UNITED STATES PATENT AND TRADEMARK OFFICE

_____

BEFORE THE PATENT TRIAL AND APPEAL BOARD

_____

New Bay Capital, LLC,

Petitioner,

v.

VirnetX Inc.

Patent Owner.

_____

IPR2013- 00376

Patent 7,490,151
Issue Date: Feb. 10, 2009
Title: ESTABLISHMENT OF A SECURE COMMUNICATION LINK BASED ON A DOMAIN NAME SERVICE (DNS) REQUEST

_____

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 7,490,151

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
United States Patent and Trademark Office
PO Box 1450
Alexandria, Virginia 22313–1450

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Cases

Statutes

Treatises

Regulations

EXHIBIT LIST

Exhibit	Description
1001.	U.S. Patent No. 7,490,151 to Munger et al. (hereinafter "the '151 Patent)
1002.	Takahiro Kiuchi and Shigekoto Kaihara, "C-HTTP - The Development of a Secure, Closed HTTP-based Network on the Internet," published by IEEE in the Proceedings of SNDSS 1996 (hereinafter "Kiuchi")
1003.	C. I. Dalton and J. F. Griffin, "Applying Military Grade Security to the Internet," published in the Proceedings of JENC8, May 1997 (hereinafter "Dalton")
1004.	W. Richard Stevens, "TCP/IP Illustrated, Volume 1: The Protocols" (Reading, Mass.: Addison-Wesley, 1994), page 187
1005.	Windows Sockets – An Open Interface for Network Programming Under Microsoft Windows, Version 1.1, January 20, 1993 (copy obtained from http://www.sockets.com/winsock.htm)
1006.	eCOS Reference Manual, Chapter 38, TCP/IP Library Reference, gethostbyname, March 13, 1997 (copy obtained from http://www.ecos.sourceware.org/docs-2.0/ref/net-common-tcpip-manpages-gethostbyname.html)
1007.	U.S. Patent No. 6,449,657 to Stanbach, Jr. et al. (hereinafter "the '657 Patent)
1008.	U.S. Patent No. 6,546,003 to Farris (hereinafter "the '003 Patent)
1009.	Declaration of Russell Housley with Appendices
1010.	Braden, "Requirements for Internet Hosts – Application and Support," Internet Engineering Task Force Request for Comments (RFC) 1123, October 1989.

1011.	MS Markman Order 7/30/09
1012.	Apple Pretrial Proceedings 10/18/12
1013.	Apple Transcript Morning 10/31/12
1014.	Apple Transcript Morning 11/1/12
1015.	Apple Memorandum Opinion and Order 2/26/13
1016.	Cisco/Apple Memorandum Opinion and Order 4/25/12
1017.	VirnetX Opening Claim Construction Brief in Apple/Cisco 11/4/11
1018.	VirnetX Reply Claim Construction Brief in Apple/Cisco 12/19/11
1019.	Microsoft Transcript Morning 3/9/10
1020.	Microsoft Transcript Afternoon 3/9/10
1021.	Definition of VPN (Virtual Private Network) from Computer Desktop Encyclopedia, 9th Edition, The McGraw-Hill Companies, 2001, page 1044
1022.	The '135 Patent Family
1023.	Definitions of HTML and HTTP from Computer Desktop Encyclopedia, 9th Edition, The McGraw-Hill Companies, 2001, pp. 435-436
1024.	C. I. Dalton and J. F. Griffin, "Applying Military Grade Security to the Internet," republished in Computer Networks and ISDN Systems, Vol. 29, No. 15, November 1, 1997 (pp. 1799-1808) (hereinafter "Dalton")

I. COMPLIANCE WITH FORMAL REQUIREMENTS

A. REAL PARTIES-IN-INTEREST

Petitioner New Bay Capital, LLC ("New Bay" or "Petitioner") requests inter partes review for claims 1 and 13 of U.S. Patent No. 7,490,151 (the '151 patent," attached as Ex.1001). The present assignee of the '151 patent is VirnetX, Inc. Petitioner certifies that the real parties-in-interests are New Bay and its parent Eastern Shore Capital, LLC.

B. STANDING

Petitioner certifies that the '151 patent, issued on February 10, 2009, is available for inter partes review and that Petitioner is not barred or estopped from requesting an inter partes review challenging the claims of the '151 patent.

C. RELATED MATTERS

Ex.1022 lists pending applications and reexaminations that may be affected by the outcome of this review. Additionally, the '151 patent has been asserted against the following companies in the following proceedings:

1. Apple, Inc. in: *VirnetX Inc. v. Cisco Systems, et al.*, Case No. 6:10cv417 (E.D. Tex.) filed August 11, 2010; *VirnetX Inc., et al. v. Apple Inc.*, Case No. 6:12cv855 (E.D. Tex.), filed November 6, 2012; and *VirnetX Inc. v. Apple Inc.*, Case No. 6:13cv211 (E.D. Tex.), filed February 26, 2013. The District Court in Case No. 6:10cv417 has entered a Final Judgment finding, *inter alia*, that Apple

infringed claims 1 and 13 of the '151 patent and that such claims were not invalid over Kiuchi, which is asserted in this proceeding based on new evidence and analysis.

2. Also, Cisco Systems, Inc. in *VirnetX Inc. v. Cisco Systems, et al.*, Case No. 6:10cv417 (E.D. Tex.), filed August 11, 2010.

3. Microsoft Corporation in *VirnetX Inc. et al. v. Microsoft Corporation*, Case No. 6:13cv351 (E.D. Tex.), filed April 22, 2013.

Also, the '151 patent is the subject of two merged pending inter partes reexaminations, 95/001,714 brought by Cisco Systems, and 95/001,697 brought by Apple, Inc. Claims 1 and 13 currently stand rejected in the merged reexaminations. The outstanding rejections include anticipation and obviousness rejections that apply Kiuchi as a primary reference. Neither of the pending reexaminations has reached the stage of an Action Closing Prosecution. In these reexaminations, the requesters are contesting all 16 claims of the patent, and have asserted more than a dozen prior art references in various combinations. The present Petition is, by contrast, highly streamlined in that it focuses on only two claims and relies on two prior art references (i.e., Kiuchi and Dalton). The present Petition advances new arguments and evidence (not presented in the litigations or pending reexaminations) for invalidating claims 1 and 13 over Kiuchi and Dalton in view of Kiuchi.

As a result of the cross-collateral estoppel provisions of 35 U.S.C. 317(b), the pending reexaminations will likely terminate before either reaches an enforceable result. The District Court in Case No. 6:10cv417 has already entered Final Judgments, finding that each of the requesters (Cisco and Apple) failed to prove the invalidity of the '151 patent. Given that the pending reexaminations have not even reached the stage of an Action Closing Prosecution, it is unlikely that the reexaminations will have time to run their full course (i.e., completing proceedings at the Examiner level, the Board level, and the Federal Circuit level) before the District Court judgments become "final," thereby necessitating termination of the reexaminations under 35 U.S.C. 317(b).

Also, the '151 Patent is the subject of a petition for inter partes review (IPR) filed by Apple on June 17, 2013 (case no. IPR2013-00354). Petitioner expressly requests that Petitioner's IPR **NOT** be joined or consolidated with the newly-filed Apple IPR or otherwise delayed or suspended because (i) Apple's IPR present a timeliness question that Petitioner's IPR does not present, (ii) Petitioner's IPR asserts different art than the Apple IPR, and (iii) Petitioner's IPR involves far fewer claims, and, as a result of (i)-(iii), consolidating the proceedings will unduly complicate matters and will place an undue burden on Petitioner to complete its IPR within the 1 year mandate.

Also, Petitioner is concurrently filing IPR requests directed to related U.S. Patent Nos. 6,502,135; 7,418,504; and 7,921,211, and requests that the reviews of the '151 Patent and U.S. Patent No. 6,502,135 be assigned to the same Board for administrative efficiency.

D. NOTICE OF LEAD AND BACKUP COUNSEL

Lead Counsel for the Petitioner is Robert M. Asher, Reg. No. 30,445, of Sunstein Kann Murphy and Timbers LLP. Back-up counsel for the Petitioner is Jeffrey Klayman, Reg. No. 39,250, of Sunstein Kann Murphy and Timbers LLP.

E. SERVICE INFORMATION

New Bay may be served through its counsel via email to rasher@sunsteinlaw.com and jklayman@sunsteinlaw.com or otherwise to

Robert M. Asher
Jeffrey Klayman
Sunstein Kann Murphy & Timbers LLP
125 Summer Street
Boston, MA 02110-1618
617 443 9292 (phone)
617 443 0004 (fax)

F. PROOF OF SERVICE ON THE PATENT OWNER

A copy of the present Petition, in its entirety, is being served to the Patent Owner's address of the attorney of record.

II. FEE

The undersigned authorizes the Director to charge the fee specified by 37 C.F.R. §42.15(a), and any additional fees due in connection with this Petition, to Deposit Account No. 19-4972.

III. STATEMENT OF PRECISE RELIEF REQUESTED

Request 1 – Cancellation of claims 1 and 13 under 35 U.S.C. §103 for obviousness over Takahiro Kiuchi and Shigekoto Kaihara, "C-HTTP - The Development of a Secure, Closed HTTP-based Network on the Internet," published in the Proceedings of SNDSS 1996 (hereinafter "Kiuchi" –Ex.1002).

Request 2 – Cancellation of claims 1 and 13 under 35 U.S.C. §102(b) as being anticipated by Kiuchi.

Request 3 - Cancellation of claims 1 and 13 under 35 U.S.C. §103(a) for obviousness over C. I. Dalton and J. F. Griffin, "Applying Military Grade Security to the Internet," published in the Proceedings of JENC8, May 1997 (hereinafter "Dalton" –Ex.1003) in view of Kiuchi.

