Electronic Patent Application Fee Transmittal

Application Number:	11840560
Filing Date:	17-Aug-2007
Title of Invention:	AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS USING SECURE DOMAIN NAMES
First Named Inventor/Applicant Name:	Victor Larson
Filer:	Toby H. Kusmer./Kelly Ciarmataro
Attorney Docket Number:	077580-0063 (VRNK-1CP3CN2

Filed as Large Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Claims in excess of 20	1202	43	52	2236
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Extension - 1 month with $0 paid	1251	1	130	130
Miscellaneous:				
			Total in USD ($)	**2366**

Electronic Acknowledgement Receipt

EFS ID:	7907919
Application Number:	11840560
International Application Number:	
Confirmation Number:	1537
Title of Invention:	AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS USING SECURE DOMAIN NAMES
First Named Inventor/Applicant Name:	Victor Larson
Customer Number:	23630
Filer:	Toby H. Kusmer./Kelly Ciarmataro
Filer Authorized By:	Toby H. Kusmer.
Attorney Docket Number:	077580-0063 (VRNK-1CP3CN2
Receipt Date:	28-JUN-2010
Filing Date:	17-AUG-2007
Time Stamp:	17:02:10
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$2366
RAM confirmation Number	3951
Deposit Account	501133
Authorized User	

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.19 (Document supply fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.20 (Post Issuance fees)

Charge any Additional Fees required under 37 C.F.R. Section 1.21 (Miscellaneous fees and charges)

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		AmendA.pdf	62014 d13a86bf188224a7964be0a28e8cbec6ee25774f	yes	15

Multipart Description/PDF files in .zip description

Document Description	Start	End
Amendment/Req. Reconsideration-After Non-Final Reject	1	1
Specification	2	2
Claims	3	9
Applicant Arguments/Remarks Made in an Amendment	10	15

Warnings:

Information:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
2	Fee Worksheet (PTO-875)	fee-info.pdf	32493 3c32ce143ac314ff8062b7696dd724072838eb7c	no	2

Warnings:

Information:

Total Files Size (in bytes):	94507

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PTO/SB/06 (07-06)
Approved for use through 1/31/2007. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PATENT APPLICATION FEE DETERMINATION RECORD Substitute for Form PTO-875	Application or Docket Number 11/840,560	Filing Date 08/17/2007	□ To be Mailed

APPLICATION AS FILED – PART I

	(Column 1)	(Column 2)	SMALL ENTITY □		OR	OTHER THAN SMALL ENTITY	
FOR	NUMBER FILED	NUMBER EXTRA	RATE ($)	FEE ($)		RATE ($)	FEE ($)
□ BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A			N/A	
□ SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A			N/A	
□ EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A			N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	minus 20 =	*	X $ =		OR	X $ =	
INDEPENDENT CLAIMS (37 CFR 1.16(h))	minus 3 =	*	X $ =			X $ =	
□APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).						
□ MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))							
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL			TOTAL	

APPLICATION AS AMENDED – PART II

		(Column 1)		(Column 2)	(Column 3)	SMALL ENTITY		OR	OTHER THAN SMALL ENTITY	
AMENDMENT	**06/28/2010**	CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	* 60	Minus	** 20	= 40	X $ =		OR	X $52=	2080
	Independent (37 CFR 1.16(h))	* 2	Minus	***3	= 0	X $ =		OR	X $220=	0
	□ Application Size Fee (37 CFR 1.16(s))									
	□ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
						TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	**2080**

		(Column 1)		(Column 2)	(Column 3)					
AMENDMENT		CLAIMS REMAINING AFTER AMENDMENT		HIGHEST NUMBER PREVIOUSLY PAID FOR	PRESENT EXTRA	RATE ($)	ADDITIONAL FEE ($)		RATE ($)	ADDITIONAL FEE ($)
	Total (37 CFR 1.16(i))	*	Minus	**	=	X $ =		OR	X $ =	
	Independent (37 CFR 1.16(h))	*	Minus	***	=	X $ =		OR	X $ =	
	□ Application Size Fee (37 CFR 1.16(s))									
	□ FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
						TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

Legal Instrument Examiner:
/WANDA D. MITCHELL/

This collection of information is required by 37 CFR 1.16. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
U.S. Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

23630 e 07/15/2010

McDermott Will & Emery
600 13th Street, NW
Washington, DC 20005-3096

Paper No.

Application No.:	**11/840,560**	**Date Mailed:**	**07/15/2010**
First Named Inventor:	Larson, Victor,	Examiner:	LIM, KRISNA
Attorney Docket No.:	077580-0063 (VRNK-1CP3CN2	Art Unit:	2453
Confirmation No.:	1537	Filing Date:	08/17/2007

Please find attached an Office communication concerning this application or proceeding.

Commissioner for Patents

PTO-90c (Rev.08-06)

New Bay Capital, LLC
Ex. 1007

Notice of Non-Compliant Amendment (37 CFR 1.121)	**Application No.** 11/840,560	**Applicant(s)** LARSON ET AL.	
		Art Unit 1700	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

The amendment document filed on *28 June, 2010* is considered non-compliant because it has failed to meet the requirements of 37 CFR 1.121 or 1.4. In order for the amendment document to be compliant, correction of the following item(s) is required.

THE FOLLOWING MARKED (X) ITEM(S) CAUSE THE AMENDMENT DOCUMENT TO BE NON-COMPLIANT:

☒ 1. Amendments to the specification:
- ☒ A. Amended paragraph(s) do not include markings.
- □ B. New paragraph(s) should not be underlined.
- □ C. Other _____.

□ 2. Abstract:
- □ A. Not presented on a separate sheet. 37 CFR 1.72.
- □ B. Other _____.

□ 3. Amendments to the drawings:
- □ A. The drawings are not properly identified in the top margin as “Replacement Sheet,” “New Sheet,” or “Annotated Sheet” as required by 37 CFR 1.121(d).
- □ B. The practice of submitting proposed drawing correction has been eliminated. Replacement drawings showing amended figures, without markings, in compliance with 37 CFR 1.84 are required.
- □ C. Other _____.

□ 4. Amendments to the claims:
- □ A. A complete listing of all of the claims is not present.
- □ B. The listing of claims does not include the text of all pending claims (including withdrawn claims)
- □ C. Each claim has not been provided with the proper status identifier, and as such, the individual status of each claim cannot be identified. Note: the status of every claim must be indicated after its claim number by using one of the following status identifiers: (Original), (Currently amended), (Canceled), (Previously presented), (New), (Not entered), (Withdrawn) and (Withdrawn-currently amended).
- □ D. The claims of this amendment paper have not been presented in ascending numerical order.
- □ E. Other: _____.

□ 5. Other (e.g., the amendment is unsigned or not signed in accordance with 37 CFR 1.4): For further explanation of the amendment format required by 37 CFR 1.121, see MPEP § 714.

TIME PERIODS FOR FILING A REPLY TO THIS NOTICE:

1. Applicant is given **no new time period** if the non-compliant amendment is an after-final amendment or an amendment filed after allowance, or a drawing submission (only) If applicant wishes to resubmit the non-compliant after-final amendment with corrections, the **entire corrected amendment** must be resubmitted.

2. Applicant is given **one month**, or thirty (30) days, whichever is longer, from the mail date of this notice to supply the correction, if the non-compliant amendment is one of the following: a preliminary amendment, a non-final amendment (including a submission for a request for continued examination (RCE) under 37 CFR 1.114), a supplemental amendment filed within a suspension period under 37 CFR 1.103(a) or (c), and an amendment filed in response to a Quayle action. If any of above boxes 1 to 4 are checked, the correction required is only the corrected section of the non-compliant amendment in compliance with 37 CFR 1.121.

Extensions of time are available under 37 CFR 1.136(a) only if the non-compliant amendment is a non-final amendment or an amendment filed in response to a *Quayle* action.

Failure to timely respond to this notice will result in:

Abandonment of the application if the non-compliant amendment is a non-final amendment or an amendment filed in response to a *Quayle* action; or

Non-entry of the amendment if the non-compliant amendment is a preliminary amendment or supplemental amendment.

Legal Instruments Examiner (LIE), if applicable /WANDA D. MITCHELL/ Telephone No: (571)272-1032

New Bay Capital, LLC
Ex. 1007

077580-0057 Subst. for form 1449/PTO	**Complete if Known**	
INFORMATION DISCLOSURE STATEMENT BY APPLICANT ***(Use as many sheets as necessary)***	Application Number	11/840,560
	Filing Date	08-17-2007
	First Named Inventor	Victor Larson
	Art Unit	2453
	Examiner Name	Krisna Lim
	Docket Number	077580-0063

CERTIFICATION STATEMENT

Please See 37 CFR 1.97 and 1.98 to make the appropriate selection(s)

[ ] Information Disclosure Statement is being filed before the receipt of a first office action.

[ ] Items contained in this Information Disclosure Statement were first cited in any communication from a foreign patent office in a counterpart foreign application.

[ ] No item of information contained in this Information Disclosure Statement was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the undersigned, after making reasonable inquiry, no item of information contained in the information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of this Information Disclosure Statement

[X] The Commissioner is hereby authorized to charge the fee pursuant to 37 CFR 1.17(P) in the amount of $180.00, or further fees which may be due, to Deposit Account 50-1133.

[ ] Information Disclosure Statement is being filed with the Request for Continued Examination. The Commissioner is hereby authorized to charge the fee pursuant to 37 CFR 1.17(P) in the amount of $810.00, or further fees which may be due, to Deposit Account 50-1133.

[X] None

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature

Date: August 3, 2010

Atabak R. Royaee, Reg. No.: 59,037
McDermott Will & Emery LLP
28 State Street
Boston, MA 02108
Tel. (617) 535-4000
Fax (617) 535-3800

Subst. for form 1449/PTO

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Use as many sheets as necessary)

Complete if Known	
Application Number	11/840,560
Filing Date	08-17-2007
First Named Inventor	Victor Larson
Art Unit	2453
Examiner Name	Krisna Lim
Docket Number	077580-0063

U.S. PATENTS

EXAMINER'S INITIALS	CITE NO.	Patent Number	Publication Date	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
	A1037	5,870,610	02/1999	Beyda et al.	

FOREIGN PATENT DOCUMENTS

EXAMINER'S INITIALS	CITE NO.	Foreign Patent Document Country Code3-Number 4-Kind Codes5 (*if known*)	Publication Date	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines Where Relevant Figures Appear	Translation	
						Yes	No
	B1003	EP0838930	4/29/1988	Digital Equipment Corporation			
	B1004	EP0814589	12/29/1997	AT&T Corp.			
	B1005	GB2317792	04/01/1998	Secure Computing Corporation			
	B1006	WO98/27783	06/25/1998	Northern Telecom Limited			
	B1007	WO99/11019	03/04/1999	V One Corp			
	B1008	GB2334181	08/11/1999	NEC Technologies			
	B1009	GB2340702	02/23/2000	Sun Microsystems Inc.			

OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)

EXAMINER'S INITIALS	CITE NO.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	
	C1244	Baumgartner et al, “Differentiated Services: A New Approach for Quality of Service in the Internet,” International Conference on High Performance Networking, 255-273 (1998)	
	C1245	Chapman et al., “Domain Name System (DNS),” 278-296 (1995)	
	C1246	Davila et al., “Implementation of Virtual Private Networks at the Transport Layer,” M. Mambo, Y. Zheng (Eds), Information Security (Second International) Workshop, ISW' 99. Lecture Notes in Computer Science (LNCS), Vol. 1729; 85-102 (1999)	
	C1247	De Raadt et al., “Cryptography in OpenBSD,” 10 pages (1999)	
	C1248	Eastlake, “Domain Name System Security Extensions,” Internet Citation, Retrieved from the Internet: URL:ftp://ftp.inet.no/pub/ietf/internet-drafts/draft-ietf-dnssec-secext2-05.txt (1998)	

Subst. for form 1449/PTO

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Use as many sheets as necessary)

Complete if Known	
Application Number	11/840,560
Filing Date	08-17-2007
First Named Inventor	Victor Larson
Art Unit	2453
Examiner Name	Krisna Lim
Docket Number	077580-0063

	C1249	Gunter et al., “An Architecture for Managing QoS-Enabled VRNs Over the Internet,” Proceedings 24th Conference on Local Computer Networks. LCN’ 99 IEEE Comput. Soc Los Alamitos, CA, pages 122-131 (1999)	
	C1250	Shimizu, “Special Feature: Mastering the Internet with Windows 2000”, Internet Magazine, 63:296-307 (2000)	
	C1251	Stallings, “Cryptography and Network Security,” Principals and Practice, 2nd Edition, pages 399-440 (1999)	
	C1252	Takata, “U.S. Vendors Take Serious Action to Act Against Crackers – A Tracking Tool and a Highly Safe DNS Software are Released”, Nikkei Communications, 257:87(1997)	
	C1253	Wells, Email (Lancasterb1be@mail.msn.com), Subject: “Security Icon,” (1998)	

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 838 930 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
29.04.1998 Bulletin 1998/18

(51) Int. Cl.^{6}: **H04L 29/06**

(21) Application number: **97118556.6**

(22) Date of filing: **24.10.1997**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV RO SI

(30) Priority: **25.10.1996 US 738155**

(71) Applicant:
DIGITAL EQUIPMENT CORPORATION
Maynard, Massachusetts 01754 (US)

(72) Inventors:
- **Alden, Kenneth F.**
  Boylston, Massachusetts 01505 (US)
- **Lichtenberg, Mitchell P.**
  Sunnyvale, CA 94087 (US)
- **Wobber, Edward P.**
  Menlo Park, California 94025 (US)

(74) Representative: **Betten & Resch**
Reichenbachstrasse 19
80469 München (DE)

(54) Pseudo network adapter for frame capture, encapsulation and encryption

(57) A new pseudo network adapter provides an interface for capturing packets from a local communications protocol stack for transmission on the virtual private network, and includes a Dynamic Host Configuration Protocol (DHCP) server emulator, and an Address Resolution Protocol (ARP) server emulator. The new system indicates to the local communications protocol stack that nodes on a remote private network are reachable through a gateway that is in turn reachable through the pseudo network adapter. A transmit path in the system processes data packets from the local communications protocol stack for transmission through the pseudo network adapter. An encryption engine encrypts the data packets and an encapsulation engine encapsulates the encrypted data packets into tunnel data frames. The network adapter further includes an interface into a transport layer of the local communications protocol stack for capturing received data packets from the remote server node, and a receive path for processing received data packets captured from the transport layer of the local communications protocol stack. The receive path includes a decapsulation engine, and a decryption engine, and passes the decrypted, decapsulated data packets back to the local communications protocol stack for delivery to a user.

