

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

OXYLABS LT, UAB,
Petitioner,

v.

BRIGHT DATA LTD.,
Patent Owner.

Inter Partes Review of
Patent 11,616,826

DECLARATION OF JAMES OLIVIER, Ph.D

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	PROFESSIONAL BACKGROUND.....	1
III.	COMPENSATION	5
IV.	OVERVIEW OF TASK & BASIS FOR OPINIONS	6
V.	LEGAL CONSIDERATIONS	7
VI.	LEVEL OF ORDINARY SKILL IN THE ART	10
VII.	OVERVIEW OF THE TECHNICAL FIELD & PRIOR ART	12
	A. HTTP-The-Definitive-Guide.....	12
	B. RFC 2616	14
	C. RFC 791	17
	D. RFC 793	20
	E. TCP Keepalive HOWTO	22
VIII.	CLAIM CONSTRUCTION	23
	A. “Client Device” and “First Server”	23
	B. “Server operating system”	25
	C. “selecting, an IP address” and “sending of the selected IP addresses to the first client device”	26
IX.	THE ’826 PATENT	27
X.	OVERVIEW OF PRIOR ART REFERENCES.....	29
	A. U.S. Pat. No. 6,785,705 (“Kocherlakota”).....	29
	B. RFC 1122	39
	C. TCP Keepalive HOWTO	41
	D. U.S. Pat. No. 6,389,462 (“Cohen”)	42
XI.	COMPARISON OF THE PRIOR ART TO THE CLAIMS.....	45
	A. Ground 1 – Kocherlakota Renders Obvious Claims 1-5, 10, 12, 15-19, and 21-25	45
	B. Ground 2 – Kocherlakota in View of RFC 1122 Renders Obvious Claims 6-9. ...	68
	C. Element-by-Element Analysis	71
	D. Ground 3 – Cohen Renders Obvious Claims 1-6, 10, 12, 15-19, 21-25.....	74
	E. Ground 4 – Cohen in View of RFC 1122 Renders Obvious Claims 7-9.	99
	F. Element-by-Element Analysis	101
XII.	CONCLUSION.....	105

PETITIONER’S EXHIBIT LIST

Exhibit	Shorthand	Description
1001	'826 Patent	U.S. Patent No. 11,190,826
1002	Olivier	Declaration of Dr. James Olivier
1003	Kocherlakota	U.S. Patent No. 6,785,705 to Kocherlakota (Aug. 31, 2004) (“Kocherlakota”)
1004	Cohen	U.S. Patent No. 6,389,462 to Cohen (May 14, 2002) (“Cohen”)
1005	RFC 2616	Internet Engineering Task Force, Hypertext Transfer Protocol – HTTP/1.1, June 1999
1006	RFC 1122	Internet Engineering Task Force, Requirements for Internet Hosts – Communication Layers, October 1989
1007	'826 FH	Prosecution History of U.S. Patent No. 11,616,826
1008	'341 FH	Prosecution History of U.S. Patent No. 11,044,341
1009		
1010	Teso CC Order	Claim Construction Opinion and Order, Document 191, Luminati Networks Ltd. v. Teso LT et al., Case No. 2:19-cv-00395 (E.D. Tex.)
1011	Code200 CC Order	Claim Construction Memorandum Opinion and Order, Document 97, Luminati Networks Ltd. v. Code200 et al., Case No. 2:19-cv-00396 (E.D. Tex.)
1012	Motion for Hearing	Defendants’ Motion for Hearing Regarding O2 Micro Issue, Document 444, Bright Data Ltd. v. Teso LT et al., Case No. 2:19-cv-00395 (E.D. Tex.)
1013	Supplemental CC Order	Supplemental Claim Construction Order, Document 453, Bright Data Ltd. v. Teso LT et al., Case No. 2:19-cv-00395 (E.D. Tex.)
1014		
1015	RFC 791	Internet Engineering Task Force, Request for Comments 1122 Internet Protocol, September 1981
1016	RFC 793	Internet Engineering Task Force, Request for Comments 1122 Transmission Control Protocol, September 1981
1017	RFC 1123	Internet Engineering Task Force, Request for Comments 1123 Requirements for Internet Hosts - Application and Support, October 1989

Exhibit	Shorthand	Description
1018	Coley	U.S. Patent 5,826,014 (issued Oct. 20, 1998)
1019	Haverstock	U.S. Patent No. 6,449,640 (issued Sep. 10, 2002)
1020	RFC Index	August 3, 2001 archived Internet Society webpage for RFC Index from Internet Archive (Wayback Machine), available at: http://web.archive.org/web/20010803120225/http://www.rfc-editor.org:80/rfc-index2.html
1021	Keepalive HowTo	June 11, 2008 archived Internet Society webpage for TCP KEEPALIVE HOWTO, available at https://web.archive.org/web/20080611153148/https://tldp.org/HOWTO/TCP-Keepalive-HOWTO/index.html
1022	Masters	U.S. Patent No. 8,108,554 to Masters (filed Oct. 27, 2008)
1023	Definitive Guide	Gourley, D., Totty, B. (2002). HTTP: The Definitive Guide. Germany: O'Reilly Media, Incorporated. Available at https://archive.org/details/httpdefinitivegu00gour/page/n9/mode/2up

Claim Listing

Claim	Limitation No.	Limitation
1	1[p]	A method for use with a web server that stores a first content identified by a first content identifier, and for use with a first client device that is addressed over the Internet using first Internet Protocol (IP) address, for use with a list of IP addresses stored in the first server, the method by the first server comprising:
	1[a]	storing, operating, or using, a server operating system;
	1[b]	receiving, from the first client device, the first content identifier;
	1[c]	selecting, an IP address from the list according to a criterion;
	1[d]	sending, in response to the receiving of the first content identifier, using the selected IP address, the first content identifier;
	1[e]	receiving, in response to the sending of the first content identifier, the first content from the web server; and
	1[f]	sending the received first content, or a part thereof, to the first client device.
2	2	The method according to claim 1, further for use with a second client device that is addressable in the Internet using the selected IP address.
3	3	The method according to claim 2, wherein the sending of the first content identifier comprises sending to the second client device.
4	4	The method according to claim 3, further comprising: receiving, by the second client device from the first server, the first content identifier; sending, by the second client device from the web server, the first content; and sending, by the second client device to the first server, the received first content or a part thereof.

Claim	Limitation No.	Limitation
5	5	The method according to claim 2, further comprising, by the first server: communicating with the second client device; and storing an IP address of the second client device in the list in response to the communicating.
6	6	The method according to claim 1, for use with a second client device, the method further comprising: sending, by the first server, a 'keep alive' message to the second client device; and waiting, by the first server, for a response to the 'keep alive' message from the second client device.
7	7	The method according to claim 6, further comprising: receiving, by the first server, the response to the 'keep alive' message from the second client device; and responsive to the receiving of the response, adding the IP address of the second client device to the list.
8	8	The method according to claim 6, further comprising responsive to not receiving, by the first server, a response to the 'keep alive' message from the second client device, removing the IP address of the second client device from the list.
9	9	The method according to claim 6, wherein the sending of the 'keep alive' message comprises periodically sending by the first server the 'keep alive' message to the second client device.
10	10	The method according to claim 1, wherein the criterion is stored in the first server.
11	11	The method according to claim 10, wherein the criterion is based on, or comprises, a numerical closeness to the web server IP address.
12	12	The method according to claim 10, wherein the criterion is based on, or comprises, a geographical location of a client device addressed by the selected IP address, or a response time when communicating with the client device addressed by the selected IP address.

Claim	Limitation No.	Limitation
13	13	The method according to claim 1, for use with a software application stored in the first server and associated with a first version number, the method further comprising sending, by the first server to the first client device, a message that comprises the first version number of the software application.
14	14	The method according to claim 13, further comprising sending, by the first server to the first client device, the first software application.
15	15	The method according to claim 1, wherein the first server is a Transmission Control Protocol/Internet Protocol (TCP/IP) server that communicates over the Internet with the first client device based on, according to, or using, TCP/IP protocol or connection.
16	16	The method according to claim 1, wherein the first server communicates over the Internet based on, or according to, one out of UDP, DNS, TCP, FTP, POP*, SMTP, or SQL standards.
17	17	The method according to claim 1, wherein the first content comprises web-page, audio, or video content, and wherein the first content identifier comprises a Uniform Resource Locator (URL).
18	18	The method according to claim 1, for use with a plurality of client devices that includes a second client device, each identified over the Internet using a respective IP address from the list, wherein the method further comprising: selecting, by the first server, multiple client devices from the plurality of client devices by selecting the IP addresses of the multiple client devices from the list; and sending, by the first server, the selected IP addresses in response to the selecting.
19	19	The method according to claim 1, further comprising sending of the selected IP addresses to the first client device.

Claim	Limitation No.	Limitation
20	20	The method according to claim 1, further comprising responsive to the receiving of the first content identifier from the first client device, <u>adding IP address of the first client device to the list.</u>
21	21	The method according to claim 1, wherein the first client device is identified by a <u>first Internet Protocol (IP) address, Media Access Control (MAC) address, or a hostname.</u>
22	22	The method according to claim 1, wherein the web server is a Hypertext Transfer Protocol (HTTP) server responding to HTTP requests.
23	23	The method according to claim 1, wherein the steps are sequentially executed.
24	24	The method according to claim 1, for use with a software application that includes computer instructions that, when executed by a computer processor, cause the processor to perform the steps of the claim 1.
25	25	A non-transitory computer readable medium containing computer instructions that, when executed by a computer processor, cause the processor to perform the method according to claim 1.

I. INTRODUCTION

1. My name is James L. Olivier, Ph.D. I have been retained by Metacluster Lt, Uab as an independent expert in the relevant art.

II. PROFESSIONAL BACKGROUND

2. I have a Ph.D. from The Ohio State University in Electrical Engineering with minors in Discrete Mathematics, Computer Science, and Microelectronics. I have published papers in the areas of coding theory and multiprocessor computer systems. I received my Bachelor of Science degree from Ohio State University in 1983. My CV is attached hereto as Appendix A, which shows my education and experience in more detail.

3. I have extensive experience in the design and development of telecommunication systems and have specialized experience in network product development.

4. I have developed and designed equipment for cellular networks since I first joined AT&T Bell Laboratories in 1990. From 1990 to 2005 I also designed and implemented a variety of telecommunication systems, including Asynchronous Transfer Mode (“ATM”) based systems for use in cellular networks. ATM is a broadband, packet-switched telecommunications standard defined by the American National Standards Institute (“ANSI”) and the International Telecommunication Union (“ITU”). I also participated in the design and development of AT&T

Autoplex Series base stations. I also participated in the design and development of AT&T's Network Control Points, "NCPs" which were massive network databases containing subscription information for customers. During this time, I first began my work with telecommunication standards bodies as a contributing member of the ATM Forum.

5. In 1995, I transitioned to the Digital Switch Corporation ("DSC") Switch Products Division and was the Senior Manager of the ATM systems engineering group developing ATM packet switches for a new generation base station for use by Motorola in their Centralized Base Station Controller, a core part of the second generation cellular network. Later, as Senior Manager for Intelligent Networks, I designed and developed a video residential gateway which was presented at SuperComm 1996.

6. I next worked at Samsung Telecommunications America from 1996 to 1999 as a Principal Engineer for Samsung's next generation wireless broadband switching system over the Universal Mobile Telecommunications System ("UMTS"). UMTS is a third-generation ("3G") broadband standard developed by the 3rd Generation Partnership Project ("3GPP"). My work at Samsung included designing their next generation UMTS cellular switch, which provided both packet and voice services, such as telephony services, wireless services, broadband services, and Internet services. While at Samsung, I also served as Samsung's North

American corporate representative to the ITU, which is the agency of the United Nations responsible for facilitating interconnectivity of information and communication technologies worldwide. The ITU participated in the 3GPP under the ITU's International Mobile Telecommunications 2000 effort. It was there that I participated in the development of standards for advanced wireless networks.

7. After that, I worked for Marconi Communications from 1999 to 2002, where I was responsible for design and development of a number of systems for the access market, including Digital Subscriber Line ("DSL") modems, along with the design of point-to-point wireless systems. I was also responsible for the servers for the Network Management System for the North American Access product division, which provide service provisioning for DSL customers. It was here where I architected a triple play residential gateway which provided voice, video and data to residential subscribers.

8. I joined Navini Networks in 2002 as a Senior Manager. The Navini system was an entirely packet-based cellular system. I was responsible for layer 2 and layer 3 network protocols for Navini's Wideband Code Division Multiple Access ("WCDMA") wireless base stations and broadband modems. These layers were responsible for packet transmissions for various services over the WCDMA air interface. I also developed network-based servers and databases for network authentication, network services, service provisioning and subscriber management

for the Navini's Worldwide Interoperability for Microwave Access 'WiMAX' Network.

9. I have been the owner of Olivier Consulting since 2003 and have provided consulting services to various companies to develop networking systems and on intellectual property matters, including many such matters relating to networking technologies specifically.

10. I am currently a research professor at the Hunt Institute for Engineering and Humanity, which is part of the Lyle School of Engineering at Southern Methodist University in Dallas, Texas. Here, I serve as the Program Lead for Transformational Technology. As Program Lead, I am responsible for the Maps4Good project which uses SMU's data center servers to provide web-based services for the community, such as a COVID Relief Map.

11. I am also currently an Adjunct Professor in the Telecommunications and Network Engineering Program at Southern Methodist University's Graduate School of Electrical Engineering, where I teach classes in networking and networked enabled applications.

12. Overall, I have over 20 years of experience in advanced network engineering and product development. I have been recognized expert by the International Trade Commission ("ITC") in security and firewalls in routers and switches, cellphone hardware and software for imaging, operating systems for

mobile devices, mobile devices hardware and software architectures, and ethernet switch packet processor hardware and software.

13. I have also opined on the meaning of claim terms to one of ordinary skill in the art on numerous occasions. For example, I provided opinions on the meanings of claim terms in QPSX Developments 5 Pty Ltd. v. Nortel Networks, Inc and In the Matter of Certain Wireless Devices, Including Mobile Phone and Tablets II, USITC Inv. No. 337-TA-905.

14. My CV, which I understand is being submitted with this declaration, includes a listing of patents applications on which I am an inventor.

15. Because of my background, training, and experience, I am qualified as an expert to explain the background of the technology encompassed by the '826 Patent as well as to opine on the validity of the claims of the '826 Patent in view of the prior art references discussed herein.

16. If called upon to do so, I could and would testify truthfully regarding the points stated below in this declaration.

III. COMPENSATION

17. My work on this matter is being billed at my customary rate of 650.00 per hour. Also, I am being reimbursed for reasonable expenses I incur in relation to my services. I will be paid regardless of the outcome of any proceeding that my work is used in.

IV. OVERVIEW OF TASK & BASIS FOR OPINIONS

18. I have been asked to review the '826 Patent. I have been asked to provide opinions related to certain issues from the perspective of a person of ordinary skill, having knowledge of the relevant art, as of October 8, 2009, and—except where otherwise noted—the opinions stated in this declaration are from that perspective. The qualifications and abilities of such a person are described in paragraph 33 below.

19. More particularly, I have been informed that this proceeding is currently at a preliminary stage during which the Patent Trial and Appeal Board will decide whether the Petition establishes a reasonable likelihood of at least one claim of the '826 Patent being invalid.

20. My opinions are based on my education, training, and experience as well as items that I reviewed to prepare my opinions, including the documents listed in the Exhibit List included with the Petition and any other items I reference in my below analysis.

21. I understand that other issues may arise that require further explanation, and I will provide that explanation if appropriate. As a result, I respectfully reserve the right to update and supplement this declaration and the information and opinions provided herein.

V. LEGAL CONSIDERATIONS

22. I am not an attorney. My understanding of the law is based on information provided by counsel for Petitioner.

23. My opinions address what would have been logical to, and within the skill level of, a person of ordinary skill in the art (“POSITA”) at the time of the alleged invention, given the state of the relevant art, the knowledge and skill that a POSITA would have, the teachings of the references discussed below, and how a POSITA would have understood those teachings. My opinions also address whether a POSITA would have had a reasonable expectation of the modified systems discussed below successfully yielding the results discussed below. I understand that a POSITA is a hypothetical person. He is not a genius. He thinks along the lines of conventional wisdom. I understand that a POSITA is also a person of ordinary creativity, not an automaton (*see* Section VI (Level of Ordinary Skill in the Art)), below, for qualifications of a POSITA). I also understand that in determining who would qualify as a POSITA, it is appropriate to consider criteria such as: (a) the type of problems encountered in the art; (b) prior art solutions to those problems; (c) the rapidity with which innovations are made; (d) the sophistication of the technology; and (e) the education level of active workers in the field. I understand that a given reference would have been relevant for a POSITA to consider if it came from the same field of endeavor or was reasonably related to a problem faced by the ’826

Patent inventors, and that this relevance determination underlies a principle known as “analogous art,” which should be construed broadly.

24. **First**, turning to the types of problems and prior art solutions in the art, the “Field of the Invention” to the specification of the ’826 Patent states that the patent “relates to Internet communication, and more particularly, to improved data communication speed and bandwidth efficiency on the Internet.” EX1001, 1:23-25. The “Background” section further explains that the patent pertains to the increasing bandwidth required for video web sites for shopping news portals and social networks. *Id.*, 1:35-38. The ’826 describes two different prior art solutions to this problem, proxy servers which cache Internet content such as Akamai and peer-to-peer file sharing solutions such as BitTorrent.

25. **Second**, turning to the sophistication of the technology and the rapidity with which innovations occur in the field, the technical knowledge required to understand techniques to improved data communication speed and bandwidth efficiency on the Internet at the time of the alleged invention required education and experience in Internet communications technology to understand. For example, the ’826 Patent specifically discusses the knowledge of a POSITA in regards to HTTP, TCP/IP, IP addresses, and Request for Comments (“RFC”) 2616. See ’826 Patent, 16:21-28, 17:20-24. A person skilled in the art would have been monitoring these developments.

26. **Third**, with respect to the educational level of active workers in the field, I personally observed that the individuals that participated in development and implementation of Internet routers/switches, network communication and Web servers typically bachelor's degree in Computer Science or related field (or equivalent experience), and at least two years' experience working with and programming networked computer systems.