IV. FACTUAL BACKGROUND

A. Admitted Prior Art Domain Name Services

Figure 25 of the '151 patent, labeled "Prior Art" and reproduced below, discloses aspects of domain name service systems that were well known at the time of the patent:

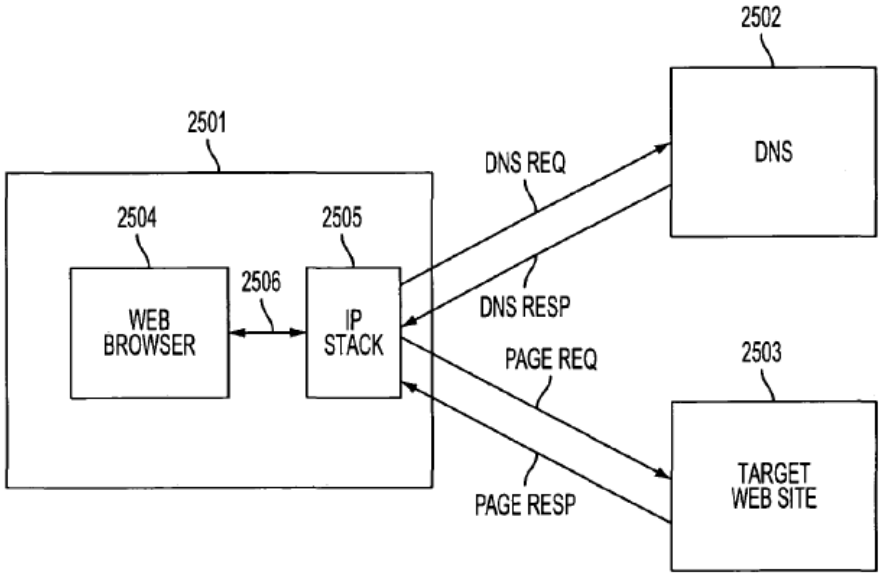


FIG. 25
(PRIOR ART)

The '151 patent describes the prior art system of Fig. 25 as including a conventional domain name server (DNS) that provides "a look-up function that returns the IP address of a requested computer or host. For example, when a computer user types in the web name "Yahoo.com," the user's web browser transmits a request to a DNS, which converts the name into a four-part IP address that is returned to the user's browser and then used by the browser to contact the destination web site. … When the user enters the name of a destination host, a request DNS REQ is made (through IP protocol stack 2505) to a DNS 2502 to look up the IP address associated with the name. The DNS returns the IP address DNS RESP to client application 2504, which is then able to use the IP address to

communicate with the host 2503 ….” (Ex. 1001,‘151 patent, column 36: line61-column 37, line 9)

B. “The Invention”

The ‘151 patent relates to a determining function that responds to a DNS request by performing either of two known functions - forwarding the DNS request to a DNS server or creating an encrypted channel for a secure server. The effective filing date for claims 1 and 13 of the ‘151 Patent is no earlier than the filing date of its parent, specifically February 15, 2000. Claim 1 reads as follows:

> 1. A data processing device, comprising memory storing a domain name server (DNS) proxy module that intercepts DNS requests sent by a client and, for each intercepted DNS request, performs the steps of: (i) determining whether the intercepted DNS request corresponds to a secure server; (ii) when the intercepted DNS request does not correspond to a secure server, forwarding the DNS request to a DNS function that returns an IP address of a nonsecure computer, and (iii) when the intercepted DNS request corresponds to a secure server, automatically initiating an encrypted channel between the client and the secure server.

In one embodiment, the proxy determines whether the DNS request is requesting access to a secure web site, and by extension, a secure server. (Ex.1001 at 37:60-38:11). When the DNS request corresponds to a secure web site/server, an encrypted channel, known as a virtual private network (VPN) is automatically initiated between the requesting computer and the secure server. (Ex.1001 at 37:62-38:2). When the DNS request does not correspond to a secure web

site/server, a look-up function is performed that returns the IP address of the non-secure web site/server. (Ex.1001 at 38:12-16).

The ’151 patent discloses various exemplary embodiments for implementing such automatic creation of an encrypted channel. The steps of intercepting the DNS request and determining if the request corresponds to a secure server may be performed by a DNS server. (Ex.1001 at 37:33-48). In another example, a DNS proxy may intercept the DNS requests and perform the determining. (Ex.1001 at 37:60-62). In various embodiments, the DNS proxy may reside on a different server than the DNS server (Ex.1001 at 38:32-34) or the DNS proxy and DNS server may be combined into a single server. (Ex.1001 at 38:30-32).

With respect to creating the encrypted channel (e.g., the VPN), the DNS server may set up the VPN between the requesting computer and the server. (Ex.1001 at 37:33-42). Alternatively, a DNS proxy may send a request to a gatekeeper to create the VPN between the requesting client computer and the server. (Ex.1001 at 39:10-19). The gatekeeper facilitates the allocation and exchange of information needed to communicate securely. (Ex.1001 at 38:24-27, 39:10-19). In any of these examples, the VPN is established without user involvement.

As with the DNS proxy and DNS server, the gatekeeper and DNS server may reside on different servers or be combined on a single server. (Ex.1001 at

38:22-24). By extension, any of the DNS proxy, DNS server, and gatekeeper may be on the same or different servers. With respect to the look-up function, the DNS proxy may send a DNS request to a DNS function in the same or different server, which performs the look-up to return an IP address. (Ex.1001 at 38:12-16). In some embodiments, the gatekeeper instructs the DNS proxy to send the DNS request to the DNS server. (Ex.1001 at 39:28-36).

V. CLAIM CONSTRUCTION

A. Legal Standards

The Board interprets a claim by applying its "broadest reasonable construction in light of the specification of the patent in which it appears." 37 C.F.R. §42.100 (b). Claim terms are given their ordinary and accustomed meaning as would be understood by one of ordinary skill in the art unless the inventor as a lexicographer has set forth a special meaning for a term. *Multiform Desiccants, Inc. v. Medzam, Ltd.*, 133 F.3d 1473, 1477 (Fed. Cir.1998); *York Prods., Inc. v. Central Tractor Farm & Family Ctr.*, 99 F.3d 1568,1572 (Fed. Cir. 1996). When an inventor acts as a lexicographer, the definition must be set forth with reasonable clarity, deliberateness, and precision. *RenishawPLC v. Marposs Societa per Azioni*, 158 F.3d 1243, 1249 (Fed. Cir. 1998)." *Garmin Int'l Inc. v.Cuozzo Speed Technologies, Inc.,* IPR2012-00001, Paper 15 (PTAB, Jan. 9, 2013). The specification has not given special meanings to any of the claim terms whether by

express definitions or otherwise. The bounds of a claim should be determined primarily by the claim language. "[I]t is the Patent Owner's burden to precisely define the invention in the claims." *AirCraft Medical LTD. v. Verathon Inc.*, Reexam. Control No. 95/000,161, Appeal 2012-007851, p. 16 (PTAB Dec. 11, 2012) (citing *In re Morris*, 127 F.3d 1048, 1056 (Fed. Cir. 1997)).

The Board should be leery of arguments of a party inconsistent with arguments presented in prior litigation. (*Cf.*, *Key Pharmaceuticals v. Hercon Laboratories Corp*., 161 F.3d 709, 715, 48 USPQ2d 1911, 1916 (Fed. Cir. 1998) ("Ordinarily, doctrines of estoppel, waiver, invited error, or the like would prohibit a party from asserting as "error" a position that it had advocated at the trial."). Given that the Board will apply the broadest reasonable construction, a patent owner such as VirnetX who has successfully argued in court for broad claim interpretations is estopped from advancing narrower constructions in these proceedings. *Cf.*, *Data General Corp. v. Johnson,* 78 F.3d 1556, 1565 (Fed.Cir.1996) ("[W]here a party successfully urges a particular position in a legal proceeding, it is estopped from taking a contrary position in a subsequent proceeding where its interests have changed."); 18 Susan Bandes & Lawrence B. Solum, Moore's Federal Practice § 134-30, at 134-63 (3d ed.1998). Instead, those prior statements should contribute to the determination of the proper meaning of the patent claims in this inter partes review. See 35 U.S.C. §301(a)(2),(d)

(statements of the patent owner filed in court taking a position on the scope of a patent claim may be used "to determine the proper meaning of a patent claim" in an inter partes review).

B. Proposed Claim Constructions

Ex.1009 ¶ 21 sets forth a proposed definition of a person of ordinary skill in the art. Petitioner proposes the following constructions:

Domain name	A name corresponding to an IP address or a group of IP addresses.
DNS or domain name service	A lookup service that returns an IP address for a requested domain name
Domain name server (DNS) proxy module	A program that responds to a domain name inquiry in place of a DNS
Domain name server (DNS) module	No construction; alternatively, a program that performs a lookup service and returns an IP address for a requested domain name
DNS request	A communication that contains a domain name and requests an IP address for the domain name
Client	A computer or program from which a DNS request is generated
Intercepts DNS requests	Receives DNS requests ahead of a DNS function
Determining	Plain and ordinary meaning – can be performed at any place in the system
Forwarding the DNS request to a DNS function	Passing a domain name sent by the client to a DNS function
Automatically initiating an encrypted channel/ Automatically creating a secure channel	Initiating/creating the channel without involvement of a user
Secure server	A server that requires authorization for access and that can communicate in an encrypted channel

A "***domain name***" means "a name corresponding to an IP address or a group of IP addresses." (Adopted in Ex.1011 at 12-13; adopted in Ex.1016 at 16. The specification describes "domain name" servers as providing a look-up function that returns "the IP address" of a requested computer or host. (Ex.1001 at 36:61-63). The claims also make clear that a "domain name" returns an IP address. (Ex.1001 at 46:61-63). With regard to parent U.S. Patent No. 6,502,135, in *Virnetx v. Microsoft Corporation*, Case No. 6:07 CV 80, Virnetx successfully argued that "domain name" was not limited to "*a hierarchical name for a computer under traditional DNS format*" but instead proposed, and the Court adopted, that a "domain name" is "a name corresponding to an IP address." (Ex.1011 at 12-13). Also, in *VirnetX Inc. v. Cisco Systems, et al.*, Case No. 6:10cv417, the same Court again adopted Virnetx's proposed definition of "domain name" as "a name corresponding to an IP address." (Ex.1016 at 16). Having succeeded in achieving a broad construction of "domain name" in court, VirnetX is estopped from arguing here that the broadest reasonable construction of "domain name" is any narrower than "a name corresponding to an IP address."

A "***DNS***" stands for "***domain name service***" which is "a lookup service that returns an IP address for a requested domain name." The specification states: 'Conventional Domain Name Servers (DNSs) provide a look-up function that

returns the IP address of a requested computer or host.' (Ex.1001 at 36:61-63). With regard to parent U.S. Patent No. 6,502,135, in VirnetX v. Microsoft Corporation, Case No. 6:07 CV 80, VirnetX successfully argued that "domain name service" is not limited to "the conventional lookup service defined by the Internet Engineering Task Force (IETF) that returns the IP address of a requested computer or host" but is merely "a look-up service that returns an IP address for a requested domain name." (Ex.1011 at 11-12). Also, in *VirnetX Inc. v. Cisco Systems, et al.,* Case No. 6:10cv417 the same Court again adopted Virnetx's definition of "domain name service" as "a look-up service that returns an IP address for a requested domain name." (Ex.1016 at 14-15). Having succeeded in achieving a broad construction of "domain name service" in court, VirnetX is estopped from arguing here that the broadest reasonable construction of "domain name service" is any narrower than "a look-up service that returns an IP address for a requested domain name."

A ***"domain name server (DNS) proxy module***" means "a program that responds to a domain name inquiry in place of a DNS." (Proposed construction of a DNS proxy module by VirnetX in Ex.1017 at 16; adopted by Ex.1016 at 16-17). The '151 Patent makes clear that functions of the DNS proxy can be combined on the same server with those of the DNS or can operate independently of the DNS server (see Ex.1001 at 38:30-34). Under the broadest reasonable interpretation of

the claims, the DNS proxy can be a function in the same computer as the client computer. VirnetX took the position in the Microsoft case (and the court agreed) that a "DNS proxy server" can be a "computer or program" and that "the DNS proxy server does not have to be separate from the client computer." (Ex.1011 at 24). Similarly, in the Apple case, VirnetX continued to assert that the DNS proxy server can be a software module in the same computer as the client computer. For example, VirnetX admits that the DNS request generated (and transmitted) from the client computer can be "from one software module on the computer to another software module on the device that makes the determination." (Ex.1012 at 209). Furthermore, one of the inventors (Dr. Short) testified that "we came to realize fairly early on that what you really want to do is put this DNS proxy software in your computer so that way every computer that's enabled to do this has its own proxy, and you don't have to have all these servers out there." (Ex.1013 at 92-93). Also, VirnetX's expert, Dr. Jones, supported this construction by testifying that "The DNS request is generated in the application, for example, Safari and is passed through an API call to iOS." (Ex.1014 at 47-48; see also Ex.1014 at 67-68). The '151 Patent makes clear that the DNS proxy server "handles requests for DNS look-ups" (Ex.1001 at 38:35-37) and if access to a secure host was not requested "the DNS request is passed to conventional DNS server 2609, which looks up the IP address of the target site and returns it to the user's application for further

processing" (Ex.1001 at 38:39-43). Thus, "responding to a DNS inquiry in place of a DNS" can involve a lookup to a domain name service to obtain an IP address.

