group if the Group Registry indicates that there is a link between the two groups.

For each target Repeater Group T:

Initialize the link cost between T and itself to zero.

Initialize the link cost between T and every other Group to infinity.

Create a list L that will contain Groups that are equidistant from the target Repeater Group T.

14

Initialize the list L to contain just the target Repeater Group T itself.

While the list L is not empty:

Create an empty list L' of neighbors of members of the list L.

For each Group G in the list L:

For each Group N that is a neighbor of G:

Let cost refer to the sum of the link cost between T and G, and the link cost between G and N. The cost between T and G was determined in the previous pass of the algorithm; the link cost between G and N is from the Group Registry.

If cost is less than the link cost between T and N:

Set the link cost between T and N to cost.

Add N to L' if it is not already on it.

Set L to L'.

Calculate Cost Sets

A Cost Set is a set of Groups that are equivalent with respect to Best Repeater Selection. That is, given the information available, the same repeater would be selected for any of them.

The "cost profile" of a Group G is defined herein as the set of costs between G and each Repeater. Two cost profiles are said to be equivalent if the values in one profile differ from the corresponding values in the other profile by a constant amount.

Once a client Group is known, the Best Repeater Selection algorithm relies on the cost profile for information about the Group. If two cost profiles are equivalent, the BRS algorithm would select the same repeater given either profile.

A Cost Set is then a set of groups that have equivalent cost profiles.

The effectiveness of this method can be seen, for example, in the case where all paths to a Repeater from some Group A pass through some other Group B. The two Groups have equivalent cost profiles (and are therefore in the same Cost Set) since whatever Repeater is best for Group A is also going to be best for Group B, regardless of what path is taken between the two Groups.

By normalizing cost profiles, equivalent cost profiles can be made identical. A normalized cost profile is a cost profile in which the minimum cost has the value zero. A normalized cost profile is computed by finding the minimum cost in the profile, and subtracting that value from each cost in the profile.

Cost Sets are then computed using the following algorithm:

For each Group G:

Calculate the normalized cost profile for G

Look for a Cost Set with the same normalized cost profile.

If such a set is found, add G to the existing Cost Set; otherwise, create a new Cost Set with the calculated normalized cost profile, containing only G.

The algorithm for finding Cost Sets employs a hash table to reduce the time necessary to determine whether the desired Cost Set already exists. The hash table uses a hash value computed from cost profile of G.

Each Cost Set is then numbered with a unique Cost Set Index number. Cost Sets are then used in a straightforward manner to generate the Link Cost Table, which gives the cost from each Cost Set to each Repeater.

As described below, the Group Reduction Table maps every IP address to one of these Cost Sets.

Build IP Map

The IP Map is a sorted list of records which map IP address ranges to Link Cost Table keys. The format of the IP map is:

US 8,478,903 B2

15

<base IP address><max IP address><Link Cost Table key>

where IP addresses are presently represented by 32-bit integers. The entries are sorted by descending base address, and by ascending maximum address among equal base addresses, and by ascending Link Cost Table key among equal base addresses and maximum addresses. Note that ranges may overlap.

The NetMap procedure generates an intermediate IP map containing a map between IP address ranges and Cost Set numbers as follows:

For each Cost Set S:

For each Group G in S:

For each IP address range in G:

Add a triple (low address, high address, Cost Set number of S) to the IP map.

The IP map file is then sorted by descending base address, and by ascending maximum address among equal base addresses, and by ascending Cost Set number among equal base addresses and maximum addresses. The sort order for the base address and maximum address minimizes the time to build the Group Reduction Table and produces the proper results for overlapping entries.

Finally, the NetMap procedure creates the Group Reduction Table by processing the sorted IP map. The Group Reduction Table maps IP addresses (specified by ranges) into Cost Set numbers. Special processing of the IP map file is required in order to detect overlapping address ranges, and to merge adjacent address ranges in order to minimize the size of the Group Reduction Table.

An ordered list of address range segments is maintained, each segment consisting of a base address B and a Cost Set number N, sorted by base address B. (The maximum address of a segment is the base address of the next segment minus one.)

The following algorithm is used:

Initialize the list with the elements [-infinity, NOGROUP], [+infinity, NOGROUP].

For each entry in the IP map, in sorted order, consisting of (b, m, s),

Insert (b, m, s) in the list (recall that IP map entries are of the form (low address, high address Cost Set number of S))

For each reserved LAN address range (b, m):

Insert (b, m, LOCAL) in the list.

For each Repeater at address a:

Insert (a, a, REPEATER) in the list

For each segment S in the ordered list:

Merge S with following segments with the same Cost Set

Create a Group Reduction Table entry with base address from the base address of S, max address=next segment's base-1, group ID=Cost Set number of S.

A reserved LAN address range is an address range reserved for use by LANs which should not appear as a global Internet address. LOCAL is a special Cost Set index different from all others, indicating that the range maps to a client which should never be reflected. REPEATER is a special Cost Set index different from all others, indicating that the address range maps to a repeater. NOGROUP is a special Cost Set index different from all others, indicating that this range of addresses has no known mapping.

Given (B, M, N), insert an entry in the ordered address list as follows:

Find the last segment (AB, AN) for which AB is less than or equal to B.

16

If AB is less than B, insert a new segment (B, N) after (AB, AN).

Find the last segment (YB, YN) for which YB is less than or equal to M.

Replace by (XB, N) any segment (XB, NOGROUP) for which XB is greater than B and less than YB.

If YN is not N, and either YN is NOGROUP or YB is less than or equal to B,

Let (ZB, ZN) be the segment following (YB, YN).

If M+1 is less than ZB, insert a new segment (M+1, YN) before (ZB, ZN).

Replace (YB, YN) by (YB, N).

Rewriting HTML Resources

As explained above with reference to FIG. 3 (B5), when a reflector or repeater serves a resource which itself includes resource identifiers (e.g., a HTML resource), that resource is modified (rewritten) to pre-reflect resource identifiers (URLs) of repeatable resources that appear in the resource. Rewriting ensures that when a browser requests repeatable resources identified by the requested resource, it gets them from a repeater without going back to the origin server, but when it requests non-repeatable resources identified by the requested resource, it will go directly to the origin server. Without this optimization, the browser would either make all requests at the origin server (increasing traffic at the origin server and necessitating far more redirections from the origin server), or it would make all requests at the repeater (causing the repeater to redundantly request and copy resources which could not be cached, increasing the overhead of serving such resources).

Rewriting requires that a repeater has been selected (as described above with reference to the Best Repeater Selector). Rewriting uses a so-called BASE directive. The BASE directive lets the HTML identify a different base server. (The base address is normally the address of the HTML resource.)

Rewriting is performed as follows:

F1. A BASE directive is added at the beginning of the HTML resource, or modified where necessary. Normally, a browser interprets relative URLs as being relative to the default base address, namely, the URL of the HTML resource (page) in which they are encountered. The BASE address added specifies the resource at the reflector which originally served the resource. This means that unprocessed relative URLs (such as those generated by Javascript™ programs) will be interpreted as relative to the reflector. Without this BASE address, browsers would combine relative addresses with repeater names to create URLs which were not in the form required by repeaters (as described above in step D1).

F2. The rewriter identifies directives, such as embedded images and anchors, containing URLs. If the rewriter is running in a reflector, it must parse the HTML file to identify these directives. If it is running in a repeater, the rewriter may have access to pre-computed information that identifies the location of each URL (placed in the HTML file in step F4).

F3. For each URL encountered in the resource to be rewritten, the rewriter must determine whether the URL is repeatable (as in steps B1-B2). If the URL is not repeatable, it is not modified. On the other hand, if the URL is repeatable, it is modified to refer to the selected repeater.

F4. After all URLs have been identified and modified, if the resource is being served to a repeater, a table is appended at the beginning of the resource that identifies the location and content of each URL encountered in the

US 8,478,903 B2

17

resource. (This step is an optimization which eliminates the need for parsing HTML resources at the repeater.)

F5. Once all changes have been identified, a new length is computed for the resource (page). The length is inserted in the HTTP header prior to serving the resource.

An extension of HTML, known as XML, is currently being developed. The process of rewriting URLs will be similar for XML, with some differences in the mechanism that parses the resource and identifies embedded URLs.

Handling Non-HTTP Protocols

This invention makes it possible to reflect references to resources that are served by protocols other than HTTP, for instance, the File Transfer Protocol (FTP) and audio/video stream protocols. However, many protocols do not provide the ability to redirect requests. It is, however, possible to redirect references before requests are actually made by rewriting URLs embedded in HTML pages. The following modifications to the above algorithms are used to support this capability.

In F4, the rewriter rewrites URLs for servers if those servers appear in a configurable table of cooperating origin server or so-called co-servers. The reflector operator can define this table to include FTP servers and other servers. A rewritten URL that refers to a non-HTTP resource takes the form:

```
http://<repeater>/<origin server>@<proxy>-<scheme>[:<type>]@/resource
```

where <scheme> is a supported protocol name such as "ftp". This URL format is an alternative to the form shown in B3.

In C3, the repeater looks for a protocol embedded in the arriving request. If a protocol is present and the requested resource is not already cached, the repeater uses the selected protocol instead of the default HTTP protocol to request the resource when serving it and storing it in the cache.

System Configuration and Management

In addition to the processing described above, the repeater network requires various mechanisms for system configuration and network management. Some of these mechanisms are described here.

Reflectors allow their operators to synchronize repeater caches by performing publishing operations. The process of keeping repeater caches synchronized is described below. Publishing indicates that a resource or collection of resources has changed.

Repeaters and reflectors participate in various types of log processing. The results of logs collected at repeaters are collected and merged with logs collected at reflectors, as described below.

Adding Subscribers to the Repeater Network

When a new subscriber is added to the network, information about the subscriber is entered in a Subscriber Table at the master repeater and propagated to all repeaters in the network. This information includes the Committed Aggregate Information Rate (CAIR) for servers belonging to the subscriber, and a list of the repeaters that may be used by servers belonging to the subscriber.

Adding Reflectors to the Repeater Network

When a new reflector is added to the network, it simply connects to and announces itself to a contact repeater, preferably using a securely encrypted certificate including the repeater's subscriber identifier.

The contact repeater determines whether the reflector network address is permitted for this subscriber. If it is, the contact repeater accepts the connection and updates the reflector with all necessary tables (using version numbers to determine which tables are out of date).

18

The reflector processes requests during this time, but is not "enabled" (allowed to reflect requests) until all of its tables are current.

Keeping Repeater Caches Synchronized

Repeaters caches are coherent, in the sense that when a change to a resource is identified by a reflector, all repeater caches are notified, and accept the change in a single transaction.

Only the identifier of the changed resource (and not the entire resource) is transmitted to the repeaters; the identifier is used to effectively invalidate the corresponding cached resource at the repeater. This process is far more efficient than broadcasting the content of the changed resource to each repeater.

A repeater will load the newly modified resource the next time it is requested.

A resource change is identified at the reflector either manually by the operator, or through a script when files are installed on the server, or automatically through a change detection mechanism (e.g., a separate process that checks regularly for changes).

A resource change causes the reflector to send an "invalidate" message to its contact repeater, which forwards the message to the master repeater. The invalidate message contains a list of resource identifiers (or regular expressions identifying patterns of resource identifiers) that have changed. (Regular expressions are used to invalidate a directory or an entire server.) The repeater network uses a two-phase commit process to ensure that all repeaters correctly invalidate a given resource.

The invalidation process operates as follows:

The master broadcasts a "phase 1" invalidation request to all repeaters indicating the resources and regular expressions describing sets of resources to be invalidated.

When each repeater receives the phase 1 message, it first places the resource identifiers or regular expressions into a list of resource identifiers pending invalidation.

Any resource requested (in C3) that is in the pending invalidation list may not be served from the cache. This prevents the cache from requesting the resource from a peer cache which may not have received an invalidation notice. Were it to request a resource in this manner, it might replace the newly invalidated resource by the same, now stale, data.

The repeater then compares the resource identifier of each resource in its cache against the resource identifiers and regular expressions in the list.

Each match is invalidated by marking it stale and optionally removing it from the cache. This means that a future request for the resource will cause it to retrieve a new copy of the resource from the reflector.

When the repeater has completed the invalidation, it returns an acknowledgment to the master. The master waits until all repeaters have acknowledged the invalidation request.

If a repeater fails to acknowledge within a given period, it is disconnected from the master repeater. When it reconnects, it will be told to flush its entire cache, which will eliminate any consistency problem. (To avoid flushing the entire cache, the master could keep a log of all invalidations performed, sorted by date, and flush only files invalidated since the last time the reconnecting repeater successfully completed an invalidation. In the presently preferred embodiments this is not done since it is believed that repeaters will seldom disconnect.)

When all repeaters have acknowledged invalidation (or timed out) the repeater broadcasts a "phase 2" invalidation request to all repeaters. This causes the repeaters to remove

US 8,478,903 B2

19

the corresponding resource identifiers and regular expressions from the list of resource identifiers pending invalidation.

In another embodiment, the invalidation request will be extended to allow a “server push”. In such requests, after phase 2 of the invalidation process has completed, the repeater receiving the invalidation request will immediately request a new copy of the invalidated resource to place in its cache.

Logs and Log Processing

Web server activity logs are fundamental to monitoring the activity in a Web site. This invention creates “merged logs” that combine the activity at reflectors with the activity at repeaters, so that a single activity log appears at the origin server showing all Web resource requests made on behalf of that site at any repeater.

This merged log can be processed by standard processing tools, as if it had been generated locally.

On a periodic basis, the master repeater (or its delegate) collects logs from each repeater. The logs collected are merged, sorted by reflector identifier and timestamp, and stored in a dated file on a per-reflector basis. The merged log for a given reflector represents the activity of all repeaters on behalf of that reflector. On a periodic basis, as configured by the reflector operator, a reflector contacts the master repeater to request its merged logs. It downloads these and merges them with its locally maintained logs, sorting by timestamp. The result is a merged log that represents all activity on behalf of repeaters and the given reflector.

Activity logs are optionally extended with information important to the repeater network, if the reflector is configured to do so by the reflector operator. In particular, an “extended status code” indicates information about each request, such as:

1. request was served by a reflector locally;
2. request was reflected to a repeater;*
3. request was served by a reflector to a repeater;*
4. request for non-repeatable resource was served by repeater;*
5. request was served by a repeater from the cache;
6. request was served by a repeater after filling cache;
7. request pending invalidation was served by a repeater.

(The activities marked with “*” represent intermediate states of a request and do not normally appear in a final activity log.)

In addition, activity logs contain a duration, and extended precision timestamps. The duration makes it possible to analyze the time required to serve a resource, the bandwidth used, the number of requests handled in parallel at a given time, and other quite useful information. The extended precision timestamp makes it possible to accurately merge activity logs.

Repeaters use the Network Time Protocol (NTP) to maintain synchronized clocks. Reflectors may either use NTP or calculate a time bias to provide roughly accurate timestamps relative to their contact repeater.

Enforcing Committed Aggregate Information Rate

The repeater network monitors and limits the aggregate rate at which data is served on behalf of a given subscriber by all repeaters. This mechanism provides the following benefits:

1. provides a means of pricing repeater service;
2. provides a means for estimating and reserving capacity at repeaters;
3. provides a means for preventing clients of a busy site from limiting access to other sites.

For each subscriber, a “threshold aggregate information rate” (TAIR) is configured and maintained at the master

20

repeater. This threshold is not necessarily the committed rate, it may be a multiple of committed rate, based on a pricing policy.

Each repeater measures the information rate component of each reflector for which it serves resources, periodically (typically about once a minute), by recording the number of bytes transmitted on behalf of that reflector each time a request is delivered. The table thus created is sent to the master repeater once per period. The master repeater combines the tables from each repeater, summing the measured information of each reflector over all repeaters that serve resources for that reflector, to determine the “measured aggregate information rate” (MAIR) for each reflector.

If the MAIR for a given reflector is greater than the TAIR for that reflector, the MAIR is transmitted by the master to all repeaters and to the respective reflector.

When a reflector receives a request, it determines whether its most recently calculated MAIR is greater than its TAIR. If this is the case, the reflector probabilistically decides whether to suppress reflection, by serving the request locally (in B2). The probability of suppressing the reflection increases as an exponential function of the difference between the MAIR and the CAIR.

Serving a request locally during a peak period may strain the local origin server, but it prevents this subscriber from taking more than allocated bandwidth from the shared repeater network.

When a repeater receives a request for a given subscriber (in C2), it determines whether the subscriber is running near its threshold aggregate information rate. If this is the case, it probabilistically decides whether to reduce its load by redirecting the request back to the reflector. The probability increases exponentially as the reflector’s aggregate information rate approaches its limit.

If a request is reflected back to a reflector, a special character string is attached to the resource identifier so that the receiving reflector will not attempt to reflect it again. In the current system, this string has the form

“src=overload”.

The reflector tests for this string in B2.

The mechanism for limiting Aggregate Information Rate described above is fairly coarse. It limits at the level of sessions with clients (since once a client has been reflected to a given repeater, the rewriting process tends to keep the client coming back to that repeater) and, at best, individual requests for resources. A more fine-grained mechanism for enforcing TAIR limits within repeaters operates by reducing the bandwidth consumption of a busy subscriber when other subscribers are competing for bandwidth.

The fine-grained mechanism is a form of data “rate shaping”. It extends the mechanism that copies resource data to a connection when a reply is being sent to a client. When an output channel is established at the time a request is received, the repeater identifies which subscriber the channel is operating for, in C2, and records the subscriber in a data field associated with the channel. Each time a “write” operation is about to be made to the channel, the Metered Output Stream first inspects the current values of the MAIR and TAIR, calculated above, for the given subscriber. If the MAIR is larger than the TAIR, then the mechanism pauses briefly before performing the write operation. The length of the pause is proportional to the amount the MAIR exceeds the TAIR. The pause ensures that tasks sending other resources to other clients, perhaps on behalf of other subscribers, will have an opportunity to send their data.

US 8,478,903 B2

21

Repeater Network Resilience

The repeater network is capable of recovering when a repeater or network connection fails.

A repeater cannot operate unless it is connected to the master repeater. The master repeater exchanges critical information with other repeaters, including information about repeater load, aggregate information rate, subscribers, and link cost.

If a master fails, a "succession" process ensures that another repeater will take over the role of master, and the network as a whole will remain operational. If a master fails, or a connection to a master fails through a network problem, any repeater attempting to communicate with the master will detect the failure, either through an indication from TCP/IP, or by timing out from a regular "heartbeat" message it sends to the master.

When any repeater is disconnected from its master, it immediately tries to reconnect to a series of potential masters based on a configurable file called its "succession list".

The repeater tries each system on the list in succession until it successfully connects to a master. If in this process, it comes to its own name, it takes on the role of master, and accepts connections from other repeaters. If a repeater which is not at the top of the list becomes the master, it is called the "temporary master".

A network partition may cause two groups of repeaters each to elect a master. When the partition is corrected, it is necessary that the more senior master take over the network. Therefore, when a repeater is temporary master, it regularly tries to reconnect to any master above it in the succession list. If it succeeds, it immediately disconnects from all of the repeaters connected to it. When they retry their succession lists, they will connect to the more senior master repeater.

To prevent losses of data, a temporary master does not accept configuration changes and does not process log files. In order to take on these tasks, it must be informed that it is primary master by manual modification of its successor list. Each repeater regularly reloads its successor list to determine whether it should change its idea of who the master is.

If a repeater is disconnected from the master, it must resynchronize its cache when it reconnects to the master. The master can maintain a list of recent cache invalidations and send to the repeater any invalidations it was not able to process while disconnected. If this list is not available for some reason (for instance, because the reflector has been disconnected too long), the reflector must invalidate its entire cache.

A reflector is not permitted to reflect requests unless it is connected to a repeater. The reflector relies on its contact repeater for critical information, such as load and Link Cost Tables, and current aggregate information rate. A reflector that is not connected to a repeater can continue to receive requests and handle them locally.

If a reflector loses its connection with a repeater, due to a repeater failure or network outage, it continues to operate while it tries to connect to a repeater.

Each time a reflector attempts to connect to a repeater, it uses DNS to identify a set of candidate repeaters given a domain name that represents the repeater network. The reflector tries each repeater in this set until it makes a successful contact. Until a successful contact is made, the reflector serves all requests locally. When a reflector connects to a repeater, the repeater can tell it to attempt to contact a different repeater; this allows the repeater network to ensure that no individual repeater has too many contacts.

When contact is made, the reflector provides the version number of each of its tables to its contact repeater. The repeater then decides which tables should be updated and

22

sends appropriate updates to the reflector. Once all tables have been updated, the repeater notifies the reflector that it may now start reflecting requests.

Using a Proxy Cache within a Repeater

Repeaters are intentionally designed so that any proxy cache can be used as a component within them. This is possible because the repeater receives HTTP requests and converts them to a form recognized by the proxy cache.

On the other hand, several modifications to a standard proxy cache have been or may be made as optimizations. This includes, in particular, the ability to conveniently invalidate a resource, the ability to support cache quotas, and the ability to avoid making an extra copy of each resource as it passes from the proxy cache through the repeater to the requester.

In a preferred embodiment, a proxy cache is used to implement C3. The proxy cache is dedicated for use only by one or more repeaters. Each repeater requiring a resource from the proxy cache constructs a proxy request from the inbound resource request. A normal HTTP GET request to a server contains only the pathname part of the URL—the scheme and server name are implicit. (In an HTTP GET request to a repeater, the pathname part of the URL includes the name of the origin server on behalf of which the request is being made, as described above.) However, a proxy agent GET request takes an entire URL. Therefore, the repeater must construct a proxy request containing the entire URL from the path portion of the URL it receives. Specifically, if the incoming request takes the form:

GET/<origin server>/<path>

then the repeater constructs a proxy request of the form:

GET http://<origin server>/<path>

and if the incoming request takes the form:

GET<origin server>@proxy=<scheme>:<type>@/ <path>

then the repeater constructs a proxy request of the form:

GET<scheme>://<origin server>/<path>

Cache Control

HTTP replies contain directives called cache control directives, which are used to indicate to a cache whether the attached resource may be cached and if so, when it should expire. A Web site administrator configures the Web site to attach appropriate directives. Often, the administrator will not know how long a page will be fresh, and must define a short expiration time to try to prevent caches from serving stale data. In many cases, a Web site operator will indicate a short expiration time only in order to receive the requests (or hits) that would otherwise be masked by the presence of a cache. This is known in the industry as "cache-busting". Although some cache operators may consider cache-busting to be impolite, advertisers who rely on this information may consider it imperative.

When a resource is stored in a repeater, its cache directives can be ignored by the repeater, because the repeater receives explicit invalidation events to determine when a resource is stale. When a proxy cache is used as the cache at the repeater, the associated cache directives may be temporarily disabled. However, they must be re-enabled when the resource is served from the cache to a client, in order to permit the cache-control policy (including any cache-busting) to take effect.

The present invention contains mechanisms to prevent the proxy cache within a repeater from honoring cache control directives, while permitting the directives to be served from the repeater.

When a reflector serves a resource to a repeater in B4, it replaces all cache directives by modified directives that are ignored by the repeater proxy cache. It does this by prefixing

US 8,478,903 B2

23

a distinctive string such as “wr-” to the beginning of the HTTP tag. Thus, “expires” becomes “wr-expires”, and “cache-control” becomes “wr-cache-control”. This prevents the proxy cache itself from honoring the directives. When a repeater serves a resource in C4, and the requesting client is not another repeater, it searches for HTTP tags beginning with “wr-” and removes the “wr-”. This allows the clients retrieving the resource to honor the directives.

Resource Revalidation

There are several cases where a resource may be cached so long as the origin server is consulted each time it is served. In one case, the request for the resource is attached to a so-called “cookie”. The origin server must be presented with the cookie to record the request and determine whether the cached resource may be served or not. In another case, the request for the resource is attached to an authentication header (which identifies the requester with a user id and password). Each new request for the resource must be tested at the origin server to assure that the requester is authorized to access the resource.

The HTTP 1.1 specification defines a reply header titled “Must-Revalidate” which allows an origin server to instruct a proxy cache to “revalidate” a resource each time a request is received. Normally, this mechanism is used to determine whether a resource is still fresh. In the present invention, Must-Revalidate makes it possible to ask an origin server to validate a request that is otherwise served from a repeater.

The reflector rule base contains information that determines which resources may be repeated but must be revalidated each time they are served. For each such resource, in B4, the reflector attaches a Must-Revalidate header. Each time a request comes to a repeater for a cached resource marked with a Must-Revalidate header, the request is forwarded to the reflector for validation prior to serving the requested resource.

Cache Quotas

The cache component of a repeater is shared among those subscribers that reflect clients to that repeater. In order to allow subscribers fair access to storage facilities, the cache may be extended to support quotas.

Normally, a proxy cache may be configured with a disk space threshold T. Whenever more than T bytes are stored in the cache, the cache attempts to find resources to eliminate.

Typically a cache uses the least-recently-used (LRU) algorithm to determine which resources to eliminate; more sophisticated caches use other algorithms. A cache may also support several threshold values—for instance, a lower threshold which, when reached, causes a low priority background process to remove items from the cache, and a higher threshold which, when reached, prevents resources from being cached until sufficient free disk space has been reclaimed.

If two subscribers A and B share a cache, and more resources of subscriber A are accessed during a period of time than resources of subscriber B, then fewer of B’s resources will be in the cache when new requests arrive. It is possible that, due to the behavior of A, B’s resources will never be cached when they are requested. In the present invention, this behavior is undesirable. To address this issue, the invention extends the cache at a repeater to support cache quotas.

The cache records the amount of space used by each subscriber in D_S , and supports a configurable threshold T_S for each subscriber.

Whenever a resource is added to the cache (at C3), the value D_S is updated for the subscriber providing the resource. If D_S is larger than T_S , the cache attempts to find resources to

24

eliminate, from among those resources associated with subscriber S. The cache is effectively partitioned into separate areas for each subscriber.

The original threshold T is still supported. If the sum of reserved segments for each subscriber is smaller than the total space reserved in the cache, the remaining area is “common” and subject to competition among subscribers.

Note, this mechanism might be implemented by modifying the existing proxy cache discussed above, or it might also be implemented without modifying the proxy cache—if the proxy cache at least makes it possible for an external program to obtain a list of resources in the cache, and to remove a given resource from the cache.

Rewriting from Repeaters

When a repeater receives a request for a resource, its proxy cache may be configured to determine whether a peer cache contains the requested resource. If so, the proxy cache obtains the resource from the peer cache, which can be faster than obtaining it from the origin server (the reflector). However, a consequence of this is that rewritten HTML resources retrieved from the peer cache would identify the wrong repeater. Thus, to allow for cooperating proxy caches, resources are preferably rewritten at the repeater.

When a resource is rewritten for a repeater, a special tag is placed at the beginning of the resource. When constructing a reply, the repeater inspects the tag to determine whether the resource indicates that additional rewriting is necessary. If so, the repeater modifies the resource by replacing references to the old repeater with references to the new repeater.

It is only necessary to perform this rewriting when a resource is served to the proxy cache at another repeater.

Repeater-Side Include

Sometimes, an origin server constructs a custom resource for each request (for instance, when inserting an advertisement based on the history of the requesting client). In such a case, that resource must be served locally rather than repeated. Generally, a custom resource contains, along with the custom information, text and references to other, repeatable, resources.

The process that assembles a “page” from a text resource and possibly one or more image resources is performed by the Web browser, directed by HTML. However, it is not possible using HTML to cause a browser to assemble a page using text or directives from a separate resource. Therefore, custom resources often necessarily contain large amounts of static text that would otherwise be repeatable.

To resolve this potential inefficiency, repeaters recognize a special directive called a “repeater side include”. This directive makes it possible for the repeater to assemble a custom resource, using a combination of repeatable and local resources. In this way, the static text can be made repeatable, and only the special directive need be served locally by the reflector.

For example, a resource X might consist of custom directives selecting an advertising banner, followed by a large text article. To make this resource repeatable, the Web site administrator must break out a second resource, Y, to select the banner. Resource X is modified to contain a repeater-side include directive identifying resource Y, along with the article. Resource Y is created and contains only the custom directives selecting an ad banner. Now resource X is repeatable, and only resource Y, which is relatively small, is not repeatable.

When a repeater constructs a reply, it determines whether the resource being served is an HTML resource, and if so, scans it for repeater-side include directives. Each such directive includes a URL, which the repeater resolves and substi-

US 8,478,903 B2

25

tutes in place of the directive. The entire resource must be assembled before it is served, in order to determine its final size, as the size is included in a reply header ahead of the resource.

Thus, a method and apparatus for dynamically replicating selected resources in computer networks is provided. One skilled in the art will appreciate that the present invention can be practiced by other than the described embodiments, which are presented for purposes of illustration and not limitation, and the present invention is limited only by the claims that follow.

What is claimed:

1. A content delivery system operative in a computer network for delivering resources associated with a plurality of content providers to multiple client machines, the system comprising:

at least one shared repeater server constructed and adapted to replicate at least some of the resources associated with a first content provider of said plurality of content providers, and to replicate at least some of the resources associated with a second content provider of said plurality of content providers, wherein the second content provider is distinct from the first content provider;

at least one table listing a plurality of alias names corresponding to content providers authorized to have resources delivered to client machines via the at least one shared repeater server, wherein the plurality of content providers comprises the first content provider and the second content provider;

wherein at least a first resource associated with the first content provider is associated with a first alias name of the plurality of alias names, said first alias name being associated with said at least one shared repeater server;

wherein, requests for the first resource from said client machines are directed to the at least one shared repeater server, based at least in part on said first alias name;

wherein at least a second resource associated with the second content provider is associated with a second alias name of the plurality of alias names, said second alias name being associated with said at least one shared repeater server;

wherein requests for the second resource made from said client machines are directed to the at least one shared repeater server, based at least in part on said second alias name; and

wherein the at least one shared repeater server is further constructed and adapted to analyze, using the table, an alias name received with a client request for a particular resource to determine a content provider associated with the particular resource.

2. The content delivery system as recited in claim 1, wherein the least one shared repeater server is addressable using the first alias name and the second alias name.

3. The content delivery system as recited in claim 1, further comprising:
a reflector.

4. The content delivery system as recited in claim 3, wherein the reflector is co-located with an origin server.

5. The content delivery system as recited in claim 4, wherein the reflector is co-located with an origin server at which the first resource originates.

6. The content delivery system as recited in claim 3, wherein the reflector is co-located with an origin server from which the first resource originates.

7. The content delivery system as recited in claim 3, wherein the reflector is constructed and adapted to associate

26

the resources with the first alias name based, at least in part, on a relative cost of transmitting the resources to the client machines.

8. The content delivery system as recited in claim 3, further comprising:

a repeater selector mechanism.

9. The content delivery system as recited in claim 8, wherein the reflector comprises the repeater selector mechanism.

10. The content delivery system of claim 3, wherein the reflector selects the at least one repeater server to process a client request for a resource.

11. The content delivery system as recited in claim 1 further comprising:

a set of shared repeater servers that comprises the at least one repeater server; and

a repeater selector mechanism constructed and adapted to associate the first resource with the first alias name after selecting the at least one shared repeater server from the set of shared repeater servers in response to a request for the first resource issued by a particular client machine.

12. The content delivery system as recited in claim 11, further comprising:

a load determination mechanism constructed and adapted to characterize load on the set of repeater servers for use in selection of the at least one shared repeater server.

13. The content delivery system as recited in claim 12, wherein the repeater selector mechanism comprises the load determination mechanism.

14. The content delivery system as recited in claim 12, further comprising:

a cost determination mechanism constructed and adapted to measure a cost of delivering resources from each of the set of shared repeater servers to the particular client machine, the cost determination mechanism for use in selection of the at least one shared repeater server.

15. The content delivery system as recited in claim 14, wherein the repeater selector mechanism comprises the cost determination mechanism.

16. The content delivery system as recited in claim 14, wherein the cost comprises information relating a location associated with the particular client machine to each of the set of shared repeater servers.

17. A content delivery system as recited in claim 14 wherein the cost determination mechanism comprises a net-map determination means.

18. The content delivery system as recited in claim 14, wherein

the cost determination mechanism is constructed and adapted to measure a cost of delivering resources from each of the set of shared repeater servers to the particular client machine, the cost determination mechanism for use by a reflector in selection of the at least one shared repeater server.

19. The content delivery system as recited in claim 12, wherein

the load determination mechanism is constructed and adapted to characterize load on the set of repeater servers for use by a reflector in selection of the at least one shared repeater server.

20. The content delivery system as recited in claim 11, wherein the repeater selector mechanism is part of a reflector.

21. The content delivery system of claim 11, wherein the repeater selector mechanism selects the at least one repeater server to process a client request for a resource.

US 8,478,903 B2

27

22. The content delivery system as recited in claim 1, wherein at least one of the first resource and the second resource is an embedded object.

23. The content delivery system as recited in claim 1, wherein the requests comprise host tag names that identify specific origin servers from which the resources originate.

24. The content delivery system as recited in claim 23, wherein the host tag names are included in HTTP headers.

25. A content delivery system as recited in claim 1, wherein at least the first alias name is a domain name, the system further comprising:

at least one name server that provides domain name service (DNS) resolution on the first alias name to identify the at least one shared repeater server to deliver the first resource in response to a client request.

26. A method as recited in claim 1, wherein the table further comprises, for each of the plurality of content providers, information from the group consisting of: a reflector name and a committed aggregate information rate (CAIR).

27. A content delivery system as recited in claim 1, wherein the at least one repeater server is configured and adapted to reject the client request for the particular resource if the alias name received with the client request is not included in the table.

28. A method, in a content delivery system operative in a computer network for delivering content to client machines, the computer network comprising a plurality of origin servers, each of said origin servers having resources associated therewith, and the content delivery system comprising at least one shared repeater server operable to replicate resources associated with the plurality of origin servers, the method comprising:

associating the at least one repeater server with a first alias name, wherein requests for a first resource located on a first origin server are directed, based at least in part on said first alias name, to the at least one repeater server for delivery of the first resource from said at least one repeater server;

associating the at least one repeater server with a second alias name, wherein requests for a second resource located on a second origin server are directed, based at least in part on said second alias name, to the at least one repeater server for delivery of the second resource from said at least one repeater server, wherein the second origin server is distinct from the first origin server;

providing a table listing origin servers having content located thereon, wherein said content is authorized for delivery to client machines via the at least one shared repeater server, and wherein the origin servers comprise the first origin server and the second origin server; and wherein the at least one repeater server is further constructed and adapted to analyze, using the table, an alias name received with a client request for a particular resource to determine an origin server associated with the particular resource.

29. The method of claim 28, further comprising:

in response to a request for the first resource issued by a particular client machine, identifying the at least one shared repeater server from a set of shared repeater servers based at least in part on a network cost of delivering resources to the particular client machine; and associating the first resource with the first alias name.

30. The method of claim 28, further comprising:

in response to a request for the first resource issued by a particular client machine, identifying the at least one repeater server from a set of shared repeater servers

28

based at least in part on load characteristics of the set of shared repeater servers; and

associating the first resource with the first alias name.

31. The method of claim 28, further comprising:

in response to a request for the first resource issued by a particular client machine, identifying the at least one repeater server from a set of shared repeater servers based at least in part on a location of the particular client machine relative to the at least one shared repeater server; and

associating the first resource with the first alias name.

32. The method as recited in claim 28, wherein the resources comprise an HTTP-based resource.

33. The method as recited in claim 28, wherein the resources comprise a File Transfer Protocol (FTP)-based resource.

34. A method as recited in claim 28, wherein at least the first alias name is a domain name and the computer network further comprises at least one name server that provides domain name service (DNS) resolution, the method further comprising:

receiving, at the at least one name server, the first alias name as part of a client request; and

resolving, at the at least one name server, the received first alias name to identify the at least one shared repeater server to deliver the first resource in response to the client request.

35. A method as recited in claim 28, wherein the table further comprises, for each of the listed origin servers, information from the group consisting of: a reflector name and a committed aggregate information rate (CAIR).

36. A method as recited in claim 28, further comprising:

rejecting the client request for the particular resource if the alias name received with the client request is not included in the table.

37. A method, in a content delivery system operative in a computer network for delivering content to client machines and comprising at least one shared repeater server operable to replicate resources stored on a plurality of origin servers, the method comprising:

associating a plurality of alias names with the at least one shared repeater server, each of said plurality of alias names being associated with an origin server, wherein a first alias name of said plurality of alias names is associated with a first origin server, and at least a second alias name of said plurality of alias names is associated with a second origin server distinct from said first origin server,

providing at least one table associating alias names with origin servers having content located thereon, wherein said content is authorized for delivery to client machines via the at least one shared repeater server, wherein the origin servers comprise the first origin server and the second origin server;

wherein requests for a first resource originating on the first origin server are directed, based at least in part on said first alias name, to the at least one shared repeater server for delivery of the first resource from said at least one repeater server; and

wherein requests for a second resource originating on the second origin server are directed, based at least in part on said second alias name, to the at least one shared repeater server for delivery of the second resource from said at least one repeater server; and

US 8,478,903 B2

29

wherein the at least one repeater server uses the at least one table and an alias name received with a client request for a resource to determine an origin server associated with the requested resource.

38. The method as in claim 37, wherein a copy of the at least one table is located on each of a plurality of repeater servers.

39. A method as recited in claim 37, further comprising: rejecting the client request for the resource if the alias name received with the client request is not included in the table.

40. A server operative in a computer network for delivering resources associated with a plurality of content providers to multiple client machines, wherein the plurality of content providers comprise at least a first content provider and a second content provider distinct from the first content provider, the server comprising:

cache storage;

a table listing content providers of said plurality of content providers having content that is authorized for delivery to client machines via the server,

wherein the server is associated with a first alias name corresponding to said first content provider, and

wherein the server is associated with a second alias name corresponding to said second content provider, said second alias name being distinct from the first alias name, and

wherein the server is operable to replicate at least some of the resources associated with the first content provider, and at least some of the resources associated with the second content provider, and

wherein, in response to a request for a resource, the server is operable to use at least the table to analyze an alias name associated with the request to determine a content provider associated with the resource.

41. A server as recited in claim 40, wherein the table further comprises, for each of the listed content providers, information from the group consisting of: a reflector name and a committed aggregate information rate (CAIR).

30

42. A server as recited in claim 40, wherein in response to a request for a resource, the server is further configured and adapted to reject the request for the resource if the alias name received with the request is not included in the table.

43. A server operative in a computer network for delivering resources to multiple client machines, the server comprising: cache storage;

a table listing a plurality of origin servers having content that is authorized for delivery to client machines via the server,

wherein the server is associated with a first alias name corresponding to a first origin server of the plurality of origin servers, and

wherein the server is associated with a second alias name corresponding to a second origin server of the plurality of origin servers, said second origin server being distinct from the first origin server and said second alias name being distinct from the first alias name, and

wherein the server is operable to replicate at least some of the resources located on the first origin server, and at least some of the resources located on the second origin server, and

wherein, in response to a request for a resource, the server uses at least the table to analyze an alias name associated with the request to determine an origin server associated with the particular resource.

44. A server as recited in claim 43, wherein the first origin server and the second origin server are associated with distinct content providers located at distinct physical locations.

45. A server as recited in claim 43, wherein the table further comprises, for each of the listed origin servers, information from the group consisting of: a reflector name and a committed aggregate information rate (CAIR).

46. A server as recited in claim 43, wherein in response to a request for a resource, the server is further configured and adapted to reject the request for the resource if the alias name received with the request is not included in the table.

* * * * *

EXHIBIT B

US008645517B2

(12) **United States Patent**
Stolorz et al.

(10) **Patent No.:** **US 8,645,517 B2**
(45) **Date of Patent:** **Feb. 4, 2014**

(54) **POLICY-BASED CONTENT DELIVERY
NETWORK SELECTION**

(52) **U.S. Cl.**
USPC **709/223**; 709/238; 709/219

(75) Inventors: **Paul E. Stolorz**, Los Altos, CA (US); **John K. Salmon**, Chappaqua, NY (US); **Michael S. Warren**, Los Alamos, NM (US); **Jeffrey G. Koller**, Oxnard, CA (US); **Maksim Yevmenkin**, Thousand Oaks, CA (US); **Mark Brady**, San Rafael, CA (US); **David Pfitzner**, College Park (AU); **Ted Middleton**, Moorpark, CA (US); **Aric Hagberg**, Sante Fe, NM (US)

(58) **Field of Classification Search**
None
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,495,570	A	1/1985	Kitajima et al.
4,591,983	A	5/1986	Bennett et al.
4,594,704	A	6/1986	Ollivier
4,726,017	A	2/1988	Krum et al.

(Continued)

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

FOREIGN PATENT DOCUMENTS

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 311 days.

AU	763380	11/2003
CA	2202572	10/1998

(Continued)

(21) Appl. No.: 12/978,537

OTHER PUBLICATIONS

(22) Filed: **Dec. 24, 2010**
(Under 37 CFR 1.47)

Chinese Reexamination Notification, dated Jul. 5, 2012, Appl. No. 02821371.8, 5 pgs.

(65) **Prior Publication Data**
US 2011/0145386 A1 Jun. 16, 2011

(Continued)

Primary Examiner — John B. Walsh

Related U.S. Application Data

(63) Continuation of application No. 11/976,648, filed on Oct. 26, 2007, now Pat. No. 7,860,964, which is a continuation-in-part of application No. 10/259,497, filed on Sep. 30, 2002, now Pat. No. 7,822,871, said application No. 12/978,537 is a continuation-in-part of application No. 11/932,162, filed on Oct. 31, 2007, which is a continuation of application No. 10/259,497.

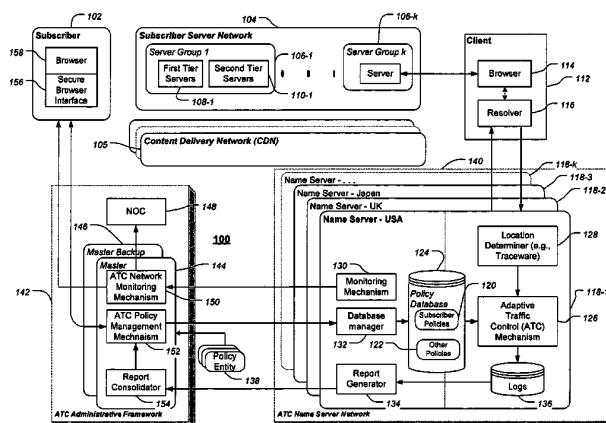
(57) **ABSTRACT**

(60) Provisional application No. 60/325,177, filed on Sep. 28, 2001.

In a framework wherein resources of a content provider may be delivered to clients from different domains, a method distributes the requests based on content-provider policies. In some cases, the domains include at least two distinct content delivery network (CDN) domains. The domains may include a content provider domain. Responsive to a request, either the content provider domain or one of the two CDN domains is selected, the selection being based at least in part on one or more policies set by the content provider.

(51) **Int. Cl.**
G06F 15/73 (2006.01)
G06F 15/16 (2006.01)

20 Claims, 31 Drawing Sheets



US 8,645,517 B2

Page 2

(56)

References Cited

U.S. PATENT DOCUMENTS

4,803,641 A	2/1989	Hardy et al.	5,774,668 A	6/1998	Choquier et al.
4,839,798 A	6/1989	Eguchi et al.	5,777,988 A	7/1998	Cisneros
4,847,784 A	7/1989	Clancey	5,777,989 A	7/1998	McGarvey
4,920,432 A	4/1990	Eggers et al.	5,784,058 A	7/1998	LaStrange et al.
4,922,417 A	5/1990	Churm et al.	5,796,952 A	8/1998	Davis
4,943,932 A	7/1990	Lark et al.	5,799,141 A	8/1998	Galipeau et al.
4,949,187 A	8/1990	Cohen	5,802,106 A	9/1998	Packer
4,949,248 A	8/1990	Caro	5,802,291 A	9/1998	Balick et al.
5,029,232 A	7/1991	Nall	5,812,769 A	9/1998	Graber et al.
5,130,792 A	7/1992	Tindell et al.	5,815,664 A	9/1998	Asano
5,132,992 A	7/1992	Yurt et al.	5,826,031 A	10/1998	Nielsen
5,136,716 A	8/1992	Harvey	5,828,847 A	10/1998	Gehr
5,172,413 A	12/1992	Bradley et al.	5,832,506 A	11/1998	Kuzma
5,191,573 A	3/1993	Hair	5,832,514 A	11/1998	Norin et al.
5,253,275 A	10/1993	Yurt et al.	5,835,718 A	11/1998	Blewett
5,253,341 A	10/1993	Rozmanith et al.	5,838,906 A	11/1998	Doyle et al.
5,287,499 A	2/1994	Nemes	5,845,303 A	12/1998	Templeman
5,287,537 A	2/1994	Newmark et al.	5,856,974 A	1/1999	Gervais et al.
5,291,554 A	3/1994	Morales	5,862,339 A	1/1999	Bonnaure
5,341,477 A	8/1994	Pitkin et al.	5,867,706 A	2/1999	Martin et al.
5,371,532 A	12/1994	Gelman et al.	5,867,799 A	2/1999	Lang et al.
5,410,343 A	4/1995	Coddington et al.	5,870,546 A	2/1999	Kirsch
5,414,455 A	5/1995	Hooper et al.	5,870,559 A	2/1999	Leshem et al.
5,442,389 A	8/1995	Blahut et al.	5,878,212 A	3/1999	Civanlar et al.
5,442,390 A	8/1995	Hooper et al.	5,884,038 A	3/1999	Kapoor
5,442,749 A	8/1995	Northcutt et al.	5,890,171 A	3/1999	Blumer et al.
5,471,622 A	11/1995	Eadline	5,893,116 A	4/1999	Simmonds et al.
5,475,615 A	12/1995	Lin	5,894,554 A	4/1999	Lowery et al.
5,508,732 A	4/1996	Bottomley et al.	5,896,533 A	4/1999	Ramos et al.
5,515,511 A	5/1996	Nguyen et al.	5,903,723 A	5/1999	Beck et al.
5,519,435 A	5/1996	Anderson	5,907,704 A	5/1999	Gudmundson et al.
5,522,070 A	5/1996	Sumitomo	5,913,028 A	6/1999	Wang et al.
5,528,281 A	6/1996	Grady et al.	5,913,033 A	6/1999	Grout
5,539,621 A	7/1996	Kikinis	5,918,010 A	6/1999	Appleman et al.
5,542,087 A	7/1996	Neimat et al.	5,919,247 A	7/1999	Van Hoff et al.
5,544,313 A	8/1996	Shachnai et al.	5,920,701 A	7/1999	Miller et al.
5,544,327 A	8/1996	Dan et al.	5,931,904 A	8/1999	Banga
5,550,577 A	8/1996	Verbiest et al.	5,933,832 A	8/1999	Suzuoka et al.
5,550,863 A	8/1996	Yurt et al.	5,935,207 A	8/1999	Logue et al.
5,550,982 A	8/1996	Long et al.	5,944,780 A	8/1999	Chase
5,557,317 A	9/1996	Nishio et al.	5,945,989 A	8/1999	Freisztat et al.
5,572,643 A	11/1996	Judson	5,956,489 A	9/1999	San Andres et al.
5,590,288 A	12/1996	Castor	5,956,716 A	9/1999	Kenner et al.
5,592,611 A	1/1997	Midgely	5,958,008 A	9/1999	Pogrebisky et al.
5,594,910 A	1/1997	Filepp et al.	5,961,596 A	10/1999	Takubo et al.
5,603,026 A	2/1997	Demers et al.	5,966,440 A	10/1999	Hair
5,619,648 A	4/1997	Canale	5,968,121 A	10/1999	Logan et al.
5,623,656 A	4/1997	Lyons	5,973,696 A	10/1999	Agranat et al.
5,625,781 A	4/1997	Cline	5,978,791 A	11/1999	Farber et al.
5,627,829 A	5/1997	Gleeson et al.	5,983,214 A	11/1999	Lang et al.
5,630,067 A	5/1997	Kindell	5,983,227 A	11/1999	Nazem et al.
5,633,999 A	5/1997	Clowes	5,987,606 A	11/1999	Cirasole et al.
5,634,006 A	5/1997	Baughner et al.	5,991,809 A	11/1999	Kriegsman
5,638,443 A	6/1997	Stefik et al.	5,996,025 A	11/1999	Day
5,644,714 A	7/1997	Kikinis	6,002,720 A	12/1999	Yurt et al.
5,646,676 A	7/1997	Dewkett et al.	6,003,030 A	12/1999	Kenner et al.
5,649,186 A	7/1997	Ferguson	6,006,264 A	12/1999	Colby et al.
5,659,729 A	8/1997	Nielsen	6,012,090 A	1/2000	Chung et al.
5,666,362 A	9/1997	Chen	6,014,686 A	1/2000	Elnozahy et al.
5,671,279 A	9/1997	Elgamai	6,014,698 A	1/2000	Griffiths
5,675,734 A	10/1997	Hair	6,018,516 A	1/2000	Packer
5,682,512 A	10/1997	Tetrick	6,021,426 A	2/2000	Douglis
5,699,513 A	12/1997	Feigen et al.	6,026,440 A	2/2000	Sharder et al.
5,708,780 A	1/1998	Levergood et al.	6,029,175 A	2/2000	Chow et al.
5,712,979 A	1/1998	Graber et al.	6,029,176 A	2/2000	Cannon
5,715,453 A	2/1998	Stewart	6,035,332 A	3/2000	Ingrassia, Jr. et al.
5,721,914 A	2/1998	DeVries	6,038,216 A	3/2000	Packer
5,734,831 A	3/1998	Sanders	6,038,310 A	3/2000	Hollywood et al.
5,740,423 A	4/1998	Logan et al.	6,038,610 A	3/2000	Belfiore et al.
5,742,762 A	4/1998	Scholl	6,041,307 A	3/2000	Ahuja et al.
5,751,961 A	5/1998	Smyk	6,041,324 A	3/2000	Earl et al.
5,761,507 A	6/1998	Govett	6,044,405 A	3/2000	Driscoll, III et al.
5,761,663 A	6/1998	Largarde et al.	6,046,980 A	4/2000	Packer
5,764,906 A	6/1998	Edelstein et al.	6,049,831 A	4/2000	Gardell et al.
5,774,660 A	6/1998	Brendel et al.	6,052,718 A	4/2000	Gifford
			6,052,730 A	4/2000	Feliciano et al.
			6,052,788 A *	4/2000	Wesinger et al. 726/11
			6,065,051 A	5/2000	Steele et al.
			6,065,062 A	5/2000	Periasamy et al.

US 8,645,517 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

6,070,191 A	5/2000	Narendran et al.	6,442,549 B1	8/2002	Schneider
6,081,829 A	6/2000	Sidana	6,456,630 B1	9/2002	Packer et al.
6,092,112 A	7/2000	Fukushige	6,460,082 B1	10/2002	Lumelsky
6,092,178 A *	7/2000	Jindal et al. 712/27	6,460,085 B1	10/2002	Toporek et al.
6,092,204 A	7/2000	Baker	6,463,454 B1	10/2002	Lumelsky
6,098,078 A	8/2000	Gehani	6,463,508 B1	10/2002	Wolf
6,105,028 A	8/2000	Sullivan et al.	6,470,389 B1	10/2002	Chung et al.
6,108,673 A	8/2000	Brandt et al.	6,473,405 B2	10/2002	Ricciulli
6,108,703 A	8/2000	Leighton et al.	6,480,893 B2	11/2002	Kriegsman
6,112,231 A	8/2000	DeSimone et al.	6,484,143 B1	11/2002	Swildens et al.
6,112,239 A	8/2000	Kenner et al.	6,484,204 B1	11/2002	Rabinovich
6,112,240 A	8/2000	Pogue et al.	6,484,261 B1 *	11/2002	Wiegel 726/11
6,115,357 A	9/2000	Packer et al.	6,490,580 B1	12/2002	Dey et al.
6,115,752 A	9/2000	Chauhan	6,493,707 B1	12/2002	Dey et al.
6,119,143 A	9/2000	Dias et al.	6,496,856 B1	12/2002	Kenner et al.
6,125,388 A	9/2000	Reisman	6,502,125 B1	12/2002	Kenner et al.
6,125,394 A	9/2000	Rabinovich	6,502,215 B2	12/2002	Raad et al.
6,128,601 A	10/2000	Van Horne et al.	6,505,248 B1	1/2003	Casper et al.
6,128,660 A	10/2000	Grimm et al.	6,529,477 B1	3/2003	Toporek et al.
6,130,890 A	10/2000	Leinwand et al.	6,553,413 B1	4/2003	Leighton et al.
6,134,583 A	10/2000	Herriot	6,553,420 B1	4/2003	Karger et al.
6,144,375 A	11/2000	Jain et al.	6,557,054 B2	4/2003	Reisman
6,144,702 A	11/2000	Yurt et al.	6,564,251 B2	5/2003	Katariya et al.
6,144,996 A	11/2000	Starnes et al.	6,577,595 B1	6/2003	Counterman
6,151,624 A	11/2000	Teare et al.	6,581,090 B1	6/2003	Lindbo et al.
6,154,738 A	11/2000	Call	6,584,083 B1	6/2003	Toporek et al.
6,154,744 A	11/2000	Kenner et al.	6,587,837 B1	7/2003	Spagna
6,154,753 A	11/2000	McFarland	6,591,299 B2	7/2003	Riddle et al.
6,154,777 A	11/2000	Ebrahim	6,611,862 B2	8/2003	Reisman
6,157,648 A	12/2000	Volt et al.	6,625,643 B1	9/2003	Colby et al.
6,163,779 A	12/2000	Mantha et al.	6,654,344 B1	11/2003	Toporek et al.
6,167,427 A	12/2000	Rabinovich et al.	6,654,807 B2	11/2003	Farber et al.
6,173,311 B1	1/2001	Hassett et al.	6,658,464 B2	12/2003	Reisman
6,173,322 B1	1/2001	Hu	6,665,706 B2	12/2003	Kenner et al.
6,178,160 B1	1/2001	Bolton et al.	6,665,726 B1	12/2003	Leighton et al.
6,181,867 B1	1/2001	Kenner et al.	6,691,148 B1	2/2004	Zinky et al.
6,185,598 B1	2/2001	Farber	6,694,358 B1	2/2004	Swildens et al.
6,185,619 B1	2/2001	Joffe et al.	6,699,418 B2	3/2004	Okada et al.
6,189,030 B1	2/2001	Kirsch et al.	6,708,137 B2	3/2004	Carley
6,189,039 B1	2/2001	Harvey	6,718,328 B1	4/2004	Norris
6,195,680 B1	2/2001	Goldszmidt et al.	6,741,563 B2	5/2004	Packer
6,205,120 B1	3/2001	Packer et al.	6,751,673 B2	6/2004	Shaw
6,226,618 B1	5/2001	Downs	6,754,699 B2	6/2004	Swildens et al.
6,226,642 B1	5/2001	Beranek et al.	6,754,706 B1	6/2004	Swildens et al.
6,230,196 B1	5/2001	Guenthner et al.	6,763,377 B1	7/2004	Belknap et al.
6,243,760 B1	6/2001	Armbruster et al.	6,763,388 B1	7/2004	Tsimelzon
6,249,810 B1	6/2001	Kiraly	6,778,502 B2	8/2004	Ricciulli
6,256,675 B1	7/2001	Rabinovich	6,785,704 B1	8/2004	McCanne
6,263,313 B1	7/2001	Milsted	6,799,221 B1	9/2004	Kenner et al.
6,266,699 B1	7/2001	Sevcik	6,801,576 B1	10/2004	Haldeman et al.
6,269,394 B1	7/2001	Kenner et al.	6,834,306 B1	12/2004	Tsimelzon
6,272,566 B1	8/2001	Craft	6,842,604 B1	1/2005	Cook et al.
6,275,470 B1	8/2001	Ricciulli	6,859,791 B1	2/2005	Spagna
6,282,569 B1	8/2001	Wallis et al.	6,870,851 B1	3/2005	Leinwand et al.
6,282,574 B1	8/2001	Voit	6,874,032 B2	3/2005	Gersht et al.
6,286,045 B1	9/2001	Griffiths et al.	6,901,604 B1	5/2005	Kiraly
6,298,041 B1	10/2001	Packer	6,915,329 B2	7/2005	Kriegsman
6,311,214 B1	10/2001	Rhoads	6,928,442 B2	8/2005	Farber et al.
6,314,565 B1	11/2001	Kenner et al.	6,934,255 B1	8/2005	Toporek et al.
6,327,622 B1	12/2001	Jindal et al.	6,950,623 B2	9/2005	Brown et al.
6,330,602 B1	12/2001	Law et al.	6,963,910 B1	11/2005	Belknap
6,332,195 B1	12/2001	Green et al.	6,963,980 B1	11/2005	Mattsson
6,338,044 B1	1/2002	Cook et al.	6,963,981 B1	11/2005	Bailey et al.
6,347,085 B2	2/2002	Kelly	6,965,890 B1	11/2005	Dey et al.
6,360,256 B1	3/2002	Lim	6,970,432 B1	11/2005	Hankins et al.
6,370,571 B1	4/2002	Medin, Jr.	6,973,485 B2	12/2005	Ebata et al.
6,370,580 B2	4/2002	Kriegsman	6,973,490 B1	12/2005	Robertson et al.
6,398,245 B1	6/2002	Gruse	6,976,090 B2	12/2005	Ben-shaul
6,405,252 B1	6/2002	Gupta et al.	6,981,050 B1	12/2005	Tobias et al.
6,405,257 B1	6/2002	Gersht et al.	6,981,180 B1	12/2005	Bailey et al.
6,412,000 B1	6/2002	Riddle et al.	6,996,616 B1	2/2006	Leighton et al.
6,415,280 B1	7/2002	Farber et al.	7,003,572 B1	2/2006	Lownsbrough et al.
6,418,421 B1	7/2002	Hurtado	7,007,089 B2	2/2006	Freedman
6,421,726 B1	7/2002	Kenner et al.	7,010,578 B1	3/2006	Lewin et al.
6,430,618 B1	8/2002	Karger et al.	7,012,900 B1	3/2006	Riddle
			7,039,633 B1	5/2006	Dey et al.
			7,047,300 B1	5/2006	Oehrke et al.
			7,054,935 B2	5/2006	Farber et al.
			7,058,706 B1	6/2006	Iyer et al.

US 8,645,517 B2

Page 4

(56)

References Cited

FOREIGN PATENT DOCUMENTS

U.S. PATENT DOCUMENTS

7,069,177 B2 6/2006 Carley
 7,096,266 B2 8/2006 Lewin et al.
 7,103,564 B1 9/2006 Ehnebuske
 7,103,645 B2 9/2006 Leighton et al.
 7,110,984 B1 9/2006 Spagna
 7,117,259 B1 10/2006 Rohwer
 7,185,052 B2 2/2007 Day
 7,188,085 B2 3/2007 Pelletier
 7,206,748 B1 4/2007 Gruse
 7,219,153 B1 5/2007 Day
 7,254,645 B2 8/2007 Nishi
 7,373,644 B2 5/2008 Aborn
 7,577,754 B2 8/2009 Garcia-Luna-Aceves et al.
 2001/0029525 A1 10/2001 Lahr
 2001/0056500 A1 12/2001 Farber et al.
 2002/0018449 A1 2/2002 Ricciulli
 2002/0023164 A1 2/2002 Lahr
 2002/0023165 A1 2/2002 Lahr
 2002/0040404 A1 4/2002 Lahr
 2002/0042817 A1 4/2002 Lahr
 2002/0046273 A1 4/2002 Lahr et al.
 2002/0046405 A1 4/2002 Lahr
 2002/0049857 A1 4/2002 Farber et al.
 2002/0059592 A1 5/2002 Kiraly
 2002/0066038 A1 5/2002 Mattsson
 2002/0073199 A1 6/2002 Levine et al.
 2002/0078233 A1 6/2002 Biliris
 2002/0082999 A1 6/2002 Lee et al.
 2002/0083124 A1 6/2002 Knox et al.
 2002/0087684 A1 * 7/2002 Foth 709/225
 2002/0099850 A1 7/2002 Farber et al.
 2002/0116444 A1 8/2002 Chaudhri et al.
 2002/0124080 A1 9/2002 Leighton et al.
 2002/0129134 A1 9/2002 Leighton et al.
 2002/0131645 A1 9/2002 Hamilton
 2002/0143798 A1 10/2002 Lisiecki et al.
 2002/0143888 A1 10/2002 Lisiecki et al.
 2002/0145975 A1 * 10/2002 McLampy et al. 370/235
 2002/0147774 A1 10/2002 Lisiecki et al.
 2002/0163882 A1 11/2002 Bornstein et al.
 2002/0199016 A1 12/2002 Freedman
 2003/0009444 A1 1/2003 Eidler et al.
 2003/0018966 A1 1/2003 Cook et al.
 2003/0028623 A1 2/2003 Hennessey et al.
 2003/0028626 A1 2/2003 Hennessey et al.
 2003/0028777 A1 2/2003 Hennessey et al.
 2003/0055972 A1 3/2003 Fuller et al.
 2003/0061263 A1 3/2003 Riddle
 2003/0061280 A1 3/2003 Bulson et al.
 2003/0078888 A1 4/2003 Lee et al.
 2003/0078889 A1 4/2003 Lee et al.
 2003/0105604 A1 6/2003 Ash et al.
 2003/0149581 A1 8/2003 Chaudhri
 2004/0010588 A1 1/2004 Slater et al.
 2004/0022194 A1 2/2004 Ricciulli
 2004/0039798 A1 2/2004 Hotz et al.
 2004/0064832 A1 4/2004 Tsukidate et al.
 2004/0139097 A1 7/2004 Farber et al.
 2004/0177148 A1 9/2004 Tsimelzon
 2005/0033858 A1 2/2005 Swildens et al.
 2005/0038851 A1 2/2005 Kriegsmann
 2005/0060493 A1 3/2005 Krissell et al.
 2005/0100027 A1 5/2005 Leinwand et al.
 2005/0114296 A1 5/2005 Farber et al.
 2005/0240966 A1 10/2005 Hindle et al.
 2005/0262104 A1 11/2005 Robertson et al.
 2006/0143293 A1 6/2006 Freedman
 2007/0055764 A1 3/2007 Dilley et al.
 2009/0254661 A1 10/2009 Fullagar et al.
 2010/0169772 A1 7/2010 Stallings et al.
 2010/0312861 A1 12/2010 Kolhi et al.
 2010/0332595 A1 12/2010 Fullagar et al.

CA 2335661 9/2001
 CA 2335662 9/2001
 CA 2467998 4/2006
 CN ZL99810853.7 8/2004
 EP 0 800 143 A 10/1997
 EP 0801487 A2 10/1997
 EP 0 817 020 A2 1/1998
 EP 0817444 1/1998
 EP 0824236 A2 2/1998
 EP 0865180 A2 9/1998
 EP 1104555 6/2001
 GB 2 281 793 3/1995
 GB 2353877 3/2004
 IL 140935 3/2006
 JP 5-130144 5/1993
 JP 05162529 6/1993
 JP 7066829 3/1995
 JP 08328583 9/1996
 JP 10-027148 1/1998
 JP 10-70571 3/1998
 JP 10-093552 4/1998
 JP 10-126445 5/1998
 JP 2000259539 9/2000
 JP 2001-053793 2/2001
 JP 2002522995 7/2002
 JP 3566626 6/2004
 JP 2005124165 5/2005
 JP 3762649 1/2006
 KR 1999-72167 9/1999
 NI 176482 8/2003
 WO WO 96/42041 12/1996
 WO WO 97/11429 3/1997
 WO WO 97/29423 8/1997
 WO WO 98/04985 2/1998
 WO WO 98/06033 2/1998
 WO WO 99 09726 A 2/1999
 WO WO 99 29083 A 6/1999
 WO WO 00/14633 3/2000
 WO WO 00 52594 A 9/2000
 WO WO-00/52594 A2 9/2000
 WO WO 00/62502 10/2000
 WO WO 01/35601 5/2001
 WO WO 01/52497 7/2001

OTHER PUBLICATIONS

"International Search Report, dated Jan. 23, 2012, Intl Appl. No. PCT/US11/51252", Intl Filing Date Sep. 12, 2011, 3 pgs.
 "Written Opinion, dated Jan. 23, 2012, Intl Appl No. PCT/US11/51252", Intl Filing Date Sep. 12, 2011, 5 pgs.
 Chinese Reexamination Notification, dated Sep. 13, 2012, Chinese App. No. 02821371.8, including translation, 10 pgs.
 European Examination Report dated Sep. 6, 2012, EP Application No. 10178695.2, 4 pgs.
 "Content Management Technology/Industry News," Content Technologies Trends and Advice, Gilbane Report, News for Jun. 1999 [21 pages].
 "Exporting Web Server Final Report," http://www.cs.technion.ac.il/Labs/Lccn/projects/spring97/project4/final_report.html, Spring 1997 (downloaded Jul. 7, 2007).
 "Local Area Network Server Replacement Procedure", IBM Technical Disclosure Bulletin, vol. 38, No. 1, (Jan. 1995), 235-236.
 "Patent Abstracts of Japan, Electronic Mail Multiplexing System and Communication Control Method in the System" (Appln. No. JP19930162529), (Jun. 30, 1993) (Pub. No. JP 7066829).
 "Patent Abstracts of Japan, Method and Device for Repeating and Converting Information", (Appln. No. JP1996 0328583) (Pub. No. JP10171727), Jun. 26, 1998.
 Adler, R. M., "Distributed Coordination Models for Client/Server Computing," Computer 28, 4 (Apr. 1995), 14-22.
 Aggarwal, A. et al., "Performance of Dynamic Replication Schemes for an Internet Hosting Service". Technical Report, AT&T Labs, Oct. 1998.
 Andresen et al., "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", Proc. IPPS, (Apr. 15, 1996), 850-856.

US 8,645,517 B2

Page 5

(56)

References Cited

OTHER PUBLICATIONS

- Andresen, D., et al., Multiprocessor scheduling with client resources to improve the response time of WWW applications, Proc. 11th Int'l Conf. on Supercomputing (Austria, Jul. 1997). ICS '97. ACM Press, NY, NY, 92-99.
- Awerbuch, B. et al., Distributed Paging for General Networks. In Proc. of the 7th ACM-SIAM Symposium on Discrete Algorithms, pp. 574-583, Jan. 1996.
- Awerbuch, et al., Competitive Distributed File Allocation. in Proc. of the 25th Ann. ACM Symp. on Theory of Computing, pp. 164-173, May 1993.
- Baentsch, M. et al., "Enhancing the Web's Infrastructure: From Caching to Replication." IEEE Internet Computing, 1(2):18-27, Mar. 1997.
- Bartal, Y., et al., "Competitive Algorithms for Distributed Data Management", 24th Annual ACM STOC, 5/92, Victoria, B.C. Canada.
- Basturk, E., et al., "Using network layer anycast for load distribution in the Internet," Tech. Rep., IBM T.J. Watson Research Center, Jul. 1997 (21 pgs.).
- Berners-Lee, T., et al., RFC 1738—Uniform Resource Locators, Dec. 1994.
- Bestavros, A., "Speculative Data Dissemination and Service to Reduce Server Load, Network Traffic and Service Time in Distributed Information Systems", In Proc. ICDE '96: The 1996 Int'l Conf. on Data Engineering, (Mar. 1996), 4 pages.
- Bestavros, et al., "Server-Initiated Document Dissemination for the WWW," IEEE Data Engineering Bulletin 19(3):3-11, Sep. 1996.
- Bhattacharjee et al., "Application-layer anycasting," in Proc. IEEE INFOCOM '97, Apr. 1997.
- Braun, H., et al., "Web traffic characterization: an assessment of the impact of caching documents from NCSA's web server," Comput. Netw. ISDN Syst. 28, 1-2 (Dec. 1995), 37-51.
- Brisco, T. P. RFC 1794: DNS support for load balancing, Apr. 1995.
- Carter et al., "Dynamic server selection using bandwidth probing in wide-area networks," Tech. Rep. BU-CS-96-007, Comp. Sci. Dept., Boston University, Mar. 1996.
- Carter et al., Server selection using dynamic path characterization in Wide-Area Networks, IEEE INFOCOM '97, Apr. 7-12, 1997 (pp. 1014-1021).
- Carter, J. Lawrence et al., "Universal Classes of Hash Functions", Journal of Computer and System Sciences, vol. 18, No. 2, (Apr. 1979), 143-154.
- Cate, V. "Alex: a global file system", in Proc. Usenix Conf. on File Systems, May 1992, pp. 1-11.
- Chankhunthod, A. et al., "A Hierarchical Internet Object Cache", Proc. of the 1996 USENIX Technical Conf., Jan. 1996, pp. 153-163.
- Cisco Systems, Inc., Cisco DistributedDirector 2500 Series Installation and Configuration Guide, pp. xix-xxii; 1-1 to 1-12; 6-1 to 6-18; 7-1 to 7-18; 8-1 to 8-24, Dec. 1997, downloaded Apr. 2007: <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distrdir/dd2501/>.
- Cisco Systems, Inc., Cisco DistributedDirector 4700-M Installation and Configuration Guide, pgs. xix-xxii; 1-1 to 1-14; 7-1 to 7-18, 8-1 to 8-20; Dec. 1997, [downloaded Apr. 2007 from <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distrdir/dd4700m/>].
- Claims (amended) filed by applicant in Japanese Patent Application No. 2003-531370, Jan. 17, 2008 (English translation provided by foreign associate) [12 pgs.].
- Claims (amended) filed by applicant in KIPO for Korean Patent Application No. KR 10 2004 7004613, Jul. 24, 2009 (English translation provided by foreign associate) [10 pgs.].
- CN Appln. No. 02821371.8—Jan. 30, 2011 SIPO Office Action.
- Cohen, J., et al., "Cache Array Routing Protocol v1.1", Sep. 29, 1997; <http://tools.ietf.org/id/draft-vinod-carp-v1-01.txt> (Last-Modified: Wed, Oct. 1, 1997).
- Colajanni, M. and Yu, P. S. 1997. Adaptive TTL schemes for load balancing of distributed Web servers. SIGMETRICS Perform. Eval. Rev. 25, 2 (Sep. 1997), 36-42.
- Communication (1 page) and European Search Report (with Annex) for Appln. No. EP 02799672.7, dated Apr. 17, 2008.
- Cormen, T. H., et al., "Introduction to Algorithms", The MIT Press, Cambridge, Massachusetts, (1994), 219-243, 991-993.
- Crovella et al., Dynamic server selection in the Internet, 3rd IEEE Workshop on the Arch. and Implementation of High Performance Computer Sys. '95, pp. 158-162, Aug. 1995.
- Danzig, P. B., et al., "An analysis of wide-area name server traffic: a study of the Internet Domain Name System," Conf. Proc. Communications Architectures & Protocols (Aug. 1992). D. Oran, Ed. SIGCOMM '92. ACM Press, New York, NY, 281-292.
- De Bra, P.M.E., et al., "Information Retrieval in the World Wide Web: Making Client-Based Searching Feasible", Computer Networks and ISDN Systems, NL, North Holland Publishing, Amsterdam, vol. 27, No. 2, ISSN: 0169-7552, (Nov. 1, 1994), 183-192.
- Deering, S. E., et al., "Multicast routing in datagram internetworks and extended LANs," ACM Trans. Comput. Syst. 8, 2 (May. 1990), 85-110.
- Devine, R., "Design and Implementation of DDH: A Distributed Dynamic Hashing Algorithm", In Proc. 4th Int'l Conf. on Foundations of Data Organizations and Algorithms, (Oct. 1993), 101-114.
- Doi, K., "Super Proxy Script—How to make distributed proxy servers by URL hashing," Sharp Corp., <http://naragw.sharp.co.jp/sps/>, dates unknown (1996-2000), download Jul. 7, 2007.
- European Patent Application No. 02799672.7, file history as of Apr. 23, 2010 (downloaded from EPO).
- Feeley, M., et al., "Implementing Global Memory Management in a Workstation Cluster", In Proc. 15th ACM Symp. on Operating Systems Principles, (Dec. 1995), 201-212.
- Floyd, S., et al., "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing", In Proc. of ACM SIGCOMM '95, 342-356, Aug. 1995.
- Fox, A., "A Framework for Separating Server Scalability and Availability from Internet Application Functionality," PhD thesis, University of California, Berkeley, Dec. 1998.
- Fox, A., et al., "Cluster-based scalable network services", Proc. 16th ACM Symp. on Operating Systems Principles (Saint Malo, France, Oct. 5-8, 1997). W. M. Waite, Ed. SOSP '97. ACM Press, New York, NY, 78-91.
- Fredman, M., et al., "Storing a Sparse Table with O(1) Worst Case Access Time", J. ACM, vol. 31, No. 3, (Jul. 1984), 538-544.
- Gadde, S., et al., "Reduce, reuse, recycle: An approach to building large internet caches," in Workshop on Hot Topics in Operating Systems (HotOS), Apr. 1997, pp. 93-98.
- Goldschmidt, et al., "Load Distribution for Scalable Web Servers: Summer Olympics 1996—A Case Study," In Proc. 8th IFIP/IEEE Int'l Workshop on Distributed Systems: Operations and Management, Sydney, Australia, Oct. 1997.
- Grigni, M., et al., "Tight Bounds on Minimum Broadcasts Networks", SIAM J. Disc. Math. 4 (May 1991), 207-222.
- Gulbrandsen, A., et al., "A DNS RR for specifying the location of services (DNS SRV)", Network Working Group, RFC 2052, Oct. 1996.
- Guyton et al., "Locating nearby copies of replicated Internet servers," Proc. ACM SIGCOMM '95, pp. 288-298, Oct. 1995.
- Gwertzman, J., et al., "The Case for Geographical Push-Caching", Proc. Workshop on Hot OS '95, (May 4, 1995), 51-55.
- Gwertzman, J., et al., "World-Wide Web Cache Consistency", Proc. 1996 USENIX Tech. Conf., pp. 141-151, San Diego, CA, Jan. 1996.
- International Preliminary Examination Report (IPER), Form PCT/IPEA/409, for PCT/US02/30921, Oct. 2005 [5 pgs.].
- International Search Report (ISR), Form PCT/ISA/210, for PCT/US02/30921, Jan. 28, 2004 [5 pgs.].
- Japanese Patent Office (JPO) "Notice of Reasons for Rejection", for Japanese Patent Application No. 2003-531370, Jul. 17, 2007, "Organized Translation" provided by foreign associate [2 pgs.].
- Japanese Patent Publication JP 10-70571, Jun. 1997, International Business Machines Corporation, English translation provided with Office Action [34 pgs.].
- Japanese Patent Publication JP 5-130144, Published May 25, 1993, Fujitsu Ltd., English translation provided with Office Action [9 pgs.].
- Jeffery, C., et al., "Proxy sharing proxy servers." In Proc. IEEE etaCOM Conf., pp. 116-119, May 1996.