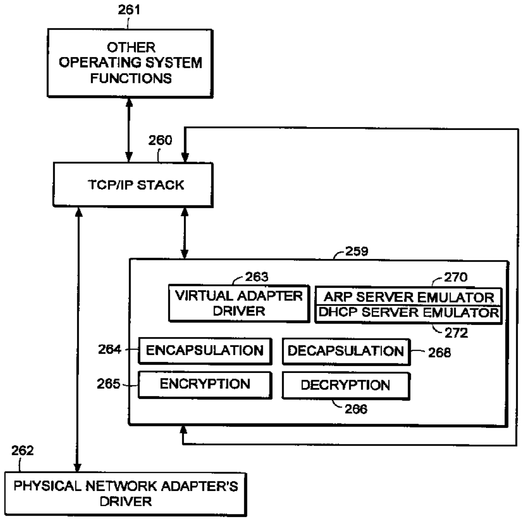


FIG. 15

Printed by Xerox (UK) Business Services
2.16.1/3.4

Description

FIELD OF THE INVENTION

The invention relates generally to establishing secure virtual private networks. The invention relates specifically to a pseudo network adapter for capturing, encapsulating and encrypting messages or frames.

BACKGROUND

In data communications it is often required that secure communications be provided between users of network stations (also referred to as "network nodes") at different physical locations. Secure communications must potentially extend over public networks as well as through secure private networks. Secure private networks are protected by "firewalls", which separate the private network from a public network. Firewalls ordinarily provide some combination of packet filtering, circuit gateway, and application gateway technology, insulating the private network from unwanted communications with the public network.

One approach to providing secure communications is to form a virtual private network. In a virtual private network, secure communications are provided by encapsulating and encrypting messages. Encapsulated messaging in general is referred to as "tunneling". Tunnels using encryption may provide protected communications between users separated by a public network, or among a subset of users of a private network.

Encryption may for example be performed using an encryption algorithm using one or more encryption "keys". When an encryption key is used, the value of the key determines how the data is encrypted and decrypted. When a public-key encryption system is used, a key pair is associated with each communicating entity. The key pair consists of an encryption key and a decryption key. The two keys are formed such that it is unfeasible to generate one key from the other. Each entity makes its encryption key public, while keeping its decryption key secret. When sending a message to node A, for example, the transmitting entity uses the public key of node A to encrypt the message, and then the message can only be decrypted by node A using node A's private key.

In a symmetric key encryption system a single key is used as the basis for both encryption and decryption. An encryption key in a symmetric key encryption system is sometimes referred to as a "shared" key. For example, a pair of communicating nodes A and B could communicate securely as follows: a first shared key is used to encrypt data sent from node A to node B, while a second shared key is to be used to encrypt data sent from node B to node A. In such a system, the two shared keys must be known by both node A and node B. More examples of encryption algorithms and keyed encryption are disclosed in many textbooks, for example "Applied Cryptography - Protocols, Algorithms, and Source Code in C", by Bruce Schneier, published by John Wiley and Sons, New York, New York, copyright 1994.

Information regarding what encryption key or keys are to be used, and how they are to be used to encrypt data for a given secure communications session is referred to as "key exchange material". Key exchange material may for example determine what keys are used and a time duration for which each key is valid. Key exchange material for a pair of communicating stations must be known by both stations before encrypted data can be exchanged in a secure communications session. How key exchange material is made known to the communicating stations for a given secure communications session is referred to as "session key establishment".

A tunnel may be implemented using a virtual or "pseudo" network adapter that appears to the communications protocol stack as a physical device and which provides a virtual private network. A pseudo network adapter must have the capability to receive packets from the communications protocol stack, and to pass received packets back through the protocol stack either to a user or to be transmitted.

A tunnel endpoint is the point at which any encryption/decryption and encapsulation/decapsulation provided by a tunnel is performed. In existing systems, the tunnel end points are pre-determined network layer addresses. The source network layer address in a received message is used to determine the "credentials" of an entity requesting establishment of a tunnel connection. For example, a tunnel server uses the source network layer address to determine whether a requested tunnel connection is authorized. The source network layer address is also used to determine which cryptographic key or keys to use to decrypt received messages.

Existing tunneling technology is typically performed by encapsulating encrypted network layer packets (also referred to as "frames") at the network layer. Such systems provide "network layer within network layer" encapsulation of encrypted messages. Tunnels in existing systems are typically between firewall nodes which have statically allocated IP addresses. In such existing systems, the statically allocated IP address of the firewall is the address of a tunnel end point within the firewall. Existing systems fail to provide a tunnel which can perform authorization based for an entity which must dynamically allocate its network layer address. This is especially problematic for a user wishing to establish a tunnel in a mobile computing environment, and who requests a dynamically allocated IP address from an Internet Service Provider (ISP).

Because existing virtual private networks are based on network layer within network layer encapsulation, they are generally only capable of providing connectionless datagram type services. Because datagram type services do not guarantee delivery of packets, existing tunnels can only easily employ encryption methods over the data contained within each transmitted packet. Encryption based on the contents of multiple packets is desirable, such as cipher block chaining or stream ciphering over multiple packets. For example, encrypted data would advantageously be formed based not only on the contents of the present packet data being encrypted, but also based on some attribute of the connection or session history between the communicating stations. Examples of encryption algorithms and keyed encryption are disclosed in many textbooks, for example "Applied Cryptography - Protocols, Algorithms, and Source Code in C", by Bruce Schneier, published by John Wiley and Sons, New York, New York, copyright 1994.

Thus there is required a new pseudo network adapter providing a virtual private network having a dynamically determined end point to support a user in a mobile computing environment. The new pseudo network adapter should appear to the communications protocol stack of the node as an interface to an actual physical device. The new pseudo network adapter should support guaranteed, in-order delivery of frames over a tunnel to conveniently support cipher block chaining mode or stream cipher encryption over multiple packets.

SUMMARY OF THE INVENTION

A new pseudo network adapter is disclosed providing a virtual private network. The new system includes an interface for capturing packets from a local communications protocol stack for transmission on the virtual private network. The interface appears to the local communications stack as a network adapter device driver for a network adapter.

The invention, in its broad form, includes a pseudo network adapter as recited in claim 1, providing a virtual network and a method therefor as recited in claim 9.

The system as described hereinafter further includes a Dynamic Host Configuration Protocol (DHCP) server emulator, and an Address Resolution Protocol (ARP) server emulator. The new system indicates to the local communications protocol stack that nodes on a remote private network are reachable through a gateway that is in turn reachable through the pseudo network adapter. The new pseudo network adapter includes a transmit path for processing data packets from the local communications protocol stack for transmission through the pseudo network adapter. The transmit path includes an encryption engine for encrypting the data packets and an encapsulation engine for encapsulating the encrypted data packets into tunnel data frames. The pseudo network adapter passes the tunnel data frames back to the local communications protocol stack for transmission to a physical network adapter on a remote server node.

Preferably, as described hereinafter, the pseudo network adapter includes a digest value in a digest field in each of the tunnel data frames. A keyed hash function is a hash function which takes data and a shared cryptographic key as inputs, and outputs a digital signature referred to as a digest. The value of the digest field is equal to an output of a keyed hash function applied to data consisting of the data packet encapsulated within the tunnel data frame concatenated with a counter value equal to a total number of tunnel data frames previously transmitted to the remote server node. In another aspect of the system, the pseudo network adapter processes an Ethernet header in each one of the captured data packets, including removing the Ethernet header.

The new pseudo network adapter further includes an interface into a transport layer of the local communications protocol stack for capturing received data packets from the remote server node, and a receive path for processing received data packets captured from the transport layer of the local communications protocol stack. The receive path includes a decapsulation engine, and a decryption engine, and passes the decrypted, decapsulated data packets back to the local communications protocol stack for delivery to a user.

Thus there is disclosed a new pseudo network adapter providing a virtual private network having dynamically determined end points to support users in a mobile computing environment. The new pseudo network adapter provides a system for capturing a fully formed frame prior to transmission. The new pseudo network adapter appears to the communications protocol stack of the station as an interface to an actual physical device. The new pseudo network adapter further includes encryption capabilities to conveniently provide secure communications between tunnel end points using stream mode encryption or cipher block chaining over multiple packets.

BRIEF DESCRIPTION OF THE DRAWINGS

A more detailed understanding of the invention may be had from the following description of a preferred embodiment, given by way of example and to be understood in conjunction with the accompanying drawing in which:

- Fig. 1 is a block diagram showing the Open Systems Interconnection (OSI) reference model;
- Fig. 2 is a block diagram showing the TCP/IP internet protocol suite;
- Fig. 3 is a block diagram showing an examplary embodiment of a tunnel connection across a public network between two tunnel servers;
- Fig. 4 is a flow chart showing an examplary embodiment of steps performed to establish a tunnel con-

nection;

- Fig. 5 is a flow chart showing an examplary embodiment of steps performed to perform session key management for a tunnel connection;
- Fig. 6 is a block diagram showing an examplary embodiment of a relay frame;
- Fig. 7 is a block diagram showing an examplary embodiment of a connection request frame;
- Fig. 8 is a block diagram showing an examplary embodiment of a connection response frame;
- Fig. 9 is a block diagram showing an examplary embodiment of a data frame;
- Fig. 10 is a block diagram showing an examplary embodiment of a close connection frame;
- Fig. 11 is a state diagram showing an examplary embodiment of a state machine forming a tunnel connection in a network node initiating a tunnel connection;
- Fig. 12 is a state diagram showing an examplary embodiment of a state machine forming a tunnel connection in a server computer;
- Fig. 13 is a state diagram showing an examplary embodiment of a state machine forming a tunnel connection in a relay node;
- Fig. 14 is a block diagram showing an examplary embodiment of a tunnel connection between a client computer (tunnel client) and a server computer (tunnel server);
- Fig. 15 is a block diagram showing an examplary embodiment of a pseudo network adapter;
- Fig. 16 is a block diagram showing an examplary embodiment of a pseudo network adapter;
- Fig. 17 is a flow chart showing steps performed by an examplary embodiment of a pseudo network adapter during packet transmission;
- Fig. 18 is a flow chart showing steps performed by an examplary embodiment of a pseudo network adapter during packet receipt;
- Fig. 19 is a data flow diagram showing data flow in an examplary embodiment of a pseudo network adapter during packet transmission;
- Fig. 20 is a data flow diagram showing data flow in an examplary embodiment of a pseudo network adapter during packet receipt;
- Fig. 21 is a diagram showing the movement of encrypted and unencrypted data in an examplary embodiment of a system including a pseudo network adapter;
- Fig. 22 is a diagram showing the movement of encrypted and unencrypted data in an examplary embodiment of a system including a pseudo network adapter; and
- Fig. 23 is a flow chart showing steps initialization of an examplary embodiment of a system including a pseudo network adapter.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Now with reference to Fig. 1 there is described for purposes of explanation, communications based on the Open Systems Interconnection (OSI) reference model. In Fig. 1 there is shown communications 12 between a first protocol stack 10 and a second protocol stack 14. The first protocol stack 10 and second protocol stack 14 are implementations of the seven protocol layers (Application layer, Presentation layer, Session layer, Transport layer, Network layer, Data link layer, and Physical layer) of the OSI reference model. A protocol stack implementation is typically in some combination of software and hardware. Descriptions of the specific services provided by each protocol layer in the OSI reference model are found in many text books, for example "Computer Networks", Second Edition, by Andrew S. Tannenbaum, published by Prentice-Hall, Englewood Cliffs, New Jersey, copyright 1988.