The term "***domain name server (DNS) module***" should not be construed as a limitation of claim 13 because it is a preamble term used for nothing more than to provide a name to the computer readable instructions. It does not provide life and meaning to any of the elements of the claim. The Federal Circuit has held that, in general, "a preamble is not limiting where a patentee defines a structurally complete invention in the claim body . . ." *Catalina Marketing Int'l., Inc. v. Coolsavings.com, Inc.*, 298 F.3d 801, 808 (Fed. Cir. 2002) (citations omitted). Moreover, it is well-settled that a term used in the preamble will not limit the claim if it "merely gives a descriptive name to the set of limitations in the body of the claim that completely set forth the invention." *IMS Tech., Inc. v. Haas Automation, Inc.,* 206 F.3d 1422, 1434 (Fed. Cir. 2000); *see Storage Tech. Corp. v. Cisco Sys., Inc.*, 329 F.3d 823, 831 (Fed. Cir. 2003) (language in preamble serving as "a convenient label for the invention as a whole" does not limit the claims).

However, should the term "***domain name server (DNS) module***" be construed as a limitation of claim 13, it would mean "a program that performs a lookup service and returns an IP address for a requested domain name" (i.e., a program that provides a "domain name service").

A "***DNS request***" is "a communication that contains a domain name and requests an IP address for the domain name." The DNS request is not limited to any particular protocol and can be internal to a computer from one software module to another software module. In litigation against Apple, VirnetX succeeded in proving infringement by arguing that a DNS proxy server can be a software module in the same computer as the client computer and therefore that the DNS request generated (and transmitted) from the client computer can be "from one software module on the computer to another software module on the device that makes the determination." (see Ex.1012 at 209). By at least 1997, it was well-known for a client function of a client computer (e.g., an application such as a web browser) to request domain name resolution from a resolver function in the client computer. For example, client applications running on Windows and Unix operating systems made function calls to the operating system (specifically, the "gethostbyname" function) in order to obtain an IP address for a given hostname (see Ex.1004 – "From an application's point of view, access to the DNS is through a resolver …. The [gethostbyname(3) library function] takes a hostname and returns an IP address…The resolver contacts one or more name servers to do the mapping"). Ex.1005 at 112 describes error return codes from gethostbyname() and gethostbyaddr() library functions "when using the resolver." Ex.1006 describes the gethostbyname eCos system library function. Ex.1007 at 2:63-3:8 states that

"When a user program, such as the browser, requests information ... a resolution request is passed in the form of a query to the resolver." Ex.1008 at 9:49-54 states that "Access to the DNS is through a resolver and software library functions. The function in this case takes a domain name or host name and returns an IP address." Thus, under the broadest reasonable interpretation, a DNS request can include the domain name passed from one software module to another software module, e.g., through an internal message or function call.

A ***"client"*** is "a computer or program from which a DNS request is generated." The claims define the "client" as the entity that sends the DNS request. (Ex.1001 at 46:57). VirnetX has admitted that the client can be an application that generates a DNS request, e.g., to an operating system (Ex.1014 at 47-48; see also Ex.1014 at 67-68).

"***Intercepts DNS requests***" means "receives DNS requests ahead of a DNS function." A DNS request contains a domain name and requests an IP address for the domain name. The IP address will be retrieved from a DNS function. Given the ordinary meaning of "intercept," to intercept a DNS request is to receive it ahead of a DNS function. This is consistent with usage of the term in the specification. With respect to Fig. 26 of the '151 patent, the patent states, "DNS proxy 2610 intercepts all DNS lookup functions from client 2605." (Ex. 1001, 37:60-61) As shown, the DNS proxy receives the DNS request and then if access to a non-secure

site is requested, the request is passed to the DNS function identified as DNS server 2609. (*Id.*, 38:12-16) The '151 patent further describes listening on the Internet. Receiving and viewing the contents of a DNS request (DNS REQ) ahead of it reaching a destination DNS function, is described as intercepting the DNS request. (*Id.*, 37:11-14).

"Determining" should be given its plain and ordinary meaning of making a determination. The claim does not specify where this step is performed, and the broadest reasonable interpretation covers implementation of the "determining" at any place in the system. (See Ex.1015 at 5 – 'While the Court has not construed the word "determining," in its claim construction opinion, the Court noted this determining step could be performed by the client computer or by the target computer.'). The '151 Patent makes clear that "determining" can be performed in in a DNS proxy (Ex.1001 at 37:60-62), which can reside in a DNS server or in a server separate from the DNS server (Ex.1001 at 38:30-35) or can reside in a client computer as discussed above.

The term "***forwarding the DNS request to a DNS function***" means "passing a domain name sent by the client to a DNS function." The domain name may be received in one format and transmitted to the DNS server using another format. As explained in the discussion of the construction of domain name server (DNS) proxy module, a DNS request may be first sent as a domain name in a client

application before it is sent to a DNS function in a DNS protocol format. Thus, the DNS request received by the DNS proxy server can be a domain name received via an internal message or function call from an internal client. The domain name is forwarded in a DNS protocol format. Nevertheless, common to the formats of these DNS requests is the domain name that is the subject of the requests. Hence, forwarding the DNS request means passing a domain name sent by the client.

"***Automatically initiating an encrypted channel***" means "initiating the encrypted channel without involvement of a user." Similarly, "***automatically creating a secure channel***" means "creating the secure channel without involvement of a user." In the specification of the '151 patent, after the user or user's computer makes the initial DNS request, the user is not further involved in setting up the encrypted channel. In one embodiment, if a user has sufficient security privileges, a secure channel is established between the user's computer and the secure target "preferably performed transparently to the user (i.e. the user need not be involved in creating the secure link)." (Ex. 1001,'151 patent, 38:59-66). Thus, the specification describes that the channel is initiated and created without further user action.

In *VirnetX v. Microsoft Corporation*, Case No. 6:07 CV 80 (E.D. Tex. 2007), the court construed "automatically initiating the VPN" in connection with U.S. Patent 6,502,135, the parent of the '151 patent. The court rejected Microsoft's

proposed construction and ruled in favor of VirnetX interpreting the clause to mean "initiating the VPN without involvement of a user." (Ex.1011 at 21-22). Having achieved a broad construction of "automatically initiating" in court, VirnetX is estopped from arguing for any narrower construction than "initiating/creating the channel without involvement of a user."

A "***secure server***" is "a server that requires authorization for access and that can communicate in an encrypted channel." The secure server is described in preferred embodiments labeled "Scenario #1" and "Scenario #2" (Ex. 1001, 39:10-27). In these embodiments, a client computer requests to access the target computer, and if the client has authorization to access the target computer, a gatekeeper computer establishes a VPN. *Id.,*39:10-19. If the client computer does not have authorization to access the target computer, the DNS proxy returns a "host unknown" message. *Id.,* 39:20-27. Further, Virnetx proposed this construction of "secure server" in the Cisco litigation and the Court adopted the same. (Ex. 1016, 6:10-cv-417, pgs. 23-24) The court relied upon Ex.1001 at 37:33-49 as demonstrating that "secure" means requiring authorization for access.

VI. FULL STATEMENT OF THE REASONS FOR CANCELLATION OF CLAIMS

A. Claims 1 and 13 are Unpatentable Over Kiuchi

Kiuchi was presented at the 1996 Symposium on Network and Distributed Systems Security (SNDSS), and published by IEEE in the Proceedings of SNDSS

1996. Mr. Russell Housley, who was also among the speakers at the Symposium, confirms that the Kiuchi paper was presented to the Symposium on Network and Distributed Systems Security (SNDSS) in 1996, and the paper was published in the symposium proceedings, distributed to the participants, and made available to the public. (Ex.1009 ¶ 28). Thus, Kiuchi is prior art pursuant to 35 U.S.C. §102(b).

As set forth in Ex.1009 ¶¶ 29-43, Kiuchi was concerned with establishing secure network links between different hosts on the Internet. In particular, the service was contemplated for use by medical institutions for private information, although Kiuchi makes clear that it can be used in other areas (Ex.1002 at 69, paragraph 5). The following Figure 1 shows the relevant components of Kiuchi's system:

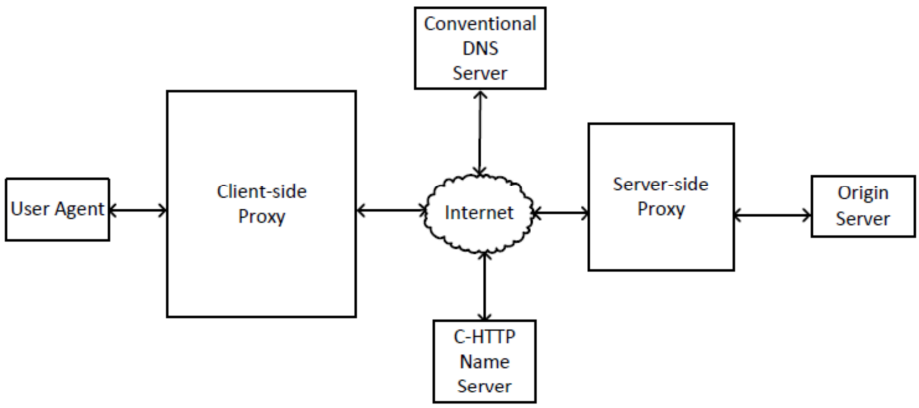


Figure 1 – Kiuchi Schematic Block Diagram

Kiuchi creates a closed network over the Internet (referred to as the "C-HTTP" system) that allows a user agent computer to access private web pages (HTML documents) stored on an origin server (i.e., a "secure target web site") in

the closed network. The user agent and origin server are members of the closed network constructed over the Internet using a client-side proxy that performs proxy functions for the user agent and a server-side proxy that performs proxy functions for the origin server. The client-side proxy and server-side proxy are installed in firewall devices situated between the user agent and the origin server, which are unaware of these proxies. (see Ex.1002 at 64, sec. 2.1). The user agent and the origin server are conventional HTTP/1.0 compatible devices, e.g., "[c]ommunications between two kinds of proxies and *HTTP/1.0 compatible servers/user agents within the firewalls* are performed based on HTTP/1.0." (Ex.1002 at 64, sec. 2.1, emphasis added).