US 8,645,517 B2

Page 6

(56)

References Cited

OTHER PUBLICATIONS

- Karger, D., et al., "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web", in Proc. 29th Annual ACM Symp. on Theory of Computing, (May 1997), 654-663.
- Korean Intellectual Property Office (KIPO), Notice of Preliminary Rejection (Non-Final) for Korean Patent Application No. KR 10 2004 7004613, Apr. 24, 2009, English translation (provided by foreign associate) [5 pgs.].
- Korkea-aho, M. (Nov. 5, 1995). Scalability in Distributed Multimedia Systems, Technical report TKO-B128, Helsinki University of Technology.
- Krishnamurthy, B. et al., Study of piggyback cache validation for proxy caches in the World Wide Web, in: Symp. on Internet Technology and Systems, USENIX Association, Dec. 1997.
- Kwan et al., NCSA's World Wide Web Server: Design and Performance, IEEE, pp. 68-74, Nov. 1995.
- Litwin, W., et al., "LH*—A Scalable, Distributed Data Structure", ACM Trans. on Database Systems, vol. 21, No. 4, pp. 480-525, Dec. 1996.
- Luotonen et al., World-Wide Web Proxies, CERN, Apr. 1994 (modified May 24, 1994).
- Malpani, R., et al., "Making World Wide Web Caching Servers Cooperate", in Proc. 4th Int'l. World Wide Web Conf. (Dec. 1995), 10 pages (downloaded from <http://www.w3.org/Conferences/WWW4/Papers/59/> on Jul. 7, 2007).
- Mockapetris et al., "Development of the Domain Name System," Proc. SIGCOMM '88 Computer Communications Review, vol. 18, No. 4, Aug. 1988.
- Mockapetris, P., RFC 1034: Domain Names—Concepts and Facilities, Nov. 1987.
- Mockapetris, P., RFC 1035: Domain Names—Implementation and Specification, Nov. 1987.
- Mourad et al., "Scalable Web Server Architectures," iscc, 2nd IEEE Symposium on Computers and Communications (ISCC '97), Jul. 1997, pp. 12-16.
- Nisan, N. 1990. Pseudorandom generators for space-bounded computations. In Proc. 22nd Annual ACM Symp. on theory of Computing (Baltimore, MD, U.S., May 13-17, 1990), H. Ortiz, Ed. STOC '90. ACM Press, New York, NY, 204-212.
- Office Action dated Dec. 8, 2009 in Japanese Application No. 2001-504633 (based on PCT/US001/16732, titled "On-Demand Overlay Routing for Computer-based communication networks," Inventor: Ricciulli, Livio) [in Japanese, 2 pgs., with 2 page English translation]. Office Action for Chinese (PRC) patent application No. 02821371.8, from the Patent Office of the People's Republic of China, dated Aug. 3, 2010 (non-official translation, 9 pgs., including 2 pg. summary). Office Action from Chinese Patent Office for application non. 02821371.8, Jan. 29, 2010, in Chinese with translation.
- Office Action from Chinese Patent Office for application non. 02821371.8, Jan. 4, 2008, in Chinese with translation.
- Office Action from Chinese Patent Office for application non. 02821371.8, May 8, 2009, in Chinese with translation.
- Oguchi et al., A Study of Caching Proxy Mechanisms Realized on Wide Area Distributed Networks, High Performance Distributed Computing, 5th Intl Symp., pp. 443-449, Aug. 1996.
- Palmer, M., et al., "Fido: A Cache that Learns to Fetch", In Proc. the 17th Int'l Conf. on Very Large Data Bases, (Sep. 1991), 255-264.
- Panigrahy, R., "Relieving Hot Spots on the World Wide Web", Master's thesis, MIT EECS, Jun. 1997, pp. 1-66.
- Patent Abstracts of Japan, "Server System for Internet", Pub. No. 10-027148, pub. date Jan. 27, 1998, Applicant: Hitachi, computer translation, 12 pgs.
- Patent Abstracts of Japan, "Communication Connection Method With a Plurality of Hosts having Common Identifier", Appln. No. JP 08-20114, Pub. No. 10-093552, Applicant Nippon Telegraph & Telephone, published Apr. 1998.
- Patent Abstracts of Japan, "Domain Name Solving Method and Domain Name Solving System", Appln. No. JP19990223763, Pub. No. 2001-053793, Applicant Nippon Telegraph & Telephone, published Feb. 23, 2001.
- Peleg, D., et al., "The Availability of Quorum Systems", Information and Computation, 123, (Dec. 1995), 210-223.
- Petri S., et al., "Load Balancing and Fault Tolerance in Workstation Clusters. Migrating Groups of Communicating Processes.", Operating Systems Review, vol. 29, No. 4, Oct. 1995, pp. 25-36.
- Plaxton, G. C., et al., "Fast Fault-Tolerant Concurrent Access to Shared Objects", In Proc. 37th IEEE Symp. of Foundations of Computer Science, (Oct. 1996), 570-579.
- Postel, RFC 1591—Domain Name System Structure and Delegation, Mar. 1994.
- Rabin, M. O., 1989, "Efficient dispersal of information for security, load balancing, and fault tolerance," J. ACM 36, 2 (Apr. 1989), 335-348.
- Rabinovich, M. et al., "Dynamic Replication on the Internet Work Project No. 3116-17-7006", AT&T Labs Research, Mar. 5, 1998.
- Rabinovich, M. et al., RaDaR: A Scalable Architecture for a Global Web Hosting Service, WWW8, May 1999.
- Ross, K. W., "Hash-Routing for Collections of Shared Web Caches", IEEE Network Magazine, 11, 7:37-44, Nov.-Dec. 1997.
- Schemers, R., "Ibnamed—A load balancing name server written in Perl," 1995 LISA IX—Sep. 17-22, 1995—Monterey, CA.
- Schuba, Christoph; "Addressing Weaknesses in the Domain Name System Protocol," COAST Laboratory, Dept. of Computer Sciences, Purdue University; West Lafayette, IN; Aug. 1993, p. 1-87.
- Smith, "What can Archives offer the World Wide Web?", Technical Report 11, University of Kent, Computing Laboratory, University of Kent, Canterbury, UK, Mar. 1994.
- SNMPV2 Working Group J Case SNMP Research et al: "Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2); rfc1907.txt" IETF Standard, Internet Engineering Task Force, IETF, CH, Jan. 1996.
- Tarjan, Robert E., et al., "Storing a Sparse Table", Commun.ACM, 22, 11, (Nov. 1979), 606-611.
- Thaler, D. G. and Ravishankar, C. V. 1998. Using name-based mappings to increase hit rates. IEEE/ACM Trans. Netw. 6, 1 (Feb. 1998), 1-14.
- U.S. P.T.O. Non-Final Office Action in U.S. Appl. No. 11/980,689, May 20, 2009.
- Vitter, J. S., et al., "Optimal Prefetching via Data Compression," Proc. 32nd Annual IEEE Symposium on Foundations of Computer Science (Oct. 1991).
- Vixie, Paul; "Name Server Operations Guide for BIND," Internet Software Consortium; La Honda, CA; p. SMM:10-2—SMM:10-30 (undated, 1996).
- Wessels, Intelligent Caching for World-Wide Web Objects, Masters Thesis, University of Colorado, Jan. 1995 (also presented at INET '95 in Jun. 1995).
- Yao, A. C. 1981. Should Tables Be Sorted?. J. ACM 28, 3 (Jul. 1981), 615-628.
- "Extended European Search Report dated Dec. 6, 2011", App. No. 10178695.2, 7 pgs.

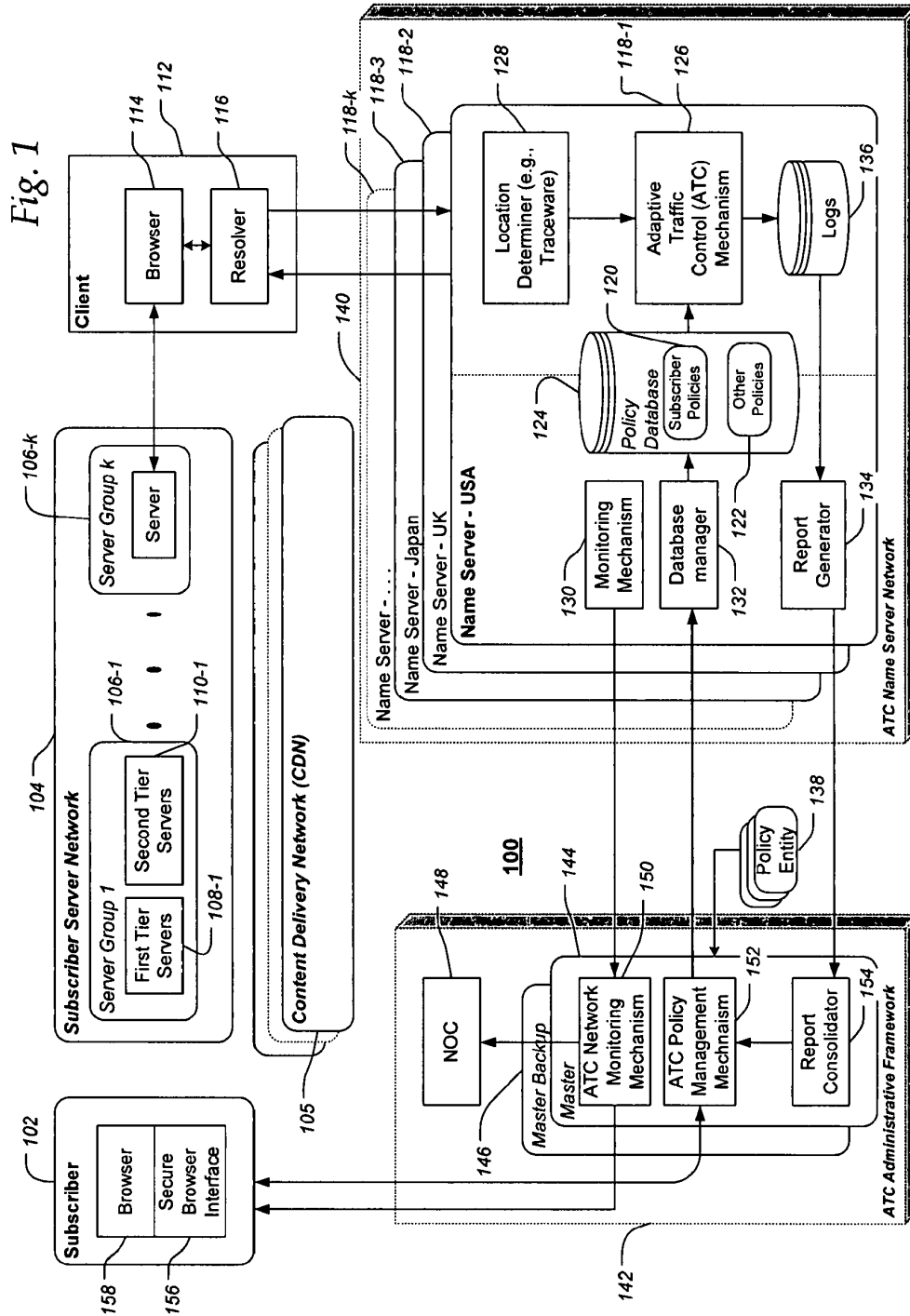
* cited by examiner

U.S. Patent

Feb. 4, 2014

Sheet 1 of 31

US 8,645,517 B2



U.S. Patent

Feb. 4, 2014

Sheet 2 of 31

US 8,645,517 B2

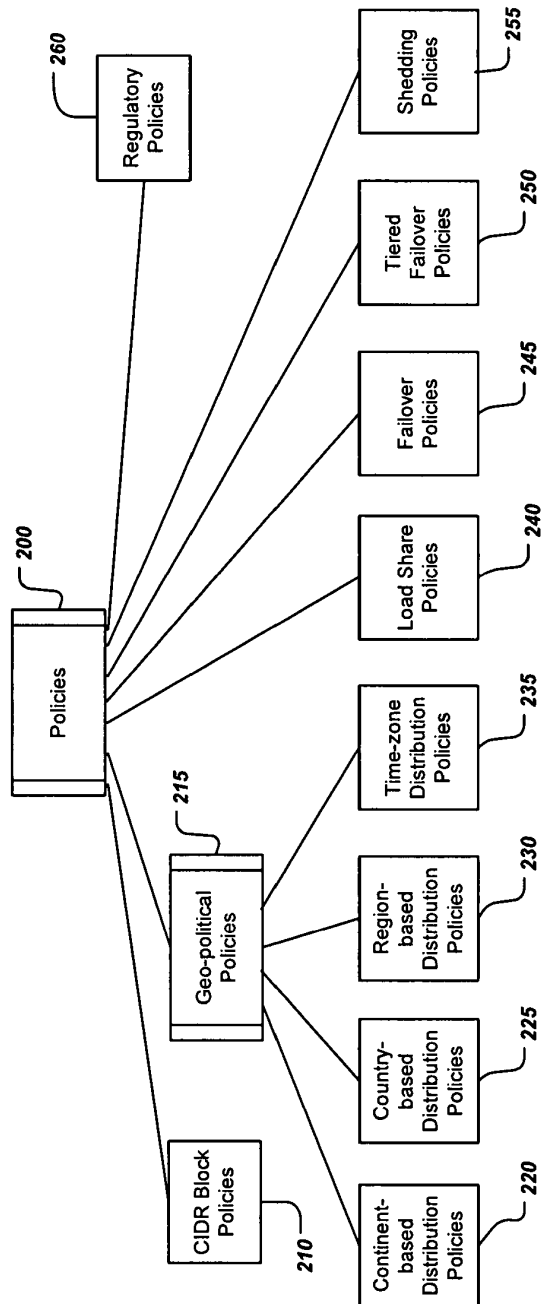


Fig. 2

U.S. Patent

Feb. 4, 2014

Sheet 3 of 31

US 8,645,517 B2

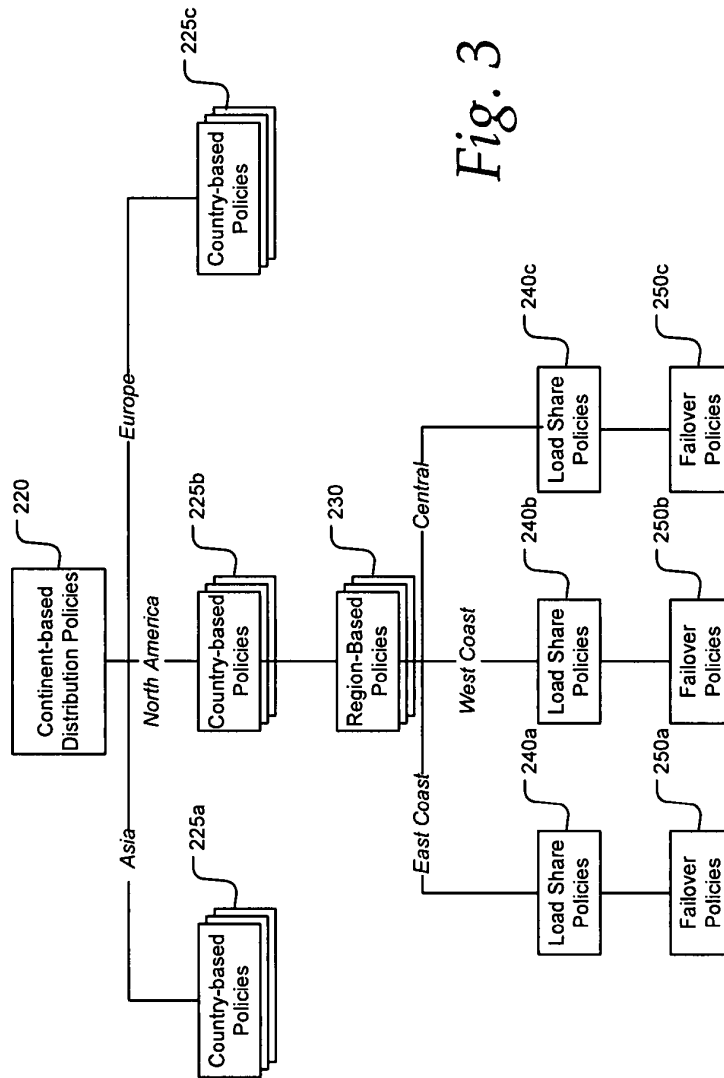


Fig. 3

U.S. Patent

Feb. 4, 2014

Sheet 4 of 31

US 8,645,517 B2

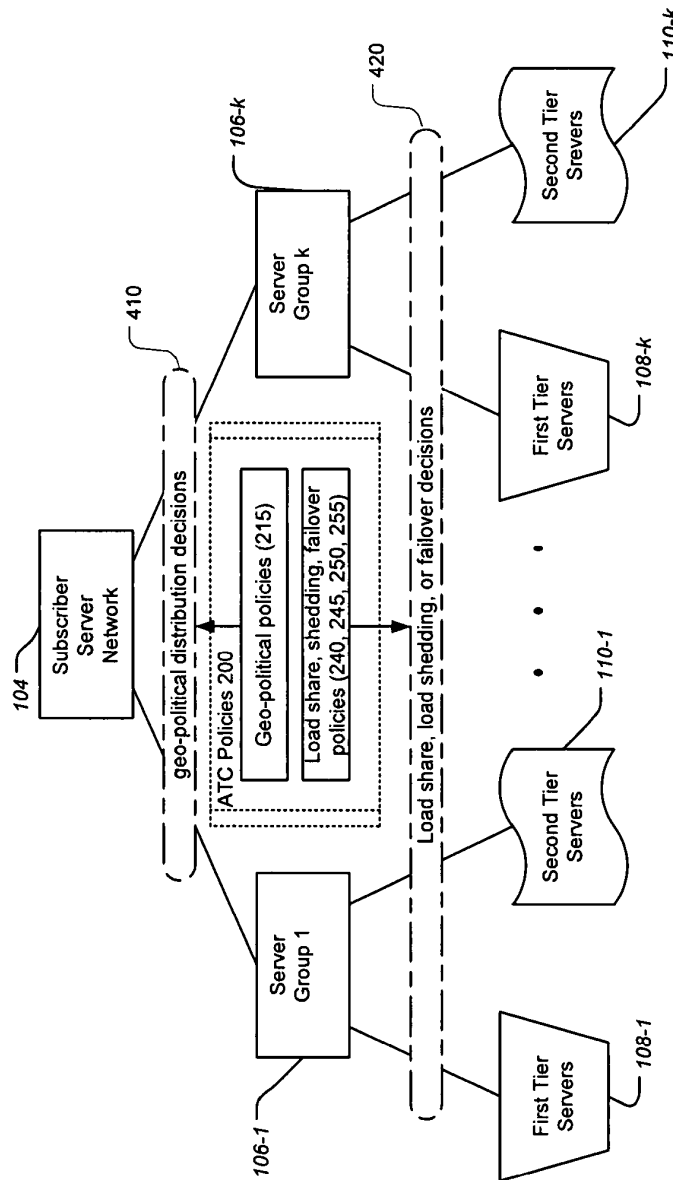


Fig. 4

U.S. Patent

Feb. 4, 2014

Sheet 5 of 31

US 8,645,517 B2

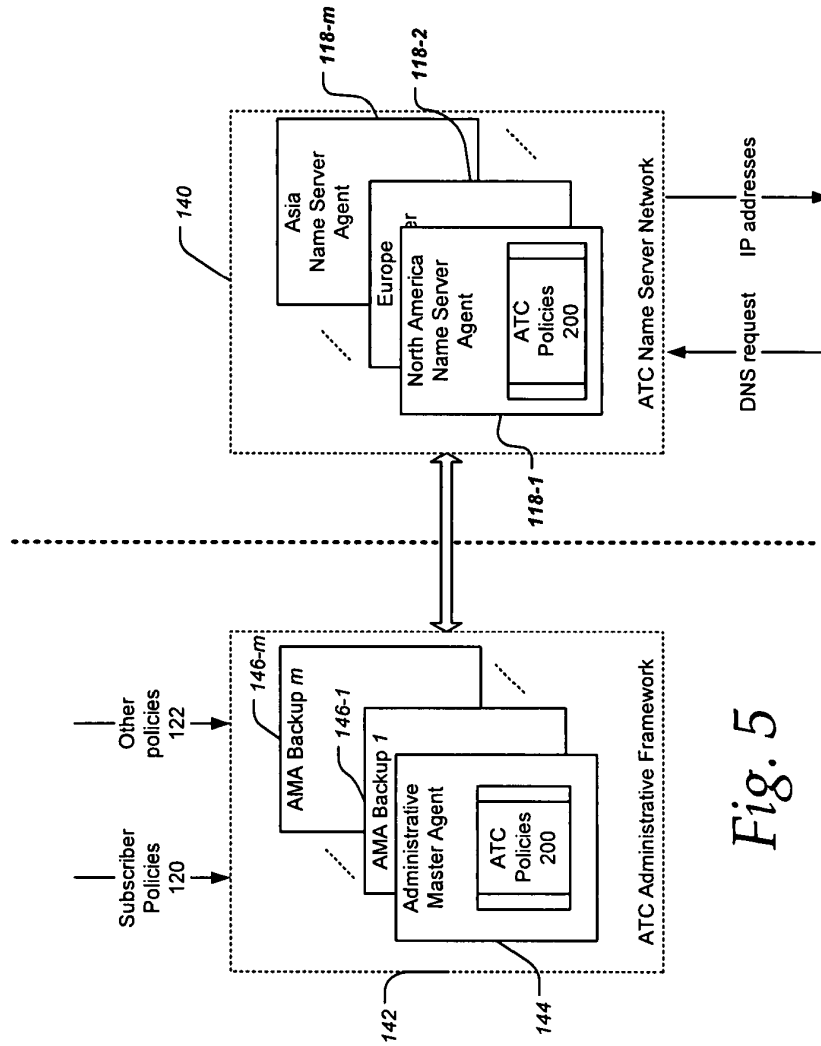


Fig. 5

U.S. Patent

Feb. 4, 2014

Sheet 6 of 31

US 8,645,517 B2

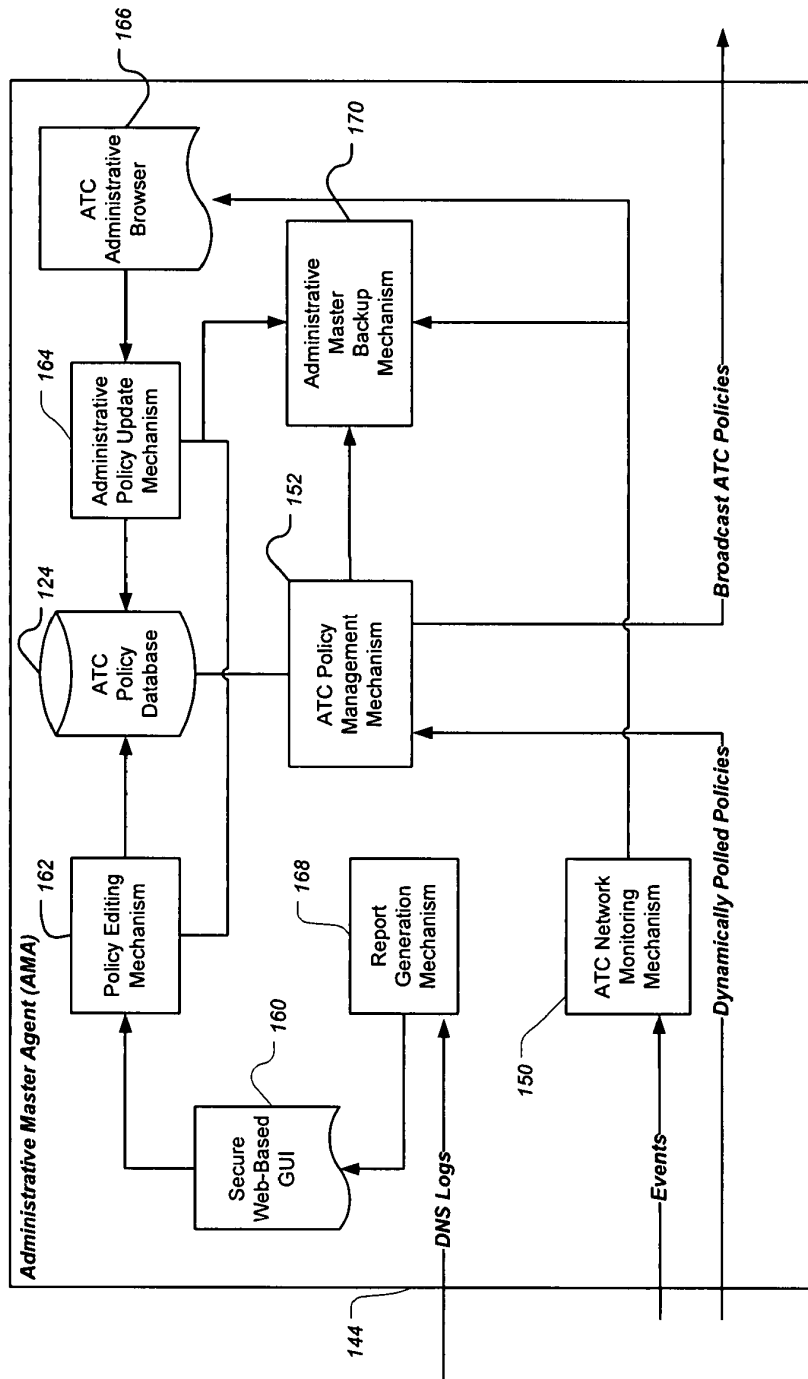


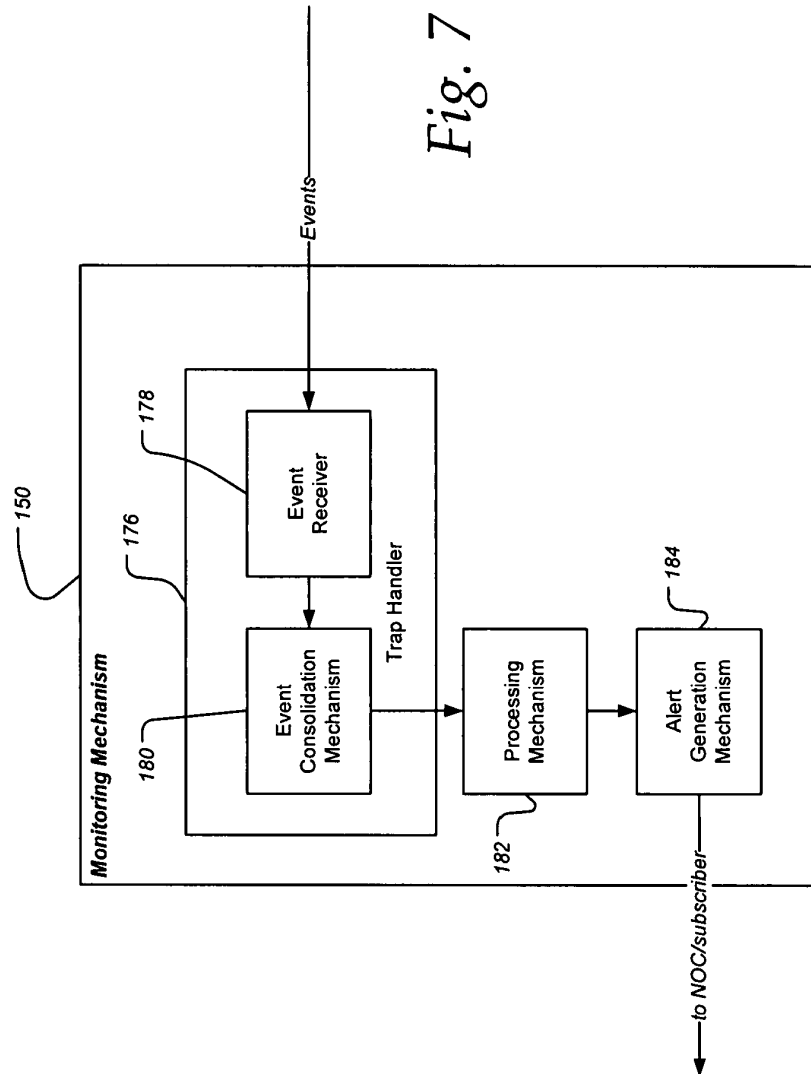
Fig. 6

U.S. Patent

Feb. 4, 2014

Sheet 7 of 31

US 8,645,517 B2

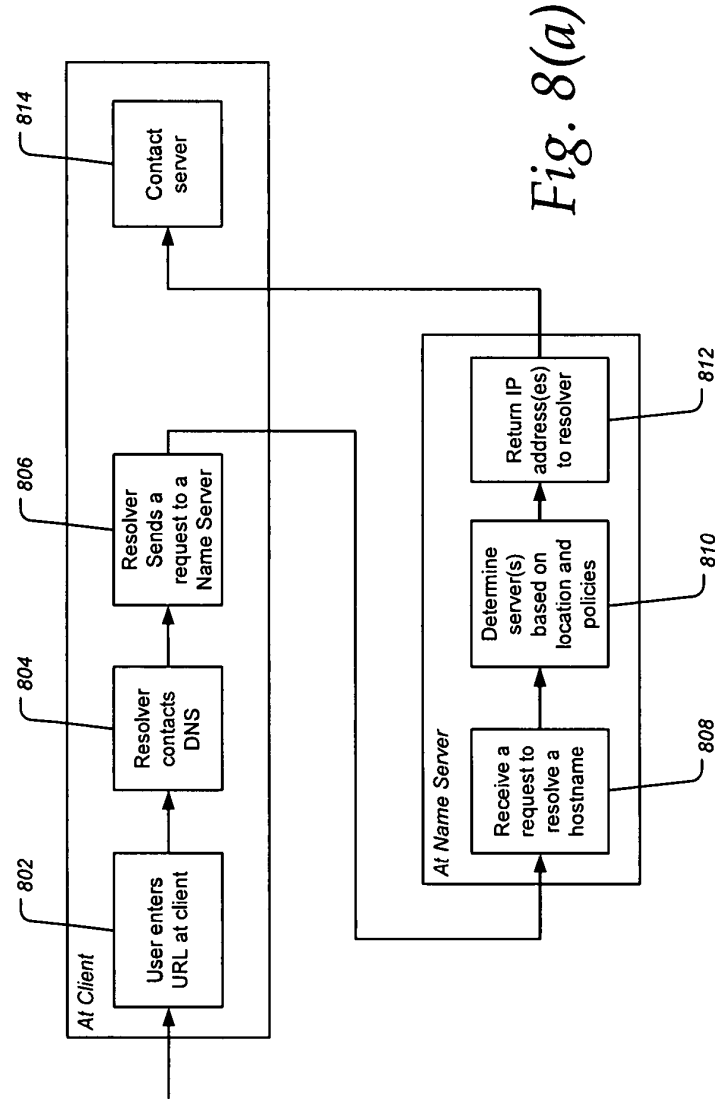


U.S. Patent

Feb. 4, 2014

Sheet 8 of 31

US 8,645,517 B2



U.S. Patent

Feb. 4, 2014

Sheet 9 of 31

US 8,645,517 B2

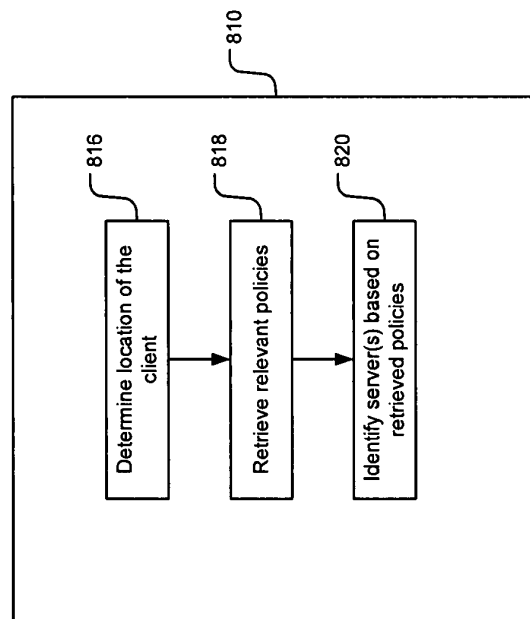


Fig. 8(b)

U.S. Patent

Feb. 4, 2014

Sheet 10 of 31

US 8,645,517 B2

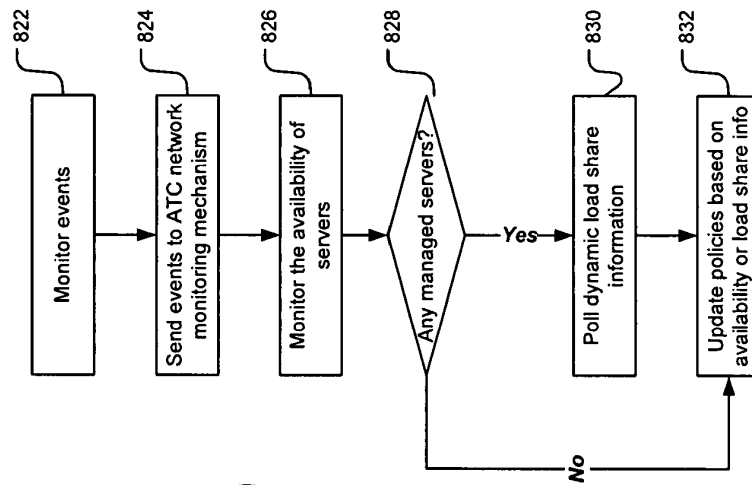


Fig. 8(c)

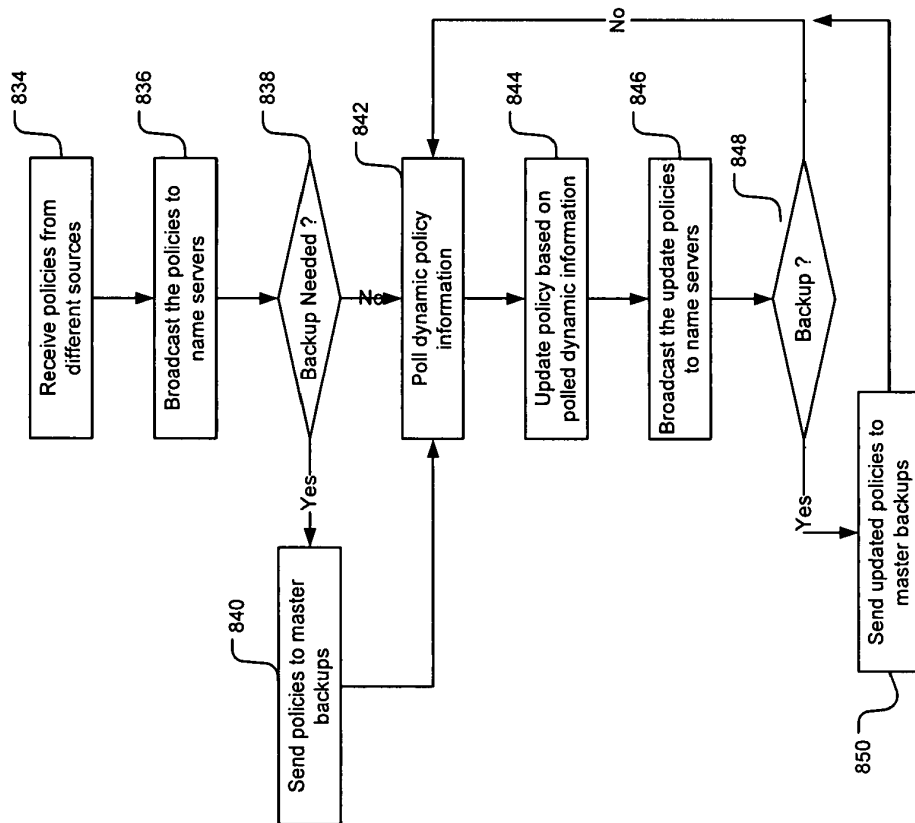
U.S. Patent

Feb. 4, 2014

Sheet 11 of 31

US 8,645,517 B2

Fig. 8(d)



U.S. Patent

Feb. 4, 2014

Sheet 12 of 31

US 8,645,517 B2

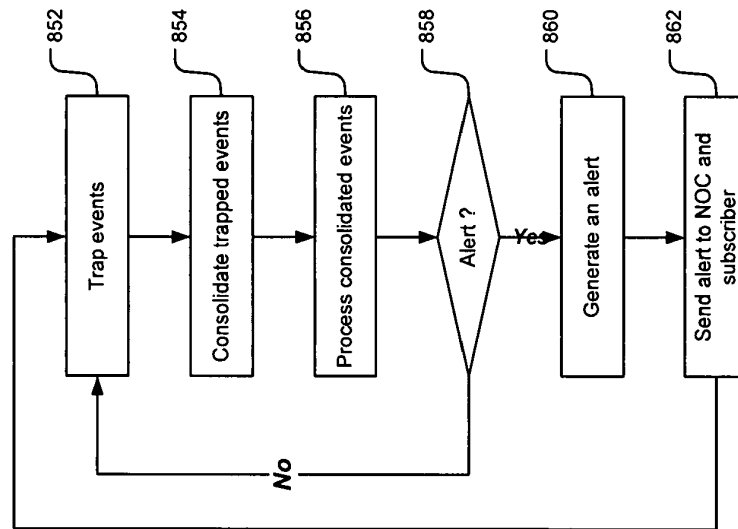


Fig. 8(e)

U.S. Patent

Feb. 4, 2014

Sheet 13 of 31

US 8,645,517 B2

MODIFY STATIC SERVER(S)						
NICKNAME (OPTIONAL)	IP ADDRESS	TTL	LOAD SHARE	SHED FRACTION	ON LINE	
NEW YORK	10.0.0.1	5m	1.0	0.2	☒	
LONDON	10.0.0.2	5m	1.0	0.3	☒	
		5m	1.0	0.0	☒	
		5m	1.0	0.0	☒	
		5m	1.0	0.0	☒	

OK CANCEL | DELETE | CHECKED STATIC SERVER(S)

FIG. 9(a)

MODIFY DYNAMIC SERVER(S)					
NICKNAME	IP ADDRESS	TTL	SERVER STATUS URL	USER NAME	PASSWORD
LONDON	10.0.0.3	5m	http://SERVER	PAUL	*****
		5m	http://SERVER		
		5m	http://SERVER		
		5m	http://SERVER		
		5m	http://SERVER		

OK CANCEL | DELETE | CHECKED DYNAMIC SERVER(S)

FIG. 9(b)

U.S. Patent

Feb. 4, 2014

Sheet 14 of 31

US 8,645,517 B2

MODIFY CNAME(S)		TTL	
NICKNAME (OPTIONAL)	CNAME		
SYDNEY	CUSTOMER.FOOTPRINT.NET	5m	
		5m	
		5m	
		5m	
		5m	

CHECKED CNAME(S)

FIG. 9(c)

U.S. Patent

Feb. 4, 2014

Sheet 15 of 31

US 8,645,517 B2

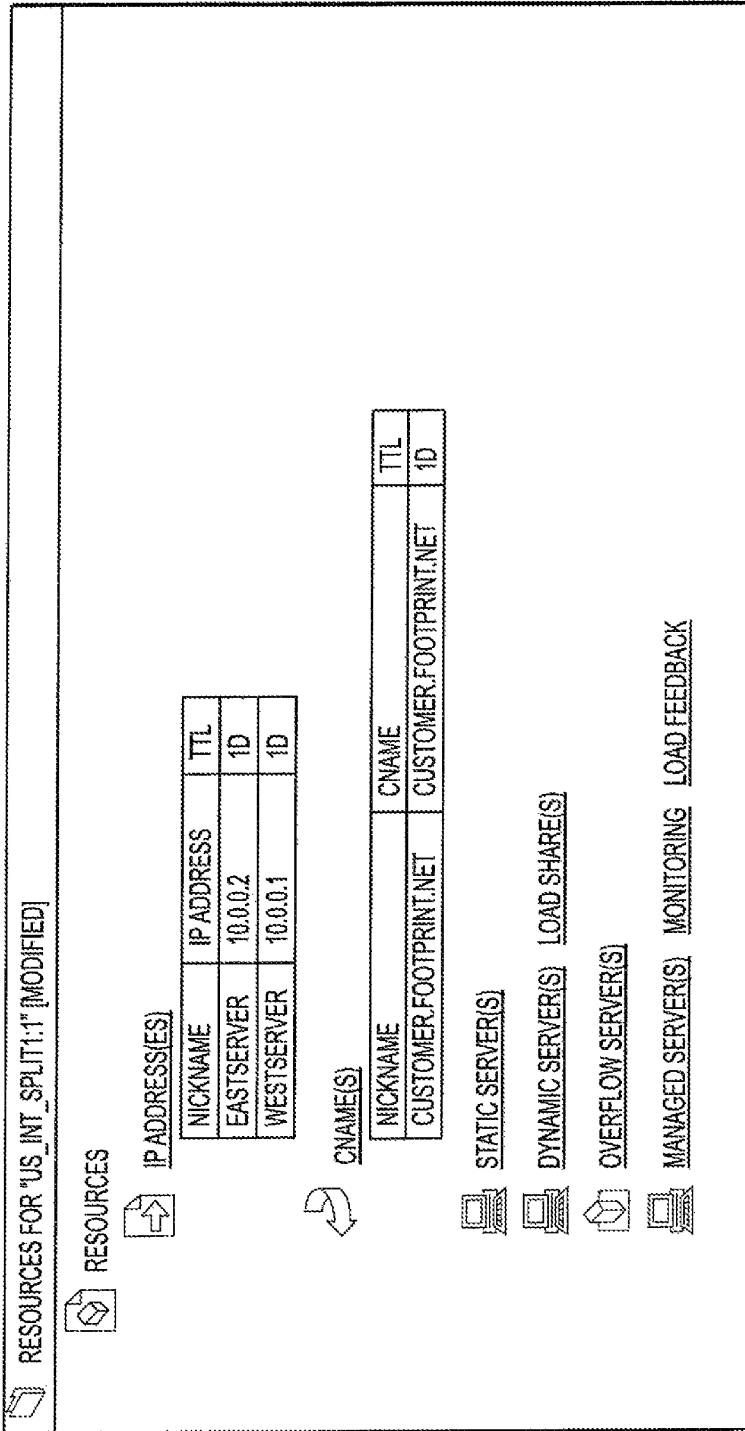


FIG. 10(a)

U.S. Patent

Feb. 4, 2014

Sheet 16 of 31

US 8,645,517 B2

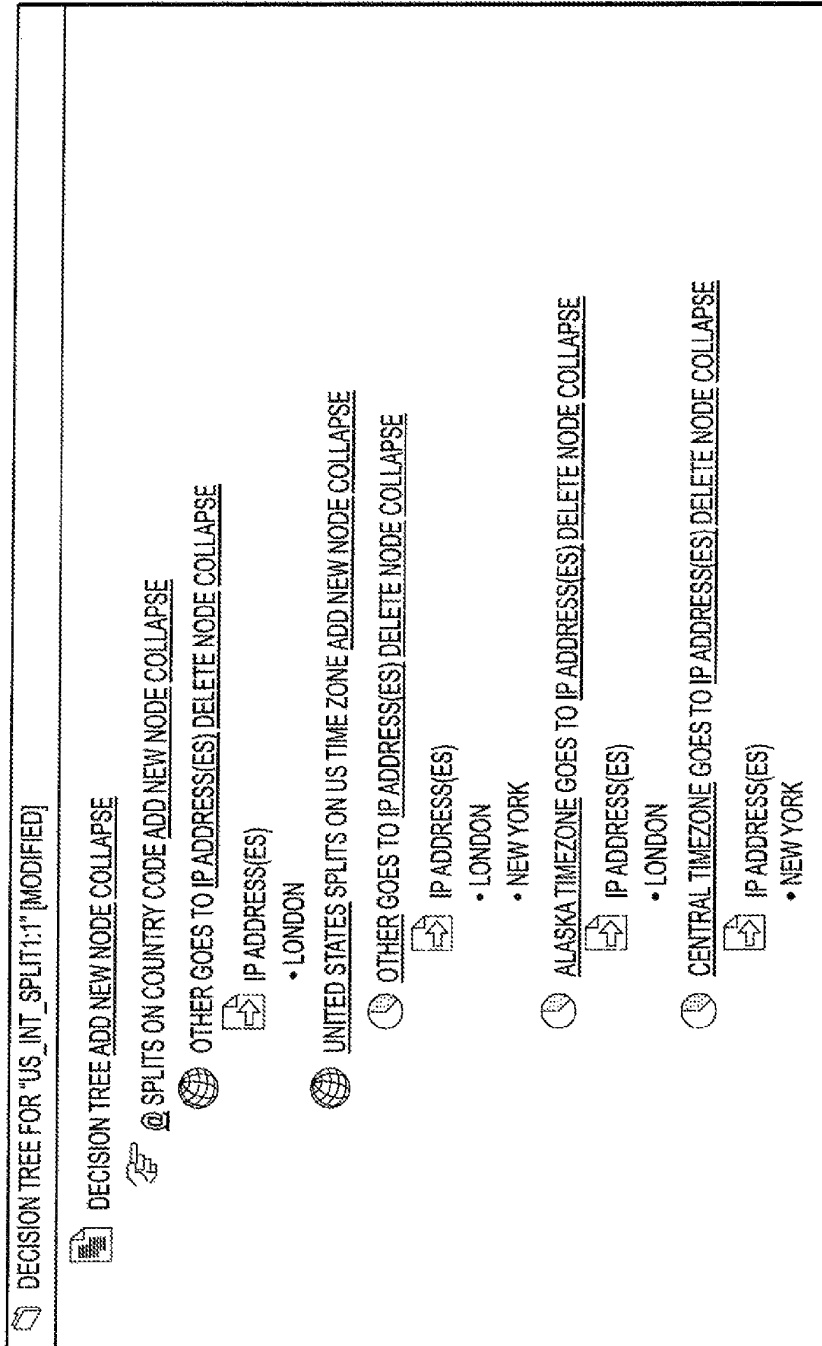


FIG. 10(b)

U.S. Patent

Feb. 4, 2014

Sheet 17 of 31

US 8,645,517 B2

RESOURCES FOR "US_INT_SPLIT:1"[MODIFIED]

RESOURCES

IP ADDRESS(ES)

NICKNAME	IP ADDRESS	TTL
LONDON	10.2.2.2	1D
NEW YORK	10.0.0.1	1D

CNAME(S)

NICKNAME	CNAME	TTL
SYDNEY	CUSTOMER.FOOTPRINT.NET	5M

STATIC SERVER(S)

DYNAMIC SERVER(S) LOAD SHARE(S)

NICKNAME	IP ADDRESS	TTL	SERVER STATUS	URL	DOWN LOAD SHARE	SHED FRACTION	DEFAULT LOAD SHARE	SHED FRACTION
LONDON	10.0.0.3	5M	HTTP://***.***@10.0.0.1/		0	0	1	0

OVERFLOW SERVER(S)

NICKNAME	IP ADDRESS/CNAME	TTL	LOAD SHARE
CDN SERVICE	CUSTOMER.FOOTPRINT.NET	5M	1

MANAGED SERVER(S) MONITORING LOAD FEEDBACK

FIG. 10(c)

U.S. Patent

Feb. 4, 2014

Sheet 18 of 31

US 8,645,517 B2

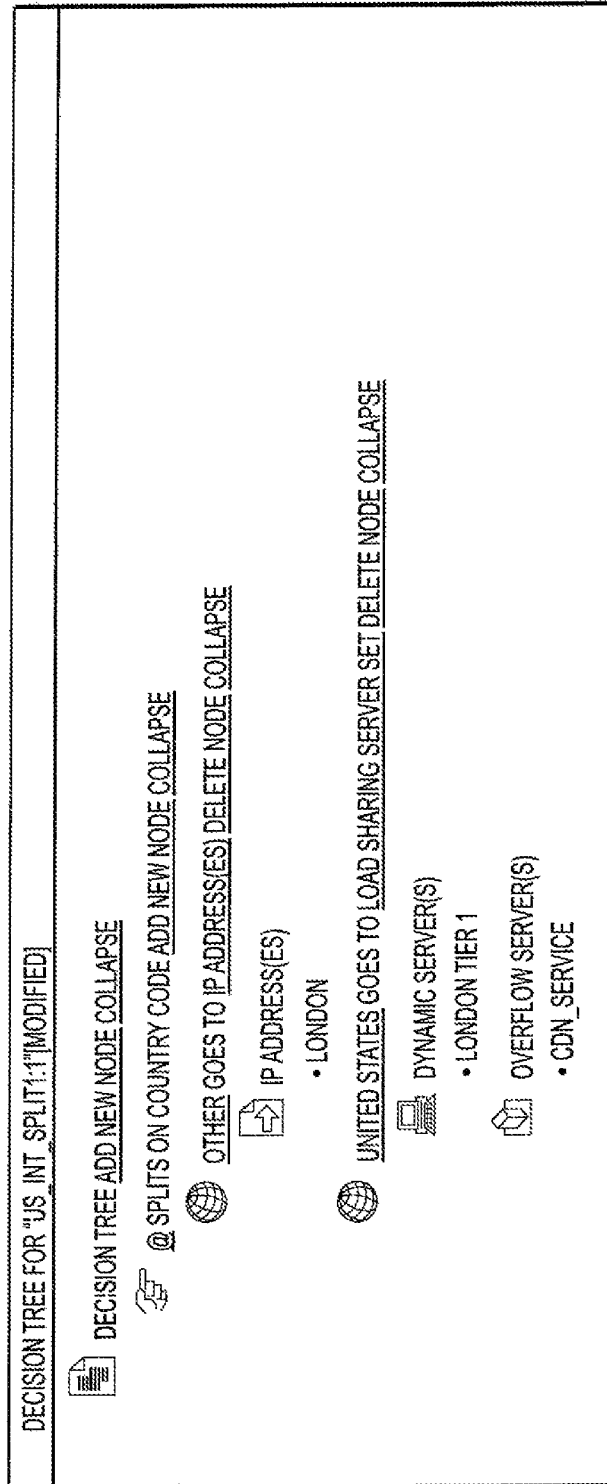


FIG. 10(d)

U.S. Patent

Feb. 4, 2014

Sheet 19 of 31

US 8,645,517 B2

RESOURCE FOR "EXAMPLE12"

RESOURCES

IP ADDRESS(ES)

CNAME(S)

MX RECORD(S)

STATIC SERVER(S)

DYNAMIC SERVER(S) LOAD SHARE(S)

OVERFLOW SERVER(S)

DL SUBSCRIBER.COM.C.FOOTPRINT.NET

DL SUBSCRIBER.COM.C.FOOTPRINT.NET

5M

1

MANAGED SERVER(S) MONITORING LOAD FEEDBACK

NICKNAME	IP ADDRESS	TTL	MONITORED	LOAD FEEDBACK	LOAD SHARE	SHED FRACTION
LONDON	10.0.0.3	5M	YES	NO	1	0
NEW YORK	10.0.0.1	5M	YES	NO	1	0
SAN FRANCISCO	10.0.0.2	5M	YES	NO	2.5	0

NS RECORD(S)

FIG. 10(e)

U.S. Patent

Feb. 4, 2014

Sheet 20 of 31

US 8,645,517 B2

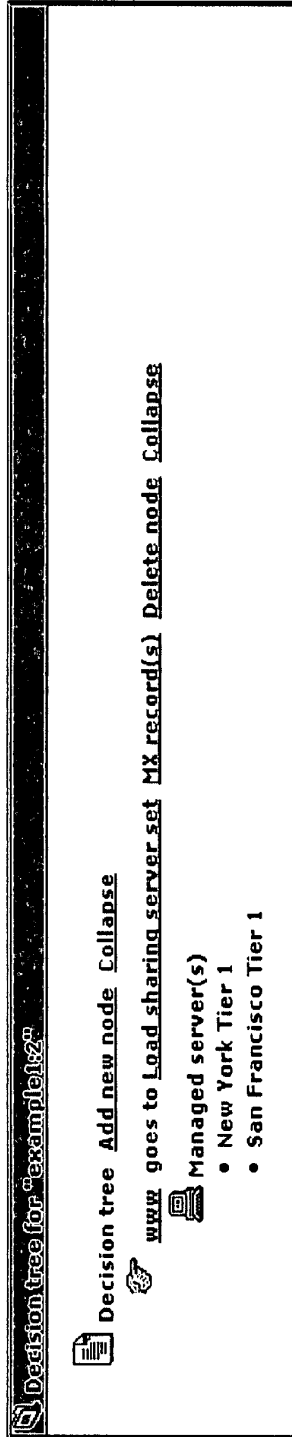


Fig. 10(f)

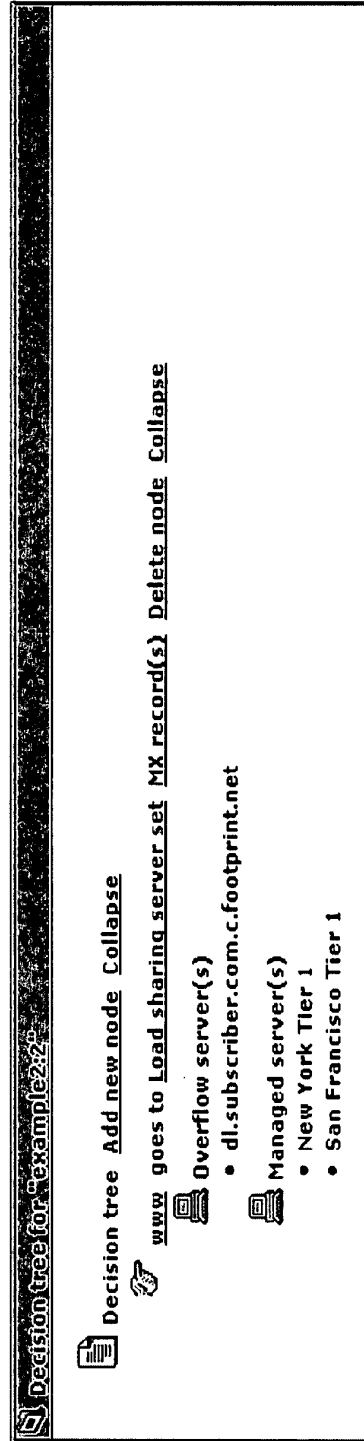


Fig. 10(g)

U.S. Patent

Feb. 4, 2014

Sheet 21 of 31

US 8,645,517 B2

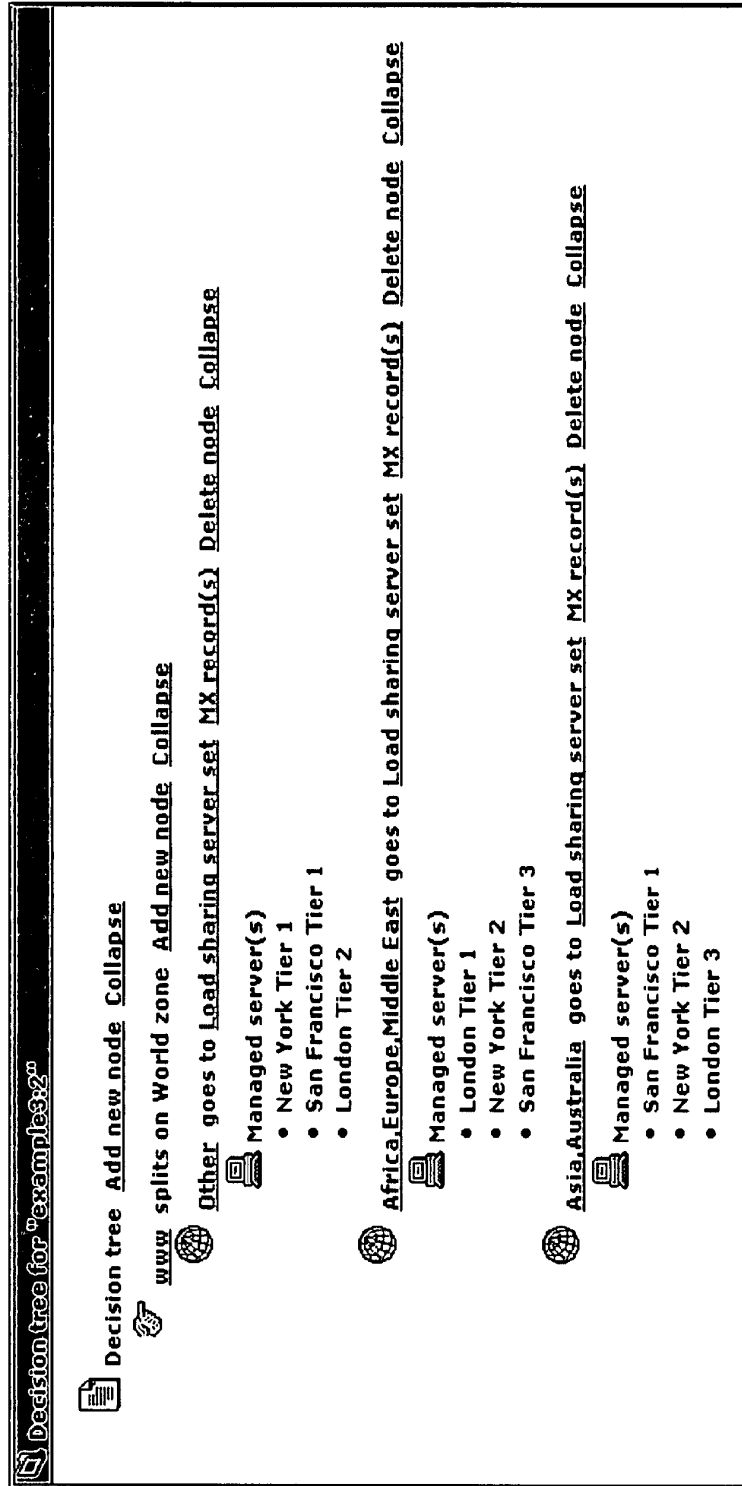


Fig. 10(h)

U.S. Patent

Feb. 4, 2014

Sheet 22 of 31

US 8,645,517 B2

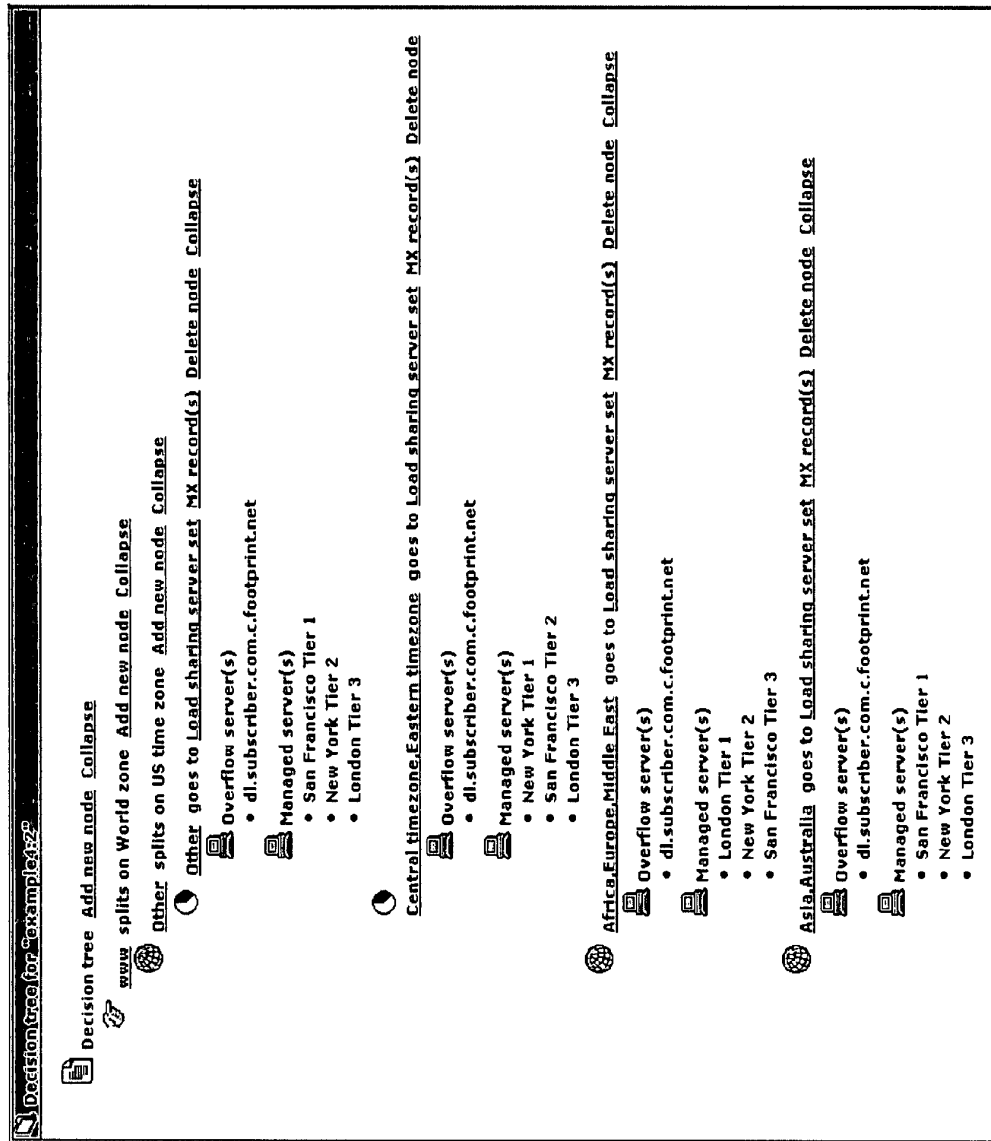


Fig. 10(i)

U.S. Patent

Feb. 4, 2014

Sheet 23 of 31

US 8,645,517 B2



RESOURCES



IP ADDRESS(ES)



CNAME(S)

NICKNAME	CNAME	TTL
INTERNATIONAL.DEMO.COM	INTERNATIONAL.DEMO.COM	5M
US.DEMO.COM	US.DEMO.COM	5M



STATIC SERVER(S)



OVERFLOW SERVER(S)

NICKNAME	IP ADDRESS / CNAME	TTL	LOAD SHARE
DEMO.CDN.COM	DEMO.CDN.COM	5M	1



MANAGED SERVER(S)

MONITORING		LOAD FEEDBACK				
NICKNAME	IP ADDRESS	TTL	MONITORED	LOAD FEEDBACK	LOAD SHARE	SHED FRACTION
NY	10.0.0.2	5M	NO	NO	1	0
SF	10.0.0.1	5M	NO	NO	2	0
UK	10.0.0.3	5M	NO	NO	1	0



NS RECORD(S)

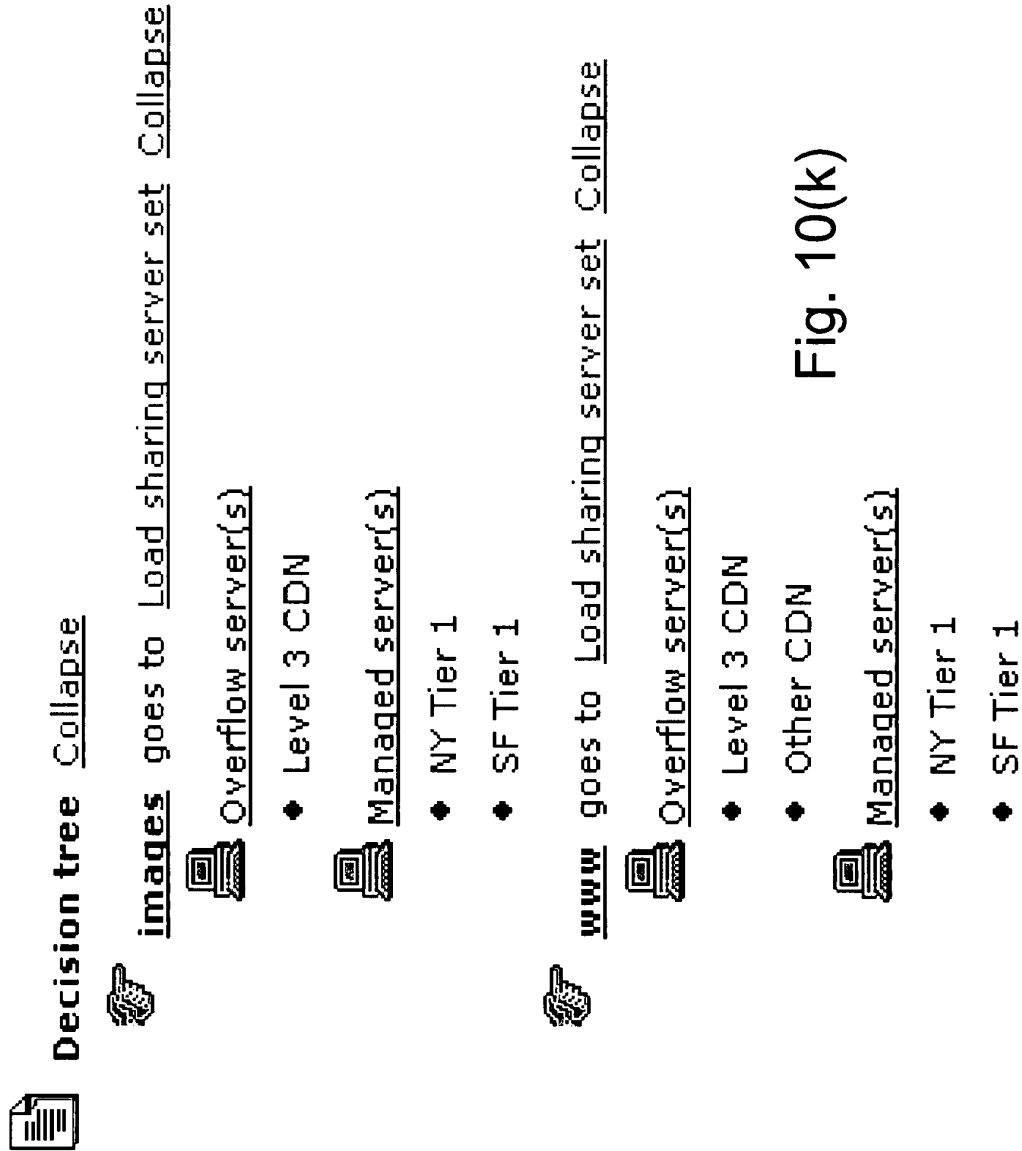
FIG. 10(j)

U.S. Patent

Feb. 4, 2014

Sheet 24 of 31

US 8,645,517 B2



U.S. Patent

Feb. 4, 2014

Sheet 25 of 31

US 8,645,517 B2

☒ MODIFY SERVER SET RESOURCES FOR 'www'									
LOAD SHARING SERVER SET PARAMETERS									
NUMBER OF ANSWERS REQUIRED 1				NUMBER OF ANSWERS DESIRED 1					
STATIC LOAD SHARING SERVER(S)									
NO RESOURCES DEFINED. ADD A RESOURCE WITH 'RESOURCES WINDOW'.									
MANAGED LOAD SHARING SERVER(S)									
TIER	NICKNAME	IP ADDRESS / CNAME	TTL	MONITORED	LOAD FEEDBACK	LOAD SHARE	SHED FRACTION		
1	NY	10.0.0.2	5M	NO	NO	1	0.8		
1	SF	10.0.0.1	5M	NO	NO	2	0.8		
OVERFLOW SERVER(S)									
	NICKNAME	IP ADDRESS / CNAME	TTL	LOAD SHARE					
☐ INCLUDE	LEVEL 3 CDN	DEMO.C.FOOTPRINT.NET	5M	5M					
☐ INCLUDE	OTHER CDN	DEMO.OTHERCDN.NET	5M	1					
CANCEL									

FIG. 10(I)

U.S. Patent

Feb. 4, 2014

Sheet 26 of 31

US 8,645,517 B2

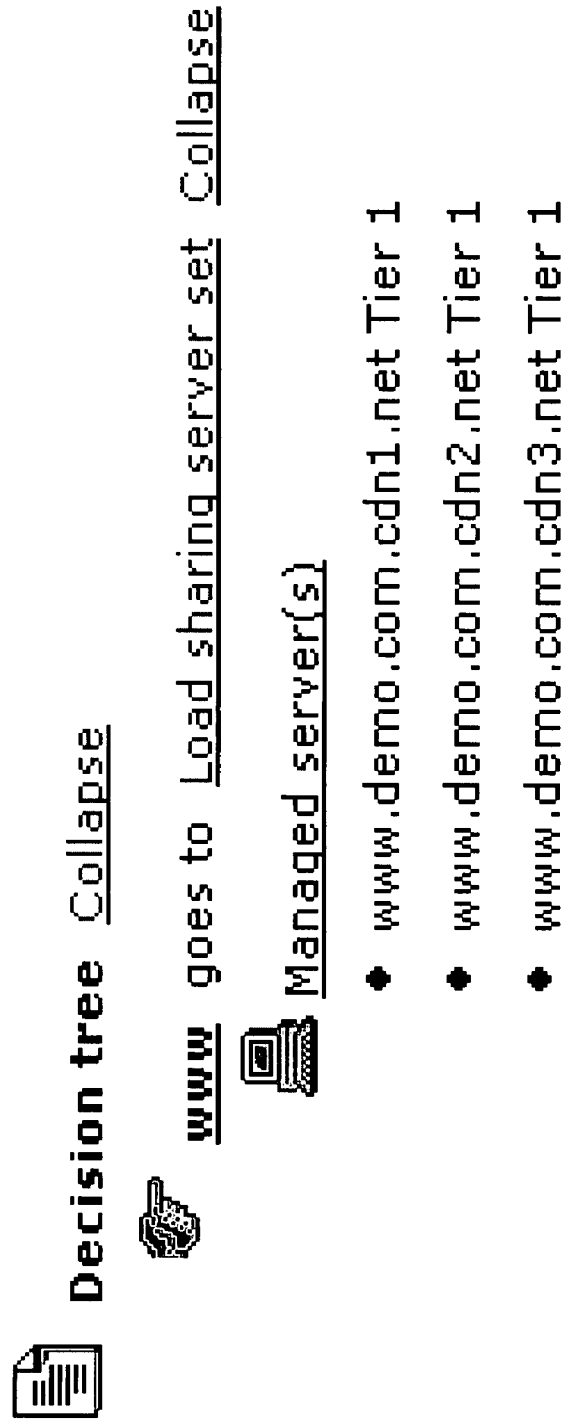


Fig. 10(m)

U.S. Patent

Feb. 4, 2014

Sheet 27 of 31

US 8,645,517 B2

MODIFY SERVER SET RESOURCES FOR "www"						
LOAD SHARING SERVER SET PARAMETERS						
NUMBER OF ANSWERS REQUIRED 1			NUMBER OF ANSWERS DESIRED 1			
STATIC LOAD SHARING SERVER(S)						
NO RESOURCES DEFINED. ADD A RESOURCE WITH "RESOURCES WINDOW".						
MANAGED LOAD SHARING SERVER(S)						
TIER	NICKNAME	IP ADDRESS / CNAME	TTL	MONITORED	LOAD FEEDBACK	LOAD SHARE SHED FRACTION
1	WWW.DEMO.COM.CDN1.NET	WWW.DEMO.COM.CDN1.NET	5M	NO	NO	0
1	WWW.DEMO.COM.CDN2.NET	WWW.DEMO.COM.CDN2.NET	5M	NO	NO	0
1	WWW.DEMO.COM.CDN3.NET	WWW.DEMO.COM.CDN3.NET	5M	NO	NO	0
OVERFLOW SERVER(S)						
NO RESOURCES DEFINED. ADD A RESOURCE WITH "RESOURCES WINDOW".						
CANCEL						

FIG. 10(n)

U.S. Patent

Feb. 4, 2014

Sheet 28 of 31

US 8,645,517 B2

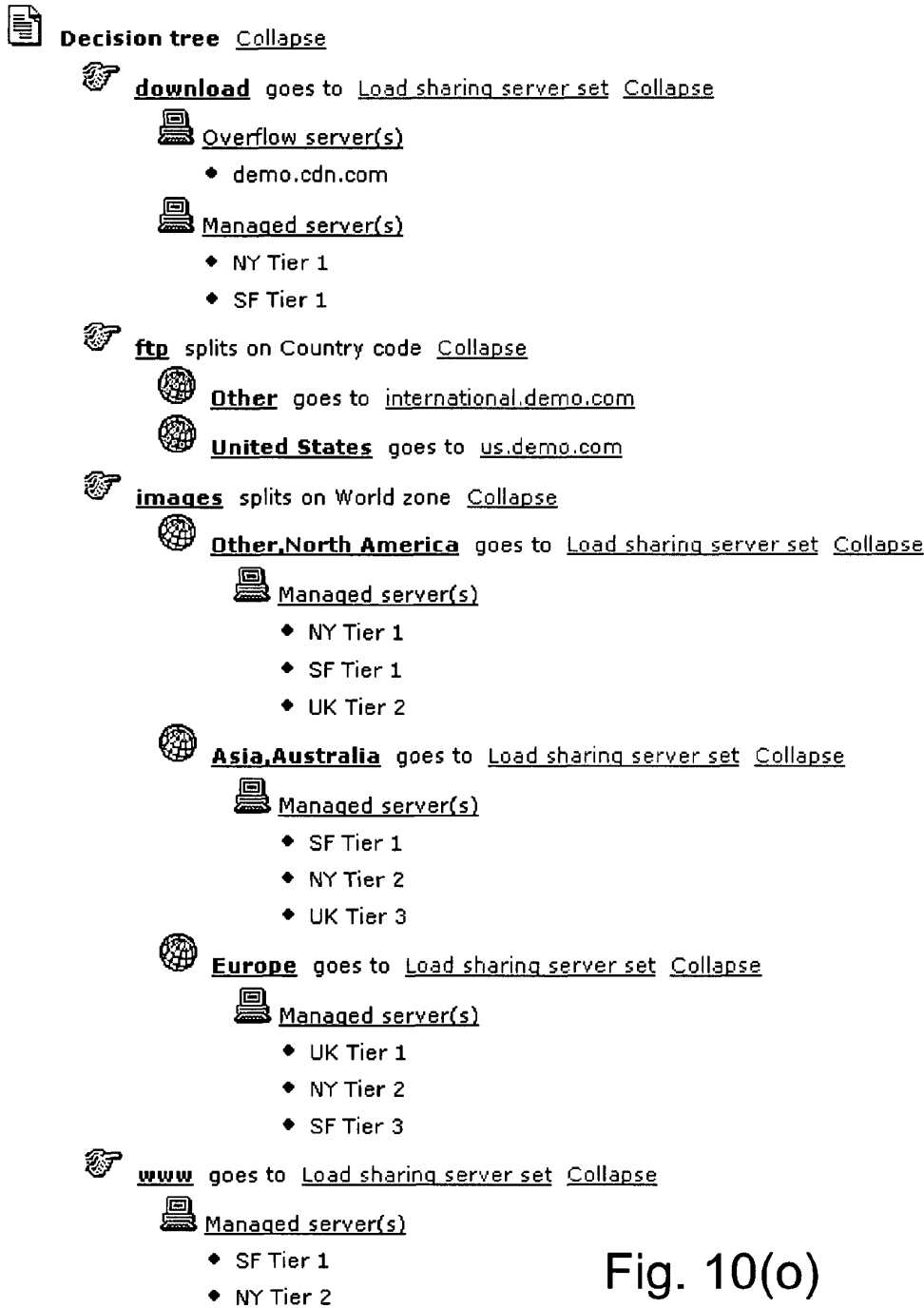


Fig. 10(o)

U.S. Patent

Feb. 4, 2014

Sheet 29 of 31

US 8,645,517 B2

VIEW ARCHIVED LOGS									
PARAMETERS									
DOMAIN NAME ORIGIN(S)			OUTPUT STYLE			TIME RESOLUTION			
[DOWNLOAD.SUBSCRIBER.COM]			[SEPARATE] [v] [v] DISPLAY			[DATA FOR SELECTED DOMAIN NAME ORIGIN(S) AND [v] IP/NAME QUERY: REPLIES WITH ANSWERS] [v] [v] AS [SEPARATE ITEMS] [v]			
TIME RANGE SELECTOR (GMT)									
☐ ABSOLUTE TIME RANGE									
START TIME		MONTH		DAY		YEAR		HOUR	
[SEP] [v]		[SEP] [v]		[1] [v]		[2002] [v]		[00] [v]	
END TIME		[SEP] [v]		[24] [v]		[2002] [v]		[03] [v]	
								MINUTE	
								[00] [v]	
								[20] [v]	
☒ RELATIVE TIME RANGE									
LAST DAY [v]									
[GRAPH]		[TABLE]		[EXPORT]		[DATA TO MS EXCEL SPREADSHEET]		[RETURN] [v]	
TO POLICY WINDOW									

FIG. 11(a)

U.S. Patent

Feb. 4, 2014

Sheet 30 of 31

US 8,645,517 B2

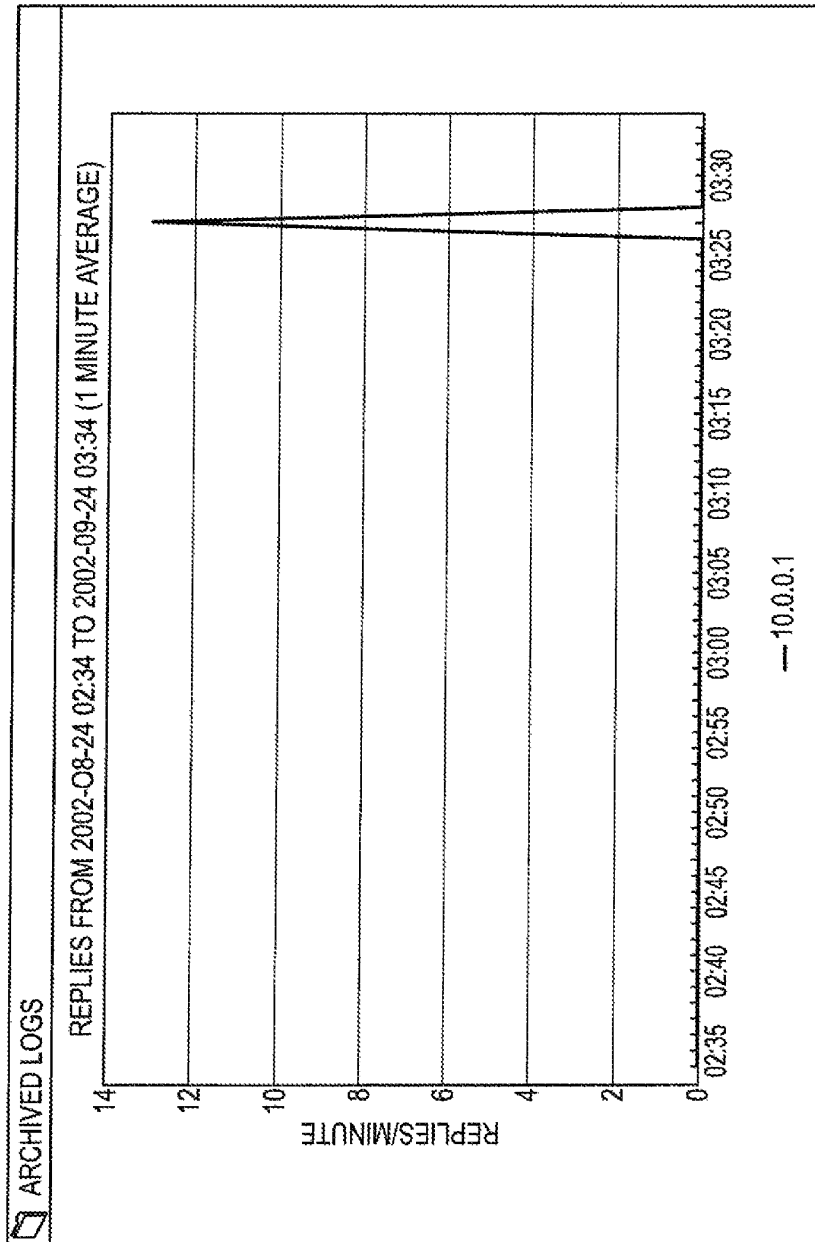


FIG. 11(b)

U.S. Patent

Feb. 4, 2014

Sheet 31 of 31

US 8,645,517 B2

ARCHIVED LOGS				
DOMAIN NAME ORIGIN(S) "DOWNLOAD.SUBSCRIBER.COM"				
TIME (GMT)	IP/CNAME QUERY: REPLIES WITH ANSWERS	NUMBER OF REPLIES	REPLIES PER MINUTE	
09/24/02 02:30:00 - 9/24/02 02:45:00	NO DATA FOUND			
09/24/02 02:45:00 - 9/24/02 03:00:00	NO DATA FOUND			
09/24/02 03:00:00 - 9/24/02 03:15:00	NO DATA FOUND			
09/24/02 03:15:00 - 9/24/02 03:30:00	10.0.0.1	13	0.87	
09/24/02 03:30:00 - 9/24/02 03:45:00	NO DATA FOUND			

FIG. 11(c)

US 8,645,517 B2

1 **POLICY-BASED CONTENT DELIVERY NETWORK SELECTION**

RELATED APPLICATIONS

This application is a continuation of U.S. application Ser. No. 11/976,648, filed Oct. 26, 2007, titled, "Policy-Based Content Delivery Network Selection," (which will issue on Dec. 28, 2010 as U.S. Pat. No. 7,860,964) and which is a continuation-in-part (CIP) of U.S. application Ser. No. 10/259,497, titled "Configurable adaptive global traffic control and management," filed Sep. 30, 2002, now U.S. Pat. No. 7,822,871, the entire contents of which are hereby incorporated herein by reference for all purposes. Application Ser. No. 10/259,497 is related to and claims priority from provisional U.S. Patent Application No. 60/325,177, titled "Configurable Adaptive Global Traffic Control and Management," filed Sep. 28, 2001, the entire contents of which are incorporated herein by reference for all purposes. This application is also related to and claims priority from U.S. application Ser. No. 11/932,162, filed Oct. 31, 2007, the entire contents of which are incorporated herein by reference for all purposes.