27. **Fourth**, turning to the educational level of the named inventors, the '826 Patent names two inventors: Derry Shribman and Ofer Vilenski. I am not personally acquainted with any of these individuals, but I located information from their LinkedIn profiles that I reasonably believe to be associated with these individuals:

28. Derry Shribman's profile indicates that he has a bachelor's degree in Computer Science from the Open University and at least two years' experience working with and programming networked computer systems, including designing or implementing of Web, Internet, or network communication systems through with work companies including Cisco. <https://www.linkedin.com/in/derryshribman/details/experience/>

29. Ofer Vilenski's profile indicates that he has bachelor's degree in Computer Science or related field (or equivalent experience) from Israel Institute of Technology, and at least two years' experience working with and programming

networked computer systems, including designing or implementing of Web, Internet, or network communication systems through with work companies including Cisco. <https://www.linkedin.com/in/ofervilenski/>

30. I have not been asked to take a position on whether a given claim would have been legally obvious to a POSITA at the time of the alleged invention, but I have been told that some or all of my opinions are being used to support the argument that the claims of the '826 Patent are obvious.

31. I understand that a claim term is given the meaning that the term would have to a POSITA at the time of the invention, which generally is the ordinary and customary meaning of the term. I further understand that the ordinary and customary meaning of a term may be evidenced by a variety of sources, including the words of the claims themselves, the specification, the prosecution history, and extrinsic evidence concerning relevant scientific principles, the meaning of technical terms, and the state of the art.

VI. LEVEL OF ORDINARY SKILL IN THE ART

32. I understand that the claims and specification of a patent must be read and construed through the eyes of a person of ordinary skill in the art ("POSITA") at the time of the earliest alleged priority date for the '826 Patent, October 8, 2009 ("Priority Date"). Unless otherwise stated, this declaration concerns the viewpoint of a POSITA with respect to the '826 Patent.

33. In my opinion, a person of “ordinary skill in the art” to which the ’826 Patent pertains would have had at least a bachelor’s degree in Computer Science or related field (or equivalent experience), and at least two years’ experience working with and programming networked computer systems, including designing or implementing of Web, Internet, or network communication systems. A POSITA would also have familiarity with the Internet Engineering Task Force (IETF) standards and protocols relating HTTP, TCP/IP, IP addresses. This definition is approximate, and more education may substitute for industry experience or vice versa.

34. In addition, in my opinion, while a person of ordinary skill in the art would have some familiarity with such underlying technologies, that familiarity would need only be sufficient to utilize the technology in connection with a broader system and method. For example, a person of ordinary skill in the art would be aware of how networked communication takes place over the Internet and how HTTP requests are processed in accordance to standards, but would not necessarily need to be an expert on the design of the Web servers or Internet caches.

35. Based on my educational and employment background, I am qualified to provide opinions concerning what a POSITA would have known and understood by October 8, 2009. Indeed, as reflected in my qualifications above, I am more than

qualified as a person of ordinary skill in the art as of the relevant date of the '826 Patent.

VII. OVERVIEW OF THE TECHNICAL FIELD & PRIOR ART

36. In this section, I describe the field of technology prior to the Priority Date of the '826 Patent in October of 2009. This would be indicative of the working knowledge of a POSITA at that time. Further, because the following discussion relates to Internet protocols, Internet communication, and proxies, this directly relates to the subject matter of the '826 Patent. Indeed, the '826 Patent specifically discusses the knowledge of a POSITA in regard to HTTP, TCP/IP, IP addresses, and even the IETF RFC 2616 (the standard for the HTTP/1.1 protocol, published in 1999), as I further discuss later in this Declaration. Also see, EX1001 (56) page 7; EX1023, Chapter 1; and EX1005, 1/114.

A. HTTP-The-Definitive-Guide

37. 'HTTP-The-Definitive-Guide' published in 2002 and describes that proxies were well known by October of 2009 and devotes an entire chapter to the subject. (See EX1023, Chapter 6.) In particular, the 'HTTP-The-Definitive-Guide' teaches that proxies are both clients and servers stating, ("HTTP proxy servers are both web servers and web clients. Because HTTP clients send request messages to proxies, the proxy server must properly handle the requests and the connections and return responses, just like a web server. At the same time, the proxy itself sends

requests to servers, so it must also behave like a correct HTTP client, sending requests and receiving responses (see Figure 6-1).)” (EX1023, Page 130). See also Figure 6-1 shown below. (EX1023, Page 130).

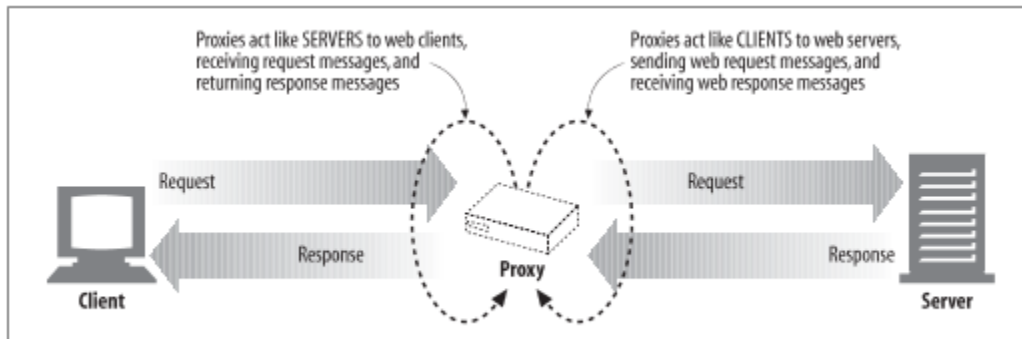


Figure 6-1. A proxy must be both a server and a client

38. The ‘HTTP-The-Definitive-Guide’ also teaches the use of chains of proxies in a proxy hierarchy. (“Proxies can be cascaded in chains called proxy hierarchies. In a proxy hierarchy, messages are passed from proxy to proxy until they eventually reach the origin server (and then are passed back through the proxies to the client, as shown in Figure 6-12), (EX1023, Page 138). Figure 6-12 is shown below. (EX1023, Page 139)

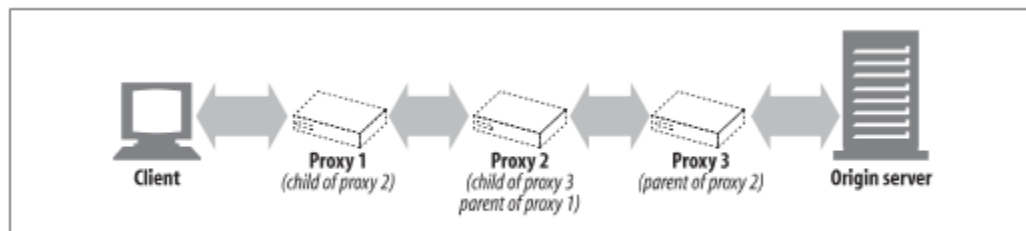


Figure 6-12. Three-level proxy hierarchy

39. The HTTP-The-Definitive-Guide goes on to teach that the selection of additional proxies can be based on load balancing, subscriptions or geographic

proximity routing, where a child proxy may select a parent responsible for the origin server's geographic region. (See EX1023, page 140.)

B. RFC 2616

40. Web communications and routing were well-known prior to the Priority Date. For example, RFC 2616, which the '826 Patent recognizes “define[s]” “the HTTP protocol,” was published ten years earlier by the HTTP Working Group of the Internet Engineering Task Force (IETF) in June 1999. EX1001, 16:21-22; EX1005, 1/114. I obtained a copy of RFC 2616 from the IETF website source that I understand to be the repository of IETF's RFCs, and which a POSITA would commonly utilize to obtain an electronic copy of the IETF's RFCs. The copy of RFC 2616 that I obtained from <https://datatracker.ietf.org/doc/html/> RFC 2616 is attached as EX1005.

41. RFC 2616 explains that “the HTTP protocol is a request/response protocol” and that “most HTTP communication is initiated by a user agent [(UA)] and consists of a request to be applied to a resource on some origin server [(O)].” EX1005, 8/114. “HTTP communication usually takes place over TCP/IP connections.” EX1005, 11/114. RFC 2616 also confirms the role-based constructions for “client” and “server” as discussed in VIII.A are correct: “Any given program may be capable of being both a client and a server; *our use of these*

terms refers only to the role being performed by the program for a particular connection.” EX1001, 16:21-22; EX1005, 8/114.

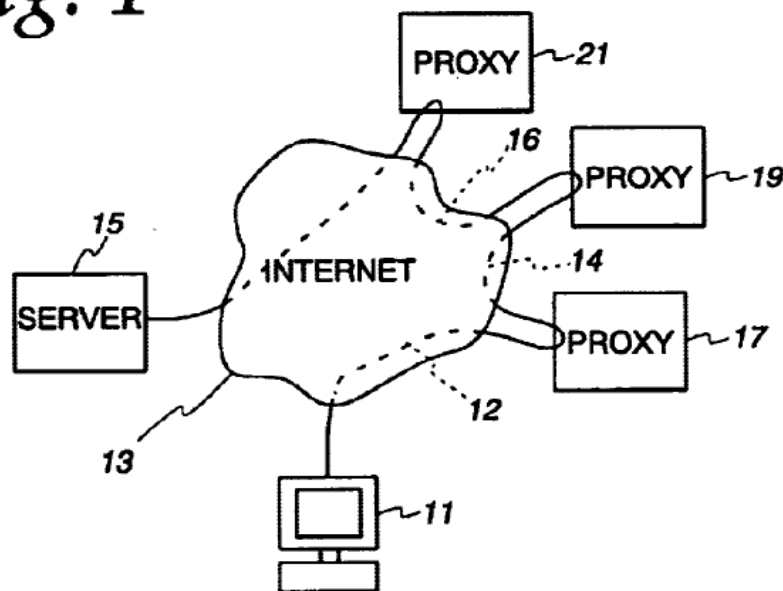
42. In additional to describing “client” and “server” operations, RFC 2616 discloses multi-hop routing using “proxy” devices running “intermediary program which acts as both a server and a client for the purpose of making requests on behalf of other clients.” EX1005, 8/114. That is, while communication may be “accomplished via a single connection (v) between the user agent (UA) and the origin server (O),” “more complicated situation[s]” were also known “when one or more intermediaries are present in the request/response chain,” such as an illustrated chain with “three intermediaries (A, B, and C).”

```
request chain ----->
UA -----v----- A -----v----- B -----v----- C -----v----- O
<----- response chain
```

Id., 10/114.

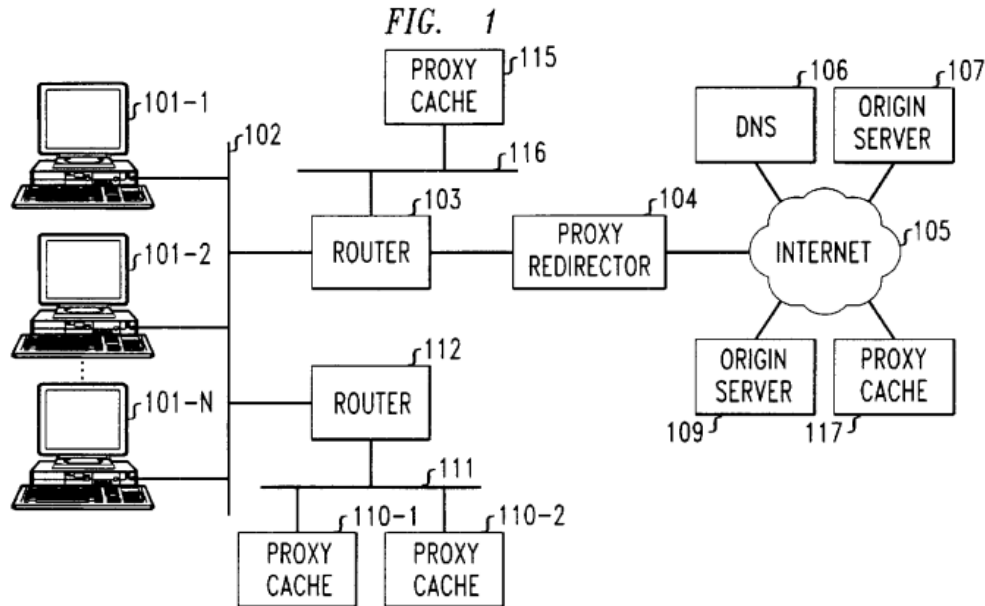
43. Prior art references, including Kocherlakota, confirm that it was well-known for an intermediary to determine routing of a received request based on that request. For example, Kocherlakota discloses proxy 17 receiving URLs from client computer 11 and selecting the IP address for forwarding to proxy 19, proxy 21, or web server 15 based on the URL. EX1003, 4:15-52, 5:18-24.

Fig. 1



Id., FIG. 1.

44. Likewise, Cohen discloses “proxy redirector 104” intercepting web requests from “client 101-1” and “select[ing] one of the available proxy caches to which to forward client requests based ... IP header information such as the origin server IP address.” EX1004, 7:36-48; FIG. 1.



Id., FIG. 1.

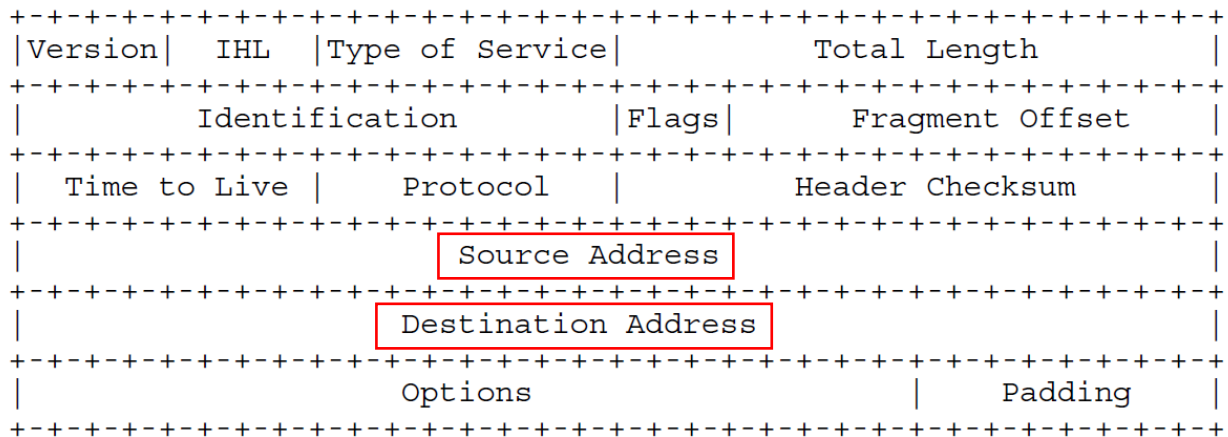
45. The ‘826 patent itself also supports this understanding of a POSITA in citing to “HTTP-The-Definitive-Guide”, which states, (“HTTP proxy servers are both web servers and web clients.”) See EX1023 at page 130. See also EX1001 page 7.

C. RFC 791

46. HTTP’s relationship to other well-known Internet standards was also known. For example, IP is an example of a well-known Network Layer protocol. In

particular, IPv4 was published as an Internet Standard in RFC 791 in September 1981. This RFC serves as the informational drafts and formal standards for many Internet protocols, and are published by the Internet Engineering Task Force (IETF). RFC 791's front page indicates that it is a September 1981 publication that served as the "DARPA Internet Program Protocol Specification" for the "Transmission Control Protocol." EX1015, 1. It was posted on the Internet and published in the ordinary course by the IETF, and was meant to be viewed by the interested Internet engineering audience at large as of its date of publication stated on its cover. I obtained a copy of RFC 791 from the IETF website source that I understand to be the repository of IETF's RFCs, and which a POSITA would commonly utilize to obtain an electronic copy of the IETF's RFCs. The copy of RFC 791 that I obtained from <https://datatracker.ietf.org/doc/rfc791/> is attached as EX1015.

47. The IP protocol (such as described in IPv4) describes a set of rules for defining a IP packet format, including a header with network source and destination address information, and a datagram body (some fixed-maximum-length message), and then transmitted those IP packets from an IP sender to an IP destination across links in a computer network. As shown below, IP Packets include a header that includes a "source address" and "destination address":



Example Internet Datagram Header

Figure 4.

EX1015, 17/51; *see also* EX1016, 57/91 (in TCP, “[a]ny lower level protocol will have to provide the source address, destination address”).

48. The datagram body, or message, carried within a single IP packet typically has a maximum length of around 1500 bytes, which corresponds to a few paragraphs of text. EX1015, 13/51. The IP protocol only offers “best effort” delivery of IP packets: An IP sender can transmit an IP packet to an IP recipient (based on the IP addresses of sender and recipients that are in the packet’s headers), and the various networking routers will “attempt” to forward the packet to its proper destination, but the packet can be arbitrarily corrupted, dropped, or delayed. Because of this, when multiple IP packets belonging to the same “stream” – such as part of a web transfer between a webserver and a web browser – are sent from a sender to recipient, the recipient may only receive some of the packets (others may be lost)

and the packets may be received “out of order” from when sent (because some packets were delayed by the network or took a different path through the network than others). These limitations by the IP layer in providing a “reliable stream” are solved by the TCP protocol.

D. RFC 793

49. The TCP protocol has also been well-known to a POSITA for quite some time, having been standardized in September 1981 in the IETF RFC 793. EX1016. RFC 793’s front page indicates that it is a September 1981 publication that served as the “DARPA Internet Program Protocol Specification” for the “Transmission Control Protocol.” EX1016, 1. It was posted on the Internet and published in the ordinary course by the IETF, and was meant to be viewed by the interested Internet engineering audience at large as of its date of publication stated on its cover. I obtained a copy of RFC 793 from the IETF website source that I understand to be the repository of IETF’s RFCs, and which a POSITA would commonly utilize to obtain an electronic copy of the IETF’s RFCs. The copy of RFC 793 that I obtained from <https://datatracker.ietf.org/doc/rfc793/> is attached as EX016.