The client-side proxy and server-side proxy work in conjunction with a C-HTTP name server over the Internet. (Ex.1002 at 64). For permitted secure communications, the C-HTTP name server is the server that responds to name service requests by looking up domain names and returning their IP address and related VPN resources, i.e., public key and Nonce values (Ex.1002 at 65, sec. 2.3(2)). Each proxy is registered with the C-HTTP name server, including a hostname, IP address, and public key for the proxy. (Ex.1002 at 65, sec. 2.2). The hostname (domain name) of the server-side proxy is used to access web pages at an origin server being proxied by the server-side proxy. For non-secure connections, a conventional DNS name service is used to return IP addresses. *Id.* Thus, domain

name services are provided by the C-HTTP name server for secure communication requests and by a conventional DNS for non-secure communication requests.

Petitioner presents the following Figure 2 to schematically show relevant functions performed in Kiuchi's system, where the dashed arrows represent communications that occur at least in part over the Internet:

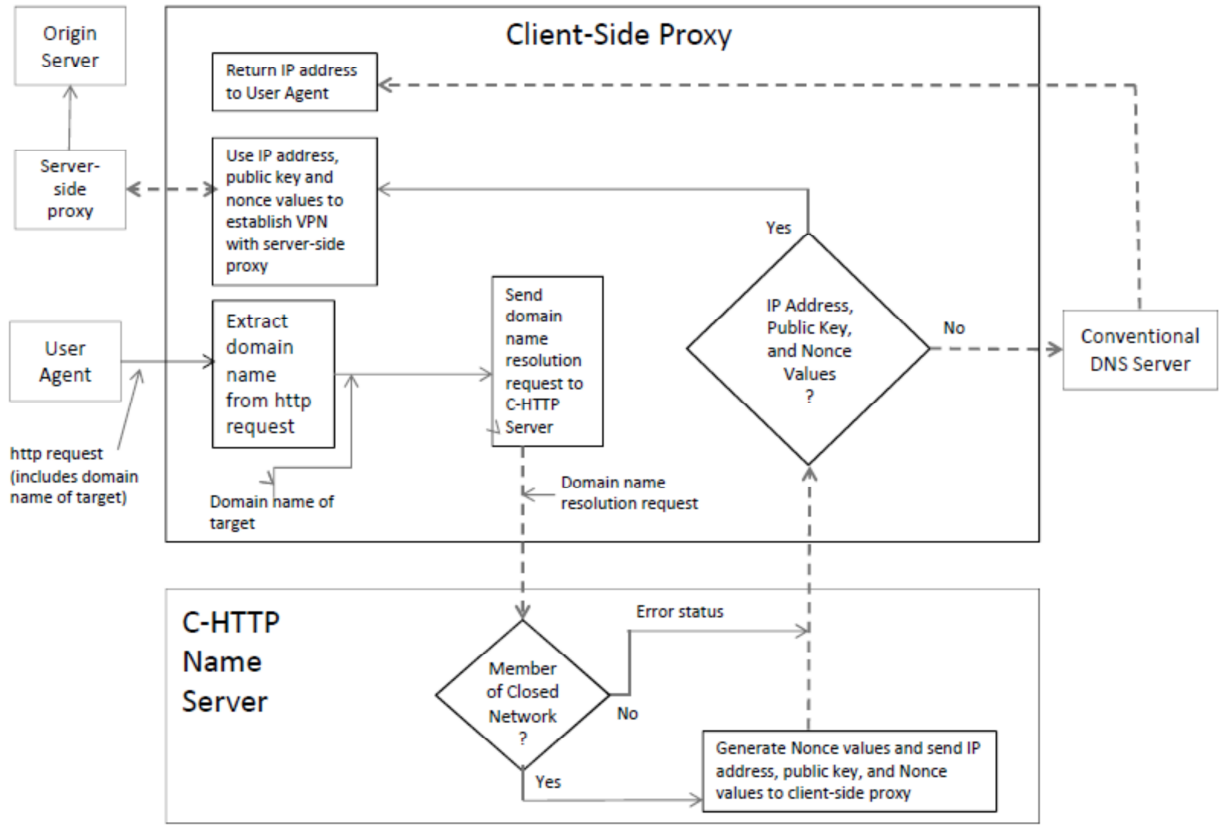


Figure 2 - Functional Diagram of Kiuchi

The client-side proxy receives, from the user agent, an HTTP request specifying a web page (HTML document) stored at the origin server and associated with a given URL. The URL in the HTTP request has the format "http://<hostname>/<web page>[connection ID]" (see the sample URL at Ex.1002

at 65, sec. 2.3(1), where "server.in.current.connection" is the hostname (i.e., "domain name") of the server-side proxy, "sample.html" is a web page stored on the origin server being proxied by the server-side proxy, and "6zdDfldfcZLj8V!i" is an optional connection ID).

In order to create a secure, encrypted channel, the client-side proxy first determines whether the HTTP request is directed to a secure site in the closed network. Specifically, the client-side proxy "asks the C-HTTP name server whether it can communicate with the host specified in [the] URL" (Ex.1002 at 65, sec. 2.3(2)), specifically by "[taking] off the connection ID and [forwarding] the stripped, the original resource name to the server in its request" to the name server. (Ex.1002 at 65, sec. 2.3(1)). Thus, the client-side proxy extracts the domain name from the HTTP request and sends the requested domain name to the C-HTTP name server to request the IP address of the host. The name server returns an IP address only if a secure connection with that host (i.e., server-side proxy) is permitted. (Ex.1002 at 65, sec. 2.3(2)). Along with the IP address, the C-HTTP name server also returns the public key of the server-side proxy and Nonce values. *Id.*

The client-side proxy uses the public key and Nonce values to create a secure, encrypted communication channel with the server-side proxy (Ex.1002 at 65, sec. 2.3(3)). When the server-side proxy receives the client-side proxy's IP address, hostname and public key, it authenticates the values and generates a

connection ID as well as a second key for response encryption (Ex.1002 at 65-6, sec. 2.3(4)). When these are accepted and checked by the client-side proxy, the secure, encrypted communication channel is established (Ex.1002 at 66, sec. 2.3(5)). Security between the proxies is made possible by the public key and Nonce values provided by the C-HTTP name server. Once the secure/encrypted communication channel is established, HTTP/1.0 messages can then be exchanged between the user agent and the origin server over the secure/encrypted communication channel via the proxies. (Ex.1002 at 66, ¶¶ (7)-(8)).

If a secure connection with the requested host is not permitted, the name server instead returns an error status to the client-side proxy. (Ex.1002 at 65, sec. 2.3(2)). The client-side proxy then acts exactly like "an ordinary HTTP/1.0 proxy"; it sends a standard domain name service lookup request asking for the corresponding IP address from a conventional public DNS server. *Id.* Once the IP address is obtained, a typical non-secure communication may take place.

In Kiuichi, the request sent by the client-side proxy to the C-HTTP name server meets the "DNS request" limitations as recited in the claims because the request is a communication that contains a domain name and requests an IP address for the domain name. If the domain name included in the request corresponds to a host that is a member of the closed network (i.e., a server-side proxy that is a "secure server"), the C-HTTP name server similarly provides a

"domain name service" because it provides a lookup service that returns an IP address for the requested domain name. To the extent that Kiuchi makes statements that "DNS" is not used in the C-HTTP system (e.g., "In a C-HTTP-based network, instead of DNS, a C-HTTP-based secure, encrypted name and certification service is used" and "The DNS name service is not used for hostname resolution as the original secure name service, including certification, is used for the C-HTTP-based network" – Ex.1002 at 64)), Kiuchi is simply explaining that a C-HTTP name service, as opposed to a conventional DNS (i.e., as defined by the Internet Engineering Task Force (IETF)), is used for resolving IP addresses in the closed network. Since the broadest reasonable interpretation of "domain name service" is not limited to the conventional lookup service as defined by the IETF, these statements in Kiuchi do nothing to diminish the fact that Kiuchi's C-HTTP name server is a "domain name service" within the meaning of the claims.

As explained above, the broadest reasonable constructions of functions such as the client function and DNS proxy function encompass implementation of such functions in software modules. Such software modules can reside on separate machines or be combined in ways where various functions reside on the same machine. For example, the '151 Patent makes clear that a DNS proxy and a DNS server can be combined into a single server or can operate in separate servers. (Ex.1001 at 38:30-33). This is the nature of software, where developers generally

have great leeway in how to divide functions into software modules and where to place the software modules. Dr. Short (an inventor of the '151 Patent) testified to this very point at trial, when asked: "Does this mean in order to practice your invention, we would have to buy servers to act as the DNS proxy and put those out in various places on the Internet?" His answer was: "*No, not at all. ... [W]hat you really want to do is put this DNS proxy software in your computer so that way every computer that's enabled to do this has its own proxy.*" (see Ex.1013 at 92-93). Kiuchi also teaches to an ordinarily skilled artisan that the functions of a DNS proxy can be included in the same machine as the client computer (i.e., the client-side proxy machine) or in a DNS server such as the C-HTTP name server.

B. Request 1 – Claims 1 and 13 are Obvious over Kiuchi

For purposes of the following analysis, domain name resolution functions that in Kiuchi's described embodiments are split between the client-side proxy and the C-HTTP name server are consolidated into an appropriately modified C-HTTP name server such that the modified C-HTTP name server includes a "DNS proxy module" or "DNS module" pursuant to the claims, as discussed in detail below. In response to receiving an HTTP request from the user agent, the client-side proxy (i.e., the "client") sends a DNS request to the C-HTTP name server. The DNS request, which contains the domain name from the HTTP request (i.e., the hostname given to the server-side proxy), requests the IP address of the server-side

proxy (i.e., the "secure server"). The DNS proxy module or DNS module of the modified C-HTTP name server determines whether the requested host is a member of the closed network. If the requested host is a member of the closed network (i.e., a server-side proxy), then a C-HTTP name service response is returned to the client-side proxy including an IP address for the server-side proxy and Nonce values, and a secure, encrypted channel is automatically initiated/created between the client-side proxy and the server-side proxy. However, if the requested host is not a member of the closed network, then the DNS proxy module or DNS module performs a lookup to the conventional DNS server and returns the IP address to the client-side proxy.

As set forth in Ex.1009 ¶¶ 44-47 and Appendix C, it would have been apparent to a person of ordinary skill in the art as a mere design choice to consolidate domain name resolution functions in Kiuchi's C-HTTP name server. Kiuchi clearly recognizes and discloses that a conventional DNS lookup for the domain name is needed when access is not being requested to a secure server, i.e., when the requested server-side proxy is not registered in the closed network. (Ex.1002 at 65, sec. 2.3(2)). This is identical to the '151 Patent, where a DNS lookup is performed when access is not being requested to a secure server. (Ex.1001 at 38:12-16).