RESERVATION OF COPYRIGHT

This patent document contains information subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent, as it appears in the U.S. Patent and Trademark Office files or records but otherwise reserves all copyright rights whatsoever.

FIELD OF THE INVENTION

Aspects of the present invention relate to network traffic management. Other aspects of the present invention relate to configurable, adaptive, global traffic control and management in networks such as the Internet.

BACKGROUND & SUMMARY

As the volume of Internet traffic grows, providers of web content and applications increasingly need to deliver content from multiple servers at widely-separated locations in order to sustain a good end-user experience under high traffic loads. This need generates several difficult challenges, including, among others:

- how to guarantee the fault-tolerance of such a multiple-server system in the face of failure of one or more individual servers;
- how to control the way in which requests from end-users are distributed to each server according to important content provider policy constraints imposed for economic, contractual or other reasons; and
- how to guarantee high performance as experienced by end-users as network conditions change.

This invention solves these and other problems by providing a means to distribute network (e.g., Internet) traffic according to a configurable set of rules. The rules can be configured to take into account key factors such as:

- server availability;
- specific requirements of content providers who deploy the invention, e.g., distribution based upon geography, position in IP address space, load share, etc.
- state of the network (Internet) at any given moment, including measures of network latency.

2

These rules together provide an extremely fine-grained level of network Internet traffic control to providers of Internet content and applications, enabling them to dramatically improve the end-user experience (measured by speed of request resolution, associated download time, and the availability of servers) over that provided by conventional web servers and mirrored server farms.

There are many potential uses for the invention. One use is to provide a stand-alone service directing traffic exclusively to a set of designated servers managed by a single organization. The invention may also be used in more general ways—for example, one or more of the designated destinations can refer to servers (or server collections) outside the organization's control. The latter case includes, for example, Content Delivery Networks (CDNs), as well as local load-balancing servers, as potential destinations. The invention can also be used, e.g., to provide the DNS (Domain Name Service) component of a Content Delivery Network itself. It can be deployed as a service on behalf of subscribers, or it can be deployed as software to be used directly by subscribers themselves.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is further described in terms of exemplary embodiments, which will be described in detail with reference to the drawings. These embodiments are non-limiting exemplary embodiments, in which like reference numerals represent similar parts throughout the several views of the drawings, and wherein:

FIG. 1 depicts a configurable adaptive global traffic control and management framework;

FIG. 2 describes exemplary types of policies;

FIG. 3 shows an exemplary hierarchy or a decision tree built based on subscriber policies;

FIG. 4 shows an exemplary relationship between a subscriber's network hierarchy and subscriber policies;

FIG. 5 depicts a high-level architecture of an adaptive traffic control framework;

FIG. 6 depicts a high level functional block diagram of an administrative master agent (AMA);

FIG. 7 depicts the internal functional block diagram of a monitoring mechanism;

FIG. 8(a) is an exemplary flowchart of a process, in which a content delivery framework provides adaptive policy-based domain name service;

FIG. 8(b) is an exemplary flowchart of a process, in which a domain name server resolves a hostname based on policies;

FIG. 8(c) is an exemplary flowchart of a process, in which a monitoring mechanism of a domain name server dynamically monitors the name service operations as well as the availability and the load share status of servers;

FIG. 8(d) is an exemplary flowchart of a process, in which an ATC administrative network maintains dynamic policies and monitors the operations of a content delivery framework;

FIG. 8(e) is an exemplary flowchart of a process, in which an ATC network monitoring mechanism traps events from different domain name servers and generates alerts when necessary;

FIG. 9(a) shows an exemplary secure web based graphical interface, through which a subscriber may define load share and shed fraction policies among static resources;

FIG. 9(b) shows an exemplary interface for defining policies for dynamic servers;

FIG. 9(c) shows an exemplary graphical user interface through which an overflow server may be defined using a canonical name;

US 8,645,517 B2

3

FIGS. 10(a)-10(o) show exemplary subscriber policies; and

FIGS. 11(a)-11(c) show exemplary archived log information that can be displayed and viewed through a graphical user interface.

DETAILED DESCRIPTION

The invention is described below, with reference to detailed illustrative embodiments. It will be apparent that the invention can be embodied in a wide variety of forms, some of which may be quite different from those of the disclosed embodiments. Consequently, the specific structural and functional details disclosed herein are merely representative and do not limit the scope of the invention.

Although there are a number of different scenarios in which the invention might be deployed, this description will focus, for clarity and example only, on a scenario in which DNS service is provided by a third party on behalf of a content or applications provider.

FIG. 1 depicts a high-level of a framework or system 100 according to embodiments of the present invention. A subscriber 102 provides content via a subscriber server network 104 which is made up of one or more server groups 106-1, . . . , 106-k (collectively 106).

As used in this description, the framework according to embodiments of this invention contemplates three groups of users:

An end-user who wishes to access content and/or applications over the Internet.

An operator who offers and supports the service on a managed network of computers.

A subscriber (typically a content or applications provider) who subscribes to the operator's service to help to support the needs of the end user.

The terminology end-user, operator and subscriber are used throughout this description to distinguish these three roles, although there are many scenarios in which more than one role can be taken by a single entity. Such scenarios are contemplated by this invention.

Each server group 106 may correspond to a server hierarchy which includes one or more tiers of servers. For example, the first tier of a server hierarchy may comprise one or more primary servers and the second tier of the server hierarchy may comprise possibly one or more overflow servers that are used when the primary servers in the first tier fail to function properly. In general, for the j-th server group, the primary servers at the first tier of server group 106-j are denoted 108-j, and the overflow servers at the second tier of server group 106-j are denoted 110-j.

Each server in a server group is any process or collection of processes that provide resources in response to requests, e.g., from a client. A server can be any off-the-shelf Web server. In some embodiments, servers are typically a Web server such as the Apache server or Netscape Communications Corporation's Enterprise™ server.

Client 112 accesses the subscriber server network 104 in order to obtain content from the subscriber. Content includes any kind of data, including, without limitation, video and audio data and the like. To achieve this access, a user at client 112 enters a resource locator, e.g., a Universal Resource Locator ("URL"), into a browser 114 on client 112. URLs specify the location of resources (information, data files, etc.) on the network. URLs are defined in detail in T. Berners-Lee et al, Uniform Resource Locators (URL), Network Working Group, Request for Comments: 1738, Category: Standards Track, December 1994, located at "http://ds.internic.net/rfc/

4

rfc1738.txt", which is hereby incorporated herein by reference. URLs generally have the following form:

scheme://host[:port]/url-path

where "scheme" can be a symbol such as "file" (for a file on the local system), "ftp" (for a file on an anonymous FTP file server), "http" (for a file on a file on a Web server), and "telnet" (for a connection to a Telnet-based service). Other schemes can also be used and new schemes are added every now and then. The port number is optional, the system substituting a default port number (depending on the scheme) if none is provided. The "host" field (the hostname) maps to one or more particular network addresses for particular computer(s). The "url-path" is relative to the computer specified in the "host" field. A url-path is typically, but not necessarily, the pathname of a file in a web server directory. Those skilled in the art will realize and understand, upon reading this description, that the client may not actually enter a URL, but needs only ran an application that needs to look up a domain name (i.e., perform a DNS query). A browser does this when a client enters a URL, but other applications may perform DNS queries in other manners.

The framework or system 100 includes at least one Domain Name Service (DNS) name server 118-1. In preferred embodiments, the system 100 also includes DNS name servers 118-1, 118-2, . . . , 118-n, (collectively referred to as name servers 118) all operated by a single particular entity. In the embodiment shown in FIG. 1, one of the name servers 118-1 serves the U.S.A., another name server 118-2 serves the U.K., another name server 118-3 serves Japan, and so on. The various name servers 118-1, 118-2, . . . , 118-n can be configured (collectively or individually) to serve geographical regions, geopolitical regions, corporate structures or any other logical organizational structure. The structure and operation of each name server 118 is described below.

When the client's browser 114 obtains a request (e.g., in the form of a URL), the browser queries its resolver 116 for an address for the hostname specified in the requested URL. The resolver 116 eventually queries a particular name server (e.g., name server 118-1). The name server 118-1 returns (provides or attempts to provide) the IP (Internet Protocol) address (or addresses) of a server (or servers) in the subscriber server network or a CNAME (a domain name). Other DNS answer types (e.g., MX record, NS record, etc.) are also possible, as described below. The determination of the particular IP address returned to the resolver 116 may be based on a number of factors, including the resolver's location (e.g., as determined from the resolver's IP address) and various policies (e.g., subscriber policies 120, other policies 122) in a policy database 124. The client's browser 114 is then able to communicate with the selected server in the subscriber server network in order to obtain the desired resource.

Name servers 118-1, 118-2, . . . , 118-n, according to embodiments of the present invention, include an adaptive traffic control (ATC) mechanism 126 which provides domain name service to a client based on policies in the policy database 124. Each name server 118 includes or has access to a location determination mechanism 128 for associating the client's request with the client's location. Any program that can determine a requestor's location may be used.

The subscriber server network 104 represents a network of servers that provides, on behalf of an underlying subscriber, Internet content or services. For example, a subscriber may be a content provider, which has its own network of servers that deliver content to end users' browsers via the Internet. The subscriber server network 104 may be configured in such a way that the processing of the service requests may be reasonably distributed among all the servers in the server net-

US 8,645,517 B2

5

work **104**, according to some criteria. To achieve that distribution, the subscriber server network **104** may be organized, for example, to have one or more server groups (e.g., server group **1106-1**, . . . , server group **k106-k**), each of which may be responsible for processing a portion of the service requests. For example, service requests coming from Finland may be routed to a server group that is physically located in Europe.

Each server group **106** in the subscriber server network **104** may comprise a plurality of servers to further share the load. For example, a service request from Finland may be subsequently routed to a server located in Scandinavia. Routing service requests to different server groups and subsequently to different servers may be based on various criteria. For example, such routing may be based on the distance between the origin of the request and the location of the server. For example, for a service request originated from Finland, it may be more effective and efficient to direct the request to a server located in Norway instead of directing it to a server located in the U.S. Server load may also be used to determine where to route a service request. For example, a service request originated from the U.S. may be routed to a server group in Mexico if the server group in the U.S. is overloaded while the server group in Mexico is relatively idle.

Servers in a server group may be organized into a hierarchy with one or more tiers of servers. Servers at different tiers may have different designated purposes. For instance, servers at the first tier of a server hierarchy may include servers that are primary functioning servers, servers at the second tier may include servers that are used as overflow servers which become active only when the primary servers at the first tier become unavailable or overloaded, and servers at the third tier that are used as second layer overflow servers that become active only when the primary servers and the overflow (or first layer overflow) servers at the second tier become unavailable or overloaded, etc.

The first tier of a server group hierarchy may include one or more primary servers. When multiple primary servers are present, they may be configured in such a way that they share the load. For example, if there are five primary servers at the first tier, they may be configured so that each takes 20% of the total service requests routed to the server group. For each of such primary servers, it may be further configured so that when a particular server is overloaded or fails, the load share originally designated to this server may be shed or directed to other server(s). The load shed may also be configured when the server is not overloaded. The portion to be shed to other server(s) may be governed by certain pre-defined policies.

As discussed above, servers in a server group **106** may be classified into different categories based on specific functionality of the underlying servers. For example, a server in a server group **106** may be a primary server, a first layer overflow server, or a second layer overflow server, etc. A server may also be classified in terms of whether and how a server is to be dynamically monitored. For instance, in some embodiments, a server can be classified as either a monitored server or a managed server.

A classification of a monitored server indicates that the underlying server is to be monitored dynamically for its availability. In this case, the server may be probed for its availability according to some pre-determined schedule. Such a probe may be sent to the underlying server from different locations of the network so that the availability can be detected accurately. That is, if the underlying server failed to respond to one probe sent from one location, it does not necessarily indicate that the server is no longer available (could be due to that only part of the network is congested).

6

A probe can be realized in different fashion. It can be simply a signal sent to the server to request an acknowledgment. It can also be a poll operation in which a file stored at a designated location on the underlying server is polled. If the file can be successfully polled, the underlying server is considered to be available. The detected availability may be used to adjust or update policies associated with the server so that the network traffic management will respond to the dynamics related to the server.

A server that is classified as a managed server may be actively participating the adaptive policy-based management scheme. Similar to a monitored server, a managed server may be regularly probed for its availability. In addition, a managed server may dynamically provide information related to its load share or load shed. A managed server may update its load share or load shed fraction according to its changed capacity or its current load. For example, when a managed server is upgraded, its capacity may be increased so that it may increase its load share or decrease its load shed fraction. On the other hand, when a managed server is overloaded, it may revise its load share to a lower level or increase its load shed fraction to prevent failure. A managed server may revise its load share or load shed by updating the corresponding load share or load shed information in a designated file stored on the server and this file may be polled by a name server so that the dynamically changed load share and load shed information can be used to direct traffic accordingly.

The distribution of service requests (from a client **112**) within the subscriber server network **104** may be controlled through a set of ATC policies (**120**, **122**) stored in the policy database **124**. Various authorities may influence the ATC policies and may create policies in the policy database **124**. For example, the subscriber may set up policies to direct traffic with respect to considerations such as the geographical locations and the capacities of the underlying servers in the subscriber's network **104**. Other policy entities **138**, including, for example, the operator of the name servers **118** or various geo-political entities may also have policies regarding how the network traffic should be managed and directed. For example, governmental or some organizational agencies may regulate some aspects of network traffic policies. Such regulation policies may be required to be incorporated so that a service request from the client **112** can be routed in a manner that satisfies regulatory policies.

Thus, the ATC policies in the policy database **124** may be a combination of subscriber policies **120** and other policies **122** from different sources such as subscriber **102** and policy entity **138**. Policies from different sources may be accessed by the ATC mechanism **126** from the policy database **124**. From the point of view of the ATC mechanism **126**, the source of a policy may not be relevant or even determinable. For example, policies may be defined in a textual file stored at a designated location, which may be downloaded to an ATC policy management mechanism **152** in an ATC administrative framework **142** and then broadcast to the database manager **132** located in each of the name servers in the ATC name server network **140**. The download may be via either a graphical user interface (GUI), a file transfer protocol (FTP), or some other mechanism. Policy makers may also enter policies directly via a web-based GUI. For example, the subscriber **102** may enter subscriber policies **120** via a browser interface **156** connected with the ATC administrative framework via, preferably a secure interface (e.g., implemented using the "https" protocol).

The policies from the policy database **124** used by the ATC mechanism **126** are collectively referred to as ATC policies, which may include, not is not limited to, the subscriber poli-

US 8,645,517 B2

7

cies 120 as well as other policies 122. The ATC policies may be organized in a manner that is appropriate to govern and/or control the traffic at different levels of the subscriber server network 104. FIG. 2 shows exemplary types of ATC policies 200 according to embodiments of the present invention.

The ATC policies (200) may be classified into different types such as geo-political policies 215, load share policies 240, failover policies 245, tiered failover policies 250, shedding policies 255, regulatory policies 260, and Classless Inter-Domain Routing (CIDR) block policies 210.

These exemplary policies in the policy database 124 are described in greater detail below:

Geographic Policy (215): Decisions are based on location of the end-user or an approximate thereof, e.g., using the IP address of an end-user's resolver. For example, if the IP address is within the U.S., the request may be directed to a "domestic" group of servers in the subscriber server network 104, otherwise it may be directed to "international" servers in the network. The request may be directed to a proprietary Content Delivery Network (CDN) or to another service specified by the subscriber.

Load Share Policy (240): The subscriber can explicitly specify the amount of traffic (load share) to be directed to each of their servers within a defined server set in the subscriber server network 104. Typically the load may be specified according to the capacity of each server. Based on such specification, the load share of each server may be derived as a percentage of the total load.

Failover policy (245): The subscriber may specify policies regarding a failover situation where some of the primary servers fail to function. To take care of such a scenario, a failover policy may instruct the ATC mechanism in terms of, for example, how often to monitor the availability of the servers and what strategy to adopt when a partial set of the primary servers are detected to be unavailable. For instance, a failover policy may specify to distribute the load of a failing server to other primary servers. It may also alternatively instruct the ATC framework to direct traffic to servers other than the primary servers.

Tiered Failover Policy (250): The subscriber may specify a strategy through tiered failover policies by which the load should be re-directed to servers at a next tier when one or more subscriber servers in a previous tier fail to function. For example, in defining the subscriber server network, each server group may be configured as a hierarchy, having the first tier of primary servers, the second tier of first layer overflow servers, and the third tier of second layer overflow servers, etc. In this case, a tiered failover policy may be defined to indicate when the load should be re-directed from the primary servers to the overflow servers. For instance, a tiered failover policy may indicate that when all the primary servers fail, the load should be directed to the second tier, and when servers at both the first tier and the second tier fail, the load should be directed to the overflow servers at the third tier. The traffic may also be re-directed to some other servers. For example, when a content delivery network (CDN) is available, the traffic may be re-directed to the entire CDN.

Shedding Policy (255): In some circumstances, a fraction of the "load" originally designated to a server may be shed or re-directed to one or more different servers. This may occur when the amount of traffic directed to the subscriber server exceeds a prescribed level. In these cases, a subscriber-specified fraction of traffic (shed fraction) that would otherwise be directed to the server may be shed to one or more other different servers. Such strategy may be adopted to prevent catastrophic failure due to overload. The servers that take the shed load may be an overflow server or some other servers

8

such as a content delivery network. An overload situation may be detected according to the response time of the server. For example, if the response time from a server becomes long, it may indicate that the server is overloaded. In this case, shedding policies 255 may be invoked to re-direct the traffic elsewhere. For instance, if there are a total of 3 primary servers in a server group with load share of (0.3, 0.3, 0.4) and the primary server that is designated to take 40% of the total load is completely overloaded, a shedding policy may specify to shed the load of this server by re-directing 50% of its original load to, for example, the servers located in the CDN 105. A shedding policy may also specify a condition upon which the traffic will be directed again to the shedding server. Such a condition may relate to a desirable level of performance of the shedding server.

CIDR Policy (210): Policy decisions are supported based upon CIDR blocks of IP address space. CIDR denotes Classless Inter-Domain Routing, an IP addressing scheme that replaces the system based on classes A, B, and C. With CIDR, a single IP address can be used to designate many unique IP addresses.

Regulatory policy (260): Certain policies may be specified by some policy entities to control network traffic. Such policies may be enforced in system 100.

A policy may be static or dynamic. Selection of a server from the subscriber server network 104 may be based on an adaptive, regularly updated map of the state of the Internet as well as adaptively updated policies. The map may cluster IP addresses together according to their network latency to a selected set of network agents. This enables the subscriber servers to be selected according to their "network proximity" to an end user's browser, optimizing resulted download time. The policies may be defined in such an adaptive manner that they reflect the dynamic status of the servers such as the availability and load.

Geo-political policies 215 may govern the selection of a server according to where the client is located. As shown in FIG. 2, the geo-political policies 215 may be further classified into continental based distribution policies 220, country based distribution policies 225, region based distribution policies 230 and time zone based distribution policies 235. These policies may be set up to govern the process of selecting a particular server group according to different geographical or political criteria.

Geo-political policies differ from load based policies (described above). The former is designed to guide selections based on geographical criteria or time criteria. The latter concerns the selection process with respect to the dynamic capacity and load of the underlying servers. The load share policies 240 govern the selection according to the capacities of the servers. Partial failover or tiered failover policies (245 and 250) govern the traffic re-direction process when functioning servers in a server group are overloaded or failed.

The ATC policies 200 may be hierarchically constructed to form a decision tree. FIG. 3 depicts an exemplary subscriber policy hierarchy or a decision tree built based on various types of subscriber policies, according to an embodiment of the present invention. In FIG. 3, the continental based distribution policies 220 are used to first direct a selection process limited to a particular continent. For example, if the requesting client 112 is located in Asia, the continental based distribution policies 220 may direct the further selection within servers that are located in Asia. Similarly, if the client 112 is located in either North America or Europe, the further selection process may be directed to the appropriate servers located in the corresponding continent.

US 8,645,517 B2

9

As noted, a policy may be created by generating a decision tree representing the ATC rules that the user (content provider/subscriber) wants to apply. At the leaves of this tree (referred to as resource nodes) a user specifies the answers (IP addresses or CNAMEs) that the user wants ATC to provide in response to a DNS request. At the branches of this tree (referred to as branch nodes) the user specifies the various decision criteria that the user wants to apply. Those skilled in the art will realize and understand that, in this discussion about policy setting and control of the ATC, the term “user” (as opposed to “end user”) generally refers to the content provider/subscriber whose content is being delivered.

Branches within a policy decision tree enable ATC decisions based, e.g., upon the IP address of the requesting resolver. Branches can be selected based upon different criteria. In a presently preferred exemplary embodiment, branches can be selected based on five different criteria:

- Split on World zone
- Split on country
- Split on U.S. state
- Split on U.S. time zone
- Split on block of IP addresses—specified, e.g., using CIDR classless notation: e.g. 1.2.0.0/16, 1.2.3.4/32 (single IP), or 0.0.0.0/0 (any IP).

Those skilled in the art will understand, upon reading this description, that different and/or other criteria may be used to make branch selections. Examples of such criteria include, without limitation, time of day, day of week, etc.

Resources are selected to occupy the leaves of a policy decision tree. Resources are used to specify the ATC answers to DNS queries. ATC offers substantial decision making power during the resource selection process. It is here, for example, that traffic can be distributed according to load share and shed fraction criteria. In a presently preferred exemplary implementation there are five types of resource nodes that can be selected in the policy decision tree:

- IP addresses (Internet Protocol addresses)
- CNAME records (Canonical Name records)
- MX records (Mail Exchange records)
- NS records (Name Server records)
- Load sharing server sets (LSSS)—consisting of Managed, Dynamic, Static and Overflow servers.

Those skilled in the art will understand, upon reading this description, that different and/or other resource nodes may be used.

IP address(es) are used when the current decision tree leaf points to one or more fixed IP addresses. CNAME records are used when the current decision tree leaf points to a destination whose CNAME is known.

MX Records are used when the current decision tree leaf points to one or more MX (Mail Exchange) records. MX records may be combined with IP address or Load Sharing Server Set resources. To select this option, the user first enters the Exchange, Pref, and TTL (Time To Live) details for the mail exchange. The Exchange field holds the domain name of the host which will receive mail. If the Exchange name lies under the Domain Name Origin, then the policy must specify an IP address for the name by means of another node label and appropriate resource (IP address or Load Sharing Server Set). A Pref field holds the preference order of the host; lower numbers are preferred (used first) over large numbers.

NS Records are used when the current decision tree leaf points to one or more NS (Name Server) records. NS records are used to specify a delegation from the ATC name servers to another set of name servers.

10

Load Sharing Server Set (LSSS) option is used when the current decision tree leaf points to a number of destinations among which the user wants to share load (including failover) according to arbitrary criteria. In presently preferred exemplary implementations, this decision takes place only after any decisions that the user wants to make via branch nodes based on the IP address of the requesting resolver. There are presently four different types of load sharing server resources, namely static servers, managed servers, dynamic servers, and overflow servers.

In a present implementation, managed Servers are preferred over the other server types.

Load sharing server sets (LSSS) are necessary to utilize the load share, tiered/overflow and failover rules. When an ATC name server encounters a server set, one or more answers are chosen based on several criteria, including, e.g., the load share of each server, the tier in which each server sits, the shed fraction of each server, and the monitoring results of each server.

Static, dynamic and managed servers have load share, shed fraction and, when in a set, tier parameters. An overflow server is, by definition, at the lowest tier and cannot shed; thus it has only the load share parameter. Dynamic and managed servers each may have the load share and shed fraction values updated continuously and may be deemed on or off-line due to monitoring.

When answering a DNS query from a LSSS, ATC preferably selects answers (IP addresses or a CNAME) from servers included in the set. ATC examines servers in tier order (tier 1 first, tier 2 next, and so on, with overflow servers last), selecting servers based on load share, shed fraction and monitoring results. Online servers are selected within a tier according to their load share values relative to other servers in the same tier. If a candidate server is shedding, e.g., 10% (i.e., has a shed fraction of 0.1), ATC will keep that server in the DNS reply nine tenths of the time, and look in the next tier for a server one tenth of the time. If there is only one tier, the shed traffic returns to the same tier and another server is selected. If the LSSS specifies that multiple answers should be returned, ATC will look for each answer starting from the first tier. If any server selection falls to the overflow servers, one of these is selected as the sole answer to the DNS reply. Typically, the overflow server is a CNAME, and DNS answers of this type should not be combined with any other record type.

In a presently preferred implementation, the number of answers chosen from Load Sharing Server Sets depends on two more user-settable parameters, the “number of answers desired” (NAD) and the “number of answers required” (NAR). These parameters control how many answers are picked from non-overflow servers.

The user sets the NAD parameter to the typical number of answers to be selected by ATC and returned in DNS queries. The NAR parameter, which must be less than or equal to the NAD, comes into play when the normal ATC decision making process cannot select the desired number of servers specified in NAD. This server count shortfall situation can arise when one or more servers are shedding or offline. Setting NAD to one (1) or more guarantees that ATC will never reply to a DNS query with no servers. When NAD is greater than one, ATC may deviate from the desired load share ratios, e.g., if there are three servers and NAD is three then DNS replies will typically have all three servers, irrespective of their load share values.

If the number of servers selected is less than NAD, but greater than or equal to NAR, then server selection is finished. However, if the number of servers selected is less than NAD and less than NAR, then ATC progressively ignores shed

US 8,645,517 B2

11

fraction and load share values until it is able to select NAR servers. As a special case, if ATC cannot select any servers because they are all offline, all servers are returned in the DNS reply (subject to size constraints of the DNS packet).

As an example, consider a LSSS with two servers in tier 1, two servers in tier 2, and two answers desired and required (i.e., NAD=2, NAR=2). If both servers in tier 1 are online and not shedding, those servers will be returned. If one of the tier 1 servers has 10% shed, then 10% of the answers contain one server from tier 1 and one server from tier 2. If both servers from tier 1 are offline, the two servers from tier 2 are returned. If all four servers are offline, then all four are returned (the special case).

In some implementations, the LSSS may also have a Style parameter which may be used to control where traffic goes when it is directed away from a shedding server, or from an offline server, according to the following table.

Style	Shed Traffic	Off-line Traffic
Constant Tier Load Ratio (default)	Next Tier	Same Tier
Constant Server Load Ratio	Next Tier	Next Tier
Maximal Tier Load	Same Tier	Next Tier

Consider, e.g., a LSSS which contains a few servers in Tier 1 and an overflow server sending traffic to a CDN. The three styles above relate to three underlying motivations for how to split traffic between the Tier 1 servers and the CDN.

If the Tier 1 servers should always take a specified percentage of traffic, as a whole, then the Constant Tier Load Ratio style is best. For example, for the CDN to always take 50% of the traffic, each of the Tier 1 servers would have their shed fraction set to 0.5. This way, even if servers are added or removed from Tier 1, that tier and the CDN will always get 50% of the traffic. This, of course, means that the load on the Tier 1 servers will vary as servers are added or removed.

If, instead, it is not acceptable for the Tier 1 servers to increase their load upon the failure of another Tier 1 server, the Constant Server Load Ratio style is best. With this style, both shed traffic and traffic that would have gone to an off-line Tier 1 server are directed to the next tier (in the example above, the CDN). This is a preferable scenario if the Tier 1 servers are generally under high load. Lastly, if the Tier 1 servers are generally under low load and the CDN is to be considered a last resort, the Maximal Tier Load style should be used. With this style, both shed traffic and traffic that would have gone to an off-line Tier 1 server are instead directed to other servers in Tier 1. Traffic only gets to the CDN if all servers in Tier 1 are offline or shedding.

The Static Server option is used to share load at the current decision tree leaf among a set of servers that the user does not expect to change often over time. The load share for a given server represents the amount of traffic that will be sent to that server within the same tier of the server set. The fraction of traffic sent to that server is computed by summing the load share numbers for all servers in the same tier of the server set and dividing the load share by that total. The shed fraction for a given server represents the fraction (from 0 to 1) of traffic that should be redirected away from a given server, after the initial decision based upon load share. The online flag can be used to remove a server from the list of ATC answers if the user wishes to take a server off-line for maintenance. (If a server has been switched off-line, end users may continue to

12

access it for the duration of the TTL, or perhaps longer if the end user's resolver or browser ignores the TTL.)

In addition to default values for load share and shed fraction, the managed server may optionally be monitored to determine the on/off-line status, and polled for load feedback information (updated values for the load share and shed fraction). Furthermore, a managed server may hold either an IP address or a CNAME. A managed server holding an IP address without monitoring or load feedback becomes equivalent in functionality to the static server. Managed servers with monitoring only (load feedback not configured) are used when the user expects the load share and shed fraction of the servers to change infrequently, but want an instant reaction if the user's server becomes unavailable. In a presently preferred implementation, ATC monitors the user's server by polling a user-specified resource (denoted by a user-specified URL) at regular intervals (e.g., every 30 seconds) from each ATC name server. The URL scheme may be, e.g., HTTP, HTTPS or TCP. With HTTP(S), ATC name servers independently send a HEAD request, optionally with HTTP's basic authentication, and examine the response code. Response codes from 200 to 399, received within 15 seconds of the probe, put the server in an online state as viewed by the probing name server; all other conditions (connection refused, server 400 or 500 error, 15 second time, etc.) lead to an offline state for the server until the next probe.

Alternatively, ATC can monitor the user's server by checking the ability to connect to a specified port. In this case, the URL format is tcp://HOST[:port], and the ATC name servers simply attempt to establish a TCP connection to determine the online state of the user's server. The same probing rate and timeout durations apply.

Suppose, for example, that a user wishes to use ATC for name service for download.example.com. Suppose too that the user has two servers and wants ATC to monitor them both and only return the IP address of an available server. In this example, the user could define two Managed servers, each with a Monitored URL as shown in the following table:

Server nick name	IP Address	TTL	Monitored URL
SF server	10.0.0.1	2 m	http://10.0.0.1
NY server	10.0.0.2	2 m	http://10.0.0.2

The Monitored URL can point to any resource available to the web server, in this case the web server's root. When these servers are used in a server set, ATC continuously polls the monitoring URLs to determine the on/off-line state of the user's servers.

A user may specify the load share and/or shed fraction of the user's servers dynamically, using a Load Feedback URL to specify an XML resource file (at a location of the subscriber's choice) in the following form (preferably plain text):

```
<ATCServerStatus
  loadShare="LS"
  shedFraction="SF"
  onLine="OL"
  expireTime="ET"/>
```

The file must be made accessible by the subscriber via http or https, optionally with basic authentication. In a presently preferred implementation, the onLine field is required, and all others are optional. LS is a number greater than (or equal to) zero, SF is a number from 0 to 1, OL is either True or False, and ET is a time (either absolute GMT or relative).

US 8,645,517 B2

13

The expireTime parameter is the absolute time at which the current XML file expires, or the relative amount of time, post-probe, at which the current XML file expires. After a retrieved XML file expires, the loadShare and shedFraction return to their default values. In this manner a user (i.e., content provider) can temporarily (or permanently) override default load share and/or shed values.

The Managed Server specification with both Monitoring and Load Feedback may then look like the following table:

Server Name	IP	TTL	Monitoring URL	Load Feedback URL
SF Server	10.0.0.1	2 m	http://10.0.0.1	http://10.0.0.1/atc-feedback.xml
NY Server	10.0.0.2	2 m	http://10.0.0.2	http://10.0.0.2/atc-feedback.xml

The load feedback resource does not need to be on the server itself. The user may wish to use an independent server to control the load share and shed fraction of some or all of the user's servers. The state of the user's managed servers is maintained by each ATC name server as follows:

If the user's server uses neither a monitored URL nor a load feedback URL, it is considered to be always on-line with default load share and shed fraction.

If the user's server uses a monitored URL but not a load feedback URL, it is considered to be on-line when the monitored URL is successfully retrieved (HTTP response code from 200 to 399 within 15 seconds), otherwise it is considered to be off-line.

If the user's server uses both a monitored URL and a load feedback URL, the server status is preferably given by the following table:

Load Feedback URL Contents	Monitoring Down	Monitoring Up
No XML	Off-line	On-line with defaults
No XML, last known still valid	Off-line	Last known
On-line	Off-line	On-line with defaults
On-line, LS and SF	Off-line	On-line with LS and SF
Off-line	Off-line	Off-line
Off-line, LS and SF	Off-line	Off-line

In the above table, "No XML" means that either the XML file could not be retrieved within the 15 second time-out period (e.g., due to network error), the XML file was retrieved but could not be parsed, or the XML file had expired. "Last known" refers to the full contents of the last successfully retrieved XML file. Last known values expire and cease to exist at the expire time if the expire time is specified in the XML file. The term "defaults" refers to the default load share and shed fraction values specified in the Managed resource server.

If the user's server uses the Load Feedback URL but no Monitored URL, the user can achieve an N minute delayed failover by putting in the XML snippet a relative expireTime of N minutes (+Nm) and a loadshare of one, while having in the policy a default Load Share of 0.

Dynamic Server

The Dynamic Server option may be used to share load at the current decision tree leaf among a set of servers when the user expect that the load share, shed fraction, or server availability may change reasonably often. (The managed server

14

option is preferred over the dynamic server option.) Load share and shed fraction are defined for this case as they are for Static Servers. However, the Dynamic Servers selection offers flexibility in exactly how they are specified. The Dynamic Server option supports specification of load share, shed fraction and the On line flag dynamically, allowing them to be defined by the subscriber via a short XML status file at a location of the subscriber's choice (as described above).

The subscriber specifies the location of this file to ATC, including any authentication requirements desired. ATC polls the file at predetermined fixed intervals, e.g., every 30 seconds, and updates the current ATC policy to reflect its contents. Subscribers can easily specify changes to load share, shed fraction and the On line flag by editing the file, either manually or by updating it using automated modules, without requiring updates to an entire ATC policy.

In order to maintain flexibility, values for the "URL Down load share" and the "URL Down shed fraction" parameters can be set in the ATC policy. They are used to control the behavior of ATC under the circumstance that the ATC polling mechanism is unable to retrieve the status file for a server, or is unable to parse its contents.

Case 1

Subscriber specifies all three ATCServerStatus attributes in the status file, with the "URL Down load share" parameter set to zero. In this case the subscriber's server will not be returned as an answer under any conditions until the polling mechanism retrieves a valid XML file.

Case 2

Subscriber specifies all three ATCServerStatus attributes in the status file, with the "URL Down load share" parameter set to one. In this case, ATC treats a failure to return valid XML according to the "Down load share" and "Down shed fraction" specified in the policy.

Case 3

Subscriber specifies only the on line ATCServerStatus attribute in the status file. In this case, ATC uses the "Default load share" and "Default shed fraction" specified in the policy if the "On line" flag in the ATCServerStatus file is set to true.

Below is a table detailing all of the possible cases. In the table, a dash represents ATC taking the dynamic server out of consideration. LS and SF should be replaced by actual numbers.

Dynamic Server Cases

Form of retrieved ATCServerStatus file	LoadShare	ShedFraction
<ATCServerStatus onLine="True" loadShare="LS" shedFraction="SF"/>	LS	SF
<ATCServerStatus onLine="True"/>	Default load share	Default shed fraction
<ATCServerStatus onLine="False" loadShare="LS" shedFraction="SF"/>	—	—
<ATCServerStatus onLine="False"/>	—	—
Bad format or un-retrievable:	Down load share	Down shed fraction

Overflow Server

A user uses the Overflow Server option if the user would like to specify what should happen if, for some reason (e.g., server unavailability), no answers are selected by the Static or Managed (or Dynamic) Server selection process. Typically,

US 8,645,517 B2

15

the overflow server will be a CName to a service, such as Level 3 Communications' Content Delivery Network (CDN), that should be available in the case that the user's individual Static or Managed (or Dynamic) servers are unavailable. Managed and Dynamic Server Status Alerts

The global ATC network preferably maintains an overall state of a user's Managed and Dynamic servers using Monitored or Load feedback URLs. A Server Status page, available from the subscriber web interface, continuously updates with the status (up or down) of the user's URLs and, if down, the relevant error condition (time-out, connection failure, file not found, etc.). Additionally, the overall server state is displayed (either on-line or off-line) along with a relevant status message. If a server changes state, an email message is sent to the user detailing the event. Email messages are preferably not sent more frequently than once in 15 minutes per domain name origin.

Once a DNS request is directed to an appropriate continent, the country based distribution policies 225 may further constrain the selection to particular servers that are in the same or close-by countries where the client 112 is located. For example, if the client 112 is located in the U.S., the country based policies 225 may direct the selection from the servers located in the U.S. Similarly, the region based distribution policies 230 may further constrain the selection to, for example, the west coast or east coast depending on where the client 112 is located.

When a particular server group is selected (e.g., after a hierarchical decisions based on the continent-based policies 220 and the region-based policies 230), the load share policies 240 govern the process whereby servers in a given server group should be selected. Such policies may be determined based on the servers' capacities or may be adaptively revised based on the dynamic performance or load of the underlying servers. The load share policies 240 may specify the percentage (share) of the total requests that each server in a server group should handle. For example, if a server group comprises a total of three primary servers (server 1, server 2, server 3), a load share policy for this server group may specify the load share as (0.3, 0.5, 0.2), indicating that server 1 should take 30% of the total load, server 2 should take 50% of the load, and server 3 should take 20% of the total load.

The tiered failover policies 250 govern the selection of a server when a particular default set of servers is no longer functioning or available. For example, primary servers in a server group may be considered as a default set of servers that provide service when operation is normal. Unavailability of such primary servers may be detected according to the response time of the server. For example, if a server is simply not responding, the server may be considered as not available. When all the primary servers are down, the tiered failover policies 250 govern where the traffic should be directed. For instance, the tiered failover policies 250 may specify to direct all traffic to the overflow servers at the next tier.

FIG. 4 depicts an exemplary relationship between the subscriber server network 104 and the subscriber policies 120, according to embodiments of the present invention. The subscriber server network 104 is depicted in FIG. 4 as a network hierarchy or a tree in which the subscriber server network 104 includes server group 1 106-1 through server group k 106-k, and each server group may include its own primary server group 108 in the first tier and overflow server group 110 in the second tier (e.g., server group 1 106-1 has primary server group 108-1 and overflow server group 110-1 associated therewith, while server group k 106-k has primary server group 108-k and overflow server group 110-k associated therewith).

16

The subscriber policies 120 govern selection of one or more servers within the subscriber server network 104 (or in some other network of servers) so that their IP addresses may be returned in response to DNS requests from the client 112 (resolver 116). To determine or to select appropriate servers in the subscriber server network 104, the decisions may be hierarchical. For example, as shown in FIG. 4, at the level of the subscriber server network 104, the ATC mechanism 126 may make geo-political distribution decisions 410 to determine which server group should be selected. Once a particular server group is selected, the ATC mechanism 126 may further determine which particular servers in the group should be selected according to, for example, how the traffic load should be shared among the servers within the server group. Decisions at server group level may be based on the load share policies 240. This decision making process illustrates that, at each branch node in the server hierarchy, appropriate ATC policies may be applied to govern the selection of an appropriate server or servers.

As illustrated in FIG. 4, the geo-political policies 215 may be applied to control the selection of a particular server group, the load share policies 240, the tiered failover policies 245 and 250, and the shedding policies 255 may be applied to re-direct traffic. That is, the subscriber server network 104 forms a tree and the ATC policies 200 govern the flow from the top of the tree to one or more leaves of the tree. The ATC mechanism 126 controls the flow by applying the ATC policies 200.

FIG. 5 depicts a high level architecture of an adaptive traffic control (ATC) framework according to embodiments of the present invention. The ATC framework comprises an ATC administrative framework 142 and an ATC name server network 140. The ATC administrative framework 142 is responsible for various administrative tasks associated with subscribers or other policy making entities (138), include manipulating and storing the ATC policies 200, propagating or broadcasting the ATC policies 200 to name servers in the ATC name server network 140, monitoring name server behavior, generating status reports to display monitoring results on appropriate media, and sending alert to a network operation center (NOC) 148 (see FIG. 1) and the subscriber 102.

The ATC name server network 140 is responsible for responding to DNS requests, including processing DNS requests, applying the ATC policies 200 to select one or more servers from the subscriber server network (or a different designated network), and replying with the IP address(es) of the selected server(s). The ATC name server network 140 dynamically maintains the ATC policies 200, that are either received from the ATC administrative framework 142 or updated according to the dynamic operational status of the servers. The ATC name server network 140 provides domain name-IP address resolutions based on dynamically updated ATC policies 200. The ATC name server network 140 may also monitor the operational status of individual name servers within the network and supply logging and monitoring data to the ATC administrative framework 142.

The ATC administrative framework 142 may be designed to have fault-tolerance. For example, as depicted in FIGS. 1 and 5, the ATC administrative framework 142 may include an administrative master agent (AMA) 144, and one or more AMA backups 146-1, . . . , 146-m (collectively 146). All AMA agents, including the master agent 144 and the backups 146 may be capable of performing the exact functions except that one of them (e.g., master AMA 144) may be a central or primary administrative master agent and others (e.g., AMA backups 146) may be backup or secondary or redundant mas-

US 8,645,517 B2

17

ter agents. The central or primary AMA **144** may be responsible for regularly backing up the AMA backup agents **146**. When, for whatever reason, the primary AMA **144** is no longer functioning properly, one of the AMA backups **146** may take on the role of the central or master AMA **144**.

The ATC name server network **140** may comprise a plurality of name server agents **118-1**, **118-2**, . . . , **118-k**, each of which may be designed to be responsible for the DNS requests of a particular geographical (or any other administrative or functional) region. For example, the name server agent **118-1** may be responsible for processing all the DNS requests from North America, the name server **118-2** may be responsible for Europe's DNS requests, and the name server **118-3** may be responsible for DNS requests from Japan. In addition, a name server agent may also serve as a backup name server agent for other name server agents in the network. For example, if the North America name server agent is not functioning properly, the Europe name server agent may be temporarily assigned to handle the DNS requests from North America. For that purpose, all ATC policies may be propagated to all of the name server agents in the ATC name server network **140**.

FIG. 6 depicts a high level functional block diagram of an administrative master agent (AMA) **144** according to embodiments of the present invention. The AMA **144** comprises a secure web-based graphical user interface **160**, a policy editing mechanism **162**, an ATC policy database **124**, an administrative policy update mechanism **164**, an ATC administrative browser **166**, an ATC policy management mechanism **152**, a report generation mechanism **168**, an ATC network monitoring mechanism **150**, and an administrative master backup mechanism **170**.

The ATC policy database **124** stores the ATC policies **200**. As discussed above, the ATC policies **200** may include policies from different sources (e.g., from subscribers and from other entities that control network traffic). Both the subscriber policies **120** and other policies **122** may be defined and provided to the AMA **144**. In the exemplary configuration shown in FIG. 6, the policies may be received at the AMA **144** through the secure web based GUI **160**. There may be other means through which the policies can be delivered to the AMA **144** and will be discussed later. The policy editing mechanism **162** organizes the received policies to form the ATC policies **200** and then stores them in the ATC policy database **124**.

The ATC policy management mechanism **152** may broadcast or propagate the ATC policies **200** to relevant name servers in the ATC name server network **140** so that the ATC policies **200** may be used to control the domain name translation service. The stored ATC policies may be dynamically updated via different means. For example, the policies may be updated through the ATC administrative browser **166**, or the administrative policy update mechanism **164** may revise existing ATC policies.

Alternatively, the AMA **144** may also be provided with policies from different sources through the ATC policy management mechanism **152**, which may regularly poll dynamically updated policies from different locations. Such locations may include designated network locations that are designated to provide dynamic policy related information or servers (either in the subscriber server network **104** or in the CDN **105**) that are classified as managed servers. For instance, a server may dynamically specify its load share via a designated file stored on the server or at some other location (the file may reside on the server being monitored or at some other location, e.g., a central load controlling server). To retrieve such dynamically defined load share information

18

from a managed server, the ATC policy management mechanism **152** may poll the designated file stored on the managed server to obtain relevant load share information. Dynamic policies may also be polled from other policy making entities.

Broadcasting ATC policies may take place periodically according to some pre-defined interval or may be triggered whenever the stored ATC policies are updated. The ATC policy management mechanism **152** may monitor changes made to the existing ATC policies. The ATC policy management mechanism **152** may poll the ATC policies stored in the ATC policy database and see whether there are changes. On the other hand, whenever the ATC policy management mechanism **152** polls dynamic policies from specified locations (such locations may be specified in existing ATC policies), it may determine whether the dynamically polled policies differ from existing ATC policies. In the event that updated policies are different from the existing ATC policies, the ATC policy management mechanism **152** may re-broadcast the updated ATC policies to the ATC name server network **140**.

The ATC network monitoring mechanism **150** may collect DNS log summaries from different name servers in the ATC name server network **140**. Such summary log data may be received in the form of events that provide information such as, for example, the number of requests directed to particular servers in a given time period. The ATC network monitoring mechanism **150** may collectively processes such DNS log summaries (or events) from the entire ATC system. The report generation mechanism **168** may generate monitoring status reports from these summaries and makes such reports available to the subscriber **102** via the secure web-based GUI **160**.

The administrative master backup mechanism **170** may periodically update the AMA backups **146-1**, . . . , **146-m** to ensure that all the backup agents are current. This may include replicating the ATC policies, the operational status of various control mechanisms (including the ATC policy management mechanism **152**), the policy editing mechanism **162**, and the administrative policy update mechanism **164**, and providing the up-to-date information to the AMA backups.

A major function of an administrative master agent is to manage the ATC policies, to make sure that updated ATC policies are supplied to the name server agents in the ATC name server network **140**, to monitor the various name servers' performance, to generate dynamic monitoring status report of system performance, and to maintain a connection through which policies may be updated dynamically and monitoring report can be examined.

Information flagging system errors and other anomalous conditions is collected by the ATC network monitoring mechanism or agent **150**. FIG. 7 depicts an internal functional block diagram of an ATC network monitoring mechanism **150**, according to embodiments of the present invention. The ATC network monitoring mechanism **150** includes a trap handler **176**, a processing mechanism **182**, and an alert generation mechanism **184**. The trap handler **176** traps events from the name servers in the name server network **140**. The processing mechanism **182** analyzes both the collected trapped events, and, based on analyzed information, the alert generation mechanism **184** generates alerts when necessary, and reports such alerts to, for example, the ATC's network operation center (NOC) **148** and the subscriber **102** (FIG. 1).

The trap handler **176** further comprises an event receiver **178** that intercepts trap events from the name servers and an event consolidation mechanism **180** which may classify the trapped events and organize them in a reasonable and appropriate fashion. The processing mechanism **182** may process the consolidated events to identify useful or informative pat-

US 8,645,517 B2

19

terms which may be further used, by the alert generation mechanism **184** to identify problematic patterns which may significantly affect the system performance.

The Adaptive Traffic Control (ATC) framework according to the present invention may be deployed as a stand-alone service directing traffic solely to the subscriber's servers, in conjunction with another content delivery network (CDN) provider, or in conjunction with any other service.

Each domain name server in the ATC name server network **140** may include (FIG. 1) a location determiner **128**, an adaptive traffic control (ATC) mechanism **126**, a monitoring agent or mechanism **130**, a database manager **132**, and a report generator **134**. When a name server (e.g., **118-1**) receives a request from the resolver **116** of the client **112**, the location determiner **128** determines the location of the resolver and sends such location information to the ATC mechanism **126**. Based on the location information, the ATC mechanism **126** retrieves relevant ATC policies from the policy database **124** (e.g., the subscriber policies **120** or the other policies **122**) and selects one or more servers in the subscriber server network **104** according to the retrieved relevant policies. The corresponding IP address(es) or the CNAME of the selected servers are then returned to the resolver **116**.

The database manager **132** maintains the policy database **124**. It received policies broadcast from the ATC policy management mechanism **152** in the ATC administrative framework **142** and populates the policies in the policy database **124**. The ATC policies received from the ATC administrative framework **142** may also include information that defines or classifies servers in the subscriber server network **104** (or in the CDN **105**). For example, some servers may be defined as monitored servers and some may correspond to managed servers.

When the database manager receives such information, it may inform the monitoring mechanism **130** of the classification of the underlying servers so that the monitoring mechanism **130** can monitor each server according to its status. When the ATC policy management mechanism **152** broadcasts updated policies, the database manager **132** accordingly updates the relevant policies stored in the policy database **124**.

The monitoring mechanism **130** monitors the operational status of the name server **118-1** and one or more servers in the subscriber server network. It may collect events occurred in the name server **118-1** during operations and send such events to the ATC network monitoring mechanism **150** in the ATC administrative framework **142**. On the other hand, it may also monitor the operations of various servers in the subscriber server group **104** (or in the CDN **105**) according to how each server is defined (monitored or managed server).

If a server is defined as a monitored server, the monitoring mechanism **130** may dynamically probe the server (as discussed earlier) to determine its availability. If a server is defined as a managed server, the monitoring mechanism **130** may monitor its availability during operation. The monitoring mechanism **130** may also poll dynamic load share information from the server. When the monitoring mechanism **130** detects that a server is no longer available, it may inform the database manager **132** to create a local policy that indicate that the server is no longer available so that the ATC mechanism **126** can take into account when resolving a hostname.

When the monitoring mechanism **130** polls the dynamic load share information from the server, it may inform the database manager **132** to update the load share policies in the policy database **124** that are affected by the dynamics of the polled load share. For example, if three primary servers in a server group originally have load share (0.3, 0.3, 0.4) and the

20

third primary server now changes its load share to 0.2, the database manager **132** may accordingly update the load share among these three primary servers into (0.4, 0.4, 0.2).

The report generator **134** generates reports related to the operations of the underlying name server based on log information **136** recorded. Such generated reports may be sent to a report consolidator **154** in the ATC administrative framework **142** so that reports from different name servers may be consolidated.

A subscriber may activate (turn up) the ATC system (DNS servers) in one of two ways: using a DNS CNAME or using NS delegation. Instead of using ATC to direct traffic for a single DNS hostname, the subscriber may have many different subdomains that it would like to direct to ATC. For example, the subscriber might want all downloads from dl.customer.com handled by ATC, together with all of its subdomains, but have all other domains that it controls, such as www.customer.com, resolved by its own name server:

```
dl.customer.com=>ATC
any.thing.dl.customer.com=>ATC
www.customer.com=>not ATC
customer.com=>not ATC
```

In this case, instead of adding the CNAME record in the subscriber's DNS zone file, it simply delegates the dl.customer.com. name to ATC via NS records in its zone file.

The delegation (using NS-records) method is presently preferred as it is more flexible and offers all of the reliability, scalability and flexibility of ATC. After the initial contact for delegation, a properly operating resolver making occasional queries will not contact the subscriber's own name server or name servers again. The result provides much better name resolution performance since the unnecessary overhead of routing each fresh DNS request through the subscriber's name server is eliminated. The CNAME method keeps the subscriber's own name server in the loop. That is, whenever the CNAME TTL expires, client resolvers will return to the subscriber's name servers to refresh the record.

For both CNAME and NS delegation methods, the procedure to turn off the ATC switch is the same—the subscriber edits its DNS zone files to remove the delegation authority to ATC. DNS requests will continue to be served in accordance with the subscriber's defined ATC policies until the TTLs have expired on the appropriate delegations.

The System in Operation

FIG. 8(a) is an exemplary flowchart of a process, in which a name server resolves a DNS request based on ATC policies, according to an embodiment of the present invention. First, a user enters a URL into the user's browser **114** (or into any application that accepts URLs as input and obtains the corresponding resource for the client) (at **802**). The client's resolver **116** attempts to resolve the hostname of the URL in order to obtain an IP address of a server from which the resource identified by the URL can be obtained (at **804**). The resolver **116** will be directed by the client's DNS (not shown) to a DNS Name Server **118** in the ATC name server network **134** (at **806**). The resolver **116** provides the name server **118** with the hostname it is attempting to resolve.

The name server **118** receives the request to resolve the hostname (at **808**) and determines one or more servers in the subscriber server network **104** or in the CDN **105** that can process the client's request according to the location of the resolver **116** as well as relevant ATC policies retrieved from the policy database **124** (at **810**). Details of this operation are described with reference to FIG. 8(b). The IP address(es) of the selected server(s) are returned to the requesting resolver **116** (at **812**). The browser **114** then connects to one of the servers (at **814**) in order to obtain the requested resource.

US 8,645,517 B2

21

FIG. 8(b) is a flowchart of an exemplary process, in which a domain name server selects one or more servers according to location of the client and relevant ATC policies. The location of the resolver 116 (or client) is first determined (at 816). Relevant ATC policies are then retrieved (at 818) from the policy database 124. One or more servers in either the subscriber server network 104 or the CDN 105 are selected according to the determined location of the client and the relevant ATC policies (at 820).

FIG. 8(c) is a flowchart of an exemplary process, in which the monitoring mechanism 130 in a domain name server monitors the operations of the name server as well as one or more servers in the subscriber server network 104 or the CDN 105. Events occurring during domain name service are monitored (at 822). Such events are sent to the ATC network monitoring mechanism 152 (at 824). In addition, the availability of the one or more servers are also monitored (at 826). Furthermore, if any of the one or more servers is defined as a managed server (determined at 828), dynamic load share information is polled (at 830). Both the availability information and the dynamic load share information (if any) are used to update some ATC policies that are local to the name server (at 832).

FIG. 8(d) is a flowchart of an exemplary process, in which the ATC policy management mechanism 152 dynamically maintains and broadcasts the ATC policies. Initially, policies from different sources are received (at 834) and broadcast to the name servers 118 (at 836). If backup is necessary (determined at 838), the ATC policy management mechanism 152 sends current policies to the master backup agents 146 (at 840).

The ATC policy management mechanism 152 also performs dynamic policy maintenance. It polls dynamic policy information (at 842) at certain defined intervals and uses such polled dynamic policy information to update existing policy (at 844). The updated policies are then broadcast to the name servers (at 846). If the updated policies need to be propagated to the backup agents (determined at 848), they are sent to the master backup agents (at 850).

FIG. 8(e) is a flowchart of an exemplary process, in which the ATC network monitoring mechanism 150 monitors operations of the name servers and sends alert to the NOC 148 and the subscriber 102. Events sent from monitoring mechanisms of different name servers are trapped (at 852). Such trapped events from different sources are then consolidated (at 854) and processed (at 856). If there is any alarming situation (determined at 858), the ATC network monitoring mechanism 150 generates an alert (at 860). The generated alert is then sent to both the NOC 148 and the subscriber 102 (at 862). Policy Administration

The ATC policies may be initially set up and later adaptively updated according to servers' dynamic operational status. The ATC policies may be formed via various means. The described approaches may also be applied to form other policies. A GUI approach or a file-based approach (or both) may be employed to set up subscriber policies. Through the GUI approach, different means to form subscriber policies may be adopted. For example, a browser may be used so that a subscriber can enter policies directly. An XML file containing descriptions of the subscriber policies may also be loaded using GUI approach so that the policies may be parsed and stored. As another alternative, a file containing descriptions of the subscriber policies may also be loaded in a similar fashion and parsed accordingly. When a file based method is used, a file containing descriptions of the subscriber policies, either constructed based on XML or some other structure, may be accessed via other means such as FTP.

22

In some preferred embodiments, a subscriber accesses a secure web-based GUI 160 (FIG. 6) using subscriber's secure browser interface 156.

FIGS. 9(a)-9(c) show exemplary secure web based graphical interfaces, through which a subscriber may define load share policies and overflow policies with respect to specified network resources, according to embodiments of the present invention. There may be different types of load sharing servers: (1) static servers, (2) dynamic servers, and (3) overflow servers. A dynamic server is one that is specified as either a monitored or a managed server whose dynamic availability may be monitored and whose load may be re-directed when it becomes unavailable. In the case of a monitored server, both of its load share and load shed fraction may also be made dynamic.

Correspondingly, the policies governing routing requests to different types of servers may also be defined accordingly. First, the policies that govern static servers are applied when servers are relatively static and do not change often over time. This may mean that the availability of the servers is fairly stable and load sharing among different servers is also relatively stable. The policies that control dynamic servers are applied when servers are expected to change frequently. Such policies include failover policies, shedding policies, and tiered failover policies (described earlier). The overflow policies control the change of flow of the requests when, for some reason, primary servers, either static or dynamic, become unavailable or overloaded. In this case, relevant overflow policies determine to which overflow server a request for name service should be directed.

The load share with respect to a given server represents the amount of traffic that will be sent to that server within a static server setting. An exemplary method to compute the load share of the traffic sent to the server is to sum the load share amount of all servers in a current static server setting and divide the load share for that server by the summed total to derive the load share.

The shed fraction for a given server represents the percentage of traffic that should be redirected away from the server. The redirection may be carried out after load share is made based upon load share policies. FIG. 9(a) gives an example interface, in which each of the servers listed is assigned various selection related parameters. For example, for each server, a TTL is specified. In addition, a load share and a shed fraction may also be specified. For example, both servers New York and London have load share of 1.0. The shed fraction assigned to server New York is 0.2, meaning that 20% of its load is re-directed to some other server, and the shed fraction assigned to server London is 0.3, meaning that 30% of its load is re-directed to some other server.

Shedding fractions may also be computed automatically on the fly. For example, when a server is detected to have slow response, indicating that it may be overloaded, a shedding fraction may be computed according to the discrepancy between its expected response time and the actual response time. Such an automatically computed shedding fraction can then be applied to re-direct the newly computed fraction of the traffic to a different server to unload some of the traffic originally intended to be directed to the overloaded server.

For each resource server, an on-line flag may also be set to indicate whether the server is currently available. This flag is shown in FIG. 9(a) in the rightmost column. With this flag, a server may be temporarily removed from the service, if, for instance, a server needs to be taken down for maintenance.

Since managed and dynamic servers are expected to change relatively frequently, policies that govern their selection may be defined in two stages. The policies for managed

US 8,645,517 B2

23

and dynamic servers may be initially specified in a similar fashion as for static servers. For instance, load share and shed fraction can be defined for dynamic servers New York and London, as shown in FIG. 9(a). During operation, however, the policies that govern the selection of managed and dynamic servers may be established dynamically. For example, initially defined policies (e.g., load share, shed fraction, and on-line flag) may be adaptively revised based on, for instance, an on-line status report retrieved from one or more specified locations on the network. Dynamic policies provide considerable flexibility in how the selection may be conducted in a manner that is adaptive to the network health or any other network performance related factors (e.g., maintenance).

The on-line status report may be provided at one or more network locations specified by the subscriber. The locations may be specified as a Uniform Resource Locator (URL) and may be accessed through an HTTP request. A location of the status report may be specified in the initial policy, providing a constant link to the location. FIG. 9(b) shows an exemplary GUI for specifying an URL link to a status report. An URL link for the location of the status report is specified as "http://server" which corresponds to a server named London with an IP address of 10.0.0.3. The access may be authenticated and such authentication requirements may also be specified in the initial policies. The on-line accessed status report may be constructed based on information gathered during monitoring the servers' performance. For instance, each server in the subscriber server network may provide a status report containing information related to its performance. Name servers in the ATC name server network may poll such information from such servers and such information may be used to update policies.

The frequency with which the status report is accessed may also be specified explicitly. For example, it may require the relevant mechanisms (specifically, the ATC policy management mechanism or the monitoring mechanism of each name server) to poll the status report at a regular time, e.g., every thirty seconds.

A status report may be constructed using some standard language such as extendible Markup Language (XML). Such a status report may contain revised policies, which may be determined by the subscriber manually based on network performance, devised by an automated process based on network performance, or may be generated by an individual managed server. In such cases, the status report includes updated policies and when it is polled, the updated policies are used in future traffic control. For example, a managed server may re-define its load share of according to its dynamic capacity. When such dynamically defined load share information is accessed and used in enforcement, the traffic is controlled in a manner that is adaptive to the network dynamics.

When a status report is accessed, the relevant mechanisms (e.g., the ATC policy management mechanism in the ATC administrative framework or the ATC mechanism in corresponding name server) updates the associated policies based on the information contained in the status report. For example, a status report may indicate that the current load share for server New York should be 0.4 (down from 1.0 previously) with the same shed fraction (0.2) and the current load share for server London should be 0.5 (down from previous 1.0) with 0.4 shed fraction (40% shed fraction). An alternative scenario may be that the status report provides network performance data about each dynamic server and the ATC mechanism, after accessing the status report, determines how the current policies may be revised accordingly.

24

When the status report provides dynamic policies, it may, in general include, for each dynamic server, the following information:

```
<ServerStatus loadShare="LS" shedFraction="SF"
online="Boolean"/>
```

where LS and SF represent numerical numbers and "Boolean" represents a logical value of either "true" or "false".

With respect to dynamic servers, with the mechanism of dynamic policies described above, a subscriber or a server can easily specify changes to existing policies (e.g., changes to load share, shed fraction policies and the On-line flag) without having to update an entire ATC policy hierarchy. In addition, a parameter (called "Down on line" in some embodiments) can be set in an ATC policy to control the behavior of the ATC mechanism under the circumstance that the ATC policy management mechanism is unable to properly access the status report due to reasons such as a failed polling or retrieval or failing to parse the status report's content. According to some embodiments of the present invention, three different exemplary approaches may be applied to handle the situation.

With the first approach, a subscriber may instruct an ATC mechanism in a name server not to select a particular server if the status report for that server can not be properly obtained. The subscriber may specify this policy by setting parameter "Down on line" to false. In this case, the ATC mechanism will not consider the underlying server as a candidate for selection process until the polling mechanism retrieves a valid status report.

The second approach to deal with a polling failure is to allow the ATC mechanism to select the underlying dynamic server according to its default policies (or original policies). To specify this solution, the subscriber may set the "Down on line" parameter true.

The third approach to handle the situation where the status report can not be properly retrieved is to allow the ATC mechanism to select the underlying server if its "on line" flag is on (it is available). To achieve this, a subscriber may specify only the "on line" ServerStatus in the status report, with parameter "Down on line" set to either true or false. In this case, the ATC mechanism will use a load share and a shed fraction specified in the policy when the "On line" flag is set to true. The load share and the load shed fraction may both be dynamically determined (either broadcasted from the ATC policy management mechanism or polled by the ATC mechanism 130 from the underlying server).

Static, dynamic and managed servers are primary servers, although their selection may be controlled by operationally different policies. The third type of servers are called overflow servers. They provide alternatives when primary servers, for some reason, can not be selected. Typically, an overflow server corresponds to a Canonical name or CName, pointing to a service providing a CDN (such as, e.g., the CDN service offered by Level 3 Communications). The choice of overflow servers may be determined based on the belief that they are in general always available. An overflow server may be defined through a window illustrated in FIG. 9(c). In FIG. 9(c), overflow servers are defined by a CName, which points to the CDN server address "customer.footprint.net" (e.g., the CDN 105, shown in FIG. 1).

EXAMPLES

Example A

FIGS. 10(a)-10(b) show example subscriber policies, defined based on a set of resource servers, that govern the

US 8,645,517 B2

25

selection of the servers according to different criteria such as geographical location of an incoming request and the time zone of each of the locations. In FIG. 10(a), three different resource servers are defined, including an “eastserver” with IP address 10.0.0.2, a “westserver” with IP address 10.0.0.1, and a service network with CName “customer.footprint.net”. Here, the service network may include more than one server.

FIG. 10(b) shows an exemplary decision tree embedded in a set of geographical policies that guide how the traffic should be directed to a set of servers. In the decision tree depicted in FIG. 10(b), the selection is first directed to different resources at the top level according to a split between the United States and rest of the world. A resource may correspond to either a single server (or CDN) or a group of servers (or CDNs). A request initiated from non-U.S. geographical regions is directed to a resource named “London”. Within the United States, time zone based policies are further applied. A request initiated from a region within the central time zone is directed to a resource named “New York”. A request initiated from a region within Alaska time zone is directed to the resource “London”. Finally, a request initiated from a region within the United States that does not fall in any of these time zones (i.e., Alaska or Central) can be directed to either “London” or “New York”.

Example B

FIGS. 10(c)-10(d) show another example decision tree constructed based on a set of resources (servers or CDNs) and subscriber policies that are defined based on geographical locations of an incoming request with overflow policies that allow the ATC mechanism to direct traffic to pre-defined overflow servers when primary servers are not available. In the Resource window shown in FIG. 10(c), three resources are defined: static resources “London” and “New York”, each defined based on its IP address, a service network named “Sydney” with CName “customer.sandpiper.net”. Each resource may correspond to a set of servers such as a server group with certain hierarchy. Resource “London” is further defined as a dynamic resource and resource “Sydney” is further defined as an overflow server network nicknamed “cdn_service”. FIG. 10(d) illustrates an example decision tree built based on these resources.