50. TCP serves as a transport layer that makes the “best effort” packet delivery of the IP layer more usable by applications, such as Web browsing using HTTP. Specifically, TCP, builds on and adds functionality to the lower-level IP

network layer by segmenting the stream of data is segmented into individual fixed-sized segments, where each segment is carried within a separate IP packet. Each segment is also associated with a sequence number, which allows a receiver to place IP packets that are received out-of-order back into the proper stream order. The sequence numbers also help receivers understand if certain segments are missing, and they subsequently can re-request missing segments from the sender, who retransmits these missing segments in order to ensure reliable delivery. As a result, most Internet traffic uses TCP as a reliable, stream-based protocol that runs on top of IP networks. For these reasons, Internet traffic is often described as “TCP over IP” or simply “TCP/IP.” When that traffic happens to also use HTTP as the application layer protocol, it is often described as “HTTP over TCP/IP.” *See* EX1005, 11/114 (stating that “HTTP communication usually takes place over TCP/IP connections”); EX1001, 17:20-26 (describing “TCPIP” as “known” and referencing a “Web server” that “is a TCPIP server”).

51. TCP also provides connection management, reliable retransmission, reordering, and other features like flow and congestion control as a component in the operating systems running on the sender and receiver, such as the web server and web browser. TCP is thus implemented on basically every common operating system, including Microsoft Windows, Apple Mac OS (including iOS running on iPhones), Linux (including Android for mobile), the BSD operating system, and

every UNIX variant. *See* EX1016, 14/191 (stating that “TCP is assumed to be a module in an operating system. The users access the TCP much like they would access the file system. The TCP may call on other operating system functions, for example, to manage data structures. The actual interface to the network is assumed to be controlled by a device driver module. The TCP does not call on the network device driver directly, but rather calls on the internet datagram protocol module which may in turn call on the device driver.”).

52. Finally, advanced features of TCP (as discussed below) were also known, notably the use of a “‘keep-alive’ mechanism” “[t]o confirm that an idle connection is still active.” EX1006, 102/116.

E. TCP Keepalive HOWTO

53. How to implement the keep alive mechanism in the TCP protocol was also well-known to a POSITA as discussed in “TCP Keepalive HOWTO” published on 2007-05-04. EX1021, at 1. I retrieved this document from the Internet Archive Wayback Machine for the URL <https://tldp.org/HOWTO/TCP-Keepalive-HOWTO/index.html>, for the date June 11th, 2008. This document is from the well known “The Linux Documentation Project” HOWTO guides. This document teaches how one would implement a TCP keep alive in a Linux operating system. In particular, ‘TCP Keepalive HOWTO’ teaches that setting up and using TCP keepalives is very simple. *See* EX1021, Section 2.1. A timer is set up and when the

timer reaches zero a problem packet is sent. If you receive a reply to the probe packet that connection is assumed to be up. (*Id.* Section 2.1) If you do not receive a reply from the probe packet, a count is kept of the unacknowledged probes, if this count exceeds a value “tcp_keepalive_probes”, the connection is considered dead. (*Id.* Section 3.)

VIII. CLAIM CONSTRUCTION

54. For purposes of this IPR proceeding, I apply the common and ordinary meanings of claim terms in the '826 Patent, unless otherwise noted. For example, I note that the Court issued three claim construction orders in the related litigation construing terms of the '511, '968, '319, and '510 Patents, which have the same specification and share claim language with the Challenged Claims, and are relevant to construction of the Challenged Claims. I specifically discuss certain claim terms below.

A. “*Client Device*” and “*First Server*”

55. The Challenged Claims use the same “client device” and “first server” terms as the claims of the '511 Patent and related patents. EX1011, 5/37; EX1010, 3-4/24. Across three Orders the Court repeatedly confirmed that “client” and “server” refer to the role the device plays, not any hardware characteristics or label.

56. In each of the *Teso* (for the '319 and '510 Patents) and *Code200* (for the '511 and '968 Patents) Orders, the Court held that “client device” means

“communication device that is *operating in the role of a client*” and rejected PO’s contention that the term required a “consumer computer.” EX1010, 10-12/24; EX1011, 11-13/37.

57. As to “first server,” in the *Code200* Order, the Court addressed “whether one component can simultaneously serve as more than one of: the client device, the first server[], and the web server,” concluded “[i]t cannot,” and construed “first server” as “server that is not the client device or the web server.” EX1011, 13-15/37. Subsequently, it became apparent that “server” required further construction, and Petitioner requested a clarified construction of: “a device that is operating in the role of a server and that is not the first client device or the web server.” EX1012, 2/5 (emphasis in original). The District Court declined to formally change the construction, but confirmed “Defendants’ understanding of the scope of the construction[], as *represented by the requested clarification*[] ..., *is correct*” and “already embedded in” the original construction. EX1013, 7-11/12. The Court confirmed that “a component can be *configured* to operate in different roles—so long as it does not ‘simultaneously serve as more than one of: the client device, the first server/second server, and the web server.’” *Id.*, 10/12 (emphasis in original).

58. RFC 2616, which the ’826 Patent confirms “define[s]” “the HTTP protocol,” further confirms that the role-based constructions are correct: “Any given program may be capable of being both a client and a server; *our use of these terms*

refers only to the role being performed by the program for a particular connection.”

EX1001, 16:21-22; EX1005, 8/114.

59. RFC 2616 also provides further guidance on each of these “roles,” defining a “client” as “establish[ing] connections for the purpose of sending requests” and a “server” “accept[ing] connections in order to service requests by sending back responses.” EX1005, 8/114. RFC 2616 explains that a “proxy,” the component of Kocherlakota and Cohen mapped to various claim elements, is an “intermediary program which *acts as both a server and a client* for the purpose of making requests on behalf of other clients.” EX1005, 8/114. These RFC 2616 statements would have carried significant weight with a POSITA.

B. “Server operating system”

60. The ‘826 Specification does not use the term “server operating system” or even describe anything more than ‘an operating system’. Because the term server has been construed as ‘a device that is operating in the role of a server’, (Section VIII.A (Client device and first server), above) the term server operating system should be construed as an operating system capable of operating in the role of a server.

61. Should the PO disagree, I will also conduct my analysis under the plain and ordinary meaning of ‘server operating system’ where a server operating system can be a general-purpose server operating system such as, e.g., UNIX, LINUX™,

Microsoft WINDOWS NT. See EX1022, Col. 8:53-55 (“A general-purpose **server operating system** may be employed, e.g., UNIX, LINUXTM, Microsoft WINDOWS NT®, or the like.”). The construction of this term does not change my analysis, below, as each of the grounds would disclose a “server operating system” under either interpretation.

C. “selecting, an IP address” and “sending of the selected IP addresses to the first client device”

62. Limitation 1 recites “selecting, an IP address” and using “the selected IP address.” See Limitation 1[c], 1[d]. Limitation 19 depends directly from Limitation 1 and recites “sending of the selected IP addresses.” Thus, there is an inconsistency in the claim language between the selected IP address (singular) in claim 1 and the selected IP addresses (plural) in claim 19. Under its plain meaning, a POSITA would have understood sending *the* selected IP addresses to correspond to sending the IP address selected in claim 1.

IX. THE '826 PATENT

63. The '826 Patent specification describes system of caching proxies to “improv[e] data communication speed and bandwidth efficiency on the Internet.” EX1001, 1:24–25. FIG. 3 of the '826 Patent, copied below, depicts “peer[s],” a “client,” and an “agent” communicating, with the “agent” forming a connection to the web server 152.

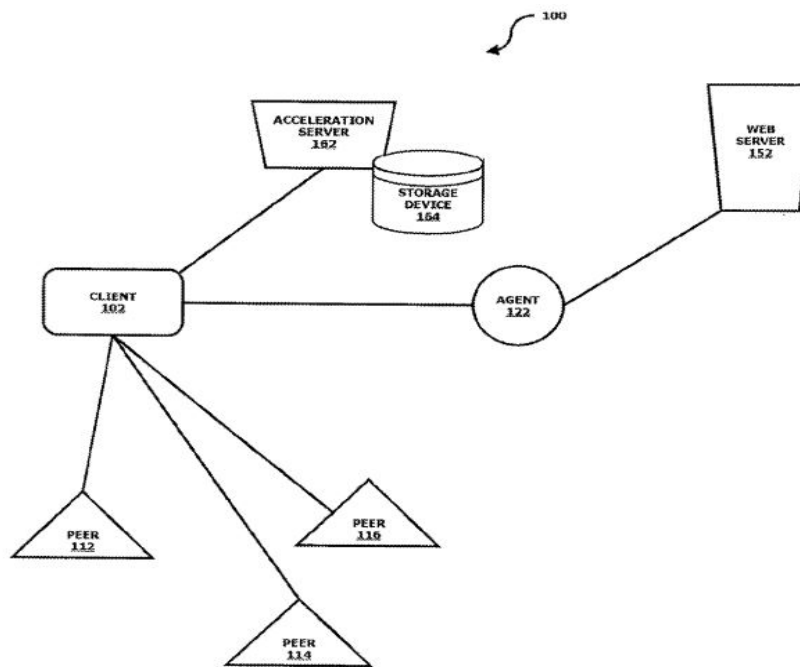


FIG. 3

64. To initiate communications, the client 102 sends a web request to acceleration server 162, which responds with a list of agents that may assist. *Id.*, 5:10-19, 13:8-22. Client 102 then contacts agent 122, which responds with a list of peers that have cached the requested web page, and client 102 contacts an identified peer to obtain the content. *Id.*, 13:62-64, 14:13-58, 15:14-20. The only device disclosed as contacting a web server is agent 122, which does so only when agent

122 does not identify any peers with cached content. *Id.*, 14:62-15:6. In that case, agent 122 directly contacts the web server, obtains the content, and responds to client 102 identifying itself as the only available peer—there is no “selecting” by either client 102 or agent 122 in this process. *Id.*, 15:6-11

65. With respect to FIG. 3, the patent states “[t]he network 100 of FIG. 3 contains multiple communication devices,” that “*each communication device may serve as a client, peer, or agent,*” and that “a detailed description of a communication device is provided with regard to the description of FIG. 4.” *Id.*, 4:44-53 (emphasis added). The patent specification confirms that standard, off-the-shelf features of the “communication device,” including that the “communication device 200 includes a processor 202, memory 210, at least one storage device 208” *Id.*, 5:59-60. The specification also confirms other standard features of the “communication device,” including that its memory may include “ROM, hard drive, tape, CDROM, etc.” and that its input/output devices may include “a keyboard, mouse, scanner, micro-phone, etc.” or a printer. *Id.*, 6:14-24, 6:61-7:3. Each communication device is also comprised of a “cache database 282,” that:

“stores responses to HTTP requests that the communication device 200 has dispatched . . . the responses to HTTP requests are stored within the cache database 282 for future use by this communication device 200, or for other communication devices within the communication network 100 that need to retrieve this information and will use this communication device as either a peer or an agent. *Id.*, 8:10-20; *see also Id.*, 7:60-8:4, 8:24-37, 9:60-10:21, 10:47-60, 11:4-11, and Fig. 7.

66. The '826 Patent only describes 'keep alive' signals in passing stating "As shown by block 552, the acceleration server sends "keep alive" signals to the network elements, and keeps track within its database as to which network elements are online." *Id.* 16:52-55. The '826 Patent does not explain how these keep alive signals operate, and therefore is left to one of ordinary skill.

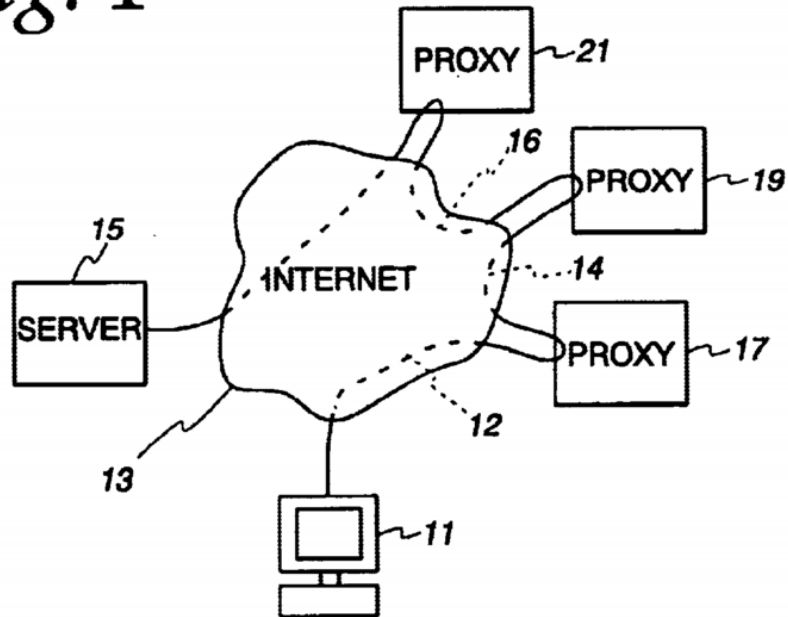
67. The '826 Patent concludes with independent claim 1 and dependent claims 2 – 21.

X. OVERVIEW OF PRIOR ART REFERENCES

A. U.S. Pat. No. 6,785,705 ("Kocherlakota")

68. Kocherlakota "relates to the use of web proxy servers by internet clients and more particularly, to the use of multiple such proxies contemporaneously." EX1003, 1:7-9. FIG. 1 "shows an internet connection through a plurality of chained proxies." *Id.*, 2:39-40.

Fig. 1



69. The client computer 11 is connected to the Internet for accessing content from web servers by sending URLs. (“the client can access the internet by sending URLs ...) *Id.*, 2:5-6. Web server 15 of FIG. 1 represents all addressable web servers on the Internet.

(“FIG. 1 is a web server 15 which is representative of any of 60 the well known data file providers and, in fact, represents the universe of addressable web servers.”) *Id.*, 2:59-61.

70. Client computer 11 can make use of proxy services provided by proxies 17, 19 and 21, while accessing the Internet content. Such services include anonymity services, language services or wait limiting services.

(“Client 11 desires to take advantage of services provided by a plurality of web proxies. For example, the client wants to be anonymous so a connection via an **anonymity web proxy 17 is desirable**. Also, the client would like to search in the French language and therefore wants to take advantage of the **language service provided by a French language web proxy 19**. Lastly, the client would like to reduce the amount of time spent retrieving data so the **services of a wait limiting proxy 21** are desired.”) *Id.*, 2:61 – 3:3.

71. To use these services while accessing content from Web servers, the client computer makes use of an applet provided by a proxy server. Proxy 17 transmits an applet to the client which provides a GUI interface for the client that presents a number of information lines into which the client can enter identities of proxy servers.

(“The client first sends out the URL of the first proxy server 17 for anonymity services in step 101 of FIG. 3. Proxy server 17 receives the contact from client 11 and establishes a session (step 109, FIG. 4) between the client and proxy 17. As a part of the session, web proxy 17 returns an applet to 25 client 11 where it is executed as is well known. The applet, which is received and executed in a step 103 (FIG. 3) presents a window as shown in FIG. 2, on the client computer. The applet window presents a number of information lines into which the client can enter identities of up to three proxy servers,..”) *Id.*, 3:20-33.

Fig. 2

PROXY CHAIN					
FIRST PROXY:	LPWA.COM	201	PORT:	8000	205
USER ID:	CLIENT 1	203	PASSWORD:	12345	207
					ADV. 241
SECOND PROXY:	FRENCH.COM	209	PORT:	100	211
USER ID:	CLIENT 1	213	PASSWORD:	23456	215
					ADV. 243
THIRD PROXY:	FAST.COM	217	PORT:	300	219
USER ID:	CLIENT 1	221	PASSWORD:	34567	223
					ADV. 245
OK	225		ADD PROXY	236	

72. When as shown in FIG. 2, the user identifies a second and third proxy, “the first proxy” receives the configuration information from the applet “and begins to establish sessions with the additional proxies.” *Id.*, 4:43-45.

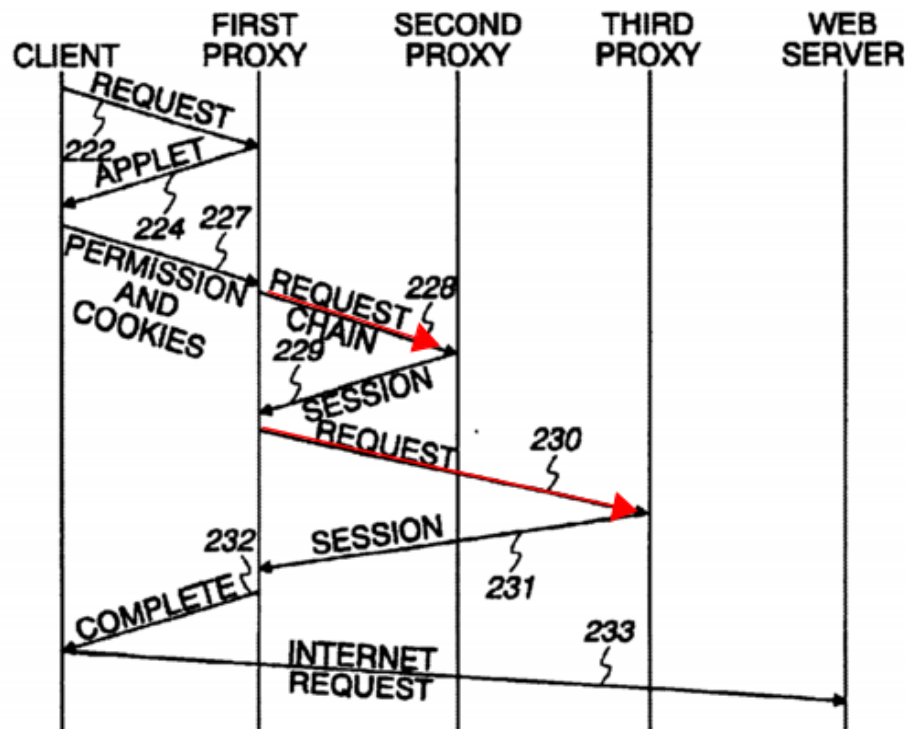
73. Specifically, proxy 17 sends a request to the second proxy which authenticates it and returns a message to the first proxy signifying the session establishment. The first proxy then sends a message to the third proxy requesting a session. The third proxy responds by returning a session message to the first proxy signifying the establishment of another session. In this way the proxy server 17 is connected to second and third proxies with sessions.