Kiuchi defines three new components for the system, namely the client-side proxy, the server-side proxy, and the C-HTTP name server. (Ex.1002 at 64, sec. 2.1). While Kiuchi describes a system in which a conventional DNS lookup request is made from the client-side proxy, it would have been apparent to a person of ordinary skill in the art based on Kiuchi's teachings to make the conventional DNS lookup request from the C-HTTP name server. As discussed above, the C-HTTP name server already determines whether the DNS request received from the client-side proxy is requesting access to a secure server (i.e., a server-side proxy). Rather than returning an error status to the client-side proxy when the DNS request is not requesting access to a secure server, it would have been trivial and obvious as a mere design choice for the C-HTTP name server to pass the domain name received in the C-HTTP name service request to the conventional DNS server (i.e., a "DNS function"), as depicted in the following Figure:

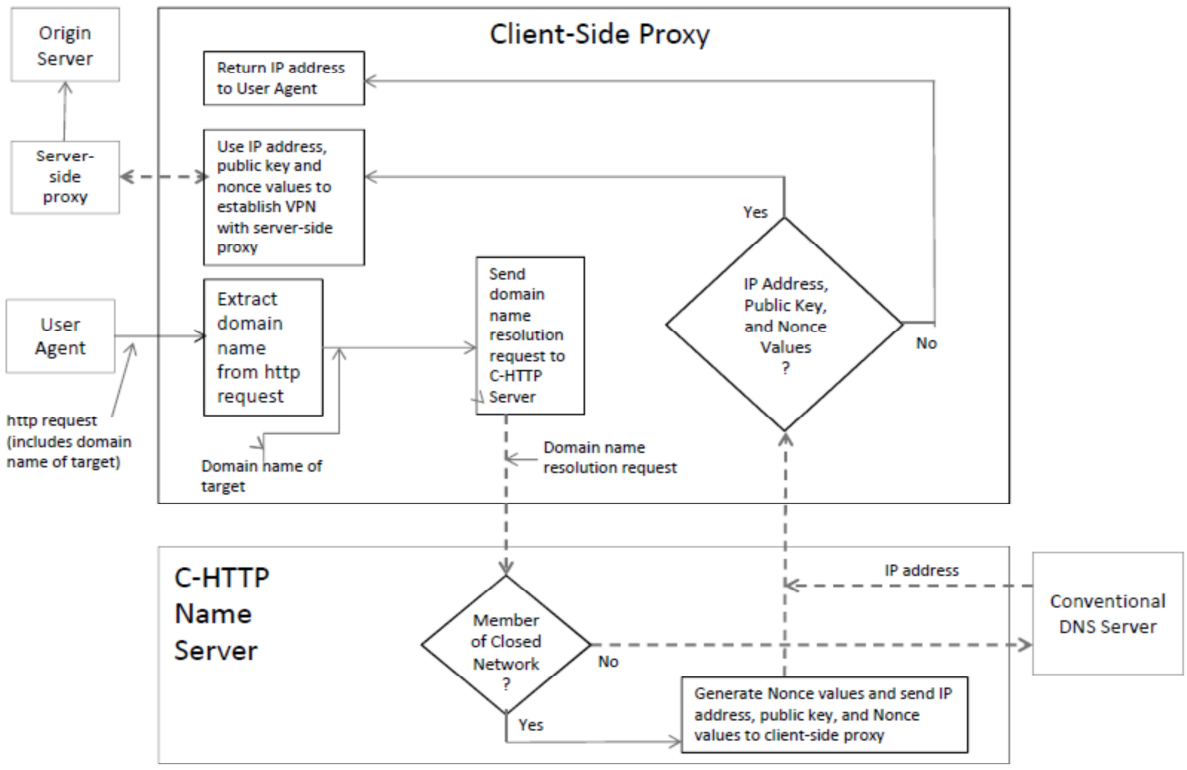


Figure 3 – Alternate Functional Diagram of Kiuchi

Such a configuration, which places a DNS proxy server function in a modified C-HTTP name server (similar to placement of the DNS proxy server function of the '151 patent in the DNS server – see Ex.1001 at FIG. 26), is merely a rearrangement of existing functions within the C-HTTP system and could be implemented with little or no modification to Kiuchi's protocols. For example, a C-HTTP name service response message containing an IP address without a public key and Nonce values (e.g., using values of zero or other convention for the public key and Nonce fields, or modifying the protocol to use a previously unused flag in the response to indicate that a public key and Nonce values are not provided) would indicate to the client-side proxy that the DNS request is not requesting

access to a secure server and hence that no secure/encrypted channel is needed. The motivation for modifying Kiuchi in this way would have been to streamline the operation of the system, e.g., instead of having the C-HTTP name server send an error status to the client-proxy which would in turn initiate a conventional DNS inquiry, the modification eliminates the error status message from the process by having the C-HTTP name server directly initiate the request to the conventional DNS server. Thus, the consolidation of domain name resolution functions in the C-HTTP name server is obvious in view of Kiuchi. See *KSR Intern. Co. v. Teleflex Inc.*, 550 U.S. 398, 401; 127 S.Ct. 1727, 173 (2007) ("a combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results"); *Sakraida v. Ag Pro, Inc*., 425 U.S. 273, 282, 96 S.Ct. 1532 (1976) (when a patent "simply arranges old elements with each performing the same function it had been known to perform" and yields no more than one would expect from such an arrangement, the combination is obvious).

The modified C-HTTP name server includes a "DNS proxy module" because it contains a program that responds to a domain name inquiry in place of a DNS (i.e., it responds to a DNS request in place of a conventional DNS server when the requested host is a member of the closed network). The modified C-HTTP name server also includes a "DNS module" because it includes a program that performs a lookup service and returns an IP address for a requested domain

name (i.e., it first performs a lookup to determine if the requested host is registered in the closed network, and then, if needed, performs a lookup to the conventional DNS server); in Kiuchi, this DNS module also performs the functions of a DNS proxy by responding to a DNS request in place of the conventional DNS server when the requested host is a member of the closed network, as discussed above.

The C-HTTP is a data processing device having a memory storing a "DNS proxy module" or "DNS module" that intercepts DNS requests sent by the client-side proxy (i.e., the "client") and performs the steps recited in the claims. The client-side proxy is a "client" because it is a computer that sends a DNS request. The server-side proxy is a "secure server" because it is a server that requires authorization for access and can communicate in an encrypted channel.

Ground 1. Claim 1 is Obvious Over Kiuchi

As set forth in Ex.1009 ¶¶ 29-47 and Appendix C, Kiuchi teaches all of the limitations of claim 1. With regard to claim 1, the modified C-HTTP name server including a DNS proxy module is a data processing device. The functions performed in the modified C-HTTP name server are necessarily stored in computer program code in memory, as is the case for any such processing device. (Ex.1009 ¶ 44).

When the client-side proxy (i.e., the "client") receives the HTTP request from the user agent, it generates (and transmits) a DNS request that is sent to the

C-HTTP name server in the form of a C-HTTP name service request to ask the C-HTTP name server whether it can communicate with the specified host. (Ex.1002 at 65, sec. 2.3(2)). The C-HTTP name service request is a "DNS request" because it is a communication that contains a domain name (i.e., the hostname from the URL in the HTTP request) and requests an IP address for the domain name. If the domain name sent in the DNS request is the hostname given to the server-side proxy, then the DNS request corresponds to a secure server (i.e., the server-side proxy).

The DNS proxy module of the modified C-HTTP name server receives DNS requests sent by the client-side proxy (i.e., the "client"). Receipt of DNS requests by the DNS proxy module from the client-side proxy constitutes intercepting DNS requests because the DNS proxy module receives such DNS requests ahead of a DNS function (e.g., ahead of the conventional DNS server). For each intercepted DNS request, the DNS proxy module in the modified C-HTTP name server performs the steps recited in the claims, as follows:

Step (1) of Claim 1 - Upon receiving the C-HTTP name service request (i.e., DNS request) from the client-side proxy, the DNS proxy module of the modified C-HTTP name server "examines whether the requested server-side proxy is registered in the closed network." (Ex.1002 at 65, sec. 2.3(2)). The DNS proxy module determines whether the intercepted DNS request sent by the client-side

proxy corresponds to a server-side proxy in the closed network (i.e., a "secure server") based on whether the requested server-side proxy is registered in the closed network. The DNS proxy module determines that the DNS request corresponds to a secure server, if the requested server-side proxy is registered in the closed network. The DNS proxy module determines that the DNS request does not correspond to a secure server, if the requested server-side proxy is not registered in the closed network. (Ex.1002 at 65, sec. 2.3(2)-(3)). Thus, step (1) of claim 1 is satisfied by the determination made by the DNS proxy module of the modified C-HTTP name server.

Step (2) of Claim 1 – As discussed above for step (1), the DNS proxy module determines that the DNS request does not correspond to a secure server, if the requested server-side proxy is not registered in the closed network. When the DNS proxy module makes this determination, it performs a DNS lookup to the conventional DNS server (i.e., a DNS function that returns an IP address of a nonsecure computer), which involves forwarding the domain name (DNS request) to the conventional DNS server and receiving an IP address back from the DNS server. (Ex.1009 ¶¶ 44-47). Specifically, Kiuchi teaches that "If a client-side proxy receives an error status, then it performs DNS lookup, behaving like an ordinary HTTP/1.0 proxy." (Ex.1002 at 65, section 2.3, paragraph 2). In the modified C-HTTP server, this DNS lookup function resides in the C-HTTP name

server rather than in the client-side proxy. Thus, the modified C-HTTP server including the “DNS proxy module” or “DNS module” performs a DNS lookup if the requested server-side proxy is not registered in the closed network and hence performs a DNS lookup when the DNS request does not correspond to a secure server. A person of ordinary skill would have understood that Kiuchi's reference to performing a "DNS lookup ... like an ordinary HTTP/1.0 proxy" involves forwarding the DNS request to the conventional DNS server and receiving an IP address from the DNS server. For example, RFC 1123 (Ex.1010) defines how computers on the Internet should operate and states that when using domain names, *"Host domain names MUST be translated to IP addresses* as described in Section 6.1." (Ex.1010 at 13 (emphasis added).) Section 6.1, in turn, states that *"Every host MUST implement a resolver for the Domain Name System* (DNS), and it MUST implement a mechanism using this DNS resolver to convert host names to IP addresses and vice-versa." *(Id.* at 72.). Thus, in response to determining that the DNS request does not correspond to a secure server, the modified C-HTTP name server with “DNS proxy module” or “DNS module” forwards the DNS request to the conventional DNS server.

Step (3) of Claim 1 - As discussed above for step (1), the DNS proxy module determines that the DNS request corresponds to a secure server, if the requested server-side proxy is registered in the closed network. When the DNS

proxy module makes this determination, it sends a C-HTTP name service response to the client-side proxy containing the IP address and public key of the server-side proxy (i.e., the "secure server") as well as request and response Nonce values. In turn, the client-side proxy "sends a request for connection to the server-side proxy," which is encrypted using the server-side proxy's public key and contains the client-side proxy's IP address, hostname, request Nonce value and symmetric data exchange key for request encryption." (Ex.1002 at 65, right column, section 2.3(3)). The sending of the C-HTTP name service response by the DNS proxy module to the client-side proxy constitutes "automatically initiating" a secure, encrypted channel within the context of the claim because the C-HTTP name service response causes the client-side proxy to send a request for connection to the server-side proxy. (Ex.1002 at 65, sec. 2.3(3)). In this regard, it should be noted that the '151 Patent provides, as one example of automatically initiating/creating a secure, encrypted channel, transmission of a message requesting that a VPN be created (see Ex.1001 at 37:66-38:2). The C-HTTP name service response is analogous because it is a message that causes the secure, encrypted channel to be created. Therefore, step (3) of claim 1 is satisfied by sending of the C-HTTP name service response message to the client-side proxy in response to determining that the DNS request corresponds to a secure server.

Therefore, steps (1)-(3) of claim 1 are satisfied by the DNS proxy module or DNS module in the modified C-HTTP name server.