The decision tree in FIG. 10(d) first splits two ways at the top level based on whether a request is from the United States. When a request is initiated from the United States, it is directed, according to the policies illustrated in FIG. 10(d), to the dynamic servers at the first tier of resource “London”. A request from the United States may also be directed to one of the overflow servers in resource “cdn_service”. This may happen when either all the primary servers at the first tier of resource “London” fail to function or when such primary servers are overloaded.

The next four examples (Examples 3-6) are based on an example subscriber which has ATC handling traffic for www.subscriber.com with a CNAME hand-off to www.subscriber.com.nsac.net. The example subscriber has three data-centers, one in San Francisco, New York and London. The set of resources used by these four examples is shown in FIG. 10(e). Managed servers are used with monitoring since the example provides failover from one server to another. The default load share values vary: the San Francisco server can handle 2.5 times the traffic of the London or New York servers. The overflow server is utilized in only two of the example policies. It is a CNAME specification to a CDN—the Level 3 CDN.

26

Example C

With this policy (FIG. 10(f)), the example subscriber does not utilize any geographic ATC rules, and the London server is not in use. End users are simply directed to one of the two servers, provided the ATC monitoring of that server indicates the server is up and functioning. Unlike standard DNS round-robin techniques, ATC directs end users to the two servers in exactly the ratios specified by the load share values in the Resources Window (FIG. 10(e)). ATC will return the IP address of the San Francisco server 2.5 times as often as the New York server. If ATC monitoring detects that the New York server is offline, ATC name servers will immediately stop returning the IP address of the New York server and only return the San Francisco server IP address until the monitoring indicates the New York server has returned. If the example subscriber determines that any one of the servers has too much traffic, the load share of that server should be adjusted in one of two ways:

- the subscriber can login to the ATC Admin GUI and change the load share value manually, or
- the subscriber can configure ATC to poll a load feedback URL for the load share value, and change that value on the fly

Changing the shed fraction for any one server is not appropriate in this example because there is no defined lower tier or overflow server to receive shed traffic. Both servers are in the same tier.

A variation of this example is to put the New York server in Tier 2, i.e., have New York function as a backup. In this configuration, all traffic would be directed to the San Francisco server unless it was monitored as down or shedding in which case traffic would then go to New York. That is, if the San Francisco server had a shed fraction of, e.g., 10%, then ATC would return the New York server 10% of the time. When there is only one server in a Tier, there is no load share balancing for ATC to do, and the load share value becomes unimportant (unless zero).

This example demonstrates load share variation and server failover handling within a load sharing server set consisting only of managed servers.

Example D

This policy (FIG. 10(g)) adds an overflow server to the server set of the previous policy example. The example subscriber has two methods to inform ATC that any one server should shed some of its traffic to the overflow server, in this case the Level 3 CDN. The overflow server is intended to be an effective large supply of bandwidth, to be used when the subscriber’s servers are at capacity or when the lowest latency is required.

- The shed fractions can be modified in one of two ways:
 - the subscriber can login to the ATC Admin GUI and change the shed fraction value manually; or
 - the subscriber can configure ATC to poll a load feedback URL for the shed fraction value, and change that value on the fly

The subscriber may maintain an even load across all subscriber servers by setting the load share. This value typically does not need continuous adjustment. Then, if traffic levels increase and all servers are nearing their capacity, all servers begin shedding traffic to the Overflow server(s) (in this example, “dl.subscriber.com.c.footprint.net”, a CName for a CDN).

Example E

This example policy (FIG. 10(h)) utilizes geographic control to direct end users to the subscriber servers nearby, on a

US 8,645,517 B2

27

continental scale, improving end-user response time (latency). A variation in tiers is used to specify preferential and failover server selection. Within a load sharing server set, all servers in Tier 1 are considered first. Only if a server cannot be selected from Tier 1 (due to shedding or because it is off-line) is Tier 2 considered, and so on for Tier 3. Requests with end-user resolvers in Europe (and Africa and the Middle East) are directed to the London server, unless it is down or shedding, in which case traffic is sent to New York, and then San Francisco. Similarly, requests from Asia and Australia are directed to San Francisco, unless that server is down or shedding, in which case the lower tiers are used.

In this case, the subscriber does not have an overflow server defined. If the load on any server becomes too high, the shed fraction can be increased on that server and traffic will be sent to the lower tiers defined within each geographic region.

If both the San Francisco and New York servers are offline, all traffic is directed to the London server until the U.S. servers return. If all three servers are offline, ATC is unable to determine which servers are preferred and thus selects all three servers, with the hope that the end user may be able to connect to one of them.

Example F

This policy (FIG. 10(i)) demonstrates finer geographic control. End users with resolvers on the West coast are directed to San Francisco rather than New York. Likewise, end users with resolvers in the Eastern U.S. are directed to New York rather than San Francisco. The overflow server, in this case “dl.subscriber.com.c.footprint.net”, a CName for a CDN, is again in use. If all servers are offline or shedding (which is considered unlikely), traffic will be directed to the CDN.

Note: The Other label is used, as before, to catch all non-specified world zones, and again within the U.S. time zone split to catch all remaining regions (in this case the western time zones of the U.S. and the rest of the Americas).

The following three examples (Examples 7-9) are based on an example subscriber which has ATC handling traffic for an origin server demo.nsac.net and for the domain nsac.net. The subscriber has three data centers in New York, San Francisco, and London, UK, with managed servers at each center. As shown in FIG. 10(j), the load shares for the managed servers are (NY: 1, SF: 2, and London: 1).

Example G

Origin Plus CDN

FIG. 10(k) shows a policy for the two sub-domains “images.nsac.net” and “www.nsac.net”. For the “images” sub-domain, end users are directed to servers in New York or San Francisco, in the ratio 1:2. That is, the ATC mechanism will return the IP address of a San Francisco server twice as often as that of a NY server. If the ATC mechanism detects that, e.g., the San Francisco server is down, it will stop return the IP address for the SF server and will only return the IP address for the NY server (until monitoring indicates that the SF server has returned). The subscriber may adjust the load share of the servers as in the earlier examples (either by logging in to the ATC to change the load share values manually, or by configuring ATC to poll a particular location for a configuration file). In case of overflow, the requests go to the CDN (with the CName “demo.c.footprint.net”).

The “www” sub-domain is handled somewhat differently in that the overflow for “www” will go to either of two CDNs.

28

In this example, as shown in FIG. 10(l), “Level 3 CDN” is a CNAME for a CDN at “demo.c.footprint.net”, and “Other CDN” is a CNAME for another CDN at “demo.othercdn.net”. The ATC is configured to share the load between the two CDNs in the ratio 1.5 to 1. In other words, as configured in FIG. 10(l), ATC will return the CNAME of the Level 3 CDN 1.5 times as often as it returns the CNAME of the “Other CDN”.

As with the load share values for the other servers, the load share for the CDNs can also be adjusted by the subscriber on the fly or by logging in to the ATC.

Those skilled in the art will realize and understand, upon reading this description, that a subscriber may wish to adjust load share values for a CDN for performance and/or business reasons. E.g., a subscriber may have to pay a higher rate for one CDN than for another.

Those skilled in the art will realize and understand, upon reading this description, that the two CDNs may be owned and operated by different entities and may be in different domains.

Using this example, a subscriber can have its traffic handled initially by its origin servers (in NY or SF), with overflow being handled by a CDN. Overflow for the “images” sub-domain is handled by a single CDN, whereas overflow for the “www” sub-domain is handled by two CDNs.

Example H

FIGS. 10(m)-10(n) show a policy in which traffic to the sub-domain “www” is split equally between three CDNs. Each CDN has a load share of 1 and so an end-user will be returned a CNAME of one of the three CDNs, with each CDN having one third of the load. Again, the subscriber can modify the load share on the fly or by logging in to the ATC.

Example I

FIG. 10(o) shows an exemplary policy with different rules for the sub-domains “ftp”, “download”, “www” and “images”.

End-user requests for the “download” sub-domain (download.nsac.net) are handled by the subscriber’s origin servers in NY and SF (in the ratio 1:2 (per FIG. 10(l)). Overflow from the “download” sub-domain is handled by the CDN “demo.cdn.com” (a CNAME for the CDN). In other words, overflow from the “download” sub-domain will be directed via a CNAME demo.cdn.com to the CDN (i.e., ATC will return the CNAME demo.cdn.com for overflow cases to the sub-domain “download”).

The “ftp” sub-domain is split based on country (i.e., on the country in which the end-user’s request is believed to originate). End-user requests from the US are handled by the domain “us.demo.com” (a CNAME), and all other requests are handled by “international.demo.com” (also a CNAME). In this manner, ftp requests can be directed to different networks based on their country of origin.

The “images” sub-domain is split initially on three geographic zones:

- I. Europe
- II. Asia and Australia
- III U.S. and other

Within each zone, end-user requests are further directed based on server load. E.g., in zone I (for Europe), end-user requests are handled by the tier 1 server(s) in the UK. If these fail (or are overloaded and need to shed load), requests then go to the tier 2 server(s) in NY. And if these fail (or are overloaded and must shed), requests are directed to the tier 3

US 8,645,517 B2

29

server(s) in SF. Thus, in response to an end-user request for the sub-domain “images”, the request originating in Europe, the ATC will return the IP address of a server in the U.K. But if that server in the UK is overloaded, the ATC will return the IP address of a server in N.Y. And if that server is also overloaded, the ATC will return the IP address of a server in S.F.

The Asia/Australia zone operates similarly, except that the servers are put into different tiers, favoring the order SF, NY, UK. All other zones favor the order NY, SF, UK.

The “www” sub-domain is not split based on geographic location of the end-user. Traffic is handled by SF first, with NY as an overflow.

In addition to the various web-based interfaces described herein, there are also other means through which subscriber policies may be defined. As discussed earlier, for example, subscriber policies may also be constructed or specified in an XML file which can be downloaded by an ATC mechanism and used to control the traffic.

Policies defined via different means (e.g., through web based GUI or XML file) may be converted into some pre-defined format within the ATC mechanism. Such pre-defined format may be designed for efficiency in manage and handling the ATC policies. For example, the internal format for ATC policies may be designed so that, internally, the AMAs can conveniently store, access, and broadcast the ATC policies to the name server agents and the name server agents can efficiently apply the policies.

As mentioned earlier, in addition to management of the ATC policies, the AMA may also monitor the performance of name servers and generates viewable DNS log reports. The monitoring mechanism may gather performance information from either the DNS logs of the name servers or the events trapped from the name servers. Such gathered information may be used by the report generation mechanism to construct informative reports. The report generation mechanism may also make such reports available to the subscribers via the secure web-based GUI. FIGS. 11(a)-11(c) show exemplary types of report information that is accessible via the secure web-based GUI. FIG. 11(a) depicts a web interface that allows a subscriber to view DNS log data of a specified resource server. A subscriber can specify the name of the server under review (e.g., “download.subscriber.com”), the form in which the log data is to be organized (e.g., specify output style of “Separate”, meaning to display IP and CName queries and replies with answers as separate items in the output), the time period under review (e.g., from Sep. 1, 2002 to Sep. 24, 2002), and the time resolution used in displaying the log data (e.g., automatically select time resolution).

FIG. 11(b) shows a plot of log data related to a specified server. The log data is presented in a plot, generated based on the log entries of a particular server against a specified period of time (X-axis) with certain resolution. The exemplary plot reflects the volume of the traffic directed to the underlying server during a period time between 2:30 pm and 3:30 pm on Sep. 24, 2002. The traffic volume is described in terms of number of replies generated by the underlying server per minute (Y-axis). It can be seen in this example plot that there is a surge in traffic volume between 2:36 pm and 2:38 pm. FIG. 11(c) shows a graphical display for the same log data in FIG. 11(b) in a table form, in which the log data is listed in a chronicle order from top to the bottom of the table. For instance, each row in the left column lists a period of time. In the exemplary table, the log data within the one hour period (2:30 pm to 3:30 pm) is divided into a plurality of sub-periods, each of which is 15 minutes. For each sub-period, the third column provides the number of replies from the correspond-

30

ing sub-period. For instance, during the sub-period of 3:15 pm to 3:30 pm, there are 13 replies and there is no reply in other sub-periods (consistent with the plot illustrated in FIG. 11(b)).

In some implementations, end-user resolvers may use a preferred subset of the ATC nameservers. This may be implemented by providing an initial pool of ATC name servers which respond to new requests from new end-user resolvers. When an end-user resolver first requires name resolution from an ATC nameserver, that resolver is directed (e.g., by the DNS system) to one of the pool of ATC nameservers. When that ATC nameserver provides an answer to the end-user resolver, it also gives that resolver a list of preferred ATC nameservers for that resolver, i.e., one or more ATC nameserver that are better suited to handle name resolution for that end-user resolver. An ATC nameserver may be better suited to handle requests from an end-user resolver for a number of reasons, e.g., because it is electronically closer to the resolver, because it is geographically appropriate for that resolver, etc. Those skilled in the art will realize, upon reading this description, that different and/or other measures of suitability may be used to provide an end-user resolver with a list of preferred ATC nameservers. Once an end-user resolver has a list of preferred ATC nameservers, that resolver will make future requests of nameservers in its list (until the entries for those nameservers expire or are replaced).

The various mechanisms described herein, including, without limitation, the adaptive traffic control (ATC) mechanism, the location determination mechanism, policy editing mechanism, administrative policy update mechanism, ATC policy management mechanism, report generation mechanism, a monitoring mechanism, and an administrative master backup mechanism may be implemented in hardware, software or a combination thereof. When implemented in software, they may be implemented in any type of appropriate interpreted or compiled programming language. When implemented fully or partially in software, aspects of the invention can reside on any memory or storage medium, including but not limited to a ROM, a disk, an ASIC, a PROM and the like. While the invention has been described with reference to particular mechanisms (algorithms, processes and functions) and architectures, one skilled in the art would realize that other mechanisms and/or architectures could be used while still achieving the invention.

When the various mechanisms of the present invention are running on a particular machine (e.g., at a client or on a server), they may reside in the memory of the machine or on a storage device or in a combination. Further, while many of the operations have been shown as being performed in a particular order, one skilled in the art would realize that other orders, including some parallelization of operations, are possible and are considered to be within the scope of the invention.

The present invention has been described above in connection with a preferred embodiment thereof; however, this has been done for purposes of illustration only, and the invention is not so limited. Indeed, variations of the invention will be readily apparent to those skilled in the art. Such variations also fall within the scope of the invention. Thus, while the invention has been described with reference to the certain illustrated embodiments, the words that have been used herein are words of description, rather than words of limitation. Changes may be made, within the purview of the appended claims, without departing from the scope and spirit of the invention in its aspects. Although the invention has been described herein with reference to particular structures, acts, and materials, the invention is not to be limited to the

US 8,645,517 B2

31

particulars disclosed, but rather extends to all equivalent structures, acts, and, materials, such as are within the scope of the appended claims.

The processing described may be performed by a properly programmed general-purpose computer alone or in connection with a special purpose computer. Such processing may be performed by a single platform or by a distributed processing platform. In addition, such processing and functionality can be implemented in the form of special purpose hardware or in the form of software being run by a general-purpose computer. Any data handled in such processing or created as a result of such processing can be stored in any memory as is conventional in the art. By way of example, such data may be stored in a temporary memory, such as in the RAM of a given computer system or subsystem. In addition, or in the alternative, such data may be stored in longer-term storage devices, for example, magnetic disks, rewritable optical disks, and so on. For purposes of the disclosure herein, a computer-readable media may comprise any form of data storage mechanism, including such existing memory technologies as well as hardware or circuit representations of such structures and of such data.

What is claimed is:

1. A method, operable in a framework in which an adaptive traffic control name server network implements policy-based traffic direction, the name server network comprising at least one domain name server comprising hardware in combination with software and constructed and adapted to provide adaptive policy-based domain name service, the method comprising:

(A) providing a graphical user interface (GUI);

(B) using said GUI to obtain at least one policy for direction of network traffic, wherein the GUI supports the setting of said at least one policy using a decision tree representing rules; and

(C) providing said at least one policy to the name server network, wherein the decision tree comprises one or more resource nodes, and one or more branch nodes, wherein the one or more resource nodes specify one or more answers to be provided in response to a Domain Name Service (DNS) request, and wherein the one or more branch nodes specify one or more decision criteria to be applied, and wherein the GUI supports the specification of:

(i) one or more answers for the one or more resource nodes, and

(ii) one or more decision criteria for the one or more branch nodes, wherein the one or more criteria are selected from criteria related to: world zones, countries, states, time zones, and blocks of Internet Protocol (IP) addresses, and wherein the one or more resource nodes are selected from: IP addresses, canonical name (CNAME) records, mail exchange (MX) records, name server (NS) records, and load sharing server sets; and wherein the answers are selected from: IP addresses and CNAMEs.

2. The method of claim 1 wherein the at least one policy relates to direction of network traffic based on sub-domains of a domain of a subscriber.

3. The method of claim 2 wherein the at least one policy that relates to direction of network traffic based on sub-domains of a domain of the subscriber relates to one or more of the sub-domains: "ftp", "download", "www", and "images".

4. The method of claim 1 further comprising:

(D) generating reports based on information gathered from the name server network; and

(E) providing said reports.

32

5. The method of claim 4 wherein the information gathered from the name server network comprises: DNS log summaries from the at least one domain name server in the name server network.

6. The method of claim 5 wherein the information includes a number of requests directed to particular servers in a given time period.

7. The method of claim 4 wherein the reports are made available via a secure web-based GUI.

8. The method of claim 1 wherein the act (C) comprises: providing said at least one policy to the name server network as an XML file.

9. An administrative agent, operable in a framework in which an adaptive traffic control name server network implements policy-based traffic direction, the name server network comprising at least one domain name server comprising hardware in combination with software and constructed and adapted to provide adaptive policy-based domain name service, said administrative agent comprising hardware in combination with software and constructed and adapted to:

(a) provide a graphical user interface (GUI);

(b) use said GUI to obtain at least one policy for direction of network traffic, wherein the GUI supports the setting of said at least one policy using a decision tree representing rules; and

(c) provide said at least one policy to the name server network, wherein the decision tree comprises one or more resource nodes, and one or more branch nodes, wherein the one or more resource nodes specify one or more answers to be provided in response to a Domain Name Service (DNS) request, and wherein the one or more branch nodes specify one or more decision criteria to be applied, and wherein the GUI supports the specification of:

(i) one or more answers for the one or more resource nodes, and

(ii) one or more decision criteria for the one or more branch nodes,

wherein the one or more criteria are selected from criteria related to: world zones, countries, states, time zones, and blocks of Internet Protocol (IP) addresses, and

wherein the one or more resource nodes are selected from: IP addresses, canonical name (CNAME) records, mail exchange (MX) records, name server (NS) records, and load sharing server sets; and

wherein the answers are selected from: IP addresses and CNAMEs.

10. The administrative agent of claim 9 wherein the at least one policy relates to direction of network traffic based on sub-domains of a domain of a subscriber.

11. The administrative agent of claim 10 wherein the at least one policy that relates to direction of network traffic based on sub-domains of a domain of the subscriber relates to one or more of the sub-domains: "ftp", "download", "www", and "images".

12. The administrative agent of claim 9 wherein said administrative agent comprises hardware in combination with software and constructed and adapted to:

(d) generate reports based on information gathered from the name server network; and

(e) provide said reports.

13. The administrative agent of claim 12 wherein the information gathered from the name server network comprises: DNS log summaries from the at least one domain name server in the name server network.

US 8,645,517 B2

33

14. The administrative agent of claim 12 wherein the information includes a number of requests directed to particular servers in a given time period.

15. The administrative agent of claim 12 wherein the reports are made available via a secure web-based GUI. 5

16. The administrative agent of claim 9 wherein said at least one policy is provided to the name server network as an XML file.

17. A system comprising:

(A) an adaptive traffic control name server network comprising at least one domain name server comprising hardware in combination with software and constructed and adapted to provide adaptive policy-based domain name service, and 10

(B) an administrative agent comprising hardware in combination with software and constructed and adapted to: 15

(a) provide a graphical user interface (GUI);

(b) use said GUI to obtain at least one policy for direction of network traffic, wherein the GUI supports the setting of said at least one policy using a decision tree representing rules; and 20

(c) provide said at least one policy to the name server network,

wherein the decision tree comprises one or more resource nodes, and one or more branch nodes, 25

wherein the one or more resource nodes specify one or more answers to be provided in response to a Domain Name Service (DNS) request, and

wherein the one or more branch nodes specify one or more decision criteria to be applied, and

34

wherein the GUI supports the specification of:

(i) one or more answers for the one or more resource nodes, and

(ii) one or more decision criteria for the one or more branch nodes,

wherein the one or more criteria are selected from criteria related to: world zones, countries, states, time zones, and blocks of Internet Protocol (IP) addresses, and

wherein the one or more resource nodes are selected from: IP addresses, canonical name (CNAME) records, mail exchange (MX) records, name server (NS) records, and load sharing server sets; and

wherein the answers are selected from: IP addresses and CNAMEs.

18. The system of claim 17 wherein said administrative agent comprises hardware in combination with software and constructed and adapted to:

(d) generate reports based on information gathered from the name server network; and

(e) provide said reports.

19. The system of claim 18 wherein the information gathered from the name server network comprises: DNS log summaries from the at least one domain name server in the name server network.

20. The system of claim 18 wherein the information includes a number of requests directed to particular servers in a given time period.

* * * * *

EXHIBIT C



US009021112B2

(12) **United States Patent**
Slocombe et al.

(10) **Patent No.:** **US 9,021,112 B2**
(45) **Date of Patent:** **Apr. 28, 2015**

(54) **CONTENT REQUEST ROUTING AND LOAD
BALANCING FOR CONTENT DISTRIBUTION
NETWORKS**

(56) **References Cited**

U.S. PATENT DOCUMENTS

(75) Inventors: **Michael Slocombe**, Acton, MA (US);
Matthew Miller, Plainville, MA (US);
Casey Ajalat, North Billerica, MA (US);
Vincent A. Fuller, III, Palo Alto, CA
(US)

4,495,570 A 1/1985 Kitajima et al.
4,591,983 A 5/1986 Bennett et al.
4,594,704 A 6/1986 Ollivier
4,726,017 A 2/1988 Krum et al.
4,803,641 A 2/1989 Hardy et al.

(Continued)

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

FOREIGN PATENT DOCUMENTS

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 0 days.

CA 2202572 10/1998
EP 0800143 A 10/1997

(Continued)

OTHER PUBLICATIONS

(21) Appl. No.: **12/050,648**

Zegura et al., Application Layer Anycasting: A Server Selection
Architecture and Use in a Replicated Web service, Aug. 2000, IEEE,
vol. 8, No. 4, pp. 455-466.*

(22) Filed: **Mar. 18, 2008**

(Continued)

(65) **Prior Publication Data**

US 2008/0235400 A1 Sep. 25, 2008

Primary Examiner — Backhean Tiv

Related U.S. Application Data

(57) **ABSTRACT**

(62) Division of application No. 09/982,721, filed on Oct.
18, 2001.

A content distribution mechanism that distributes content of a
content provider at various sites across a network and selects
the site that is nearest a content requestor using an anycast
address that resides at each of the sites. The sites are config-
ured as nodes (or clusters) and each node includes a content
server and a DNS server. The DNS servers are so associated
with the content servers at their respective nodes as to resolve
the name of the content provider to the IP address of the
content servers at the nodes. The DNS servers each are
assigned the anycast address in addition to a unique address,
and the anycast address is advertised to the network (in par-
ticular, the network routing infrastructure) using Border
Gateway Protocol (BGP). Node selection occurs when the
network routing infrastructure selects a shortest path to the
anycast address during DNS name resolution.

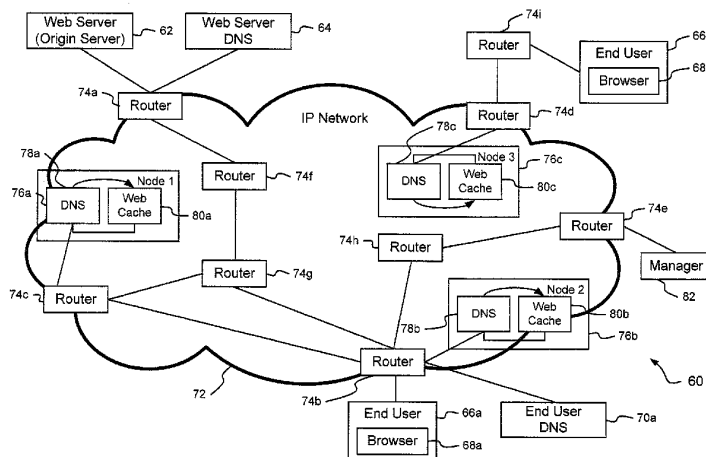
(51) **Int. Cl.**
H04L 29/08 (2006.01)
H04L 29/12 (2006.01)
H04L 29/06 (2006.01)

(52) **U.S. Cl.**
CPC **H04L 29/12066** (2013.01); **H04L 29/06**
(2013.01); **H04L 29/12207** (2013.01); **H04L**
61/1511 (2013.01); **H04L 61/20** (2013.01);
H04L 67/2895 (2013.01); **H04L 67/2842**
(2013.01); **H04L 67/288** (2013.01)

(58) **Field of Classification Search**
None

See application file for complete search history.

23 Claims, 5 Drawing Sheets



US 9,021,112 B2

Page 2

(56)

References Cited

U.S. PATENT DOCUMENTS

4,839,798 A	6/1989	Eguchi et al.	5,796,952 A	8/1998	Davis
4,847,784 A	7/1989	Clancey	5,799,141 A	8/1998	Galipeau et al.
4,920,432 A	4/1990	Eggers	5,802,106 A	9/1998	Packer
4,922,417 A	5/1990	Churm et al.	5,802,291 A	9/1998	Balick et al.
4,943,932 A	7/1990	Lark et al.	5,812,769 A	9/1998	Graber
4,949,187 A	8/1990	Cohen	5,815,664 A	9/1998	Asano
4,949,248 A	8/1990	Caro	5,819,092 A	10/1998	Ferguson
5,029,232 A	7/1991	Nall	5,828,847 A	10/1998	Gehr
5,130,792 A	7/1992	Tindell et al.	5,832,506 A	11/1998	Kuzma
5,132,992 A	7/1992	Yurt et al.	5,832,514 A	11/1998	Norin et al.
5,136,716 A	8/1992	Harvey	5,835,718 A	11/1998	Blewett
5,172,413 A	12/1992	Bradley	5,845,303 A	12/1998	Templeman
5,191,573 A	3/1993	Hair	5,856,974 A	1/1999	Gervais et al.
5,253,275 A	10/1993	Yurt et al.	5,862,339 A	1/1999	Bonnaure et al.
5,253,341 A	10/1993	Rozmanith	5,867,706 A	2/1999	Martin et al.
5,287,499 A	2/1994	Nemes	5,867,799 A	2/1999	Lang et al.
5,287,537 A	2/1994	Newmark et al.	5,870,546 A	2/1999	Kirsch
5,291,554 A	3/1994	Morales	5,870,559 A	2/1999	Leshem et al.
5,341,477 A	8/1994	Pitkin et al.	5,878,212 A	3/1999	Civanlar et al.
5,371,532 A	12/1994	Gelman	5,884,038 A	3/1999	Kapoor
5,410,343 A	4/1995	Coddington	5,890,171 A	3/1999	Blumer et al.
5,414,455 A	5/1995	Hooper	5,893,116 A	4/1999	Simmonds et al.
5,442,389 A	8/1995	Blahut	5,894,554 A	4/1999	Lowery et al.
5,442,390 A	8/1995	Hooper	5,896,533 A	4/1999	Ramos et al.
5,442,749 A	8/1995	Northcutt	5,903,723 A	5/1999	Beck et al.
5,471,622 A	11/1995	Eadline	5,907,704 A	5/1999	Gudmundson et al.
5,475,615 A	12/1995	Lin	5,913,028 A	6/1999	Wang et al.
5,508,732 A	4/1996	Bottomley	5,913,033 A	6/1999	Grout
5,515,511 A	5/1996	Nguyen	5,918,010 A	6/1999	Appleman et al.
5,519,435 A	5/1996	Anderson	5,919,247 A	7/1999	Van Hoff et al.
5,528,281 A	6/1996	Grady	5,920,701 A	7/1999	Miller
5,539,621 A	7/1996	Kikinis	5,933,832 A	8/1999	Suzuoka et al.
5,542,087 A	7/1996	Neimat et al.	5,935,207 A	8/1999	Logue et al.
5,544,313 A	8/1996	Shachnai	5,945,989 A	8/1999	Freishtat et al.
5,544,327 A	8/1996	Dan	5,956,489 A	9/1999	San Andres et al.
5,550,577 A	8/1996	Verbiest	5,956,716 A	9/1999	Kenner
5,550,863 A	8/1996	Yurt	5,958,008 A	9/1999	Pogrebisky et al.
5,550,982 A	8/1996	Long	5,961,596 A	10/1999	Takubo et al.
5,557,317 A	9/1996	Nishio	5,966,440 A	10/1999	Hair
5,572,643 A	11/1996	Judson	5,968,121 A	10/1999	Logan et al.
5,590,288 A	12/1996	Castor	5,978,791 A	11/1999	Faber et al.
5,592,611 A	1/1997	Midgely	5,983,214 A	11/1999	Lang et al.
5,594,910 A	1/1997	Filepp et al.	5,983,227 A	11/1999	Nazem et al.
5,603,026 A	2/1997	Demers et al.	5,987,606 A	11/1999	Cirasioie et al.
5,619,648 A	4/1997	Canale	5,991,809 A	11/1999	Kriegsman
5,623,656 A	4/1997	Lyons	6,002,720 A	12/1999	Yurt et al.
5,625,781 A	4/1997	Cline	6,003,030 A	12/1999	Kenner et al.
5,627,829 A	5/1997	Gleeson et al.	6,006,264 A	12/1999	Colby et al.
5,630,067 A	5/1997	Kindell	6,012,090 A	1/2000	Chung et al.
5,633,999 A	5/1997	Clowes	6,014,686 A	1/2000	Elnozahy et al.
5,634,006 A	5/1997	Baughner et al.	6,014,698 A	1/2000	Griffiths
5,638,443 A	6/1997	Stefik et al.	6,018,516 A	1/2000	Packer
5,644,714 A	7/1997	Kikinis	6,026,440 A	2/2000	Sharder et al.
5,646,676 A	7/1997	Dewkett et al.	6,029,175 A	2/2000	Chow et al.
5,649,186 A	7/1997	Ferguson	6,029,176 A	2/2000	Cannon
5,659,729 A	8/1997	Nielsen	6,035,332 A	3/2000	Ingrassia, Jr. et al.
5,666,362 A	9/1997	Chen	6,038,216 A	3/2000	Packer
5,671,279 A	9/1997	Elgamal	6,038,310 A	3/2000	Hollywood et al.
5,675,734 A	10/1997	Hair	6,038,610 A	3/2000	Belfiore et al.
5,682,512 A	10/1997	Tetrick	6,041,324 A	3/2000	Earl et al.
5,699,513 A	12/1997	Feigen et al.	6,044,405 A	3/2000	Driscoll, III et al.
5,712,979 A	1/1998	Graber et al.	6,046,980 A	4/2000	Packer
5,715,453 A	2/1998	Stewart	6,049,831 A	4/2000	Gardell et al.
5,721,914 A	2/1998	DeVries	6,052,718 A	4/2000	Gifford
5,734,831 A	3/1998	Sanders	6,052,730 A	4/2000	Felciano et al.
5,740,423 A	4/1998	Logan et al.	6,065,051 A	5/2000	Steele et al.
5,742,762 A	4/1998	Scholl	6,065,062 A	5/2000	Periasamy et al.
5,751,961 A	5/1998	Smyk	6,070,191 A	5/2000	Narendran et al.
5,761,507 A	6/1998	Govett	6,078,943 A	6/2000	Yu
5,761,663 A	6/1998	Lagarde et al.	6,081,829 A	6/2000	Sidana
5,764,906 A	6/1998	Edelstein et al.	6,081,835 A	6/2000	Antcliff
5,774,660 A	6/1998	Brendel et al.	6,092,112 A	7/2000	Fukushige
5,774,668 A	6/1998	Choquier et al.	6,092,204 A	7/2000	Baker
5,777,989 A	7/1998	McGarvey	6,105,028 A	8/2000	Sullivan et al.
5,784,058 A	7/1998	LaStrange et al.	6,108,673 A	8/2000	Brandt et al.
			6,108,703 A	8/2000	Leighton
			6,112,231 A	8/2000	DeSimone et al.
			6,112,239 A	8/2000	Kenner
			6,112,240 A	8/2000	Pogue et al.

US 9,021,112 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

6,115,357 A	9/2000	Packer et al.	6,484,143 B1	11/2002	Swildens et al.
6,115,752 A	9/2000	Chauhan	6,484,204 B1	11/2002	Rabinovich
6,119,143 A	9/2000	Dias et al.	6,490,580 B1	12/2002	Dey et al.
6,125,388 A	9/2000	Reisman	6,490,615 B1 *	12/2002	Dias et al. 709/219
6,128,601 A	10/2000	Van Horne et al.	6,493,707 B1	12/2002	Dey et al.
6,128,660 A	10/2000	Grimm et al.	6,496,856 B1	12/2002	Kenner et al.
6,130,890 A	10/2000	Leinwand et al.	6,502,125 B1	12/2002	Kenner et al.
6,134,583 A	10/2000	Herriot	6,502,215 B2	12/2002	Raad et al.
6,144,375 A	11/2000	Jain et al.	6,505,248 B1	1/2003	Casper et al.
6,144,702 A	11/2000	Yurt et al.	6,529,477 B1	3/2003	Toporek et al.
6,144,996 A	11/2000	Starnes et al.	6,553,413 B1	4/2003	Leighton et al.
6,148,410 A	11/2000	Baskey et al.	6,553,420 B1	4/2003	Karger et al.
6,151,624 A	11/2000	Teare et al.	6,557,054 B2	4/2003	Reisman
6,154,738 A	11/2000	Call	6,564,251 B2	5/2003	Katariya et al.
6,154,744 A	11/2000	Kenner et al.	6,574,612 B1	6/2003	Baratti et al.
6,154,753 A	11/2000	McFarland	6,577,595 B1	6/2003	Counterterman
6,154,777 A	11/2000	Ebrahim	6,581,090 B1	6/2003	Lindbo et al.
6,163,779 A	12/2000	Mantha et al.	6,584,083 B1	6/2003	Toporek et al.
6,167,427 A	12/2000	Rabinovich	6,591,299 B2	7/2003	Riddle et al.
6,167,438 A	12/2000	Yates et al.	6,601,084 B1	7/2003	Bhaskaran et al.
6,173,311 B1	1/2001	Hassett et al.	6,611,862 B2	8/2003	Reisman
6,173,322 B1	1/2001	Hu	6,614,757 B1	9/2003	Rochberger et al.
6,175,869 B1	1/2001	Ahuja et al.	6,625,643 B1	9/2003	Colby et al.
6,178,160 B1	1/2001	Bolton	6,636,499 B1	10/2003	Dowling
6,181,867 B1	1/2001	Kenner et al.	6,654,344 B1	11/2003	Toporek et al.
6,185,598 B1	2/2001	Farber et al.	6,654,807 B2	11/2003	Farber et al.
6,185,619 B1	2/2001	Joffe et al.	6,658,464 B2	12/2003	Reisman
6,189,030 B1	2/2001	Kirsch et al.	6,665,706 B2	12/2003	Kenner et al.
6,189,039 B1	2/2001	Harvey et al.	6,665,726 B1	12/2003	Leighton et al.
6,195,680 B1 *	2/2001	Goldschmidt et al. 709/203	6,687,731 B1	2/2004	Kavak
6,205,120 B1	3/2001	Packer et al.	6,691,148 B1	2/2004	Zinky et al.
6,226,642 B1	5/2001	Beranek et al.	6,694,358 B1	2/2004	Swildens et al.
6,226,684 B1 *	5/2001	Sung et al. 709/238	6,699,418 B2	3/2004	Okada et al.
6,230,196 B1	5/2001	Guenthner et al.	6,708,137 B2	3/2004	Carley
6,240,461 B1 *	5/2001	Cieslak et al. 709/235	6,711,152 B1	3/2004	Kalmanek, Jr. et al.
6,243,760 B1	6/2001	Armbruster	6,718,328 B1	4/2004	Norris
6,249,810 B1	6/2001	Kiraly	6,722,211 B1	4/2004	Ciobanu et al.
6,256,675 B1	7/2001	Rabinovich	6,741,563 B2	5/2004	Packer
6,266,339 B1 *	7/2001	Donahue et al. 370/432	6,751,673 B2	6/2004	Shaw
6,266,699 B1	7/2001	Sevcik	6,754,699 B2	6/2004	Swildens et al.
6,269,394 B1	7/2001	Kenner et al.	6,754,706 B1	6/2004	Swildens et al.
6,275,470 B1	8/2001	Ricciulli	6,763,388 B1	7/2004	Tsimelzon
6,282,569 B1	8/2001	Wallis et al.	6,778,502 B2	8/2004	Ricciulli
6,282,574 B1	8/2001	Voit	6,779,017 B1	8/2004	Lamberton et al.
6,286,045 B1	9/2001	Griffiths et al.	6,785,704 B1 *	8/2004	McCanne 718/105
6,298,041 B1	10/2001	Packer	6,795,858 B1	9/2004	Jain et al.
6,311,214 B1	10/2001	Rhoads	6,799,221 B1	9/2004	Kenner et al.
6,314,465 B1	11/2001	Paul et al.	6,801,576 B1	10/2004	Haldeman et al.
6,314,565 B1	11/2001	Kenner et al.	6,823,362 B2 *	11/2004	Eshghi 709/203
6,330,605 B1	12/2001	Christensen et al.	6,834,306 B1	12/2004	Tsimelzon
6,332,195 B1	12/2001	Green et al.	6,842,604 B1	1/2005	Cook et al.
6,338,044 B1	1/2002	Cook et al.	6,870,851 B1	3/2005	Leinwand et al.
6,347,085 B2	2/2002	Kelly	6,874,032 B2	3/2005	Gersht et al.
6,360,256 B1	3/2002	Lim	6,901,604 B1	5/2005	Kiraly
6,370,571 B1	4/2002	Medin, Jr.	6,915,329 B2	7/2005	Kriegsman
6,370,580 B2	4/2002	Kriegsman	6,928,442 B2	8/2005	Farber et al.
6,374,299 B1	4/2002	Ford et al.	6,934,255 B1	8/2005	Toporek et al.
6,405,252 B1	6/2002	Gupta et al.	6,938,095 B2	8/2005	Basturk et al.
6,405,256 B1 *	6/2002	Lin et al. 709/231	6,950,623 B2	9/2005	Brown et al.
6,405,257 B1	6/2002	Gersht et al.	6,963,980 B1	11/2005	Mattsson
6,412,000 B1	6/2002	Riddle et al.	6,963,981 B1	11/2005	Bailey et al.
6,412,002 B1	6/2002	Denman et al.	6,965,890 B1	11/2005	Dey et al.
6,415,280 B1	7/2002	Farber et al.	6,970,432 B1	11/2005	Hankins et al.
6,421,726 B1	7/2002	Kenner et al.	6,973,485 B2	12/2005	Ebata et al.
6,424,992 B2 *	7/2002	Devarakonda et al. 709/203	6,973,490 B1	12/2005	Robertson et al.
6,430,618 B1	8/2002	Karger et al.	6,981,050 B1	12/2005	Tobias et al.
6,434,608 B1 *	8/2002	Desai 709/217	6,981,180 B1	12/2005	Bailey et al.
6,438,652 B1 *	8/2002	Jordan et al. 711/120	6,996,616 B1	2/2006	Leighton et al.
6,442,549 B1	8/2002	Schneider	7,003,572 B1	2/2006	Lownsbrough et al.
6,456,630 B1	9/2002	Packer et al.	7,007,089 B2	2/2006	Freedman
6,460,082 B1	10/2002	Lumelsky et al.	7,010,578 B1	3/2006	Lewin et al.
6,460,085 B1	10/2002	Toporek et al.	7,012,900 B1	3/2006	Riddle
6,463,454 B1	10/2002	Lumelsky et al.	7,039,633 B1	5/2006	Dey et al.
6,473,405 B2	10/2002	Ricciulli	7,047,300 B1	5/2006	Oehrke et al.
6,480,893 B2	11/2002	Kriegsman	7,054,935 B2	5/2006	Farber et al.
			7,058,706 B1	6/2006	Iyer et al.
			7,069,177 B2	6/2006	Carley
			7,075,897 B2	7/2006	Uematsu
			7,096,266 B2	8/2006	Lewin et al.

US 9,021,112 B2

Page 4

(56)

References Cited

U.S. PATENT DOCUMENTS

7,103,645 B2 9/2006 Leighton et al.
 7,103,647 B2 9/2006 Aziz
 7,136,374 B1 11/2006 Kompella
 7,181,523 B2 2/2007 Sim
 7,209,959 B1* 4/2007 Campbell et al. 709/219
 7,286,479 B2 10/2007 Bragg et al.
 7,343,422 B2 3/2008 Garcia-Luna-Aceves et al.
 7,653,700 B1* 1/2010 Bahl et al. 709/217
 7,693,959 B2* 4/2010 Leighton et al. 709/217
 7,734,730 B2* 6/2010 McCanne 709/219
 7,808,968 B1 10/2010 Kalmanek, Jr. et al.
 2001/0029525 A1 10/2001 Lahr
 2001/0042139 A1 11/2001 Jeffords et al.
 2001/0052016 A1* 12/2001 Skene et al. 709/226
 2001/0056500 A1 12/2001 Farber et al.
 2002/0018449 A1 2/2002 Ricciulli
 2002/0021675 A1 2/2002 Feldmann
 2002/0023164 A1 2/2002 Lahr
 2002/0023165 A1 2/2002 Lahr
 2002/0032777 A1 3/2002 Kawata et al.
 2002/0040404 A1 4/2002 Lahr
 2002/0042817 A1 4/2002 Lahr
 2002/0046273 A1 4/2002 Lahr et al.
 2002/0046405 A1 4/2002 Lahr
 2002/0049857 A1 4/2002 Farber et al.
 2002/0059592 A1 5/2002 Kiraly
 2002/0066038 A1 5/2002 Mattsson
 2002/0073199 A1 6/2002 Levine et al.
 2002/0075836 A1 6/2002 Uematsu
 2002/0078263 A1 6/2002 Darling et al.
 2002/0082999 A1 6/2002 Lee et al.
 2002/0083124 A1 6/2002 Knox et al.
 2002/0099850 A1 7/2002 Farber et al.
 2002/0116481 A1* 8/2002 Lee 709/220
 2002/0124080 A1 9/2002 Leighton et al.
 2002/0129134 A1 9/2002 Leighton et al.
 2002/0131645 A1 9/2002 Hamilton
 2002/0143798 A1 10/2002 Lisiecki et al.
 2002/0143888 A1 10/2002 Lisiecki et al.
 2002/0147774 A1 10/2002 Lisiecki et al.
 2002/0163882 A1 11/2002 Bornstein et al.
 2002/0166117 A1 11/2002 Abrams et al.
 2002/0199016 A1 12/2002 Freedman
 2003/0009444 A1 1/2003 Eidler et al.
 2003/0018966 A1 1/2003 Cook et al.
 2003/0028623 A1 2/2003 Hennessey et al.
 2003/0028626 A1 2/2003 Hennessey et al.
 2003/0028777 A1 2/2003 Hennessey et al.
 2003/0041238 A1 2/2003 French et al.
 2003/0055972 A1 3/2003 Fuller et al.
 2003/0061263 A1 3/2003 Riddle
 2003/0061280 A1 3/2003 Bulson et al.
 2003/0065761 A1 4/2003 Cereja et al.
 2003/0078888 A1 4/2003 Lee et al.
 2003/0078889 A1 4/2003 Lee et al.
 2003/0079005 A1 4/2003 Myers et al.
 2003/0079027 A1* 4/2003 Slocombe et al. 709/229
 2003/0105604 A1 6/2003 Ash et al.
 2004/0022194 A1 2/2004 Ricciulli
 2004/0139097 A1 7/2004 Farber et al.
 2004/0177148 A1 9/2004 Tsimelzon
 2004/0249960 A1 12/2004 Hardy et al.
 2005/0010653 A1* 1/2005 McCanne 709/219
 2005/0033858 A1 2/2005 Swildens et al.
 2005/0038851 A1 2/2005 Kriegsman
 2005/0100027 A1 5/2005 Leinwand et al.
 2005/0114296 A1 5/2005 Farber et al.
 2005/0262104 A1 11/2005 Robertson et al.
 2006/0143293 A1 6/2006 Freedman
 2006/0271705 A1 11/2006 Garcia-Luna-Aceves
 2008/0235400 A1* 9/2008 Slocombe et al. 709/245

FOREIGN PATENT DOCUMENTS

EP 0801487 A2 10/1997
 EP 0817444 A2 1/1998
 EP 0824236 A2 2/1998
 EP 0865180 A2 9/1998
 GB 2281793 3/1995
 JP 07-066829 3/1995
 JP 10-027148 1/1998
 JP 10-093552 4/1998
 JP 10-126445 5/1998
 JP 10-171727 6/1998
 JP 2001-053793 2/2001
 WO WO 96 42041 12/1996
 WO WO 97 11429 3/1997
 WO WO 97 29423 8/1997
 WO WO 98 04985 2/1998
 WO WO 98 06033 2/1998
 WO WO 99 09726 A 2/1999
 WO WO 99 29083 A 6/1999
 WO WO 99 40514 8/1999
 WO WO 00 52594 A 9/2000

OTHER PUBLICATIONS

Adler, R. M., "Distributed Coordination Models for Client/Server Computing," Computer 28, 4 (Apr. 1995), 14-22.
 Andresen et al., "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", Proc. IPPS, (Apr. 15, 1996), 850-856.
 Andresen, D., et al., Multiprocessor scheduling with client resources to improve the response time of WWW applications, Proc. 11th Intl Conf. on Supercomputing (Austria, Jul. 1997). ICS '97. ACM Press, NY, NY, 92-99.
 Basturk, E., et al., "Using network layer anycast for load distribution in the Internet", Tech. Rep., IBM T.J. Watson Research Center, 1997 (21 pgs.).
 Berners-Lee et al., RFC 1738—"Uniform Resource Locators (URL)", University of Minnesota, Dec. 1994, 1-25.
 Bestavros, A., "Speculative Data Dissemination and Service to Reduce Server Load, Network Traffic and Service Time in Distributed Information Systems", In Proc. ICDE '96: The 1996 Intl Conf. on Data Engineering, (Mar. 1996), 4 pages.
 Bestavros, et al., "Server-Initiated Document Dissemination for the WWW," IEEE Data Engineering Bulletin 19(3):3-11, Sep. 1996, 1-8.
 Bhattacharjee et al., "Application-layer anycasting," in Proc. IEEE INFOCOM '97, 1997, 1-9.
 Braun, H., et al., "Web traffic characterization: an assessment of the impact of caching documents from NCSA's web server", Comput. Netw. ISDN Syst. 28, 1-2 (Dec. 1995), 37-51.
 Brisco, T. P. RFC 1794: DNS support for load balancing, Apr. 1995, 1-7.
 Carter et al., "Dynamic server selection using bandwidth probing in wide-area networks," Tech. Rep. BU-CS-96-007, Comp. Sci. Dept., Boston University, Mar. 1996, 1-20.
 Carter et al., Server selection using dynamic path characterization in Wide-Area Networks, IEEE INFOCOM '97, (pp. 1014-1021), 1997, 1-8.
 Carter, J. Lawrence et al., "Universal Classes of Hash Functions", Journal of Computer and System Sciences, vol. 18, No. 2, 143-154 (Apr. 1979), 106-112.
 Chankhunthod, A. et al., "A Hierarchical Internet Object Cache", Proc. of the 1996 USENIX Technical Conf., Jan. 1996, pp. 153-163.
 Cisco Systems, Inc., Cisco DistributedDirector 4700-M Installation and Configuration Guide, pp. xix-xxii; 1-1 to 1-14; 7-1 to 7-18, 8-1 to 8-20; pub. date unknown, [downloaded Apr. 2007 from <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distrdir/dd4700m/>].
 Cisco Systems, Inc., Cisco DistributedDirector 2500 Series Installation and Configuration Guide, pp. xix-xxii; 1-1 to 1-12; 6-1 to 6-18; 7-1 to 7-18; 8-1 to 8-24, pub. date unknown, downloaded Apr. 2007: <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/distrdir/dd2501/>.
 Cisco Systems, Inc., "Cisco Distributed Director" http://www.cisco.com/warp/public/751/distrdir/dd_wp.htm, 1997, 16 pages.

US 9,021,112 B2

Page 5

(56)

References Cited

OTHER PUBLICATIONS

- Cohen, J., et al., "Cache Array Routing Protocol v1.1", Sep. 29, 1997; <http://tools.ietf.org/id/draft-vinod-carp-v1-01.txt> (Last-Modified: Wed, Oct. 1, 1997), 8 pages.
- Colajanni, M. and Yu, P. S. 1997. "Adaptive TTL schemes for load balancing of distributed Web servers. SIGMETRICS" Perform. Eval. Rev. 25, 2 (Sep. 1997), 36-42.
- "Content Management Technology/Industry News," Content Technologies Trends and Advice, Gilbane Report, News for Jun. 1999 [21 pages].
- Cormen, T. H., et al., "Introduction to Algorithms", The MIT Press, Cambridge, Massachusetts, (1990), 219-243, 991-993.
- Crovella et al., "Dynamic server selection in the Internet", 3rd IEEE Workshop on the Arch. and Implementation of High Performance Computer Sys. '95, pp. 158-162, Aug. 1995.
- Thaler, D. G. and Ravishankar, C. V. 1998. "Using name-based mappings to increase hit rates". IEEE/ACM Trans. Netw. 6, 1 (Feb. 1998), 1-14.
- Danzig, P. B., et al., An analysis of wide-area name server traffic: a study of the Internet Domain Name System, Conf. Proc. Communications Architectures & Protocols (Aug. 1992). D. Oran, Ed. SIGCOMM '92. ACM Press, New York, NY, 281-292.
- De Bra, P.M.E., et al., "Information Retrieval in the World Wide Web: Making Client-Based Searching Feasible", Computer Networks and ISDN Systems, NL, North Holland Publishing, Amsterdam, vol. 27, No. 2, ISSN: 0169-7552, (Nov. 1, 1994), 183-192.
- Deering, S. E., et al., "Multicast routing in datagram internetworks and extended LANs," ACM Trans. Comput. Syst. 8, 2 (May 1990), 85-110.
- Devine, R., "Design and Implementation of DDH: A Distributed Dynamic Hashing Algorithm", in Proc. 4th Int'l Conf. on Foundations of Data Organizations and Algorithms, (1993), 101-114.
- Doi, K. "Super Proxy Script—How to make distributed proxy servers by URL hashing," Sharp Corp., <http://naragw.sharp.co.jp/sps/>, dates unknown (1996-2000), download Jul. 7, 2007.
- "Exporting Web Server Final Report," http://www.cs.technion.ac.il/Labs/Lccn/projects/spring97/project4/final_report.html, Spring 1997 (downloaded Jul. 7, 2007).
- Feeley, M., et al., "Implementing Global Memory Management in a Workstation Cluster", in Proc. 15th ACM Symp. on Operating Systems Principles, (1995), 201-212.
- Floyd, S., et al., "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing", in Proc. of ACM SIGCOMM '95, 342-356, Aug. 1995.
- Fox, A., "A Framework for Separating Server Scalability and Availability from Internet Application Functionality", PhD thesis, University of California, Berkeley, 1998, 163 pages.
- Fox, A., et al., "Cluster-based scalable network services", Proc. 16th ACM Symp. on Operating Systems Principles (Saint Malo, France, Oct. 5-8, 1997). W. M. Waite, Ed. SOSP '97. ACM Press, New York, NY, 78-91.
- Fredman, M., et al., "Storing a Sparse Table with O(1) Worst Case Access Time", J. ACM, vol. 31, No. 3, (Jul. 1984), 538-544.
- Goldschmidt, et al., "Load Distribution for Scalable Web Servers: Summer Olympics 1996—A Case Study," in Proc. 8th IFIP/IEEE Int'l Workshop on Distributed Systems: Operations and Management, Sydney, Australia, Oct. 1997, 10 pages.
- Grigni, M., et al., "Tight Bounds on Minimum Broadcasts Networks", SIAM J. Disc. Math. 4 (1991), 207-222, (1991) 25 pages.
- Gulbrandsen, A., et al., "A DNS RR for specifying the location of services" (DNS SRV), Network Working Group, RFC 2052, Oct. 1996, 1-10.
- Guyton et al., "Locating nearby replicated Internet servers," Proc. ACM SIGCOMM '95, pp. 288-298, 1995, 1-18.
- Gwertzman, J., et al., "The Case for Geographical Push-Caching", Proc. Workshop on Hot OS '95, (May 4, 1995), 51-55.
- Gwertzman, J., et al., "World-Wide Web Cache Consistency", Proc. 1996 USENIX Tech. Conf., pp. 141-151, San Diego, CA, Jan. 1996, 11 pages.
- IBM Technical Disclosure Bulletin, "Local Area Network Server Replacement Procedure", vol. 38, No. 1, (Jan. 1995), 235-236.
- Jeffrey et al., Proxy-Sharing Proxy Servers, IEEE, pp. 116-119, 1996, 1-4.
- Karger, D., et al., "Consistent Hashing and Random Trees: Distributed Caching 43 Protocols for Relieving Hot Spots on the World Wide Web", in Proc. 29th Annual ACM Symp. on Theory of Computing, (May 1997), 654-663.
- Kwan et al., NCSA's World Wide Web Server: Design and Performance, IEEE, pp. 68-74, Nov. 1995, 1-23.
- Schemers, R., "Ibnamed—A load balancing name server written in Perl," 1995 LISA IX Sep. 17-22, 1995 Monterey, CA, 1-12.
- Litwin, W., et al., "LH*—A Scalable, Distributed Data Structure", ACM Trans. on Database Systems, vol. 21, No. 4, pp. 480-525, 1996, 1-43.
- Luotonen et al., World-Wide Web Proxies, CERN, Apr. 1994 (modified May 24, 1994), 1-8.
- Malpani, R., et al., "Making World Wide Web Caching Servers Cooperate", in Proc. 4th Int'l. World Wide Web Conf. (Dec. 1995), 10 pages (downloaded from <http://www.w3.org/Conferences/WWW4/Papers/59/> on Jul. 7, 2007), 10 pages.
- Panigrahy, R., "Relieving Hot Spots on the World Wide Web", Master's thesis, MIT EECS, Jun. 1997, pp. 1-66.
- Mockapetris et al., "Development of the Domain Name System," Proc. SIGCOMM '88 Computer Communications Review, vol. 18, No. 4, Aug. 1988, 112-122.
- Mockapetris, P., RFC 1034: Domain Names—Concepts and Facilities, Nov. 1987, 1-55.
- Mockapetris, P., RFC 1035: Domain Names—Implementation and Specification, Nov. 1987, 1-55.
- Mourad et al., "Scalable Web Server Architectures," iscc, 2nd IEEE Symposium on Computers and Communications (ISCC '97), 1997, 12-16.
- Nisan, N. 1990. Pseudorandom generators for space-bounded computations. In Proc. 22nd Annual ACM Symp. on theory of Computing (Baltimore, MD, U.S., May 13-17, 1990). H. Ortiz, Ed. STOC '90. ACM Press, New York, NY, 204-212.
- Oguchi et al., A Study of Caching Proxy Mechanisms Realized on Wide Area Distributed Networks, High Performance Distributed Computing, 5th Int'l Symposium, 1996 443-449.
- Palmer, M., et al., "Fido: A Cache that Learns to Fetch", in Proc. The 17th Intl Conf. on Very Large Data Bases, (Sep. 1991), 255-264.
- Peleg, D., et al., "The Availability of Quorum Systems", Information and Computation, 123, 210-223, (1995), 31 pages.
- Peterson, Eric, "Cisco Takes Global Route", PC Week News, (Feb. 17, 1997), p. 23.
- Petri S., et al., "Load Balancing and Fault Tolerance in Workstation Clusters. Migrating Groups of Communicating Processes.", Operating Systems Review, vol. 29, No. 4, Oct. 1995, 25-36.
- Plaxton, G. C., et al., "Fast Fault-Tolerant Concurrent Access to Shared Objects", in Proc. 37th IEEE Symp. of Foundations of Computer Science, 570-579, (1996), 10 pages.
- J. Postel, "Domain Name System Structure and Delegation", RFC 1591, Mar. 1994 1-7.
- Rabin, M. O., 1989, "Efficient dispersal of information for security, load balancing, and fault tolerance", J. ACM 36, 2 (Apr. 1989), 335-348.
- Ross, K.W., "Hash-Routing for Collections of Shared Web Caches", IEEE Network Magazine, 11, 7:37-44, Nov.-Dec. 1997, 1-21.
- Schuba, Christoph; "Addressing Weaknesses in the Domain Name System Protocol," COAST Laboratory, Dept. of Computer Sciences, Purdue University, West Lafayette, IN; Aug. 1993, p. 1-87.
- Neil Smith, "What can Archives offer the World Wide Web?", Technical Report 11, University of Kent, Computing Laboratory, University of Kent, Canterbury, UK, 1994, 1-12.
- Tarjan, Robert E., et al., "Storing a Sparse Table", Commun.ACM, 22, 11, (Nov. 1979), 606-611.
- Vitter, J. S., et al., "Optimal Prefetching via Data Compression," Proc. 32nd Annual IEEE Symposium on Foundations of Computer Science (Oct. 1991), 21 pages.
- Vixie, Paul; "Name Server Operations Guide for BIND," Internet Software Consortium; La Honda, CA; p. SMM:10-2-SMM:10-30 (undated, 1996), 1-30.

US 9,021,112 B2

Page 6

(56)

References Cited

OTHER PUBLICATIONS

- Jeff Walsh, "Global IP/PX Service Should Keep Network Delays Down", Infoworld, (Jan. 20, 1997), 1-2.
- Duane Wessels, "Intelligent Caching for World-Wide Web Objects", Masters Thesis, University of Colorado, 1995, 1-84.
- Yao, A. C. 1981. "Should Tables Be Sorted?". J. ACM 28, 3 (Jul. 1981), 615-628.
- Narten, T., IBM, "Neighbor Discovery for IP Version 6", RFC 2461, Dec. 1998.
- Bestavros, "Demand-Based Document Dissemination to Reduce Traffic and Balance Load in Distributive Information Systems," IEEE Symposium on Parallel and Distributed Processing (PDP), San Antonio, TX, Oct. 1995.
- Cisco Distributed Director (1996).
- Mockapetris, "Domain Names—Concepts and Facilities," IETF RFC 1034, Nov. 1987.
- Mockapetris, "Domain Names—Implementation and Specification," IETF RFC 1035, Nov. 1987.
- Brisco, T., "DNS Support for Load Balancing," IETF RFC 1794, Apr. 1995.
- Berners-Lee, T. & Connolly, D., "Hypertext Markup Language—2.0," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1866, Nov. 1995.
- Wessels, D., et al., "Application of Internet Cache Protocol (ICP), version 2" Internet Engineering Task Force (IETF) Request for Comments (RFC) 2187, Sep. 1997.
- Berners-Lee, T., et al., "Hypertext Transfer Protocol—HTTP/1.0," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1945, May 1996.
- Berners-Lee, T. "Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1630, Jun. 1994.
- Berners-Lee, T., et al., RFC 1738—Uniform Resource Locators, Dec. 1994.
- Callon, R., "Use of OSI IS-IS for Routing in TCP/IP and Dual Environments," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1195, Dec. 1990.
- Chandra, R. et al., "BGP Communities Attribute," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1997, Aug. 1996.
- Chen, E., et al., "An Application of the BGP Community Attribute in Multi-home Routing," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1998, Aug. 1996.
- Colajanni, M., et al., "Scheduling Algorithms for Distributed Web Servers," International Conf. on Distributed Computing Systems (ICDCS), Baltimore, Maryland, USA, May 1997.
- Eriksson, H., "MBONE: The Multicast Backbone," Communications of the ACM, vol. 37, No. 8, p. 54-60, Aug. 1994.
- Wessels, D., et al., "Internet Cache Protocol (ICP), version 2," Internet Engineering Task Force (IETF) Request for Comments (RFC) 2186, Sep. 1997.
- Fielding, R., et al., "Hypertext Transfer Protocol—HTTP/1.1," Internet Engineering Task Force (IETF) Request for Comments (RFC) 2068, Jan. 1997.
- Malkin, G., "RIP Version 2 Carrying Additional Information," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1388, Jan. 1993.
- Moy, J., "OSPF Version 2," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1583, Mar. 1994.
- Parris C., et al., "A Dynamic Connection Management Scheme for Guaranteed Performance Services in Packet-Switching Integrated Services Networks," UC Berkeley Computer Science Division Tech. Report TR-93-005, 1993.
- Partridge, C., et al., "Host Anycasting Service," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1546, Nov. 1993.
- Rekhter Y., et al., "A Border Gateway Protocol 4 (BGP-4)," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1771, Mar. 1995.
- Rekhter Y., et al., "Application of the Border Gateway Protocol in the Internet," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1772, Mar. 1995.
- Topolcic, C., "Experimental Internet Stream Protocol, Version 2 (ST-II)," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1190, Oct. 1990.
- Touch, J. et al., "The X-Bone," Third Global Internet Mini-Conference at Globecom '98, Sydney, Australia, Nov. 1998, pp. 59-68 (pp. 44-52 of the mini-conference).
- Traina, P., "BGP-4 Protocol Analysis," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1774, Mar. 1995.
- Traina, P., "Experience with the BGP-4 protocol," Internet Engineering Task Force (IETF) Request for Comments (RFC) 1773, Mar. 1995.
- Baentsch, M., et al., "Introducing Application-Level Replication and Naming into Today's Web," Computer Networks and ISDN Systems, vol. 28, No. 7-11, pp. 921-930, May 1996.
- Bhattacharjee et al., "Application-layer anycasting," in Proc. IEEE INFOCOM '97, Apr. 1997.
- Almeroth, K., et al. "Scalable Delivery of Web Pages Using Cyclic Best-Effort (UDP) Multicast", IEEE INFOCOM, San Francisco, California, USA, Jun. 1998.
- Collins, A., "The Detour Framework for Packet Rerouting," University of Washington, Oct. 29, 1998.
<http://web.archive.org/web/19990202122310/http://www.cs.washington.edu/research/networking/detour/>.
- Savage, S., et al., "Detour: Informed Internet Routing and Transport," IEEE Micro, Jan.-Feb. 1999.
- Schemers, R., "Ibnamed: A Load Balancing Name Server in Perl," 1995 LISA IX, Monterey, California, Sep. 17-22, 1995.
- Berners-Lee, T., et al., "Hypertext Transfer Protocol—HTTP/1.0," Internet Engineering Task Force (IETF) Internet Draft, draft-ietf-http-v10-spec-00.ps, Mar. 1995.
- Fielding, R., et al., "Hypertext Transfer Protocol—HTTP/1.1," Internet Engineering Task Force (IETF) Internet Draft, draft-ietf-http-v11-spec-00.txt, Nov. 1995.
- Final Office Action, mailed Oct. 22, 2013, U.S. Appl. No. 09/982,721, filed Oct. 18, 2001, 14 pgs.

* cited by examiner

U.S. Patent

Apr. 28, 2015

Sheet 1 of 5

US 9,021,112 B2

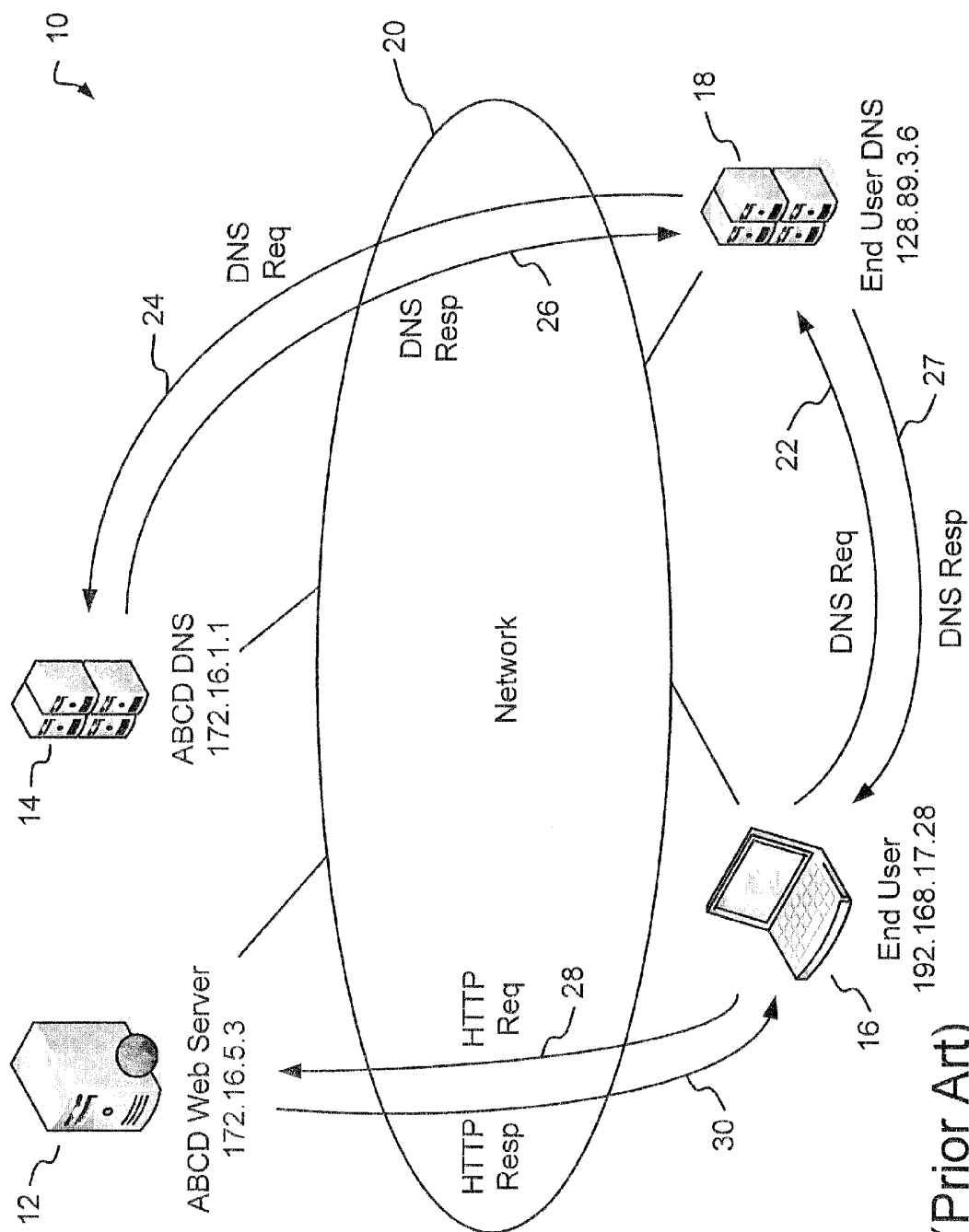


FIG. 1 (Prior Art)

U.S. Patent

Apr. 28, 2015

Sheet 2 of 5

US 9,021,112 B2

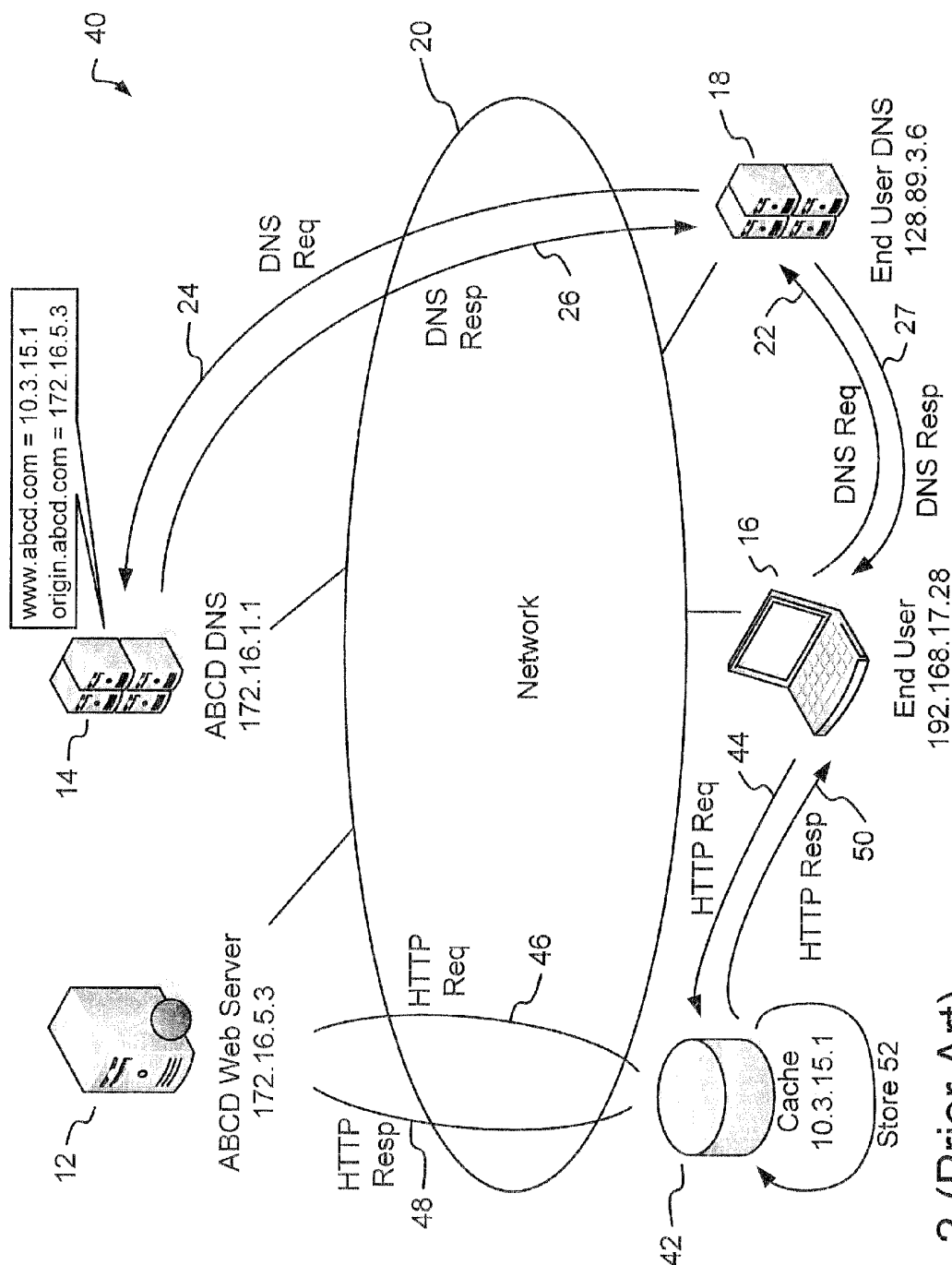


FIG. 2 (Prior Art)

U.S. Patent

Apr. 28, 2015

Sheet 3 of 5

US 9,021,112 B2

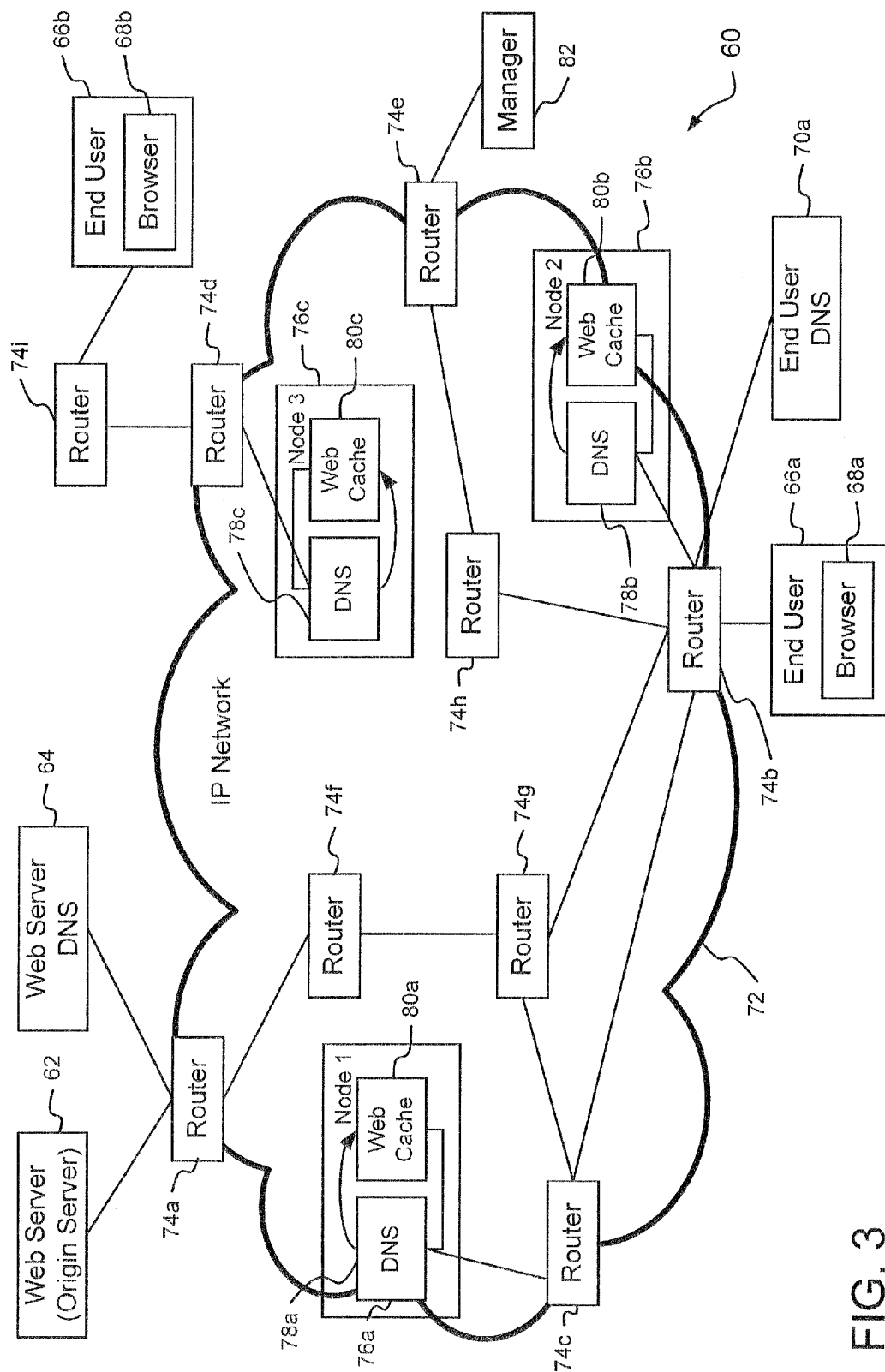


FIG. 3

U.S. Patent

Apr. 28, 2015

Sheet 4 of 5

US 9,021,112 B2

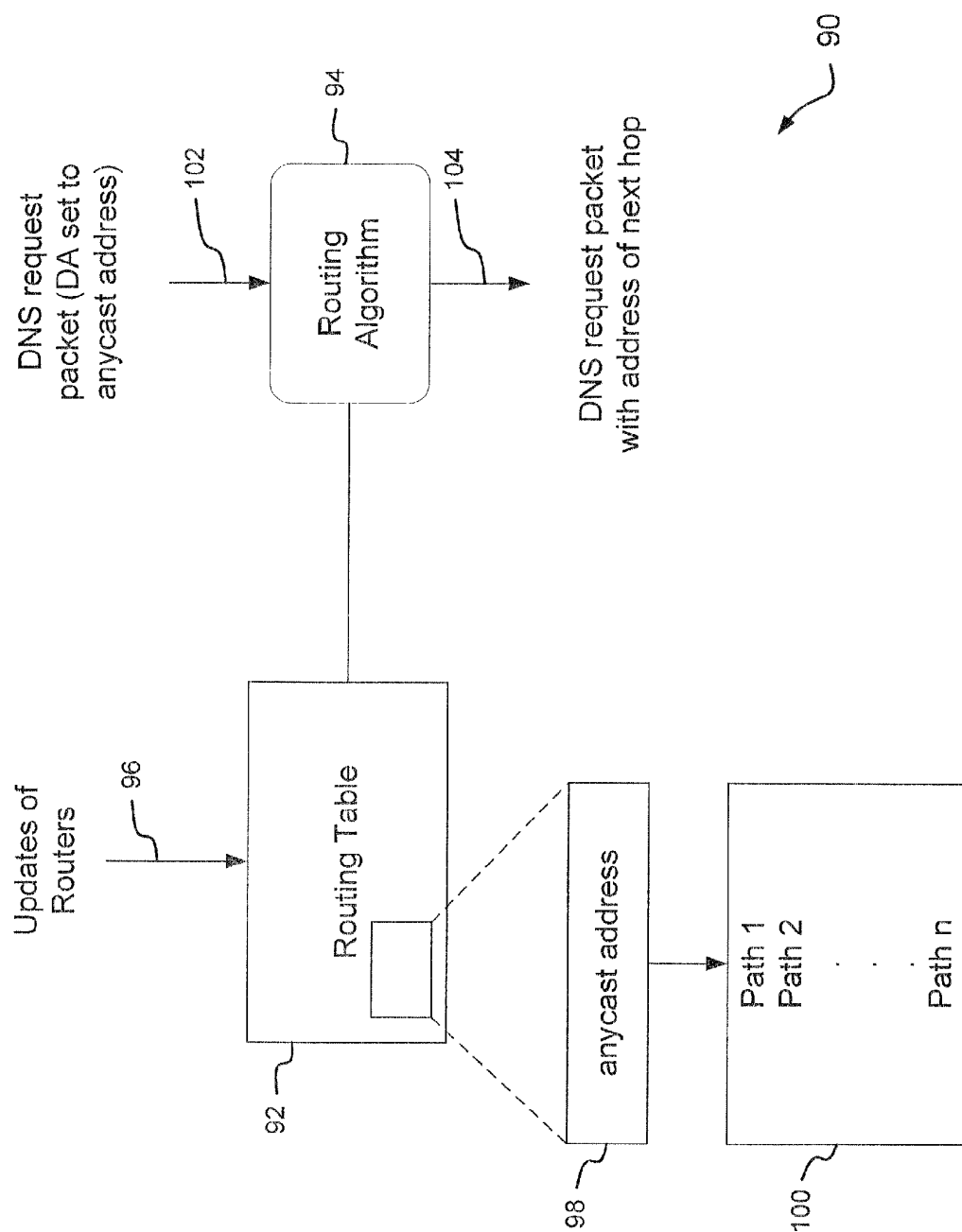
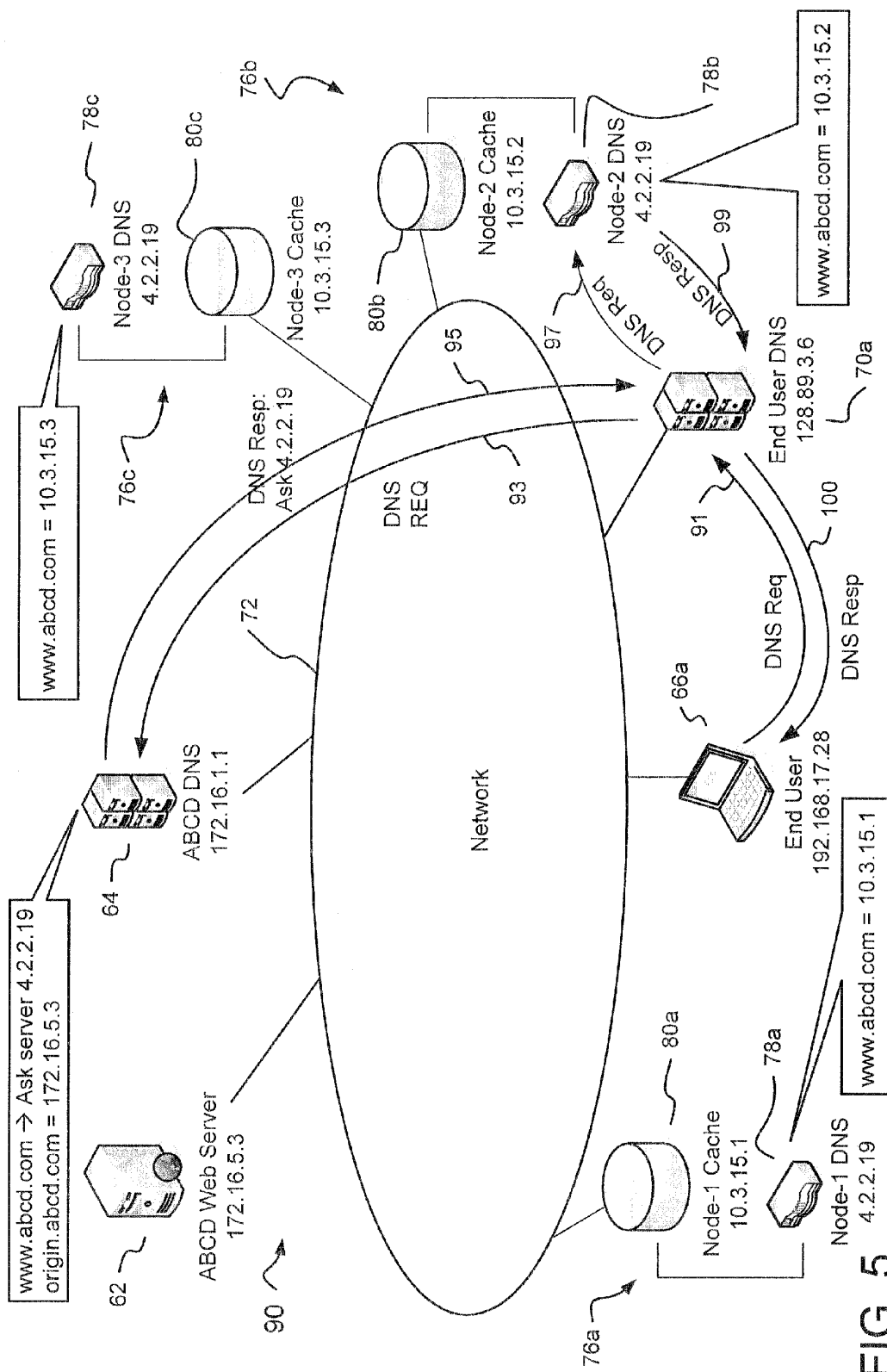


FIG. 4



US 9,021,112 B2

1

CONTENT REQUEST ROUTING AND LOAD BALANCING FOR CONTENT DISTRIBUTION NETWORKS

CROSS REFERENCE TO RELATED APPLICATIONS

This is a divisional application of U.S. application Ser. No. 09/982,721 entitled "Content Request Routing and Load Balancing for Content Distribution Networks", filed Oct. 18, 2001, which is hereby incorporated by reference in its entirety.