(“The identities and permission data are then sent 227 to the **first proxy** which parses the message and **begins to establish sessions with the additional proxies**. A request is sent 228 to the second proxy which authenticates it and returns a message 229 to the first proxy signifying the session establishment. **The first proxy** then sends a message 230 to the **third proxy requesting a session**. The third proxy responds by returning a Session message 231 to the first proxy and the first proxy signifies 232 to the client that the chain is complete.”) *Id.*, 4:42:52.

(“In step 113 the **proxy 17 contacts the named second proxy server, i.e., french.com, to establish a session therewith**. The contact is received in step 119 (FIG. 5) and a session will be established in Step 121 using the client

provided applet response information for the second proxy as well as the cookie from the client computer 17 relating to french.com. This session is maintained in step 125. **The first proxy server 17 then identifies in step 115 whether a third proxy** has been named in the applet response information (FIG. 2).”) *Id.*, 4 2-11.

Fig. 6



74. With these sessions established, web server 15 is accessed through a chain of proxies, proxy 17, 19 and 21.

(“After the proxy chain has been established, the URL of web server 15, or any other web server, can be sent from the client to the internet via the chained proxies 17, 19 and 21. The proxy 17 provides the necessary anonymity to the chain, the language proxy 19 provides the French language features and the wait avoidance proxy 21 provides its function for packets transmitted on the chain.) *Id.*, 4:29-35

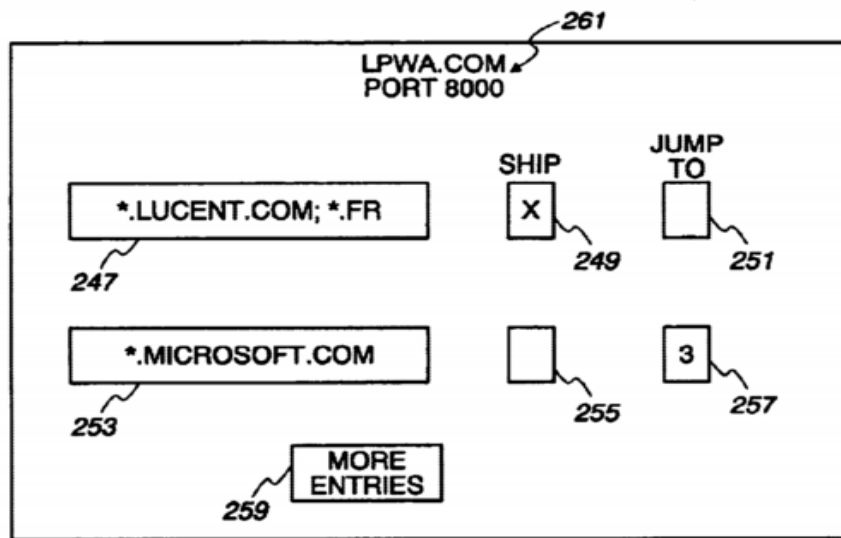
75. Kocherlakota also teaches more advanced routing of URLs to web proxies.

(“The preceding example of FIG. 2 relates to the standard features of proxy chaining. The chaining applet may also provide advanced features to the users who want to take advantage of them.”) *Id.*, 4:55-58.

76. With these advanced features, proxy 17 can route URLs to particular proxy servers, 19 or 21 depending on the particular URL and user configuration information. FIG. 7 shows how the user defines this URL and configuration information for the first proxy, LPWA.com.

(“The window of FIG. 7 relates to the first proxy **lpwa.com** which name and port number are displayed at 261. The window initially presents to the user two rows of text windows and control boxes.”) *Id.*, 5:3-7

Fig. 7



77. With the window of FIG. 7, a user can define advanced features to apply to specific predetermined URLs. A specific URL can cause the first proxy to execute a “skip” or “jump to” command.

(“The text window 247 is used to enter URL's for which the first proxy is to be skipped or a “jump to” command is to be entered.”) *Id.*, 5:9-11.

78. As shown in FIG. 7, LPWA.com, proxy 17, can be configured to have the services of LPWA.com skipped for specific predetermined URLs, such as *.LUNCENT.COM or *.FR.com, if box 249 is checked. The asterisk represents well

known matching criteria for text strings, and are particularly known as “wildcards” which can match one or more characters.

(“As shown in FIG. 7, when a URL matching *.lucent.com or *.fr is transmitted to the first proxy, the services of that proxy are to be skipped as defined by the “X” in box 249.”) *Id.*, 5:17-21

79. Also, as shown in FIG. 7, LPWA.com, proxy 17, can be configured to jump to the third proxy and skip the second proxy entirely for specific predetermined URLs, in this case *.Microsoft.com or *.companyx.com.

(“Similarly, whenever a URL of *.**companyx.com** is sent to the first proxy, the first proxy will **skip the second proxy and jump to the third proxy** because the number 3 is entered into the jump to box 251.”) *Id.*, 5:21-24

80. In this way a user can define which proxy servers are used for specific predetermined URLs.

(“The advanced features allow the user to **define which proxy 25 servers are employed for predetermined URLs.**”) *Id.*, 5:24-26

81. Thus, Kocherlakota teaches a number of ways URLs will be routed by proxy 17. If advanced features are not used, i.e. a URL is not predetermined, as it does not match any URL in the windows of FIG. 7, the URL will be sent via the chained proxies 17, 19, and 21. That is, the URL will be sent by proxy 17 to proxy 19. If the URL matches the skip criteria, this URL will be directly sent to proxy 19.

If the URL matches the criteria of a “jump to” URL, proxy 17 will skip all the proxy servers and jump to the proxy server designated in “jump to” box. Thus, proxy 17 selects different proxies (e.g., proxy 19 or proxy 21) for routing based on comparing the received URL to configured rules.

82. A POSITA would understand that the session setup and maintenance of these sessions, as well as the routing of URLs through different proxy servers to web servers, necessarily requires, or at a minimum renders obvious, proxy 17 storing a group or list IP addresses of proxies 19 and 21 as well as client computer 11. If these addresses were not stored, then proxy 17 would have no way of creating message to send to the intended receivers over the Internet. As a POSITA would understand, the Internet makes use of IP packets, which make use of IP addresses to establish sessions and send messages over the sessions. Thus, A POSITA would understand that devices on the Internet are addressed by IP address. EX1015, 17/51; EX1016, 57/91. Kocherlakota confirms this understanding.

(“An anonymity proxy **establishes a session with a client using the client's IP address** and user information and requests information from the internet without identifying the requesting client.”) *Id.*, 1:24-27

83. Thus, in order for proxy 17 to establish and maintain sessions with proxies 19 and 21, proxy 17 would have and store the IP address corresponding to the domain name of the other proxy servers (e.g., “french.com”, “fast.com”) that the

user enters for proxies 19 and 21, otherwise proxy 17 could not establish a session with these proxies over the Internet. Likewise, to establish and maintain sessions with client computer 11 over the Internet, proxy 17 must store the IP address of client computer 11, which would be received as the “sending address” when client computer 11 contacts proxy 17.

84. Furthermore, a POSITA would understand that the proxy 17 must also subsequently store and make use of these IP addresses so that the sessions between proxy servers remain active. Accordingly, when proxy 17 selects proxy 19 or 21 for routing based on the received URL, proxy 17 selects the IP address of the proxy from the group of proxy IP addresses that are stored in order to route to that proxy. If for some reason the session became inactive, a POSITA would understand that there are well known techniques to convert a URL of the proxy server into an IP address using DNS. *See e.g.* EX1005, 94/114 (“clients using HTTP rely heavily on the Domain Name Server...”).

85. Kocherlakota was issued by the USPTO on August 31, 2004. Kocherlakota thus predates the earliest claimed priority date for the '826 Patent, October 8, 2009. Therefore, I understand Kocherlakota to qualify as prior art.

B. RFC 1122

86. RFC 1122 is titled “Requirements for Internet Hosts – Communication Layers” and provides requirements for link layer, Internet layer, and transport

protocols over the Internet. This RFC is a one of a pair of RFCs which define the requirements of Internet host software, the other being RFC 1123, which is listed as a reference in RFC 2616. *See* EX1005, 97/114. RFC 1122 was published twenty years before the '826 Patent's priority date by the Network Working Group of the Internet Engineering Task Force (IETF) in October 1989. Applicants submitted RFC 1122 to the Patent Office during prosecution of the '826 Patent. EX1007, 181 and 541 (Information Disclosure Statement listing RFC 1122 as prior art). RFCs (and similar standards documents) posted on the Internet are published in the ordinary course by established standards organizations, and are intended to be viewed by the interested Internet engineering audience at large as of their dates of publication as stated on the cover of each. Therefore, I understand RFC 1122 to qualify as prior art.

87. I obtained a copy of RFC 1122 from the IETF website source that I understand to be the repository of IETF's RFCs, and which a POSITA would commonly utilize to obtain an electronic copy of the IETF's RFCs. The copy of RFC 1122 that I obtained from <https://datatracker.ietf.org/doc/html/rfc1122> is attached as EX1006.

88. RFC 1122 describes various features of the TCP protocol and in particular describes a "keep alive" mechanism to confirm that TCP connections are still active. As described in RFC 1122, this keep alive mechanism periodically

probes the other end of a connection when the connection is otherwise idle, even when there is no data to be sent.” EX1006, 102/116. “To confirm that an idle connection is still active,” some TCP “implementations send a probe segment designed to elicit a response from the peer TCP.” *Id.* “[T]he probe causes the receiver to return an acknowledgment segment, confirming that the connection is still live. If the peer has dropped the connection due to a network partition or a crash, it will respond with a RST instead of an acknowledgment segment.” *Id.*

C. TCP Keepalive HOWTO

89. How to implement the keep alive probes mechanism in the TCP protocol was also well-known to a POSITA as it was discussed in “TCP Keepalive HOWTO” published on 2007-05-04. EX1021, Section 1.

90. This document teaches several reasons for using TCP keep alive messages, for example, checking for dead peers and preventing disconnections due to network inactivity. EX1021, Section 2.2. The way keep alive works in TCP is also described.

(“The keepalive concept is very simple: when you set up a TCP connection, you associate a set of timers. Some of these timers deal with the keepalive procedure. When the keepalive timer reaches zero, you send your peer a keepalive probe packet with no data in it and the ACK flag turned on. You can do this because of the TCP/IP specifications, as a sort of duplicate ACK,

and the remote endpoint will have no arguments, as TCP is a stream-oriented protocol. On the other hand, you will receive a reply from the remote host (which doesn't need to support keepalive at all, just TCP/IP), with no data and the ACK set. If you receive a reply to your keepalive probe, you can assert that the connection is still up and running without worrying about the user-level implementation.”) EX1021, Section 2.1.

D. U.S. Pat. No. 6,389,462 (“Cohen”)

91. Cohen discloses a system and method for intercepting client web requests and redirecting them to proxy caches. EX1004, 1:7-9. The proxy cache has a stored copy of the object on the origin server.

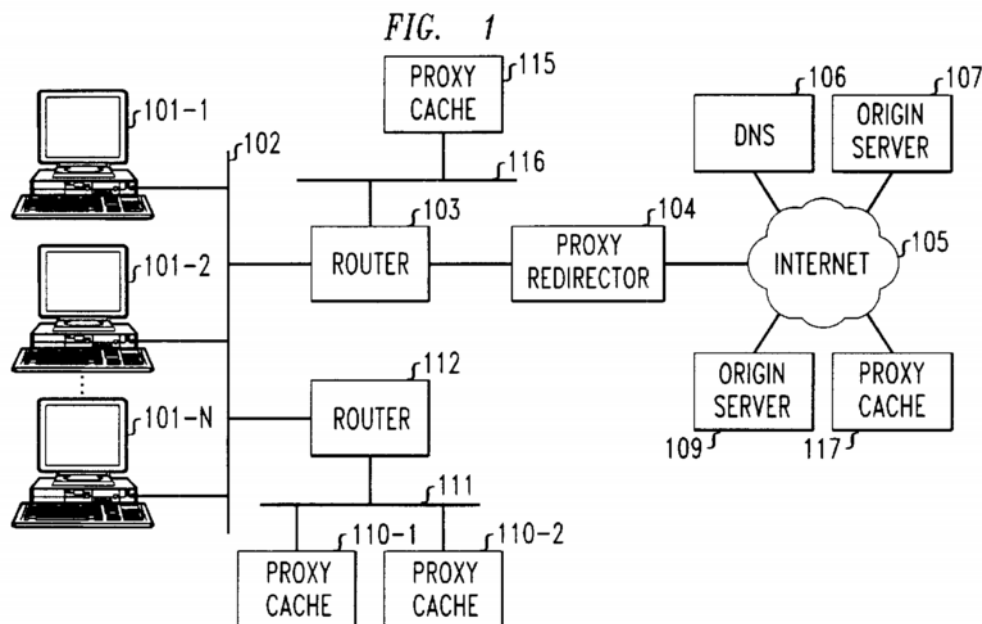
(“Proxy caching is currently used to decrease both the latency of object retrieval and traffic on the Internet backbone. **As is well known, if a proxy cache has stored a copy of an object from an origin server that has been requested by a client, the requested object is supplied to the client from the proxy cache rather than from the origin server.**”) EX1004, 1:13-18.

92. The process described in Cohen begins when a client makes a request with a URL through a web browser, which initiates a DNS lookup which returns the associated IP address.

(“When any of the clients connected to LAN 102, such as **client 101-1**, makes a request through a browser for an object **by specifying a logical URL**, a

domain name server (DNS) 106 connected locally or on Internet 105, as shown, is accessed to perform a database look-up based on that logical name. An associated **IP address is then returned to the browser.**") *Id.*, 6:47-54; ("That IP address then serves as the IP address to which packets directed to the origin server from the client are directed."). *Id.*, 6:55-65.

93. Then according to the invention, "once the IP address of the origin server corresponding to the logical URL name is determined through the DNS look-up, **proxy redirector 104**, rather than establishing a TCP connection to the origin server at the determined IP address, **transparently establishes a TCP connection between the client and a proxy**. If the requested object is stored in the cache, a copy of that object is transparently returned to the requesting client." *Id.*, 7:12-19; FIG. 1 below.



94. Specifically, the “proxy redirector 104” intercepts a web request from a “client 101-1” seeking content from a target web server and “selects one of the available proxy caches [e.g. 110-1, 110-2, 115, and 117] to which to forward client requests based on a metric such as least-loaded or round-robin, [or] based on IP header information such as the origin server IP address.” EX1004, 7:36-48; see also *Id.*, 2:15-20

95. Cohen teaches that after the TCP connection is established, the client sends “one or more packets to the origin server [via the proxy redirector 104], which packets include a GET request. The GET request includes a complete URL, which identifies to that server the specific object within the origin server site that the client wants a copy of.” *Id.*, 9:19-22

96. The proxy redirector 104 then forms the absolute URL by prefixing the complete URL “with the IP address of the origin server obtained from the destination IP address of the packet(s) containing the GET request.” *Id.*, 9:28-10:3. “Responsive packets from the proxy cache are sent to proxy redirector 104 where they are redirected to client.” *Id.*, 8:55-58. “If the proxy cache to which the request is directed does not contain the requested object, a separate TCP connection is established between the proxy cache and the origin server to obtain a copy of the requested object.” *Id.*, 7:27-35.

XI. COMPARISON OF THE PRIOR ART TO THE CLAIMS

97. I understand claims are to be given their ordinary and customary meaning to a POSITA in view of the claims, specification, and prosecution history. I also understand that Applicants can define their own claim terms in the specification, which will be used when interpreting the claims.

A. Ground 1 – Kocherlakota Renders Obvious Claims 1-5, 10, 12, 15-19, and 21-25

1. Limitation 1

a. Limitation 1[p]: A method for use with a web server that stores a first content identified by a first content identifier, and for use with a first client device that is addressed over the Internet using first Internet Protocol (IP) address, for use with a list of IP addresses stored in the first server, the method by the first server comprising

98. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota teaches a web server that stores a first content, a web page, identified by a first content identifier, a URL. Kocherlakota teaches web server 15 stores content such as data files.

(“Also, shown in FIG. 1 is a **web server 15** which is representative of any of 60 the **well known data file providers** and, in fact, represents the universe of addressable web servers.”) EX1003, 2:58 – 61.

99. The first identifier is for example, a URL for which, by applying the logic shown in FIGs. 2 and 7, proxy 17 selects proxy 19. As described above in

Section X.A (“Kocherlakota”), Kocherlakota teaches a number of ways URLs will be routed by proxy 17. If advanced features are not used, i.e. a URL is not predetermined, as it does not match any URL in the windows of FIG. 7, the URL will be sent via the chained proxies 17, 19, and 21. That is this URL will be sent by proxy 17 to proxy 19. If the URL matches the skip criteria, this URL will be directly sent to proxy 19. Thus, a URL is sent by the client to the web server identifying the content identified requested by the client. EX1003, 4:29 – 31 (“After the proxy chain has been established, the **URL of web server 15**, or any other web server, **can be sent from the client** to the internet via the chained proxies 17, 19 and 21.”)

100. Client computer 11 meets the construction of “client device” because it is a device that communicates over the Internet and it is operating in the role of a client by establishing a connection with proxy 17 to make requests. EX1003, 2:54-58 (“client computer 11 connected to the internet 13”), 4:29-31 (“the URL of web server 15, or any other web server, can be sent from the client to the internet via the chained proxies 17, 19 and 21”). *See also Id.*, 1:43-45, 2:4-7, 2:30-32.

101. Proxy 17 meets the construction of “first server” because it is a device operating in the role of a server by accepting a connection from client computer 11 to service requests (URL’s) by sending back responses (content from the web server) and because it is not the same physical device as client computer 11 (the claimed “first client device”). *Id.*, 5:3-24.

102. Kocherlakota teaches that proxy servers are a special type of server. EX1003 Col 1:10-15 (“The internet includes many different types of servers which provide special services to clients in session there with. Included in these servers are proxies which provide services to a user but which do so by establishing sessions with web servers and responding to internet data from those web servers.”).