As shown in Fig. 1, data 11 to be transmitted from a sending process 13 to a receiving process 15 is passed down through the protocol stack 10 of the sending process to the physical layer 9 for transmission on the data path 7 to the receiving process 15. As the data 11 is passed down through the protocol stack 10, each protocol layer prepends a header (and possibly also appends a trailer) portion to convey information used by that protocol layer. For example, the data link layer 16 of the sending process wraps the information received from the network layer 17 in a data link header 18 and a data link layer trailer 20 before the message is passed to the physical layer 9 for transmission on the actual transmission path 7.

Fig. 2 shows the TCP/IP protocol stack. Some protocol layers in the TCP/IP protocol stack correspond with layers in the OSI protocol stack shown in Fig. 1. The detailed services and header formats of each layer in the TCP/IP protocol stack are described in many texts, for example "Internetworking with TCP/IP, Vol. 1: Principles, Protocols, and Architecture", Second Edition, by Douglas E. Comer, published by Prentice-Hall, Englewood Cliffs, New Jersey, copyright 1991. The Transport Control Protocol (TCP) 22 corresponds to the Transport layer in the OSI reference model. The TCP protocol 22 provides a connection-oriented, end to end transport service with guaranteed, in-sequence packet delivery. In this way the TCP protocol 22 provides a reliable, transport layer connection.

The IP protocol 26 corresponds to the Network layer of the OSI reference model. The IP protocol 26 provides no guarantee of packet delivery to the upper layers. The hardware link level and access protocols 32 correspond to the Data link and Physical layers of the OSI reference model.

The Address Resolution Protocol (ARP) 28 is used to map IP layer addresses (referred to as "IP addresses") to addresses used by the hardware link level and access protocols 32 (referred to as "physical addresses" or "MAC addresses"). The ARP protocol layer in each network station typically contains a table of mappings between IP addresses and physical addresses (referred to as the "ARP cache"). When a mapping between an IP address and the corresponding physical address is not known, the ARP protocol 28 issues a broadcast packet (an "ARP request" packet) on the local network. The ARP request indicates an IP address for which a physical address is being requested. The ARP protocols 28 in each station connected to the local network examine the ARP request, and if a station recognizes the IP address indicated by the ARP request, it issues a response (an "ARP response" or "ARP reply" packet) to the requesting station indicating the responder's physical address. The requesting ARP protocol reports the received physical address to the local IP layer which then uses it to send datagrams directly to the responding station. As an alternative to having each station respond only for its own IP address, an ARP server may be used to respond for a set of IP addresses it stores internally, thus potentially eliminating the requirement of a broadcast request. In that case, the ARP request can be sent directly to the ARP server for physical addresses corresponding to any IP address mappings stored within the ARP server.

At system start up, each station on a network must determine an IP address for each of its network interfaces before it can communicate using TCP/IP. For example, a station may need to contact a server to dynamically obtain an IP address for one or more of its network interfaces. The station may use what is referred to as the Dynamic Host Configuration Protocol (DHCP) to issue a request for an IP address to a DHCP server. For example, a DHCP module broadcasts a DHCP request packet at system start up requesting allocation of an IP address for an indicated network interface. Upon receiving the DHCP request packet, the DHCP server allocates an IP address to the requesting station for use with the indicated network interface. The requesting station then stores the IP address in the response from the server as the IP address to associate with that network interface when communicating using TCP/IP.

Fig. 3 shows an example configuration of network nodes for which the presently disclosed system is applicable. In the example of Fig. 3, the tunnel server A is an initiator of the tunnel connection. As shown in Fig. 3, the term "tunnel relay" node is used to refer to a station which forwards data packets between transport layer connections (for example TCP connections).

For example, in the present system a tunnel relay may be dynamically configured to forward packets between transport layer connection 1 and transport layer connection 2. The tunnel relay replaces the header information of packets received over transport layer connection 1 with header information indicating transport layer connection 2. The tunnel relay can then forward the packet to a firewall, which may be conveniently programmed to pass packets received over transport layer connection 2 into a private network on the other side of the firewall. In the present system, the tunnel relay dynamically forms transport layer connections when a tunnel connection is established. Accordingly the tunnel relay is capable of performing dynamic load balancing or providing redundant service for fault tolerance over one or more tunnel servers at the time the tunnel connection is established.

Fig. 3 shows a Tunnel Server A 46 in a private network N1 48, physically connected with a first Firewall 50. The first Firewall 50 separates the private network N1 48 from a public network 52, for example the Internet. The first Firewall 50 is for example physically connected with a Tunnel Relay B 54, which in turn is virtually connected through the public network 52 with a Tunnel Relay C. The connection between Tunnel Relay B and Tunnel Relay C may for example span multiple intervening forwarding nodes such as routers or gateways through the public network 52.

The Tunnel Relay C is physically connected with a second Firewall 58, which separates the public network 52 from a private network N2 60. The second Firewall 58 is physically connected with a Tunnel Server D 62 on the private network N2 60. During operation of the elements shown in Fig. 3, the Tunnel Server D 62 provides routing of IP packets between the tunnel connection with Tunnel Server A 46 and other stations on the private network N2 60. In this way the Tunnel Server D 62 acts as a router between the tunnel connection and the private network N2 60.

During operation of the elements shown in Fig. 3, the present system establishes a tunnel connection between the private network N1 48 and the private network N2 60. The embodiment of Fig. 3 thus eliminates the need for a dedicated physical cable or line to provide secure communications between the private network 48 and the private network 60. The tunnel connection between Tunnel Server A 46 and Tunnel Server D 62 is composed of reliable, pair-wise transport layer connections between Tunnel Server A 46 (node "A"), Tunnel Relay B 54 (node "B"), Tunnel Relay C 56 (node "C"), and Tunnel Server D 62 (node "D"). For example, such pair-wise connections may be individual transport layer connections between each node A and node B, node B and node C, and node C and node D. In an alternative embodiment, as will be described below, a tunnel connection may alternatively be formed between a standalone PC in a public network and a tunnel server within a private network.

Fig. 4 and Fig. 5 show an example embodiment of steps performed during establishment of the tunnel connection between Tunnel Server A 46 (node "A") and Tunnel Server D 62 (node "D") as shown in Fig. 3. Prior to the steps shown in Fig. 4, node A selects a tunnel path to reach node D. The tunnel path includes the tunnel end points and any intervening tunnel relays. The tunnel path is for example predetermined by a system administrator for node A. Each tunnel relay along the tunnel path is capable of finding a next node in the tunnel path, for example based on a provided next node name (or "next node arc"), using a predetermined naming convention and service, for example the Domain Name System (DNS) of the TCP/IP protocol suite.

During the steps shown in Fig. 4, each of the nodes A, B and C perform the following steps:

- resolve the node name of the next node in the tunnel path, for example as found in a tunnel relay frame;

- establish a reliable transport layer (TCP) connection to the next node in the tunnel path;

- forward the tunnel relay frame down the newly formed reliable transport layer connection to the next node in the tunnel path.

As shown for example in Fig. 4, at step 70 node A establishes a reliable transport layer connection with node B. At step 72 node A identifies the next downstream node to node B by sending node B a tunnel relay frame over the reliable transport layer connection between node A and node B. The tunnel relay frame contains a string buffer describing all the nodes along the tunnel path (see below description of an example tunnel relay frame format). At step 74, responsive to the tunnel relay frame from node A, node B searches the string buffer in the relay frame to determine if the string buffer includes node B's node name. If node B finds its node name in the string buffer, it looks at the next node name in the string buffer to find the node name of the next node in the tunnel path.

Node B establishes a reliable transport layer connection with the next node in the tunnel path, for example node C. Node B further forms an association between the reliable transport layer connection between Node A and Node B, over which the relay frame was received, and the newly formed reliable transport layer connection between Node B and Node C, and as a result forwards subsequent packets received over the reliable transport layer connection with Node A onto the reliable transport layer connection with Node C, and vice versa. At step 76 node B forwards the tunnel relay frame on the newly formed reliable transport layer connection to node C.

At step 78, responsive to the relay frame forwarded from node B, node C determines that the next node in the tunnel path is the last node in the tunnel path, and accordingly is a tunnel server. Node C may actively determine whether alternative tunnel servers are available to form the tunnel connection. Node C may select one of the alternative available tunnel servers to form the tunnel connection in order to provide load balancing or fault tolerance. As a result node C may form a transport layer connections with one of several available tunnel servers, for example a tunnel server that is relatively underutilized at the time the tunnel connection is established. In the example embodiment, node C establishes a reliable transport layer connection with the next node along the tunnel path, in this case node D.

Node C further forms an association between the reliable transport layer connection between Node B and Node C, over which the relay frame was received, and the newly formed reliable transport layer connection between Node C and Node D, and as a result forwards subsequent packets received over the reliable transport layer connection with Node B to the reliable transport layer connection with Node D, and vice versa. At step 80 node C forwards the relay frame to node D on the newly formed reliable transport layer connection.

Fig. 5 shows an example of tunnel end point authentication and sharing of key exchange material provided by the present system. The present system supports passing authentication data and key exchange material through the reliable transport layer connections previously established on the tunnel path. The following are provided by use of a key exchange/authentication REQUEST frame and a key exchange/authentication RESPONSE frame:

a) mutual authentication of both endpoints of the tunnel connection;

b) establishment of shared session encryption keys and key lifetimes for encrypting/authenticating subsequent data sent through the tunnel connection;

d) agreement on a shared set of cryptographic transforms to be applied to subsequent data; and

e) exchange of any other connection-specific data between the tunnel endpoints, for example strength and type of cipher to be used, any compression of the data to be used, etc. This data can also be used by clients of this protocol to qualify the nature of the authenticated connection.

At step 90 a key exchange/authentication request frame is forwarded over the reliable transport layer connections formed along the tunnel path from node A to node D. At step 92, a key exchange/authentication response frame is forwarded from node D back to node A through the reliable transport layer connections. The attributes exchanged using the steps shown in Fig. 5 may be used for the lifetime of the tunnel connection. In an alternative embodiment the steps shown in Fig. 5 are repeated as needed for the tunnel end points to exchange sufficient key exchange material to agree upon a set of session parameters for use during the tunnel connection such as cryptographic keys, key durations, and choice of encryption/decryption algorithms.

Further in the disclosed system, the names used for authentication and access control with regard to node A and node D need not be the network layer address or physical address of the nodes. For example, in an alternative embodiment where the initiating node sending the tunnel relay frame is a stand-alone PC located within a public network, the user's name may be used for authentication and/or access control purposes. This provides a significant improvement over existing systems which base authorization on predetermined IP addresses.

Fig. 6 shows the format of an example embodiment of a tunnel relay frame. The tunnel frame formats shown in Figs. 6, 7, 8 and 9 are encapsulated within the data portion of a transport layer (TCP) frame when transmitted. Alternatively, another equivalent, connection-oriented transport layer protocol having guaranteed, in-sequence frame delivery may be used. The example TCP frame format, including TCP header fields, is conventional and not shown.

The field 100 contains a length of the frame. The field 102 contains a type of the frame, for example a type of RELAY. The field 104 contains a tunnel protocol version number. The field 106 contains an index into a string buffer field 112 at which a name of the originating node is located, for example a DNS host name of the node initially issuing the relay frame (node A in Fig. 3). The fields following the origin index field 106 contain indexes into the string buffer 112 at which names of nodes along the tunnel path are located. For example each index may be the offset of a DNS host name within the string buffer 112. In this way the field 108 contains the index of the name of the first node in the tunnel path, for example node B (Fig. 3). The field 110 contains the index of the name of the second node in the tunnel path, etc. The field 112 contains a string of node names of nodes in the tunnel path.

During operation of the present system, the initiating node, for example node A as shown in Fig. 3, transmits a tunnel relay frame such as the tunnel relay frame shown in Fig. 6. Node A sends the tunnel relay frame to the first station along the tunnel path, for example node B (Fig. 3), over a previously established reliable transport layer connection. Node B searches the string buffer in the tunnel relay frame to find its node name, for example its DNS host name. Node B finds its node name in the string buffer indexed by path index 0, and then uses the contents of path index 1 110 to determine the location within the string buffer 112 of the node name of the next node along the tunnel path. Node B uses this node name to establish a reliable transport layer connection with the next node along the tunnel path. Node B then forwards the relay frame to the next node. This process continues until the end node of the tunnel route, for example tunnel server D 62 (Fig. 3) is reached.