BACKGROUND

Embodiments of the invention relate generally to information retrieval in a network and, more particularly, to hosting and distributing content on a content delivery network such as the Internet.

The World Wide Web is the Internet's content retrieval system. In the Web environment, client systems effect transactions to Web servers using the Hypertext Transfer Protocol (HTTP), which provides clients with access to files (e.g., text, graphics, images, sound, video, etc.) using a standard page description language, for example, Hypertext Markup Language (HTML). A network path to a server is identified by a so-called Uniform Resource Locator (URL) having a special syntax for defining a network connection. Use of a Web browser at a client (end user) system involves specification of a link via the URL. In response, the client system makes a request to the server (sometimes referred to as a "Web site") identified in the link and, in return, receives content from the Web server. The Web server can return different content data object types, such as .gif and jpeg files (graphics), .mpeg files (video), .wav files (audio) and the like.

As the Web server content provided by World Wide Web has continued to grow over time, so too has the number of users demanding access to such content. Unfortunately, the ever-increasing number of end users requesting Web content from Web sites has resulted in serious bandwidth and latency issues, which manifest themselves in delay to the end user.

To address these problems, many networking product and service providers have developed solutions that distribute Web site content across the network in some manner. One class of solutions involves replicating Web servers at multiple locations and directing traffic (by modifying the URL and forwarding, or using HTTP re-direct) to the "best" server based on a predefined selection policy, e.g., load balancing, network topology. Another class of solutions distributes content strategically and/or geographically, and often uses some type of centralized or hierarchical Domain Name System (DNS)-based site selection. The distributed sites include servers that perform reverse proxy with (or without) caching. One such technique routes traffic to a content distribution site nearest the requestor by modifying URLs in the top-level Web page. Other DNS techniques use a round robin traffic distribution to distribute load to the content sites, but do not take into account the location of the requestor relative to those content sites.

SUMMARY

In one aspect, embodiments of the invention provide a method of content delivery in a network. The method includes associating devices in a Domain Name System (DNS) with content server systems located in the network, the content server systems maintaining and serving content of a content

2

provider, each DNS device configured to resolve the name of the content provider to an address for the content server system with which such DNS device is associated. The method further includes assigning to the DNS devices a common address, the common address being usable to resolve the name of the content provider such that a request for content of the content provider by a content requester is sent to the content server system nearest the content requestor.

Particular implementations of the invention may provide one or more of the following advantages.

A performance benefit is gained because a content requestor can generally retrieve content from a content site closer than the origin server of the content provider. In addition, because there are multiple sites serving the content, the load from many end users is distributed among multiple systems and across different parts of the network. Also, an end user's DNS request can be routed to a content site nearest the requestor using pre-existing routing infrastructure. Because DNS uses a stateless protocol (UDP) for routing, the solution can handle anycast addressable caching without the problems associated with anycast service, namely, the potential packet-by packet load balancing site effects of protocols like TCP which maintain state information.

Other features and advantages of the invention will be apparent from the following detailed description and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

FIG. 1 is a depiction of a prior Web transaction.

FIG. 2 is a depiction of a prior Web transaction using reverse-proxy caching.

FIG. 3 is a block diagram of an exemplary content distribution network which includes content distribution nodes that support reverse-proxy caching and anycast address service to achieve distributed content delivery and load balancing.

FIG. 4 is a simplified block diagram of routing configuration software and associated table(s) (provided in the routing infrastructure of the content distribution network of FIG. 3) used to perform distributed content site selection based on anycast address routing and Border Gateway Protocol (BGP).

FIG. 5 is a depiction of an exemplary Web transaction using the content distribution network shown in FIG. 3.

Like reference numbers will be used to represent like elements.

DETAILED DESCRIPTION

Embodiments of the present invention feature a content distribution mechanism for routing a content request to the nearest content distribution site in a content distribution network. It also provides for load sharing across multiple content distribution sites in a content distribution network. The content distribution mechanism has particular utility in and is therefore described within the context of an Internet-based, World Wide Web ("Web") network infrastructure.

Hereinafter, the following terminology is used:

"Content provider" refers to an entity having a Web site on a network. Generally, the entity produces the content for the Web site. The entity may operate a Web server system or may use the services of a hosting provider.

"End user" refers to a person who wishes to view the Web content. Typically, an end user uses a Web browser, such as Netscape Navigator or Internet Explorer, executing on a computer system, e.g., a personal computer.

US 9,021,112 B2

3

"Domain Name System" (DNS) refers to a collection of systems available on the public Internet that can resolve a domain name to a specific Internet Protocol (IP) address, as is known in the art.

It will be understood that reference to distance on the network, such as one server being "closer" to an end user than another server, refers to a network distance. Thus, a shorter distance implies a better path based on network criteria, and not necessarily a shorter geographic distance.

In the description to follow, a fictitious company "ABCD" is used as an example of a content provider having a Web site on a network.

FIG. 1 shows a conventional content delivery Web transaction 10. The transaction 10 involves a Web server 12 (also referred to as an "origin server"), an authoritative DNS system 14, an end-user system 16 and an end-user DNS system 18, all of which are coupled to a network 20, for example, the public Internet. The Web server 12 is a computer system containing content (e.g., Web pages) for a content provider, with the ability to provide this content in response to a Web request via the HTTP protocol. The authoritative DNS system 14 is a DNS system that can resolve domain names within the content provider's namespace.

For example, the DNS system 14 for the company "ABCD" would have information for host names ending in ".abcd.com" (such as "www.abcd.com"). Typically, the end-user system 16 is a computer being operated by an end user to perform Web "browsing" (that is, view Web pages). The end-user DNS system 18 is a DNS server that the end-user system 16 uses to resolve domain names to IP addresses.

When the end user wishes to view a Web page or object (such as "www.abcd.com/PriceList"), the transaction 10 occurs as follows. First, the end user using the end user system 16 enters the name of a Web page into a browser (not shown) executing on the end user system 16. The end-user system 16 requests a DNS resolution for the host name ("www.abcd.com") from the end-user DNS system 18 ("DNS Req 22"). The end-user DNS system 18 determines which of the DNS systems that make up the DNS for the network 20 can resolve this host name by sending a DNS request to the authoritative DNS system 14 ("DNS Req 24"). The authoritative DNS system 14 resolves the name to an IP address and returns a response containing the IP address to the end-user system 16 via the end-user DNS system 18 ("DNS Resp 26"). The end-user DNS system 18, in turn, communicates the IP address to the end-user station 16 in a DNS response to the end-user system 16 ("DNS Resp 27"). The end-user system 16 contacts the Web server 12 at the specified IP address and requests the Web object ("www.abcd.com/PriceList") ("HTTP Req 28"). The Web server 12 returns the Web page corresponding to the requested Web object to the end-user system 16 ("HTTP Resp" 30). The browser running on the end-user station 16 displays the returned Web page on the end-user system 16 for viewing by the end user.

FIG. 2 illustrates a Web transaction with reverse proxy caching 40. That is, the transaction 40 is the same basic Web transaction as shown in FIG. 1, but now employs a reverse proxy content server (shown as a cache server) 42 acting on behalf of the Web server 12. During reverse-proxy caching, the cache server 42 assumes the identity of the Web server 12 so that a Web request directed to the Web server 12 (in the example, "www.abcd.com") is instead directed to the cache server 42. This re-direction is accomplished by changing the entry for the Web site ("www.abcd.com") in the authoritative DNS system 14 so that the host name resolves to the address of the cache server 42 instead of the address of the original Web server 12. A new name is assigned to the address of the

4

original Web server 12 (e.g., "origin.abcd.com"). Thus, 22, 24, 26 and 27 are the same as in FIG. 1. However, the IP address returned by the authoritative DNS system 14 is that of the cache server 42, not the Web server itself, as was previously described with reference to FIG. 1. Consequently, instead of sending the subsequent content request to and receiving a response from the Web server 12 (as was shown in steps 28 and 30 of FIG. 1), the end-user system 16 sends the content request to the cache server 22 ("HTTP Req 44") and the cache server 22 fetches the requested content from the Web server ("HTTP Req 46"). The Web server 12 returns the requested content to the cache server 42 ("HTTP Resp 48"). The cache server 42 completes the transaction by serving the requested content to the end-user system 16 ("HTTP Resp 50") and caching the content so the content will be readily available to support future requests for the same content ("store" 52).

FIG. 3 shows a content distribution network ("CDN") 60. The CDN 60 includes at least one Web server, shown as a Web server 62, a DNS system 64 for Web server 62, and end-user stations 66a and 66b, all connected to a network 72. The network 72 is implemented as the public Internet. The end-user stations execute Web browsers 68a and 68b, respectively. Each end-user station 66 has an associated end user DNS, however, only an end user DNS 70a for the end-user station 66a is shown. Of course, additional end-user stations may be connected to the network 72. Also connected to the network 72 are multiple content distribution nodes 76a, 76b and 76c, which support distributed content delivery services for one or more Web sites on the network 72. The CDN nodes 76 interact with the origin server 62 containing the original Web content, the various DNS systems 64, 70 and, of course, the end-user systems 66.

Each of the CDN nodes 76 includes a DNS system 78 coupled to and associated with a Web content server system or site 80. In one embodiment, as described herein, each content server system 80 is implemented as a cache server system. The techniques described herein could also apply to other types of content servers, such as mirrored Web content servers, or Web content servers having different content (e.g., customized for geographic area). Each DNS system 78 in each node holds a table that includes an address entry which the DNS system 78 uses to map the domain name of the content provider to the IP address of the cache server in that same node. Although only one such Web site (Web site 62) is shown, it will be appreciated that other Web sites may be connected to the network 72 and use the DNS and content caching services of the nodes 76, as will be described. The nodes 76 are deployed at different locations in the network 72. Preferably, the nodes 76 are geographically distributed across the network 72.

Optionally, the CDN 60 may include a CDN manager 82 that can be used by a network administrator (for example, a CDN node hardware and/or CDN node service provider) to configure the CDN to use the CDN nodes.

The network 72 is intended to represent a simplified view of the Internet. In the simplified depiction of FIG. 3, the network 72 includes a plurality of interconnected routers or routing networks, e.g., routers 74a, 74b, . . . , 74g, for routing packets from one domain to another within network 60. In actuality, the Internet is made up of many private "routing networks" (networks including one or more routers, and possibly other types of networking components as well), e.g., local, regional and centralized Internet Service Providers (ISPs), some of which are connected to Network Access Points (NAPs) or exchanges, or each other at public or private peering points. In the simplified Internet configuration shown

US 9,021,112 B2

5

in FIG. 3, routers **74a**, **74b**, **74c** **74d** and **74e** are located at network entry points. The end user station **66a** and end user DNS system **70a** connect to the network **72** via the router **74a**. The end user station **66b** is coupled to another router **74i**, which connects to the router **74d**. The Web Server **62** and associated DNS system **64** are connected to the network **72** via the router **74b**. Preferably, to the extent possible, and for reasons which will be discussed below, the end user DNS systems such as system **70a** are located near the end user systems with which they are associated.

Also, preferably, the geographically dispersed nodes **76** are located so as to be as close as possible to various network entry points, exchanges or both. The network entry points each may correspond to an ISP Point of Presence (POP). In FIG. 3, for illustrative purposes, the nodes **76a**, **76b** and **76c** are shown as being connected to entry point routers **74c**, **74b** and **74d**, respectively, but need not be directly connected to network access routers in this manner.

The caching servers **80** have unique IP addresses. The DNS systems **78** share a common IP address as well as have unique IP addresses. The end-user DNS systems, e.g., end user DNS system **70a**, resolve to the common address. That is, the end-user DNS system **70a** knows which DNS system (in this example, the DNS system **64**) has an address for a high level domain server, e.g., .com, org, and maintains tables of all domain names and knows which server (authoritative DNS server) to consult for the address of the domain server. Thus, the address lookup table in the DNS system **64** is configured to indicate that a server corresponding to the common address can resolve the domain name of the content provider to an IP address.

One way to implement this content distribution configuration is to use an anycast address as the common address. An anycast address is a unicast address that is used in multiple places. Various Internet Engineering Task Force (IETF) Internet Requests for Comments (RFCs) describe implementations of anycast addresses in IP networks. The IETF is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. The following anycast-related RFCs are hereby incorporated by reference in their entirety for all purposes: RFC 1546 (November 1993); RFC 2372 (July 1998); RFC 2373 (August 1998); and RFC 2526 (March 1999).

As described in RFC 1546, an anycast address may include a subnet prefix identifier and an anycast identifier. The subnet prefix may be used to specify the network providing the anycast addresses. The anycast identifier is used to specify one of many possible anycast addresses on a particular subnet. A unicast address, or conventional IP address, specifies a single interface on a computer network. In contrast, an anycast address may specify more than one interface. For example, anycast addresses may be used to specify a group of one or more servers on a computer network. These servers may provide a redundant service. Routers forward packets destined to anycast addresses to the closest anycast destination for a particular address. Thus, anycast addresses provide a way to distribute load across one or more servers.

The anycast address is advertised to the network **72** from each node **76** using a dynamic routing protocol, the Border Gateway Protocol (BGP). The BGP is a routing protocol used to exchange network reachability information between Internet border routers. It enables those routers to make intelligent routing decisions as to the best path. The BGP is used by such routers as their exterior routing protocol in order to advertise routes to other border routers. BGP uses TCP as its transport protocol for exchanging routing information. Adjacent rout-

6

ers running BGP set up a TCP connection to exchange entire routing tables. Each router has enough information to determine an optimal next hop to a destination. The BGP is also described in various RFCs, including RFC 1267 (October 1991) and RFC 1654 (July 1994), incorporated herein by reference.

Referring to FIG. 4, routing configuration support **90** in a router such as router **74b** includes a routing algorithm (software) **94** and a routing table **92**. The router **74b** receives address path information from the nodes **76** in accordance with the BGP route information exchanges and updates **96**, including the anycast address **98** and associated paths **100** for each of the nodes, and stores them in the routing table **92**. As a result of the BGP table information exchanges and updates, routers in the network **72** maintain pointers (that is, the paths or routes) that allow it to determine the next hop to every unique address in the network, as well as multiple pointers to the anycast address. The routing algorithm **94**, in response to receipt of a DNS request packet **102** from an end-user system **66** for resolution of a DNS name for the content provider, uses the path information **100** stored in the routing table **92** to select a path to the nearest CDN node. Thus, a router may see multiple connections to the anycast address, but selects the path that represents the shortest network distance (e.g., the topologically shortest path). More specifically, as this routing occurs as part of a DNS resolution, the router selects a route to the closest DNS system **78**. Because the selected DNS system resolves to the address of the cache server in the DNS system's node, the DNS anycast routing, in essence, serves to select the local content site (cache) from which content will be served.

Referring now to FIG. 5, a Web transaction **90** occurring over a CDN (such as the CDN **60** shown in FIG. 3) is illustrated for the running example of the "ABCD" Web site. To provide content distribution service for this site, the following configurations are implemented. The original Web server **62** is renamed in the DNS system **64** to reflect a change in its role from being the primary Web server to becoming the original content source for the cache servers **80**. The domain "www.abcd.com" is renamed "origin.abcd.com". In the end-user DNS system **70a**, an entry is made to indicate that the CDN DNS system **78** is the authoritative server for the end user web domain. For example, in the dns.abcd.com system, the entry for "www.abcd.com" no longer resolves to a specific address but instead refers the content requester to the CDN DNS server **78** at the anycast address of 4.2.2.19. The CDN DNS server entry resolves to the address of the associated node's caching server **80**. As a result, each node **76** resolves the Web site name to a different address. In the above example, DNS server **78a** in node **76a** (node 1) resolves the name "www.abcd.com" to 10.3.15.1, whereas in node **76b** (node 2), the DNS server **78b** resolves the same address to 10.3.15.2. In this manner, therefore, the entry made in the CDN DNS server **78** in each node enables distribution of the Web site among the cache servers **80** in their respective nodes and locations, as will be described in further detail below.

Once the appropriate configurations have been completed, it is assumed that the Web site www.abcd.com is being handled by the CDN nodes **76**. Each node **76** advertises the anycast address of its own DNS server **78** to the network **72** using the BGP protocol, as discussed above. The address of the DNS server at each node is identical. That is, from a network point of view, the network "thinks" it is connected to a single host at multiple points.

Referring to FIGS. 3 and 5, the Web transaction operation **90** is as follows. When an end user of the end-user station **66a** requests an object from the now "accelerated" Web site, the

US 9,021,112 B2

7

end-user system 66a first resolves the content provider name via DNS. That is, the end-user station (acting as a content requestor) sends a request to the local DNS server 70a ("DNS Req 92"). That server 70a resolves the name, ultimately by sending a request to the CDN DNS anycast address. The DNS server 70a sends a request to the ABCD DNS server 64 ("DNS Req 94"), which returns the anycast address of the CDN nodes 76 to the requesting DNS server 70a (DNS Resp 96). The DNS server 70a then transmits a DNS name resolution request addressed to the anycast address and that request is generally routed (by various ones of the network routers 74a-74h) to the CDN node nearest the user's DNS system ("DNS Req 96"). In this instance, because the request enters the network via the router 74b, the router 74b determines that the shortest path to the anycast address is the path to the node 76b (node 2). At that node, the DNS server 78b resolves the name to the cache server address for that node, that is, IP address 10.3.15.2 assigned to the cache server 80b, and returns the cache server address to the end-user DNS server 70a ("DNS Resp 98"). The end-user DNS server 70a forwards the address to the end-user system 66a ("DNS Resp 100"). From this point on, the remaining steps of the transaction are much the same as steps 44 through 52, shown in FIG. 2. That is, the end-user system 70a requests the Web object from the cache server in the nearby node where the DNS name was resolved, i.e., node 76b, and that node's cache server 80b, in turn, checks for a cached version of the object (if cached content already resides on the cache server). If no content has yet been cached in the server, or the cached copy is stale or otherwise invalid, the cache server 80b retrieves the Web object from the origin server 62, and serves the object to the end-user system that requested it.

Typically, as is known in the art, Web content can be marked with certain caching attributes, for example, whether or not the content is at all cacheable, how long the content may be held in cache. In the case of the former attribute, if the content is marked as uncacheable (e.g., dynamic or content containing sensitive information), the cache server discards the content after serving it to the requester. Otherwise, if the content is cacheable, the cache server will store the content in local storage and maintain the cached content according to any other cache attributes. Content can be localized, for example, using ad insertions with local content. Content in the caches can be pre-loaded (all the cache servers receiving the same content). That is, the content can be replicated on all cache servers so that even the first request will have a fast response time. Preferably, the caches are not preloaded with content but instead build their cached content based on user requests/usage over time. Content is retrieved from the origin server 12 when a first user request is received, and then stored locally. If subsequent requests are received for the same content, the cached copy is used if it is still valid, as was mentioned earlier. Thus, the cache server need not retrieve the content from the origin server again. Each cache server contains a translation table so that it knows where to retrieve any particular Web page from the origin server. For example, the cache server 80b would know that the page "www.abcd.com/PriceList" can be retrieved from "origin.abcd.com/PriceList".

Preferably, the CDN node contains software to monitor the load in various parts of the node cache system (disk, CPU, I/O, et cetera) by determining at least one load metric value (based on metrics such as utilization, latency, etc.) and comparing each such metric value to a predefined overload threshold. Upon reaching a predefined overload threshold, the monitoring software informs the routing software in the CDN DNS server to withdraw its BGP routing advertisement.

8

Thus, under normal conditions, all CDN nodes are advertising the address of their DNS servers to the network, and so a DNS request will be directed to the nearest CDN node. If a node becomes heavily loaded and detects an overload condition through its internal monitoring, the node stops advertising its DNS address to the network so that no further requests will be directed to that node. Consequently, DNS requests that normally would have been routed to that node as a first choice are routed to the next closest active node.

This overload detection and load balancing mechanism has the advantage that Web transactions already in progress are not interrupted by a shift in resources. Any system that has already resolved a DNS name to the now inactive node will continue using that node until the DNS name expires. The load in that node will slowly decrease until such time as the node can start accepting new clients, at which time it will start advertising its DNS system address to the network again.

Other embodiments are contemplated. For example, it is possible to use an anycast scheme with the cache servers themselves. With reference to the system shown in FIG. 3, the network routing directs the DNS request from the end user's DNS system, such as system 70a, to the closest of the CDN nodes 76. Although the node is the node closest to the end user's DNS system, it may not necessarily be the closest to the end-user system. In those cases where the end user's DNS system is a substantial distance from the end user, it is possible that the end user system will use a CDN node that is not the closest one to the end user system. Allowing the cache servers to use a common address would ensure that the end user's Web request is indeed routed to the nearest CDN node. There is a significant drawback associated with using the anycast-addressable cache server approach, however. The client/server portion of the transaction uses the TCP protocol, thus requiring multiple exchanges between the end user and the cache server to complete a transaction. With anycasting, there is no guarantee that subsequent packets in a transaction will be routed to the same server. In cases where the packets are split between two or more cache servers, a successful transaction cannot occur. In contrast, although requiring that the end user DNS system be located in close proximity to the end user system for optimal CDN performance, the anycast-based DNS resolution is completed using a single packet exchange with the stateless UDP protocol, thus eliminating the packet-by-packet load distribution problem seen with TCP.

In addition, the Web caches, i.e., the cache systems 80 each can be implemented to include multiple caches servers connected, for example, in a cluster configuration. There may be multiple servers available to support one customer (origin server) or, alternatively, one or more cache servers available to support multiple customers' content cached at one node (site). In yet another alternative, the cache server clusters can include a switch to select from among the cache servers in a given node/cluster based on a predetermined selection policy, for example, content-aware selection (which enables the clustered servers store different content, and maps requested objects to the appropriate servers), load balancing, and so forth, using known techniques.

Other embodiments are within the scope of the following claims.

What is claimed is:

1. A method of content delivery in a network, the method comprising:

(A) providing a plurality of Domain Name System (DNS) servers associated with a Content Delivery Network (CDN), said plurality of CDN DNS servers sharing a

US 9,021,112 B2

9

common anycast address, wherein each CDN DNS server is associated with a respective plurality of content servers;

(B) causing said plurality of CDN DNS servers to be authoritative for a hostname associated with a content provider by causing said common anycast address to be associated with said hostname;

(C) responsive to a request for content associated with the content provider and issued by a client, said request including at least said hostname,

(c1) causing said hostname to be resolved to said common anycast address by an Internet Service Provider (ISP) DNS server; and then

(c2) by one of said plurality of CDN DNS servers, resolving said hostname to identify an IP address for use by the client to retrieve the content from a content server.

2. The method as recited in claim 1 wherein at least some of said plurality of CDN DNS servers are located at network Points of Presence (POPs).

3. The method as recited in claim 2 wherein each of said plurality of CDN DNS servers is located at a network POP.

4. The method as recited in claim 2 wherein at least one of said plurality of CDN DNS servers and respective plurality of content servers is grouped as a cluster.

5. The method as recited in claim 1 wherein each of said plurality of content servers is associated with a network Point of Presence (POP).

6. The method as recited in claim 1 further comprising:

(D) by the content server, attempting to serve the content.

7. The method as recited in claim 6 wherein said step (D) of attempting comprises:

(d1) if a valid version of the content is available on the content server, serving the content; otherwise

(d2) obtaining the content from a content source and then serving the content.

8. The method as recited in claim 7 wherein, said content source is an origin server associated with said content provider.

9. The method as recited in claim 7 wherein, if a valid version of the content is not available on the content server, the content server uses a table to ascertain where to obtain the content.

10. The method as recited in claim 1, wherein selection of the content server is based on a load balancing policy.

11. The method of claim 1 further comprising:

(c3) by at least one switch, selecting the content server from said respective plurality of content servers associated with said resolving CDN DNS.

12. The method of claim 11, wherein the step of (c3) is performed in accordance with a predetermined selection policy.

13. A method of content delivery for delivering content on behalf of a plurality of content providers, the method comprising:

(A) providing a plurality of Domain Name System (DNS) servers, said plurality of DNS servers sharing a common anycast address, wherein each DNS server is associated with a respective plurality of content servers;

(B) configuring each DNS server and respective plurality of content servers to support content delivery for a plurality of content providers, wherein each of the plurality of content providers has a respective hostname associated therewith;

(C1) responsive to a first request for first content associated with a first content provider of said plurality of content

10

providers, said first request including at least a first hostname associated with the first content provider,

(c11) by a first DNS server of said plurality of DNS servers, resolving said first hostname to identify a first IP address to be used for retrieval of the first content from a first content server of said content servers associated with said first DNS server;

(c12) by a switch associated with said resolving DNS server of step (c11), selecting the first content server from said respective plurality of content servers associated with the DNS server of step (c11); and

(c13) attempting to serve said first content from the first content server; and

(C2) responsive to a second request for said first content associated with said first content provider of said plurality of content providers, said second request including at least the first hostname associated with the first content provider,

(c21) by a second DNS server of said plurality of DNS servers, resolving said first hostname to identify a second IP address for retrieval of the first content from a content server associated with the second DNS server;

(c22) by a switch associated with said resolving DNS server of step (c21), selecting the second content server from said respective plurality of content servers associated with the DNS server of step (c21); and

(c23) attempting to serve said second content from the second content server.

14. The method as recited in claim 13, wherein selection of at least one of the first content server and the second content server is based on a load balancing policy.

15. The method as recited in claim 13, wherein the resolving DNS server of step (c11) is the same as the resolving DNS server performing of step (c21).

16. The method as recited in claim 15, wherein the switch of step (c12) is the same as the switch of step (c22).

17. The method as recited in claim 15, wherein the first content server is the same as the second content server.

18. A content delivery system for delivering content on behalf of a plurality of content providers, each of said plurality of content providers having a hostname associated therewith, the system comprising:

(A) a plurality of Domain Name System (DNS) servers, said plurality of DNS servers sharing a common anycast address, said plurality of DNS servers being authoritative for said hostnames associated with said plurality of content providers;

(B) a set of content servers associated with each DNS server, each set comprising at least a plurality of content servers; and

(C) at least one switch associated with each set of content servers, wherein

(D) responsive to a request to resolve a hostname associated with a particular content provider of said plurality of content providers, said request being associated with an end-user request for content of said particular content provider, one of said plurality of DNS servers is operable to resolve said hostname to identify an IP address for use by the end-user to retrieve the content from a content server,

(E) the at least one switch associated with the set of content servers of the resolving DNS server is operable to perform load balancing to select the content server from the set of content servers associated with the resolving DNS server, and

US 9,021,112 B2

11

- (F) the content server is operable to serve the content to the end-user.
19. A method of content delivery, the method comprising:
- (A) providing a plurality of CDN Domain Name System (DNS) servers, said plurality of DNS servers sharing a common anycast address, each DNS server being associated with a respective plurality of content servers, wherein at least some of said plurality of content servers are located at network Points of Presence (POPs);
- (B) causing said plurality of DNS servers to be authoritative for a hostname associated with a content provider by causing said common anycast address to be associated with said hostname;
- (C) responsive to a request for content associated with the content provider by a client, said request including at least said,
- (c1) by one of said plurality of DNS servers, resolving said hostname to identify an IP address for use by the client to retrieve the content from a content server; and
- (c2) by at least one switch, selecting the content server from said respective plurality of content servers associated with said resolving DNS server,
- (D) by the content server,
- (d1) attempting to serve the content by:
- (d11) if a valid version of the content is available on the content server, serving the content; otherwise
- (d12) obtaining the content from a content source and then serving the content.
20. The method as recited in claim 19, wherein selection of the content server is based on a load balancing policy.
21. A method of content delivery in a network, the method comprising:
- (A) providing a plurality of Domain Name System (DNS) servers associated with a Content Delivery Network (CDN), the plurality of DNS servers sharing a common anycast address, wherein each DNS server is associated with a set of content servers, each set comprising at least a plurality of content servers;
- (B) causing the plurality of DNS servers to be authoritative for a hostname associated with a content provider by causing the common anycast address to be associated with the hostname;
- (C) responsive to a request for content associated with the content provider and issued by a client, the request including at least the hostname,
- (c1) by one of the plurality of DNS servers, resolving the hostname to identify an IP address for use by the client to retrieve the content from a content server; and

12

- (c2) by at least one switch associated with the set of content servers of the resolving DNS server, selecting the content server from the set of content servers associated with the resolving DNS server.
22. A content delivery system comprising a plurality of nodes, wherein each node comprises:
- a Domain Name System (DNS) server sharing an anycast address with DNS servers in other nodes of the content delivery system, wherein the DNS server is authoritative for hostnames associated with a plurality of content providers;
- a plurality of content servers; and
- at least one switch, wherein
- responsive to a request by a client for content associated with a particular content provider of the plurality of content providers, the request including at least a hostname associated with the particular content provider, the DNS server is operable to resolve the hostname to identify an IP address for use by the client to retrieve the content from a content server,
- the at least one switch is operable to select the content server from the plurality of content servers, and
- the content server is operable to serve the content to the client.
23. A method comprising:
- (A) providing a plurality of Domain Name System (DNS) servers, said plurality of DNS servers sharing a common anycast address, wherein each DNS server is associated with a respective plurality of content servers, wherein each DNS server and respective plurality of content servers is configured to support content delivery for a plurality of content providers;
- (B) responsive to a first request for first content associated with a first content provider of said plurality of content providers, said first request including at least a first hostname associated with the first content provider, by a first DNS server of said plurality of DNS servers, resolving said first hostname to identify a first IP address of a content server of the plurality of content servers associated with said first DNS server; and
- (C) responsive to a second request for second content associated with a second content provider of said plurality of content providers, said second request including at least said first hostname associated with the first content provider, by a second DNS server of said plurality of DNS servers, resolving said first hostname to identify a second IP address associated with a content server of the plurality of content servers associated with the second DNS server.

* * * * *

EXHIBIT D



US010924573B2

(12) **United States Patent**
Newton et al.

(10) **Patent No.: US 10,924,573 B2**
(45) **Date of Patent: *Feb. 16, 2021**

(54) **HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)**

(58) **Field of Classification Search**
CPC H04L 67/2814; H04L 67/1014; H04L 67/1097; H04L 67/2842; H04N 21/25891;
(Continued)

(71) Applicant: **Level 3 Communications, LLC**,
Broomfield, CO (US)

(56) **References Cited**

(72) Inventors: **Christopher Newton**, Thousand Oaks, CA (US); **Laurence Lipstone**, Calabasas, CA (US); **David Fullagar**, Boulder, CO (US)

U.S. PATENT DOCUMENTS

4,495,570 A 1/1985 Kitajima et al.
4,591,983 A 5/1986 Bennett et al.
(Continued)

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

FOREIGN PATENT DOCUMENTS

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

AU 763380 6/2003
CA 2202572 10/1998
(Continued)

This patent is subject to a terminal disclaimer.

OTHER PUBLICATIONS

(21) Appl. No.: **16/284,791**

Summons to attend oral proceedings, dated May 8, 2019, Application No. 09727342.9, filed Mar. 21, 2019; 6 pgs.

(22) Filed: **Feb. 25, 2019**

(Continued)

(65) **Prior Publication Data**

US 2019/0320034 A1 Oct. 17, 2019

Primary Examiner — Patrice L Winder

(57) **ABSTRACT**

Related U.S. Application Data

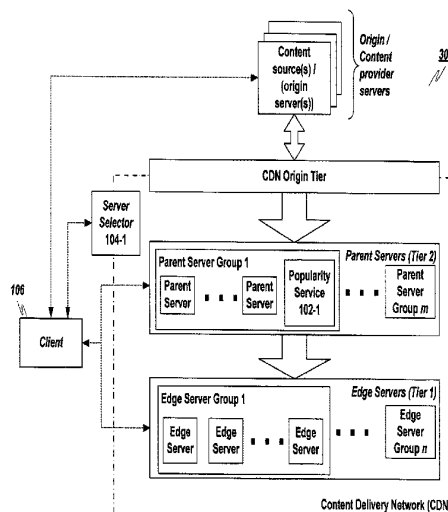
(63) Continuation of application No. 15/645,584, filed on Jul. 10, 2017, now Pat. No. 10,218,806, which is a
(Continued)

A content delivery network has at least a first tier of servers. A content delivery method includes, at a first server in the first tier of servers, obtaining a request from a client for a resource. If the resource is available at the first server or at a peer of the first server, then the resource is served to the client from the first server. Otherwise, it is determined whether the resource is popular, and if the resource is determined to be popular, then the first server obtains the resource and the first server serves the resource to the client. If the resource is determined not to be popular, the client is directed to a second server, not in the first tier of servers, and the second server serves the resource to the client. The second server may be in a second tier of servers or it may be an origin server.

(51) **Int. Cl.**
H04L 29/08 (2006.01)
H04N 21/258 (2011.01)
(Continued)

(52) **U.S. Cl.**
CPC **H04L 67/2814** (2013.01); **H04L 67/1014** (2013.01); **H04L 67/1097** (2013.01);
(Continued)

20 Claims, 7 Drawing Sheets



US 10,924,573 B2

Page 2

Related U.S. Application Data		5,592,611 A	1/1997	Midgely et al.
continuation of application No. 12/880,324, filed on		5,594,910 A	1/1997	Filepp et al.
Sep. 13, 2010, now Pat. No. 9,762,692, which is a		5,603,026 A	2/1997	Demers et al.
continuation-in-part of application No. 12/408,681,		5,614,940 A	3/1997	Cobbley et al.
filed on Mar. 21, 2009, now Pat. No. 8,930,538.		5,619,648 A	4/1997	Canale et al.
		5,623,656 A	4/1997	Lyons
		5,625,781 A	4/1997	Cline et al.
		5,627,829 A	5/1997	Gleeson et al.
(60) Provisional application No. 61/042,412, filed on Apr.		5,630,067 A	5/1997	Kindell et al.
4, 2008.		5,633,999 A	5/1997	Clowes et al.
		5,634,006 A	5/1997	Baughner et al.
(51) Int. Cl.		5,638,443 A	6/1997	Stefik et al.
H04N 7/173 (2011.01)		5,644,714 A	7/1997	Kikinis
H04N 21/231 (2011.01)		5,646,676 A	7/1997	Dewkett et al.
H04N 21/222 (2011.01)		5,649,186 A	7/1997	Ferguson
H04N 21/472 (2011.01)		5,659,729 A	8/1997	Nielsen
		5,666,362 A	9/1997	Chen et al.
(52) U.S. Cl.		5,671,279 A	9/1997	Elgamal
CPC H04L 67/2842 (2013.01); H04N 7/17354		5,675,734 A	10/1997	Hair
(2013.01); H04N 21/222 (2013.01); H04N		5,682,512 A	10/1997	Tetrick
21/23103 (2013.01); H04N 21/23106		5,699,513 A	12/1997	Feigen et al.
(2013.01); H04N 21/25891 (2013.01); H04N		5,712,979 A	1/1998	Grabner et al.
21/472 (2013.01)		5,715,453 A	2/1998	Stewart
		5,721,914 A	2/1998	DeVries
(58) Field of Classification Search		5,734,719 A	3/1998	Tsevdos et al.
CPC H04N 7/17354 ; H04N 21/23106 ; H04N		5,734,831 A	3/1998	Sanders
21/23103 ; H04N 21/222 ; H04N 21/472		5,740,423 A	4/1998	Logan et al.
See application file for complete search history.		5,742,762 A	4/1998	Scholl et al.
		5,751,961 A	5/1998	Smyk
		5,761,507 A	6/1998	Govett
(56) References Cited		5,761,663 A	6/1998	Lagarde et al.
U.S. PATENT DOCUMENTS		5,764,906 A	6/1998	Edelstein et al.
		5,774,660 A	6/1998	Brendel et al.
		5,774,668 A	6/1998	Choquier et al.
		5,777,988 A	7/1998	Cisneros
4,594,704 A	6/1986	5,777,989 A	7/1998	McGarvey
4,726,017 A	2/1988	5,778,187 A	7/1998	Monteiro et al.
4,803,641 A	2/1989	5,781,909 A	7/1998	Logan et al.
4,839,798 A	6/1989	5,784,058 A	7/1998	LaStrange et al.
4,847,784 A	7/1989	5,796,952 A	8/1998	Davis et al.
4,920,432 A	4/1990	5,799,141 A	8/1998	Galipeau et al.
4,922,417 A	5/1990	5,802,106 A	9/1998	Packer
4,943,932 A	7/1990	5,802,291 A	9/1998	Balick et al.
4,949,187 A	8/1990	5,812,769 A	9/1998	Grabner et al.
4,949,248 A	8/1990	5,815,662 A	9/1998	Ong
5,029,232 A	7/1991	5,815,664 A	9/1998	Asano
5,130,792 A	7/1992	5,819,092 A	10/1998	Ferguson et al.
5,132,992 A	7/1992	5,828,847 A	10/1998	Gehr et al.
5,136,716 A	8/1992	5,832,069 A	11/1998	Waters et al.
5,172,413 A	12/1992	5,832,506 A	11/1998	Kuzma
5,191,573 A	3/1993	5,832,514 A	11/1998	Norin et al.
5,253,275 A	10/1993	5,835,718 A	11/1998	Blewett
5,253,341 A	10/1993	5,845,090 A	12/1998	Collins et al.
5,287,499 A	2/1994	5,845,303 A	12/1998	Templeman
5,287,537 A	2/1994	5,856,974 A	1/1999	Gervais et al.
5,291,554 A	3/1994	5,862,339 A	1/1999	Bonnaure et al.
5,341,477 A	8/1994	5,867,706 A	2/1999	Martin et al.
5,371,532 A	12/1994	5,867,799 A	2/1999	Lang et al.
5,410,343 A	4/1995	5,870,546 A	2/1999	Kirsch
5,414,455 A	5/1995	5,870,559 A	2/1999	Leshem et al.
5,442,389 A	8/1995	5,878,212 A	3/1999	Civanlar et al.
5,442,390 A	8/1995	5,884,038 A	3/1999	Kapoor
5,442,749 A	8/1995	5,890,171 A	3/1999	Blumer et al.
5,471,622 A	11/1995	5,893,116 A	4/1999	Simmonds et al.
5,475,615 A	12/1995	5,894,554 A	4/1999	Lowery et al.
5,508,732 A	4/1996	5,896,533 A	4/1999	Ramos et al.
5,515,511 A	5/1996	5,898,456 A	4/1999	Wahl
5,519,435 A	5/1996	5,903,723 A	5/1999	Beck et al.
5,528,281 A	6/1996	5,907,704 A	5/1999	Gudmundson et al.
5,539,621 A	7/1996	5,913,028 A	6/1999	Wang et al.
5,542,087 A	7/1996	5,913,033 A	6/1999	Grout
5,544,313 A	8/1996	5,918,010 A	6/1999	Appleman et al.
5,544,327 A	8/1996	5,919,247 A	7/1999	Van Hoff et al.
5,550,577 A	8/1996	5,920,701 A	7/1999	Miller et al.
5,550,863 A	8/1996	5,931,904 A	8/1999	Banga
5,550,982 A	8/1996	5,933,832 A	8/1999	Suzuoka et al.
5,557,317 A	9/1996	5,933,835 A	8/1999	Adams et al.
5,568,181 A	10/1996	5,935,207 A	8/1999	Logue et al.
5,572,643 A	11/1996	5,940,831 A	8/1999	Takano
5,590,288 A	12/1996			

US 10,924,573 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

5,944,780	A	8/1999	Chase	6,173,311	B1	1/2001	Hassett et al.
5,951,694	A	9/1999	Choquier et al.	6,173,322	B1	1/2001	Hu
5,956,489	A	9/1999	San Andres et al.	6,178,160	B1	1/2001	Bolton et al.
5,956,716	A	9/1999	Kenner et al.	6,181,867	B1	1/2001	Kenner et al.
5,958,008	A	9/1999	Pogrebisky et al.	6,185,598	B1	2/2001	Farber et al.
5,961,596	A	10/1999	Takubo et al.	6,185,619	B1	2/2001	Joffe et al.
5,966,440	A	10/1999	Hair	6,189,030	B1	2/2001	Kirsch et al.
5,968,121	A	10/1999	Logan et al.	6,189,039	B1	2/2001	Harvey
5,973,696	A	10/1999	Agranat et al.	6,195,680	B1	2/2001	Goldszmidt et al.
5,978,791	A	11/1999	Farber et al.	6,205,120	B1	3/2001	Packer et al.
5,983,214	A	11/1999	Lang et al.	6,212,565	B1	4/2001	Gupta
5,983,227	A	11/1999	Nazem et al.	6,226,618	B1	5/2001	Downs
5,987,606	A	11/1999	Cirasole et al.	6,226,642	B1	5/2001	Beranek et al.
5,991,809	A	11/1999	Kriegsman et al.	6,230,196	B1	5/2001	Guenthner et al.
5,996,025	A	11/1999	Day	6,233,623	B1	5/2001	Jeffords et al.
6,002,720	A	12/1999	Yurt et al.	6,240,462	B1	5/2001	Agraharam et al.
6,003,030	A	12/1999	Kenner et al.	6,249,810	B1	6/2001	Kiraly
6,012,090	A	1/2000	Chung et al.	6,256,675	B1	7/2001	Rabinovich
6,014,686	A	1/2000	Elnozahy et al.	6,263,313	B1	7/2001	Milsted
6,014,698	A	1/2000	Griffiths	6,266,335	B1	7/2001	Bhaskaran
6,016,509	A	1/2000	Dedrick	6,266,699	B1	7/2001	Sevcik
6,016,512	A	1/2000	Huitema	6,269,394	B1	7/2001	Kenner et al.
6,018,516	A	1/2000	Packer	6,272,566	B1	8/2001	Craft
6,021,426	A	2/2000	Douglis	6,275,470	B1	8/2001	Ricciulli
6,023,470	A	2/2000	Lee et al.	6,282,569	B1	8/2001	Wallis et al.
6,029,175	A	2/2000	Chow et al.	6,286,045	B1	9/2001	Griffith et al.
6,029,176	A	2/2000	Cannon	6,298,041	B1	10/2001	Packer
6,035,332	A	3/2000	Ingrassia et al.	6,314,565	B1	11/2001	Kenner et al.
6,038,216	A	3/2000	Packer	6,317,787	B1	11/2001	Boyd et al.
6,038,310	A	3/2000	Hollywood et al.	6,324,580	B1	11/2001	Jindal et al.
6,041,324	A	3/2000	Earl et al.	6,324,582	B1	11/2001	Sridhar et al.
6,044,405	A	3/2000	Driscoll et al.	6,330,602	B1	12/2001	Law et al.
6,046,980	A	4/2000	Packer	6,332,157	B1	12/2001	Mighdoll et al.
6,049,831	A	4/2000	Gardell et al.	6,338,044	B1	1/2002	Cook et al.
6,052,718	A	4/2000	Gifford	6,343,298	B1	1/2002	Savchenko et al.
6,052,730	A	4/2000	Felciano et al.	6,345,294	B1	2/2002	O'Toole et al.
6,065,051	A	5/2000	Steele et al.	6,347,085	B2	2/2002	Kelly et al.
6,065,062	A	5/2000	Periasamy et al.	6,351,775	B1	2/2002	Yu
6,070,191	A	5/2000	Narendran et al.	6,351,776	B1	2/2002	O'Brien et al.
6,078,943	A	6/2000	Yu	6,360,256	B1	3/2002	Lim
6,081,829	A	6/2000	Sidana	6,370,571	B1	4/2002	Medin
6,081,835	A	6/2000	Antcliff	6,370,580	B2	4/2002	Kriegsman
6,081,840	A	6/2000	Zhao	6,389,462	B1	5/2002	Cohen et al.
6,085,193	A	7/2000	Malkin et al.	6,398,245	B1	6/2002	Gruse
6,092,112	A	7/2000	Fukushige	6,405,252	B1	6/2002	Gupta et al.
6,092,204	A	7/2000	Baker et al.	6,405,257	B1	6/2002	Gersht et al.
6,094,680	A	7/2000	Hokanson	6,412,000	B1	6/2002	Riddle et al.
6,094,706	A	7/2000	Factor et al.	6,415,280	B1	7/2002	Farber et al.
6,098,078	A	8/2000	Gehani	6,415,368	B1	7/2002	Glance et al.
6,098,096	A	8/2000	Tsirigotis et al.	6,418,421	B1	7/2002	Hurtado
6,105,028	A	8/2000	Sullivan et al.	6,421,714	B1	7/2002	Rai et al.
6,108,673	A	8/2000	Brandt et al.	6,421,726	B1	7/2002	Kenner et al.
6,108,703	A	8/2000	Leighton et al.	6,427,170	B1	7/2002	Sitaraman et al.
6,112,231	A	8/2000	DeSimone et al.	6,430,618	B1	8/2002	Karger et al.
6,112,239	A	8/2000	Kenner et al.	6,442,588	B1	8/2002	Clark et al.
6,112,240	A	8/2000	Pogue et al.	6,453,319	B1	9/2002	Mattis et al.
6,115,357	A	9/2000	Packer et al.	6,456,630	B1	9/2002	Packer et al.
6,115,752	A	9/2000	Chauhan	6,460,082	B1	10/2002	Lumelsky
6,119,143	A	9/2000	Dias et al.	6,460,085	B1	10/2002	Toporek et al.
6,125,388	A	9/2000	Reisman	6,463,454	B1	10/2002	Lumelsky et al.
6,125,394	A	9/2000	Rabinovich	6,463,508	B1	10/2002	Wolf
6,128,601	A	10/2000	Van Horne et al.	6,466,949	B2	10/2002	Yang et al.
6,128,623	A	10/2000	Mattis et al.	6,470,389	B1	10/2002	Chung et al.
6,128,660	A	10/2000	Grimm et al.	6,473,405	B2	10/2002	Ricciulli
6,130,890	A	10/2000	Leinwand et al.	6,480,893	B2	11/2002	Kriegsman
6,131,095	A	10/2000	Low et al.	6,484,143	B1	11/2002	Swildens et al.
6,134,583	A	10/2000	Herriot	6,484,204	B1	11/2002	Rabinovich
6,144,375	A	11/2000	Jain et al.	6,487,555	B1	11/2002	Bharat et al.
6,144,702	A	11/2000	Yurt et al.	6,490,580	B1	12/2002	Dey et al.
6,151,624	A	11/2000	Teare et al.	6,493,707	B1	12/2002	Dey et al.
6,154,744	A	11/2000	Kenner et al.	6,496,856	B1	12/2002	Kenner et al.
6,154,753	A	11/2000	McFarland	6,502,125	B1	12/2002	Kenner et al.
6,157,648	A	12/2000	Volt et al.	6,502,205	B1	12/2002	Yanai et al.
6,161,137	A	12/2000	Johnson et al.	6,502,215	B2	12/2002	Raad et al.
6,163,779	A	12/2000	Mantha et al.	6,505,248	B1	1/2003	Casper et al.
				6,529,477	B1	3/2003	Toporek et al.
				6,553,413	B1	4/2003	Leighton et al.
				6,553,420	B1	4/2003	Karger et al.
				6,557,054	B2	4/2003	Reisman

US 10,924,573 B2

Page 4

(56)	References Cited		7,117,259 B1	10/2006	Rohwer	
	U.S. PATENT DOCUMENTS		7,133,905 B2 *	11/2006	Dilley	H04L 29/06 709/219
6,564,251 B2	5/2003	Katariya et al.	7,185,052 B2	2/2007	Day	
6,577,595 B1	6/2003	Counterman	7,188,085 B2	3/2007	Pelletier	
6,581,090 B1	6/2003	Lindbo et al.	7,206,748 B1	4/2007	Gruse	
6,584,083 B1	6/2003	Toporek et al.	7,219,153 B1	5/2007	Day	
6,587,837 B1	7/2003	Spagna	7,254,645 B2	8/2007	Nishi	
6,591,288 B1	7/2003	Edwards et al.	7,274,658 B2 *	9/2007	Bornstein	H04L 45/00 370/227
6,591,299 B2	7/2003	Riddle et al.	7,359,955 B2	4/2008	Menon et al.	
6,594,260 B1	7/2003	Aviani, Jr. et al.	7,373,644 B2	5/2008	Aborn	
6,598,121 B2	7/2003	Challenger et al.	7,552,233 B2 *	6/2009	Raju	G06F 12/1483 709/238
6,611,862 B2	8/2003	Reisman	7,562,153 B2	7/2009	Biliris et al.	
6,625,643 B1	9/2003	Colby et al.	7,577,754 B2	8/2009	Garcia-Luna-Aceves et al.	
6,651,141 B2	11/2003	Adrangi	8,332,484 B2 *	12/2012	Afergan	H04L 67/14 709/218
6,654,344 B1	11/2003	Toporek et al.	8,417,816 B2 *	4/2013	Esteban	H04L 67/2842 709/226
6,654,807 B2	11/2003	Farber et al.	8,930,538 B2	1/2015	Fullagar et al.	
6,658,464 B2	12/2003	Reisman	8,949,533 B2 *	2/2015	Puthalath	H04L 67/1095 711/119
6,665,706 B2	12/2003	Kenner et al.	10,462,252 B2 *	10/2019	Newton	H04L 67/2842
6,678,659 B1	1/2004	Van Kommer	2001/0027491 A1	10/2001	Terretta	
6,691,148 B1	2/2004	Zinky et al.	2001/0029525 A1	10/2001	Lahr	
6,694,358 B1	2/2004	Swildens et al.	2001/0049732 A1	12/2001	Raciborski et al.	
6,699,418 B2	3/2004	Okada et al.	2001/0051980 A1	12/2001	Raciborski et al.	
6,708,137 B2	3/2004	Carley	2001/0056500 A1	12/2001	Farber et al.	
6,718,328 B1	4/2004	Norris	2002/0010798 A1	1/2002	Ben-Shaul et al.	
6,732,237 B1	5/2004	Jacobs et al.	2002/0016831 A1	2/2002	Peled et al.	
6,741,563 B2	5/2004	Packer	2002/0016835 A1	2/2002	Gammerman	
6,741,990 B2	5/2004	Nair et al.	2002/0018449 A1	2/2002	Ricciulli	
6,751,673 B2	6/2004	Shaw	2002/0023164 A1	2/2002	Lahr	
6,754,699 B2	6/2004	Swildens et al.	2002/0023165 A1	2/2002	Lahr	
6,754,706 B1	6/2004	Swildens et al.	2002/0040404 A1	4/2002	Lahr	
6,763,377 B1	7/2004	Belknap	2002/0042817 A1	4/2002	Lahr et al.	
6,763,388 B1	7/2004	Tsimelzon	2002/0046273 A1	4/2002	Lahr	
6,778,502 B2	8/2004	Ricciulli	2002/0046405 A1	4/2002	Lahr	
6,785,704 B1	8/2004	McCanne	2002/0049857 A1	4/2002	Farber et al.	
6,799,221 B1	9/2004	Kenner et al.	2002/0059592 A1	5/2002	Kiraly	
6,801,576 B1	10/2004	Haldeman et al.	2002/0066038 A1	5/2002	Mattsson	
6,834,306 B1	12/2004	Tsimelzon	2002/0073199 A1	6/2002	Levine et al.	
6,839,758 B2	1/2005	Sorensen	2002/0078233 A1	6/2002	Biliris et al.	
6,842,604 B1	1/2005	Cook et al.	2002/0082999 A1	6/2002	Lee et al.	
6,859,791 B1	2/2005	Spagna	2002/0083124 A1	6/2002	Knox et al.	
6,859,840 B2	2/2005	Singal et al.	2002/0087797 A1	7/2002	Adrangi	
6,870,851 B1	3/2005	Leinwand et al.	2002/0089470 A1	7/2002	Raman et al.	
6,874,032 B2	3/2005	Gersht et al.	2002/0092026 A1	7/2002	Janniello et al.	
6,901,604 B1	5/2005	Kiraly	2002/0099850 A1	7/2002	Farber et al.	
6,915,307 B1	7/2005	Mattis et al.	2002/0116444 A1	8/2002	Chaudhri et al.	
6,915,329 B2	7/2005	Kriegsman	2002/0124080 A1	9/2002	Leighton	
6,928,442 B2	8/2005	Farber et al.	2002/0129134 A1	9/2002	Leighton et al.	
6,934,255 B1	8/2005	Toporek et al.	2002/0131645 A1	9/2002	Hamilton	
6,950,623 B2	9/2005	Brown et al.	2002/0133537 A1	9/2002	Lau	
6,963,910 B1	11/2005	Belknap et al.	2002/0143798 A1	10/2002	Lisiecki et al.	
6,963,980 B1	11/2005	Mattsson	2002/0143888 A1	10/2002	Lisiecki et al.	
6,963,981 B1	11/2005	Bailey et al.	2002/0147774 A1	10/2002	Lisiecki et al.	
6,970,432 B1	11/2005	Hankins et al.	2002/0152318 A1	10/2002	Menon et al.	
6,973,490 B1	12/2005	Robertson et al.	2002/0163882 A1	11/2002	Bornstein et al.	
6,976,090 B2	12/2005	Ben-Shaul et al.	2002/0174168 A1	11/2002	Beukema et al.	
6,981,050 B1	12/2005	Tobias et al.	2002/0184357 A1	12/2002	Traversat et al.	
6,981,180 B1	12/2005	Bailey et al.	2002/0198953 A1	12/2002	O'Rourke et al.	
6,996,616 B1	2/2006	Leighton et al.	2002/0199016 A1	12/2002	Freedman	
6,999,988 B2	2/2006	Buddhikot et al.	2003/0009444 A1	1/2003	Eidler et al.	
7,003,572 B1	2/2006	Lownsbrough et al.	2003/0018966 A1	1/2003	Cook et al.	
7,007,089 B2	2/2006	Freedman	2003/0028623 A1	2/2003	Hennessey et al.	
7,010,578 B1	3/2006	Lewin et al.	2003/0028626 A1	2/2003	Hennessey et al.	
7,012,900 B1	3/2006	Riddle	2003/0028777 A1	2/2003	Sim	
7,013,342 B2	3/2006	Riddle	2003/0031176 A1	2/2003	Fuller et al.	
7,024,466 B2 *	4/2006	Outten	2003/0055972 A1	3/2003	Riddle	
			2003/0061263 A1	3/2003	Bulson et al.	
			2003/0061280 A1	3/2003	Aborn	
			2003/0065703 A1	4/2003	Stolorz et al.	
			2003/0065762 A1	4/2003	Lee et al.	
			2003/0078888 A1	4/2003	Lee et al.	
			2003/0078889 A1	4/2003	Slocombe et al.	
			2003/0079027 A1	4/2003	Lee et al.	
			2003/0095660 A1	5/2003		
7,032,072 B1	4/2006	Quinn et al.				
7,039,633 B1	5/2006	Dey et al.				
7,047,300 B1	5/2006	Oehrke et al.				
7,054,902 B2	5/2006	Toporek et al.				
7,054,935 B2	5/2006	Farber et al.				
7,058,706 B1	6/2006	Iyer et al.				
7,069,177 B2	6/2006	Carley				
7,096,266 B2	8/2006	Lewin et al.				
7,103,564 B1	9/2006	Ehnebuske				
7,103,645 B2	9/2006	Leighton et al.				
7,110,984 B1	9/2006	Spagna				

US 10,924,573 B2

Page 5

(56)

References Cited

U.S. PATENT DOCUMENTS

2003/0105604	A1	6/2003	Ash et al.	
2003/0112792	A1	6/2003	Cranor et al.	
2003/0140111	A1	7/2003	Pace et al.	
2003/0149581	A1	8/2003	Chaudhri et al.	
2003/0154239	A1	8/2003	Davis et al.	
2003/0158923	A1	8/2003	Burkhart	
2003/0158928	A1	8/2003	Knox et al.	
2004/0010588	A1	1/2004	Slater et al.	
2004/0022194	A1	2/2004	Ricciulli	
2004/0064832	A1	4/2004	Tsukidate et al.	
2004/0139097	A1	7/2004	Farber et al.	
2004/0162871	A1	8/2004	Pabla et al.	
2004/0177148	A1	9/2004	Tsimelzon, Jr.	
2004/0205162	A1	10/2004	Parikh	
2004/0221019	A1*	11/2004	Swildens	H04L 67/1008 709/217
2005/0010653	A1	1/2005	McCanne	
2005/0021863	A1	1/2005	Jungck	
2005/0033858	A1	2/2005	Swildens et al.	
2005/0038851	A1	2/2005	Kriegsman	
2005/0060493	A1	3/2005	Krissell et al.	
2005/0100027	A1	5/2005	Leinwand et al.	
2005/0114296	A1	5/2005	Farber et al.	
2005/0240966	A1	10/2005	Hindle et al.	
2005/0262104	A1	11/2005	Robertson et al.	
2006/0015574	A1	1/2006	Seed et al.	
2006/0112176	A1	5/2006	Liu et al.	
2006/0143293	A1	6/2006	Freedman	
2007/0055764	A1	3/2007	Dilley et al.	
2008/0065724	A1	3/2008	Seed et al.	
2008/0071859	A1	3/2008	Seed et al.	
2008/0147866	A1	6/2008	Stolorz et al.	
2008/0215718	A1	9/2008	Stolorz et al.	
2009/0254661	A1	10/2009	Fullagar et al.	
2010/0169772	A1	7/2010	Stallings et al.	
2010/0257258	A1	10/2010	Liu et al.	
2010/0312861	A1	12/2010	Kolhi et al.	
2010/0332595	A1	12/2010	Fullagar et al.	
2011/0099290	A1	4/2011	Swildens et al.	
2011/0145386	A1	6/2011	Stolorz et al.	
2011/0320592	A1*	12/2011	Kemmerer, Jr.	H04L 67/2852 709/224
2017/0339241	A1	11/2017	Newton	

FOREIGN PATENT DOCUMENTS

CA	2288488	6/2000
CA	2335661	3/2001
CA	2335662	3/2001
CA	2467998	4/2006
CN	ZL99810853.7	8/1999
CN	1264476 A	8/2000
EP	0649121	4/1995
EP	0651554	5/1995
EP	0801487 A2	10/1997
EP	0817444 A2	1/1998
EP	0824236 A2	2/1998
EP	0865180 A2	9/1998
EP	1063831 A2	12/2000
EP	1104555	6/2001
GB	2281793 A1	3/1995
GB	2353877	4/2004
IL	140935	3/2006
JP	05162529	6/1993
JP	07066829	3/1995
JP	08328583	12/1996
JP	10027148 A2	1/1998
JP	10093552	4/1998
JP	2000207270	7/2000
JP	2001007844	1/2001
JP	2001290787	10/2001
JP	2001312434	11/2001
JP	2002522995	7/2002
JP	3566626	6/2004

JP	2005124165	5/2005
JP	3762649	1/2006
KR	20010023599	3/2001
TW	NI176482	8/2003
WO	WO-1996042041 A2	12/1996
WO	WO-1997011429 A1	3/1997
WO	WO-1997029423 A1	8/1997
WO	WO-199742582	11/1997
WO	WO-1998004985 A1	2/1998
WO	WO-1998006033 A1	2/1998
WO	WO-199859486	12/1998
WO	WO-199917227	4/1999
WO	WO-199948246	9/1999
WO	WO-199953422	10/1999
WO	WO-2000029990 A1	5/2000
WO	WO-2000052594	9/2000
WO	WO-2000060861	10/2000
WO	WO-200193533	12/2001
WO	WO-2003088065	10/2003
WO	WO-2009108593	9/2009
WO	WO-2009123868 A3	10/2009
WO	WO-2012037039 A1	3/2012

OTHER PUBLICATIONS

Canada Examination Report, dated Aug. 16, 2013, Application No. 2,720,353, filed Mar. 21, 2009; 3 pgs.

Canadian Examination Report, dated Jun. 13, 2017, Application No. 2,811,210, filed Sep. 12, 2011; 3 pgs.

Canadian Examination Report, dated Sep. 8, 2014, Application No. 2,720,353, filed Mar. 21, 2009; 4 pgs.

Chinese Examination Report, dated Jun. 13, 2012, Application No. 200980120043.0, 6 pgs.

European Examination Report, dated Mar. 30, 2016, Application No. 09727342.9, filed Mar. 21, 2009; 4 pgs.

Exporting Web Server Final Report, http://www.cs.technion.ac.il/Labs/Lccn/projects/spring97/project4/final_report.html (downloaded Jul. 7, 2007).

Extended European Search Report dated Dec. 6, 2011, App. No. 10178695.2, 7 pgs.

Extended European Search Report, dated Nov. 9, 2015, Application No. 11825742.7, filed Sep. 12, 2011; 6 pgs.

IBM Technical Disclosure Bulletin; Local Area Network Server Replacement Procedure, vol. 38, No. 1, 235-236.

International Preliminary Report on Patentability, dated Mar. 19, 2013, Int'l Appl. No. PCT/US11/051252, Int'l Filing Date Sep. 12, 2011, 6 pgs.

International Search Report, dated Jan. 23, 2012, Intl Appl. No. PCT/US11/51252, Intl Filing Date Sep. 12, 2011, 3 pgs.

JP 10(1998)-27148 machine translation prepared by Japanese Patent Office (JPO), published Jan. 27, 1998, Title: "Server System for Internet," Applicant: Hitachi, 23 pgs.

Patent Abstracts of Japan "Server System for Internet", Pub. No. 10-027148, pub. date Jan. 27, 1998, Applicant: Hitachi, computer translation, 12 pgs.

Patent Abstracts of Japan, "Electronic Mail Multiplexing System and Communication Control Method in the System," (Appln. No. JP19930162529), (Jun. 30, 1993) (Pub. No. JP 7066829).

Patent Abstracts of Japan, "Method and Device for Repeating and Converting Information", (Appln. No. JP19960328583) (Pub. No. JP10171727) Jun. 26, 1998.

Supplemental European Search Report, dated Nov. 5, 2012, Application No. 09727342.9, 7 pgs.

Written Opinion, dated Jan. 23, 2012, Intl Appl No. PCT/US11/51252, Intl Filing Date Sep. 12, 2011, 5 pgs.

Adler, R. M. "Distributed Coordination Models for Client/Server Computing", Computer 28, 4 Apr. 1, 1995, 14-22.

Aggarwal, A. et al., "Performance of Dynamic Replication Schemes for an Internet Hosting Service", Technical Report, AT&T Labs Oct. 1, 1998.

Almeroth, K. et al., "Scalable Delivery of Web Pages Using Cyclic Best-Effort (UDP) Multicast", *IEEE INFOCOM* San Francisco, California, USA Jun. 1, 1998.