103. FIG. 1 of Kocherlakota shows a client computer 11 (the claimed “first client device”) and proxy 17 (the claimed “first server”) communicate over the Internet. EX1003, FIG. 1, 3:3-8. A POSITA would understand that client computer 11 is addressed over the Internet using an IP address (the claimed “first IP address”) because a POSITA would understand that devices on the Internet are addressed by IP address. (See EX1015, 17/51; EX1016, 57/91). Further, as discussed above in Section X.A (“Kocherlakota”), in order to establish sessions with proxies 19 and 21, proxy 17 must store a list of IP addresses associated with proxies 19 and 21 (the claimed “list of IP addresses”). See EX1003, 2:30-32.

b. Limitation 1[a]: storing, operating, or using, a server operating system

104. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota describes proxy 17 (the claimed “first server”) as an example of a “proxy server” that “provide services to a user ... by establishing sessions with web servers and responding to internet data from those web servers.” EX1003, 1:10-27, 4:1-31, 5:7-26. As proxy 17 is acting as a server by responding

to requests for services from clients, its operating system must be a server operating system.

105. Moreover, Kocherlakota teaches that proxies such as proxy 17 are server machines. (“The **internet includes many different types of servers** which provide special services to clients in session there with. **Included in these servers are proxies** which provide services to a user but which do so by establishing sessions with web servers and responding to internet data from those web servers.”) EX1003, Col. 1:10-15. A POSITA would know that a proxy server, such as proxy 17 (the claimed “first server”), makes use of an operating system as they are required to manage, store and execute programs that allow hardware to establish connections with web servers. This is particularly true for servers, such as proxy 17, that communicate over the internet with web servers through HTTP and TCP/IP protocols. *See* EX1005, 71/114 (describing the use of “HTTP servers” as using “UNIX, Microsoft Windows, and other operating systems”); *see also* EX1016, 14/191 (stating that “TCP is assumed to be a module in an operating system”).

106. Examples of server operating systems around October of 2009 are Unix, Linux, and Microsoft Windows NT. *See* EX1022, Col. 8:53-55. (“A general-purpose server operating system may be employed, e.g., UNIX, LINUX™, Microsoft WINDOWS NT®, or the like.”) It would have been obvious for a POSITA to store, operate, or use one of these operating systems in proxy 17.

c. Limitation 1[b]: receiving, from the first client device, the first content identifier

107. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota teaches proxy 17 receives the first content identifier, the URL, from the client 11.

(“After the proxy chain has been established, **the URL** of web server 15, or any other web server, **can be sent from the client to the internet via the chained proxies 17, 19 and 21**”) EX1003, 4:29 – 31.

(“FIG. 1 includes dotted line segments 12, 14 and 16 and solid line "loop" segments connecting each web proxy 17, 19 and 21 to the internet. These segments are functional in nature and illustrate the chained **connection of the client 11 to the web server 15 via the chained proxies 17, 19 and 21.**”) *Id.*, 3:3-8

108. By way of example, Kocherlakota teaches that the matching URL *.lucent.com is transmitted to proxy 17.

(“As shown in FIG. 7, when a **URL matching *.lucent.com** or *.fr is **transmitted to the first proxy**, the services of that proxy are to be skipped as defined by the “X” in box 249.”) *Id.*, 5:18-21

109. FIG. 1 and FIG. 6 annotated below also show this disclosure.

Fig. 1

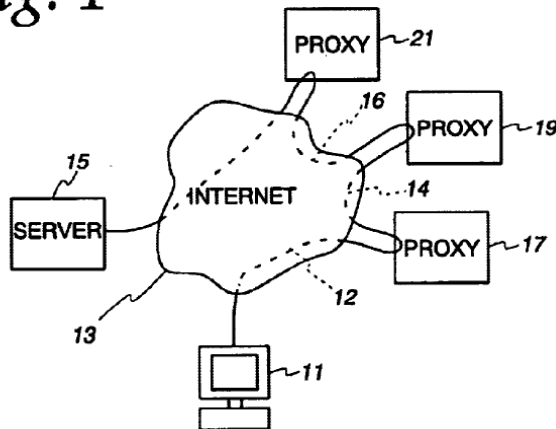
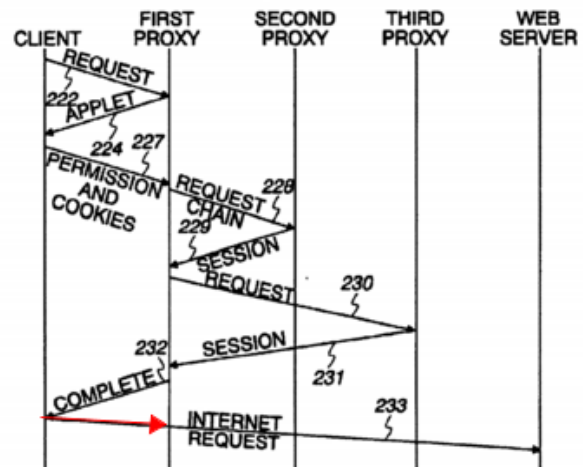


Fig. 6



d. Limitation 1[c] selecting, an IP address from the list according to a criterion

110. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota teaches that Proxy 17 (the claimed “first server”) selects the IP address from the group of IP addresses for proxies 19 or 21 to route the URL through the proxy chain.

(“As shown in FIG. 7, when a URL matching *.lucent.com or *.fr is transmitted to the first proxy, the services of that proxy are to be skipped as defined by the “X” in box 249. Similarly, whenever a URL of *.companyx.com is sent to the first proxy, the **first proxy will skip the second proxy and jump to the third proxy** because the number 3 is entered into the

jump to box 251. The advanced features allow the user to define which proxy servers are employed for predetermined URLs.”) EX1003, 5:18-26.

111. As discussed above in Section X.A (“Kocherlakota”), upon receiving the URL, proxy 17 (the claimed “first server”) selects the IP address of either proxy 19 or 21 from the group of those IP addresses to route the URL through the proxy chain.

112. When proxy 17 receives from the client computer 11 a URL not matching the criteria in either in text window 247 or 253, including www.uspto.gov, proxy 17 will not “skip” or “jump to” any server in the proxy chain because the URL does not match the criteria and will therefore select the IP address of the next proxy in the chain (i.e. proxy 19) so that the request passes “via the chained proxies 17, 19 and 21.” EX1003, 4:29-31, FIGs. 6-7. Proxy 17 will thus select the IP address of proxy 19 as the address used to forward the URL.

113. Alternatively, when proxy 17 receives from the client computer 11 a URL matching the criteria in text window 247 but not text window 253, including www.lucent.com, proxy 17 will “skip” the function of proxy 17, but will not “jump” over any server in the proxy chain because the URL does not match the “jump” criteria and will therefore select the IP address of proxy 19 so that the request passes “via the chained proxies 17, 19 and 21.” EX1003, 4:29-31, 5:10-23, FIGs. 6-7.

Proxy 17 will thus will again select the IP address of proxy 19 as the address used to forward the URL list according to a criterion.

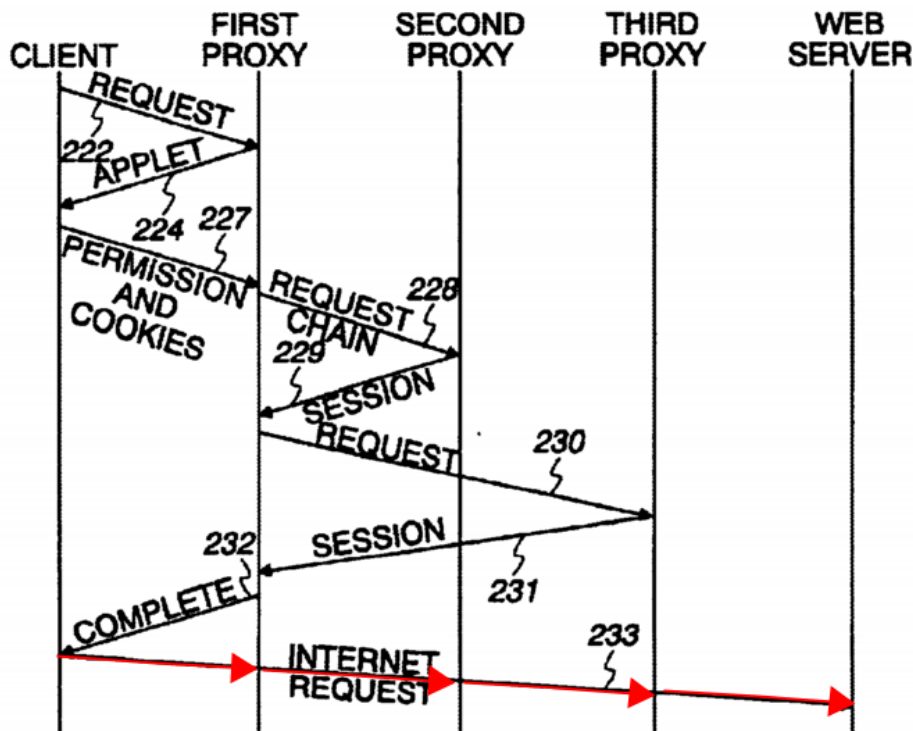
114. Limitation 1[d]: sending, in response to the receiving of the first content identifier, using the selected IP address, the first content identifier.

115. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota teaches that the client access web servers via the chain of proxies.

(“Thereafter, the **client may access a web server** via the chain of proxies as represented by message 233.”)

116. As discussed for Limitation 1[c], proxy 17 (the claimed “first server”) selects the IP address of proxy 19 and, in response to the selecting, forwards the URL to proxy 19. EX1003, 4:29-31, 5:10-23, FIGs. 6-7. Proxy 17 forwards the URL to proxy 19 in order for the URL to be further forwarded by proxy 19 to proxy 21 and proxy 21 to the web server hosting the first URL (the claimed “first web server”) as shown in annotated FIG. 6 below.

Fig. 6



117. When proxy 17 sends a URL to an intermediary such as proxy 19, it uses the IP address of proxy 19 as a destination address. As this URL is eventually sent to the web server, the first server, proxy 17, is therefore using the selected IP address to send the first URL to proxy 19.

e. Limitation 1[e]: receiving, in response to the sending of the first content identifier, the first content from the web server

118. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota teaches that proxies establish sessions to web servers and respond to internet data from those web servers.

(“Included in these servers are **proxies** which provide services to a user but which do so by establishing sessions with web servers and **responding to internet data from those web servers.**”) EX1003, 1:12-15.

119. As discussed for Limitations 1[c] and 1[d], proxy 17 forwards the first URL along the proxy chain to the web server corresponding to the first URL. A POSITA would understand that the function of a proxy such as proxy 17 in the proxy chain is both to send URLs to web servers, as well as receive responses from web servers and return them to the requesting client, otherwise the client would not be making use of accessing these web servers. Kocherlakota describes users “access[ing] the internet ‘through’ a proxy which provides value added to the **information accessed**” and proxy servers “responding to internet data from those web servers.” EX1003, 1:12-17. That is, client computer 11 obtains “access to web server 15 [] via the three web proxies 17, 19 and 21.” *Id.*, 3:9-11. By way of example, proxy 19 “provide[s] translations of identified files,” which requires that proxy 19 receives the files from the web server that are identified by the requested URL and performs language translation on the received files. *Id.*, 1:28-31, 4:29-35 (“the language proxy 19 provides the French language features”);

120. Further, Kocherlakota describes web server 15 as a “data file provider[]” and “[a]fter the establishment of proxy sessions the client can surf the web through the established chain of proxy servers.” EX1003, 2:58-61, 2:30-32.

POSITA would understand that the “web server” of Kocherlakota responds to web requests by providing a web page to the requesting device, as that is the purpose of web servers.

121. In the disclosed proxy chain the web server corresponding to the first URL returns the web page corresponding to the first URL to proxy 21, proxy 21 forwards the web page to proxy 19, and proxy 19 forwards the web page to proxy 17. *Id.*, 2:58-61, 2:30-32, 4:29-35. Proxy 17 (the claimed “first server”) thus receives the web page corresponding to the first URL (the claimed “first content”) in response to the sending of the first URL.

f. Limitation 1[f]: sending the received first content, or a part thereof, to the first client device

122. In my opinion Kocherlakota discloses or renders obvious this limitation. Client computer 11 obtains “access to web server 15 [] via the three web proxies 17, 19 and 21.” EX1003, 3:9-11. As discussed for Limitation 1[e], the function of a proxy such as proxy 17 in the proxy chain is both to send URLs to web servers, as well as receive responses from web servers and return them to the requesting client, otherwise the client would never see the web pages accessed. As explained for Limitation 1[e], proxy 17 (the claimed “first server”) receives the web page corresponding to the first URL (the claimed “first content”). In accordance with its function as a proxy and providing client computer 11 “access to web server

15,” proxy 17 sends the received first content to client computer 11 (the claimed “first client device”).

2. Limitation 2: The method according to claim 1, further for use with a second client device that is addressable in the Internet using the selected IP address.

123. In my opinion Kocherlakota discloses or renders obvious this limitation. Proxy 19 and proxy 21 are client devices as each is a device that communicates over the Internet and operate in the role of a client by establishing a connection down the proxy chain to make requests, as defined in RFC 2616. Section VII.B (“RFC1122”), above; *See also* EX1005, 8/114 (proxy devices “act[s] as both a server and a client”). As described above in Limitation 1[p] and Section X.A (“Kocherlakota”) above, proxies 19 and 21 are connected to proxy 17 via sessions over the Internet. Kocherlakota teaches that the second proxies, proxy 19 and 21 connected over the Internet.

(“FIG. 1 includes dotted line segments 12, 14 and 16 and solid line “loop’ segments **connecting each web proxy 17, 19 and 21 to the internet.**”)

EX1003, 3:3-5.

124. A POSITA would understand that in order to be connected or communicate over the Internet, each of the proxies 19 and 21 must be addressable in the Internet using the selected IP address so the IP packet can be sent to these devices. See EX1015, 17/51; EX1016, 57/91. As described above in Limitation 1[c]

and Section X.A (“Kocherlakota”) above, in order for proxy 17 to establish and maintain sessions with proxies 19 and 21, proxy 17, proxies 19 and 21 would have to be addressable in the Internet using the selected IP addresses for each of proxies 19 and 21.

3. Limitation 3: The method according to claim 2, wherein the sending of the first content identifier comprises sending to the second client device.

125. In my opinion Kocherlakota discloses or renders obvious this limitation. As explained for Limitations 1[c] and 1[d], the process of proxy 17 sending the web request (containing the first URL, e.g., www.uspto.gov or www.lucent.com) to web server 15 includes proxy 17 (the claimed “first server”) sending the first URL to proxy 19 (the claimed “client device addressed by the selected IP address”) over the Internet.

(“FIG. 1 includes dotted line segments 12, 14 and 16 and solid line “loop’ segments **connecting each web proxy 17, 19 and 21 to the internet.**”)

EX1003, 3:3-5.

4. Limitation 4: The method according to claim 3, further comprising: receiving, by the second client device from the first server, the first content identifier; sending, by the second client device from the web server, the first content; and sending, by the second client device to the first server, the received first content or a part thereof.

126. In my opinion Kocherlakota discloses or renders obvious this limitation. As explained for Limitations 1[e], proxy 21 receives from the web server

the web page corresponding to the first URL from web server 15 and then proxy 19 receives the web page from proxy 21, as these proxy servers are chained together.

(“When the **third proxy session is established, a session continues (step 125) between the client and a web server, e.g. 15, through proxy servers 17, 19 and 21.** The third proxy server establishes the session in a manner substantially similar to FIG. 5.”) EX1003, 4:23-28; *see also id*, 2:58-61, 2:30-32, 3:3-5, 4:29-35.

127. As explained for Limitation 1[d], the process of proxy 17 sending the first URL, e.g., www.uspto.gov or www.lucent.com to web server 15 includes proxy 19 (the claimed “second client device”) sending the first URL to proxy 21 for forwarding to the web server. Proxy 19 sending to an intermediary, proxy 21, for eventual sending to the web server meets the claim language “sending ... to the first web server” because this language includes sending to intermediaries for forwarding to the web server. Thus, proxy 19 receives the first URL from proxy 17.

128. As I explained for Limitation 1[e], proxy 19 (the claimed “second client device”) forwards the web page (the claimed “first content”) to proxy 17 (the claimed “first server”). EX1003, 2:58-61, 2:30-32, 4:29-35. When the proxy 19 receives the URL it is receiving, by the second client device from the first server, the first content identifier. When the web page on web server is sent to proxy 19 and on to proxy 17, it is sending, by the second client device from the web server,

the first content; and sending, by the second client device to the first server, the received first content or a part thereof.

5. Limitation 5: The method according to claim 2, further comprising by the first server: communicating with the second client device; and storing an IP address of the second client device in the list in response to the communicating

129. In my opinion Kocherlakota discloses or renders obvious this limitation. Proxy 17 (the claimed “first server”) establishes and maintains sessions, and thereby communicates, with each of proxies 19 and 21 (the claimed “second client device”). EX1003, 4:43-54.

(“When the third proxy session is established, **a session continues (step 125) between the client and a web server, e.g. 15, through proxy servers 17, 19 and 21.** The third proxy server establishes the session in a manner substantially similar to FIG. 5.”) EX1003, 4:23-28.

(“FIG. 1 includes dotted line segments 12, 14 and 16 and solid line ‘loop’ **segments connecting each web proxy 17, 19 and 21 to the internet.**”) *Id.*, 3:3-5.

130. A POSITA would understand that the session setup and maintenance, as well as subsequent routing of URLs through the proxy chains to proxies and on to web servers, necessarily requires, or at a minimum renders obvious, proxy 17 storing a list of IP addresses of proxies 19 and 21, otherwise they could not communicate. For example, proxy 17 stores the IP addresses of proxies 19 and 21

to track of an ongoing session, that storing is in response to the communicating with proxies 19 and 21 during session setup.