Ground 2. Claim 13 is Obvious Over Kiuchi

Claim 13 is virtually identical to claim 1. Instead of being directed to the data processing device, claim 13 is a Beauregard claim directed to the computer readable medium contained within the data processing device and containing the programs for directing the steps. These program steps are contained in the program code for a C-HTTP name server accordingly modified.

As discussed above, modified C-HTTP name server includes a "DNS module" for purposes of claim 13 because it is a program that performs a lookup service and returns an IP address for a requested domain name (i.e., it first determines whether the requested host is a member of the closed network, and then, if needed, performs a lookup to the conventional DNS server); in Kiuchi, this DNS module also performs the functions of a DNS proxy by responding to a DNS request in place of the conventional DNS server when the requested host is a member of the closed network, as discussed above.

The main difference in the steps as claimed is that claim 13 recites "automatically creating a secure channel between the client and the secure server." The use of public key and Nonce values create an encrypted channel which is therefore secure since those VPN resources are only made available to the client-

side proxy (i.e., the "client") and the server-side proxy (i.e., the "secure server"). There is no user involvement in the creation of the secure channel. It is automatic. Just as the '151 patent describes "Use of a DNS Proxy to Transparently Create Virtual Private Networks" by sending a message to a gatekeeper (Ex.1001 at 37:66-38:2), so too does Kiuchi disclose creating a secure, encrypted channel when the C-HTTP name server sends out public key and Nonce values in response to a DNS request. Thus, for the reasons set out above with respect to claim 1 and in view of the claim charts, claim 13 should be rejected for obviousness over Kiuchi.

C. Request 2 – Claims 1 and 13 are Anticipated by Kiuchi

A careful consideration of the inner workings of the client-side proxy reveals that Kiuchi's client-side proxy performs a "resolver" function that receives a domain name resolution request from an internal client (in this case, the domain name extracted from the received HTTP request) and returns an IP address for the domain name. (Ex.1009 ¶ 49). Petitioner presents the following Figure 4 to schematically show the resolver and client functions in the client-side proxy:

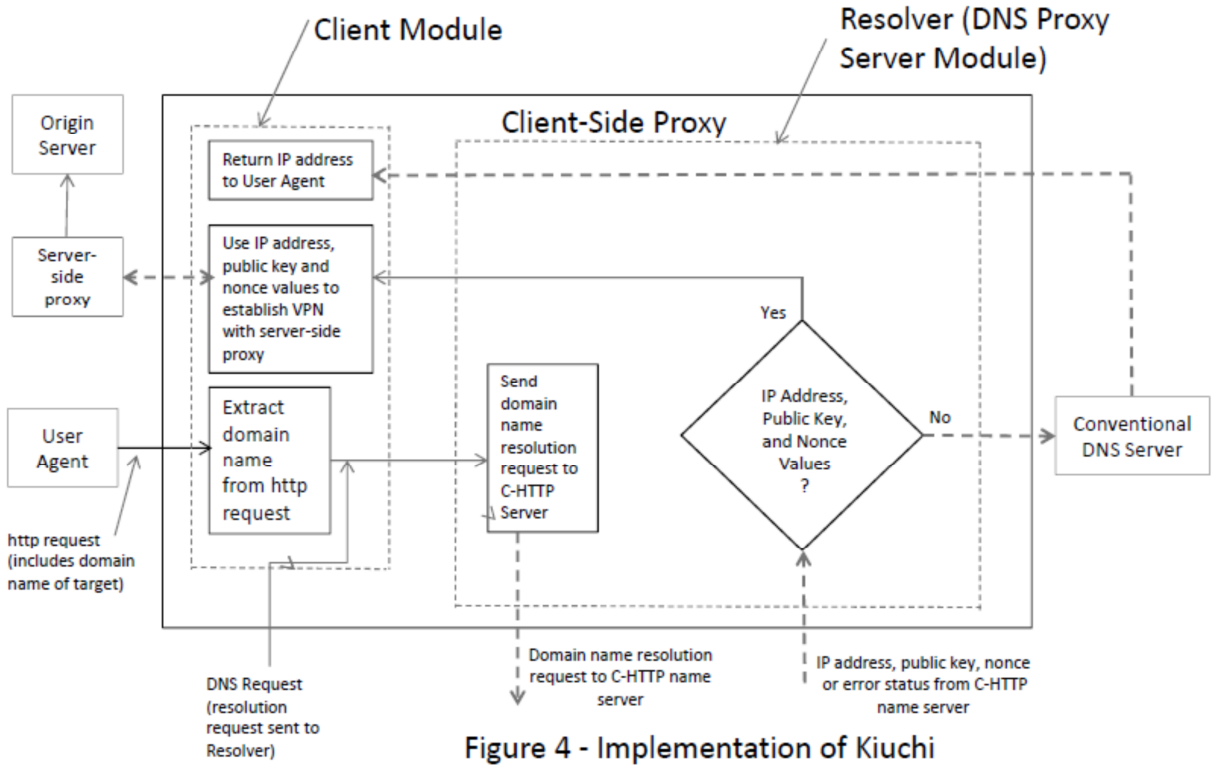


Figure 4 - Implementation of Kiuchi

As discussed above with reference to construction of "Domain Name Service (DNS) request," resolver functions were known in the art well before the '151 Patent was filed. (Ex.1009 ¶ 50). Thus, in Kiuchi, an internal resolution request (which is a "DNS request" because it is a communication that contains a domain name and requests an IP address for the domain name) is made to the resolver function (which is a collection of software functions within the client-side proxy), which acts as a DNS proxy server to contact the C-HTTP name server and optionally also the conventional DNS server to obtain an IP address for the domain name and return the IP address to the internal client. (Ex.1009 ¶ 51).

Furthermore, a careful consideration of the inner workings of the client-side proxy also reveals that the resolver function performs functions that map directly to the functions performed by the DNS proxy server of the '151 Patent. (Ex.1009 ¶ 52). The DNS proxy server of the '151 Patent receives a DNS request, determines whether the DNS request corresponds to a secure server (e.g., based on a domain name extension or by reference to an internal table of sites), automatically initiates/creates a secure, encrypted channel if the DNS request corresponds to a secure server, and forwards the DNS request to a conventional DNS server if the DNS request does not correspond to a secure server (see Ex.1001 at 38:23-47). Similarly, the resolver function of Kiuchi receives a DNS request, determines whether the DNS request corresponds to a secure server (based on a query to the C-HTTP name server, which essentially is just a remote table lookup similar to the internal table lookup of the '151 Patent), automatically initiates/creates a secure, encrypted channel if the DNS request corresponds to a secure server, and forwards the DNS request to a conventional DNS server if the DNS request does not correspond to a secure server.

Thus, Kiuchi's resolver function is a "DNS proxy module" within the client-side proxy because it is a program that responds to a domain name inquiry in place of a DNS (i.e., it responds to a DNS request in place of a conventional DNS server when the requested host is a member of the closed network). Kiuchi's resolver

function also is a "DNS module" within the client-side proxy because it is a program that performs a lookup service and returns an IP address for a requested domain name (i.e., it first performs a lookup to the C-HTTP name server, and then, if needed, performs a lookup to the conventional DNS server, in both cases by remote function calls); in Kiuchi, this DNS module also performs the functions of a DNS proxy by responding to a DNS request in place of the conventional DNS server when the requested host is a member of the closed network, as discussed above. Consequently, Kiuchi's client-side proxy effectively includes a Client Module and a DNS proxy module/DNS module (referred to in Figure 4 as the "DNS Proxy Server Module"). For purposes of the following analysis, the Client Module of the client-side proxy is the "client," the server-side proxy is the "secure server," and the DNS Proxy Server Module is a "DNS proxy module" or "DNS module" in the client-side proxy.

In response to receiving an HTTP request from the user agent, the Client Module sends the domain name from the URL (i.e., a "DNS request") to the DNS Proxy Server Module. This domain name will be the hostname of the server-side proxy, if the HTTP request is directed to a resource on the origin server. The client-side proxy is a data processing device having a memory storing a "DNS proxy module" or "DNS module" that intercepts DNS requests sent by the Client Module (i.e., the "client") and performs the steps recited in the claims. The Client

Module is a "client" because it is a program that sends a DNS request. The server-side proxy is a "secure server" because it is a server that requires authorization for access and can communicate in an encrypted channel. A secure, encrypted channel is automatically initiated/created between the client-side proxy and the server-side proxy if the DNS request corresponds to the server-side proxy (i.e., the "secure server").

Ground 3. Claim 1 is Anticipated by Kiuchi

As set forth in Ex.1009 ¶¶ 53-59 and Appendix D, Kiuchi discloses all of the limitations of claim 1. With regard to claim 1, the client-side proxy including the DNS Proxy Server Module is a data processing device. The functions performed in the client-side proxy, including the DNS Proxy Server Module, are necessarily stored in computer program code in memory, as is the case for any such processing device. (Ex. 1009 ¶ 44). As discussed above, the DNS Proxy Server Module is a "DNS proxy module" for purposes of claim 1 because it is a program that responds to a domain name inquiry in place of a DNS (i.e., it responds to a DNS request in place of a conventional DNS server when the requested host is a member of the closed network).

When the Client Module (i.e., the "client") receives the HTTP request from the user agent, it extracts the hostname (i.e., domain name) from the URL received in the HTTP request and sends an internal resolver request containing the domain

name to the DNS Proxy Server Module. This internal resolver request is a "DNS request" because it is a communication that contains a domain name (i.e., the hostname from the URL in the HTTP request) and requests an IP address for the domain name. If the domain name sent in the DNS request is the hostname given to the server-side proxy, then the DNS request corresponds to a secure server (i.e., the server-side proxy).

The DNS Proxy Server Module receives DNS requests sent by the Client Module (i.e., the "client"). Receipt of DNS request by the DNS Proxy Server Module from the Client Module constitutes intercepting DNS requests because the DNS Proxy Server Module receives such DNS requests ahead of a DNS function (e.g., ahead of the C-HTTP name server and ahead of the conventional DNS server). For each intercepted DNS request, the DNS Proxy Server Module performs the steps recited in the claims, as follows:

Step (1) of Claim 1 (Ex.1009 ¶¶ 53-56) - Upon receiving the DNS request from the Client Module, the DNS Proxy Server Module sends the domain name to the C-HTTP name server in the form of a C-HTTP name service request to ask the C-HTTP name server whether the client-side proxy can communicate with the specified host. (Ex.1002 at 65, sec. 2.3(2)). The C-HTTP name server "examines whether the requested server-side proxy is registered in the closed network." (Ex.1002 at 65, sec. 2.3(2)). The C-HTTP name server sends a C-HTTP name

service response to the DNS Proxy Server Module containing "the IP address and public key of the server-side proxy and both request and response Nonce values" (Ex.1002 at 65, sec. 2.3(2)), if the requested server-side proxy is registered in the closed network. The C-HTTP name server sends "a status code which indicates an error," if the requested server-side proxy is not registered in the closed network. (Ex.1002 at 65, sec. 2.3(2)-(3)).

The DNS Proxy Server Module determines whether the DNS request sent by the Client Module corresponds to a secure server based on the type of response received from the C-HTTP name server. In particular, the DNS Proxy Server Module determines that the DNS request corresponds to a secure server, only if a C-HTTP name service response is returned, and determines that the DNS request does not correspond to a secure server, if the response is an error status.