US 10,924,573 B2

Page 6

(56)

References Cited

OTHER PUBLICATIONS

- Andresen, D. et al., "Multiprocessor scheduling with client resources to improve the response time of WWW applications", *ACM Press*, NY, Proc. 11th Int'l Conf. on Supercomputing (Austria, ICS '97) Jul. 1, 1997, 92-99.
- Andresen, et al., "SWEB: Towards a Scalable World Wide Web Server on Multicomputers", *Proc. IPPS* Apr. 15, 1996, 850-856.
- Awerbuch, M. et al., "Competitive Distributed File Allocation", in *Proc. of the 25th Ann. ACM Symp. on Theory of Computing* May 1, 1993, pp. 164-173.
- Awerbuch, B. et al., "Distributed Paging for General Networks", In *Proc. of the 7th ACM-SIAM Symposium on Discrete Algorithms*, Jan. 1, 1996, pp. 574-583.
- Baentsch, M. et al., "Enhancing the Web's Infrastructure: From Caching to Replication", *IEEE Internet Computing*, 1(2):18-27.
- Baentsch, M. et al., "Introducing Application-Level Replication and Naming into Today's Web", *Computer Networks and ISDN Systems* vol. 28, Nos. 7-11 May 1, 1996, pp. 921-930.
- Bartal, Y. et al., "Competitive Algorithms for Distributed Data Management", 24th Annual ACM STOC, May 1992, Victoria, B.C. Canada.
- Basturk, E. et al., "Using network layer anycast for load distribution in the Internet", *Tech. Rep.*, IBM T.J. Watson Research Center Jul. 1, 1997, 21 pgs.
- Berners-Lee, T. et al., RFC 1738—Uniform Resource Locators Dec. 1, 1994, pp. 1-25.
- Berners-Lee, T. et al., "Hypertext Markup Language—2.0", Internet Engineering Task Force (IETF) Request for Comments (RFC) 1866 Nov. 1, 1995, pp. 1-77.
- Berners-Lee, T. et al., "Hypertext Transfer Protocol—HTTP/1.0", Internet Engineering Task Force (IETF) Request for Comments (RFC) 1945 May 1, 1996, pp. 1-60.
- Berners-Lee, T. et al., "Hypertext Transfer Protocol—HTTP/1.0", Internet Engineering Task Force (IETF), Internet Draft, draft-ietf-http-v1.0-spec-00.ps, Mar. 1, 1995, pp. 1-59.
- Berra, P. B. et al., "Architecture for Distributed Database Systems", *Computer Communications* vol. 13, No. 4 May 1, 1990, pp. 217-231.
- Bestavros, A. Demand-Based Document Dissemination to Reduce Traffic and Balance Load in Distributed Information Systems, In *Proc. IEEE Symp. on Parallel and Distributed Processing*, San Antonio, TX, 8 pgs.
- Bestavros, et al., "Server-Initiated Document Dissemination for the WWW", *IEEE Data Engineering Bulletin*, 19(3): 3-11.
- Bestavros, A. Speculative Data Dissemination and Service to Reduce Server Load Network Traffic and Service Time in Distributed Information Systems, In *Proc. ICDE '96: The 1996 Int'l Conf. on Data Engineering*, 4 Pages.
- Bhattacharjee, S. et al., "Application-Layer Anycasting", in *Proc. IEEE INFOCOM '97* Apr. 1, 1997, 9 pgs.
- Braun, H. et al., "Web traffic characterization: an assessment of the impact of caching documents from NCSA's web server", *Comput. Netw. ISDN Syst.* 28, 1-2, 37-51.
- Brisco, T. DNS Support for Load Balancing, IETF RFC 1794 Apr. 1, 1995, pp. 1-7.
- Carter, et al., "Dynamic server selection using bandwidth probing in wide-area networks", *Tech. Rep.* BU-CS-96-007, Compo Sci. Dept., Boston University, 1-20.
- Carter, et al., "Server selection using dynamic path characterization in Wide-Area Networks", *IEEE INFOCOM '97* Jan. 1, 1997, pp. 1014-1021.
- Carter, J. L. et al., "Universal Classes of Hash Functions", *Journal of Computer and System Sciences* vol. 18, No. 2 Apr. 1, 1979, 106-112.
- Cate, V. Alex: a global file system, in *Proc. Usenix Conf. on File Systems* May 1, 1992, pp. 1-11.
- Chankhunthod, A. et al., "A Hierarchical Internet Object Cache", *Proc. of the 1996 USENIX Technical Conf.*, 153-163.
- Cohen, J. et al., "Cache Array Routing Protocol v1.1", <http://tools.ietf.org/id/draft-vinod-carp-v1-01.txt> (Last-Modified: Wed, Oct. 1, 1997), 8 pages.
- Colajanni, et al., "Adaptive TTL schemes for load balancing of distributed Web servers", *SIGMETRICS Perform. Eval. Rev.* 25,2, 36-42.
- Colajanni, M. et al., "Scheduling Algorithms for Distributed Web Servers", *Intl Conf. on Distributed Computing Systems (ICDCS)*, Baltimore, Maryland, USA May 1, 1997, pp. 169-176.
- Crovella, et al., "Dynamic server selection in the Internet", 3rd IEEE Workshop on the Arch. and Implementation of High Performance Computer Sys. '95, pp. 158-162.
- Danzig, P. B. et al., "An analysis of wide-area name server traffic: a study of the Internet Domain Name System", *Conf. Proc. Communications Architectures & Protocols*; D. Oran, Ed. SIGCOMM '92; ACM Press, New York, NY, 281-292.
- De Bra, P.M.E. et al., "Information Retrieval in the World Wide Web: Making Client-Based Searching Feasible", *Computer Networks and ISDN System*, NL, North Holland Publishing, Amsterdam, vol. 27, No. 2, ISSN: 0169-7552 Nov. 1, 1994, 183-192.
- Deering, S. E. et al., "Multicast routing in datagram internetworks and extended LANs", *ACM Trans. Comput. Syst.* 8,2, 85-110.
- Doi, K. Super Proxy Script—How to make distributed proxy servers by URL hashing, Sharp Corp., <http://naragw.sharp.co.jp/sps/>; download Jul. 7, 2007.
- Eriksson, H. Mbone: The Multicast Backbone, *Communications of the ACM* vol. 37, No. 8 Aug. 1, 1994, pp. 54-60.
- Feeley, M. et al., "Implementing Global Memory Management in a Workstation Cluster", In *Proc. 15th ACM Symp. on Operating Systems Principles*, 201-212.
- Fielding, R. et al., "Hypertext Transfer Protocol—HTTP/1.1", Internet Engineering Task Force (IETF) Internet Draft, draft-ietf-http-v1.1-spec-00.txt Nov. 1, 1995, pp. 1-81.
- Fielding, R. et al., "Hypertext Transfer Protocol—HTTP/1.1", Internet Engineering Task Force (IETF) Request for Comments (RFC) 2068 Jan. 1, 1997, pp. 1-162.
- Floyd, S. et al., "A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing", In *Proc. of ACM SIGCOMM '95*, Aug. 1, 1995, 342-356.
- Fox, A. et al., "A Framework for Separating Server Scalability and Availability from Internet Application Functionality", PhD thesis, University of California, Berkeley, Jan. 1, 1998, 163 pgs.
- Fox, A. et al., "Cluster-based scalable network services", *Proc. 16th ACM Symp. on Operating Systems Principles (Saint Malo, France, Oct. 5-8, 1997)*, W. M. Waite, Ed. SOSP '97. ACM Press, New York, NY, 78-91.
- Gadde, S. et al., "Reduce, reuse, recycle: An approach to building large internet caches", in *Workshop on Hot Topics in Operating Systems (HotOS)* Apr. 1, 1997, pp. 93-98.
- Goldszmidt, M. et al., "Load Distribution for Scalable Web Servers: Summer Olympics 1996—A Case Study", In *Proc. 8th IFIP/IEEE Int'l Workshop on Distributed Systems: Operations and Management*, Sydney, Australia, 10 pgs.
- Grigni, M. et al., "Tight Bounds on Minimum Broadcasts Networks", *SIAM J. Disc. Math.* 4 (May 1991), 207-222.
- Guyton, et al., "Locating nearby copies of replicated Internet servers", *Proc. ACM SIGCOMM '95*; pp. 288-298 Oct. 1, 1995.
- Gwertzman, J. et al., "The Case for Geographical Push-Caching", *Proc. Workshop on Hot OS '95*, 51-55.
- Gwertzman, J. et al., "World-Wide Web Cache Consistency", *Proc. 1996 USENIX Tech. Conf.*, pp. 141-151, San Diego, CA.
- Jeffery, C. et al., "Proxy sharing proxy servers", In *Proc. IEEE etaCOM Conf.* May 1, 1996, pp. 116-119.
- Karger, D. et al., "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web", In *Proc. 29th Annual ACM Symp. on Theory of Computing* May 1, 1997, 654-663.
- Kim, Y. J. et al., "Clustered multi-media NOD: Popularity-based article prefetching and placement", *Sixteenth IEEE Symp. on Mass Storage Systems*, San Diego, CA, pp. 194-202.
- Korkea-Aho, M. Scalability in Distributed Multimedia Systems, Technical report TKO-B128, Helsinki University of Technology.

US 10,924,573 B2

Page 7

(56)

References Cited

OTHER PUBLICATIONS

- Kostadinova, R. Peer-to-Peer Video Streaming, [online; retrieved on Jan. 25, 2013]; Retrieved from the Internet <URL: http://www.ee.kth.se/php/modules/publications/reports/2008/XR-EE-LCN_2008_004.pdf>, especially section 5.4.1 Jan. 1, 2008, 1-53.
- Krishnamurthy, B. et al., "Study of piggyback cache validation for proxy caches in the World Wide Web", in: Symp. on Internet Technology and Systems, USENIX Association.
- Kroeger, T. M. et al., "Exploring the Bounds of Web Latency Reduction from Caching and Prefetching", Proc. Usenix Symp. Internet Technologies and Systems, USENIX Dec. 1, 1997, pp. 13-22.
- Kwan, et al., "NCSA's World Wide Web Server: Design and Performance", IEEE , pp. 68-74.
- Little, T.D. C. et al., "Selection and Dissemination of Digital Video via the Virtual Video Browser", *Multimedia Tools and Applications* vol. 1, No. 2 (Netherlands) , pp. 149-172.
- Luotonen, et al., "World-Wide Web Proxies", CERN modified May 24, 1994 , 1-8.
- Malkin, G. RIP Version 2 Carrying Additional Information, Internet Engineering Task Force (IETF) Request for Comments (RFC) 1388 Jan. 1, 1993, pp. 1-7.
- Malpani, R. et al., "Making World Wide Web Caching Servers Cooperate", In Proc. 4th Int'l. World Wide Web Conf. (downloaded from <http://www.w3.org/Conferences/WWW4/Papers/59/> on Jul. 7, 2007), 10 pages.
- Mockapetris, P. Domain Names—Concepts and Facilities, IETF RFC 1034 Nov. 1, 1987, 55 pgs.
- Mockapetris, P. Domain Names—Implementation and Specification, IETF RFC 1035 Nov. 1, 1987, 55 pgs.
- Mourad, et al., "Scalable Web Server Architectures", iscc, 2nd IEEE Symposium on Computers and Communications (ISCC '97) Jul. 1, 1997, pp. 12-16.
- Moy, J. OSPF Version 2, Internet Engineering Task Force (IETF) Request for Comments (RFC) 1583 Mar. 1, 1994, 132 pgs.
- Niki, Hitoshi Storage Network, *Computopia* vol. 36, No. 415, Japan, Computer Age Co., Ltd. (translation) Apr. 1, 2001, p. 52.
- Oguchi, et al., "A Study of Caching Proxy Mechanisms Realized on Wide Area Distributed Networks", High Performance Distributed Computing, 5th Int'l Symposium Aug. 1, 1996, pp. 443-449.
- Palmer, M. et al., "Fido: A Cache that Learns to Fetch", In Proc. the 17th Int'l Conf. on Very Large Data Bases , 255-264.
- Panigrahy, R. Relieving Hot Spots on the World Wide Web, Master's thesis, MIT EECS Jun. 1, 1997, pp. 1-66.
- Parris, C. et al., "A Dynamic Connection Management Scheme for Guaranteed Performance Services in Packet-Switching Integrated Services Networks", UC Berkeley Computer Science Division Tech. Report TR-93-005 Jan. 1, 1993, 37 pgs.
- Parris, C. et al., "The Dynamic Management of Guaranteed Performance Connections in Packet Switched Integrated Service Networks", UC Berkeley Computer Science Division and International Computer Science Institute Tech. Report CSD-94-859 Jan. 1, 1994.
- Partridge, C. et al., "Host Anycasting Service", Internet Engineering Task Force (IETF) Request for Comments (RFC) 1546 Nov. 1, 1993, pp. 1-9.
- Petri, S. et al., "Load Balancing and Fault Tolerance in Workstation Clusters. Migrating Groups of Communicating Processes", Operating Systems Review, vol. 29, No. 4 , 25-36.
- Plaxton, G. C. et al., "Fast Fault-Tolerant Concurrent Access to Shared Objects", In Proc. 37th IEEE Symp. of Foundations of Computer Science Oct. 1, 1996, pp. 570-579.
- Povey, D. et al., "A distributed internet cache", in Proc. of the 20th Australasian Computer Science Conf.
- Rabin, M. O. Efficient dispersal of information for security, load balancing, and fault tolerance, *JACM* 36, 2 , pp. 335-348.
- Rabinovich, M. et al., "Dynamic Replication on the Internet Work Project No. 3116-17-7006", AT&T Labs Research.
- Rabinovich, M. et al., "RaDaR: A Scalable Architecture for a Global Web Hosting Service", WWW8 May 1, 1999.
- Reisslein, M. et al., "Interactive video streaming with proxy servers", in Proc. of First International Workshop on Intelligent Multimedia Computing and Networking (IMMCN) 2000 , pp. 1-16.
- Rodriguez, P. et al., "Improving the WWW: Caching or Multicast?", *Computer Networks and ISDN Systems* North Holland Publishing, Amsterdam, NL, vol. 30, No. 22-23, ISSN: 0169-7552 , pp. 2223-2243.
- Ross, K. W. Hash-Routing for Collections of Shared Web Caches, *IEEE Network Magazine* 11, 7:37-44 , pp. 1-21.
- Sen, S. et al., "Proxy Prefix Caching for Multimedia Streams", *IEEE* Jun. 1, 1999, 10 pgs.
- Smith, Neil What can Archives offer the World Wide Web?, Technical Report 11, University of Kent, Computing Laboratory, University of Kent, Canterbury, UK Mar. 1, 1994, 1-12.
- Takahashi, Takao How to customize Apache Q&A—Web server using Apache, AI Publishing, Apr. 30, 1999, First Edition , pp. 63-64 (translation).
- Thaler, D. G. et al., "Using name-based mappings to increase hit rates", IEEE/ACM Trans. Netw. 6, 1 , 1-14.
- Topolcic, C. Experimental Internet Stream Protocol, Version 2 (ST-II), Internet Engineering Task Force (IETF) Request for Comments (RFC) 1190 Oct. 1, 1990, pp. 1-148.
- Touch, J. et al., "The X-Bone", Third Global Internet Mini-Conference at Globecom '98, Sydney, Australia (pp. 44-52 of mini-conference) Nov. 1, 1998, pp. 59-68.
- Traina, P. BGP-4 Protocol Analysis, Internet Engineering Task Force (IETF) Request for Comments (RFC) 1774 Mar. 1, 1995, pp. 1-10.
- Vin, H. Multimedia Broadcasting Over the Internet: Part 1, *IEEE Multimedia* IEEE Computer Society, US, vol. 5, NR. 4 , pp. 78-82.
- Vitter, J. S. et al., "Optimal Prefetching via Data Compression", Proc. 32nd Annual IEEE Symposium on Foundations of Computer Science Oct. 1, 1991, 21 pages.
- Wessels, D. Configuring Hierarchical Squid Caches, , 62 pgs.
- Wessels, Duane Intelligent Caching for World-Wide Web Objects, Masters Thesis, University of Colorado (also presented at INET '95 in Jun. '95) Jan. 1, 1995, 1-85.
- Wessels, D. et al., "Internet Cache Protocol (ICP), Version 2", Internet Engineering Task Force (IETF) Request for Comments (RFC) 2186 , pp. 1-9.
- Wolfson, O. et al., "An Adaptive Data Replication Algorithm", ACM Transactions on Database Systems (TODS) vol. 22(4) Jun. 1, 1997, pp. 255-314.
- Xiao, Li et al., "Exploiting neglected data locality in browsers", in Proc. 10th Int'l World Wide Web Conference, (WWW10), Hong Kong (extended abstract) , 2 pgs.
- European Decision to Refuse, dated Feb. 7, 2020, Application No. 09727342.9, filed Mar. 21, 2009; 15 pgs.

* cited by examiner

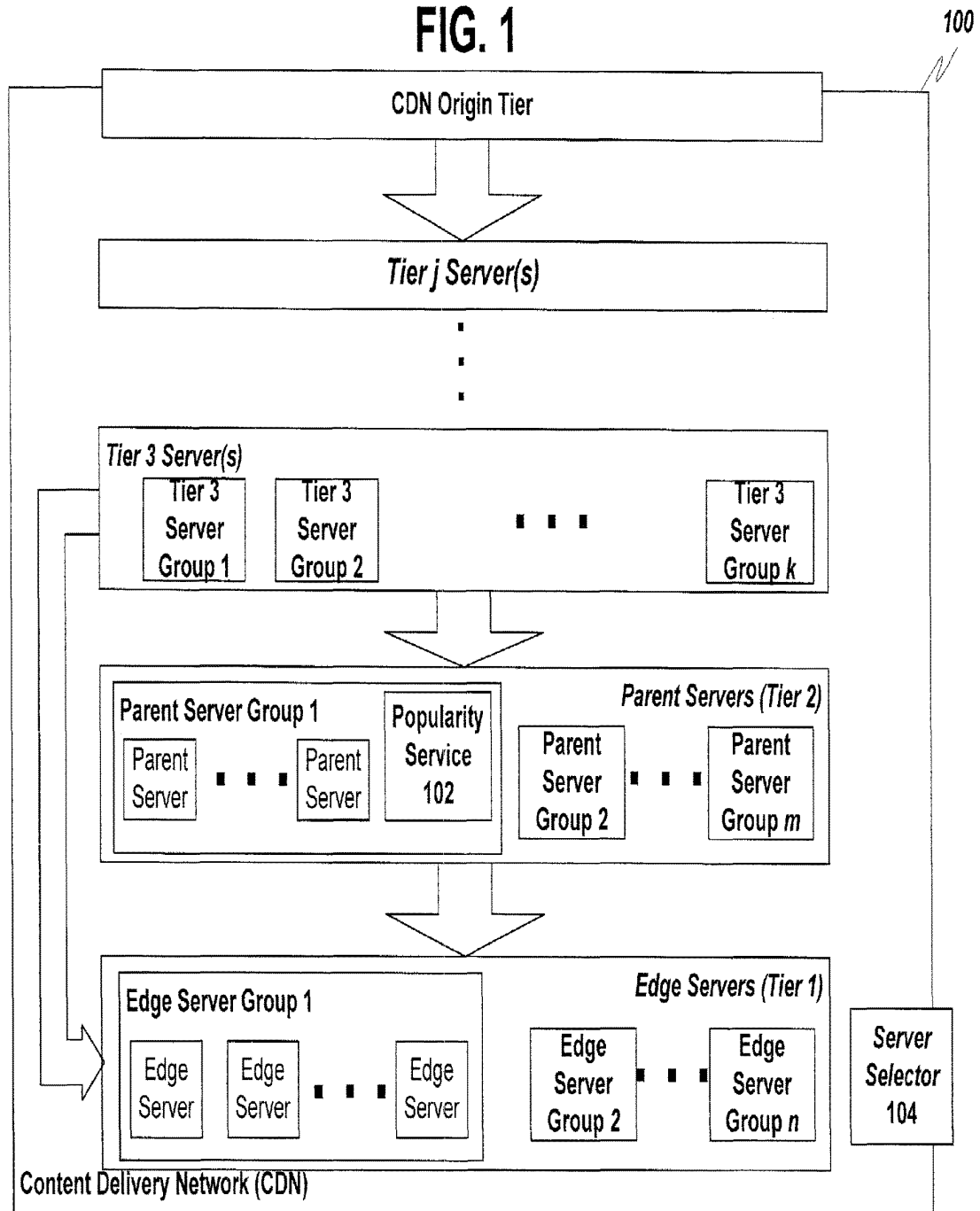
U.S. Patent

Feb. 16, 2021

Sheet 1 of 7

US 10,924,573 B2

FIG. 1



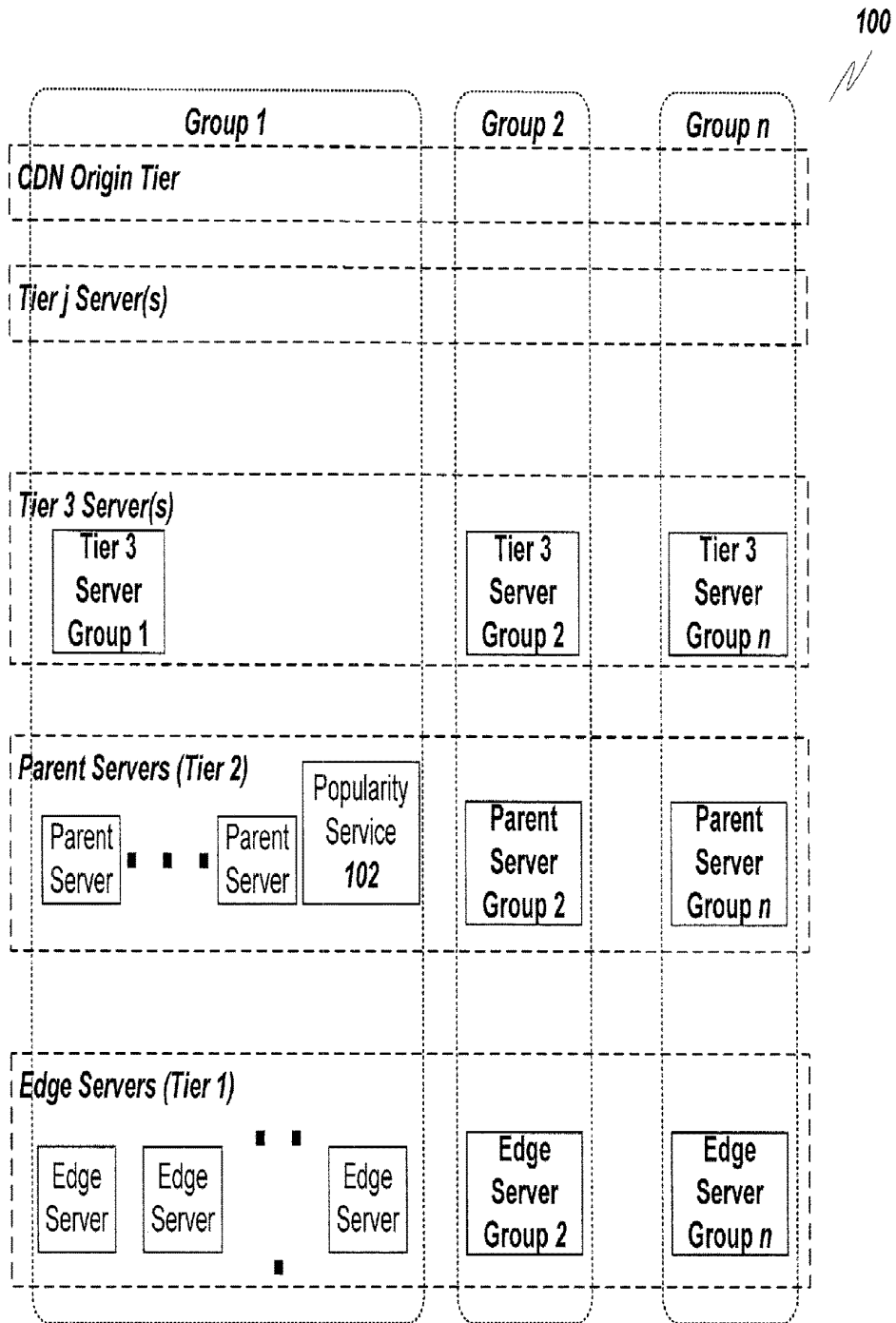
U.S. Patent

Feb. 16, 2021

Sheet 2 of 7

US 10,924,573 B2

FIG. 2



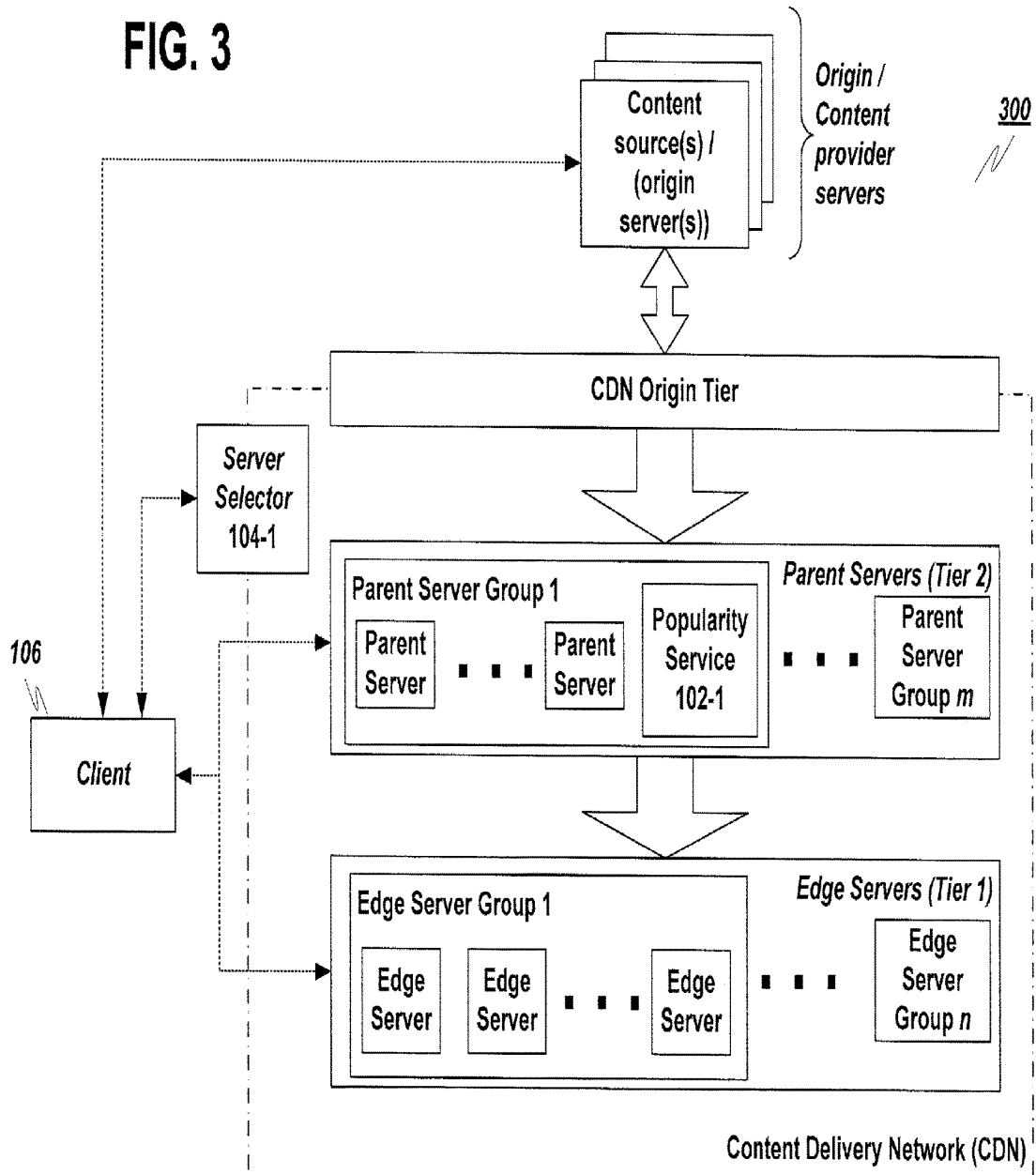
U.S. Patent

Feb. 16, 2021

Sheet 3 of 7

US 10,924,573 B2

FIG. 3

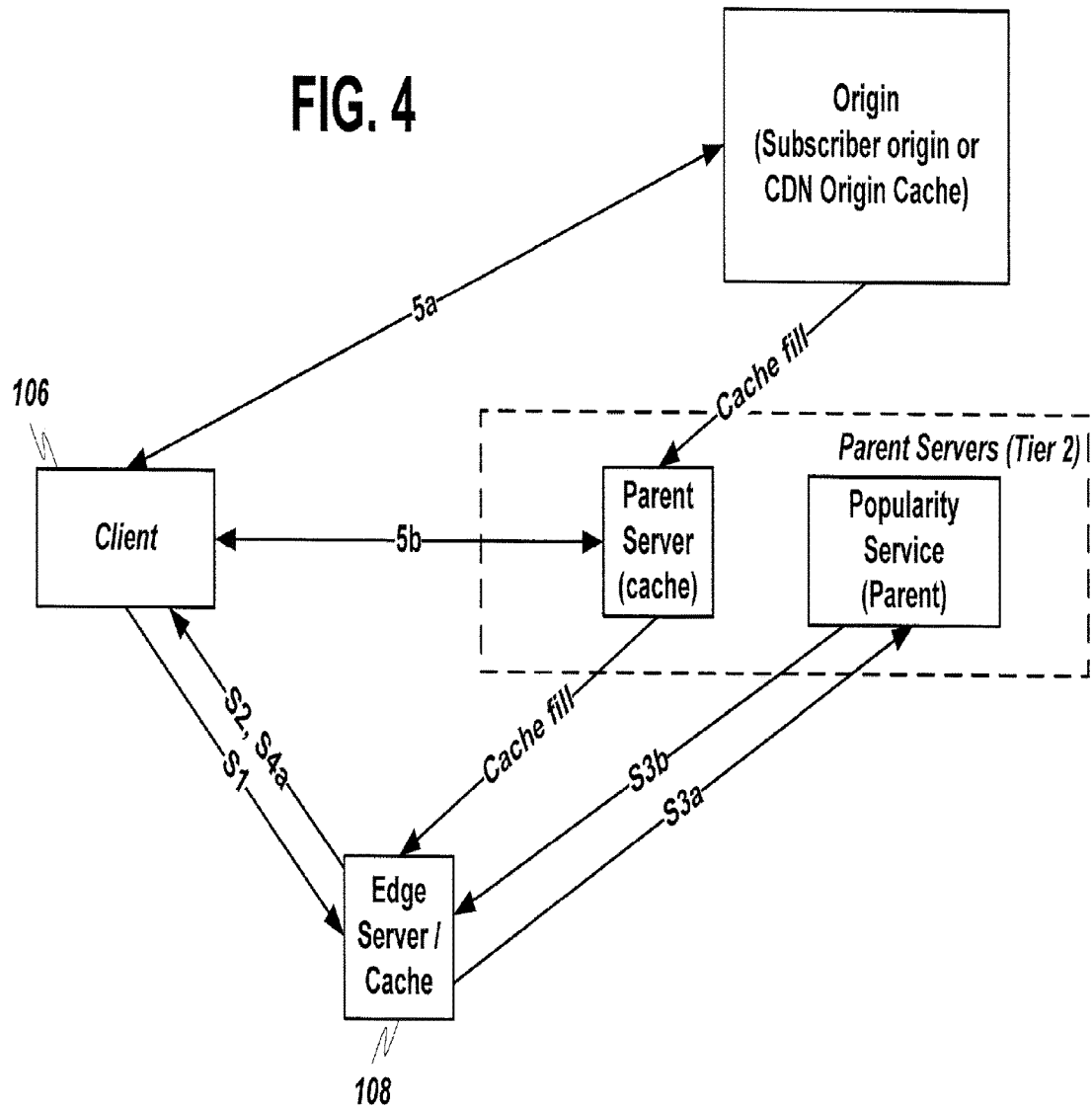


U.S. Patent

Feb. 16, 2021

Sheet 4 of 7

US 10,924,573 B2

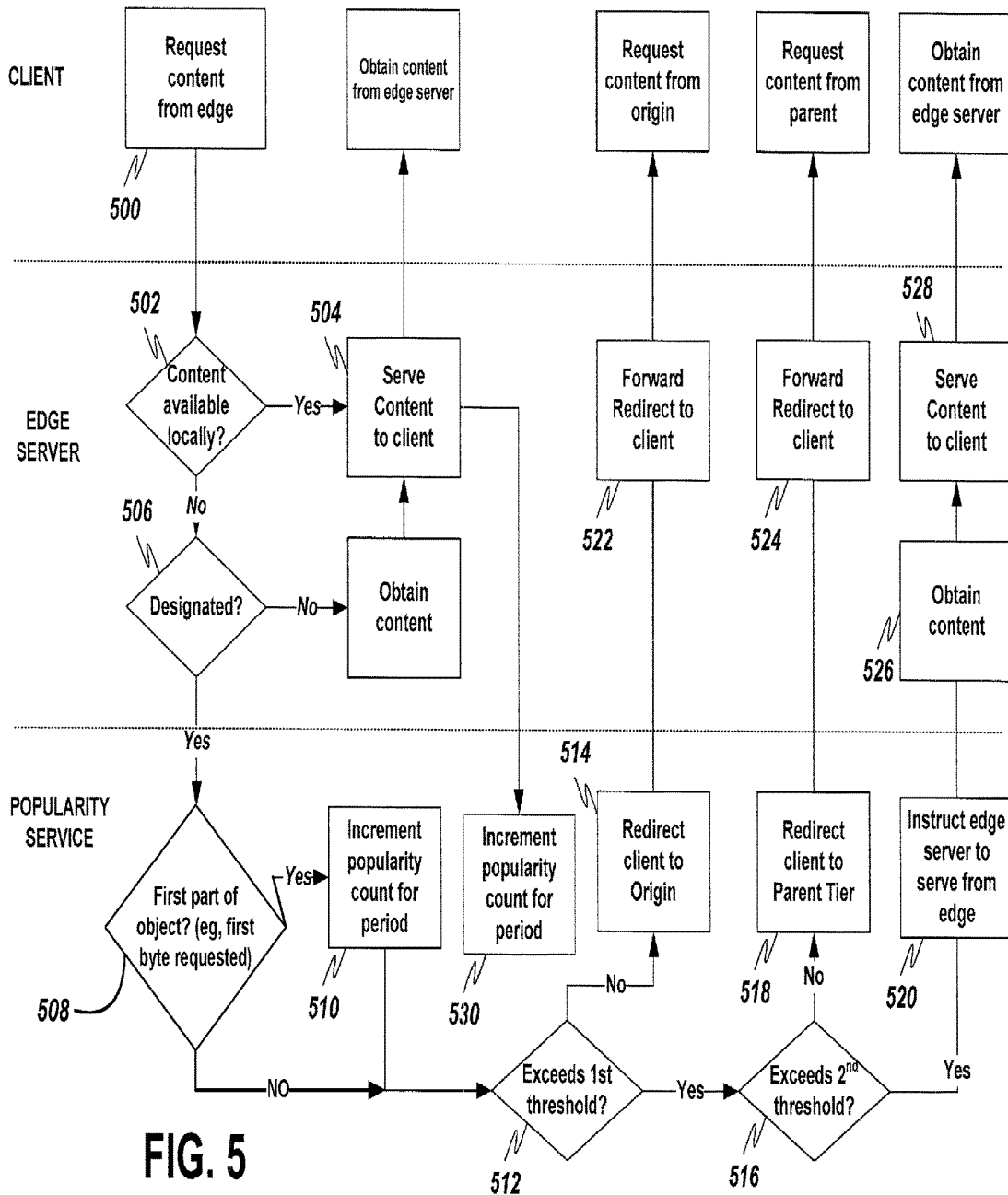


U.S. Patent

Feb. 16, 2021

Sheet 5 of 7

US 10,924,573 B2



U.S. Patent

Feb. 16, 2021

Sheet 6 of 7

US 10,924,573 B2

FIG. 6

600	SLOT	Time Bucket 1	Time Bucket 2	...	Time Bucket 16
				...	
				...	
				..	

U.S. Patent

Feb. 16, 2021

Sheet 7 of 7

US 10,924,573 B2

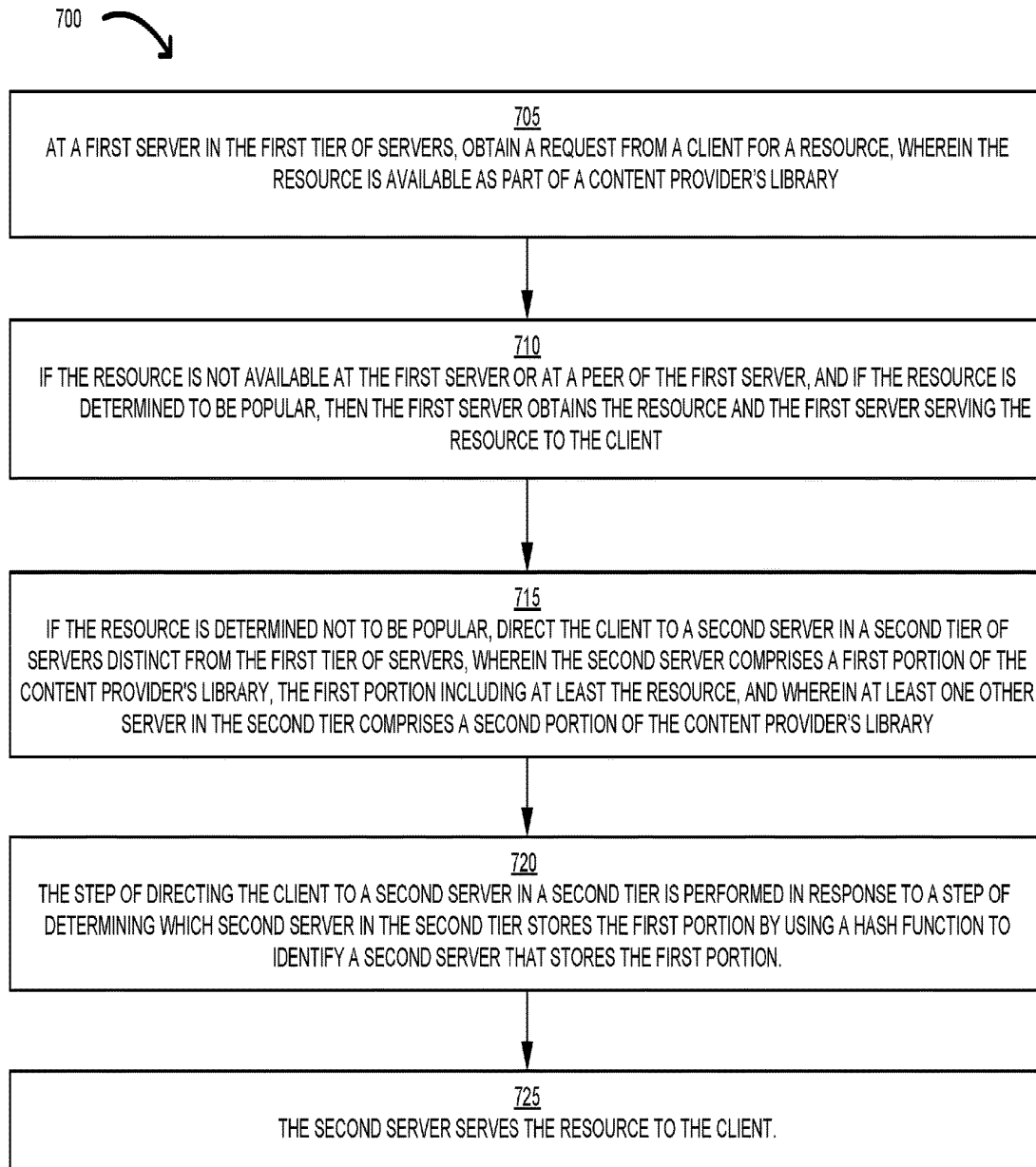


FIG. 7

US 10,924,573 B2

1

HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of and claims the benefit of priority from U.S. patent application Ser. No. 15/645,584, entitled "HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)," filed Jul. 10, 2017, the entire contents of which are fully incorporated by reference herein for all purposes. Application Ser. No. 15/645,584 is a continuation of and claims the benefit of priority from U.S. patent application Ser. No. 12/880,324, entitled "HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)," filed Sep. 13, 2010, the entire contents of which are fully incorporated by reference herein for all purposes. Application Ser. No. 12/880,324 is a continuation-in-part (CIP) of and claims the benefit of priority from U.S. patent application Ser. No. 12/408,681, entitled "HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)," filed Mar. 21, 2009 (now U.S. Pat. No. 8,930,538 issued Jan. 6, 2015), the entire contents of which are fully incorporated by reference herein for all purposes. Application Ser. No. 12/408,681 claims priority as a non-provisional application from U.S. Provisional Application No. 61/042,412, entitled "HANDLING LONG-TAIL CONTENT IN A CONTENT DELIVERY NETWORK (CDN)," filed Apr. 4, 2008 the entire contents of which are fully incorporated by reference herein for all purposes.

This application is also related to the following co-owned and patent applications, the contents of each of which are fully incorporated herein by reference for all purposes: application Ser. No. 10/073,938 filed Feb. 14, 2002, application Ser. No. 11/715,316 filed Mar. 8, 2007, application Ser. No. 11/978,656 filed Oct. 30, 2007, Application Ser. No. 11/980,672 filed Oct. 31, 2007, application Ser. No. 10/259,497 filed Sep. 30, 2002, application Ser. No. 11/932,162 filed Oct. 31, 2007, application Ser. No. 11/976,648 filed Oct. 26, 2007, and application Ser. No. 12/390,560 filed Feb. 23, 2009.

FIELD OF THE INVENTION

This invention relates to content delivery, to content delivery networks (CDNs), and to frameworks and systems using CDNs.

DETAILED DESCRIPTION

Glossary

As used herein, unless stated otherwise, the following terms or abbreviations have the following meanings:

1. IP means Internet Protocol.
2. "IP address" means an address used in the Internet Protocol to identify electronic devices such as servers and the like.
3. HTTP means Hypertext Transfer Protocol.
4. URL means Uniform Resource Locator.
5. DNS means Domain Name System.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention can be better understood with reference to the following drawings. The elements of the

2

drawings are not necessarily to scale, emphasis instead being placed upon clearly illustrating the principles of the present invention. Furthermore, like referenced numerals designate corresponding parts throughout the several views.

FIG. 1 depicts a general hierarchical, multi-tiered content delivery network (CDN);

FIG. 2 shows the logical organization of servers into groups or clusters in a CDN;

FIG. 3 depicts a content delivery framework (CDF) using a content delivery network (CDN);

FIG. 4 depicts the operation of a two-level content delivery network in a content delivery framework;

FIG. 5 is a flowchart showing operation of the popularity services of the CDN of FIG. 4;

FIG. 6 is an exemplary data structure for maintaining the popularity data for a particular popularity server; and

FIG. 7 is a flowchart showing processing steps of a multi-tiered content delivery network.

OVERVIEW

The Internet and the so-called World Wide Web (the "WWW") have become ubiquitous. Thousands or even tens of thousands of so-called content providers (publishers) now use the Internet (and, particularly, the WWW) to provide all sorts of content to tens or even hundreds of thousands of clients all over the world.

In order to offload the job of serving some or all of their content, many content providers now subscribe to so-called content delivery networks (CDNs). Using a CDN, some (or all) of a content provider's content can be served to clients from the CDN (i.e., from one or more servers in the CDN) instead of from the content provider's server(s). In a caching CDN, content that is served may also be cached on some or all of the CDN servers, either before being served or in response to specific requests for that content.

The term content as used herein means any kind of data, in any form, regardless of its representation and regardless of what it represents. Content may include, without limitation, static and/or dynamic images, text, audio content, including streamed audio, video content, including streamed video, web pages, computer programs, documents, files, and the like. Some content may be embedded in other content, e.g., using markup languages such as HTML and XML. Content includes content which is created or formed or composed specifically in response to a particular request. The term "resource" is sometimes used herein to refer to content.

Certain publishers can have large content libraries in which only a small proportion of the content (the so-called "short head") is popular enough to benefit from serving through a caching CDN, while the majority of the content (the so-called "long tail") is accessed only occasionally and not generally worth caching (or even serving from an edge server). This situation would be typical for a content publisher with a very large music or video library. Some music content—the popular content—may be regularly requested, whereas other music—the not popular (also referred to as unpopular) content—may be seldom if ever requested.

Content can become popular (by various measures of popularity) or fade into relative obscurity dynamically, so a content library cannot easily be explicitly partitioned. Instead, the CDN tracks popularity of certain content, and selectively migrates content toward the edge (i.e., toward the tier 1 servers) as that content becomes popular.

A CDN may have one or more tiers of servers, organized hierarchically. FIG. 1 depicts a content delivery network

US 10,924,573 B2

3

that includes multiple tiers of servers. Specifically, the CDN **100** of FIG. 1 shows j tiers of servers, denoted Tier 1, Tier 2, Tier 3, . . . , Tier j . Each tier of servers may comprise a number of servers organized into server groups (sometimes referred to as server clusters). The Tier 1 servers are also referred to as edge servers, and Tier 1 is sometimes also referred to as the “edge” or the “edge of the CDN.” The Tier 2 servers (when present in a CDN) are also referred to as parent servers.

Furthermore, content stored in any of the intermediate tier servers (e.g., tier 2, tier 3, . . . tier j in FIG. 1) may be logically partitioned across two or more servers in the respective tier group. For example, a first portion (or subset) of a content provider’s library may be stored on a first parent server in Group 1, a second portion may be stored on a second server in Group 1, and the last third may be stored on a third server in Group 1. Various partitioning methodologies are described in further detail below. In operation, the partitioning of content libraries across servers in an intermediate tier of a CDN provides a type of preemptive load-balancing such that certain parent servers are only responsible for handling a pre-defined subset of a content provider’s library. In other words, redirected requests from the client to a parent/intermediate tier are distributed within the tier in accordance with the particular partitioning methodology being implemented.

For example, in the CDN **100** of FIG. 1, Tier 1 has n groups of servers (denoted “Edge Server Group 1”, “Edge Server Group 2”, . . . , “Edge Server Group n ”); tier 2 (the parent servers’ tier) has m server groups (the i -th group being denoted “Parent Server Group i ”); and tier 3 has k server groups, and so on. Preferably each tier has the same number of server groups.

FIG. 2 shows the logical organization/grouping of servers in a CDN of FIG. 1. In the exemplary CDN of FIG. 2, each tier of servers has the same number (n) of server groups. Those of skill in the art will know and understand, upon reading this description, that each server group may have the same or a different number of servers. Additionally, the number of servers in a server group may vary dynamically. For example, additional servers may be added to a server group to deal with increased load on the group.

The servers in a server group may be homogenous or heterogeneous, and each server in a server group may comprise a cluster of physical servers sharing the same name and/or network address. An example of such a cluster is described in co-owned patent application No. 61/064,339 (titled “Load-balancing cluster”, filed Feb. 28, 2008), the entire contents of which are incorporated herein by reference for all purposes.

Servers in the same tier and the same group are referred to as peers or peer servers.

A typical CDN has only one or two tiers of servers. A CDN with only one tier will have only edge servers, whereas a CDN with two tiers will have edge servers and parent servers. (At a minimum, a CDN should have at least one tier of servers—the edge servers.)

The grouping of servers in a tier may be based, e.g., on their physical or geographical location. For example, a particular CDN may have six groups—four groups of servers in the United States, Group 1 for the West Coast, Group 2 for the mid-west, Group 3 for the northeast and Group 4 for the south east; and one group each for Europe and Asia.

A particular CDN server is preferably in only one server group.

In general, some or all of the servers in each tier can exchange data with some or all of the servers in each other

4

tier. Thus, some or all of the parent servers can exchange information with some or all of the edge servers. For the sake of simplicity, in the drawings, each tier of servers is shown as being operationally connectable to each other tier. In some CDNs, however, it may be preferable that the servers in a particular tier can only exchange information with other servers in the same group (i.e., with peer servers) and/or with other servers in the same group in a different tier. For example, in some CDNs, the edge servers in edge server group k , can exchange information with each other and with all servers in parent server group k , and so on.

A content provider’s/customer’s server (or servers) are also referred to as origin servers. A content provider’s origin servers may be owned and/or operated by that content provider or they may be servers provided and/or operated by a third party such as a hosting provider. The hosting provider for a particular content provider may also provide CDN services to that content provider.

A CDN may also include a CDN origin/content cache tier which may be used to cache content from the CDN’s subscribers (i.e., from the CDN subscribers’ respective origin servers). Those of skill in the art will know and understand, upon reading this description, that a CDN can support one or more subscribers, i.e., that a CDN can function as a shared infrastructure supporting numerous subscribers. The CDN origin tier may also consist of a number of servers, and these servers may also be organized (physically and logically) into a number of regions and/or groups. The server(s) in the CDN origin tier obtain content from the subscribers’ origin servers, either on an as needed basis (a pull) or in advance (via a push).

A popularity service **102** (described in greater detail below) is associated with one or more of the server groups in one or more tiers. In a presently preferred exemplary embodiment, some of the parent server groups have popularity services **102** associated therewith. Although shown as a separate component of the group, the popularity service **102** may be integrated into one of the servers in the group. In some cases, the popularity service may have its own server, distinct from any of the CDN servers. The terms “popularity service” and “popularity server” are used interchangeably herein.

In operation, when a client requests content that is to be served using a content delivery framework, the client may be served that content from a server in the CDN or, in some cases, from the subscriber/customer’s origin server.

A client may be directed to a CDN and/or to a server in the CDN in any manner using any kind of server selector system **104**. As understood by those of skill in the art, the server selector system **104** generally operates to direct a client’s requests for content to an appropriate server in order for that content to be served to the requesting client. An appropriate server may be one which is close to the client (by some measure of cost) and/or one which is not too heavily loaded. All sorts of conditions may be applied to the term “appropriate”, and all sorts of information and tests, both static and dynamic, may be used to determine an appropriate server. The server selector system **106** may, e.g., include or operate fully or partially in Domain Name Service (DNS) servers, standalone devices, or a combination thereof. For example, the server selector system **106** may comprise a single level DNS server that selects an appropriate server based at least in part on some combination of the location of the requesting client and the load on some or all of the CDN servers. Those of skill in the art will know and understand, upon reading this description, that a client’s location in a network such as the Internet may sometimes

US 10,924,573 B2

5

only be roughly determined, and the term “location of the client” is generally taken to be a network location corresponding to the client’s network service provider.

Although shown as a component in the drawings, the server selector **106** may comprise numerous components. For example, some or all of the server selection may be based on anycast routing, and the server selector **106** may then include routers and associated tables.

In a presently preferred embodiment the server selector **106** is an intelligent traffic manager (ITM)/adaptive traffic controller (ATC) such as described in co-pending U.S. patent application Ser. No. 10/259,497, filed Sep. 30, 2002, and titled “Configurable Adaptive Global Traffic Control And Management,” (published as US 2003-0065762 A1); and in U.S. patent application Ser. No. 11/976,648, filed Oct. 26, 2007, titled “Policy-based content delivery network selection,” (collectively the “ITM applications”), the entire contents of each of which have been incorporated herein by reference for all purposes. In some embodiments the server selector **106** may include a “best” or “optimal” server selector such as disclosed in U.S. Pat. No. 6,185,598 titled, “Optimized Network Resource Location,” the entire contents of which are incorporated herein by reference for all purposes. The ’598 patent refers to CDN servers as so-called repeater servers, and describes a so-called “Best Repeater Selector (BRS) mechanism”.

FIG. 3 shows a content delivery framework **300** with a two-level hierarchical CDN consisting of a tier of edge servers (Tier 1) and a tier of parent servers (Tier 2). Some or all of the edge servers can communicate with some or all of the parent servers. The edge servers are divided into *n* edge server groups, and the parent servers are divided into *m* parent server groups. In a presently preferred embodiment, the value of *m* is equal to the value of *n*, i.e., in this presently preferred embodiment there are the same number of edge server groups as there are parent server groups. A CDN origin/content cache tier stores subscriber content which it obtains from the various subscribers’ origin servers. At least one of the parent server groups (in the drawing, Group 1) has a popularity service **102-1** associated therewith. Preferably more than one parent server group has an associated popularity service, and more preferably each parent server group has an associated popularity service.

Although shown in a parent tier, the popularity service may be located anywhere in the system, including in the edge tier.

The popularity service may be used by certain, though not necessarily all, content. When only certain content uses the popularity service, content should be designated in order to use the popularity service.

Some or all of the edge servers in a group may use a popularity service to manage the long-tail content of various subscribers. Each edge server that uses a popularity service is referred to as being bound to that popularity service. An edge server that is bound to a popularity service is sometimes referred to herein as a “longtail coserver.”

FIGS. 4 and 5 show the operation of the popularity services of the CDN of FIG. 3.

When a client **106** requests content (e.g., using an HTTP GET request), that request is directed (e.g., by server selector **104-1**) to an edge server in order for the content to be served to the client. For certain designated content a popularity check is interposed into the fill side of the caching operation. FIG. 4 illustrates the flow of messages and data in the content delivery framework **300**, and FIG. 5 is a flow-chart show operation of the popularity services of the CDN of FIG. 4. For the sake of this particular explanation, assume

6

that the client’s request has been directed to edge server **108**. (Those of skill in the art will know and understand, upon reading this description, that the client’s initial request may be directed to any tier in the CDN hierarchy, including, e.g., to the parent tier.) This server is selected using the server selection mechanisms **104-1** associated with the CDN, e.g., using one or more DNS servers and selecting an edge server based on such factors as the location of the requesting client, the load on the network, network traffic conditions, CDN policies, subscriber policies, and the like.

A client **104** requests content from an edge server **108** (at **500** in FIG. 5). The request from the client **104** arrives at edge server **108** (**S1** in FIG. 4). The edge server **108** checks to see if object is present (locally or on a peer) and fresh (at **502**). If so, the edge server **108** serves the object to the client **104** from the cache (**S2**, **504**), obtaining the object from a peer, if necessary.

In some embodiments, the system may distinguish between on-net and off-net peers and same-switch peers. An on-net peer is a peer on the same backbone network; an off-net peer is a peer located on a different backbone network; and a same-switch peer is a peer directly connected to the same switch as the agent performing the check. In some embodiments, the edge server **108** may only look for the object on some of its peers (e.g., only on same-switch peers) (at **502**).

If the object is not available on the edge server **108** or on a peer, the edge server **108** ascertains whether this object is served based on its popularity (i.e., whether this object has been designated so that the object’s popularity will be used to determine where it will be served from) (at **506**). If so, then the request is sent to the popularity service **102** associated with the edge server **108**, in this case, to the popularity server for the this group (**S3a**).

The determination as to whether this object is designated to be served from a different location, depending on its popularity (at **506**), may be made based, at least in part, on the name (hostname) used to request the object.

It is preferably to allow for a mix of edge servers, some performing popularity checking (as described above), while others do not. For those that are not running the popularity service, the name (hostname) used to request an object will resolve to a parent server (that may or may not provide popularity services). If the parent server does not provide popularity services, then the content will be obtained by the edge server from that parent server, and the content will be served to the client. On the other hand, if that parent server does provide popularity services, it can determine whether or not the edge server is a Longtail coserver based, e.g., on the IP (Internet Protocol) address of the edge server. For no-coservers, the parent server can handle the request without any popularity processing.

A request for content may be an initial request for an object or it may be a request for another part of an object, the initial part having already been served to the client. If the request is for the first part of the object (at **508**), e.g., the request includes a request for the first byte of the resource (i.e., it is not a range request that starts after the beginning of the file), the popularity service **102** determines (as described below) if the object is currently popular. First, the popularity count for the current period is incremented (at **510**). Based on its determination, the popularity service **102** returns one of three possible responses to the edge server **108** (**S3b**):

1. If the object has not reached a first I minimal level of popularity (at **512**): the popularity service sends the edge server an instruction (e.g., HTTP 302) to redirect

US 10,924,573 B2

7

the client's request to the origin server (or to the CDN origin cache) (if origin redirects are enabled) (at **514**).

2. If the object's popularity has exceeded the first/minimal level of popularity but has not yet exceeded a second, mid-tier threshold (at **516**): the popularity service sends the edge server an instruction (e.g., HTTP 302) to redirect the client's request to a parent server (if mid-tier redirects are enabled) (at **518**). Note that the redirect processing may further involve a determination of which particular parent server is responsible for storing the requested content. Such a determination would be based on how the distinct subsets of a content provider's library are partitioned across parent servers in the same intermediate tier. Further note that it is also possible for subsets of a content provider's library to be partitioned across servers residing in different intermediate tiers of a content delivery network.
3. If the object's popularity has exceeded the mid-tier threshold (i.e., the object is popular): The popularity service sends the edge server an instruction to serve the content itself (at **520**). In a presently preferred implementation, the popularity service sends the edge server a redirect (HTTP 302) with a "follow me" flag set, to the origin server or, if there is one, to the parent tier.

If the edge server **108** receives a redirect from the popularity service **102** without the "follow me" flag set (cases **1** and **2** above), it simply forwards the redirect to the client **104** (**S4a**, **522**, **524**). If the edge server **108** receives a "follow me" redirect, it obtains and caches the resource (at **526**) and serves it to the client (at **528**).

If the popularity service **102** is unreachable, unresponsive, or returns a status code indicating an error (other than HTTP 404), the object is served out of the edge's cache server (and an alert condition is raised).

Once content has been cached at an edge server, the edge server will send notifications (e.g., in the form of revalidations) to the popularity service every time it gets another request for that content. E.g., with reference to the flowchart of FIG. 5, if the edge server **108** determines (at **502**) that it has the requested content (or can obtain it from a peer), then, in addition to serving the content (at **504**), it also instructs the popularity server to increment the object's popularity count for the current period (at **530**). This process keeps the popularity servers up to date on the relative popularity of content being served in their region.

In presently preferred embodiments, the server selection mechanism **104** does not rendezvous clients to parent servers/caches. In other words, in these embodiments, client requests are always initially directed by the server selection mechanism to an edge server. In these cases, when a request for a resource arrives at a parent server/cache, that request should preferably be served (and filled if necessary) unconditionally (since any request from a client is assumed to be the result of a redirect served by an edge server).

In an embodiment where the server selector **104** can direct client requests directly to parent servers (or to any tier other than the edge tier), a server obtaining a client request may choose to redirect that request, e.g., based on popularity. However, those of skill in the art will know and understand, upon reading this description, that it is advisable to track the redirection of a request to avoid circular and/or infinite redirection. One way to avoid such a problem is to limit the number of levels of redirection (i.e., to limit the number of redirects to follow). In a presently preferred implementation, if no final server is selected after following, e.g., thirty two redirects, an error is issued. In some embodiments, if no final sever is selected after a predefined number of redirects, then

8

the last server reached may be used to serve the content. One way to prevent looping is to use different server names (aliases) or IP addresses when redirecting requests so that a server receiving a request can tell whether or not it is a redirect. Those of skill in the art will know and understand, upon reading this description, that information can be transferred between servers using, e.g., HTTP headers or the like.

Those of skill in the art will know and understand, upon reading this description, that in a multi-tier CDN, the popularity service may be located at any tier, or there may be popularity services at more than one tier.

The middle (Parent) tier is optional.

Step (**4a**) may reply with content (if popular), or with a redirect to a parent or origin server (if not), in which the client will make another request (**5a** or **5b**) to that tier to obtain the content.

If the request is an HTTP GET request or the like, it is forwarded to the popularity service. HTTP POST requests should always be forwarded directly to the origin, since that is where they will need to be processed, and the response to a POST request should not be cached. It may sometimes be preferable to direct GET requests to a different origin server than POST requests.

While the invention has been described with reference to the HTTP protocol, those of skill in the art will know and understand, upon reading this description, that different and/or other protocols may be used and are contemplated by the inventors. HTTP is described in various documents, e.g., Hypertext Transfer Protocol—HTTP/1.1, RFC 2616, Network Working Group, the entire contents of which are incorporated herein by reference.

Those of skill in the art will know and understand, upon reading this description, that different thresholds may be established for each tier in the CDN. Further, those of skill in the art will know and understand, upon reading this description, that each content item may have its own thresholds associated therewith. In this manner, the system can check all content for popularity, with the default thresholds being zero. In this manner, every request will automatically cause the popularity to exceed the threshold and will cause the content to be cached.

By positioning Popularity Servers regionally (paired with parent cache servers), popularity and cache tiers can be managed independently, on a regional basis. Content that is popular in one region/group may not be popular in another region/group (especially if each region/group corresponds to a geographic and/or political region).

We consider it desirable that rendezvous to popularity servers prioritize so-called "regional" proximity, so that clients within the same region will tend to cast their popularity "votes" within that region and get consistent treatment of popular resources. However, if there are multiple parent cache servers available, there will generally be no attempt to rendezvous particular clients to particular parents.

Defining & Measuring Popularity

In preferred embodiments, popularity of an object/resource is measured based on the number of times that object/resource is requested in various time periods. FIG. 6 is an exemplary data structure for maintaining the popularity data for a particular popularity server. The data structure **600** in FIG. 6 is a so-called tally hash structure.

In preferred embodiments, some or all edge servers are associated with (or bound to) popularity servers. An edge server that is bound to a popularity server is sometimes referred to as a bound Longtail coserver. Each popularity

US 10,924,573 B2

9

server in the system allocates a tally hash structure **800** per bound Longtail coserver. A configuration provides the number of resource (hash) slots to allocate. For a presently preferred implementation, the number of hash slots is on the order of 100 million slots per coserver. Each slot is divided into a number of time buckets, preferably 16 time buckets, each bucket being represented by, e.g., a 4-bit unsigned integer. Those of skill in the art will know and understand, upon reading this description, that the selection of the size of the value in each time bucket depends on policy decisions about bounds for the popularity thresholds, and for keeping very popular resources at the edge. The size, however, is heavily influenced by a need for compactness. One 8-byte word can store all time buckets for one resource slot, and therefore, 800 MB would be required per property, and five to eight such properties could be managed per popularity server without paging.

Each time bucket represents a time period, preferably a number of seconds.

The mapping of requests/content to slots is based on some function of the object name and perhaps other information associated with the request for the object. Preferably the mapping of objects to slots is based on a hash or message digest function (such as MAD or the like) over the object name (and preferably including some parts of the query string). Each slot may therefore represent one or more resources. Each time a query/request arrives at a popularity server for an object, the hash is computed and the slot in the table **800** (for the appropriate co-server) is determined, and the counts in that slot are used. In event of a hash collision, it is therefore possible that one bucket will be receiving and representing counts for more than one object. Since this result is generally undesirable (since it could result in cache fills and edge caching of unpopular objects), the number of buckets should be chosen to be as large as practical.

Those of skill in the art will know and understand, upon reading this description, that different and/or other data structures may be used to implement the popularity counting. For example, since in most cases the total number of resources is expected to far exceed the number of popular resources, a balanced b-tree may be preferable to a hash table. In addition, it is possible to reduce the size of the hash slot by using only some part of the hash. However, reducing the number of bytes of the hash used can result in more name collisions.

Although described above with respect to popularity, those of skill in the art will know and understand, upon reading this description, that other factors may be used along with (or instead of) popularity to determine whether or not to redirect requests. A rule base may be used to augment and/or override the popularity measures for certain resources. The rules in the rule base may be static or dynamic and may be set by the CDN administrator and/or the subscriber. For example, a subscriber may not want to pay for certain content to be served from the edge, regardless of its popularity, and may set a rule accordingly (this particular result could also be achieved by setting the thresholds for that particular content to prevent it from ever being cached at the edge).

Occasional log mining could be used to look for hash collisions in actual subscriber content libraries, and the hash function and bucket sizes could be tuned as needed.

At each time bucket boundary, the popularity service will logically "rotate" the buckets and zero out the oldest tally data for each object.

10

Whenever a coserver's enrollment in the popularity service changes (added or dropped, or perhaps hints changed), the data structures are to be updated.

The popularity of a given object may be determined as a weighted sum of its popularity over successive time periods. More recent time periods may be given higher weights.

In order to determine which content is to be managed by the popularity service, the CDN operator and/or the subscriber may specify:

The tiers at which the content will be managed—edge, intermediate (i.e., parent), or origin (subscriber's or storage tier). In order to be meaningful, at least one of intermediate and origin service should be enabled.

content that is to be managed based on its popularity, rather than simply always being served from a cache.

There are several reasons why a publisher/subscriber may not want the so-called "long-tail" content served from a caching CDN service, for example:

If cache fills are done from the subscriber's origin server and the subscriber pays for cache fill bandwidth, unnecessary fills for unpopular resources increase bandwidth costs without providing any value.

Serving an unpopular resource through a CDN cache adds latency (due to the cache fill), and risks forcing actually popular resources out of cache, potentially causing thrashing. The result is lower efficiency and the risk of degraded service.

For related or similar reasons, a CDN provider generally also does not want to serve long-tail content from an edge cache:

If cache fills are done from origin storage, unnecessary cache fills consume bandwidth both from the storage and cache systems, increasing (doubling) the internal cost to serve. This lowers efficiency, requiring relatively more bandwidth internally to serve the same content to the outside.

The second argument above also applies from the CDN's perspective: a CDN operator wants to minimize latency as well as the risk of thrashing in order to satisfy all of its subscribers.

Names, Addresses & Configuration Data

As is well known, each server in a network may be addressed by means of one or more network address (e.g., Internet Protocol or IP addresses). Each server in a network may also be known by one or more names (so-called hostnames—fully qualified domain names). Hostnames may be mapped to one or more IP addresses. A hostname may correspond to (and thus resolve to) more than one server.

A system such as ITM (described in the ITM patent applications mentioned above), allows a kind of hostname (called a supename) to refer to multiple servers, and resolves the supename to a nearby server.

Preferably the server selection mechanism is ITM, and each popularity server will have supename that resolves to reach nearby popularity server.

When a popularity server shares or is co-located with a parent server, the parent server may use the name by which it was addressed to determine whether to direct a request to the popularity service. That is, parent cache servers that provide popularity service may recognize requests that use one of the aliases reserved for popularity requests, and call into the popularity service to make the fill/no fill decision and return a redirect as described above.

As noted earlier, if the server selector mechanism does not send initial requests to non-edge servers, then all parent cache servers must recognize requests that have been redi-

US 10,924,573 B2

11

rected and serve the requested resource, filling it from the origin (or another tier), if necessary.

A servers hostnames are also referred to as its aliases. Each Longtail coserver preferably has at least two aliases (three if a parent cache/server tier is used): the published
5 surname, the hostname used for popularity service requests, and (if used) the hostname used for parent cache redirects.

Popularity servers will preferably be reached via an ITM supname, and ITM will monitor for the service's availability across the set of servers. Popularity servers should be reached using real IP addresses, and not virtual IPs, and will not necessarily be redundant within a cluster. Redundancy can be provided by having multiple servers per supname. However, preferably there will be no attempt to synchronize the popularity tallies on popularity servers, with the expected desirable effect of managing popularity separately on a "regional" basis, the granularity being determined by the number and distribution of popularity servers deployed. Should a popularity server fail, this could cause a discontinuity in popularity responses as a new server becomes active for a given edge location, but this may be mitigated (for very popular resources) by periodic background refreshes.

Information about the resources and caching strategy include the following:

the expected total number of resources associated with this coserver.

the number of buckets used to store hit counts for each resource.

the number of seconds that each bucket represents. Every time this interval goes by, the count in the oldest bucket is thrown out and a new bucket is started with a count of zero.

when the sum of all buckets for a given resource reaches this number on any popularity server, the parent caches (if any) that use that server will start to cache the resource.

when the sum of all buckets for a given resource reaches this number on any popularity server, the edge caches that use that server will start to cache the resource.

the hash algorithm to apply to the resource names (optional). If not specified, a default algorithm (e.g. MD5) will be used.

the maximum number of this coserver's resources that should be in cache at an edge at any given time. This value may be ignored in certain embodiments.

the maximum number of this coserver's resources that should be in cache at a parent at any given time. This value may be ignored in certain embodiments.

Preferably resources should not be pre-expired on parent cache servers, as that will cause unnecessary requests to the origin server or queries to peer caches.

Those of skill in the art will know and understand, upon reading this description, that a decision to serve at a given tier that is based only on popularity counts, will not take into account capacity to serve at that tier—so this scheme could overload an origin server or parent tier if they do not have sufficient capacity. Further, if popularity is measured in terms of absolute thresholds on numbers of requests, and if the library is sufficiently large, this could cause cache thrashing at the parent or edge tiers.

Authentication with the origin cache or server, if needed, should be done by the server that receives the initial request from the client. During processing of a redirected request.

Various documents, including patents and patent applications, have been incorporated by reference into this application. In case of any conflict between an incorporated

12

document and the present application, the present application, including any definitions herein, will control.

Thus is provided a feature that allows a CDN to be responsive to increasing or decreasing "popularity" of content by shaping where in the CDN content is positioned and served from.

FIG. 7 is a flowchart 700 of processing steps associated with a multi-tiered content delivery network.

In step 705, a first server in a first tier of servers obtains a request from a client for a resource. In one embodiment, the resource is available as part of a content provider's library (e.g., the resource may be a digital movie requested from an on-line streaming movie provider).

In step 710, if the resource is not available at the first server or at a peer of the first server, and additionally if it is determined that the resource is popular (using processing and functionality previously described), then the first server obtains the resource and serves it to the client. For example, the first server may obtain the resource from an origin server and/or a parent server (in any intermediate tier between the first server and the origin server).

In step 715, if the resource is determined not to be popular (again, using processing and functionality previously described), the client is directed to a second server in a second tier of servers (e.g., any intermediate tier) distinct from the first tier of servers. In this example embodiment, the second server comprises a first portion (or subset) of the content provider's library. For the purposes of this example, the first portion comprises at least the resource that was requested by the client. Furthermore, the second tier includes at least one other server that comprises a second portion of the content provider's library. According to one example embodiment, the first portion is distinct from the second portion (i.e., the portions/subsets of the library do not overlap, or overlap only minimally).

In another example embodiment, the portions of the content provider's library are logically partitioned across servers in the second tier. For example, the portions/subsets of a content provider's library can be partitioned alphabetically across servers in the second tier based on a naming convention associated with each distinct resource in the content provider's library (e.g., digital movies having titles beginning with A-F are stored on server A in a second/intermediate tier, digital movies having titles beginning with G-R are stored on server B in the second/intermediate tier, and digital movies having titles beginning with S-Z are stored on server C in the second/intermediate tier).

Assume, for example, that a content provider's library comprises multiple content-types (e.g., movies, video games, music, software patches, etc.). Per one example embodiment, it would be advantageous to store a proportionate amount of each content type on each participating intermediate tier server (e.g., 60% movies, 30% video games, 10% software downloads on each participating server in the tier). In such a scenario, if the demand for a particular content-type increases during a given time period (and/or demand for a different content-type declines), then such fluctuations in demand will generally affect each participating server proportionately.

Noted that resources may be partitioned across multiple servers by naming convention (e.g., alphabetically) as well as having proportionally distributed content per server (e.g., by content-type). With this in mind, it should be further noted that logical partitioning of resources/content among intermediate servers is not limited to just alphabetic naming convention segmentation and/or content-type proportionality. It is contemplated that the partitioning may be based on

US 10,924,573 B2

13

various metrics such as, but not limited to, numeric hash values (e.g., MD5) associated with content/resource URLs, the size or file type of a resource, other identifiers associated with other various naming conventions (e.g., numerical, alphanumeric, proprietary, encrypted/mangled, etc.), the relative size and traffic behavior of various resources in a content provider's library or the content provider's library as a whole, and the like.

In step 720, the step of directing the client to a second server in the second tier is performed in response to a determination of which second server actually stores the first portion. In one example, the server that stores the first portion (or, for that matter, any portion that comprises the requested resource) is identified using a hash function (similar to the processing and functionality previously described).

In step 725, the second server that stores the first portion serves the resource to the client.

The foregoing description has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Obvious modifications or variations are possible in light of the above teachings. The embodiment or embodiments discussed were chosen and described to provide the best illustration of the principles of the invention and its practical application to thereby enable one of ordinary skill in the art to utilize the invention in various embodiments and with various modifications as are suited to the particular use contemplated. All such modifications and variations are within the scope of the invention as determined by the appended claims when interpreted in accordance with the breadth to which they are fairly and legally entitled.

We claim:

1. A method of content delivery in a content delivery network comprising:

receiving, at a first server of a first tier of servers of the content delivery network, a request from a requesting device for a resource available from the content delivery network;

accessing a popularity service associated with the content delivery network to determine a popularity designation associated with the requested resource;

requesting the resource from a second server of the content delivery network;

processing, at the first server of the first tier of servers, a redirect instruction from the second server of the content delivery network to obtain the resource from a content server of the content delivery network;

receiving an instruction to not cache the portion of the resource at the first server of the first tier of servers when the portion of the resource is obtained from the content server of the content delivery network; and providing the obtained resource to the requesting device.

2. The method as recited in claim 1 wherein the popularity designation associated with the requested resource indicates that the requested resource is not popular.

3. The method as recited in claim 1 wherein the popularity designation associated with the requested resource indicates that the requested resource is popular, the method further comprising: receiving an instruction to cache the resource at the first server of the first tier of servers when the resource is obtained from the content server of the content delivery network.

4. The method as recited in claim 1 wherein the second server is a metro mid-tier server of the content delivery network.

14

5. The method as recited in claim 1 wherein the second server is a peer server of the first tier of servers of the content delivery network.

6. The method as recited in claim 1 wherein the popularity designation associated with the requested resource comprises an indication whether a current popularity value for the resource exceeds a first predetermined popularity threshold.

7. The method as recited in claim 6 wherein the popularity service is associated with the second server of the content delivery network.

8. The method as recited in claim 6 wherein the popularity service is associated with the first server of the first tier of servers of the content delivery network.

9. The method as recited in claim 1 wherein the content delivery network further comprises a second tier of servers distinct from the first tier of servers, and wherein the second server is in the second tier of servers.

10. The method as recited in claim 3 further comprising: receiving, at the first server of the first tier of servers of the content delivery network, an additional request for the resource available from the content delivery network;

determining that the resource is cached at the first server of the first tier of servers; and

providing the cached resource from the first server of the first tier of servers in response to the additional request for the resource.

11. A content delivery network comprising:

a first tier of servers comprising a first plurality of servers, an edge server of the first tier of servers receiving a request from a requesting device in communication with the first server for a resource available from the content delivery network;

a second tier of servers comprising a second plurality of servers, a first server of the second plurality of servers receiving a request from the edge server for the content and, in response, transmitting a redirect instruction intended for the requesting device to the edge server to obtain the resource from a content server and transmitting an instruction to not cache the portion of the resource at the edge server when the portion of the resource is obtained from the content server; and

a popularity service tracking a popularity designation associated with the requested resource;

wherein the edge server processes the redirect instruction to obtain the content from the content server and provides the obtained content to the requesting device.

12. The content delivery network as recited in claim 11 wherein the popularity designation associated with the requested resource indicates that the requested resource is not popular.

13. The content delivery network as recited in claim 11 wherein the popularity designation associated with the requested resource indicates that the requested resource is popular and the first server of the second plurality of servers transmits an instruction to cache the resource at the content server of the first tier of servers when the resource is obtained from the content server.