6. Limitation 10: The method according to claim 1, wherein the criterion is stored in the first server

131. In my opinion Kocherlakota discloses or renders obvious this limitation. As discussed for Limitation 1[c], proxy 17 performs the selecting step according to the criterion shown in Kocherlakota's FIG. 7 and FIG. 2 (e.g., applying "skip" and "jump"). EX1003, 5:10-26. "The applet information and relevant session information," which include the information in FIGs. 2 and 7, is "received by the first proxy server" and, therefore, must be stored in proxy 17 to be used in routing the URLs. EX1003, 3:61-67.

7. Limitation 12: The method according to claim 10, wherein the criterion is based on, or comprises, a geographical location of a client device addressed by the selected IP address, or a response time when communicating with the client device addressed by the selected IP address

132. In my opinion Kocherlakota discloses or renders obvious this limitation. First Kocherlakota discloses using a geographical location in selecting a proxy. Kocherlakota teaches that a "French language web proxy 19" provides "language service[s]," which, as discussed for Limitation 1[c], may be selected by IP address. EX1003, 2:65-67. A POSITA would understand the French language services are, or would be obvious to be, associated with a geographic location—France. Further, Kocherlakota discloses matching criteria "*.fr," which a POSITA

would understand is the top-level domain for France—a geographic location. EX1003, FIG. 7.

133. Kocherlakota also discloses using response time to select a proxy. Kocherlakota describes a wait limiting proxy, stating that (“the client would like to reduce the amount of time spent retrieving data so the services of a wait limiting proxy 21 are desired.”) EX1003, 1:1-3. A POSITA would understand that response time and wait times are measurements of the same thing, how long it takes to get a response. Therefore, choosing a proxy based on wait limiting is the same as choosing a proxy based on response time.

134. Further, it been obvious for a POSITA to base the selection of a proxy on response time in order to reduce the amount of time, wait time, spent retrieving data.

135. In addition, a POSITA would understand that the selection of proxies can be done by a variety of criteria, such as load balancing and geographic location. As described above in Section VII (Overview of the Technical Field), the HTTP-The-Definitive-Guide teaches that the selection of the proxies can be based on load balancing, subscriptions or geographic proximity routing, where a child proxy may select a parent responsible for the origin server’s geographic region.

(“Here are a few other examples of dynamic parent selection:

Load balancing

A child proxy might pick a parent proxy based on the current level of workload on the parents, to spread the load around.

Geographic proximity routing

A child proxy might select a parent responsible for the origin server's geographic region.") (See EX1023, page 140.)

136. A POSITA seeking to lower response time would select a proxy based on load, e.g. load balancing or may choose to select a proxy responsible for the origin server's geographic region, e.g. selecting based on geographical location.

8. *Limitation 15: The method according to claim 1, wherein the first server is a Transmission Control Protocol/Internet Protocol (TCP/IP) server that communicates over the Internet with the first client device based on, according to, or using, TCP/IP protocol or connection.*

137. In my opinion Kocherlakota discloses or renders obvious this limitation. As explained for Limitation 6, proxy 17 communicates, with each of proxies 19 and 21 (the claimed "each of the client devices") over the Internet. EX1003, 3:2-5; A POSITA would have knowledge of TCP and understand that one of the most common ways to communicate over the Internet is to use TCP.

("As is known by those having ordinary skill in the art, TCPIP is a relatively low-level protocol, as opposed to HTTP, which is a high level protocol.") EX1001, 17:22-24.

138. Further, as described in U.S. Patent 5,826,014, a cited reference to Kocherlakota, Internet protocols are based on TCP and IP.

(All of the **Internet-compatible protocols**, however, **find some basis in the two original Internet protocols: TCP (Transmission Control Protocol) and IP (Internet Protocol).**) EX1018, 1:23-27.

139. Therefore, in my opinion, it would have been obvious for a POSITA for to use the TCP protocol to communicate between the first server, proxy 17 to client devices, proxies 19 and 21.

140. A POSITA would also understand that the disclosed sending of URLs in Kocherlakoda indicates the communication is HTTP-based. EX1003, 3:21-4:35. A POSITA would understand that, as RFC 2616, which the '826 Patent recognizes “define[s]” “the HTTP protocol,” explains, “HTTP communication usually takes place over **TCP/IP connections.**” See EX1001, 16:21-22, 17:20-26; EX1005, 11/114. A POSITA would also understand that communicating over TCP requires establishing, by the communicating devices, a TCP session. See EX1016, 16-18/91. Thus, a POSITA would understand that Kocherlakota’s disclosure of proxy 17 communicating with each of proxies 19 and 21 discloses or renders obvious proxy 17 communicating using TCP/IP, including establishing a TCP connection with each of proxies 19 and 21.

9. Limitation 16: The method according to claim 1, wherein the first server communicates over the Internet based on, or according to, one out of UDP, DNS, TCP, FTP, POP*, SMTP, or SQL standards.

141. See Limitation 15 for communication over the Internet based the use of TCP.

10.Limitation 17: The method according to claim 1, wherein the first content comprises web-page, audio, or video content, and wherein the first content identifier comprises a Uniform Resource Locator (URL).

142. The first content includes a web page and the first content identifier includes a URL. See Limitation 1[p], 1[b].

11. Limitation 18: The method according to claim 1, for use with a plurality of client devices that includes a second client device, each identified over the Internet using a respective IP address from the list, wherein the method further comprising: selecting, by the first server, multiple client devices from the plurality of client devices by selecting the IP addresses of the multiple client devices from the list; and sending, by the first server, the selected IP addresses in response to the selecting.

143. In my opinion Kocherlakota discloses or renders obvious this limitation. As explained in Limitation 2, proxy 17 is utilized with multiple other proxies, such as proxy 19 and proxy 21, each of which meet the construction of “second client device” and are addressable over the Internet using the IP address that are selected from the list of IP addresses stored in proxy 17.

144. As explained for Limitation 1[c], the process of proxy 17 selecting proxies 19 and 21 includes selecting the IP address of proxy 19 and 21 from the list of those IP addresses to route the URL through the proxy chain. For example, when

proxy 17 receives a URL not matching the criteria in either in text window 247 or 253, proxy 17 will select the IP address of the next proxy in the chain (i.e., proxy 19). *See* Section X.A (“Kocherlakota”). To further illustrate, when proxy 17 receives a URL matching the criteria of text window 253 (e.g., www.microsoft.com), proxy 17 “will skip the second proxy and jump to the third proxy because the number 3 is entered into the jump to box 257” and will therefore select the IP address of proxy 21. *Id.* Thus, proxy 17 selects the IP addresses of multiple client devices from the list of IP addresses. When proxy 17 sends the URL to the selected proxy it is, sending, by the first server, the selected IP addresses in response to the selecting.

12. *Limitation 19: The method according to claim 1, further comprising sending of the selected IP addresses to the first client device*

145. In my opinion Kocherlakota renders obvious this limitation. Kocherlakota teaches that the first proxy signifies in message 232 to the client that the session chain is complete. (“The third proxy responds by returning a session message 231 to the first proxy and the first proxy signifies 232 to the client that the chain is complete.”) EX1003 4:49-53. It would have been obvious for the first proxy to send the IP address of the proxies in the chain or their near equivalent, the URL of the selected proxies so that the first client could verify which chain was set up and that the chain was set up correctly.

13. Limitation 21: The method according to claim 1, wherein the first client device is identified by a first Internet Protocol (IP) address, Media Access Control (MAC) address, or a hostname.

146. In my opinion Kocherlakota discloses or renders obvious this limitation. As shown in FIG. 1, client computer 11 (the claimed “first client device”) and proxy 17 (the claimed “first server”) communicate over the Internet. EX1003, FIG. 1, 3:3-8. A POSITA would understand that client computer 11 is addressed over the Internet using an IP address (the claimed “first IP address”) because a POSITA would understand that devices on the Internet are addressed by IP address. See EX1015, 17/51; EX1016, 57/91. Thus, client computer 11 (the claimed “first client device”) is identified by an IP address (the claimed “first IP address”).

14. Limitation 22: The method according to claim 1, wherein the web server is a Hypertext Transfer Protocol (HTTP) server responding to HTTP requests.

147. In my opinion Kocherlakota discloses or renders obvious this limitation. Kocherlakota describes “web server 15” (the claimed “web server”) as “representative of any of the well-known data file providers and, in fact, represents the universe of addressable web servers.” EX1003, 2:58-61. HTTP servers responding to HTTP requests were within “the universe of addressable web servers” See EX1019, 2:43-53; EX1005, 94/114); EX1001, 4:63-67.

15.Limitation 23: The method according to claim 1, wherein the steps are sequentially executed.

148. In my opinion Kocherlakota discloses or renders obvious this limitation. As discussed for Limitations 1[p]-1[f], Kocherlakota discloses proxy 17 performing the steps of Claim 1 in a sequential manner. The claimed steps follow the known function of a proxy—sending URLs to web servers, receiving responses from web servers, and returning them to the requesting client in a sequential order. See EX1005 RFC2616 Page 10.

16.Limitation 24: The method according to claim 1, for use with a software application that includes computer instructions that, when executed by a computer processor, cause the processor to perform the steps of the claim 1.

149. In my opinion Kocherlakota discloses or renders obvious this limitation. As discussed for Limitations 1[p]-1[f], Kocherlakota discloses proxy 17 performing the steps of Limitation 1. A POSITA would understand that, as is the nature of computing devices, proxy 17 would include a processor and perform those steps by executing a software application that includes computer instructions causing the processor to perform the steps.

17. Limitation 25: A non-transitory computer readable medium containing computer instructions that, when executed by a computer processor, cause the processor to perform the method according to claim 1.

150. In my opinion Kocherlakota discloses or renders obvious this limitation. As discussed for Limitations 1[p]-1[f], Kocherlakota discloses proxy 17

performing the steps of Limitation 1. A POSITA would understand that, as is the nature of computing devices, proxy 17 would include a processor and a non-transitory computer readable medium containing computer instructions that are to be executed by the processor to cause the processor to perform the steps.

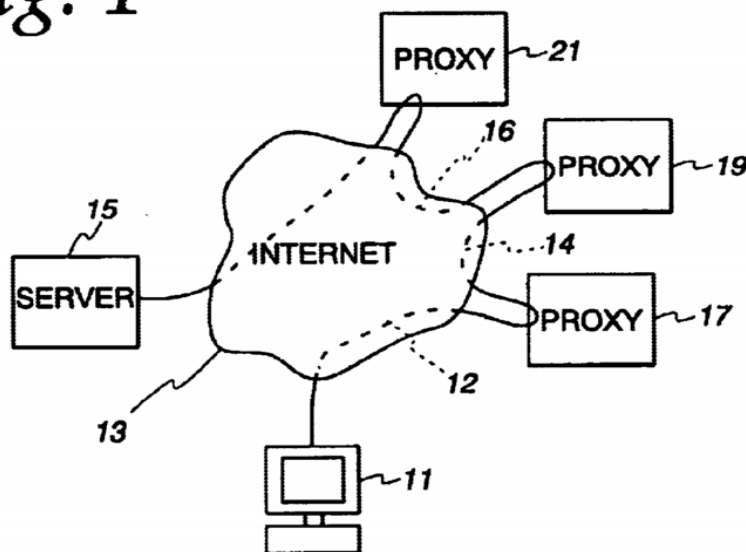
B. Ground 2 – Kocherlakota in View of RFC 1122 Renders Obvious Claims 6-9.

151. In my opinion Kocherlakota in view of RFC 1122 renders obvious claims 6-9.

1. Motivation to combine

152. A POSITA would have been motivated to use the TCP keep-alive functionality of RFC 1122, as discussed in Section X.B (“RFC1122”) above, when maintaining the TCP sessions as disclosed by Kocherlakota. Both references cover connecting Internet hosts over the Internet. RFC 1122 describes an Internet host as

Fig. 1



“A host computer, or simply "host," is the ultimate consumer of communication services.” EX1006, Section 1.1.1, page 6. Kocherlakota describes methods and apparatus for establishing internet communication from a user to a server through a plurality of proxy servers. [Abstract] FIG. 1 demonstrates that the web server 15, client 11, and proxies 17, 19 and 21 are the consumer, or terminator of Internet communications services described in RFC 1122.

153. A POSITA would have recognized the benefits of Kocherlakota’s proxy servers exchanging TCP keep-alives with each other. One benefit, TCP keep-alives help detect when “a client crashes or aborts a connection during a network failure” and thus prevent server applications from “hang[ing] indefinitely and consum[ing] resources unnecessarily.” EX1006, 102/116. For example, following the establishment of sessions between proxy 17 and the client computer 11, proxy 19 and/or proxy 21, one of client computer 11, proxy 19, or proxy 21 could crash. This could lead to the situation where proxy server 17 sends Internet traffic to a proxy that no longer operational. This could lead to lost messages and unnecessarily expend resources keeping the “dead” TCP connection open. Another benefit a POSITA would have known of is that TCP keep-alives could also prevent periods of inactivity between client devices from causing the TCP connection between them to terminate. Which again could result in lost messages.

154. RFC 1122 does specify some potential disadvantages of TCP keep-alives, but a POSITA would have recognized that these disadvantages are outweighed by the benefits listed above in a system such as the interconnected proxy servers of Kocherlakota.

2. Modification of Kocherlakota in view of RFC 1122

155. A POSITA would have modified Kocherlakota so that the sessions between proxy 17 and client computer 11, proxy 19 and/or proxy 21, would periodically communicate by exchanging TCP keep-alive messages with each other over established TCP connections. A "keep-alive" mechanism was a well known technique which periodically probes the other end of a connection when the connection is otherwise idle, even when there is no data to be sent. EX1006, 102/116. In the proxy chain comprising a client computer 11 -> proxy 17 -> proxy 19 or the proxy chain comprising a computer 11 -> proxy 17 -> proxy 21, the proxy 17 would periodically send client computer 11 and proxies 19 and/or 21 keep-alive messages over the TCP connections between them. Further, in accordance with RFC 1122, the proxy 17 would wait for an "acknowledgment" from proxies 19 and/or 21, confirming that the connection is still live.

156. Such as modification would be simple for a POSITA, as all that is required is to send a probe segment designed to elicit a response from the peer TCP periodically. EX1006, 102/116.

C. Element-by-Element Analysis

3. Limitation 6: The method according to claim 1, for use with a second client device, the method further comprising: sending, by the first server, a ‘keep alive’ message to the second client device; and waiting, by the first server, for a response to the ‘keep alive’ message from the second client device

157. In my opinion, Kocherlakota-RFC 1122 renders obvious this limitation. As described above in Limitation 15, Kocherlakota discloses or makes obvious that each of the client devices are connected over the Internet using TCP/IP. As described above, the combination of RFC1122 would make it obvious for a POSITA to use TCP keep-alives to maintain the sessions between proxy 17 and proxies 19, and 21.

158. An example of how a POSITA would use TCP keep-alives is found in “TCP Keepalive HOWTO” described in Section VII.E (TCP Keepalive HOWTO) above. In particular, ‘TCP Keepalive HOWTO’ teaches that setting up and using TCP keepalives is very simple. (EX1021, Section 2.1.) A timer is set up and when the timer reaches zero a problem packet is sent and you wait for a reply. (EX1021, Section 2.1.) If you receive a reply to the probe packet that connection is assumed to be up. (EX1021, Section 2.1). If you do not receive a reply from the probe packet, a count is kept of the unacknowledged probes, if this count exceeds a value “tcp_keepalive_probes”, the connection is considered dead. (EX1021, Section 3.). Thus, a POSITA using TCP keep-alives would be sending, by the first server, a ‘keep alive’ message to the second client device and waiting for a response.

4. Limitation 7: The method according to claim 6, further comprising: receiving, by the first server, the response to the ‘keep alive’ message from the second client device; and responsive to the receiving of the response, adding the IP address of the second client device to the list.

159. In my opinion, Kocherlakota-RFC 1122 renders obvious this limitation. As described above in Claim 2, Kocherlakota discloses how the client devices are identified over the Internet using a respective IP address. The addition of TCP keep-alives from RFC1122 would allow a POSITA to identify if a TCP connection were alive, and if it was alive, A POSITA would know the client device was online. As described in ‘TCP Keepalive HOWTO’ setting up and using TCP keepalives is very simple. (EX1021, Section 2.1). A timer is set up and when the timer reaches zero a problem packet is sent. (EX1021, Section 2.1). If you receive a reply to the probe packet that connection is assumed to be up. (EX1021, Section 2.1). If you do not receive a reply from the probe packet, a count is kept of the unacknowledged probes, if this count equals or exceeds a value “tcp_keepalive_probes”, the connection is considered dead. (EX1021, Section 3.). Thus, a connection is not determined to be down unless the number of unacknowledged probes reaches a set value. Thus, when there exists unacknowledged probes less than the value, tcp_keepalive_probes, the connection is neither alive or dead. In this state it would be prudent for the POSITA to remove this proxy from the list of available proxies, as opposed to leaving it on the available proxy list in case the proxy itself was dead.

160. If a keep-alive probe is subsequently received by proxy 17, the first server, a POSITA would then place the proxy from which the tcp-keepalive probe was received from back on the list of available proxies.

5. Limitation 8: The method according to claim 6, further comprising responsive to not receiving, by the first server, a response to the ‘keep alive’ message from the second client device, removing the IP address of the second client device from the list.

161. In my opinion, Kocherlakota-RFC 1122 renders obvious this limitation. As discussed for Limitation 7, it would have been obvious to remove the IP address of a non-response. If proxy 17 determines a connection is not active, then it may consider the session terminated and remove the IP address of the terminated device from the list of IP addresses of devices under active session. *Id.* Thus, proxy 17 maintains a list of IP addresses whose respective client devices are current returning TCP keep-alives.

6. Limitation 9: The method according to claim 6, wherein the sending of the ‘keep alive’ message comprises periodically sending by the first server the ‘keep alive’ message to the second client device.

162. In my opinion, Kocherlakota-RFC 1122 renders obvious this limitation RFC 1122 teaches that the TCP keep alive mechanism **periodically** probes the connection. EX1006, 102/116. (“A “**keep-alive**” mechanism **periodically** probes the other end of a connection when the connection is otherwise idle, even when there is no data to be sent.”) ‘TCP Keepalive HOWTO’ confirms that this is how a TCP keep-alive mechanism would work. (Section 3).