Fig. 7 shows the format of an example embodiment of a key exchange/key authentication request frame. The field 120 contains a length of the frame. The field 122 contains a type of the frame, for example a type of REQUEST indicating a key exchange/key authentication request frame. The field 124 contains a tunnel protocol version number. The field 126 contains an offset of the name of the entity initiating the tunnel connection, for example the name of a user on the node originally issuing the request frame. This name and key exchange material in the request frame are used by the receiving tunnel end point to authenticate the key exchange/authentication REQUEST. The name of the entity initiating the tunnel connection is also use to authorize any subsequent tunnel connection, based on predetermined security policies of the system. The field 128 contains an offset into the frame of the node name of the destination node, for example the end node of the tunnel shown as node D 62 in Fig. 3.

The field 130 contains an offset into the frame at which key exchange data as is stored, for example within the string buffer field 138. The key exchange data for example includes key exchange material used to determine a shared set of encryption parameters for the life of the tunnel connection such as cryptographic keys and any validity times associated with those keys. The key exchange data, as well as the field 132, further include information regarding any shared set of cryptographic transforms to be used and any other connection-specific parameters, such as strength and type of cipher to be used, type of compression of the data to be used, etc. The field 134 contains flags, for example indicating further information about the frame. The field 136 contains client data used in the tunnel end points to configure the local routing tables so that packets for nodes reachable through the virtual private network are sent through the pseudo network adapters. In an example embodiment, the string buffer 138 is encrypted using a public encryption key of the receiving tunnel end point.

During operation of the present system, one of the end nodes of the tunnel sends a key exchange/authentication REQUEST frame as shown in Fig. 7 to the other end node of the tunnel in order to perform key exchange and authentication as described in step 90 of Fig. 5.

Fig. 8 shows the format of an example embodiment of a key exchange/key authentication response frame, referred to as a connection RESPONSE frame. The field 150 contains a length of the frame. The field 152 contains a type of the frame, for example a type of connection RESPONSE indicating a key exchange/key authentication request frame. The field 154 contains a tunnel protocol version number.

The field 156 contains an offset into the frame at which key exchange data as is stored, for example within the string buffer field 163. The key exchange data for example includes key exchange material to be used for encryption/decryption over the life of the tunnel connection and any validity times associated with that key exchange material. The key exchange data, as well as the field 158, further includes information regarding any shared set of cryptographic transforms to be applied to subsequent data and any other connection-specific parameters, such as strength and type of cipher to be used, any compression of the data to be used, etc. The field 160 contains flags, for example indicating other information about the frame. The client data field 162 contains data used by the pseudo network adapters in the tunnel end points to configure the local routing tables so that packets for nodes in the virtual private network are sent through the pseudo network adapters. The string buffer includes key exchange material. The string buffer is for example encrypted using a public encryption key of the receiving tunnel end point, in the this case the initiator of the tunnel connection.

During operation of the present system, one of the end nodes of the tunnel sends a key exchange/authentication RESPONSE frame as shown in Fig. 7 to the other end node of the tunnel in order to perform key exchange and authentication as described in step 92 of Fig. 5.

Fig. 9 shows the format of an example embodiment of an tunnel data frame used to communicate through a tunnel connection. Fig. 9 shows how an IP datagram may be encapsulated within a tunnel frame by the present system for secure communications through a virtual private network. The field 170 contains a length of the frame. The field 172 contains a type of the frame, for example a type of DATA indicating a tunnel data frame. The field 174 contains a tunnel protocol version number.

The fields 176, 178 and 182 contain information regarding the encapsulated datagram. The field 180 contains flags indicating information regarding the frame. The field 184 contains a value indicating the length of the optional padding 189 at the end of the frame. The frame format allows for optional padding in the event that the amount of data in the frame needs to be padded to an even block boundary for the purpose of being encrypted using a block cipher. The field 186 contains a value indicating the length of the digest field 187.

The data frame format includes a digital signature generated by the transmitting tunnel end point referred to as a "digest". The value of the digest ensures data integrity, for example by detecting invalid frames and replays of previously transmitted valid frames. The digest is the output of a conventional keyed cryptographic hash function applied to both the encapsulated datagram 190 and a monotonically increasing sequence number. The resulting hash output is passed as the value of the digest field 187. The sequence number is not included in the data frame. In the example embodiment, the sequence number is a counter maintained by the transmitter (for example node A in Fig. 3) of all data frames sent to the receiving node (for example node D in Fig. 3) since establishment of the tunnel connection.

In order to determine if the data frame is invalid or a duplicate, the receiving node decrypts the encapsulated datagram 190, and applies the keyed cryptographic hash function (agreed to by the tunnel end nodes during the steps shown in Fig. 5) to both the decrypted encapsulated datagram and the value of a counter indicating the number of data frames received from the transmitter since establishment of the tunnel connection. For example the keyed hash function is applied to the datagram concatenated to the counter value. If the resulting hash output matches the value of the digest field 187, then the encapsulated datagram 190 was received correctly and is not a duplicate. If the hash output does not match the value of the digest field 187, then the integrity check fails, and the tunnel connection is closed. The field 188 contains an encrypted network layer datagram, for example an encrypted IP datagram.

The encapsulated datagram may be encrypted using various encryption techniques. An example embodiment of the present system advantageously encrypts the datagram 190 using either a stream cipher or cipher block chaining encryption over all data transmitted during the life of the tunnel connection. This is enabled by the reliable nature of the transport layer connections within the tunnel connection. The specific type of encryption and any connection specific symmetric encryption keys used is determined using the steps shown in Fig. 5. The fields in the tunnel data frame other than the encapsulated datagram 188 are referred to as the tunnel data frame header fields.

Fig. 10 is a block diagram showing an example embodiment of a "close connection" frame. The field 190 contains the length of the frame. The field 191 contains a frame type, for example having a value equal to CLOSE. Field 192 contains a value equal to the current protocol version number of the tunnel protocol. The field 193 contains a status code indicating the reason the tunnel connection is being closed.

During operation of the present system, when end point of a tunnel connection determines that the tunnel connection should be closed, a close connection frame as shown in Fig. 10 is transmitted to the other end point of the tunnel connection. When a close connection close frame is received, the receiver closes the tunnel connection and no further data will be transmitted or received through the tunnel connection.

Fig. 11 is a state diagram showing an example embodiment of forming a tunnel connection in a node initiating a tunnel connection. In Fig. 11, Fig. 12, and Fig. 13, states are indicated by ovals and actions or events are indicated by rectangles. For example the tunnel server node A as shown in Fig. 3 may act as a tunnel connection initiator when establishing a tunnel connection with the tunnel server node D. Similarly the client system 247 in Fig. 14 may act as a tunnel connection initiator when establishing a tunnel connection with the tunnel server. The tunnel initiator begins in an idle state 194. Responsive to an input from a user indicating that a tunnel connection should be established, the tunnel initiator transitions from the idle state 194 to a TCP Open state 195. In the TCP Open state 195, the tunnel initiator establishes a reliable transport layer connection with a first node along the tunnel path. For example, the tunnel initiator opens a socket interface associated with a TCP connection to the first node along the tunnel path. In Fig. 3 node A opens a socket interface associated with a TCP connection with node B.

Following establishment of the reliable transport layer connection in the TCP Open state 195, the tunnel initiator enters a Send Relay state 197. In the Send Relay state 197, the tunnel initiator transmits a relay frame at 198 over the reliable transport layer connection. Following transmission of the relay frame, the tunnel initiator enters the connect state 199. If during transmission of the relay frame there is a transmission error, the tunnel initiator enters the Network Error state 215 followed by the Dying state 208. In the Dying state 208, the tunnel initiator disconnects the reliable transport layer connection formed in the TCP Open state 195, for example by disconnecting a TCP connection with Node B. Following the disconnection at 209, the tunnel initiator enters the Dead state 210. The tunnel initiator subsequently transitions back to the Idle state 194 at a point in time predetermined by system security configuration parameters.

In the Connect state 199, the tunnel initiator sends a key exchange/authentication REQUEST frame at 200 to the tunnel server. Following transmission of the key exchange/authentication REQUEST frame 200, the tunnel initiator enters the Response Wait state 201. The tunnel initiator remains in the Response Wait state 201 until it receives a key exchange/authentication RESPONSE frame 202 from the tunnel server. After the key exchange/authentication RESPONSE frame is received at 202, the tunnel initiator enters the Authorized state 203, in which it may send or receive tunnel data frames. Upon receipt of a CLOSE connection frame at 216 in the Authorized state 203, the tunnel initiator transitions to the Dying state 208.

Upon expiration of a session encryption key at 211, the tunnel initiator enters the Reconnect state 212, and sends a CLOSE connection frame at 213 and disconnects the TCP connection with the first node along the tunnel path at 214. Subsequently the tunnel initiator enters the TCP Open state 195.

If during the authorized state 203, a local user issues an End Session command at 204, or there is a detection of an authentication or cryptography error in a received data frame at 205, the tunnel initiator enters the Close state 206. During the Close state 206 the tunnel initiator sends a CLOSE connection frame at 207 to the tunnel server. The tunnel initiator then enters the Dying state at 208.

Figure 12 is a state diagram showing the states within an example embodiment of a tunnel server, for example node D in Fig. 3 or tunnel server 253 in Fig. 14. The tunnel server begins in an Accept Wait state 217. In the Accept Wait state 217, the tunnel server receives a request for a reliable transport layer connection, for example a TCP connection reguest 218 from the last node in the tunnel path prior to the tunnel server, for example Node C in Fig. 3. In response to a TCP connection request 218 the tunnel server accepts the request and establishes a socket interface associated with the resulting TCP connection with Node C.

Upon establishment of the TCP connection with the last node in the tunnel path prior to the tunnel server, the tunnel server enters the Receive Relay state 219. In the Receive Relay state 219, the tunnel server waits to receive a relay frame at 220, at which time the tunnel server enters the Connect Wait state 221. If there is some sort of network error 234 during receipt of the relay frame at 219, the tunnel server enters the Dying state 230. During the Dying state 230 the tunnel server disconnects at 231 the transport layer connection with the last node in the tunnel path prior to the tunnel server. After disconnecting the connection, the tunnel server enters the Dead state 232.

In the Connect Wait state 221, the tunnel server waits for receipt of a key exchange/authentication REQUEST frame at 222. Following receipt of the key exchange/authentication REQUEST frame at 222, the tunnel server determines whether the requested tunnel connection is authorized at step 223. The determination of whether the tunnel connection is authorized is based on a name of the tunnel initiator, and the key exchange material within the key exchange/authentication REQUEST frame.

If the requested tunnel connection is authorized the tunnel server sends a key exchange/authentication RESPONSE frame at 224 back to the tunnel initiator. If the requested tunnel connection is not authorized, the tunnel server enters the Close state 228, in which it sends a close connection frame at 229 to the tunnel client. Following transmission of the CLOSE connection frame at 229, the tunnel server enters the Dying state 230.

If the requested tunnel connection is determined to be authorized at step 223, the tunnel server enters the Authorized state 225. In the Authorized state, the tunnel server transmits and receives tunnel data frames between itself and the tunnel initiator. If during the Authorized state 225, the tunnel server receives a CLOSE connection frame at 233, the tunnel server transitions to the Dying state 230. If during the authorized state 225, the tunnel server receives an end session command from a user at 226, then the tunnel server transitions to the Close state 228, and transmits a close connection frame at 229 to the tunnel initiator. If the tunnel server in the Authorized state 225 detects an integrity failure in a received packet, the tunnel server transitions to the Close state 228. In the close state 228 the tunnel server sends a CLOSE connection frame at 229 and subsequently enters the Dying state 230.

Fig. 13 is a state diagram showing an example embodiment of a state machine within a tunnel relay node. The tunnel relay node begins in an Accept Wait state 235. When a request is received to form a reliable transport layer connection at 236, a reliable transport layer connection is accepted with the requesting node. For example, a TCP connection is accepted between the relay node and the preceding node in the tunnel path.

The relay node then transitions to the Receive Relay state 237. During the Receive Relay state 237, the relay node receives a relay frame at 238. Following receipt of the relay frame at 238, the relay node determines what forwarding address should be used to forward frames received from the TCP connection established responsive to the TCP connect event 236. If the next node in the tunnel path is a tunnel server, the forwarding address may be selected at 239 so as to choose an underutilized tunnel server from a group of available tunnel servers or to choose an operational server where others are not operational.

Following determination of the forwarding address or addresses in step 239, the relay node enters the Forward Connect state 240. In the Forward Connect state 240, the relay node establishes a reliable transport layer connection with the node or nodes indicated by the forwarding address or addresses determined in step 239.

Following establishment of the new connection at event 241, the tunnel relay enters the Forward state 242. During the Forward state 242, the relay node forwards all frames between the connection established at 236 and those connections established at 241. Upon detection of a network error or receipt of a frame indicating a closure of the tunnel connection at 243, the tunnel relay enters the Dying state 244. Following the Dying state 244, the relay node disconnects any connections established at event 241. The relay node then enters the Dead state 246.