14. The content delivery network as recited in claim 11 wherein the first server of the second plurality of servers is a metro mid-tier server of the content delivery network.

15. The content delivery network as recited in claim 11 wherein the first server of the second plurality of servers is a peer server of the first tier of servers of the content delivery network.

US 10,924,573 B2

15

16. The content delivery network as recited in claim 11 wherein the popularity designation associated with the requested resource comprises an indication whether a current popularity value for the resource exceeds a first predetermined popularity threshold.

17. The content delivery network as recited in claim 16 wherein the popularity service is associated with the first server of the second plurality of servers of the content delivery network.

18. The content delivery network as recited in claim 16 wherein the popularity service is associated with the content server of the first tier of servers of the content delivery network.

19. A method of content delivery in a content delivery network comprising:

receiving, at a first server of a first tier of servers of the content delivery network, a request from a requesting device for a resource available from the content delivery network;

16

accessing a popularity service associated with the first server of the first tier of the content delivery network to determine a popularity designation associated with the requested resource;

receiving an instruction to not cache the resource at the first server of the first tier of servers when the portion of the resource is obtained from the content server of the content delivery network;

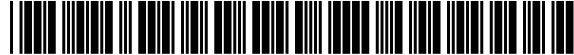
requesting the resource from a second server of the content delivery network based at least on the popularity designation associated with the requested resource; and

providing the obtained resource to the requesting device.

20. The method as recited in claim 19 wherein the second server of the content delivery network is a content server storing the requested content.

* * * * *

EXHIBIT E



US010057322B2

(12) **United States Patent**
Newton et al.

(10) **Patent No.: US 10,057,322 B2**

(45) **Date of Patent: Aug. 21, 2018**

(54) **NETWORK ADDRESS RESOLUTION**

(56) **References Cited**

(71) Applicant: **Level 3 Communications, LLC**,
Broomfield, CO (US)

U.S. PATENT DOCUMENTS

(72) Inventors: **Christopher Newton**, Westlake Village,
CA (US); **William R. Power**, Boulder,
CO (US)

6,185,598 B1 2/2001 Farber et al.
6,654,807 B2 11/2003 Farber et al.
7,054,935 B2 5/2006 Farber et al.
7,945,693 B2 5/2011 Farber et al.
7,949,779 B2 5/2011 Farber et al.
8,060,613 B2 11/2011 Farber et al.
8,250,211 B2 8/2012 Swildens et al.
8,281,035 B2 10/2012 Farber et al.
8,291,046 B2 10/2012 Farber et al.
8,296,396 B2 10/2012 Farber et al.
8,412,823 B1 * 4/2013 Richardson H04L 47/70
709/225
8,463,877 B1 * 6/2013 Richardson H04L 67/327
709/219

(73) Assignee: **Level 3 Communications, LLC**,
Broomfield, CO (US)

(*) Notice: Subject to any disclaimer, the term of this
patent is extended or adjusted under 35
U.S.C. 154(b) by 290 days.

8,468,245 B2 6/2013 Farber et al.
8,473,613 B2 6/2013 Farber et al.

(Continued)

(21) Appl. No.: **14/985,968**

(22) Filed: **Dec. 31, 2015**

(65) **Prior Publication Data**

OTHER PUBLICATIONS

US 2016/0191455 A1 Jun. 30, 2016

International Preliminary Report on Patentability dated Jul. 4, 2017,
Int'l Appl. No. PCT/US15/068284, Int'l Filing Date Dec. 31, 2015
9 pgs.

(Continued)

Related U.S. Application Data

(60) Provisional application No. 62/098,930, filed on Dec.
31, 2014.

Primary Examiner — Lashonda Jacobs

(51) **Int. Cl.**

H04L 29/08 (2006.01)

H04L 29/12 (2006.01)

G06F 17/30 (2006.01)

(52) **U.S. Cl.**

CPC **H04L 67/02** (2013.01); **G06F 17/30038**
(2013.01); **G06F 17/30876** (2013.01); **H04L**
61/1511 (2013.01); **H04L 61/3055** (2013.01)

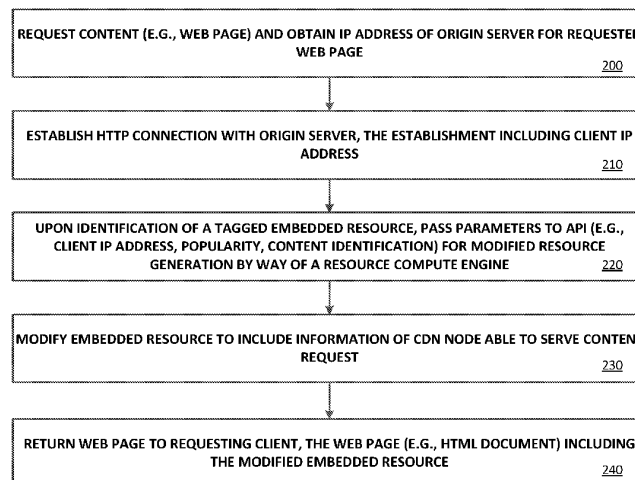
(58) **Field of Classification Search**

CPC H04L 67/02; H04L 67/1021
See application file for complete search history.

(57) **ABSTRACT**

A content delivery method, and related apparatus, that
involves the operations of receiving a request for a content
resource including at least one embedded resource with a
tag. Upon identification of the tag, using at least one delivery
parameter to generate a modified embedded resource.
Embedding the modified embedded resource in a content
resource, such as an HTML document. Delivering, such
as in response to request from a browser, the content
resource with the modified embedded resource.

15 Claims, 6 Drawing Sheets



US 10,057,322 B2

Page 2

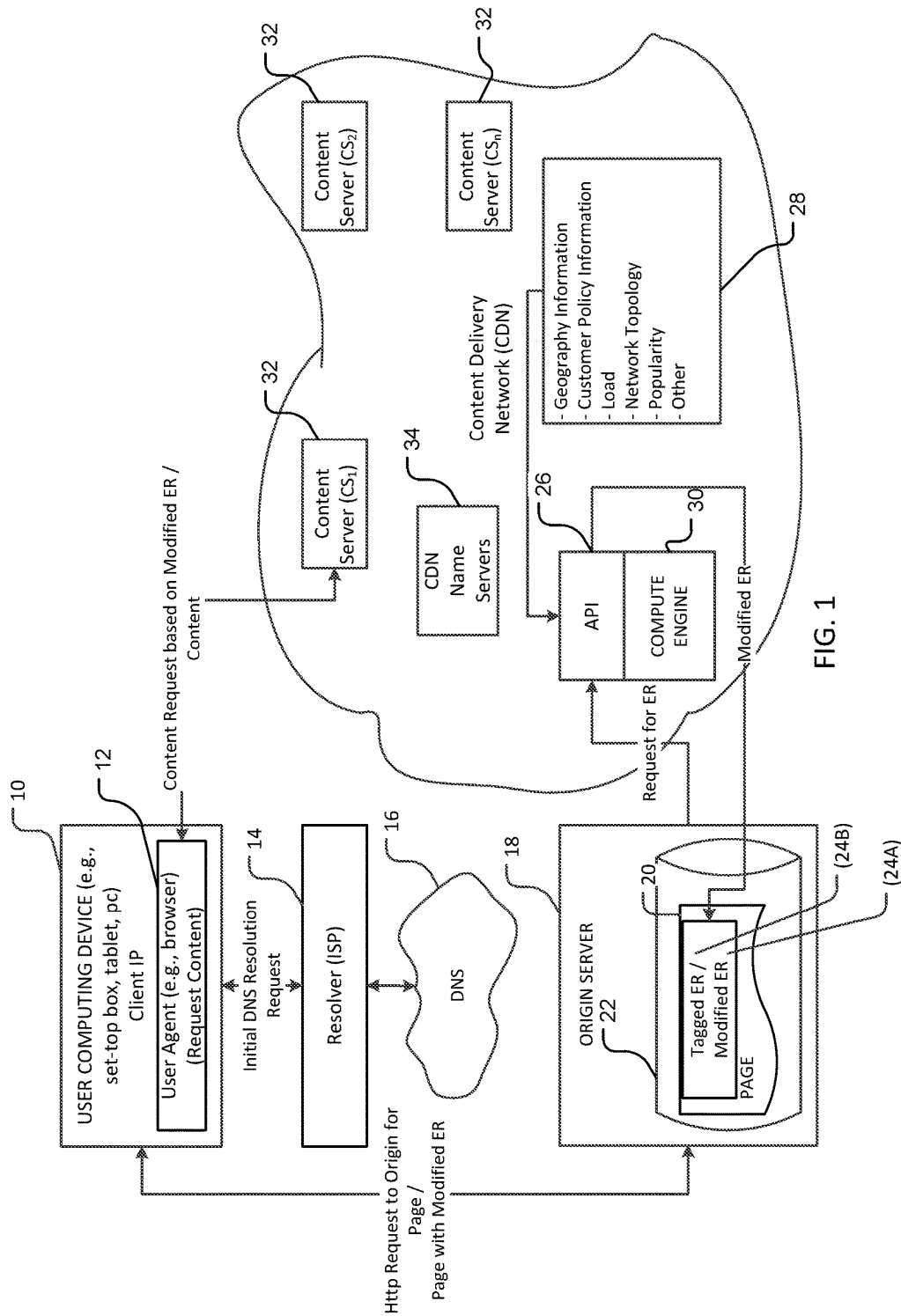
(56)	References Cited	2014/0053237 A1 * 2/2014 Hopen G06F 21/6218
	U.S. PATENT DOCUMENTS	2014/0059208 A1 * 2/2014 Yan H04L 43/0817
	8,478,903 B2 7/2013 Farber et al.	2014/0068005 A1 * 3/2014 Rosenfield G06F 17/30902
	8,572,208 B2 10/2013 Farber et al.	2014/0280479 A1 * 9/2014 Kazerani H04L 43/10
	8,572,210 B2 10/2013 Farber et al.	2014/0289319 A1 * 9/2014 Richardson H04L 67/2814
	8,683,076 B2 3/2014 Farber et al.	2014/0344425 A1 * 11/2014 Varney H04L 41/0813
	8,756,341 B1 * 6/2014 Richardson H04L 67/2814	2015/0222681 A1 * 8/2015 Basile H04L 65/60
	9,154,551 B1 * 10/2015 Watson H04L 67/1014	2017/0195447 A1 * 7/2017 Panagos H04L 67/2842
	9,191,458 B2 * 11/2015 Richardson H04L 67/2814	
	9,787,599 B2 * 10/2017 Richardson H04L 47/70	
	9,794,216 B2 * 10/2017 Richardson H04L 61/1511	
	9,800,539 B2 * 10/2017 Richardson H04L 61/1511	
	2003/0204529 A1 * 10/2003 Hertling G06F 17/30902	
	2007/0006155 A1 * 1/2007 Maennel G06F 11/3664	
	2008/0086574 A1 * 4/2008 Raciborski H04L 29/12066	
	2009/0254661 A1 * 10/2009 Fullagar H04N 7/17354	
	2012/0066360 A1 * 3/2012 Ghosh H04L 43/0864	
	2012/0198043 A1 * 8/2012 Hesketh H04L 29/1265	
	2013/0086358 A1 4/2013 Archer	
		OTHER PUBLICATIONS
		International Search Report dated Mar. 14, 2016, Int'l Appl. No. PCT/US15/068284, Int'l Filing Date Dec. 31, 2015; 3 pgs. Written Opinion of the International Searching Authority dated Mar. 14, 2016, Int'l Appl. No. PCT/US15/068284, Int'l Filing Date Dec. 31, 2015; 7 pgs. Singapore Written Opinion, dated Feb. 2, 2018, Application No. 11201705147U, filed Dec. 31, 2015; 6 pgs.
		* cited by examiner

U.S. Patent

Aug. 21, 2018

Sheet 1 of 6

US 10,057,322 B2



U.S. Patent

Aug. 21, 2018

Sheet 2 of 6

US 10,057,322 B2

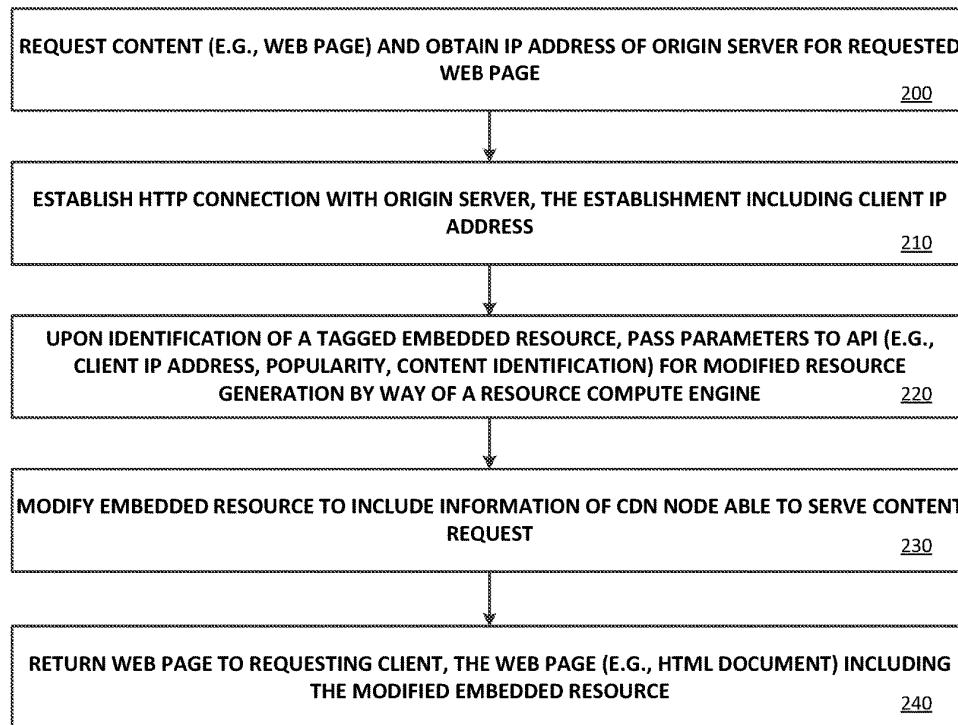


FIG. 2

U.S. Patent

Aug. 21, 2018

Sheet 3 of 6

US 10,057,322 B2

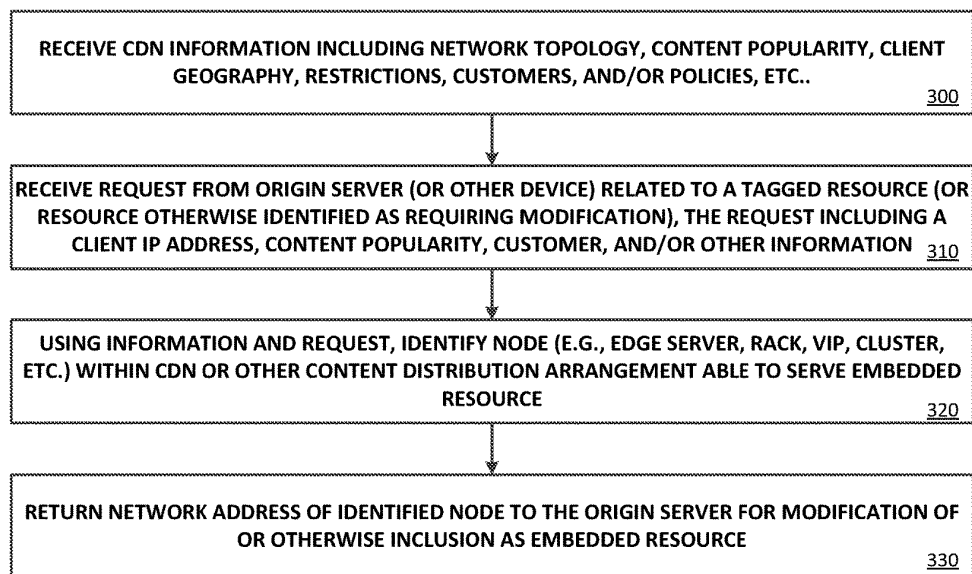


FIG. 3

U.S. Patent

Aug. 21, 2018

Sheet 4 of 6

US 10,057,322 B2

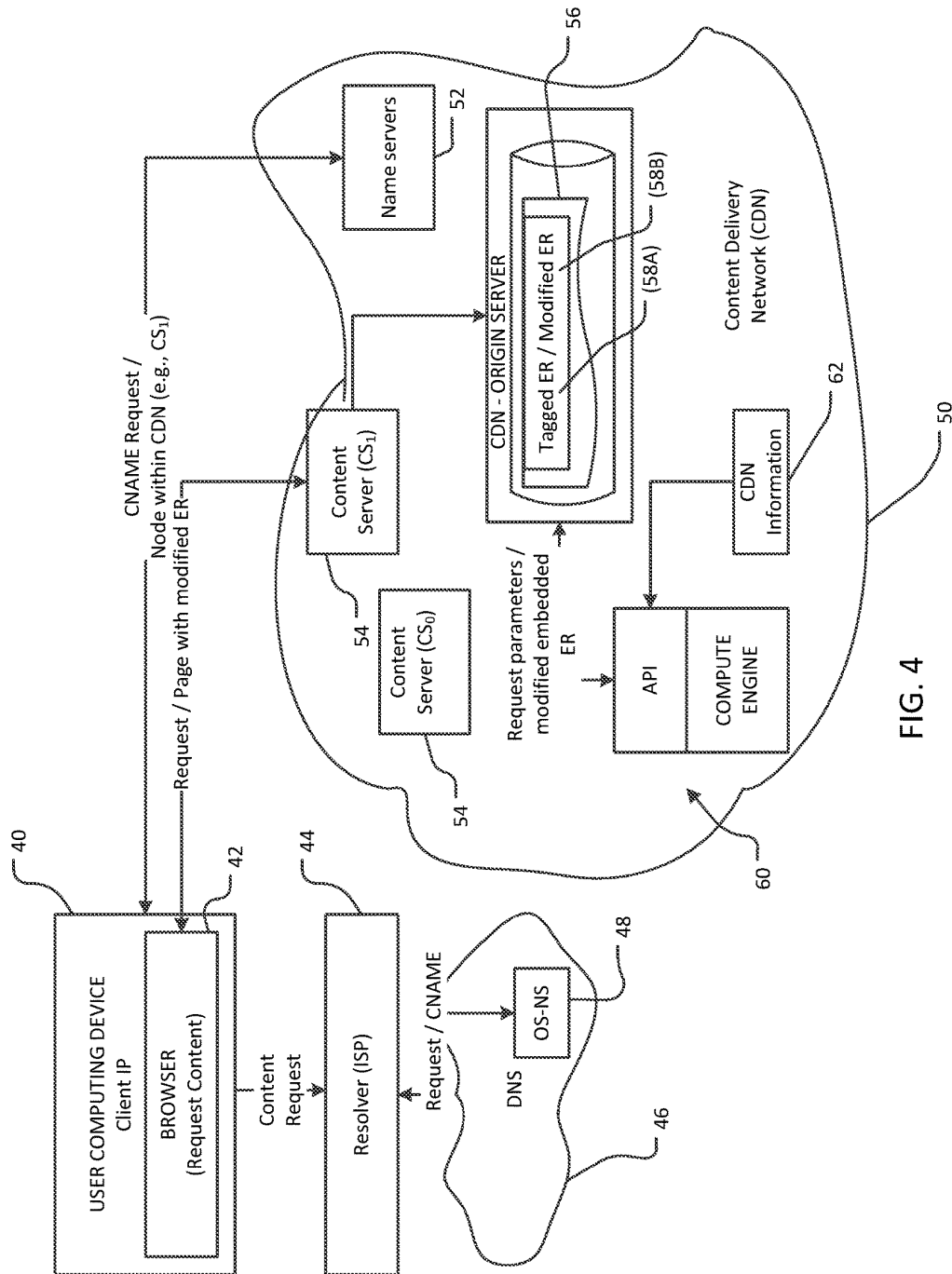


FIG. 4

U.S. Patent

Aug. 21, 2018

Sheet 5 of 6

US 10,057,322 B2

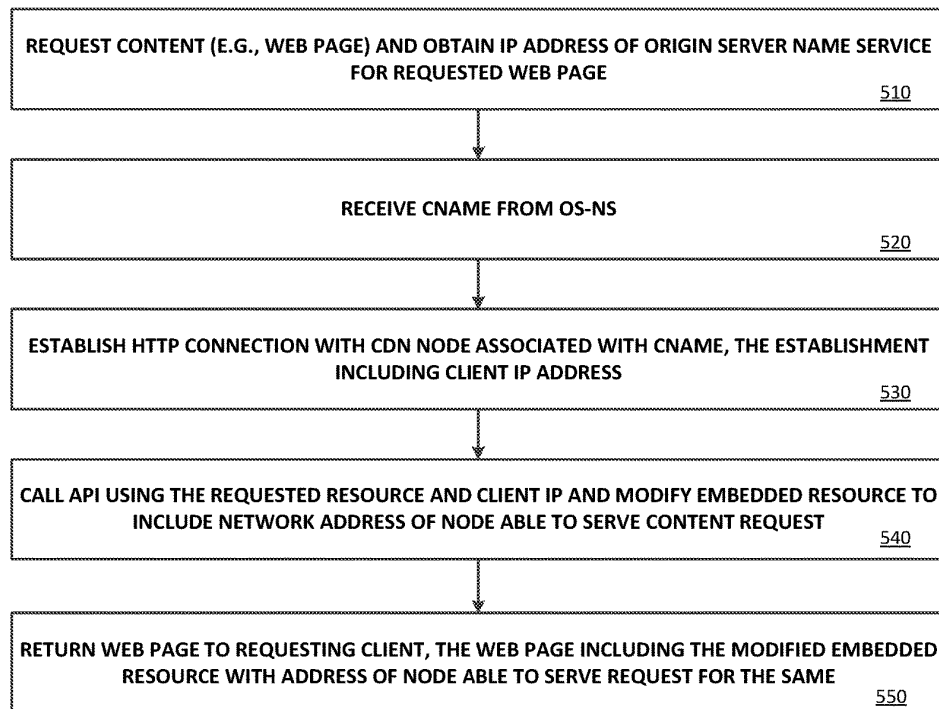


FIG. 5

U.S. Patent

Aug. 21, 2018

Sheet 6 of 6

US 10,057,322 B2

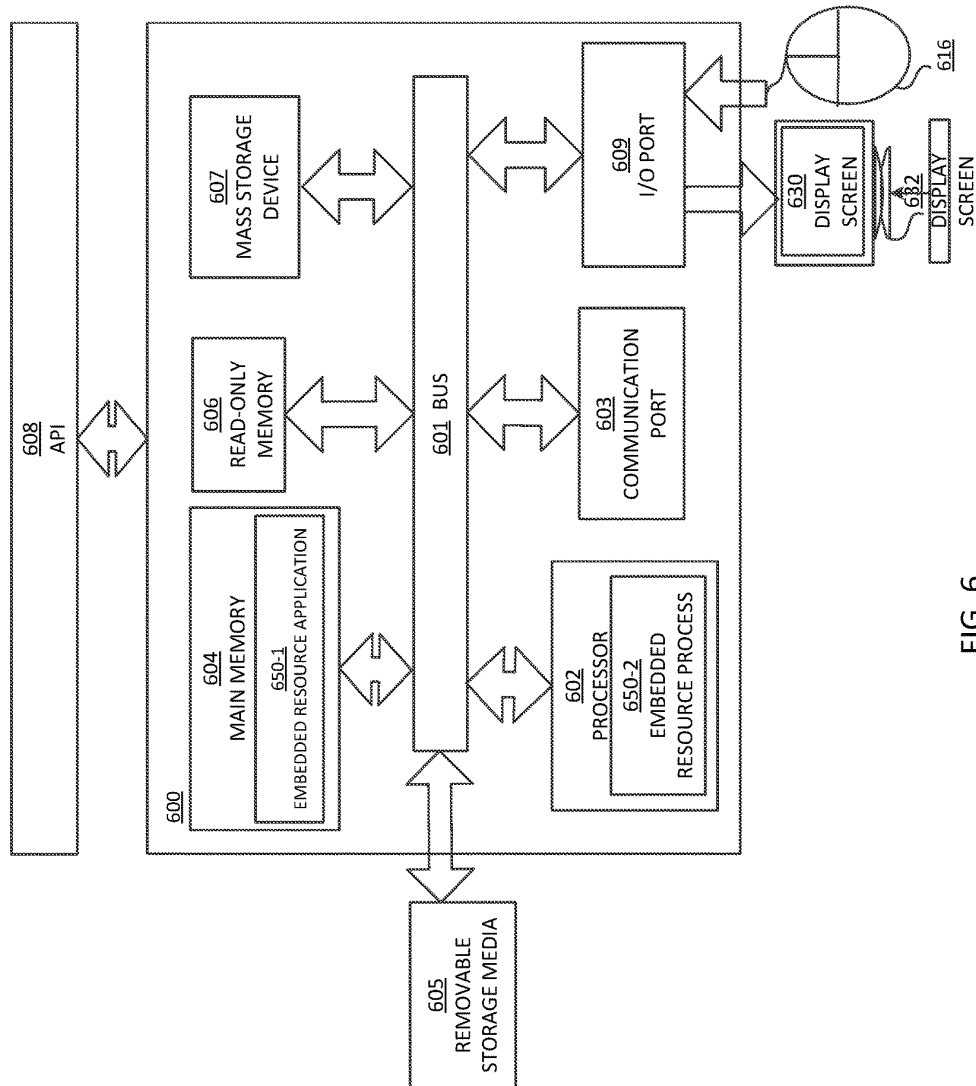


FIG. 6

US 10,057,322 B2

1 NETWORK ADDRESS RESOLUTION

RELATED APPLICATIONS

This application claims priority under 35 U.S.C. § 119 from U.S. provisional application No. 62/098,930 entitled “NETWORK ADDRESS RESOLUTION,” filed on Dec. 31, 2014, the entire contents of which are fully incorporated by reference herein for all purposes.

TECHNICAL FIELD

Aspects of the present disclosure relate to network address resolution, and particularly involve introduction of an application programming interface into the resolution stream to embed link resolution information within an HTML document or other content returned to a requesting client.

BACKGROUND

The Internet and the World Wide Web (the “Web”) have become ubiquitous. Thousands or even tens of thousands of content providers (publishers) now use the Internet (and, particularly, the Web) to provide all sorts of content to numerous clients all over the world. In order to offload the job of serving some or all of their content, many content providers now subscribe to content delivery networks (CDNs). Using a CDN, content can be served to clients from the CDN (e.g., from one or more servers in the CDN) instead of from the content provider’s server(s). In a caching CDN, content may also be cached on some or all of the CDN servers, either before being served or in response to specific requests for that content. Having content cached within edge servers of the CDN enhances the performance of the CDN because the content does not have to be retrieved from mid-tier servers origin servers or other locations, which are less efficient than edge servers in providing content.

Content is constantly being accessed from networks. For example, television shows and movies may now be accessed from any number of Web sites. Print newspapers have migrated to the Web and provide portals through which clients operating some form of computing device (e.g., PC, smart phone, or tablet), with a browser may access numerous forms of content, such as short video clips, articles, images, and audio tracks. Software updates and patches, once only provided on disc and mailed to recipients, are now routinely distributed to devices using only network connections. All of these forms of content may be distributed using CDNs.

It is with these observations in mind, among others, that various aspects of the present disclosure were conceived and developed.

SUMMARY

Embodiments of the present disclosure involve a content delivery method including the operations of receiving a request for a content resource including at least one embedded resource with a tag. Upon identification of the tag, using at least one delivery parameter to generate a modified embedded resource. And, delivering the content resource with the modified embedded resource.

Embodiments may also include a content delivery device comprising at least one processor in communication with a non-transitory computer readable medium including computer readable instructions for generating an application

programming interface (API). At the API, receiving an embedded resource and at least one parameter associated with the embedded resource. The instructions may further include generating a modified embedded resource providing information as to a location within a content delivery network for obtaining the embedded resource.

Embodiments may also include a non-transitory computer readable media comprising an object document, such as an HTML document, comprising at least one embedded resource, the at least one embedded resource including a tag that causes the embedded resource to be analyzed to construct a modified embedded resource to replace the embedded resource, the modified embedded resource causing a browser receiving the document with the modified embedded resource to initiate communication with a content delivery network to obtain content associated with the embedded resource.

Finally, embodiments may also include a content delivery method comprising initiating an HTTP connection with a server, the HTTP connection including a client IP address that is available to the server. Receiving an HTML document including at least one embedded resource, the embedded resource including a reference to a content delivery network whereby the reference to the content delivery network is influenced by the client IP address. And, using the reference, initiating a request directly to the content delivery network to obtain content associated with the at least one embedded resource.

These and other embodiments are discussed in more detail below.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects, features, and advantages of the present disclosure set forth herein should be apparent from the following description of particular embodiments of those inventive concepts, as illustrated in the accompanying drawings. Also, in the drawings the like reference characters may refer to the same parts throughout the different views. The drawings depict only typical embodiments of the present disclosure and, therefore, are not to be considered limiting in scope.

FIG. 1 is a network diagram and message flow related to generating an embedded resource for inclusion in a network asset, such as an HTML document, the embedded resource providing a connection to a node of a CDN bypassing DNS, in one example;

FIG. 2 is a flow diagram illustrating a method of identifying a tagged resource and thereafter providing a modified embedded resource by passing various parameters to an API for identification of a node of a CDN able to serve content associated with the embedded resource;

FIG. 3 is a flow diagram illustrating a method of generating a modified resource for embedding in a requested asset having a tagged resource;

FIG. 4 is a network diagram and message flow related to generating an embedded resource from within a CDN for inclusion in a network asset, such as an HTML document, the embedded resource providing a connection to a node of the CDN bypassing DNS, in one example;

FIG. 5 is a flow diagram illustrating a method of identifying a tagged resource and thereafter providing a modified embedded resource by passing various parameters to an API within a CDN, the API for identification of a node of a CDN able to serve content associated with the embedded resource;

US 10,057,322 B2

3

FIG. 6 is an example of a computing system that may implement various systems and methods discussed herein.

DETAILED DESCRIPTION

Aspects of the present disclosure involve systems and methods for selecting an edge server, rack, cluster or other device, which may be in a content delivery network (CDN), to serve content to a client. Aspects of the present disclosure involve an alternative to using conventional domain name system (DNS) servers for edge selection by instead calling an API or otherwise querying the CDN when constructing a uniform resource locator (URL) for the resource to be delivered from the CDN. The API call may include a name of the resource, a location of the client requesting the resource and/or an IP address for the client which may be used to determine the location of the client, and/or the popularity of the resource being requested (or embedded within a resource). The API or other computing engine associated therewith generates a target or list of targets (nodes capable of serving the content from the CDN), and provides one or more locations from which the content (or embedded content) may be served. In one particular form, the API may return one or more virtual IP addresses (VIPs) for one or more clustered edge servers. The API may also return the protocol to use for requesting the content. The API may also return a list of absolute URL's from which the content may be retrieved, and/or a list of hostnames and/or IP addresses that could be used to construct the URLs at the calling application.

The term "content" as used herein means any kind of data, in any form, regardless of its representation and regardless of what it represents. The term "content" may include, without limitation, static and/or dynamic images, text, audio content, including streamed audio, video content, including streamed video, web pages, computer programs, documents, files, and the like. Some content may be embedded in other content, e.g., using markup languages such as HTML and XML. Content includes content which is created or formed or composed specifically in response to a particular request. The term "resource" is sometimes used herein to refer to content.

FIG. 1 illustrates one example network configuration for practicing aspects of the present disclosure, while FIGS. 2 and 3 describe methods. The illustrated network would include many other components involved in servicing a request and providing content; however, the diagram here focuses on only some components involved in the system and one of ordinary skill will readily understand other conventional components within the network. With respect to content delivery, in this example a request originates from a computing device 10 operating a user agent 12, such as a browser. The computing device may be some form of computing device such as personal computer, a thin client, a tablet, a smart phone, a set-top box, a cable box, or the like, which is capable of running a browser and providing a user with the ability to request content over a network connection, or some other application suitable for interacting with a network and requesting content therefrom. The user device may be a consumer device deployed in a public network, may be an enterprise device deployed in a private network, or may be some other type of device. The network connection, in many instances, will be provided by an Internet service provider (ISP) operating one or more resolvers 14. A resolver (also referred to as a DNS resolver) is a client-side device used to resolve a network address (e.g., translate a domain name into an IP address) for a requested resource.

4

Generally speaking, the resolver contacts (e.g., queries) a DNS infrastructure 16, which may involve many different components, to resolve a network address for a resource (operation 200).

To illustrate aspects of the disclosure, consider when a user enters, into the browser Web page, an address `http://www.example.com`. The browser 12 contacts the resolver 14, which in turn requests the internet protocol (IP) address for the Web page from DNS 16. DNS returns the IP address of an origin server 18 capable of delivering a hypertext markup language (HTML) document 20 that will render the Web page in the browser. In order to obtain the HTML document, the client device establishes a TCP connection, which may encapsulate an HTTP protocol communication, with the origin server 18 (operation 210). An origin server is a device (or collection of devices) maintaining original copies of content for distribution to client's and/or distribution of copies of content to other nodes. While the present disclosure is illustrated in association with a connection to an origin server, the connection may instead be made with some other device designated in DNS to serve the requested resource. In this example, the origin server is connected with a database 22, which organizes and manages stored resources in some form of storage device (or devices), such as a storage appliance, that includes the HTML document 20 with an embedded resource 24. The HTML document is what the browser uses to display the Web page `http://www.example.com/`. In this example, the embedded resource 24A is a link to some form of content that the user can obtain by selecting the link when the Web page is displayed.

The embedded resource may include a tag indicating that a resource address compute engine 30 should be contacted via a call to an API 26 (operation 220). For instance, the HTML document may include an embedded resource of the form:

```
http://{CALLAPI:ADDRONLY}/path/resource.txt, or
${CALLAPI:F*ULLURL,path=$ENV{PATH},
pop=$POP}.
```

In both instances, the tag is "\$" which causes the origin server 18 to call the API 26 to retrieve the address (only) or the complete URL. Hence, the presence of the tagged resource 24A in the HTML document 20 at the origin servers causes the origin server to call the API, the result of which causes the HTML document returned to the requesting client to include a modified embedded resource 24B with an address or URL to the CDN whereby the content may be retrieved.

When the origin server receives the request (or at some time prior to receiving the request), the origin server calls an API in order to communicate with the compute engine 30 to format the address for the embedded resource. The presence of the tagged embedded resource prompts the API call. In this example, the address for the embedded resource is not static. Instead, an address that is optimally suited to serve the content to the requesting client is returned. So, in a simple example, the HTML document is initially published with a tagged embedded resource 24A that would identify the resource, and may also include other attributes such as popularity. If the user were in New York, which may be determined by including the requesting client's IP address or information, such as geographical information, based on the client IP address in the API call, and the CDN is capable of serving the requested content from both California and New York, it may not be optimal to serve the content from California due to any number of reasons including cost, latency, quality, optimal CDN use, and the like. In the

US 10,057,322 B2

5

present system, the API/compute engine may instead cause the creation of an embedded resource **24B** (also referred to as a modified embedded resource) that will direct the client to the New York node. While described herein as an API call (or calls) in conjunction with an associated compute engine, other applications may be used to compute and return the modified embedded resource. In some instances, only an API is referenced but it should be recognized that such reference includes not only the interface but also the associated compute routine, function, and the like for processing inputs to the API and generating a modified embedded resource. In any event, a modified embedded resource is generated and substituted for the resource, where the modified resource identified a node able to serve the content (operation **230**).

The API may obtain information used in determining the address from various sources. To begin, the request from the origin server may include information. Since the client has contacted the origin server over a TCP connection, the client IP address may be included in the request from the client. Accordingly, the origin server may transmit the client IP address to the API and/or the origin server may use the client IP to obtain or generate information **28** to share with the API. For example, the origin server may obtain geographic information for the client, based on the client IP address, or may transmit the IP address to the API, which may obtain geographic information for the client. The location of the client may be used in selecting an optimal node to serve a request for the embedded resource. Note, by embedding an IP address to serve the content directly, the client device may bypass DNS and instead connect directly to the source of the embedded resource. Regardless, the HTML document **20** with the modified imbedded resource **24B** is returned to the client device (operation **240**).

The origin server, or other devices associated with the customer managing the origin server, may also obtain popularity information for the content. For example, consider the case where <http://www.example.com/> is the home page for an on-line video provider and the embedded resource is for a recently released movie that did well in theaters. In such an example, when the movie is released by the video provider, it may initially be quite popular, and this popularity may be tracked and shared with the API (and its compute engine). In the same example, the customer may preset the popularity to "high," for example, based on its understanding that the initial release of the video will be highly popular among its subscribers (user base). The scale of popularity may be in any number of possible forms—1=low, 2=medium, 3=high, scale of 1-10 with 1 being unpopular and 10 being the most popular, etc. The customer defined popularity may be included in the tagged embedded resource of the HTML page as initially defined at the origin or may be provided as a separate input to the API. The scale may be based on the number of requests for the content, which may be based on discrete times (e.g., number of requests per day, per minute, per second, etc.). Finally, popularity may be predicted, and may also be regionalized. For example, a resource in the case of a news story may initially become popular in some particular region or at some particular time. Say, for example, an early morning national breaking news story may become actually popular on the East Coast of the United States. The customer may then predict its popularity on the West Coast and the system may automatically set it to a "high" value for all requests.

Besides client information, including geography, and popularity, the origin server may also pass other information such as policy information, political information, restric-

6

tions, etc. For example, the distribution of some content may be geographically limited—e.g., obtaining real-time sporting content may be limited in the geographic area of one or both teams participating in the event. Some content may have geopolitical distribution restrictions. This information may also be stored separately at the CDN or otherwise separately accessible by the CDN and passed to or otherwise used in generating the modified embedded resource. Regardless, referring to FIG. 3, the compute engine receives CDN information, as well as other information, it may use to identify a node, collection of nodes, VIP, or the like, capable of receiving the content (operation **300**).

Similar information may be provided to the API or accessible by the API from other sources. The CDN, for example, may maintain its own popularity information and may track the same, which may account for requests originating with other customers of the CDN. It is often the case that a CDN will have numerous customers, and handle some or all content delivery for those customers. Further, in many instances, customers will have Web properties with the same content. Accordingly, clients may contact any of those Web properties to obtain the content. Hence, the CDN has a larger view of the popularity of some content.

In any event, the API **26** provides the interface to a computing engine **30** that uses the information, passed to the engine through the API, to determine one or more nodes **32** in the CDN to serve the content upon receiving a request from the origin server (or from some other request source) related to a tagged resource, or resource otherwise identified for modification (operation **310**). The computing engine may take into account and use client IP, customer (through, for example a customer ID), popularity in some or all forms, the delivery protocol of the resource, and the various other forms of information in determining the node from which to serve the content (operation **320**). For example, the computing engine may use geographical information to pick the node (CS1) closest to the client to serve the content. The computing engine may further take into account network traffic, and if the closest node is too busy, then select another node (e.g., CS2) that is close but less busy. The computing engine may further embed an instruction, along with the IP address, for the node to cache a copy of the content should the content meet some popularity threshold, and likely be requested again. In some instances, for example, the node may not yet have the content, and upon receiving the request, have to request the content from an origin server, another node, or elsewhere. In such instances, the instruction would cause the node to cache and retain the content and be able to directly serve the next request.

The API returns an absolute URL from which the content may be retrieved, and/or a list of hostnames and/or IP addresses that could be used to construct the URLs at the calling application (operation **330**). Similarly, the origin server may pass the document to the API, and the API causes the modified embedded resource to be directly embedded and return the modified document to the origin server. Thus, when the client browser displays the Web page, for example, the modified embedded resource (e.g., link) may include an absolute URL for a node (e.g., edge server(s)) within the CDN that can deliver the content. When the user selects the link, the browser will directly contact the node for the content. In the case of a hostname, the embedded resource may include specific reference to the CDN such that name servers **34** of the CDN are contacted directly, bypassing any other DNS processing such that the CDN may perform any form of intelligent traffic management (ITM) or other forms of edge selection and return an IP address of a node within

US 10,057,322 B2

7

the CDN able to process the request. It may be the case that the hostname will also include a reference to a node capable of serving the request as determined by the API. In such a case, the hostname references a CDN location as is closely tailored to a location for serving the request but allows for some level of fail over, load balancing, or the like, relative to an absolute URL. In such a case, the hostname may be mapped to a VIP associated with various machines capable of serving the request.

In an alternative implementation, which may also involve an API and calling the same in order to resolve an optimal node to serve a request, the process of contacting the API and constructing the optimal network address is instead handled by the CDN and does not necessarily involve a customer origin server or other devices outside the CDN. While embodiments are discussed herein as involving an API, it should be noted that an API is a convenient and efficient way for disparate systems to uniformly and predictably pass information to the computing engine and for the computing engine to return information. However, other forms of communicating with and passing information to the computing engine are also possible. Moreover, the API may include the functionality of the engine and not necessarily be considered a discrete mechanism from the computing engine functions.

Referring now to FIGS. 4 and 5, in the same manner as the system and process of FIGS. 1-3, a client device 40 makes an initial request for a network resource using a browser 42, which request is received (operation 510). For example, the client requests <http://www.example.com>. The client's resolver 44 queries DNS 46, which returns a CNAME from an origin server-name service (OS-NS) 48 associated with the requested content (operation 520). A CNAME is a mechanism used in DNS to specify the name or address of another domain. In this case, the CNAME points to the domain of the CDN 50—e.g., www.example.com.cdn.net. In such a case, the client device (or other accessing device) will seamlessly be directed to contact the name server 52 or servers for the cdn.net domain. The name server will resolve the request and return an entry point to the CDN, e.g., a node 54, from which to obtain the page 56. Regardless, however, of how the client is directed to the CDN, like the example above, the request to the CDN from the client device may involve an HTTP connection and may include the IP address of the requesting device (client IP) and information as to the requested page (operation 530). The page (e.g., www.example.com) has one or more embedded resources, which may be tagged resources 58A as discussed above. Hence, the CDN node (e.g., an edge server or the CDN origin server) contacted for the page will recognize the presence of a tagged embedded resource 58A and contact the API/compute node 60 to return an embedded resource (referred to as the “modified embedded resource”) 58B to include in the page returned to the requesting client device 40 (operations 540 and 550).

The API/compute engine will have access to various types of information from various possible sources. For example, as discussed above, the API may be passed information and may receive information from other sources, which information can be used by the compute engine to generate an address for an embedded link within the page. The page and embedded link (e.g., link to resource) are returned to the requesting client. In one possible example, the embedded resource may include a VIP or a direct link to an edge server to deliver the resource. The compute engine may take into account the location of the client resolver, the location of the client (based on client IP), policy, popularity as tracked by

8

the CDN or included in the original embedded resource, geography restrictions, protocol, and other parameters, when generating the modified embedded resource 58B, which information may be stored, computed or otherwise accessed from a CDN information store 62. Accordingly, when the page 56 (e.g., the HTML document) is returned to the client device 40, and the modified embedded resource 58B is selected, the client device (e.g., browser) may directly contact the node as defined by the API/compute engine or otherwise contact the CDN to obtain the resource.

FIG. 6 is example schematic diagram of a computing system 600 implementing a CDN node compute engine (and API) 608 that may collect the various data discussed herein, receive requests from origin server, a CDN-OS, or otherwise and process the data and request to determine an optimal CDN node. The computing system for the application 608 includes a bus 601 (i.e., interconnect), at least one processor 602 or other compute element, at least one communication port 603, a main memory 604, a removable storage media 605, a read-only memory 606, and a mass storage device 607. Processor(s) 602 can be any known processor, such as, but not limited to, an Intel® Itanium® or Itanium 2® processor(s), AMD® Opteron® or Athlon MP® processor(s), or Motorola® lines of processors. Communication port 603 can be any of an RS-232 port for use with a modem based dial-up connection, a 10/100 Ethernet port, a Gigabit port using copper or fiber, or a USB port. Communication port(s) 603 may be chosen depending on a network such as a Local Area Network (LAN), a Wide Area Network (WAN), or any network to which the computer system 600 connects. The rendezvous application may be in communication with peripheral devices (e.g., display screen 630, input device 616 via Input/Output (I/O) port 609).

Main memory 604 can be Random Access Memory (RAM) or any other dynamic storage device(s) commonly known in the art. Read-only memory 606 can be any static storage device(s) such as Programmable Read-Only Memory (PROM) chips for storing static information such as instructions for processor 602. Mass storage device 607 can be used to store information and instructions. For example, hard disks such as the Adaptec® family of Small Computer Serial Interface (SCSI) drives, an optical disc, an array of disks such as Redundant Array of Independent Disks (RAID), such as the Adaptec® family of RAID drives, or any other mass storage devices, may be used.

Bus 601 communicatively couples processor(s) 602 with the other memory, storage and communications blocks. Bus 601 can be a PCI/PCI-X, SCSI, or Universal Serial Bus (USB) based system bus (or other) depending on the storage devices used. Removable storage media 605 can be any kind of external hard drives, thumb drives, Compact Disc-Read Only Memory (CD-ROM), Compact Disc-Re-Writable (CD-RW), Digital Video Disk-Read Only Memory (DVD-ROM), etc.

Embodiments herein may be provided as a computer program product, which may include a machine-readable medium having stored thereon instructions which may be used to program a computer (or other electronic devices) to perform a process. The machine-readable medium may include, but is not limited to, optical discs, CD-ROMs, magneto-optical disks, ROMs, RAMs, erasable programmable read-only memories (EPROMs), electrically erasable programmable read-only memories (EEPROMs), magnetic or optical cards, flash memory, or other type of media/machine-readable medium suitable for storing electronic instructions

US 10,057,322 B2

9

As shown, main memory 604 is encoded with an embedded resource application 650-1, which may include functionality described relative the API/compute engine discussed herein and that supports functionality as discussed above and as discussed further below. For example, in one embodiment, the application 650-1 may include or otherwise implement the various processes and/or instructions described herein. The application 650-1 (and/or other resources as described herein) can be embodied as software code such as data and/or logic instructions (e.g., code stored in the memory or on another computer readable medium such as a disk) that supports processing functionality according to different embodiments described herein. During operation of one embodiment, processor(s) 602 accesses main memory 604 via the use of bus 601 in order to launch, run, execute, interpret or otherwise perform the logic instructions of the 650-1.

The description above includes example systems, methods, techniques, instruction sequences, and/or computer program products that embody techniques of the present disclosure. However, it is understood that the described disclosure may be practiced without these specific details. In the present disclosure, the methods disclosed may be implemented as sets of instructions or software readable by a device. Further, it is understood that the specific order or hierarchy of steps in the methods disclosed are instances of example approaches. Based upon design preferences, it is understood that the specific order or hierarchy of steps in the method can be rearranged while remaining within the disclosed subject matter. The accompanying method claims present elements of the various steps in a sample order, and are not necessarily meant to be limited to the specific order or hierarchy presented.

The described disclosure may be provided as a computer program product, or software, that may include a machine-readable medium having stored thereon instructions, which may be used to program a computer system (or other electronic devices) to perform a process according to the present disclosure. A machine-readable medium includes any mechanism for storing information in a form (e.g., software, processing application) readable by a machine (e.g., a computer). The machine-readable medium may include, but is not limited to, magnetic storage medium, optical storage medium (e.g., CD-ROM); magneto-optical storage medium, read only memory (ROM); random access memory (RAM); erasable programmable memory (e.g., EPROM and EEPROM); flash memory; or other types of medium suitable for storing electronic instructions.

It is believed that the present disclosure and many of its attendant advantages should be understood by the foregoing description, and it should be apparent that various changes may be made in the form, construction and arrangement of the components without departing from the disclosed subject matter or without sacrificing all of its material advantages. The form described is merely explanatory, and it is the intention of the following claims to encompass and include such changes.

While the present disclosure has been described with reference to various embodiments, it should be understood that these embodiments are illustrative and that the scope of the disclosure is not limited to them. Many variations, modifications, additions, and improvements are possible. More generally, embodiments in accordance with the present disclosure have been described in the context of particular implementations. Functionality may be separated or combined in blocks differently in various embodiments of the disclosure or described with different terminology. These

10

and other variations, modifications, additions, and improvements may fall within the scope of the disclosure as defined in the claims that follow.

What is claimed is:

1. A content delivery method comprising:

receiving, by an origin system, a request from a client device for a content resource, the content resource including at least one embedded resource with a tag; upon identification of the tag, using at least one delivery parameter to obtain, by the origin system, a modified embedded resource from a content delivery network that is separate from the origin system; and

delivering, by the origin system, the content resource to the client device with the modified embedded resource, wherein the modified embedded resource includes a direct link to a node within the content delivery network that can deliver content associated with the embedded resource.

2. The content delivery method of claim 1 wherein the request is received over an http connection and the at least one delivery parameter includes an IP address of the client device, and further comprising:

passing information based on the client IP address to an application programming interface (API) of the content delivery network;

in association with the API, obtaining a parameter associated with the embedded resource;

identifying, based on the IP address and the parameter, location information for the node in the content delivery network that can serve the content associated with the embedded resource; and

generating the modified embedded resource to include the direct link to the node.

3. The content delivery method of claim 2 wherein the parameter is a popularity parameter.

4. The content delivery method of claim 2 wherein the parameter is a delivery protocol parameter.

5. The content delivery method of claim 2 wherein the information is the IP address.

6. The content delivery method of claim 2 wherein the information is geographic information.

7. The content delivery method of claim 2 wherein the location information includes at least one of an IP address, a virtual IP address, one or more absolute uniform resource locators.

8. The content delivery method of claim 1 wherein the tag is a unique tag.

9. The content delivery method of claim 1 wherein the request is over an http connection and the at least one delivery parameter is a location identifier for the client device, the method further comprising:

passing, by the origin system, the location identifier to an application programming interface (API) of the content delivery network to generate the modified embedded resource relating to the node.

10. The content delivery method of claim 1 wherein the content resource is a hypertext markup language (HTML) document and the embedded resource is a link.

11. A content delivery network comprising:

at least one processor in communication with a non-transitory computer readable medium including computer readable instructions for:

providing an application programming interface (API); at the API, receiving, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource;

US 10,057,322 B2

11

generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;
delivering, by the content delivery network, the modified embedded resource to the origin system;
receiving, at the node, a request from a client device to obtain the content; and
providing the content to the client device.

12. The content delivery network of claim **11** wherein the computer readable instructions are further for replacing the embedded resource with the modified embedded resource in a resource.

13. The content delivery network of claim **11** wherein the resource is an HTML document and the modified embedded resource is a link.

14. The content delivery network of claim **11** wherein the modified embedded resource includes at least one of an IP address to facilitate communication with the node to obtain the content associated with the embedded resource, a virtual IP address to facilitate communication with the node to

12

obtain the content associated with the embedded resource, and one or more absolute uniform resource locators to facilitate communication with the node to obtain the content associated with the embedded resource.

15. A method comprising:

providing, at a content delivery network, an application programming interface (API);

at the API, receiving, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource;

generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;

delivering, by the content delivery network, the modified embedded resource to the origin system;

receiving, at the node, a request from a client device to obtain the content; and

providing the content to the client device.

* * * * *

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Exhibit F

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

U.S. Patent No. 8,478,903

Sandpiper CDN

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
28. A method, in a content delivery system operative in a computer network for delivering content to client machines, the computer network comprising a plurality of origin servers, each of said origin servers having resources associated therewith, and the content delivery system comprising at least one shared repeater server operable to replicate resources associated with the plurality of origin servers, the method comprising:	Google uses and provides content delivery network(s) (“Google CDN”), including Google’s internal services and services Google offers to third parties that are used to provide content, such as webpages and/or video streams, over a network. Google CDN includes services such as Cloud CDN and Media CDN, for example, along with Google’s own use of technologies to provide content in a network. As one example of Google CDN, Cloud CDN is a content delivery system operative in a computer network including a plurality of origin servers, where the origin servers have resources associated therewith, such as resources associated with a Cloud CDN customer. Cloud CDN uses edge caches and replicates resources associated with origin servers in a system for delivering content to end users (for example end users of client machines). https://cloud.google.com/cdn?hl=en https://cloud.google.com/cdn/docs/overview As another example of Google CDN, Media CDN is a content delivery system operative in a computer network including a plurality of origin servers, where the origin servers have resources associated therewith, such as resources associated with Media CDN customers.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	Media CDN uses edge caches and replicates resources associated with origin servers to deliver content to end users. Media CDN overview  Send feedback Media CDN is Google Cloud's media delivery solution. Media CDN complements Cloud CDN, which is Google Cloud's web acceleration solution. Media CDN is optimized for high-throughput egress workloads, such as streaming video and large file downloads. An edge cache is typically server infrastructure that stores content closer to end users, located within points of presence (PoPs) or partner ISPs. Media CDN uses Google's global edge-caching infrastructure to serve your content as close to your users as possible. By using Google's infrastructure to serve content, you can reduce load on your origin infrastructure. Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure. You can control how content is cached for each URI you serve in a <i>route</i>. Using a route lets you optimize behavior based on the type of content, client attributes, and your freshness requirements for each route you define with Media CDN. For guidance on which CDN product to use, see Choose a CDN product. You can use both products if they both meet your needs. https://cloud.google.com/media-cdn/docs/overview
associating the at least one repeater server with a first alias name, wherein requests for a first resource located on a first origin server are directed, based at least in part on said first alias name, to the at least one repeater server for delivery of the first resource from said at least one repeater server;	Google associates at least one repeater server with an alias name. For example, Google CDN includes “edge caches” associated with alias names. On information and belief, Google CDN directs requests for a first resource on an origin server based at least in part on the alias name. Google CDN uses Google’s “edge-caching infrastructure” to serve content to end users, including content requested based on a name associated with a Google CDN customer.

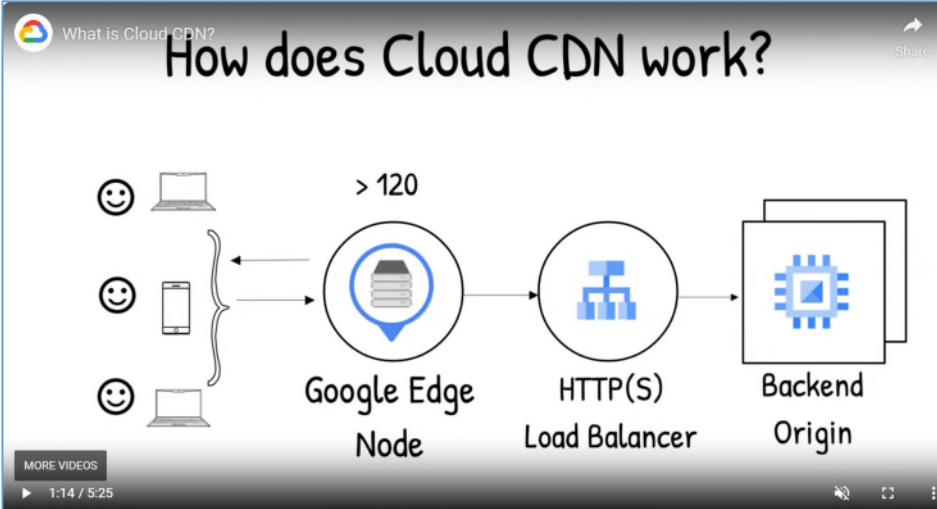
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	Media CDN overview  Send feedback Media CDN is Google Cloud's media delivery solution. Media CDN complements Cloud CDN, which is Google Cloud's web acceleration solution. Media CDN is optimized for high-throughput egress workloads, such as streaming video and large file downloads. An edge cache is typically server infrastructure that stores content closer to end users, located within points of presence (PoPs) or partner ISPs. Media CDN uses Google's global edge-caching infrastructure to serve your content as close to your users as possible. By using Google's infrastructure to serve content, you can reduce load on your origin infrastructure. Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure. You can control how content is cached for each URI you serve in a <i>route</i>. Using a route lets you optimize behavior based on the type of content, client attributes, and your freshness requirements for each route you define with Media CDN. For guidance on which CDN product to use, see Choose a CDN product. You can use both products if they both meet your needs. https://cloud.google.com/media-cdn/docs/overview Cloud CDN also uses repeater edge servers to cache and serve content from edge locations. CDNs are especially beneficial for content driven web applications. Since CDNs reduce network latency, they offer a better user experience when using content-driven applications, especially those with large media files. CDNs provide scalable infrastructure and can distribute requests across various edge servers. Therefore, users can access the content from another server if one goes down. They usually offer security features such as DDoS protection, WAF services, and SSL termination as well. Google Cloud CDN is designed to reduce the latency of web applications by caching and serving content from strategically located edge locations. https://developers.google.com/solutions/content-driven/hosting/cdn

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	 https://cloud.google.com/cdn
associating the at least one repeater server with a second alias name, wherein requests for a second resource located on a second origin server are directed, based at least in part on said second alias name, to the at least one repeater server for delivery of the second resource from said at least one repeater server, wherein the second origin server is distinct from the first origin server;	Google associates the repeater server with a second alias name, where requests for a second resource on a second origin server are directed based on the second alias name, where the first and second origin servers are distinct. For example, Media CDN has distinct origin servers associated with various, multiple customers as distinct HTTP endpoints, such as servers hosted by Cloud Storage. Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure. https://cloud.google.com/media-cdn/docs/overview#certificate-support As another example, Cloud CDN responds to user requests for content from various, multiple Cloud CDN customers, using cached content that is a copy of content stored on origin servers.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN												
	How Cloud CDN works When a user requests content from an external Application Load Balancer, the request arrives at a GFE that is at the edge of Google's network as close as possible to the user. If the load balancer's URL map routes traffic to a backend service or backend bucket that has Cloud CDN configured, the GFE uses Cloud CDN. Cache hits and cache misses A cache is a group of servers that stores and manages content so that future requests for that content can be served faster. The cached content is a copy of cacheable content that is stored on origin servers. If the GFE looks in the Cloud CDN cache and finds a cached response to the user's request, the GFE sends the cached response to the user. This is called a <i>cache hit</i>. When a cache hit occurs, the GFE looks up the content by its <i>cache key</i> and responds directly to the user, shortening the round-trip time and saving the origin server from having to process the request. https://cloud.google.com/cdn/docs/overview The following example shows a simple scenario that might apply if you are just getting started with Cloud CDN. Suppose you have the following Cloud CDN usage pattern in a given month. <table><tr><th>Pricing category</th><th>Type of usage</th><th>Amount</th></tr><tr><td>Cache data transfer out</td><td>Cache data transfer out in North America</td><td>500 GiB</td></tr><tr><td>Cache fill</td><td>Data transfer to Cloud CDN caches</td><td>25 GiB</td></tr><tr><td>Cache lookup requests</td><td>GET and HEAD HTTP methods</td><td>5,000,000 operations</td></tr></table> https://cloud.google.com/cdn/pricing#pricing-notes	Pricing category	Type of usage	Amount	Cache data transfer out	Cache data transfer out in North America	500 GiB	Cache fill	Data transfer to Cloud CDN caches	25 GiB	Cache lookup requests	GET and HEAD HTTP methods	5,000,000 operations
Pricing category	Type of usage	Amount											
Cache data transfer out	Cache data transfer out in North America	500 GiB											
Cache fill	Data transfer to Cloud CDN caches	25 GiB											
Cache lookup requests	GET and HEAD HTTP methods	5,000,000 operations											

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
providing a table listing origin servers having content located thereon, wherein said content is authorized for delivery to client machines via the at least one shared repeater server, and wherein the origin servers comprise the first origin server and the second origin server; and	Google provides logical tables of origin servers with content. For example, Media CDN lists the HTTP endpoints and provides routing to origins. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. https://cloud.google.com/media-cdn/docs/origins#origin-requirements Advanced routing features  Media CDN provides advanced HTTP routing capabilities that let you map traffic to specific edge configurations and origins at a fine-grained level. For more information, see Advanced routing. https://cloud.google.com/media-cdn/docs/overview#certificate-support As another example, Cloud CDN looks up content to respond to an end user.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	How Cloud CDN works When a user requests content from an external Application Load Balancer, the request arrives at a GFE that is at the edge of Google's network as close as possible to the user. If the load balancer's URL map routes traffic to a backend service or backend bucket that has Cloud CDN configured, the GFE uses Cloud CDN. Cache hits and cache misses A cache is a group of servers that stores and manages content so that future requests for that content can be served faster. The cached content is a copy of cacheable content that is stored on origin servers. If the GFE looks in the Cloud CDN cache and finds a cached response to the user's request, the GFE sends the cached response to the user. This is called a <i>cache hit</i>. When a cache hit occurs, the GFE looks up the content by its <i>cache key</i> and responds directly to the user, shortening the round-trip time and saving the origin server from having to process the request. https://cloud.google.com/cdn/docs/overview
wherein the at least one repeater server is further constructed and adapted to analyze, using the table, an alias name received with a client request for a particular resource to determine an origin server associated with the particular resource.	As stated above, Google performs advanced routing to origins based on its list of endpoints. Advanced routing features ➡ Media CDN provides advanced HTTP routing capabilities that let you map traffic to specific edge configurations and origins at a fine-grained level. For more information, see Advanced routing. https://cloud.google.com/media-cdn/docs/overview#certificate-support

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	As another example, Cloud CDN uses an alias name associated with a request for a resource to determine an origin server associated with the resource. Google explains Cloud CDN uses Google’s global edge network to serve content from Google’s CDN customers’ “websites and applications,” for example. Cloud CDN (Content Delivery Network) uses Google’s global edge network to serve content closer to users, which accelerates your websites and applications. https://cloud.google.com/cdn/docs/overview
32. The method as recited in claim 28, wherein the resources comprise an HTTP-based resource.	Google provides resources that are HTTP-based resources. For example, as shown below, Media CDN provides HTTP-based resources. Network protocol support Media CDN supports HTTP/3, HTTP/2, and HTTP/1.1 connections from clients. Media CDN supports both ALPN (Application Layer Protocol Negotiation) as well as the <code>Alt-Svc</code> (alternative service) HTTP response header for advertising protocol support. https://cloud.google.com/media-cdn/docs/client-connectivity
37. A method, in a content delivery system operative in a computer network for delivering content to client machines and comprising at least one shared repeater server operable to replicate resources stored on a plurality of origin servers, the method comprising:	Google uses and provides content delivery network(s), including Google’s internal services and services Google offers to third parties that are used to provide content, such as webpages and/or video streams, over a network. Google CDN includes services such as Cloud CDN and Media CDN, for example, along with Google’s own use of technologies to provide content in a network. As one example of Google CDN, Cloud CDN is a content delivery system operative in a computer network including a plurality of origin servers, where the origin servers have resources associated therewith, such as resources associated with a Cloud CDN customer. Cloud CDN uses edge caches and replicates resources associated with origin servers in a system for delivering content to end users (for example end users of client machines).

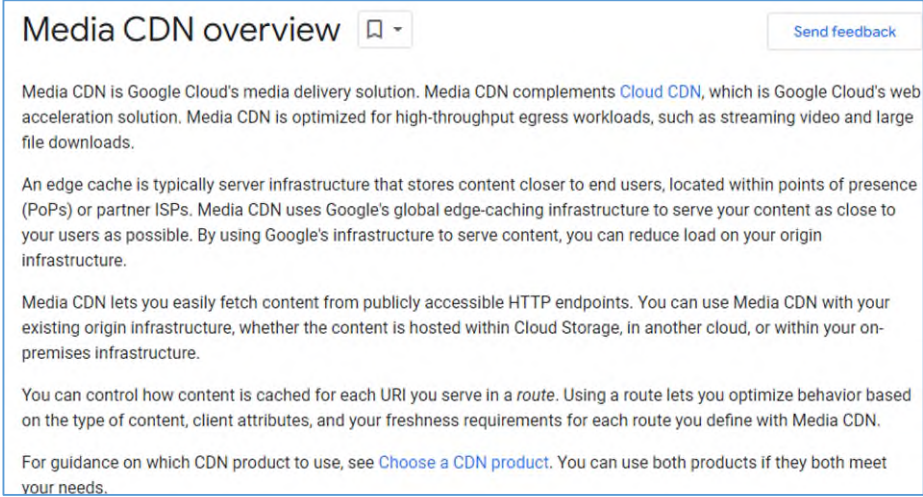
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	Cloud CDN and Media CDN Leverage Google's decade of experience delivering content Google's content delivery networks—Cloud CDN and Media CDN—scale to bring content closer to a global audience. https://cloud.google.com/cdn?hl=en Cache hits and cache misses A cache is a group of servers that stores and manages content so that future requests for that content can be served faster. The cached content is a copy of cacheable content that is stored on origin servers. https://cloud.google.com/cdn/docs/overview As another example of Google CDN, Media CDN is a content delivery system operative in a computer network including a plurality of origin servers, where the origin servers have resources associated therewith, such as resources associated with Media CDN customers. Media CDN uses edge caches and replicates resources associated with origin servers to deliver content to end users.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	 https://cloud.google.com/media-cdn/docs/overview
associating a plurality of alias names with the at least one shared repeater server, each of said plurality of alias names being associated with an origin server, wherein a first alias name of said plurality of alias names is associated with a first origin server, and at least a second alias name of said plurality of alias names is associated with a second origin server distinct from said first origin server,	Google associates alias names with a repeater server. For example, Google CDN includes “edge caches” associated with alias names. On information and belief, Google CDN includes alias names associated with distinct origin servers. Google CDN uses Google’s “edge-caching infrastructure” to serve content to end users, including content requested based on a name associated with a Google CDN customer, where each CDN customer has a distinct origin server associated with its alias name, for example.

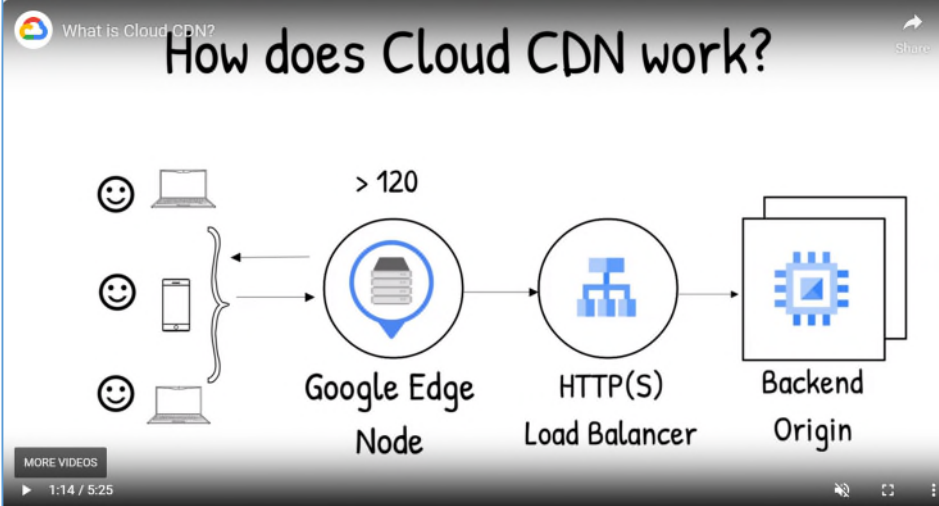
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	Media CDN overview  Send feedback Media CDN is Google Cloud's media delivery solution. Media CDN complements Cloud CDN, which is Google Cloud's web acceleration solution. Media CDN is optimized for high-throughput egress workloads, such as streaming video and large file downloads. An edge cache is typically server infrastructure that stores content closer to end users, located within points of presence (PoPs) or partner ISPs. Media CDN uses Google's global edge-caching infrastructure to serve your content as close to your users as possible. By using Google's infrastructure to serve content, you can reduce load on your origin infrastructure. Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure. You can control how content is cached for each URI you serve in a <i>route</i>. Using a route lets you optimize behavior based on the type of content, client attributes, and your freshness requirements for each route you define with Media CDN. For guidance on which CDN product to use, see Choose a CDN product. You can use both products if they both meet your needs. https://cloud.google.com/media-cdn/docs/overview Cloud CDN also uses repeater edge servers to cache and serve content from edge locations. CDNs are especially beneficial for content driven web applications. Since CDNs reduce network latency, they offer a better user experience when using content-driven applications, especially those with large media files. CDNs provide scalable infrastructure and can distribute requests across various edge servers. Therefore, users can access the content from another server if one goes down. They usually offer security features such as DDoS protection, WAF services, and SSL termination as well. Google Cloud CDN is designed to reduce the latency of web applications by caching and serving content from strategically located edge locations. https://developers.google.com/solutions/content-driven/hosting/cdn

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	 https://cloud.google.com/cdn
providing at least one table associating alias names with origin servers having content located thereon, wherein said content is authorized for delivery to client machines via the at least one shared repeater server, wherein the origin servers comprise the first origin server and the second origin server;	Google provides logical tables of origin servers with content. For example, Media CDN lists the HTTP endpoints and provides routing to origins. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. https://cloud.google.com/media-cdn/docs/origins#origin-requirements

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
	Advanced routing features  Media CDN provides advanced HTTP routing capabilities that let you map traffic to specific edge configurations and origins at a fine-grained level. For more information, see Advanced routing. https://cloud.google.com/media-cdn/docs/overview#certificate-support As another example, Cloud CDN looks up content to respond to an end user. How Cloud CDN works When a user requests content from an external Application Load Balancer, the request arrives at a GFE that is at the edge of Google's network as close as possible to the user. If the load balancer's URL map routes traffic to a backend service or backend bucket that has Cloud CDN configured, the GFE uses Cloud CDN. Cache hits and cache misses A cache is a group of servers that stores and manages content so that future requests for that content can be served faster. The cached content is a copy of cacheable content that is stored on origin servers. If the GFE looks in the Cloud CDN cache and finds a cached response to the user's request, the GFE sends the cached response to the user. This is called a <i>cache hit</i>. When a cache hit occurs, the GFE looks up the content by its cache key and responds directly to the user, shortening the round-trip time and saving the origin server from having to process the request. https://cloud.google.com/cdn/docs/overview

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
wherein requests for a first resource originating on the first origin server are directed, based at least in part on said first alias name, to the at least one shared repeater server for delivery of the first resource from said at least one repeater server; and	Google includes requests for a resource originating on an origin server to a repeater server for delivery of the resource. As stated above, Media CDN performs advanced routing to origins based on its list of endpoints. Advanced routing features  Media CDN provides advanced HTTP routing capabilities that let you map traffic to specific edge configurations and origins at a fine-grained level. For more information, see Advanced routing. https://cloud.google.com/media-cdn/docs/overview#certificate-support As another example, Cloud CDN uses an alias name associated with a request for a resource to determine an origin server associated with the resource. Google explains Cloud CDN uses Google’s global edge network to serve content from Google’s CDN customers’ “websites and applications,” for example. Cloud CDN (Content Delivery Network) uses Google’s global edge network to serve content closer to users, which accelerates your websites and applications. https://cloud.google.com/cdn/docs/overview
wherein requests for a second resource originating on the second origin server are directed, based at least in part on said second alias name, to the at least one shared repeater server for delivery of the second resource from said at least one repeater server; and	Google includes requests for a second resource originating on another origin server directed to the repeater server for delivery of the second resource, when an end user requests content from a second Google CDN customer that is cached on Google’s edge infrastructure, as discussed above.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,478,903	Google CDN
wherein the at least one repeater server uses the at least one table and an alias name received with a client request for a resource to determine an origin server associated with the requested resource.	On information and belief, Google includes a repeater server using the table and alias name received with a request to determine an origin server. For example, Google's CDN includes servers in the edge infrastructure that cache content for multiple Google CDN customers and which use a table to determine an origin server for a CDN customer.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Exhibit G

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

U.S. Patent No. 8,645,517

Sandpiper CDN

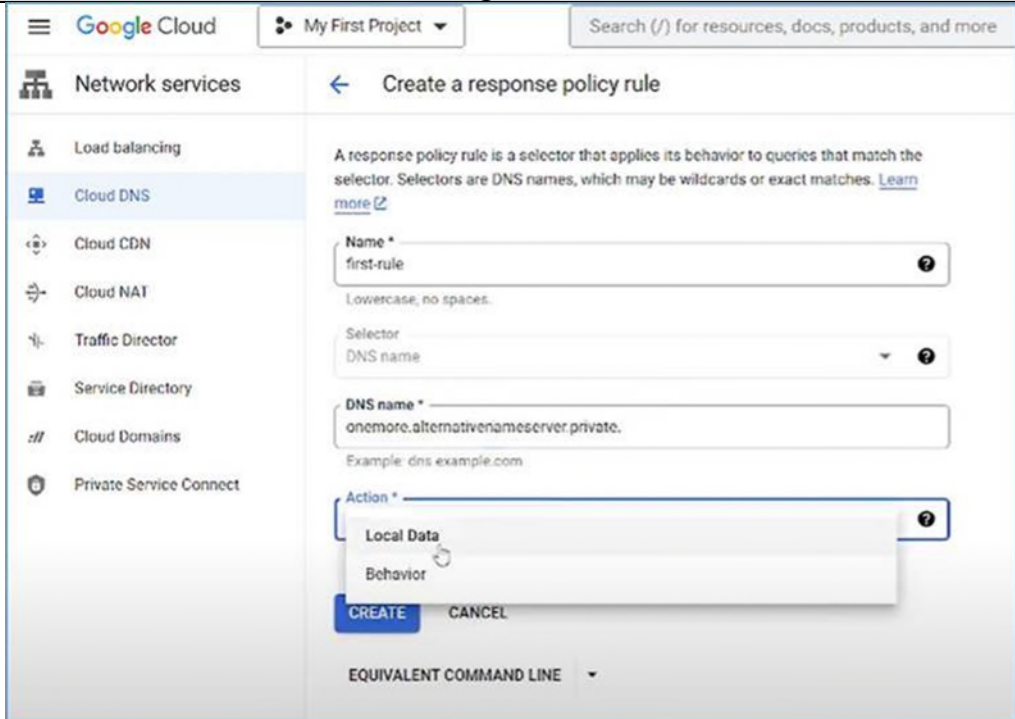
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
1. A method, operable in a framework in which an adaptive traffic control name server network implements policy-based traffic direction, the name server network comprising at least one domain name server comprising hardware in combination with software and constructed and adapted to provide adaptive policy-based domain name service, the method comprising:	Google uses and provides content delivery network(s) (“Google CDN”), including Google’s internal services and services Google offers to third parties that are used to provide content, such as webpages and/or video streams, over a network. Google CDN includes services such as Google Cloud DNS, for example, along with Google’s own use of technologies to provide content in a network. As one example, Google Cloud/Cloud DNS provides a method in which a name server network implements policy-based traffic direction, including a domain name server constructed to provide adaptive policy-based domain name service. https://cloud.google.com/dns/docs/policies-overview
A) providing a graphical user interface (GUI);	Google provides a graphical user interface (GUI), for example as shown below as part of the interface labeled “Create a response policy rule.”