Step (2) of Claim 1 (Ex.1009 ¶ 57) - As discussed above for step (1), the DNS Proxy Server Module determines that the DNS request does not correspond to a secure server, if the response from the C-HTTP name server is an error status. When the DNS Proxy Server Module makes this determination, it performs a DNS lookup to the conventional DNS server (i.e., a DNS function that returns an IP address of a nonsecure computer), which involves forwarding the domain name (DNS request) to the conventional DNS server and receiving an IP address back from the DNS server. (Ex.1009 ¶¶ 59). Specifically, Kiuchi teaches that "If a

client-side proxy receives an error status, then it performs DNS lookup, behaving like an ordinary HTTP/1.0 proxy." (Ex.1002 at 65, section 2.3, paragraph 2). Thus, the client-side proxy, and more specifically the DNS Proxy Server Module in the client-side proxy, performs a DNS lookup in response to receiving the error status and hence performs a DNS lookup when the DNS request does not correspond to a secure server. A person of ordinary skill would have understood that Kiuchi's reference to performing a "DNS lookup ... like an ordinary HTTP/1.0 proxy" involves forwarding the DNS request to the conventional DNS server and receiving an IP address from the DNS server. For example, RFC 1123 (Ex.1010) defines how computers on the Internet should operate and states that when using domain names, *"Host domain names MUST be translated to IP addresses* as described in Section 6.1." (Ex.1010 at 13 (emphasis added).) Section 6.1, in turn, states that *"Every host MUST implement a resolver for the Domain Name System* (DNS), and it MUST implement a mechanism using this DNS resolver to convert host names to IP addresses and vice-versa." *(Id.* at 72.). Thus, in response to determining that the DNS request does not correspond to a secure server, the DNS Proxy Server Module forwards the DNS request to the conventional DNS server.

Step (3) of Claim 1 (Ex.1009 ¶ 59) – As discussed above for step (1), the DNS Proxy Server Module determines that the DNS request corresponds to a secure server, only if the response from the C-HTTP name server is a C-HTTP

name service response. When the DNS Proxy Server Module makes this determination, it sends the IP address and VPN resources (e.g., the public key and Nonce values) received in the C-HTTP name service response to the Client Module. In turn, the Client Module "sends a request for connection to the server-side proxy," which is encrypted using the server-side proxy's public key and contains the client-side proxy's IP address, hostname, request Nonce value and symmetric data exchange key for request encryption." (Ex.1002 at 65, right column, section 2.3(3)). The sending of the IP address and VPN resources by the DNS Proxy Server Module to the Client Module constitutes "automatically initiating" a secure, encrypted channel within the context of the claim because this transaction causes the client-side proxy to send a request for connection to the server-side proxy. (Ex.1002 at 65, sec. 2.3(3)). In this regard, it should be noted that the '151 Patent provides, as one example of automatically initiating/creating a secure, encrypted channel, transmission of a message requesting that a VPN be created (see Ex.1001 at 37:66-38:2). Sending the IP address and VPN resources by the DNS Proxy Server Module to the Client Module is analogous because it is a message that causes the secure, encrypted channel to be created.

Therefore, steps (1)-(3) of claim 1 are satisfied by the DNS Proxy Server Module.

Ground 4. Claim 13 is Anticipated by Kiuchi

Claim 13 is virtually identical to claim 1. Instead of being directed to the data processing device, claim 13 is a Beauregard claim directed to the computer readable medium contained within the data processing device and containing the programs for directing the steps. These program steps are contained in the program code for a client-side proxy. As discussed above, the DNS Proxy Server Module is a "DNS module" for purposes of claim 13 because it is a program that performs a lookup service and returns an IP address for a requested domain name (i.e., it first performs a lookup to the C-HTTP name server, and then, if needed, performs a lookup to the conventional DNS server, in both cases by remote function calls); in Kiuchi, this DNS module also performs the functions of a DNS proxy by responding to a DNS request in place of the conventional DNS server when the requested host is a member of the closed network, as discussed above.

The main difference in the steps as claimed is that claim 13 recites "automatically creating a secure channel between the client and the secure server." The use of public key and Nonce values create an encrypted channel which is therefore secure since those VPN resources are only made available to the client-side proxy (i.e., the "client computer") and the server-side proxy (i.e., the "secure server"). There is no user involvement in the creation of the secure channel. It is automatic. Just as the '151 patent describes "Use of a DNS Proxy to Transparently Create Virtual Private Networks" by sending a message to a gatekeeper (Ex.1001

at 37:66-38:2), so too does Kiuchi disclose creating a secure channel when the C-HTTP name server sends out public key and Nonce values in response to a DNS request. Thus, for the reasons set out above with respect to claim 1 and in view of the claim charts, claim 13 should be rejected for obviousness over Kiuchi.

D. Request 3 – Claims 1 and 13 are Unpatentable over Dalton in view of Kiuchi

Dalton was published in the Proceedings of JENC8, May 1997. (Ex. 1003) Dalton was republished in Computer Networks and ISDN Systems, Vol. 29, No. 15, November 1, 1997 (pp1799-1808). (Ex. 1024) Thus, Dalton (like Kiuchi, discussed above) is prior art under 35 U.S.C. §102(b).

Prior to October 1998, use of the Internet was expanding exponentially. Organizations with multiple locations were moving away from costly private circuits for local communication to closed virtual private networks implemented on the Internet (Ex. 1009 ¶62). Kiuchi is just one of many references describing how to implement a closed network on the Internet. The '151 patent concedes that in the prior art a "tremendous variety of methods have been proposed and implemented to provide security and anonymity for communications over the Internet." (Ex. 1001, 1:27-29) The cost incentive to set up secure connections over the Internet instead of resorting to private leased circuits was a major factor in the shift to reliance on the Internet. (Ex.1009 ¶62)

As discussed above, Kiuchi addressed the hospital space, recognizing that its technology was equally applicable for use by other institutions. Kiuchi explicitly explained the incentive to privately communicate over the Internet:

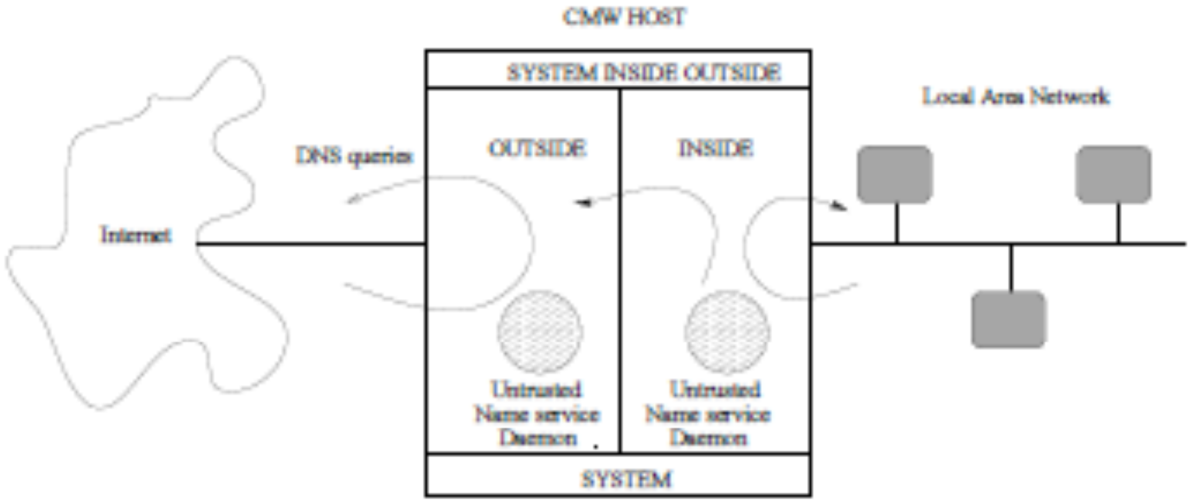
> The Internet is expected to become available to almost all major hospitals. Although a closed network can be constructed using a privately-leased circuit, additional investment for its construction is necessary. If a closed network can be constructed on the Internet, it would be *convenient, speedy and reasonable in terms of cost*. In addition, if a closed network is realized by privately leased circuits, it is not always easy to operate several closed networks flexibly and simultaneously.

(Ex. 1002, p. 69, sec. 4.5, emphasis added)

Kiuchi disclosed "a closed HTTP-based virtual network [that] can be constructed for closed groups; for example, the headquarters and branches of a given corporation." (Ex. 1002, p. 69, sec. 5) Kiuchi re-emphasized the cost incentive for moving to the Internet: "if resources which might otherwise be invested in private circuits are channeled into the Internet, it will contribute to its further development." *Id.*

Meanwhile, Dalton described a firewalled Domain Name System in a single machine, referred to as a Compartmented Mode Workstation (CMW) that provides a DNS service such that clients on an internal (closed) network can have access to both public and private hosts on the internal network as well as access to hosts on an external network, while hosts on the external network can have access through the CMW to only public hosts on the internal network (Ex. 1009¶63). Dalton

illustrates a simple example of the CMW-based system having two zones (i.e., internal/protected and external/public) as follows (Ex.1003 at 711-4, Figure 2):



As set forth in Ex.1009 ¶¶ 64-65, the CMW intercepts and processes DNS requests from external clients on the public Internet and internal clients on the closed, private Local Area Network (LAN). External clients are only permitted to access specific servers on the LAN, thus the LAN is protected by the CMW. However, internal clients are permitted to access all servers on the LAN as well as servers on the Internet. When the CMW receives a DNS request from an internal client on the LAN, it determines whether the DNS request is requesting access to an internal host on the LAN based on a set of DNS records maintained in the CMW. If the DNS request is requesting access to an internal host on the LAN, then the CMW can resolve the IP address locally, as represented by the arrow looping back to the LAN in the above figure. However, if the DNS request is requesting access to an external host on the Internet, then the CMW forwards the

DNS request to a DNS server on the Internet, as represented by the arrow from the name service daemon in the "INSIDE" box to the "OUTSIDE" box in the above figure. Dalton makes clear that an internal host can be a World Wide Web (web) server (Ex.1003 at 711-3) and that a client can include a browser (Ex.1003 at 711-6). Thus, Dalton teaches secure access to web sites by an internal client to an internal host.

The CMW includes a front end daemon that intercepts a DNS request from an internal client and forwards the DNS request to a name service daemon that includes a full set of DNS data for the internal LAN (referred to as the "SYSTEM INSIDE" level) and is responsible for resolving the DNS request. (Ex.1009 ¶ 66-67). If the name service daemon has a DNS record for the DNS request (indicating that the requested host is on the inside network), then the name service daemon can resolve the IP address locally and pass the IP address back to the internal querying client via the front end daemon. (Ex.1009 ¶ 68). However, when the internal DNS client sends a DNS request for an external host such that the name service daemon needs to query a non-local name server on the Internet in order to resolve the client query, it sends the DNS request to the external DNS server on the Internet. (Ex.1009 ¶ 69). Thus, it should be noted that Dalton's CMW is both a DNS server, because it provides a lookup service that returns an IP address for a requested

domain name, and a DNS proxy server, because it responds to a domain name inquiry in place of a DNS (e.g., in place of an external DNS server).