Fig. 14 shows an example embodiment of a virtual private network 249 formed by a pseudo network adapter 248 and a tunnel connection between a tunnel client 247 and a tunnel server 253 across a public network 251. The tunnel server 253 and tunnel client 247 are for example network stations including a CPU or microprocessor, memory, and various I/O devices. The tunnel server 253 is shown physically connected to a private LAN 256 including a Network Node 1 257 and a Network Node 2 258, through a physical network adapter 254. The tunnel server 253 is further shown physically connected with a firewall 252 which separates the private LAN 256 from the public network 251. The firewall 252 is physically connected with the public network 251. The tunnel server 253 is further shown including a pseudo network adapter 255. The client system 247 is shown including a physical network adapter 250 physically connected to the public network 251.

During operation of the elements shown in Fig. 14, nodes within the virtual private network 249 appear to the tunnel client 247 as if they were physically connected to the client system through the pseudo network adapter 248. Data transmissions between the tunnel client and any nodes that appear to be within the virtual private network are passed through the pseudo network adapter 248. Data transmissions between the tunnel client 247 and the tunnel server 253 are physically accomplished using a tunnel connection between the tunnel client 247 and the tunnel server 253.

Fig. 15 shows elements in an example embodiment of a pseudo network adapter such as the pseudo network adapter 248 in Fig. 14. In an example embodiment the elements shown in Fig. 15 are implemented as software executing on the tunnel client 247 as shown in Fig. 14. In Fig. 15 there is shown a pseudo network adapter 259 including a virtual adapter driver interface 263, an encapsulation engine 264, an encryption engine 265, a decapsulation engine 268, and a decryption engine 266. Further shown in the pseudo network adapter 259 are an ARP server emulator 270 and a Dynamic Host Configuration Protocol (DHCP) server emulator.

The pseudo network adapter 259 is shown interfaced to a TCP/IP protocol stack 260, through the virtual adapter driver interface 260. The TCP/IP protocol stack 260 is shown interfaced to other services in an operating system 261, as well as a physical network adapter's driver 262. The physical network adapter's driver 262 is for example a device driver which controls the operation of a physical network adapter such as physical network adapter 250 as shown in Fig. 14.

During operation of the elements shown in Fig. 15, the pseudo network adapter 259 registers with the network layer in the TCP/IP stack 260 that it is able to reach the IP addresses of nodes within the virtual private network 249 as shown in Fig. 14. For example, the pseudo network adapter on the client system registers that it can reach the pseudo network adapter on the server. Subsequently, a message from the tunnel client addressed to a node reachable through the virtual private network will be passed by the TCP/IP stack to the pseudo network adapter 259. The pseudo network adapter 259 then encrypts the message, and encapsulates the message into a tunnel data frame. The pseudo network adapter 259 then passes the tunnel data frame back to the TCP/IP protocol stack 260 to be sent through to the physical network adapter in the tunnel server. The tunnel server passes the received data frame to the pseudo network adapter in the server, which de-encapsulates and decrypts the message.

Fig. 16 shows a more detailed example embodiment of a pseudo network adapter 280. The pseudo network adapter 280 includes a virtual network adapter driver interface 288. The transmit path 290 includes an encryption engine 292, and an encapsulation engine 294. The encapsulation engine 294 is interfaced with a TCP/IP transmit interface 312 within a TCP/IP protocol stack, for example a socket interface associated with the first relay node in the tunnel path, or with the remote tunnel end point if the tunnel path includes no relays.

In the example embodiment of Fig. 16, the pseudo network adapter 280 appears to the TCP/IP protocol stack 282 as an Ethernet adapter. Accordingly, ethernet packets 286 for a destination addresses understood by the TCP/IP protocol stack to be reachable through the virtual private network are passed from the TCP/IP protocol stack 282 to the virtual network adapter interface 288 and through the transmit path 290. Similarly, ethernet packets 284 received through the pseudo network adapter 280 are passed from the receive path 296 to the virtual network adapter interface 288 and on to the TCP/IP protocol stack 282.

Further shown in the pseudo network adapter 280 of Fig. 16 is a receive path 296 having a decryption engine 298 interfaced to the virtual network adapter interface 288 and a decapsulation engine 300. The decapsulation engine 300 in turn is interfaced to a TCP/IP receive function 314 in the TCP/IP protocol stack 282, for example a socket interface associated with the first relay in the tunnel path, or with the remote tunnel end point if the tunnel path includes no relays. The pseudo network adapter 280 further includes an ARP server emulator 304 and a DHCP server emulator 306. ARP and DHCP request packets 302 are passed to the ARP server emulator 304 and DHCP server emulator 306 respectively. When a received packet is passed from the receive path 296 to the TCP/IP stack 282, a receive event must be indicated to the TCP/IP stack 282, for example through an interface such the Network Device Interface Specification (NDIS), defined by Microsoft™ Corporation.

Also in Fig. 16 is shown is an operating system 310 coupled with the TCP/IP protocol stack 282. The TCP/IP protocol stack 282 is generally considered to be a component part of the operating system. The operating system 310 in Fig. 16 is accordingly the remaining operating system functions and procedures outside the TCP/IP protocol stack 282. A physical network adapter 308 is further shown operated by the TCP/IP protocol stack 282.

During operation of the elements shown in Fig. 16, a user passes data for transmission to the TCP/IP protocol stack 282, and indicates the IP address of the node to which the message is to be transmitted, for example through a socket interface to the TCP layer. The TCP/IP protocol stack 282 then determines whether the destination node is reachable through the virtual private network. If the message is for a node that is reachable through the virtual private network, the TCP/IP protocol stack 282 an ethernet packet 286 corresponding to the message to the pseudo network adapter 280. The pseudo network adapter 280 then passes the ethernet packet 286 through the transmit path, in which the ethernet packet is encrypted and encapsulated into a tunnel data frame. The tunnel data frame is passed back into the TCP/IP protocol stack 282 through the TCP/IP transmit function 312 to be transmitted to the tunnel server through the tunnel connection. In an example embodiment, a digest value is calculated for the tunnel data frame before encryption within the transmit path within the pseudo network adapter.

Further during operation of the elements shown in Fig. 16, when the TCP/IP protocol stack 282 receives a packet from the remote endpoint of the TCP/IP tunnel connection, for example the tunnel server, the packet is passed to the pseudo network adapter 280 responsive to a TCP receive event. The pseudo network adapter 280 then decapsulates the packet by removing the tunnel header. The pseudo network adapter further decrypts the decapsulated data and passes it back to the TCP/IP protocol stack 282. The data passed from the pseudo network adapter 280 appears to the TCP/IP protocol stack 282 as an ethernet packet received from an actual physical device, and is the data it contains is passed on to the appropriate user by the TCP/IP protocol stack 282 based on information in the ethernet packet header provided by the pseudo network adapter.

Fig. 17 is a flow chart showing steps performed by an example embodiment of a pseudo network adapter during packet transmission, such as in the transmit path 290 of Fig. 14. The TCP/IP protocol stack determines that the destination node of a packet to be transmitted is reachable through the virtual LAN based on the destination IP address of the packet and a network layer routing table. At step 320 the packet is passed to the pseudo network adapter from the TCP/IP protocol stack. As a result, a send routine in the pseudo adapter is triggered for example in the virtual network adapter interface 288 of Fig. 16.

At step 322 the pseudo network adapter send routine processes the Ethernet header of the packet provided by the TCP/IP stack, and removes it. At step 324, the send routine determines whether the packet is an ARP request packet. If the packet is an ARP request packet for an IP address of a node on the virtual LAN, such as the pseudo network adapter of the tunnel server, then step 324 is followed by step 326. Otherwise, step 324 is followed by step 330.

At step 326, the ARP server emulator in the pseudo network adapter generates an ARP reply packet. For example, if the ARP request were for a physical address corresponding to the IP address of the pseudo network adapter on the tunnel server, the ARP reply would indicate a predetermined, reserved physical address to be associated with that IP address. At step 328 the pseudo network adapter passes the ARP response to the virtual network adapter interface. The virtual network adapter interface then indicates a received packet to the TCP/IP protocol stack, for example using an NDIS interface. The TCP/IP protocol stack then processes the ARP response as if it had been received over an actual physical network.

At step 330 the send routine determines whether the packet is a DHCP request packet requesting an IP address for the pseudo network adapter. If so, then step 330 is followed by step 332. Otherwise, step 330 is followed by step 334.

At step 334, the DHCP server emulator in the pseudo network adapter generates a DHCP response. The format of DHCP is generally described in the DHCP RFC. At step 328 the pseudo network adapter passes the DHCP response to the virtual network adapter interface, for example indicating an IP address received from the tunnel server in the client data field of the key exchange/authentication RESPONSE frame. The virtual network adapter interface then indicates a received packet to the TCP/IP protocol stack. The TCP/IP protocol stack then processes the DHCP response as if it had been received over an actual physical network.

At step 334 the pseudo network adapter encrypts the message using an encryption engine such that only the receiver is capable of decrypting and reading the message. At step 336 the pseudo network adapter encapsulates the encrypted message into a tunnel data frame. At step 338 the pseudo network adapter transmits the tunnel data frame through the tunnel connection using the TCP/IP protocol stack.

Fig. 18 is a flow chart showing steps performed by an example embodiment of a pseudo network adapter during packet receipt, such as in the receive path 296 of Fig. 14.

At step 350, the pseudo network adapter is notified that a packet has been received over the tunnel connection. At step 352 the pseudo network adapter decapsulates the received message by removing the header fields of the tunnel data frame. At step 354 the pseudo network adapter decrypts the decapsulated datagram from the tunnel data frame. At step 356, in an example embodiment, the pseudo network adapter forms an Ethernet packet from the decapsulated message. At step 358 the pseudo network adapter indicates that an Ethernet packet has been received to the TCP/IP protocol stack through the virtual network adapter interface. This causes the TCP/IP protocol stack to behave as if it had received an Ethernet packet from an actual Ethernet adapter.

Fig. 19 shows the data flow within the transmit path in an example embodiment of a pseudo network adapter. At step 1 370, an application submits data to be transmitted to the TCP protocol layer 372 within the TCP/IP protocol stack. The application uses a conventional socket interface to the TCP protocol layer 372 to pass the data, and indicates the destination IP address the data is to be transmitted to. The TCP protocol layer 372 then passes the data to the IP protocol layer 374 within the TCP/IP protocol stack. At step 2 376, the TCP/IP protocol stack refers to the routing table 378 to determine which network interface should be used to reach the destination IP address.

Because in the example the destination IP address is of a node reachable through the virtual private network, the IP layer 374 determines from the routing table 378 that the destination IP address is reachable through pseudo network adapter. Accordingly at step 3 380 the TCP/IP protocol stack passes a packet containing the data to the pseudo network adapter 382.

At step 4 384, the pseudo network adapter 382 encrypts the data packets and encapsulates them into tunnel data frames.

The pseudo network adapter 382 then passes the tunnel data frames packets back to the TCP protocol layer 372 within the TCP/IP protocol stack through a conventional socket interface to the tunnel connection with the first node in the tunnel path.

The TCP protocol layer 372 then forms a TCP layer packet for each tunnel data frame, having the tunnel data frame as its data. The TCP frames are passed to the IP layer 374. At step 5 386 the routing table 378 is again searched, and this time the destination IP address is the IP address associated with the physical network adapter on the tunnel server, and accordingly is determined to be reachable over the physical network adapter 390. Accordingly at step 6 388 the device driver 390 for the physical network adapter is called to pass the packets to the physical network adapter. At step 7 392 the physical network adapter transmits the data onto the physical network 394.

Fig. 20 is a data flow diagram showing data flow in an example embodiment of packet receipt involving a pseudo network adapter. At step 1 410 data arrives over the physical network 412 and is received by the physical network adapter and passed to the physical network driver 414. The physical network driver 414 passes the data at step 2 418 through the IP layer 420 and TCP layer 422 to the pseudo network adapter 426 at step 3 424, for example through a conventional socket interface. At step 4 428 the pseudo network adapter 426 decrypts and decapsulates the received data and passes it back to the IP layer of the TCP/IP protocol stack, for example through the TDI (Transport Layer Dependent Interface API) of the TCP/IP stack. The data is then passed through the TCP/IP protocol stack and to the user associated with the destination IP address in the decapsulated datagrams at step 5 430.

Fig. 21 shows data flow in an example embodiment of packet transmission involving a pseudo network adapter. Fig. 21 shows an example embodiment for use on a Microsoft™ Windows 95™ PC platform. In Fig. 21 a user application 450 passes unencrypted data to an interface into the TCP layer of the TCP/IP protocol, for example the WinSock API 452. The user indicates a destination IP address associated with a node reachable through a virtual private network accessible through the pseudo network adapter.