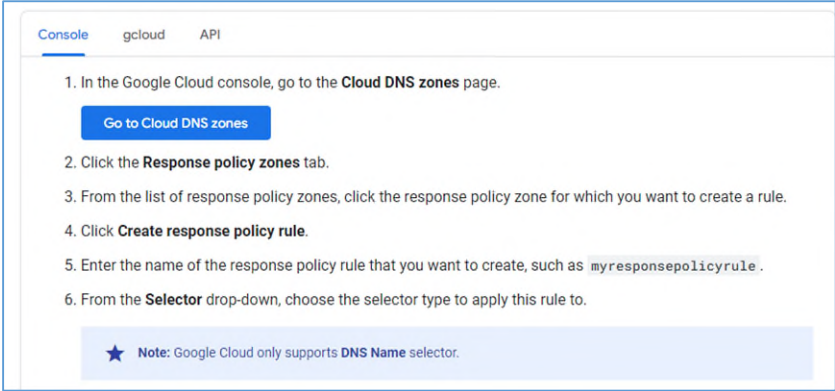
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	 https://www.youtube.com/watch?v=Z-5PcJ_y4Qg at 14:23.
(B) using said GUI to obtain at least one policy for direction of network traffic, wherein the GUI supports the setting of said at least one policy using a decision tree representing rules; and	Google uses a GUI to obtain at least one policy for direction of network traffic, including supporting the setting of a policy using a decision tree representing rules. For example, Google Cloud uses a GUI to obtain policies for directing traffic to certain DNS names, as shown above, and the hierarchy of drop-down menus used by Google support setting of a policy using a tree representing rules. As shown below, in the GUI for “Creat[ing] a response policy rule,” the exemplary “first rule” for a DNS name can be selected to have an “Action” of “Local Data,” which can include multiple “items” such as sets of IP addresses (e.g., for IPv4, IPv6, etc.).

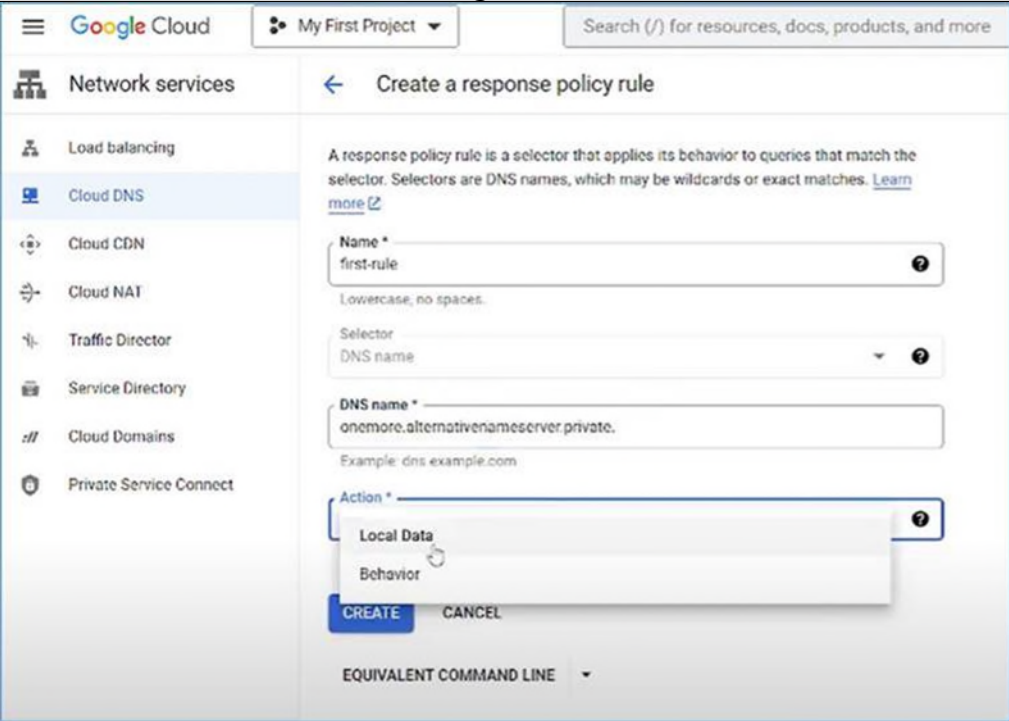
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	 The screenshot shows the Google Cloud Console interface with tabs for Console, gcloud, and API. A list of six steps is provided for creating a response policy rule. Step 1: In the Google Cloud console, go to the Cloud DNS zones page. Step 2: Click the Response policy zones tab. Step 3: From the list of response policy zones, click the response policy zone for which you want to create a rule. Step 4: Click Create response policy rule. Step 5: Enter the name of the response policy rule that you want to create, such as myresponsepolicyrule. Step 6: From the Selector drop-down, choose the selector type to apply this rule to. A note at the bottom states: ★ Note: Google Cloud only supports DNS Name selector. For example, the Cloud DNS GUI uses a drop down menu representing decision-tree rules. A geolocation (GEO) routing policy lets you map traffic originating from source geographies (Google Cloud regions) to specific DNS targets. Use this policy to distribute incoming requests to different service instances based on the traffic's origin. You can use this feature with the internet, with external traffic, or with traffic originating within Google Cloud and https://cloud.google.com/dns/docs/policies-overview

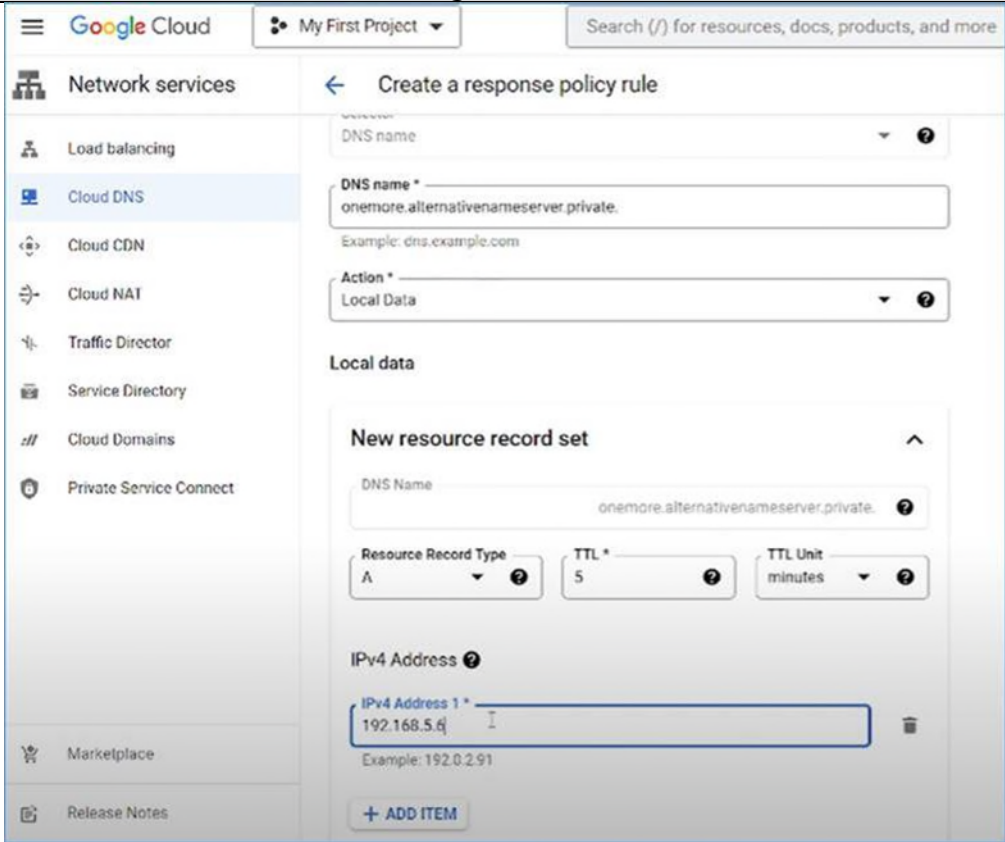
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	 https://www.youtube.com/watch?v=Z-5PcJ_y4Qg at 14:23.

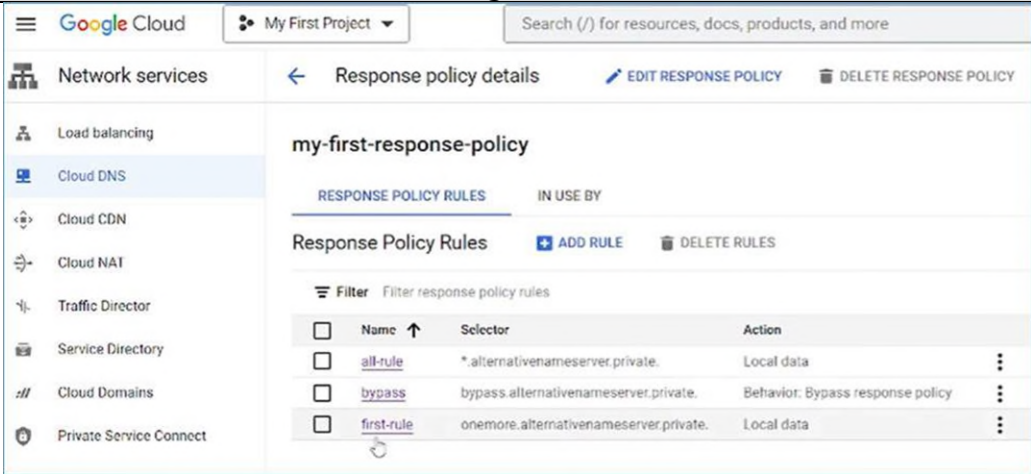
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	 https://www.youtube.com/watch?v=Z-5PcJ_y4Qg at 14:40.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	 https://www.youtube.com/watch?v=Z-5PcJ_y4Qg at 19:19.
(C) providing said at least one policy to the name server network, wherein the decision tree comprises one or more resource nodes, and one or more branch nodes, wherein the one or more resource nodes specify one or more answers to be provided in response to a Domain Name Service (DNS) request, and wherein the one or more branch nodes specify one or more decision criteria to be applied, and wherein the GUI supports the specification of:	Google allows for policies that provide answers (for example, resource nodes) based on criteria (for example, branch nodes) to DNS queries. For example, an interface supports the specification of at least one answer to a resource node and selected from at least types of IP addresses (or, alternatively, world zones), for example, and where the answers to the queries are IP addresses.

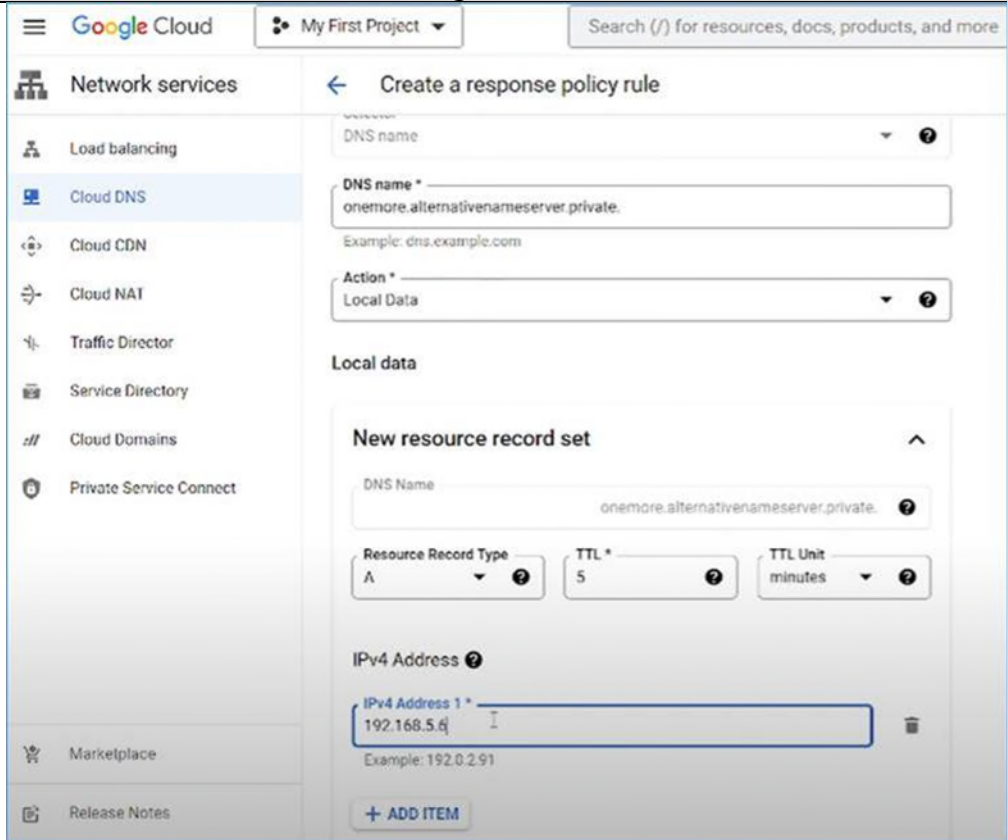
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	2. If you selected Geolocation, do the following: a. For Geo fencing, select Disabled or Enabled. Enabling geo fencing restricts the traffic to a specific geolocation even if all the endpoints in that geolocation are unhealthy. b. In the Source region menu, select a valid Google Cloud source region, such as <code>asia-east1</code>. c. In the Add health checked target section, do the following: i. In the Project list, select the project where the forwarding rule exists. ii. In the Type list, select internal passthrough Network Load Balancer, internal Application Load Balancer, or cross-region internal Application Load Balancer. iii. In the Forwarding rule list, select a forwarding rule. The forwarding rule specifies an internal IP address, port, and a regional backend service or an HTTP(S) proxy. For Cloud DNS to work with health checks, you must enable global access for the internal load balancer. d. To allow IPv4 addresses without health checking, select Allow IPv4 addresses without health checking. e. In the IPv4 Address field, enter an IPv4 address. https://cloud.google.com/dns/docs/zones/manage-routing-policies
(i) one or more answers for the one or more resource nodes, and ii) one or more decision criteria for the one or more branch nodes, wherein the one or more criteria are selected from criteria related to: world zones, countries, states, time zones, and blocks of Internet Protocol (IP) addresses, and wherein the one or more resource nodes are selected from: IP addresses, canonical name	As one example, the following are supported by the Cloud DNS GUI: Decision criteria: DNS name, which can comprise a block of IP addresses because at least some DNS record types supports multiple IP addresses (see screenshot below); Resource nodes: IP addresses – e.g., IPv4, IPv6 Addresses under “New resource record set” below including option to “+ ADD ITEM”; and Answers: specific IP addresses such as 192.168.5.6 below.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
(CNAME) records, mail exchange (MX) records, name server (NS) records, and load sharing server sets; and wherein the answers are selected from: IP addresses and CNAMEs.	 https://www.youtube.com/watch?v=Z-5PcJ_y4Qg at 14:40. Additionally, Cloud DNS allows geolocation routing policies or geofenced routing policies. A geolocation routing policy allows a user to map traffic based on a query source location (e.g., a branch node). The user selects the IP address for forwarding the request when the source is, for example, asia-east1. The answer is a resource node (and an IP address).

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

8,645,517	Google CDN
	Use Cloud Domains to register and manage domains in Google Cloud and automatically set up DNS zones for your domains. https://cloud.google.com/dns?hl=en#section-3; https://docs.infoblox.com/space/vniosgcp/35419584/Creating+a+Cloud+DNS+Policy; https://cloud.google.com/dns/docs/policies-overview Set access control policy for a specific resource To set the access control IAM policy on a specific managed zone, follow these steps. Console gcloud API - In the Google Cloud console, go to the Cloud DNS zones page. Go to Cloud DNS zones - Select one or more zones for which you want to add access control permissions. - On the Permissions to resources page, click Add principal. - On the Grant access to resource page, under New principals, add the email address of the user, group, domain, or service account that you want to add as the new principal. - From the Assign roles list, select the role that you want to assign to the principal. - To assign additional roles, click Add another role. - Click Save. https://cloud.google.com/dns/docs/zones/iam-per-resource-zones

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Exhibit H

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

U.S. Patent No. 9,021,112

Sandpiper CDN

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN
1. A method of content delivery in a network, the method comprising:	Google uses and provides content delivery network(s) (“Google CDN”), including Google’s internal services and services Google offers to third parties that are used to provide content, such as webpages and/or video streams, over a network. Google CDN includes services such as Cloud DNS, for example, along with Google’s own use of technologies to provide content in a network. For example, Cloud DNS provides a plurality of Domain Name System (DNS) servers. For example, Google provides “anycast name servers” for “DNS serving” as part of its Cloud DNS service. Google Cloud DNS is a scalable, reliable and managed authoritative Domain Name System (DNS) service running on the same infrastructure as Google. It has low latency, high availability and is a cost-effective way to make your applications and services available to your users. https://console.cloud.google.com/marketplace/product/google-cloud-platform/cloud-dns Use Google’s infrastructure for production quality and high-volume authoritative DNS serving. Your users will have reliable, low-latency access from anywhere in the world using our anycast name servers. Cloud DNS can scale to large numbers of DNS zones and records. You can reliably create and update millions of DNS records. Our name servers automatically scale to handle query volume. https://cloud.google.com/dns?hl=en
(A) providing a plurality of Domain Name System (DNS) servers associated with a Content Delivery Network (CDN), said plurality of CDN DNS	Google provides DNS servers that share a common anycast address. For example, the “redundant locations” described by Google show multiple servers share a common anycast address in Google Cloud DNS. Cloud DNS servers include servers associated with content servers.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN
servers sharing a common anycast address, wherein each CDN DNS server is associated with a respective plurality of content servers;	Fast anycast name servers Cloud DNS uses our global network of anycast name servers to serve your DNS zones from redundant locations around the world, providing high availability and lower latency for your users. https://cloud.google.com/dns?hl=en
(B) causing said plurality of CDN DNS servers to be authoritative for a hostname associated with a content provider by causing said common anycast address to be associated with said hostname;	Google causes CDN DNS servers to be authoritative for a hostname, for example by causing an anycast address to be associated with a hostname. Google Cloud DNS is a scalable, reliable and managed authoritative Domain Name System (DNS) service running on the same infrastructure as Google. It has low latency, high availability and is a cost-effective way to make your applications and services available to your users. https://console.cloud.google.com/marketplace/product/google-cloud-platform/cloud-dns
(C) responsive to a request for content associated with the content provider and issued by a client, said request including at least said hostname,	Google’s Cloud DNS servers, and the respective plurality of content servers, are configured to support content delivery (e.g., “applications and services” for “users”) for content providers. For example, the list below shows a plurality of content providers.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN																																			
	List of Top 20 Companies That Use Google Cloud Platform The Google Cloud market share contributes to 9% of the global cloud market. The platform has seen a revenue growth up to 45% in the last few years and places third in size of market share. Furthermore, the public cloud platform is also used by a wide variety of popular companies around the world. Some companies that use Google Cloud are as follows: <table><thead><tr><th>Company Name</th><th>Website URL</th><th>Country</th><th>Revenue</th><th>Employee Size</th></tr></thead><tbody><tr><td>Verizon</td><td>verizon.com</td><td>United States</td><td>\$ 34.07 Billion</td><td>118,400</td></tr><tr><td>Twitch Interactive</td><td>twitch.tv</td><td>United States</td><td>\$ 2.6 Billion</td><td>700</td></tr><tr><td>CenturyLink</td><td>centurylink.com</td><td>United States</td><td>\$ 6.3 Billion</td><td>39,000</td></tr><tr><td>LinkedIn Corp</td><td>linkedin.com</td><td>United States</td><td>\$ 10 Billion</td><td>18,000</td></tr><tr><td>NewsCorp</td><td>newscorp.com</td><td>Australia</td><td>\$ 10.39 Billion</td><td>24,000</td></tr><tr><td>SAP</td><td>sap.com</td><td>Germany</td><td>\$ 33.20 Billion</td><td>107,415</td></tr></tbody></table> https://www.thomsondata.com/customer-base/companies-that-use-google-cloud.php	Company Name	Website URL	Country	Revenue	Employee Size	Verizon	verizon.com	United States	\$ 34.07 Billion	118,400	Twitch Interactive	twitch.tv	United States	\$ 2.6 Billion	700	CenturyLink	centurylink.com	United States	\$ 6.3 Billion	39,000	LinkedIn Corp	linkedin.com	United States	\$ 10 Billion	18,000	NewsCorp	newscorp.com	Australia	\$ 10.39 Billion	24,000	SAP	sap.com	Germany	\$ 33.20 Billion	107,415
Company Name	Website URL	Country	Revenue	Employee Size																																
Verizon	verizon.com	United States	\$ 34.07 Billion	118,400																																
Twitch Interactive	twitch.tv	United States	\$ 2.6 Billion	700																																
CenturyLink	centurylink.com	United States	\$ 6.3 Billion	39,000																																
LinkedIn Corp	linkedin.com	United States	\$ 10 Billion	18,000																																
NewsCorp	newscorp.com	Australia	\$ 10.39 Billion	24,000																																
SAP	sap.com	Germany	\$ 33.20 Billion	107,415																																
(c1) causing said hostname to be resolved to said common anycast address by an Internet Service Provider (ISP) DNS server; and then	Google resolves a first hostname to identify an IP address of a content server, in response to a request for content associated with a content provider (where the request includes the hostname). For example, Google Cloud DNS allows requests that include a hostname (e.g., “example.com”). To verify that your configuration is working, <i>after</i> the name servers are updated, navigate to your domain name (for example, <code>example.com</code>). The domain should resolve to your IP address and should point to the Compute Engine VM displaying the <i>Hello World!</i> page that you created in Run a basic Apache web server.																																			

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN
	Set up your domain using Cloud DNS If you are migrating from an existing provider, you can import your existing zones to Cloud DNS. For instructions, see Export your DNS configuration from your existing provider. Otherwise, follow these steps to create a new public zone. 1. In the Google Cloud console, go to the Create a DNS zone page. Go to Create a DNS zone 2. For the Zone type, select Public. 3. For the Zone name, enter <code>my-new-zone</code>. 4. For the DNS name, enter a DNS name suffix for the zone by using a domain name that you registered (for example, <code>example.com</code>). 5. For DNSSEC, ensure that the <code>Off</code> setting is selected. 6. Click Create to create a zone populated with the NS and SOA records. 7. To point your registered domain name to the IP address of the hosting server, you must add an A record to your zone: https://cloud.google.com/dns/docs/tutorials/create-domain-tutorial

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN
	Change your domain registrar's name servers for your domain Now that you have the list of Cloud DNS name servers hosting your managed zone, use your domain registrar to update the name servers for your domain. Your domain registrar might be Google Domains, Cloud Domains, or a third-party registrar. ★ Note: Instructions vary based on your registrar. Typically, you must provide at least two Cloud DNS name servers to the domain registrar. To benefit from Cloud DNS's high availability, you must use all the name servers. After changing your domain registrar's name servers, it can take a while for resolver traffic to be directed to your new Cloud DNS name servers. Resolvers could continue to use your old name servers until the TTL on the old NS records expire. https://cloud.google.com/dns/docs/update-name-servers?sjid=13749068997963802575-NC To process a request, Cloud DNS first tries a DNS query by contacting the forwarding target with the highest ranking. If that server does not respond, Cloud DNS repeats the request to the next highest ranked forwarding target. If no forwarding targets reply, Cloud DNS synthesizes a SERVFAIL response. https://cloud.google.com/dns/docs/zones/zones-overview
(c2) by one of said plurality of CDN DNS servers, resolving said hostname to identify an IP address for use by the	Google resolves hostnames to identify IP addresses for use by the client.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

9,021,112	Google CDN
client to retrieve the content from a content server.	Google Cloud DNS is a scalable, reliable and managed authoritative Domain Name System (DNS) service running on the same infrastructure as Google. It has low latency, high availability and is a cost-effective way to make your applications and services available to your users. https://console.cloud.google.com/marketplace/product/google-cloud-platform/cloud-dns What is Cloud CDN? Cloud CDN is a content delivery network that accelerates your web and video content delivery by using Google's global edge network to bring content as close to your users as possible. As a result latency, cost, and load on your backend servers is reduced, making it easier to scale to millions of users. Global anycast IP provides a single IP for global reach. It enables Google Cloud to route users to the nearest edge cache automatically and avoid DNS propagation delays that can impact availability. It supports HTTP/2 end-to-end and the QUIC protocol from client to cache. QUIC is a multiplexed stream transport over UDP, which reduces latency and makes it ideal for lossy mobile networks. https://cloud.google.com/blog/topics/developers-practitioners/what-cloud-cdn-and-how-does-it-work

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Exhibit I

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

U.S. Patent No. 10,924,573

Sandpiper CDN

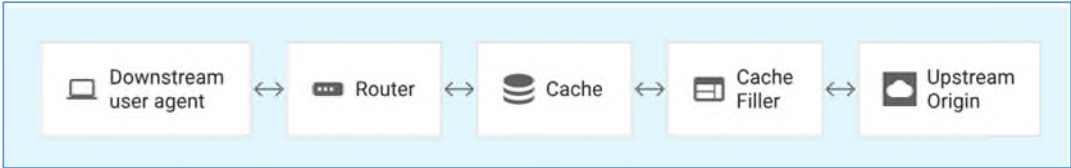
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
19. A method of content delivery in a content delivery network comprising: receiving, at a first server of a first tier of servers of the content delivery network, a request from a requesting device for a resource available from the content delivery network;	Google uses and provides content delivery network(s) (“Google CDN”), including Google’s internal services and services Google offers to third parties that are used to provide content, such as webpages and/or video streams, over a network. Google CDN includes services such as Google Cloud CDN and Google Media CDN, for example, along with Google’s own use of technologies to provide content in a network. As one example, Google provides a method of content delivery that includes receiving a request from a device for a resource available from a content delivery network at a server in a first tier of the network. For example, a server in the Media CDN “edge” (e.g., “EdgeCacheService”), as discussed below, includes a server in a first tier that receives client HTTP requests. You can control how content is cached for each URI you serve in a <i>route</i>. Using a route lets you optimize behavior based on the type of content, client attributes, and your freshness requirements for each route you define with Media CDN. The router and cache component are configured with the IP addresses and protocols that they use, security and cache policies, and a routing configuration. The configuration is contained in an EdgeCacheService resource. The router uses the HeaderAction, described in EdgeCacheService, to determine how the router modifies requests (including header manipulation and path rewrites). Media CDN terminates user HTTP requests at a reverse proxy. The reverse proxy matches the HTTP requests to a Routing configuration. The routing configuration defines how HTTP requests are routed, modified, cached, and filled.

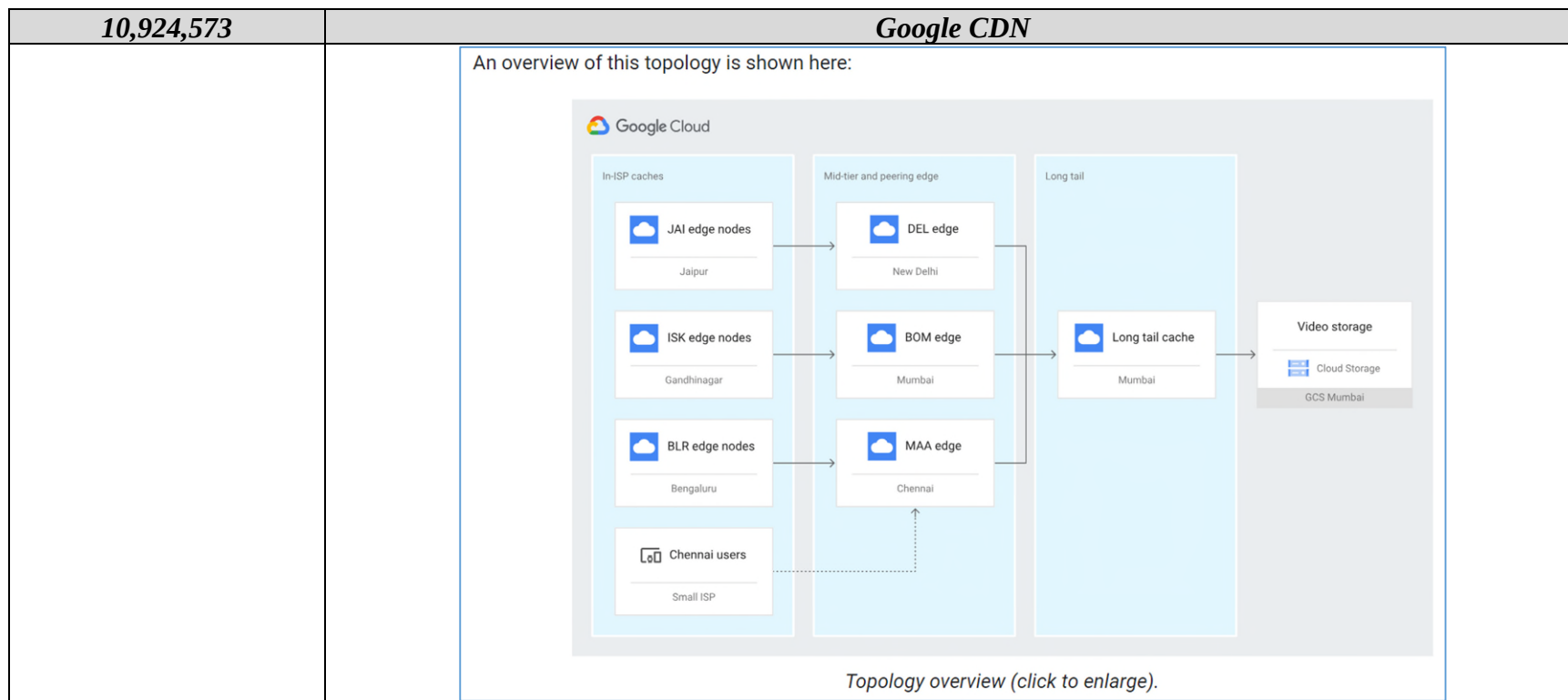
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	Media CDN logs each HTTP request between the client and edge, and between Media CDN and the origin, to Cloud Logging (Logging). Logs are  <pre> graph LR A[Downstream user agent] <--> B[Router] B <--> C[Cache] C <--> D[Cache Filler] D <--> E[Upstream Origin] </pre> https://cloud.google.com/media-cdn/docs/overview The Google CDN includes a multi-tiered network with a “long tail” cache at the top of the hierarchy. The Google CDN includes long-tail caches with “substantial storage capacity,” indicating that a mid-tier and/or edge nodes in the Google CDN have less storage capacity. Origin shielding ↔ Media CDN provides a deeply tiered edge infrastructure that is designed to actively minimize cache fill wherever possible. There are three primary layers of caching infrastructure: 1. Deep edge caches, which serve the majority of traffic and off-load within a service providers network. 2. Google's peering edge, which is connected to thousands of ISPs and acts as the mid-tier cache for edge caches, and for cases where those are not present within a given ISP, the user-facing cache. 3. Long-tail caches within Google's network that other downstream caches fill from prior to the origin. These caches support significant fan-in, have substantial cache storage capacity, and act as an origin <i>shield</i>.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.



PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	All layers of caching support request collapsing (or <i>coalescing</i>) to further reduce origin load. Based on observed, real-world workloads at scale: • > 95% of cache fill uses a dedicated long-tail cache node within the region, in order to reduce cache fill costs and latency. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. • Media CDN has significant storage capacity across caches, which minimizes the eviction rates even for long-tail, less popular content. Customers might see different offload rates depending on their cache configuration, user load, workloads (such as live versus on-demand), user distribution, and how much long-tail content (total corpus size) they serve to users across regions. https://cloud.google.com/media-cdn/docs/origins#origin-shielding
accessing a popularity service associated with the first server of the first tier of the content delivery network to determine a popularity designation associated with the requested resource;	Google accesses a popularity service to determine a popularity designation associated with a requested resource. For example, Google Media CDN caches “the most popular content” closer to users, while eventually evicting “unpopular content.” Google CDN analyzes the “relative’ popularity of content. As one specific example, on information and belief, the Google CDN includes caches that “cross-fill from each other where advantageous to do so,” indicating a popularity is determined.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	All layers of caching support request collapsing (or <i>coalescing</i>) to further reduce origin load. Based on observed, real-world workloads at scale: • > 95% of cache fill uses a dedicated long-tail cache node within the region, in order to reduce cache fill costs and latency. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. • Media CDN has significant storage capacity across caches, which minimizes the eviction rates even for long-tail, less popular content. Customers might see different offload rates depending on their cache configuration, user load, workloads (such as live versus on-demand), user distribution, and how much long-tail content (total corpus size) they serve to users across regions. https://cloud.google.com/media-cdn/docs/origins#origin-shielding Revalidation, eviction, and expiry Content delivery networks, including Media CDN, operate by caching the most popular content as close to users as possible. Media CDN's extensive storage, as well as origin shielding, limits the need to evict even unpopular content. Content that is accessed a small number of times per day might eventually be evicted.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	https://cloud.google.com/media-cdn/docs/caching the nature of caches, it is impossible to predict whether a particular request is served out of a cache. You can, however, expect that popular requests for cacheable content are served from a cache most of the time, yielding significantly reduced latencies, reduced cost, and reduced load on your origin servers. <i>Unpopular content</i> means content that hasn't been accessed in a while. A <i>while</i> and <i>unpopular</i> are both relative to the bulk of other items in the cache. Multiple Google Cloud projects share a common pool of cache space because the projects are served from the same set of GFEs. The relative popularity of content is compared across multiple projects, not only within a single project. https://cloud.google.com/cdn/docs/overview Time to live (TTL) overrides let you set default TTL values for cached content and override TTL values set in the <code>max-age</code> and <code>s-maxage</code> cache control directives (or <code>Expires</code> headers) set by your origins. TTLs, whether set by overrides or by using a cache directive, are optimistic. Content that is rarely accessed or unpopular might be evicted from the cache prior to the TTL being reached. https://cloud.google.com/media-cdn/docs/caching#cache-modes

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	Content delivery networks, including Media CDN, operate by caching the most popular content as close to users as possible. Media CDN's extensive storage, as well as origin shielding, limits the need to evict even unpopular content. Content that is accessed a small number of times per day might eventually be evicted. • Cached responses that reach their configured TTL might not be immediately evicted. For popular content, Media CDN revalidates that the cached response is the latest version by issuing a <code>HEAD</code> request to the origin to confirm that the headers have not changed. Under some circumstances, Media CDN instead sends a request to the origin with an <code>If-None-Match</code> and/or <code>If-Modified-Since</code> request header. In this case, correctly configured origins should return an HTTP 304 (Not Modified) response, without the body bytes, if the cache has the "latest" copy of that response. https://cloud.google.com/media-cdn/docs/caching#cache-modes
receiving an instruction to not cache the resource at the first server of the first tier of servers when the portion of the resource is obtained from the content server of the content delivery network;	Google receives instructions to not cache, for example in Media CDN when caches “cross-fill from each other,” a cache providing a resource includes at least some popularity information for a receiving cache to be used to decide whether to cache the resource or not in response to an instruction.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	All layers of caching support request collapsing (or <i>coalescing</i>) to further reduce origin load. Based on observed, real-world workloads at scale: • > 95% of cache fill uses a dedicated long-tail cache node within the region, in order to reduce cache fill costs and latency. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. • Media CDN has significant storage capacity across caches, which minimizes the eviction rates even for long-tail, less popular content. Customers might see different offload rates depending on their cache configuration, user load, workloads (such as live versus on-demand), user distribution, and how much long-tail content (total corpus size) they serve to users across regions. https://cloud.google.com/media-cdn/docs/origins#origin-shielding Revalidation, eviction, and expiry Content delivery networks, including Media CDN, operate by caching the most popular content as close to users as possible. Media CDN's extensive storage, as well as origin shielding, limits the need to evict even unpopular content. Content that is accessed a small number of times per day might eventually be evicted.

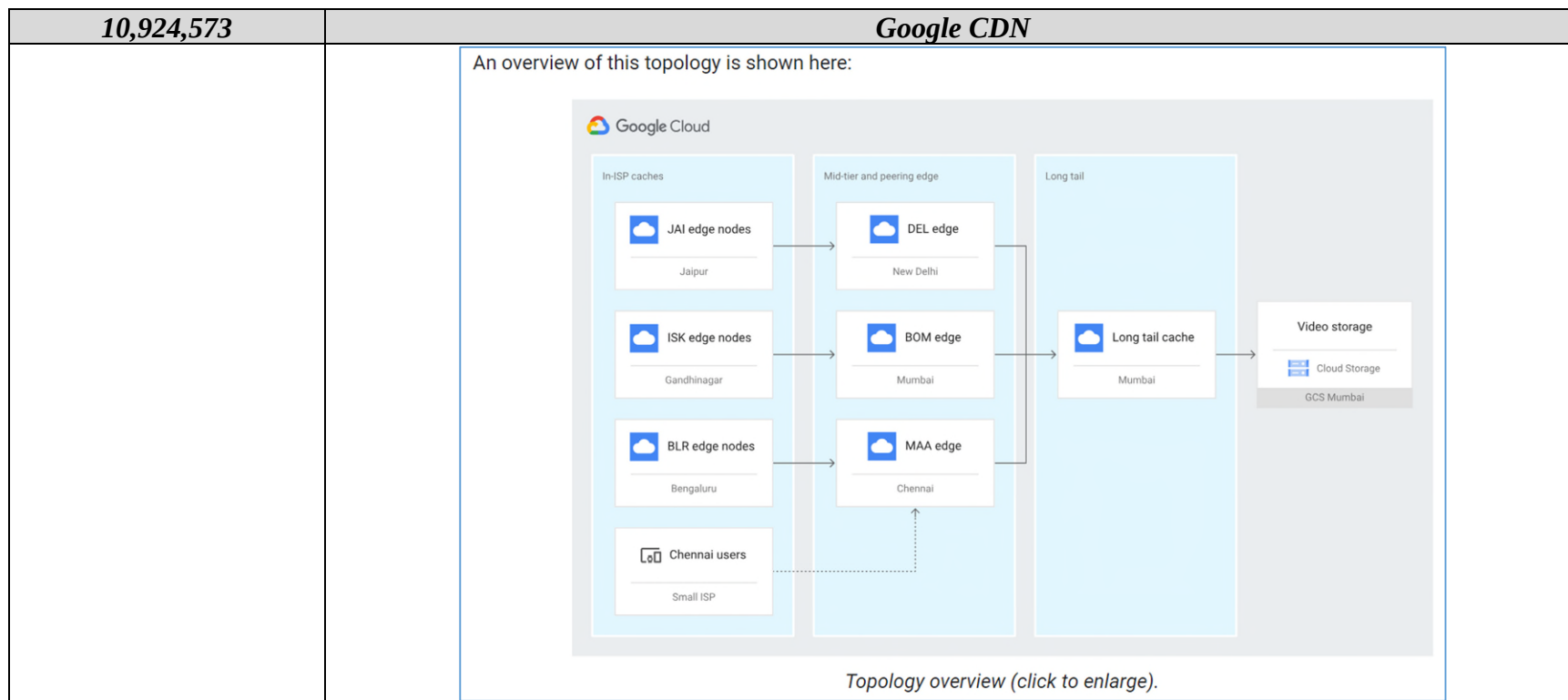
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	https://cloud.google.com/media-cdn/docs/caching
requesting the resource from a second server of the content delivery network based at least on the popularity designation associated with the requested resource; and	Google requests a resource from a second server of the Google CDN based on a popularity designation. Origin shielding  Media CDN provides a deeply tiered edge infrastructure that is designed to actively minimize cache fill wherever possible. There are three primary layers of caching infrastructure: 1. Deep edge caches, which serve the majority of traffic and off-load within a service providers network. 2. Google's peering edge, which is connected to thousands of ISPs and acts as the mid-tier cache for edge caches, and for cases where those are not present within a given ISP, the user-facing cache. 3. Long-tail caches within Google's network that other downstream caches fill from prior to the origin. These caches support significant fan-in, have substantial cache storage capacity, and act as an origin <i>shield</i>.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.



PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	All layers of caching support request collapsing (or <i>coalescing</i>) to further reduce origin load. Based on observed, real-world workloads at scale: • > 95% of cache fill uses a dedicated long-tail cache node within the region, in order to reduce cache fill costs and latency. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. • Media CDN has significant storage capacity across caches, which minimizes the eviction rates even for long-tail, less popular content. Customers might see different offload rates depending on their cache configuration, user load, workloads (such as live versus on-demand), user distribution, and how much long-tail content (total corpus size) they serve to users across regions. https://cloud.google.com/media-cdn/docs/origins#origin-shielding
providing the obtained resource to the requesting device.	Google provides a resource, for example Google Media CDN provides the resource to a requesting device. Media CDN uses YouTube's infrastructure to bring video streams (VoD and live) and large file downloads closer to users for fast and reliable delivery. https://cloud.google.com/cdn?hl=en

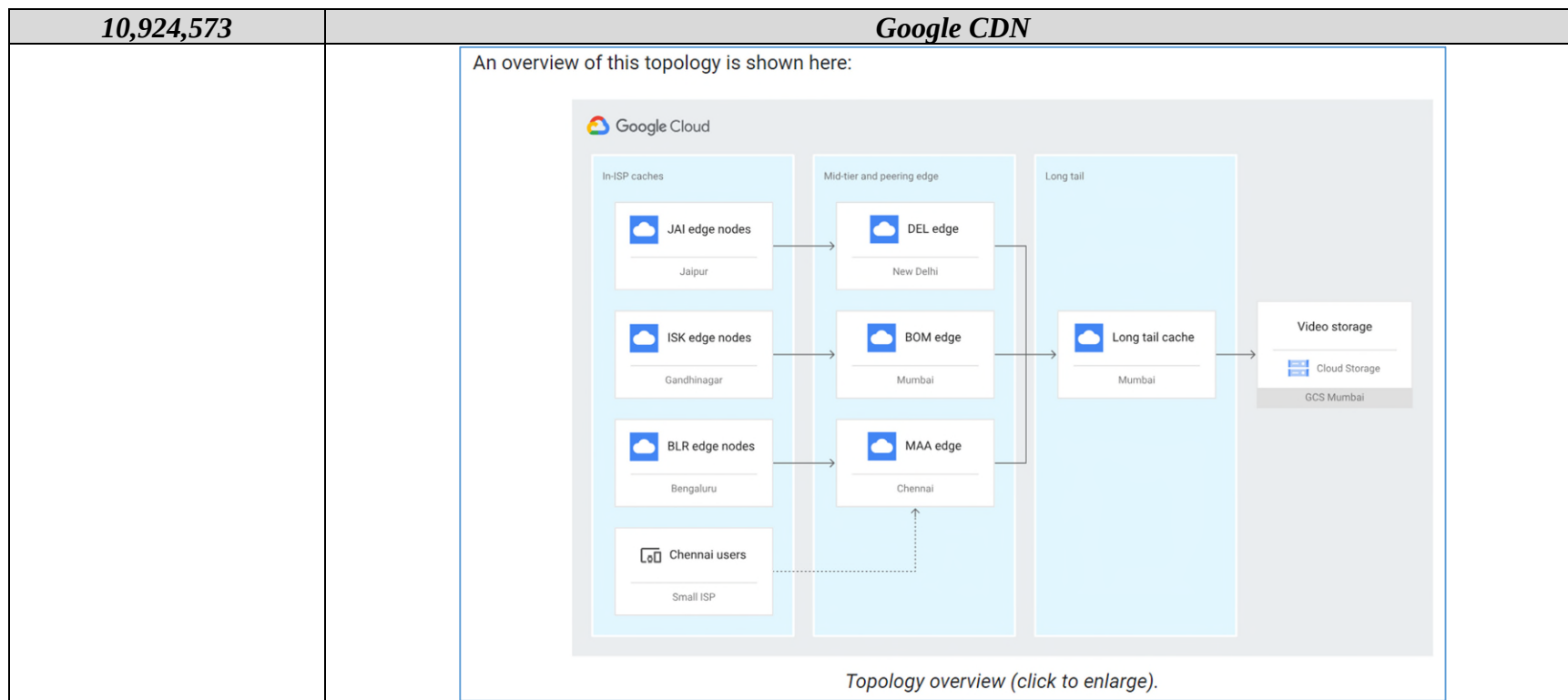
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
20. The method as recited in claim 19 wherein the second server of the content delivery network is a content server storing the requested content.	Google includes a second server of the CDN that is a content server storing the requested content. Origin shielding  Media CDN provides a deeply tiered edge infrastructure that is designed to actively minimize cache fill wherever possible. There are three primary layers of caching infrastructure: 1. Deep edge caches, which serve the majority of traffic and off-load within a service providers network. 2. Google's peering edge, which is connected to thousands of ISPs and acts as the mid-tier cache for edge caches, and for cases where those are not present within a given ISP, the user-facing cache. 3. Long-tail caches within Google's network that other downstream caches fill from prior to the origin. These caches support significant fan-in, have substantial cache storage capacity, and act as an origin <i>shield</i>.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.



PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

10,924,573	Google CDN
	All layers of caching support request collapsing (or <i>coalescing</i>) to further reduce origin load. Based on observed, real-world workloads at scale: • > 95% of cache fill uses a dedicated long-tail cache node within the region, in order to reduce cache fill costs and latency. • Cache fill between the origin and Google's own edge infrastructure is entirely over Google's global private backbone network, which reduces cache fill latency and improves reliability—both are active benefits for live streaming workloads. • Caches cross-fill from each other where advantageous to do so, further driving down cache fill rates. • Media CDN has significant storage capacity across caches, which minimizes the eviction rates even for long-tail, less popular content. Customers might see different offload rates depending on their cache configuration, user load, workloads (such as live versus on-demand), user distribution, and how much long-tail content (total corpus size) they serve to users across regions. https://cloud.google.com/media-cdn/docs/origins#origin-shielding

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Exhibit J

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

U.S. Patent No. 10,057,322

Sandpiper CDN

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

Pat. No. 10,057,322	Google CDN
15. A method comprising: providing, at a content delivery network, an application programming interface (API);	Google provides a method that includes providing, at a content delivery network (CDN), an application programming interface (API). For example, YouTube and Media CDN utilize the same infrastructure and/or certain APIs. Media CDN is Google Cloud's media delivery CDN platform designed for streaming video and large object downloads. Media CDN complements Cloud CDN, Google Cloud's web acceleration CDN platform. With Google Cloud's Media CDN, customers can now leverage the same infrastructure that YouTube uses to serve its global users. Media CDN gives its customers a platform providing massive coverage and presence globally in over 206+ countries/territories and 1300+ cities. In this blog post, we will walk you through the steps of turning up Google Cloud Media CDN. We will cover the following topics: https://cloud.google.com/blog/products/networking/deploy-streaming-service-with-media-cdn Cloud CDN accelerates web applications using Google's global edge network. Media CDN uses YouTube's infrastructure to bring video streams (VoD and live) and large file downloads closer to users for fast and reliable delivery. https://cloud.google.com/cdn

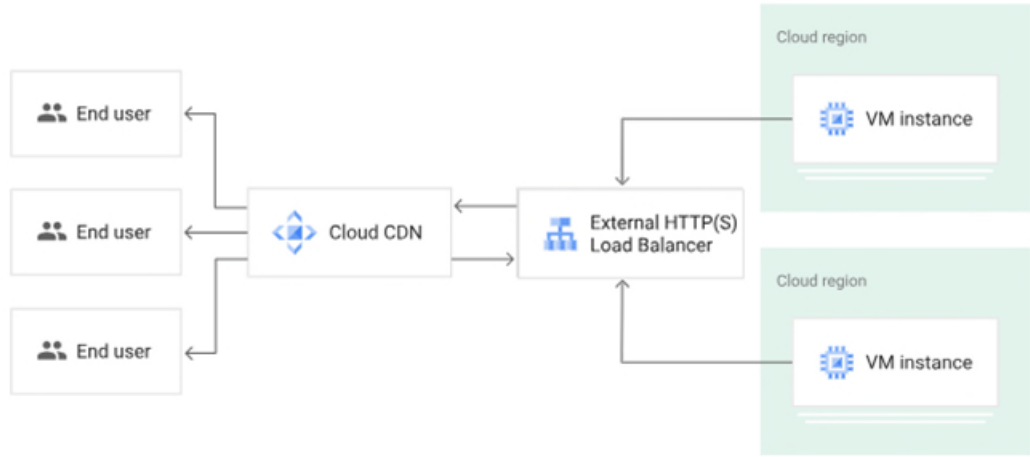
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

	Google documents various API methods for managing requests through Google Media CDN. In one example, Google documentation describes a configurable method for matching requests to content on the CDN based on factors such as network topology, path parameters, customer policy information, geography information, and the like. Routing Defines how requests are routed, modified, cached, and which origin the content is filled from. JSON representation <pre>{ "hostRules": [{ object (HostRule) }], "pathMatchers": [{ object (PathMatcher) }] }</pre> “Excerpt from Google Media CDN documentation explaining configuration parameters for routing requests to the CDN. https://cloud.google.com/media-cdn/docs/reference/rest/v1/projects.locations.edgeCacheServices#routing”
at the API, receiving, from an origin system that is separate from the content delivery network, an	Google provides logic and infrastructure that receives, at the API, from an origin system that is separate from the content delivery network, an embedded resource (such as a request URL) and at least one parameter associated with the embedded resource. For example, Media CDN includes an API receiving, from an origin system, an embedded resource such as an HTTP request with a hostname and/or other parameters such as additional path and/or query parameters associated with the embedded resource.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

embedded resource and at least one parameter associated with the embedded resource;	 Figure 1. Responses flow from origin servers through Cloud CDN to clients. https://cloud.google.com/cdn/docs/overview Google provides API functionality to facilitate communication between an origin system (such as an origin system associated with YouTube’s infrastructure) and the CDN. Google describes how users can utilize the Media CDN service in conjunction with separate established infrastructure. Media CDN lets you easily fetch content from publicly accessible HTTP endpoints. You can use Media CDN with your existing origin infrastructure, whether the content is hosted within Cloud Storage, in another cloud, or within your on-premises infrastructure. “Excerpt from Google Media CDN documentation https://cloud.google.com/media-cdn/docs/overview”
generating a modified	Google provides logic and infrastructure that generates a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;	In a first example, a modified embedded resource (e.g., the “https://rr4---sn-q4flrnsk.googlevideo.com/videoplayback?expire=...” URL highlighted on the screenshot below) is generated and provides a direct link to a node with the content delivery network. As shown, a request made to the YouTube hostname (e.g., the origin system) results in a response that includes embedded resources (e.g., URLs) corresponding to a local host on which the requested content is stored. Therefore, a modified embedded resource (e.g., a modified URL) is returned by the Google Media CDN to the YouTube origin server so that YouTube can deliver information to the client, thereby enabling the client to directly receive the embedded object, for example, from a local server within the Google Media CDN. <pre> "streamingData": { "expiresInSeconds": "21540", "formats": [{ "itag": 18, "url": "https://rr4---sn-q4flrnsk.googlevideo.com/videoplayback?expire=1732247818\u0026ei=qqw_Z-2hA", "mimeType": "video/mp4; codecs=\"avc1.42001E, mp4a.40.2\"", "bitrate": 342693, "width": 640, "height": 360, "lastModified": "1697929619917530", "quality": "medium", "fps": 30, "qualityLabel": "360p", "projectionType": "RECTANGULAR", "audioQuality": "AUDIO_QUALITY_LOW", "approxDurationMs": "1575195", "audioSampleRate": "44100", "audioChannels": 2 }], </pre> Excerpt from HTML code containing reference to the CDN node on which the video is stored. This code is returned as a response when requesting the following YouTube URL; https://www.youtube.com/watch?v=Tx69m9Jb0EY
---	---

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

In a second example, a request was made for the same video resource using a different network from the network used in the first example. In this second example, the embedded resource (e.g., URL) pointing to the CDN is modified prior to serving the client device. As shown in the highlighted portion of the second example below, the embedded resource (e.g., URL) is modified differently as compared to the embedded resource (e.g., URL) from the first example above.

```
"streamingData": {
  "expiresInSeconds": "21540",
  "formats": [{
    "itag": 18,
    "url": "https://rr3---sn-qxo7rn7y.googlevideo.com/videoplayback?expire=1732325796\u0000",
    "mimeType": "video/mp4; codecs=\"avc1.42001E, mp4a.40.2\"",
    "bitrate": 342693,
    "width": 640,
    "height": 360,
    "lastModified": "1697929619917530",
    "quality": "medium",
    "fps": 30,
    "qualityLabel": "360p",
    "projectionType": "RECTANGULAR",
    "audioQuality": "AUDIO_QUALITY_LOW",
    "approxDurationMs": "1575195",
    "audioSampleRate": "44100",
    "audioChannels": 2
  ]},
```

Excerpt from HTML code containing reference to the CDN node on which the video is stored.

Based on request from network separate from the previous example.

<https://www.youtube.com/watch?v=Tx69m9Jb0EY>

In further support, Google provides webpages with embedded videos. In doing so, Google further generates a modified embedded resource providing a direct link (e.g., a link pointing to specific cache server), as indicated below.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

	First, we need to know how the basic setup of YouTube’s video and load distribution works. When a client requests a video from YouTube’s website youtube.com, the web frontend returns a rendered HTML webpage which contains a link pointing to a specific cache server (e.g. r4---sn-4g5e6nsd.googlevideo.com). YouTube does not disclose how and based Another interesting observation we made while uploading videos through YouTube’s web interface was that the video server cache URLs differ between the video uploader and other users or viewers. The uploader gets a URL of the form https://r1---sn-bg07dnsr.c.youtube.com, whereas others get URL likes https://r1---sn-5hne6nsd.googlevideo.com. There seem https://blog.cubieserver.de/2020/investigations-into-video-streaming/report1-analysis-of-youtube-cdn.pdf
delivering, by the content delivery network, the modified embedded resource to the origin system;	Google provides logic and infrastructure that delivers, by the content delivery network, the modified embedded resource to the origin system. For example, YouTube provides videos requested by users with the link to a specific cache server within Google’s CDN. In one example, the modified embedded resource includes the direct link (e.g., a link pointing) to specific cache server. The following image shows how a modified embedded resource (e.g., the “https://rr4---sn-q4flrnsk.googlevideo.com/videoplayback?expire= . . .” URL highlighted on the screenshot below) is generated and delivered. In the two example images below, the modified embedded resource differs when made from different networks. Additionally, the following two images show the response body generated from an API request to YouTube for a video. In particular, the response includes code for displaying the webpage, logic required to make additional requests and accompanying CDN node URL (e.g., “the modified embedded resource”) which reference the node(s) on which the requested video was cached and the functionality to make requests utilizing the node(s).

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

```
"streamingData": {  
  "expiresInSeconds": "21540",  
  "formats": [{  
    "itag": 18,  
    "url": "https://rr4---sn-q4flrnsk.googlevideo.com/videoplayback?expire=1732247818\u0026ei=qqw_Z-2hA",  
    "mimeType": "video/mp4; codecs=\"avc1.42001E, mp4a.40.2\"",  
    "bitrate": 342693,  
    "width": 640,  
    "height": 360,  
    "lastModified": "1697929619917530",  
    "quality": "medium",  
    "fps": 30,  
    "qualityLabel": "360p",  
    "projectionType": "RECTANGULAR",  
    "audioQuality": "AUDIO_QUALITY_LOW",  
    "approxDurationMs": "1575195",  
    "audioSampleRate": "44100",  
    "audioChannels": 2  
  }],  
}
```

Excerpt from HTML code containing reference to the CDN node on which the video is stored.

This code is returned as a response when requesting the following YouTube URL;

<https://www.youtube.com/watch?v=Tx69m9Jb0EY>

In a second example, a request was made for the same video resource from the client device connected to a network different from the network used in the first example. In this case, the URL is pointing to the CDN is modified prior to being served to the client device.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

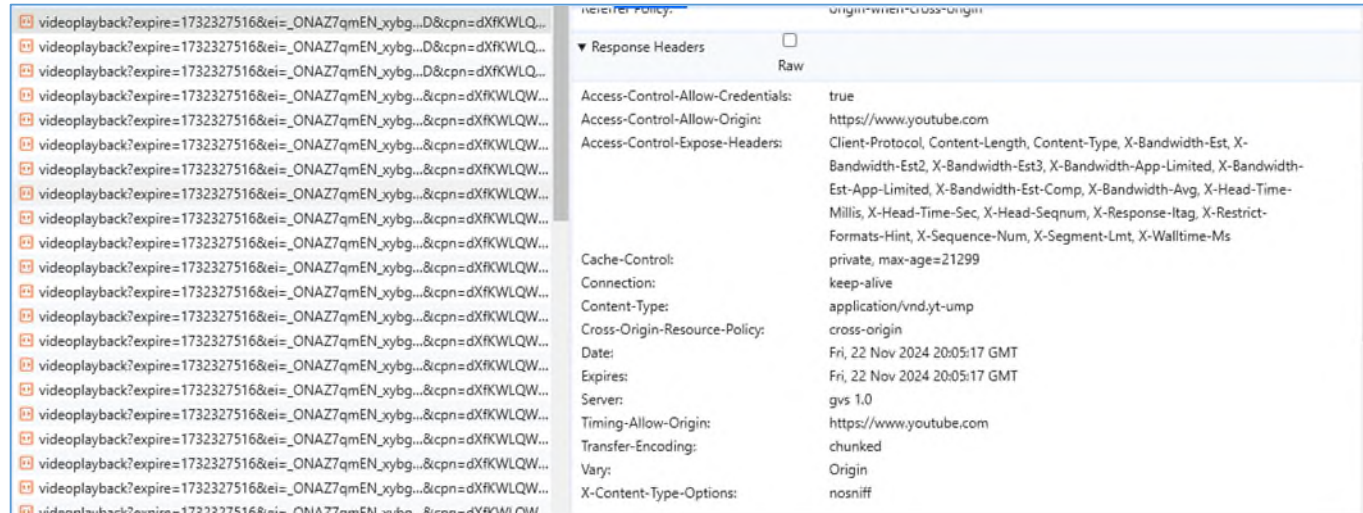
	<pre> "streamingData": { "expiresInSeconds": "21540", "formats": [{ "itag": 18, "url": "https://rr3---sn-qxo7rn7y.googlevideo.com/videoplayback?expire=1732325796\u00 "mimeType": "video/mp4; codecs=\"avc1.42001E, mp4a.40.2\"", "bitrate": 342693, "width": 640, "height": 360, "lastModified": "1697929619917530", "quality": "medium", "fps": 30, "qualityLabel": "360p", "projectionType": "RECTANGULAR", "audioQuality": "AUDIO_QUALITY_LOW", "approxDurationMs": "1575195", "audioSampleRate": "44100", "audioChannels": 2 }], </pre> Excerpt from HTML code containing reference to the CDN node on which the video is stored. Based on request from network separate from the previous example. https://www.youtube.com/watch?v=Tx69m9Jb0EY In further support, YouTube provides videos embedded in webpages requested by users, for example using a link to a specific cache server, as indicated above. https://blog.cubieserver.de/2020/investigations-into-video-streaming/report1-analysis-of-youtube-cdn.pdf Upon information and belief, based on the processes described in the above sections and examples provided herein, the modified embedded resource is provided to the origin system by the content delivery network, for example, through the request URL modification.
receiving, at the node, a request	Google provides logic and infrastructure for receiving, at the node, a request from a client device to obtain the content.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

from a client device to obtain the content; and

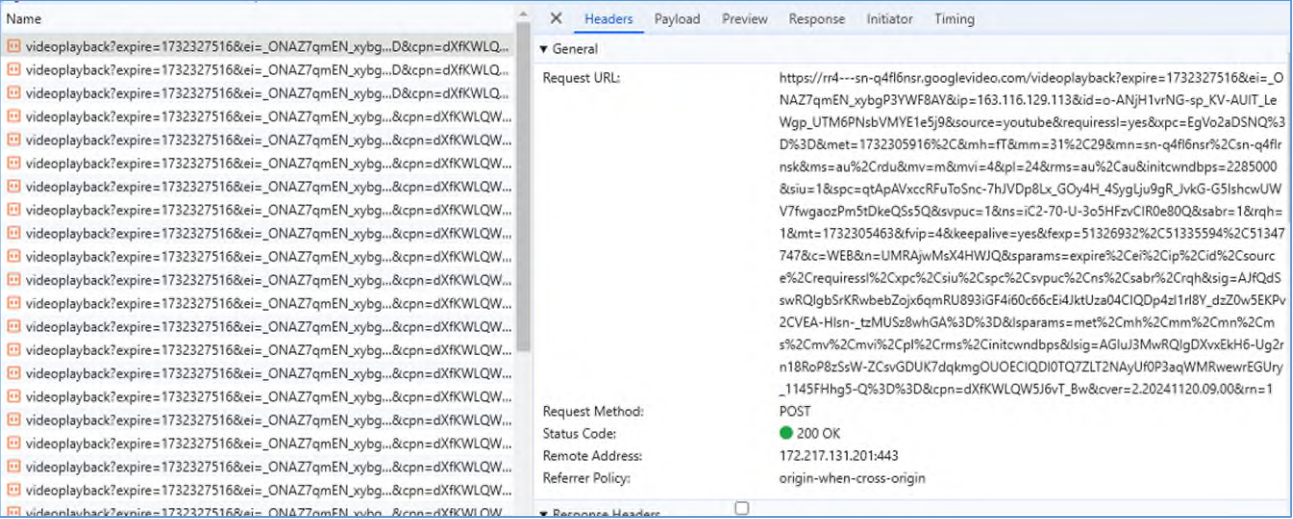
In one example, client devices request YouTube videos. After the origin system exposes the modified embedded resource mentioned in the prior section to the client device, the client device sends additional requests to obtain data associated with the content such as video and audio files. The following two images show requests to Google's CDN for audio and video playback from the node that is specified in the modified embedded resource.



“Excerpt of HTTP requests made to Google CDN when playing content title ‘Pressure Canning 4 Recipes to Fill the Pantry with 16 Convenience Meals’ On chrome browser session on a Windows PC as displayed by chrome browser tools.”
<https://www.youtube.com/watch?v=Tx69m9Jb0EY>

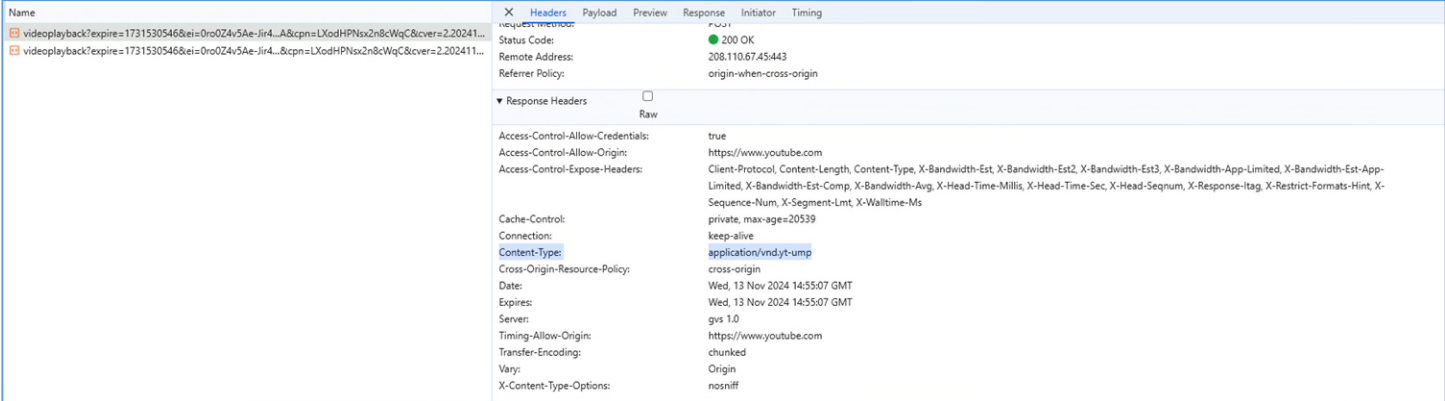
PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

	 “Excerpt of an individual HTTP request generated by YouTube streaming service detailing request headers including the specific CDN node utilized for the aforementioned content title” https://www.youtube.com/watch?v=Tx69m9Jb0EY
providing the content to the client device.	Google provides the content to the client device.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

	 Excerpt of the same HTTP request, showing the content type returned in the Universal MIDI Packet format. In the example above, Google CDN provides content, including video data to the client device in a specialized Universal MIDI Packet (e.g., UMP) format. In another example, a client device was used to download a YouTube video without using a web browser. The following image shows how YouTube provides URLs pointing to the CDN nodes that contain the video information that is then sent to the client device. The following two URLs represent the video and audio components (highlighted below) of a YouTube video respectively, and are used to locally download videos from YouTube.
--	--

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

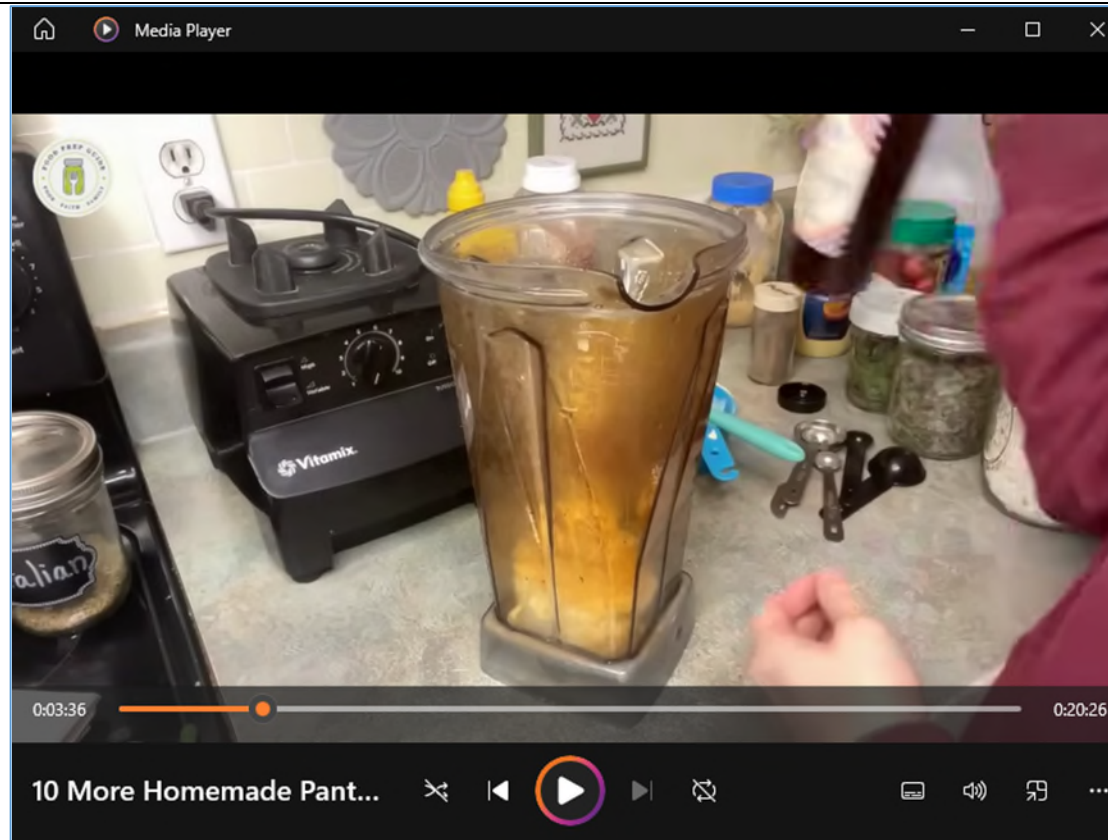
```
PS C:\Users\lvanfossan\Downloads\YTDownloads> yt-dlp --no-check-certificate -g 'https://www.youtube.com/watch?v=
c0cEmQgwc18'
https://rr4---sn-q4fl6nsk.googlevideo.com/videoplayback?expire=1732327240&ei=60JAZ5SIIoC1qvAPgLPUIQo&ip=108.233.
53.103&id=o-AJ9hL3_JlQsq-Y5Kl6hfbn9N5clpzVqguRJuTVkVi5EP&itag=397&source=youtube&requiressl=yes&xpc=EgVo2aDSNQ%3
D%3D&met=1732305640%2C&mh=ha&mm=31%2C29&mn=sn-q4fl6nsk%2Csn-q4flrnes&ms=au%2Crdu&mv=m&mvi=4&pl=20&rms=au%2Cau&in
itcwndbps=608750&bui=AQn3pFT6T0FdCdqaSH_b388col742pAwJSITc2CLQHIIaP3_Bh7kt1fhW_fhBcMGJGG_5Vlq6VBFjWuy&spc=qtApAf
e4yn1tAcmedIY1h86N4B7f1zLizQM5e4nthr-vkEA&vprv=1&svpuc=1&mime=video%2Fmp4&rqh=1&gir=yes&clen=67217109&dur=1442.0
33&lmt=1731034029756494&mt=1732305213&fvip=3&keepalive=yes&fexp=51326932%2C51335594&c=IOS&txp=543G434&sparams=ex
pire%2Ce%2Cip%2Cid%2Citag%2Csource%2Crequiressl%2Cxcpc%2Cbui%2Cspc%2Cvprv%2Csvpuc%2Cmime%2Crqh%2Cgir%2Cclen%2Cdu
r%2Clmt&sig=AJfQdSswRAIgIJ_OXTzirYqHFrEeo7nMlemZLqer6mJdfa-Vfpn5rYCIHOjFMKgxFTbLXfTRoeSQtUtG0qYPVh8XIZFYnIuRza-
&lparams=met%2Cmh%2Cmm%2Cmn%2Cms%2Cmv%2Cmvi%2Cpl%2Crms%2Cinitcwndbps&sig=AGluJ3MwRQIhAJu0-i6wr3KBa4_HCZzIj20To
H5GH2xgirkDySfEMHTvAiBI1NqTBJ87WGStGzmtLrFLZ4Kt5aNB6LNkr97-fkJTg%3D%3D
https://rr4---sn-q4fl6nsk.googlevideo.com/videoplayback?expire=1732327240&ei=60JAZ5SIIoC1qvAPgLPUIQo&ip=108.233.
53.103&id=o-AJ9hL3_JlQsq-Y5Kl6hfbn9N5clpzVqguRJuTVkVi5EP&itag=251&source=youtube&requiressl=yes&xpc=EgVo2aDSNQ%3
D%3D&met=1732305640%2C&mh=ha&mm=31%2C29&mn=sn-q4fl6nsk%2Csn-q4flrnes&ms=au%2Crdu&mv=m&mvi=4&pl=20&rms=au%2Cau&in
itcwndbps=608750&bui=AQn3pFT6T0FdCdqaSH_b388col742pAwJSITc2CLQHIIaP3_Bh7kt1fhW_fhBcMGJGG_5Vlq6VBFjWuy&spc=qtApAf
e4yn1tAcmedIY1h86N4B7f1zLizQM5e4nthr-vkEA&vprv=1&svpuc=1&mime=audio%2Fwebm&rqh=1&gir=yes&clen=20588227&dur=1442.
061&lmt=1731028138841268&mt=1732305213&fvip=3&keepalive=yes&fexp=51326932%2C51335594&c=IOS&txp=5432434&sparams=e
xpire%2Ce%2Cip%2Cid%2Citag%2Csource%2Crequiressl%2Cxcpc%2Cbui%2Cspc%2Cvprv%2Csvpuc%2Cmime%2Crqh%2Cgir%2Cclen%2Cd
ur%2Clmt&sig=AJfQdSswRgIhAIRr4qADHVb8l-Q-V815an1BjBZdAYeJ-8LxZA0Hc1w0AiEA-aP9zI1cJf8N4dTxxCNSRcCTrMZQ-8_fmovaMP
jKsg%3D&lparams=met%2Cmh%2Cmm%2Cmn%2Cms%2Cmv%2Cmvi%2Cpl%2Crms%2Cinitcwndbps&sig=AGluJ3MwRQIhAJu0-i6wr3KBa4_HCZ
zIj20ToH5GH2xgirkDySfEMHTvAiBI1NqTBJ87WGStGzmtLrFLZ4Kt5aNB6LNkr97-fkJTg%3D%3D
PS C:\Users\lvanfossan\Downloads\YTDownloads>
```

Excerpt of download information extracted from the YouTube video, the above represents the CDN nodes through which video/audio data for the video can be retrieved.

Further, the following image shows the result of downloading this video and audio data and causing client playback on a local media player.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.



Still image of YouTube video retrieved from the URL <https://www.youtube.com/watch?v=cOcEmQgwcl8> showing the video played at the 0:03:36 mark.

PRELIMINARY – This chart is an initial mapping prepared before discovery and prior to any Infringement Contentions.