D. Ground 3 – Cohen Renders Obvious Claims 1-6, 10, 12, 15-19, 21-25

1. Limitation 1

- a. Limitation 1[p]: A method for use with a web server that stores a first content identified by a first content identifier, and for use with a first client device that is addressed over the Internet using first Internet Protocol (IP) address, for use with a list of IP addresses stored in the first server, the method by the first server comprising.**

163. In my opinion Cohen discloses or renders obvious this limitation. Cohen discloses a method by for fetching by a client 101-1 (the claimed “first client device”) a web page, (the claimed first content), identified by a first URL (the claimed first content identifier) associated with an origin server (the claimed “web server”) through proxy redirector 104. EX 1004 Fig. 1.

164. Cohen discloses proxy redirector 104 which intercepts a web request from a “client 101-1” seeking content from a target web server and establishes a TCP/IP connection between the client and the selected proxy.

(“In accordance with the present invention, once the IP address of the origin server corresponding to the logical URL name is determined through the DNS look-up, **proxy redirector 104**, rather than establishing a TCP connection to the origin server at the determined IP address, **transparently establishes a TCP connection between the client and a proxy.**”) EX1004, 7:12-19

165. Proxy selects one of the available proxy caches to which to forward client requests based on a metric such as least-loaded or round-robin, [or] based on IP header information such as the origin server IP address.

(“**Proxy redirector 104 selects one of the available proxy caches** to which to forward client requests...”)*Id.*, 43-47. *See* FIG.1 below.

166. A POSITA would have understood that when the proxy redirector selects one of the available proxy caches to which to forward client requests to it must contain a list of IP addresses for these proxy caches.

167. As discussed in Section X.D (“Cohen”) above, in order to select one of the many available proxy caches, proxy redirector 104 must store a list of the IP addresses associated with each proxy cache (the claimed “list of IP addresses”).

(“**Proxy redirector 104 selects one of the available proxy caches** to which to forward client requests based on a metric such as least-loaded or round-robin, based on IP header information such as the origin server IP address.”)*Id.*, 7:43-47;

168. Client 101-1 meets the construction of “client device” because it is a device that communicates over a network and it is operating in the role of a client by making requests for objects.

169. Proxy redirector 104 meets the construction of “first server” because it is a device operating in the role of a server by receiving client 101-1’s web requests

and sending the responses back to client 101-1 and because it is not the same physical device as client 110-1 (the claimed “first client device”). (“Responsive packets from the proxy cache are sent to proxy redirector 104 where they are redirected to client 101-1.”) Id., 8:57-59.

b. Limitation 1[a]: storing, operating, or using, a server operating system

170. In my opinion Cohen discloses this limitation. Cohen discloses that the claimed first server, proxy 104 stores, operates and using a server operating system, Linux. (“In the preferred embodiment described herein, **proxy redirector 104** includes a programmable network element of the type described in co-pending U.S. patent application Ser. No. 09/190,355, filed Nov. 12, 1998, which application is incorporated herein by reference. As described in that application, that programmable network element in the **preferred embodiment runs on a Linux machine.**”) EX1004 Col. 7:49-55. A POSITA would understand that a Linux operating system is a server operating system.

c. Limitation 1[b] receiving, from the first client device, the first content identifier

171. In my opinion Cohen discloses this limitation. Cohen discloses that the client sends “one or more packets to the origin server [via the proxy redirector 104], which packets include a GET request. The GET request includes a complete URL,

which identifies to that server the specific object within the origin server site that the client wants a copy of.” EX1004, 2:15-20.

172. As shown in Table 1 (annotated below), the proxy redirector 104 (the claimed “first server”) receives the GET request from client 101-1 (the claimed “first client device”) with the included complete URL “/a/ya/yahoomail/promo1.gif HTTP/1.0” (the claimed “first URL”) designated as the arriving packet. EX1004, 9:19-23; Table 1; see also *Id.*, 2:15-20.

TABLE 1

Arriving packet:

```
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.243 daddr=204.71.200.244
---> TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1, *,utf 8
```

d. Limitation 1[c]: selecting, an IP address from the list according to a criterion

173. In my opinion Cohen discloses this limitation. As shown in Table 1 (annotated below), upon receiving the URL, proxy redirector 104 (the claimed “first server”) selects the IP address “daddr=135.104.25.31” of a proxy cache 110-1 from the group of IP addresses associated with at least two or more proxies (i.e. proxies 110-1, 110-2, 115, and/or 117) to forward the URL to the selected proxy. EX1004, 11:12-15.

TABLE 1

Arriving packet:

```

--> beginning of packet header dump <--
--> IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
    frag_off=4000H TTL=64 protocol=6 cksum=1792H
    saddr=135.104.25.243 daddr=204.71.200.244
--> TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517
    tcp_hdr_len=5 flags=ACK PSH
    res1=0H res2=0H window=31856 cksum=162H urgent=0
--> beginning of packet data dump <--
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1, *,utf 8

```

Modified to:

```

--> beginning of packet header dump <--
--> IP header: version=4 hdr_len=5 TOS=0 pkt_len=367 id=60276
    frag_off=4000H TTL=64 protocol=6 cksum=1792H
    saddr=135.104.25.245 daddr=135.104.25.31
--> TCP header: sport=5000 dport=7000 seq=2189084427 ack_seq=3266449517
    tcp_hdr_len=5 flags=ACK PSH
    res1=0H res2=0H window=31856 cksum=162H urgent=0
--> beginning of packet data dump <--
GET http://204.71.200.244/a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com

```

174. As explained in Section X.D (“Cohen”), proxy redirector 104 uses a stored criterion when it “selects one of the available proxy caches.” EX1004, 7:43-47. For example, “[p]roxy redirector 104 selects one of the available proxy caches ... based on “a metric such as least-loaded or round-robin [or] IP header information such as the origin server IP address.” EX1004, 7:43-47. This selection is “according to a criterion” because, for example the selection of the URL can be “based on IP header information such as the origin server IP address. *Id.*, 7:36-48.

e. Limitation 1[d] sending, in response to the receiving of the first content identifier, using the selected IP address, the first content identifier

175. In my opinion Cohen discloses this limitation. As discussed for Limitation 1[c], the proxy redirector 104 selects the IP address of proxy cache 110-1 and, in response to the selecting, forwards the URL to selected proxy cache 110-1. EX1004, 7:36-48; *see also id.*, 11:12-15 (“As can be noted in the modified packet, the packet is redirected to a proxy cache 110-1 at IP address 135.104.25.31 port 7000 by changing the daddr and dport header information.”).

176. A POSITA would understand that when proxy redirector sends the URL to the first web server through to proxy cache 110-1, as an intermediary, proxy redirector **uses the IP address** associated with proxy cache 110-1 (the claimed “selected IP address”) as the destination IP address in the IP packets.

177. This sending of the URL is in response to the receiving of the first content identifier because the selection can be “based on IP header information such as the origin server IP address.” EX1004, 7:43-48. This origin server IP address corresponds to part of the URL and, thus, reflects analysis of the URL in the “selecting.”

f. Limitation 1[e]: receiving, in response to the sending of the first content identifier, the first content from the web server;

178. In my opinion Cohen discloses this limitation. As discussed for Limitations 1[c] and 1[d], proxy redirector 104 forwards the first URL to proxy cache 110-1, which establishes a TCP connection to the origin server to obtain the requested object (the claimed “first content”) if it does not have the stored content. A POSITA would understand that the function of a proxy such as proxy redirector 104 is both to forward URLs to web servers, as well as receive responses from web servers and return them to the requesting client. If not, then the client would not receive the object that it requested.

(“The GET request includes a complete URL, which identifies to that server the **specific object within the origin server site that the client wants a copy of.**”) EX1004, 2:17-19

(“[P]ackets from a client 101-1 are transparently directed by proxy redirector 104 to a proxy cache. Responsive packets from the proxy cache are sent to proxy redirector 104 where they are redirected to client 101-1.”)

EX1004, 8:55-58; see id., 6:39-46, 10:23-25, 12:18-20.

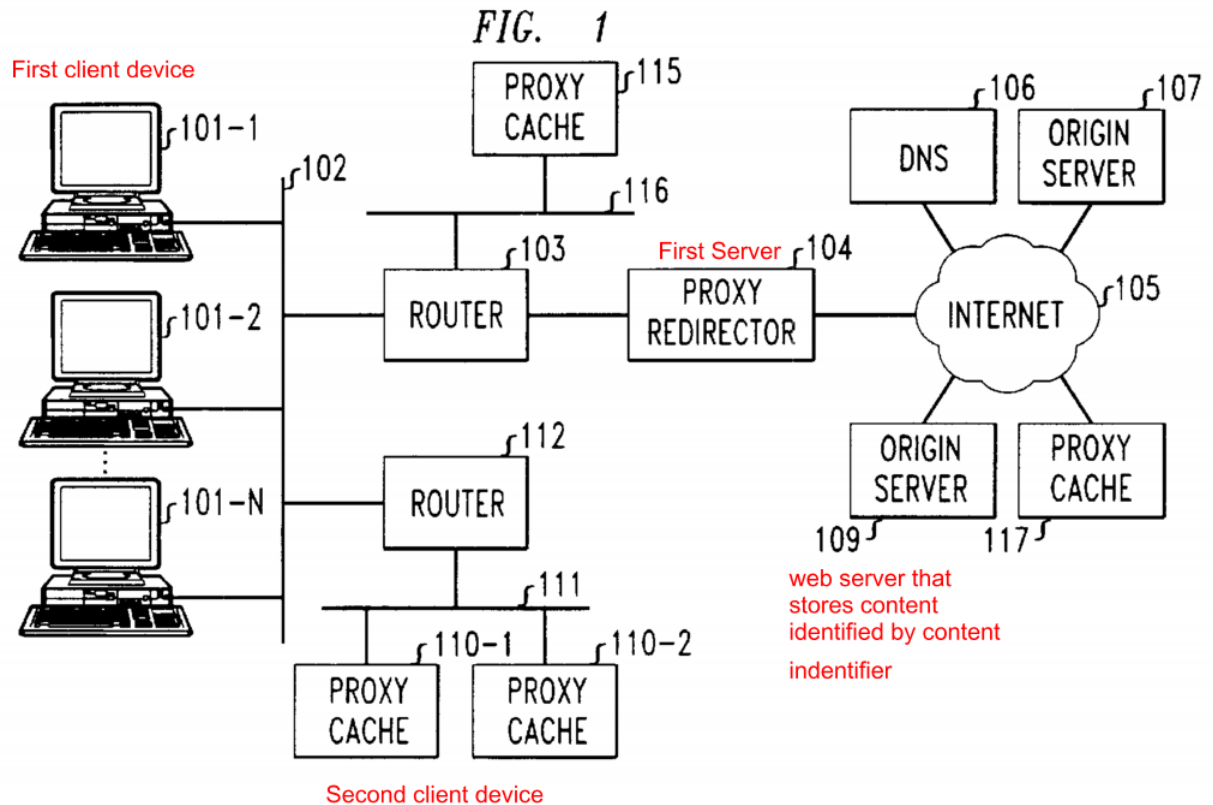
179. Cohen further explains that the proxy cache returns the “requested objects” (i.e., the claimed “first content”) to the address from which a request originated. EX1004, 16:28-41.

g. Limitation 1[f]: sending the received first content, or a part thereof, to the first client device.

180. In my opinion Cohen discloses this limitation. Cohen discloses that the “invention relates to... a network for transparently intercepting **client web requests** and redirecting them to proxy caches.” EX1004, 1:6-10 As explained above for Limitation 1[e], A POSITA would understand that the function of a proxy such as proxy redirector 104 is both to forward URLs to web servers, as well as receive responses from web servers and return them to the requesting client. If not, then the client would not receive the object that it requested. (“[r]esponsive packets from the proxy cache are sent to proxy redirector 104 where **they are redirected to client 101-1.**”) EX1004, 8:55-58. Thus, the proxy redirector 104 returns the “requested objects” (i.e., the claimed “first content”) to client 101-1. EX1004, 16:28-41.

2. Limitation 2: The method according to claim 1, further for use with a second client device that is addressable in the Internet using the selected IP address.

181. In my opinion Cohen discloses or renders obvious this limitation. Proxies 110-1, 110-2, 115, and/or 117 each meet the construction of “client device” because each is a device that communicates over the Internet and operate in the role of a client by establishing a connections between the origin server and the proxy redirector 104. See Limitation 1[c]; *see also* EX1005, 8/114 (proxy devices “act[s] as both a server and a client”).



182. As discussed in Section X.D (“Cohen”) above, to be selectable by the proxy redirector as Cohen describes, a POSITA would have understood that each of the proxies 110-1, 110-2, 115, and/or 117 are addressable using a respective IP address. EX1015, 17/51; EX1016, 57/91. (“As can be noted in the modified packet, the packet is redirected to a proxy cache 110-1 at **IP address 135.104.25.31** port 7000 by changing the daddr and dport header information.”) EX1004, 11:12-15.

3. Limitation 3: The method according to claim 2, wherein the sending of the first URL to the first web server comprises sending, by the first server, the first URL to the client device addressed by the selected IP address.

183. In my opinion Cohen discloses or renders obvious this limitation. As explained in Section X.D (“Cohen”) and Limitations 1[c] and 1[d], the process of proxy redirector 104 sending the GET request (containing the first URL, e.g. “/a/ya/yahoomail/promo1.gif HTTP/1.0”) to the origin server (e.g. origin server 107 or 109) includes proxy redirector 104 (the claimed “first server”) sending the first URL (the claimed “first content identifier”) to proxy cache 110-1 (the claimed “second client device”). (“As can be noted in the modified packet, the **packet is redirected** to a proxy cache 110-1 at **IP address 135.104.25.31** port 7000 by changing the daddr and dport header information.’) EX1004, 11:12-15.

4. Limitation 4: The method according to claim 3, further comprising: receiving, by the second client device from the first server, the first content identifier;

sending, by the second client device from the web server, the first content;

and sending, by the second client device to the first server, the received first content or a part thereof.

184. In my opinion Cohen discloses or renders obvious this limitation. As discussed in explained for Limitation 3, the proxy redirector 104 sends the GET request (containing the first URL, e.g. “/a/ya/yahoomail/promo1.gif HTTP/1.0”) to the proxy cache 110-1 (the claimed “client device”). The proxy cache 110-1 receives

the GET request with the URL (the claimed “first content identifier”) from proxy redirector 104 (the claimed “first server”). EX1004, 7:27-35.

185. As explained for Limitation 1[d], the process of proxy redirector 104 sends the GET request containing the URL to the origin server includes proxy cache 110-1 sending the URL to the web server through a TCP connection that “is established between the proxy cache and the origin server to obtain a copy of the requested object.” EX1004, 7:27-35

186. Also as explained for Limitation 1[e], proxy cache 110-1 receives from the origin server corresponding to the first URL the web page corresponding to the first URL from the origin and proxy redirector 104 receives the web page from proxy cache 110-1. EX1004, 8:55-58; *see also id.*, 6:39-46, 10:23-25, 12:18-20.

187. Thus, the web page is sent from the origin server (the claimed “web server”) to proxy cache 110-1 and then is sent, again, by proxy cache 110-1 (the claimed “second client device”) to proxy redirector 104 (the claimed “first server”).

5. Limitation 5: The method according to claim 2, further comprising by the first server: communicating with the second client device;

and storing an IP address of the second client device in the list in response to the communicating.

188. In my opinion Cohen discloses or renders obvious this limitation. As explained in Proxy redirector 104 (the claimed “first server”) establishes and maintains connections to determine availability, and thereby communicates, with

each of the proxy caches 110-1, 110-2, 115, and/or 117, (the claimed “second client devices”). (“Proxy redirector 104 **selects one of the available proxy caches** to which to forward client requests based on a metric such as least-loaded or round-robin, based on IP information such as the origin server IP address.”) EX1004, 7:36-48.

189. As discussed in Section X.D (“Cohen”), a POSITA would understand that when proxy redirector 104 “selects one of the available proxy caches [e.g. 110-1, 110-2, 115, and/or 117] to which to forward client requests, ” such connection necessarily requires, or at a minimum renders obvious, proxy redirector 104 storing a group of IP addresses for the proxies. EX1004, 7:36-48, 2:15-20. Further, a POSITA would also understand that when proxy redirector 104 “transparently establishes a TCP connection between the client and a proxy” such connection necessarily requires, or at a minimum, renders obvious, proxy redirector 104 storing the IP address of the proxy. EX1004, 7:12-17.

6. *Limitation 6. The method according to claim 1, for use with a second client device, the method further comprising:*

sending, by the first server, a ‘keep alive’ message to the second client device; and

waiting, by the first server, for a response to the ‘keep alive’ message from the second client device

190. In my opinion Cohen discloses or renders obvious this limitation. As shown in Table 1 (annotated below), proxy redirector 104 establishes a “keep alive”

connection with the proxies. EX1004, 11:12-15. This HTTP “keep-alive” feature allows the HTTP client (i.e., the cache proxy) to send multiple request/response pairs over the same TCP connection. This decreases latency for subsequent HTTP requests since the proxy redirector 104 (the claimed “first server”) does not close the TCP connection after each response but waits some time to allow for additional HTTP requests.

TABLE 1

Arriving packet:

```
--->  beginning of packet header dump <---
--->  IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.243 daddr=204.71.200.244
--->  TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
--->  beginning of packet data dump <---
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1, *,utf 8
Modified to:
```

```
--->  beginning of packet header dump <---
--->  IP header: version=4 hdr_len=5 TOS=0 pkt_len=367 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.245 daddr=135.104.25.31
--->  TCP header: sport=5000 dport=7000 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
--->  beginning of packet data dump <---
GET http://204.71.200.244/a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
```

191. A POSITA would have been motivated to make use of this “keep-alive” functionality with each of the proxy caches (e.g., 110-1, 110-2, 115, and/or 117) to help maintain TCP connections by preventing periods of inactivity and avoid unnecessarily expending resources to reestablish connections for each HTTP request.

192. After sending the HTTP request over a connection of type ‘Keep-Alive’ the, proxy redirector 104 (the claimed “first server”) waits for response from the selected proxy cache.