The TCP layer 454 passes the data to the IP layer 456, which in turn passes the data to the Network Device Interface Specification Media Access Control (NDIS MAC) interface 458. The pseudo network adapter 459 has previously registered with the routing layer (IP) that it is able to reach a gateway address associated with the destination IP address for the user data. Accordingly the IP layer uses the NDIS MAC layer interface to invoke the virtual device driver interface 460 to the pseudo network adapter 459. The pseudo network adapter 459 includes a virtual device driver interface 460, an ARP server emulator 462, and a DHCP server emulator 464.

In the example embodiment of Fig. 19, the pseudo network adapter 459 passes the data to a tunnel application program 466. The tunnel application program 466 encrypts the IP packet received from the IP layer and encapsulates it into a tunnel data frame. The tunnel application then passes the tunnel data frame including the encrypted data to the WinSock interface 452, indicating a destination IP address of the remote tunnel end point. The tunnel data frame is then passed through the TCP layer 454, IP layer 456, NDIS MAC layer interface 458, and physical layer 468, and transmitted on the network 470. Since the resulting packets do not contain a destination IP address which the pseudo network adapter has registered to convey, these packets will not be diverted to the pseudo network adapter.

Fig. 22 is a data flow diagram showing data flow in an example embodiment of packet transmission involving a pseudo network adapter. The embodiment shown in Fig. 22 is for use on a UNIX platform. In Fig. 20 a user application 472 passes unencrypted data to a socket interface to the TCP/IP protocol stack in the UNIX socket layer 474, indicating a destination IP address of a node reachable through the virtual private network.

The UNIX socket layer 474 passes the data through the TCP layer 476 and the IP layer 478. The pseudo network adapter 480 has previously registered with the routing layer (IP) that it is able to reach a gateway associated with the destination IP address for the user data. Accordingly the IP layer 478 invokes the virtual device driver interface 482 to the pseudo network adapter 480. The IP layer 478 passes the data to the pseudo network adapter 480. The pseudo network adapter 480 includes a virtual device driver interface 482, and a DHCP server emulator 484.

In the example embodiment of Fig. 22, the pseudo network adapter 480 passes IP datagrams to be transmitted to a UNIX Daemon 486 associated with the tunnel connection. The UNIX Daemon 486 encrypts the IP packet(s) received from the IP layer 478 and encapsulates them into tunnel data frames. The UNIX Daemon 486 then passes the tunnel data frames to the UNIX socket layer 474, through a socket associated with the tunnel connection. The tunnel data frames are then processed by the TCP layer 476, IP layer 478, data link layer 488, and physical layer 490 to be transmitted on the network 492. Since the resulting packets are not addressed to an IP address which the pseudo network adapter 480 has registered to convey, the packets will not be diverted to the pseudo network adapter 480.

Fig. 23 is a flow chart showing steps to initialize a example embodiment of a virtual private network. The steps shown in Fig. 23 are performed for example in the tunnel client 247 as shown in Fig. 14. At step 500 a tunnel application program executing in the tunnel client sends a tunnel relay frame to the tunnel server. At step 502 the tunnel application program sends a tunnel key exchange/authentication REQUEST frame to the tunnel server. The tunnel application in the tunnel server ignores the contents of the client data field in the tunnel key exchange/authentication REQUEST frame. The tunnel application in the tunnel server fills in the client data field in the tunnel key exchange/authentication RESPONSE frame with Dynamic Host Configuration Protocol (DHCP) information, for example including the following information in standard DHCP format:

1) IP Address for tunnel client Pseudo Network Adapter
2) IP Address for tunnel server Pseudo Network Adapter
3) Routes to nodes on the private network physically connected to the tunnel server which are to be reachable over the tunnel connection.

At step 504 the tunnel application receives a tunnel key exchange/authentication RESPONSE frame from the tunnel server. The client data field 508 in the tunnel connection response is made available to the pseudo network adapter in the tunnel client. The tunnel application in the tunnel client tells the TCP/IP stack that the pseudo network adapter in the tunnel client is active. The pseudo network adapter in the tunnel client is active and ready to be initialized at step 510.

The tunnel client system is configured such that it must obtain an IP address for the tunnel client pseudo network adapter dynamically. Therefore the TCP/IP stack in the tunnel client broadcasts a DHCP request packet through the pseudo network adapter. Accordingly, at step 512 the pseudo network adapter in the client receives a conventional DHCP request packet from the TCP/IP stack requesting a dynamically allocated IP address to associate with the pseudo network adapter. The pseudo network adapter passes the DHCP request packet to the DHCP server emulator within the pseudo network adapter, which forms a DHCP response based on the client data 508 received from the tunnel application. The DHCP response includes the IP address for the client pseudo adapter provided by the tunnel server in the client data. At step 514 the pseudo network adapter passes the DHCP response to the TCP/IP stack.

At step 520, the tunnel application modifies the routing tables within the tunnel client TCP/IP stack to indicate that the routes to the nodes attached to the private network to which the tunnel server is attached all are reachable only through the pseudo network adapter in the tunnel server. The IP address of the pseudo network adapter in the tunnel server provided in the client data is in this way specified as a gateway to the nodes on the private network to which the tunnel server is attached. In this way those remote nodes are viewed by the TCP/IP stack as being reachable via the virtual private network through the client pseudo network adapter.

At step 516 the pseudo network adapter in the tunnel client receives an ARP request for a physical address associated with the IP address of the pseudo network adapter in the tunnel server. The pseudo network adapter passes the ARP request to the ARP server emulator, which forms an ARP reply indicating a reserved physical address to be associated with the IP address of the pseudo network adapter in the tunnel server. At step 518 the pseudo network adapter passes the ARP response to the TCP/IP stack in the tunnel client. In response to the ARP response, the TCP/IP stack determines that packets addressed to any node on the virtual private network must be initially transmitted through the pseudo network adapter.

In an example embodiment the present system reserves two physical addresses to be associated with the pseudo network adapter in the client and the pseudo network adapter in the server respectively. These reserved physical addresses are used in responses to ARP requests passed through the pseudo network adapter for physical addresses corresponding to the IP addresses for the pseudo network adapter in the client and the pseudo network adapter in the server respectively. The reserved physical addresses should have a high likelihood of not being used in any actual network interface.

While the invention has been described with reference to specific example embodiments, the description is not meant to be construed in a limiting sense. Various modifications of the disclosed embodiments, as well as other embodiments of the invention, will be apparent to persons skilled in the art upon reference to this description. Specifically, while various embodiments have been described using the TCP/IP protocol stack, the invention may advantageously be applied where other communications protocols are used. Also, while various flow charts have shown steps performed in an example order, various implementations may use altered orders of step in order to apply the invention. And further, while certain specific software and/or hardware platforms have been used in the description, the invention may be applied on other platforms with similar advantage. It is therefore contemplated that the appended claims will cover any such modifications or embodiments which fall within the scope of the invention.

Claims

1. A pseudo network adapter providing a virtual private network, comprising:

an interface for capturing packets from a local communications protocol stack for transmission on said virtual private network, said interface appearing to said local communications protocol stack as a network adapter device driver for a network adapter connected to said virtual private network;
a first server emulator, providing a first reply packet responsive to a first request packet captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network, said first request packet requesting a network layer address for said pseudo network adapter, said first reply indicating a network layer address for said pseudo network adapter; and
a second server emulator, providing a second reply packet responsive to an second request packet captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network, said second request packet requesting a physical address corresponding to a network layer address of a second pseudo network adapter, said second pseudo network adapter located on a remote server node, said second reply indicating a predetermined, reserved physical address.

2. The pseudo network adapter of claim 1, further comprising a means for indicating to said local communications protocol stack that said predetermined, reserved physical address is reachable through said pseudo network adapter, wherein said means for indicating modifies a data structure in said local communications protocol stack indicating which nodes or networks are reachable through each network interface of the local system.

3. The pseudo network adapter of claim 1, further comprising a means for indicating to said local communications protocol stack that one or more nodes on a remote private network connected to said remote server node are reachable through a gateway node equal to said second pseudo network adapter on said remote server node.

4. The pseudo network adapter of claim 1, further comprising:

a transmit path for processing data packets captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network;
an encryption engine, within said transmit path, for encrypting said data packets;
an encapsulation engine, within said transmit path, for encapsulating said encrypted data packets into tunnel data frames; and
a means for passing said tunnel data frames back to said local communications protocol stack for transmission to a physical network adapter on said remote server node.

5. The pseudo network adapter of claim 4, wherein said transmit path further includes means for storing a digest value in a digest field in each of said tunnel data frames, said digest value equal to an output of a keyed hash function applied to said data packet encapsulated within said tunnel data frame concatenated with a counter value equal to a total number of tunnel data frames previously transmitted to said remote server node.

6. The pseudo network adapter of claim 4, wherein said transmit path further includes means for processing an Ethernet header in each one of said captured data packets, said processing of said Ethernet header including removing said Ethernet header.

7. The pseudo network adapter of claim 1, further comprising:

an interface into a transport layer of said local communications protocol stack for capturing received data packets from said remote server node.

8. The pseudo network adapter of claim 7, further comprising:

a receive path for processing received data packets captured by said interface into said transport layer of said local communications protocol stack for capturing received data packets from said remote server node;
an decapsulation engine, within said receive path, for decapsulating said received data packets by removing a tunnel frame header;
an decryption engine, within said receive path, for decrypting said received data packets; and
a means for passing said received data packets back to said local communications protocol stack for delivery to a user.

9. A method for providing a pseudo network adapter for a virtual private network, comprising the steps of:

capturing packets from a local communications protocol stack for transmission on said virtual private network, said capturing through an interface appearing to said local communications stack as a network adapter device driver for a network adapter connected to said virtual private network;
issuing a first reply packet responsive to a first request packet captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network, said first request packet requesting a network layer address for said pseudo network adapter, said first reply indicating a network layer address for said pseudo network adapter; and
issuing a second reply packet responsive to a second request packet captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network, said second request packet requesting a physical address corresponding to a network layer address of a second pseudo network adapter, said second pseudo network adapter located on a remote server node, said ARP Reply indicating a predetermined, reserved physical address.

10. The method of claim 9, further comprising indicating to said local communications protocol stack that said predetermined, reserved physical address is reachable through said pseudo network adapter, wherein said step of indicating to said local communications protocol stack modifies a data structure in said local communications protocol stack indicating which nodes or networks are reachable through each network interface of the local system.

11. The method of claim 9, further comprising indicating to said local communications protocol stack that one or more nodes on a remote private network connected to said remote server node are reachable through a gateway node equal to said second pseudo network adapter on said remote server node, wherein said step of indicating to said local communications protocol stack that one or more nodes on said remote private network connected to said remote server node are reachable through a gateway node equal to said second pseudo network adapter on said remote server node modifies a network layer routing table in said local communications protocol stack.

12. The method of claim 9, further comprising:

processing data packets captured by said interface for capturing packets from said local communications protocol stack for transmission on said virtual private network in a transmit data path;

encrypting said data packets in an encryption engine, within said transmit path;

encapsulating said encrypted data packets into tunnel data frames by an encapsulation engine, within said transmit path; and

passing said tunnel data frames back to said local communications protocol stack for transmission to a physical network adapter on said remote server node, wherein said transmit path further includes storing a digest value in a digest field in each of said tunnel data frames, said digest value equal to an output of a keyed hash function applied to said data packet encapsulated within said tunnel data frame concatenated with a counter value equal to a total number of tunnel data frames previously transmitted to said remote server node.

13. The method of claim 12, wherein said transmit path further includes processing an Ethernet header in each one of said captured data packets, said processing of said Ethernet header including removing said Ethernet header.

14. The method of claim 9, further comprising capturing received data packets from said remote server node through an interface into a transport layer of said local communications protocol stack, further comprising:

processing received data packets captured by said interface into said transport layer of said local communications protocol stack for capturing received data packets from said remote server node in a receive path;

decapsulating said received data packets by removing a tunnel frame header in an decapsulation engine, within said receive path;

decrypting said received data packets in a decryption engine within said receive path; and

passing said received data frames packets back to said local communications protocol stack for delivery to a user.

15. The method of claim 9, wherein said network layer address for said pseudo network adapter and said predetermined, reserved physical address is communicated to said pseudo network adapter from said remote server node as client data in a connection response frame.