The LAN of Dalton would have been a costly solution for institutions with several geographically separate facilities at the time. As Kiuchi explained, the convenience, speed and costs associated with a private network implementation on the Internet were a great incentive for institutions. (see Ex.1002 at 69, ¶¶ 4.5, 5). Thus, those of ordinary skill in the art would have been clearly motivated to replace Dalton's closed, private local area network with a closed HTTP-based virtual private network. For example, the closed network of Kiuchi maintains secure communications between the two, possibly geographically separate, computers as if the two computers were still operating on the closed, private LAN.

Ground 5. Claim 1 is Obvious in view of Dalton and Kiuchi

As set forth in Ex.1009 ¶¶ 62-73 and Appendix E, the obvious combination of Dalton and Kiuchi discloses all limitations of claim 1. With regard to claim 1, the CMW is a data processing device. The functions performed in the CMW are necessarily stored in computer program code in memory, as is the case for any such processing device. (Ex. 1009, ¶ 64) An internal querying client (i.e., the "client") sends a DNS request to the CMW requesting the IP address of an internal web server on the LAN (i.e., the "target computer"). (Ex.1009 ¶¶ 65-66). A DNS request packet is intercepted by the CMW, in particular, the front end daemon

portion of the software in the CMW, before the request can be forwarded to an external DNS server or be otherwise acted upon.(Ex. 1009, ¶66)

Dalton makes clear that the Domain Name System (DNS) used in the CMW-based DNS service is the conventional DNS defined by the IETF that returns an IP address for a domain name. (Ex.1009 ¶ 63). As such, the DNS query generated and transmitted from the internal querying client is a "DNS request" as it requests an IP address for a domain name.

As discussed above, Dalton's CMW intercepts the DNS request ahead of any external DNS server. The DNS request is processed by the name service daemon. The CMW determines whether the DNS request from the internal client is requesting access to a server on the internal network based on the DNS data held by the name service daemon. (see Ex.1003 at 711-3 to 711-4). Dalton makes clear that a host on the LAN can be a World Wide Web server – see Ex.1003 at 711-3 – and therefore such a host corresponds to a secure server. It is secured by the CMW, which limits access to only authorized clients. Thus, the CMW determines whether the DNS request is requesting access to a secure server based on whether there is a DNS record for the domain name at the SYSTEM INSIDE level. Step (1) is satisfied by this determination made by the CMW. If there is any question about whether such a World Wide Web server constitutes a secure server, such question

disappears when considering Dalton in combination with the closed virtual network of Kiuchi.

As discussed above and set forth in Ex.1009 Appendix E, Dalton teaches that if the DNS request intercepted by the CMW from the internal client is requesting access to an external host on the Internet, the CMW forwards the DNS request to a DNS server on the Internet (represented by the arrow in above figure from the name service daemon in the "INSIDE" box to the "OUTSIDE" box). The DNS server passes back the IP address of the targeted external host. (Ex.1009 ¶ 69). The external hosts are not at all protected by the CMW. The external hosts correspond to nonsecure computers. Thus, upon determining the intercepted DNS request does not correspond to a protected host on the LAN, Dalton's CMW forwards the DNS request to a DNS server that returns an IP address of the nonsecure computer as recited in step (2).

If the CMW determines that the DNS request is requesting access to an internal host on the internal network, the CMW resolves the IP address for the domain name locally specifically using the set of DNS data maintained by the name service daemon running at the SYSTEM INSIDE level. Thus, Dalton teaches responding to the determination that the DNS request corresponds to an internal host by returning an IP address to initiate communications between the client and

the internal host without user involvement. This corresponds to automatically initiating.

Dalton does not teach initiating/creating a secure, encrypted channel between the client and the internal host. However, it was well-known for a DNS server or DNS proxy server to return VPN resources along with an IP address and for a VPN to be automatically initiated based on those VPN resources. (Ex.1009 ¶ 70). For example, Kiuchi teaches a DNS server (i.e., the C-HTTP name server) that receives a DNS request and, in response to determining that the DNS request is requesting access to a secure server, returns an IP address along with VPN (virtual private network) resources (e.g., a public key and Nonce values) used for automatically initiating an encrypted channel between a client computer and a target computer. (Ex.1002 at 65, sec. 2.3(2), Ex. 1009 ¶70). The client computer (i.e., client-side proxy) in turn sends a request for a secure connection to the target server (i.e., the server-side proxy) using the IP address, the public key, and the Nonce values. (Ex.1002 at 65, sec. 2.3(3), Ex. 1009 ¶70). The servers registered with the closed network are secure given that authorization by the name server is required and communications can be conducted over an encrypted channel. Thus, in Kiuchi, the encrypted channel is automatically initiated and created in response to a determination that the DNS request is requesting access to a secure server.

As discussed above, it would have been obvious to replace Dalton's private, closed LAN with a closed virtual network of the type taught by Kiuchi so that a encrypted channel is automatically initiated between an "internal" client computer and an "internal" target computer (i.e., a "secure server") in order to maintain secure communications between the two computers as if the two computers were still operating on the closed, private network. (Ex.1009 ¶ 71). Replacing Dalton's private, closed LAN with a closed virtual network of the type taught by Kiuchi would involve merely modifying Dalton's CMW to perform Kiuchi's name server functions (i.e., to return an IP address along with VPN resources if the DNS request from the "internal" client computer is requesting access to an "internal" target computer) and connecting the internal hosts to the Internet using the CMW as their DNS (proxy) server (Ex.1009 ¶ 72). It would have been obvious to modify Dalton's CMW in this way because Dalton's CMW and Kiuchi's C-HTTP name server are both DNS servers that perform a lookup service and return an IP address for a requested domain name; thus, it would have been obvious to include functionality from one device in the other. (Ex.1009 ¶ 72). Modification of Dalton also would involve adding VPN establishment functions performed by Kiuchi's client-side proxy and server-side proxy (e.g., implemented respectively in the client and target computers, or implemented respectively in firewall devices protecting the client and target computers as in Kiuchi) to automatically initiate a

VPN based on the VPN resources returned by the modified CMW. (Ex.1009 ¶ 72). Such a modified CMW also would handle requests from the server-side VPN establishment function to authenticate the client computer as in Kiuchi (Ex.1009 ¶ 72). In other words, the "internal" computers would be connected over the Internet by encrypted channels created in accordance with the closed network set up by Kiuchi. As mentioned above, there are many advantages to replacing a private, closed network of the type used in Dalton (i.e., the Local Area Network) with a closed virtual network constructed on the Internet, e.g., convenience, speed, and cost (see Ex.1002 at 69), thus providing the motivation for modifying Dalton's CMW to include Kiuchi's VPN establishment functions.

A modified CMW implementing Kiuchi's name server functions performs "automatically initiating an encrypted channel" by returning VPN resources along with an IP address in response to the DNS request, which effectively initiates and creates the encrypted channel, as discussed in Kiuchi. The '151 patent provides, as one example of automatically initiating the encrypted channel, transmission of a message requesting that a VPN be created (see Ex.1001 at 37:66-38:2). Returning VPN resources by the CMW to the client-side VPN establishment function is analogous because it is a message that causes the encrypted channel to be created without user involvement. Therefore, step (3) of claim 1 is satisfied by returning the VPN resources by the modified CMW to the client-side VPN establishment

function. Thus, the modified CMW includes a "DNS proxy module" because it contains a program that responds to a domain name inquiry in place of a DNS (i.e., it responds to a DNS request in place of a conventional DNS server when the internal client requests access to a secure internal server). The DNS proxy module intercepts DNS requests in that it receives DNS requests ahead of a DNS function (i.e., the external DNS server on the Internet), and it performs the enumerated steps in the claim.

Thus, the use of a closed network on the Internet as taught by Kiuchi in place of the LAN in Dalton discloses all elements of claim 1. At its most basic level, the claimed invention is a process offering a choice of two well-known responses to a request. Dalton teaches the CMW which is a proxy offering that choice of two responses. With respect to a DNS request for an external nonsecure computer, Dalton and the claimed data processing device respond in the same way by forwarding the request to a DNS server. With respect to a DNS request for a secure server (a server on the protected LAN in Dalton's case), Dalton automatically initiates a channel by returning the IP address. Upon replacing the LAN with a closed virtual network as taught by Kiuchi adding VPN establishment functions, this choice addresses a secure server in the closed virtual network, for whom an encrypted channel is established. As a result the claimed device is fully

taught by Dalton in view of Kiuchi. Claim 1 should be rejected for obviousness over Dalton in view of Kiuchi.

Ground 6. Claim 13 is Obvious in view of Dalton and Kiuchi

Claim 13 is virtually identical to claim 1. Instead of being directed to the data processing device, claim 13 is a Beauregard claim directed to the computer readable medium contained within the data processing device and containing the programs for directing the steps. The combination of Dalton and Kiuchi teach that all of the recited steps are contained in the program code for a CMW modified to include the VPN establishment functions. The modified CMW includes a "DNS module" because it includes a program that performs a lookup service and returns an IP address for a requested domain name (i.e., it first performs a lookup to determine if the requested host is registered in the closed network, and then, if needed, performs a lookup to the conventional DNS server by remote function call); in Kiuchi, this DNS module also performs the functions of a DNS proxy by responding to a DNS request in place of the conventional DNS server when the requested host is a member of the closed network, as discussed above. The DNS module performs the enumerated steps in the claim.

The main difference in the steps as claimed is that claim 13 recites "automatically creating a secure channel between the client and the secure server." The use of public key and Nonce values create an encrypted channel which is

therefore secured by the private encrypted communications. There is no user involvement in the creation of the secure channel. It is automatic. Just as the '151 patent describes "Use of a DNS Proxy to Transparently Create Virtual Private Networks" by sending a message to a gatekeeper (Ex. 1001, 36:55-56, 37:63-38:2), so too does Dalton/Kiuchi disclose creating a virtual private network when the CMW sends out public key and Nonce values in response to a DNS request. Thus, for the reasons set out above with respect to claim 1 and in view of the claim charts, claim 13 should be rejected for obviousness over Dalton in view of Kiuchi.

VII. CONCLUSION

Because the information presented in this petition shows that there is a reasonable likelihood that the Petitioner would prevail with respect to at least one of the claims challenged in the petition, the Petitioner respectfully requests that a Trial be instituted and that claims 1 and 13 be canceled as unpatentable.

Dated: June 23, 2013

Respectfully submitted,

/Robert M. Asher, #30,445/

Robert M. Asher, Reg. No. 30,445
Jeffrey T. Klayman, Reg. No. 39,250
Sunstein Kann Murphy & Timbers LLP
125 Summer Street, 11th Floor
Boston, MA 02110-1618
(617) 443-9292
Attorneys for Petitioner, New Bay Capital, LLC.

CERTIFICATE OF SERVICE

The undersigned certifies service pursuant to 37 C.F.R. §§42.6(e) and 42.105(b) on the Patent Owner by overnight delivery by Airport Courier Service of East Boston, MA of a copy of this Petition for Inter Partes Review and supporting materials in its entirety at the correspondence address of record for the '151 patent:

Finnegan, Henderson, Farabow, Garrett & Dunner LLP

901 New York Avenue, NW

Washington, DC 20001-4413

Date: June 23, 2013 _____/Robert M. Asher/_____

Robert M. Asher, Reg. No. 30,445