7. Limitation 10: The method according to claim 1, wherein the criterion is stored in the first server.

193. In my opinion Cohen discloses or renders obvious this limitation. As explained in Section X.D (“Cohen”) and Limitation 1[c], proxy redirector 104 uses a stored criterion during the selection, such as “a metric such as least-loaded or round-robin [or] IP header information such as the origin server IP address.” EX1004 7:36-48.

8. Limitation 12: The method according to claim 10, wherein the criterion is based on, or comprises, a geographical location of a client device addressed by the selected IP address, or a response time when communicating with the client device addressed by the selected IP address.

194. In my opinion Cohen discloses or renders obvious this limitation. Cohen discloses or renders obvious this limitation. As explained in Section X.D (“Cohen”), a POSITA would have understood that, “a metric such as least-loaded” is based on a response time when communicating. (“Proxy redirector 104 selects one of the available proxy caches to which to forward client requests based on a metric such as **least-loaded** or round-robin, **based on IP header information such as the origin server IP address.**”) EX1004, 7:36-48;

195. The '826 Patent teaches that the “second clients” the agents are chosen by being numerically close, which a POSITA would understand to nearby in some manner. EX1001, 13:25-29 (“The list of agents is created by the acceleration server 162 by finding the communication devices of the communication network 100 that are currently online, and whose **IP address is numerically close to the IP of the destination Web server 152.**”). A POSITA would understand that when Cohen teaches basing the selection of the proxy server on the origin server’s IP address, it would have been obvious to choose a proxy server whose IP address was numerically close to the origin server’s IP address.

9. Limitation 15: The method according to claim 1, wherein the first server is a Transmission Control Protocol/Internet Protocol (TCP/IP) server that communicates over the Internet with the first client device based on, according to, or using, TCP/IP protocol or connection.

196. In my opinion Cohen discloses or renders obvious this limitation. Cohen refers throughout to the use of TCP/IP, including with respect to proxy redirector 104 (“first server”) and the client devices. For example, “once the IP address of the origin server corresponding to the logical URL name is determined through the DNS look-up, proxy redirector 104, rather than establishing a TCP connection to the origin server at the determined IP address, transparently establishes a TCP connection between the client and a proxy.” EX1004, 7:12-17. Table 1 shows that the arriving packet from the first client has a protocol value of 6. A POSITA would know that protocol 6 is reserved for TCP.

TABLE 1

Arriving packet:

```
--->  beginning of packet header dump <---  
--->  IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276  
      frag_off=4000H TTL=64 protocol=6 cksum=1792H  
      saddr=135.104.25.243 daddr=204.71.200.244  
--->  TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517  
      tcp_hdr_len=5 flags=ACK PSH  
      res1=0H res2=0H window=31856 cksum=162H urgent=0  
--->  beginning of packet data dump <---  
GET /a/ya/yahoomail/promo1.gif HTTP/1.0  
Referer: http://www.yahoo.com/  
Connection: Keep-Alive  
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)  
Host: us.yimg.com  
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png  
Accept-Language: en  
Accept-Charset: iso-8859-1, *,utf 8  
Modified to:
```

197. A POSITA would therefore understand that protocol 6 in this message refers the first server using the TCP/IP protocol to communicate with the first client.

10. Limitation 16: The method according to claim 1, wherein the first server communicates over the Internet based on, or according to, one out of UDP, DNS, TCP, FTP, POP*, SMTP, or SQL standards.

198. See Limitation 15 above for Cohen teaching communicating according to TCP standards.

11.Limitation 17: The method according to claim 1, wherein the first content comprises web-page, audio, or video content, and wherein the first content identifier comprises a Uniform Resource Locator (URL).

199. See Limitation 1[p] above for Cohen teaching the first content includes a web page and the first content identifier includes a URL.

12. Limitation 18: The method according to claim 1, for use with a plurality of client devices that includes a second client device, each identified over the Internet using a respective IP address from the list, wherein the method further comprising:

selecting, by the first server, multiple client devices from the plurality of client devices by selecting the IP addresses of the multiple client devices from the list;

and sending, by the first server, the selected IP addresses in response to the selecting

200. In my opinion Cohen discloses or renders obvious this limitation. As explained in Limitation 2 above, proxy redirector 104 utilizes multiple proxies, such as proxy caches 110-1, 110-2, 115, and/or 117, each of which meet the construction of “second client device” and are identified over the Internet using the IP address that are selected from the list of IP addresses stored in proxy redirector 104. See Limitation 2 above where the second client devices are addressable and therefore identified by their IP address.

201. Cohen discloses that the proxy redirector selects from multiple client devices from the plurality of client devices by selecting the IP addresses of the multiple client devices from the list. EX1004, 1:6-10. (“Proxy redirector 104 selects one of the available proxy caches to which to forward client requests based on a metric such as least-loaded or round-robin, based on IP information such as the origin server IP address.”) EX1004, 7:36-48.

202. As explained for Limitation 1[c], the process of proxy redirector 104 selecting proxy caches 110-1, 110-2, 115, and/or 117 includes selecting the IP address of the proxy caches from the list of those IP addresses to route the URL through the proxy chain. For example, “([p]roxy redirector 104 **selects one** of the **available proxy caches** ... based on “a metric such as least-loaded or round-robin [or] IP header information such as the origin server IP address.”) EX1004 7:42-46.

203. Upon receiving the multiple URLs, proxy redirector 104 (the claimed “first server”) selects an IP address corresponding to a proxy cache from the group of IP addresses associated with two or more proxies (i.e., proxies 110-1, 110-2, 115, and/or 117) to forward the URLs to the selected proxies. EX1004, 7:43-48; 11:12-15. Proxy redirector may select a different IP address (the claimed “IP addresses of the multiple client devices”) corresponding to different proxy caches for each URL (e.g., 110-1 and 110-2).

204. As explained in Section X.D (“Cohen”) and for Limitation 1[d] above, proxy redirector 104 sends the URL to proxy cache 110-1 in response to the selecting. Communications by proxy redirector 104 to each proxy cache use the stored IP address of the respective proxy cache as the “destination address” in accordance with IP protocols. See EX1004 Table 1, also

TABLE 1

```

Arriving packet:
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.243 daddr=204.71.200.244
---> TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1, *,utf 8
Modified to:
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=367 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.245 daddr=135.104.25.31
---> TCP header: sport=5000 dport=7000 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET http://204.71.200.244/a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com

```

205. When proxy redirector 104 sends IP packets to proxy cache 110-1 and proxy cache 110-2, it uses the IP address of proxy cache 110-1 and proxy cache 110-2, respectively, (the claimed “selected IP address”) as the “destination address.” The “destination address” is part of the IP header and is thus sent with the IP packet as a POSITA would know that the IP destination address is part of the IP packet. (See also EX1015, 17/51; EX1016, 57/91)

13.Limitation 19: The method according to claim 1, further comprising sending of the selected IP addresses to the first client device.

206. In my opinion Cohen discloses or renders obvious this limitation. Cohen teaches an alternative where the proxy is modified so that the proxy cache knows the client IP address, (the claimed first client). (“If, however, the proxy is modified, using a programmable network element as previously described, for example, the requested object retrieved by the proxy from its own cache or received from an origin server, can be sent directly back to the client, thereby obviating the need to send such packets back to the proxy redirector for address translations and redirection to the client”) EX1004 16:44-51. In this way the packets from the proxy cache are sent directly back to the client. EX1004 16:67 – 17:1. A POSITA would understand that one obvious way to do this would be for the packets to sent directly from the proxy cache back to the client on another TCP connection, using the IP address of the proxy cache as the source address in the IP packet. A POSITA would be motivated to do this to reduce delay. EX1004 17:1-3. In this manner, the selected IP address is sent to the first client.

14. Limitation 21: The method according to claim 1, wherein the first client device is identified by a first Internet Protocol (IP) address, Media Access Control (MAC) address, or a hostname.

207. In my opinion Cohen discloses or renders obvious this limitation. Client 101-1 (the claimed “first client device”) is identified by a first IP address—

specifically its own IP address—in TCP sessions with proxy redirector 104.

EX1004, Table 1.

TABLE 1

Arriving packet:

```
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=346 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.243 daddr=204.71.200.244
---> TCP header: sport=1273 dport=80 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET /a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg image/png
Accept-Language: en
Accept-Charset: iso-8859-1, *,utf 8
Modified to:
```

```
---> beginning of packet header dump <---
---> IP header: version=4 hdr_len=5 TOS=0 pkt_len=367 id=60276
      frag_off=4000H TTL=64 protocol=6 cksum=1792H
      saddr=135.104.25.245 daddr=135.104.25.31
---> TCP header: sport=5000 dport=7000 seq=2189084427 ack_seq=3266449517
      tcp_hdr_len=5 flags=ACK PSH
      res1=0H res2=0H window=31856 cksum=162H urgent=0
---> beginning of packet data dump <---
GET http://204.71.200.244/a/ya/yahoomail/promo1.gif HTTP/1.0
Referer: http://www.yahoo.com/
Connection: Keep-Alive
User-Agent: Mozilla/4.05 [en] (X11; U; Linux 2.1.103 i686)
Host: us.yimg.com
```

15.Limitation 22: The method according to claim 1, wherein the web server is a Hypertext Transfer Protocol (HTTP) server responding to HTTP requests

208. In my opinion Cohen discloses or renders obvious this limitation. As explained in Sections VII (Overview of the Technical Field) and X.D (“Cohen”), a

POSITA would understand Cohen's disclosure of web requests and responses to disclose HTTP requests and responses.

209. For example, Cohen explains that "HTTP connection requests are generally directed to port 80 of an origin server." EX1004, 3:3-4. Table 1 of Cohen and the surrounding text illustrates an example "HTTP" request directed to a web server at www.yahoo.com with a particular IP address. EX1004, Table 1, 10:33-11:23.

16.Limitation 23: The method according to claim 1, wherein the steps are sequentially executed

210. In my opinion Cohen discloses or renders obvious this limitation. As discussed for Limitations 1[p]-1[f] above, Cohen discloses proxy redirector 104 performing the steps of Claim 1 in a sequential manner. The claimed steps follow the known function of a proxy—sending URLs to web servers, receiving responses from web servers, and returning them to the requesting client in a sequential order. See EX1005, Page 10.

17.Limitation 24: The method according to claim 1, for use with a software application that includes computer instructions that, when executed by a computer processor, cause the processor to perform the steps of the claim 1.

211. In my opinion Cohen discloses or renders obvious this limitation. See Limitation [1a]. Cohen discloses that the proxy redirector ("first server") may be a Level 4 switch that also may be combined into the same unit as a router. EX1004,

6:23-31. The computer functions may be carried out by “software in any form, including, therefore, firmware, microcode or the like, combined with appropriate circuitry for executing that software to perform the function.” EX1004 18:1-3. The computer functions are executed by processors. EX1004, 17:36-56. Thus, the processors disclosed in Cohen would execute computer instructions and perform the recited functions.

212. Further, a POSITA would understand that, as is the nature of computing devices, proxy redirector 104 would include a processor to execute the computer instructions from the software and perform the steps of claim 1.

18.Limitation 25: A non-transitory computer readable medium containing computer instructions that, when executed by a computer processor, cause the processor to perform the method according to claim 1.

213. Cohen discloses or renders obvious this limitation. See Limitations [1a], [24]. The processes disclosed in Cohen “may be substantially represented in computer readable medium and so executed by a computer or processor:” EX1004, 17:36-41.

214. Further, a POSITA would understand that, as is the nature of computing devices, proxy redirector 104 would include a processor and a non-transitory computer readable medium containing computer instructions that are to be executed by the processor to cause the processor to perform the steps

E. Ground 4 – Cohen in View of RFC 1122 Renders Obvious Claims 6-9.

215. In my opinion Cohen in view of RFC 1122 renders obvious claims 6-9.

1. Motivation to combine

216. A POSITA would have been motivated to use the TCP keep-alive functionality of RFC 1122 as discussed in Section X.B (“RFC 1122”), above, when maintaining the TCP connections as disclosed by Cohen. Both references cover connecting Internet hosts over the Internet. RFC 1122 describes an Internet host as “A host computer, or simply "host," is the ultimate consumer of communication services.” EX1006, Section 1.1.1, page 6. Cohen describes client requesting objects stored in servers connected to the Internet. (“All requests from any of the clients connected to LAN 102 for objects stored in servers connected to the Internet 105 traverse proxy redirector 104 onto the Internet. The packets comprising such requests, which include the standardized packets that establish a TCP connection, are directed to an IP destination address and port number indicated in the IP header of each packet originating from a client source address that includes a client IP address and port number.”) EX1004 6:33-39. Cohen specifically teaches that the proxy redirector should only select for available proxy caches. (“([p]roxy **redirector** 104 **selects** one of the **available proxy caches** ... based on “a metric such as least-loaded or round-robin [or] IP header information such as the origin server IP address.”) EX1004 7:42-46.

217. A POSITA would have recognized the benefits of Cohen's proxy redirector exchanging TCP keep-alives with the proxy caches 110-1, 110-2, 117 to know if these proxy caches were available. One benefit, TCP keep-alives help detect when "a client crashes or aborts a connection during a network failure" and thus prevent server applications from "hang[ing] indefinitely and consum[ing] resources unnecessarily." EX1006, 102/116. For example, following the establishment of connections between the proxy redirector and the proxy caches, one of the proxy caches could crash. This could lead to the situation where the proxy redirector sends Internet traffic to a proxy cache that no longer operational. This could lead to lost messages and unnecessarily expend resources keeping the "dead" TCP connection open. Another benefit a POSITA would have known of is that TCP keep-alives could also prevent periods of inactivity between the proxy redirector and the proxy caches from causing the TCP connection between them to terminate. Which again could result in lost messages.

218. RFC 1122 does specify some potential disadvantages of TCP keep-alives, but a POSITA would have recognized that these disadvantages are outweighed by the benefits listed above in a system such as Cohen's, which explicitly disclose the need of keep-alive.

2. Modification of Cohen in view of RFC 1122

219. A POSITA would have modified Cohen so that the TCP connections between the proxy redirector and the proxy caches would periodically communicate by exchanging TCP keep-alive messages with each other over established TCP connections. A "keep-alive" mechanism was a well known technique which periodically probes the other end of a connection when the connection is otherwise idle, even when there is no data to be sent. EX1006, 102/116. In the proxy chain comprising a client computer 101 -> proxy redirector 104 -> proxy cache 117, the proxy redirector 104 would periodically send client computer 101 and proxy caches 117 keep-alive messages over the TCP connections between them. Further, in accordance with RFC 1122, the proxy 17 would wait for an "acknowledgment" from proxy caches, confirming that the connection is still live. *See* EX1006, 102/116

220. Such a modification would be simple for a POSITA, as all that is required is to send a probe segment designed to elicit a response from the peer TCP periodically. EX1006, 102/116.

F. Element-by-Element Analysis

- 1. Limitation 6. The method according to claim 1, for use with a second client device, the method further comprising: sending, by the first server, a 'keep alive' message to the second client device; and waiting, by the first***

server, for a response to the ‘keep alive’ message from the second client device

221. In my opinion Cohen-RFC 1122 renders obvious this limitation. As discussed above proxy redirector 104 would periodically send client computer 101 and proxy caches 117 keep-alive messages over the TCP connections and would wait for an “acknowledgment” from proxy caches, confirming that the connection is still live.

2. Limitation 7: The method according to claim 6, further comprising: receiving, by the first server, the response to the ‘keep alive’ message from the second client device; and responsive to the receiving of the response, adding the IP address of the second client device to the list.

222. In my opinion, Cohen-RFC 1122 renders obvious this limitation. As described above in Ground 3, Limitation 2, Cohen discloses how the proxy caches are identified over the Internet using a respective IP address. The addition of TCP keep-alives from RFC1122 would allow a POSITA to identify if a TCP connection were alive, and if it was alive, A POSITA would know the proxy cache was available and online. See EX1006, 102/116. As described in ‘TCP Keepalive HOWTO’ setting up and using TCP keepalives is very simple. (EX1021, Section 2.1). A timer is set up and when the timer reaches zero a problem packet is sent. If you receive a reply to the probe packet that connection is assumed to be up. (EX1021, Section 2.1). If you do not receive a reply from the probe packet, a count is kept of the unacknowledged probes, if this count equals or exceeds a value

“tcp_keepalive_probes”, the connection is considered dead. (EX1021, Section 3.). Thus, a connection is not determined to be down unless the number of unacknowledged probes reaches a set value. Thus, when there exists unacknowledged probes less than the value, tcp_keepalive_probes, the connection is neither alive or dead. In this state it would be prudent for the POSITA to remove this proxy cache from the list of available proxies, as opposed to leaving it on the available proxy cache list in case the proxy cache itself was dead.

223. If a keep-alive probe is subsequently received by proxy redirector, the first server, a POSITA would then place the proxy cache from which the tcp-keepalive probe was received from back on the list of available proxies.

3. Limitation 8: The method according to claim 6, further comprising responsive to not receiving, by the first server, a response to the ‘keep alive’ message from the second client device, removing the IP address of the second client device from the list.

224. In my opinion, Cohen-RFC 1122 renders obvious this limitation. As discussed for Limitation 7, it would have been obvious to remove the IP address of a non-response. If the proxy redirector determines a connection is not active, then it may consider the connection terminated and remove the IP address of the terminated proxy caches from the list of IP addresses of proxy caches devices under active connections. Thus, proxy redirector maintains a list of IP addresses whose respective proxy caches are current returning TCP keep-alives.

4. Limitation 9: The method according to claim 6, wherein the sending of the ‘keep alive’ message comprises periodically sending by the first server the ‘keep alive’ message to the second client device.

225. In my opinion, Cohen-RFC 1122 renders obvious this limitation RFC 1122 teaches that the TCP keep alive mechanism **periodically** probes the connection. EX1006, 102/116. (“A **"keep-alive"** mechanism **periodically** probes the other end of a connection when the connection is otherwise idle, even when there is no data to be sent.”) ‘TCP Keepalive HOWTO’ confirms that this is how a TCP keep-alive mechanism would work. (EX1021, Section 3).

XII. Conclusion

226. I declare that all statements made herein of my knowledge are true, and that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

11/16/2023

Date

James Olivier Ph.D.

James Olivier, Ph.D.