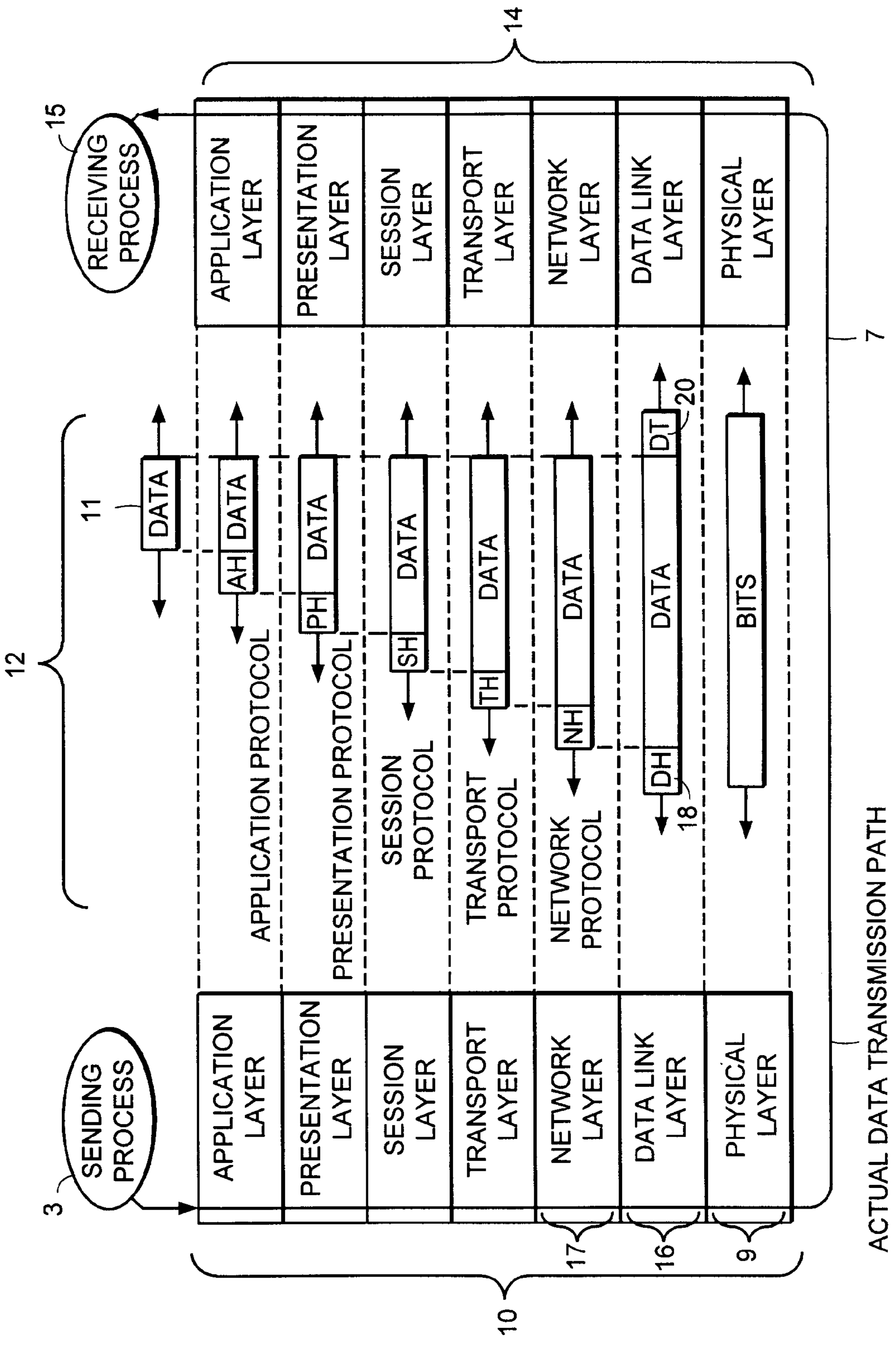

3
SENDING PROCESS
15
RECEIVING PROCESS
10
14
12
11
DATA
APPLICATION LAYER
PRESENTATION LAYER
SESSION LAYER
TRANSPORT LAYER
NETWORK LAYER
DATA LINK LAYER
PHYSICAL LAYER
APPLICATION PROTOCOL
PRESENTATION PROTOCOL
SESSION PROTOCOL
TRANSPORT PROTOCOL
NETWORK PROTOCOL
AH
PH
SH
TH
NH
DH
DT
BITS
18
20
17
16
9
7
ACTUAL DATA TRANSMISSION PATH

FIG. 1

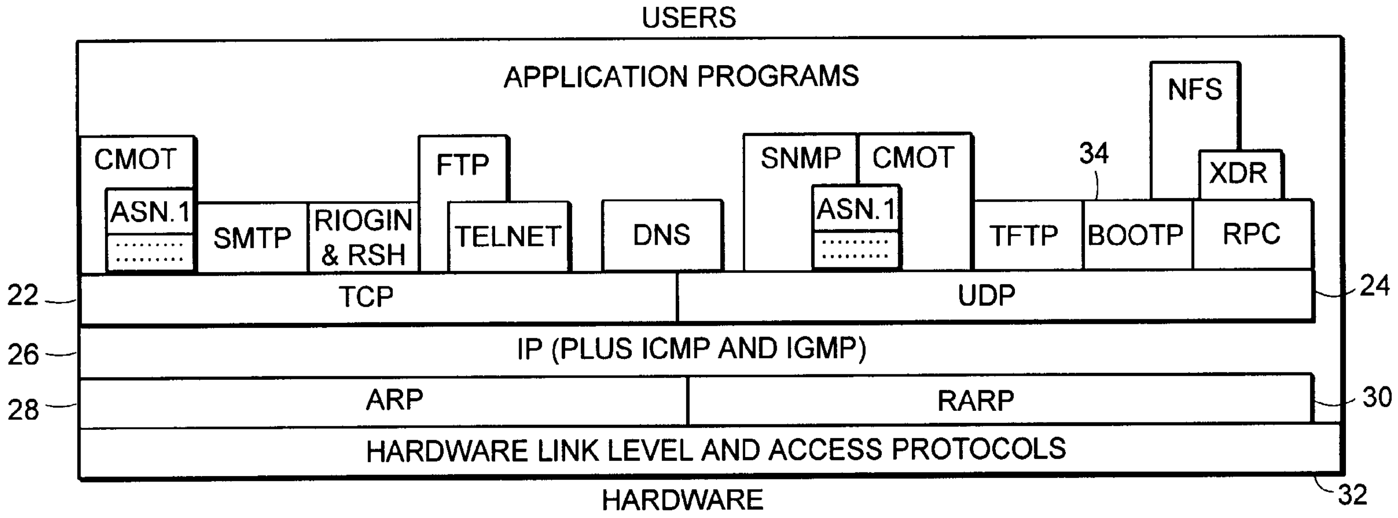

USERS
APPLICATION PROGRAMS
CMOT
ASN.1
SMTP
RIOGIN & RSH
FTP
TELNET
DNS
SNMP
CMOT
ASN.1
34
TFTP
BOOTP
NFS
XDR
RPC
22
TCP
UDP
24
26
IP (PLUS ICMP AND IGMP)
28
ARP
RARP
30
HARDWARE LINK LEVEL AND ACCESS PROTOCOLS
HARDWARE
32

FIG. 2

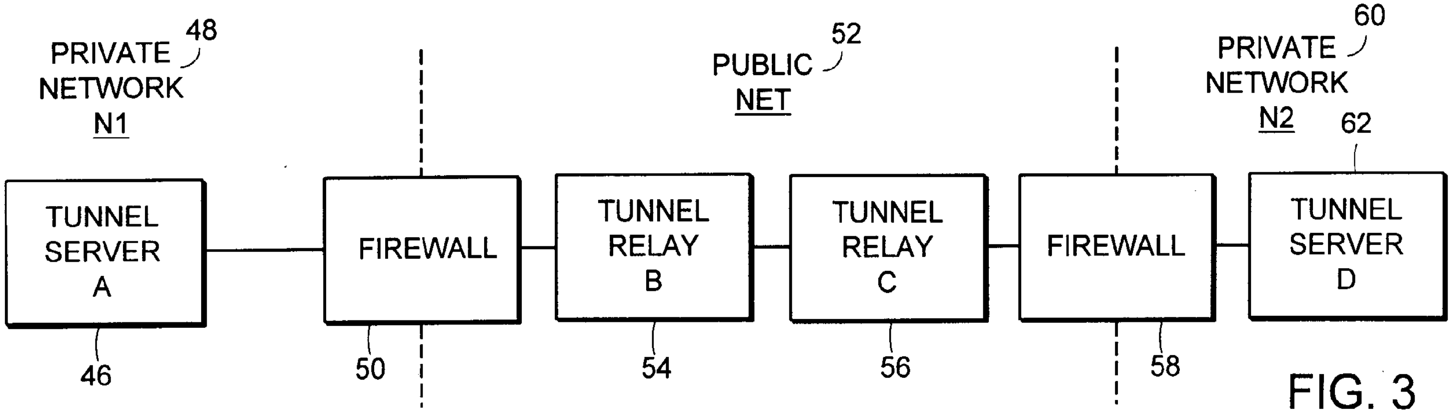

PRIVATE NETWORK N1
48
PUBLIC NET
52
PRIVATE NETWORK N2
60
62
TUNNEL SERVER A
FIREWALL
TUNNEL RELAY B
TUNNEL RELAY C
FIREWALL
TUNNEL SERVER D
46
50
54
56
58

FIG. 3

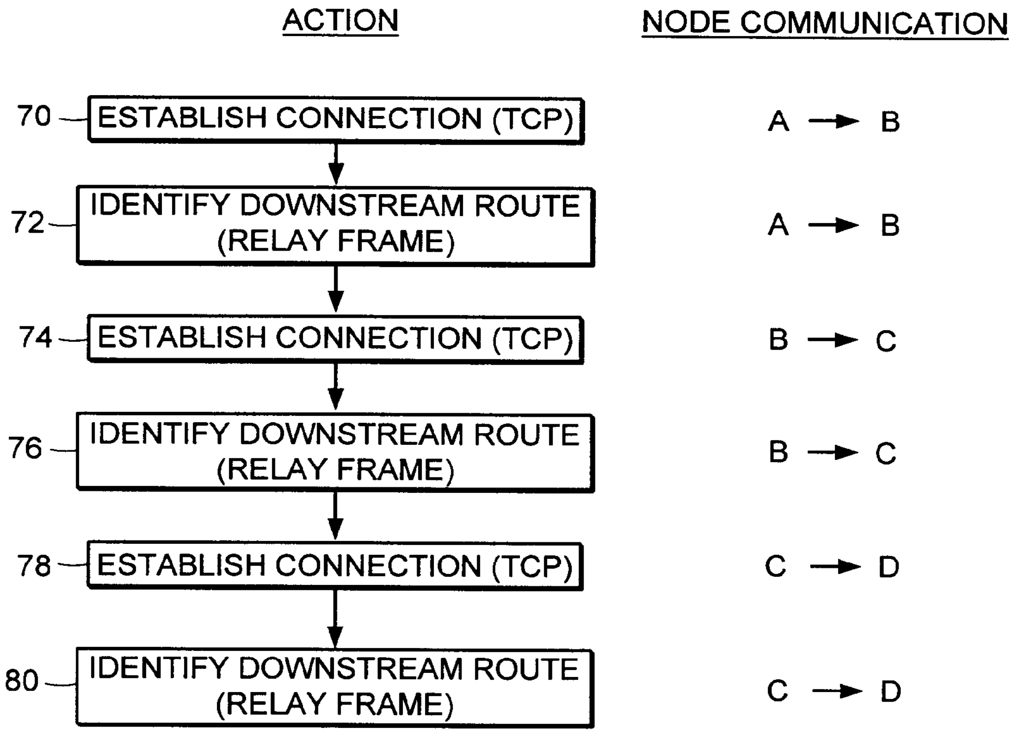

ACTION
NODE COMMUNICATION
70
ESTABLISH CONNECTION (TCP)
A → B
72
IDENTIFY DOWNSTREAM ROUTE (RELAY FRAME)
A → B
74
ESTABLISH CONNECTION (TCP)
B → C
76
IDENTIFY DOWNSTREAM ROUTE (RELAY FRAME)
B → C
78
ESTABLISH CONNECTION (TCP)
C → D
80
IDENTIFY DOWNSTREAM ROUTE (RELAY FRAME)
C → D

FIG. 4

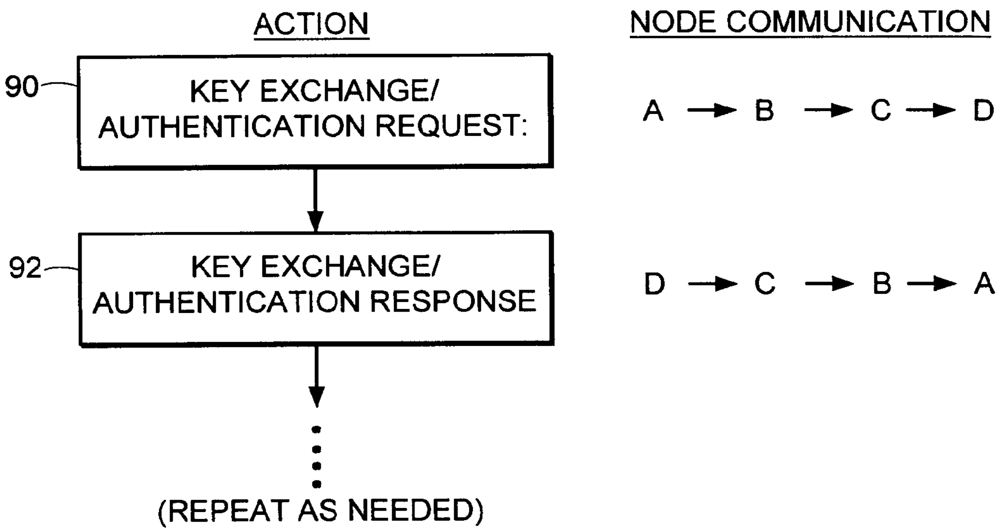

ACTION
NODE COMMUNICATION
90
KEY EXCHANGE/ AUTHENTICATION REQUEST:
A → B → C → D
92
KEY EXCHANGE/ AUTHENTICATION RESPONSE
D → C → B → A
(REPEAT AS NEEDED)

FIG. 5

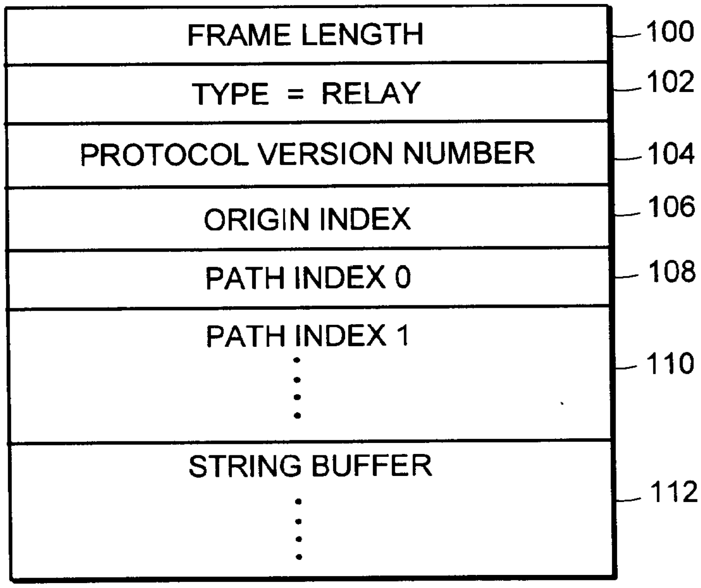

FRAME LENGTH
100
TYPE = RELAY
102
PROTOCOL VERSION NUMBER
104
ORIGIN INDEX
106
PATH INDEX 0
108
PATH INDEX 1
110
STRING BUFFER
112

FIG. 6

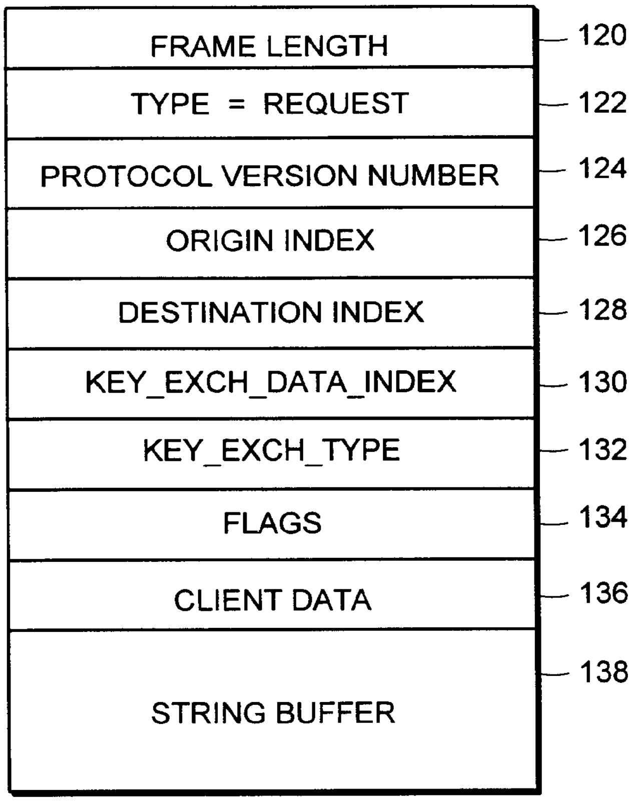

FRAME LENGTH
120
TYPE = REQUEST
122
PROTOCOL VERSION NUMBER
124
ORIGIN INDEX
126
DESTINATION INDEX
128
KEY_EXCH_DATA_INDEX
130
KEY_EXCH_TYPE
132
FLAGS
134
CLIENT DATA
136
STRING BUFFER
138

FIG. 7

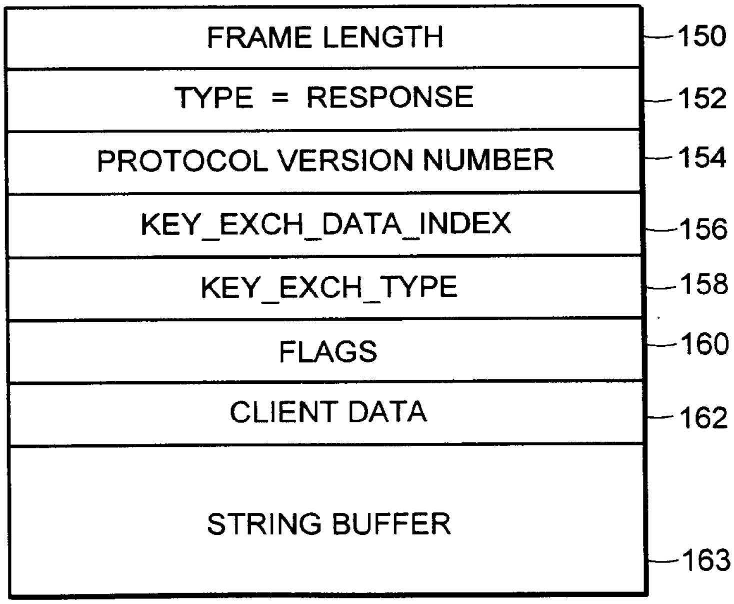

FRAME LENGTH
150
TYPE = RESPONSE
152
PROTOCOL VERSION NUMBER
154
KEY_EXCH_DATA_INDEX
156
KEY_EXCH_TYPE
158
FLAGS
160
CLIENT DATA
162
STRING BUFFER
163

FIG. 8

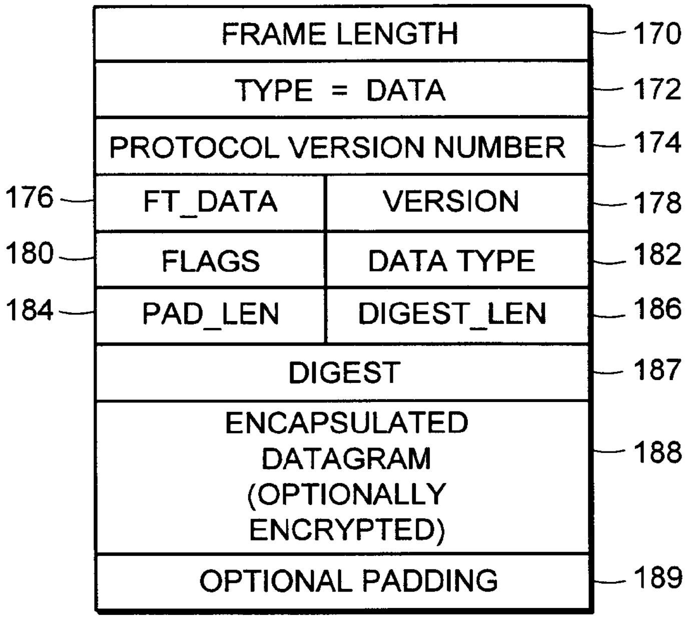

FRAME LENGTH
170
TYPE = DATA
172
PROTOCOL VERSION NUMBER
174
176
FT_DATA
VERSION
178
180
FLAGS
DATA TYPE
182
184
PAD_LEN
DIGEST_LEN
186
DIGEST
187
ENCAPSULATED DATAGRAM (OPTIONALLY ENCRYPTED)
188
OPTIONAL PADDING
189

FIG. 9

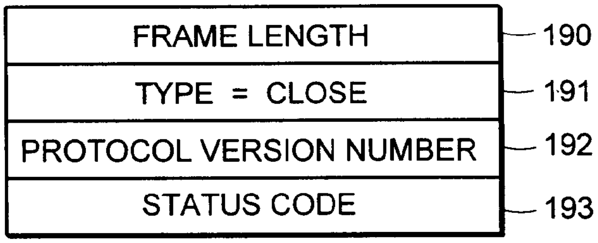

FRAME LENGTH
190
TYPE = CLOSE
191
PROTOCOL VERSION NUMBER
192
STATUS CODE
193

FIG. 10

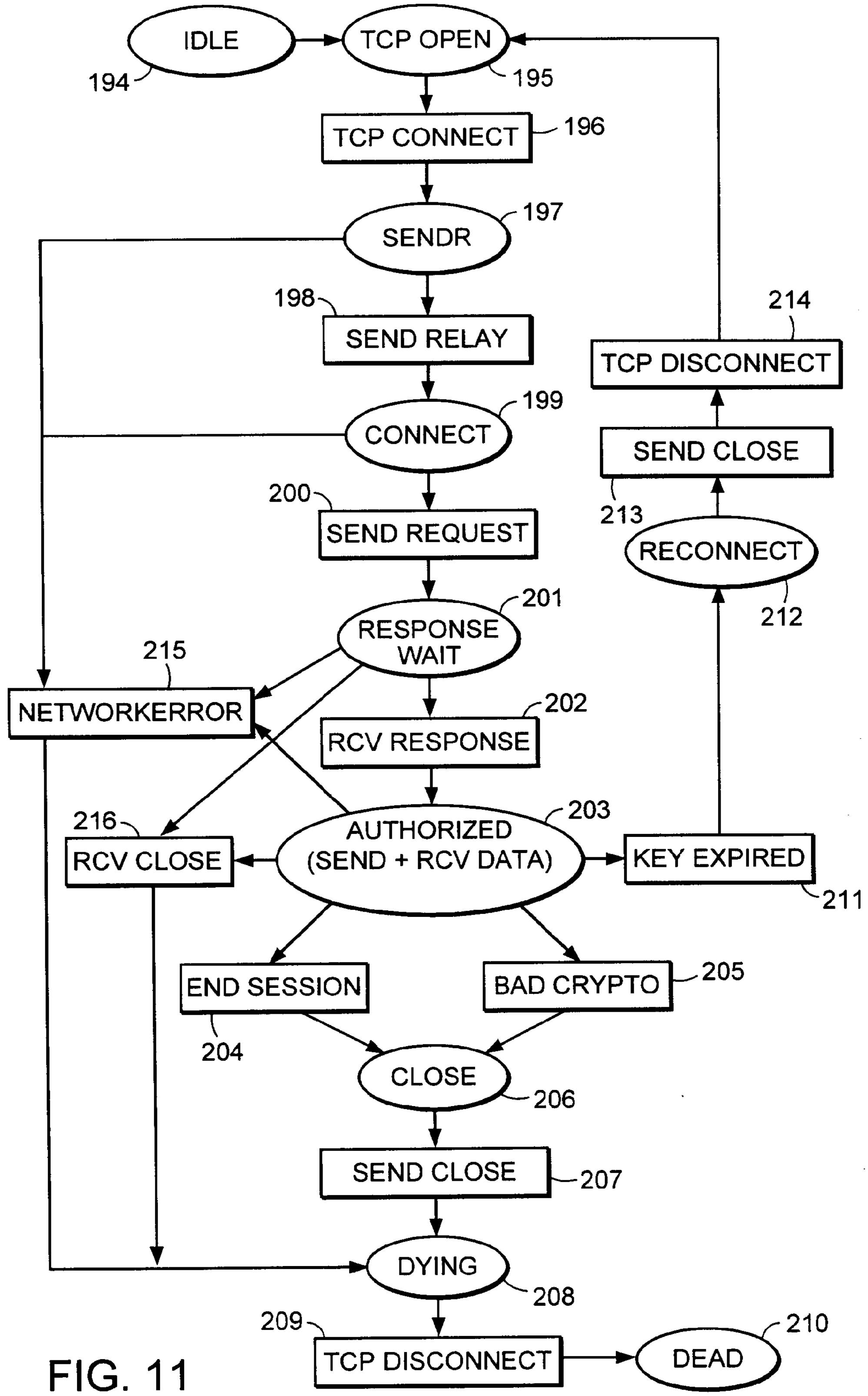

IDLE
194
TCP OPEN
195
TCP CONNECT
196
SENDR
197
198
SEND RELAY
CONNECT
199
200
SEND REQUEST
RESPONSE WAIT
201
215
NETWORKERROR
RCV RESPONSE
202
216
RCV CLOSE
AUTHORIZED (SEND + RCV DATA)
203
KEY EXPIRED
211
END SESSION
204
BAD CRYPTO
205
CLOSE
206
SEND CLOSE
207
DYING
208
209
TCP DISCONNECT
DEAD
210
RECONNECT
212
SEND CLOSE
213
TCP DISCONNECT
214

FIG. 11

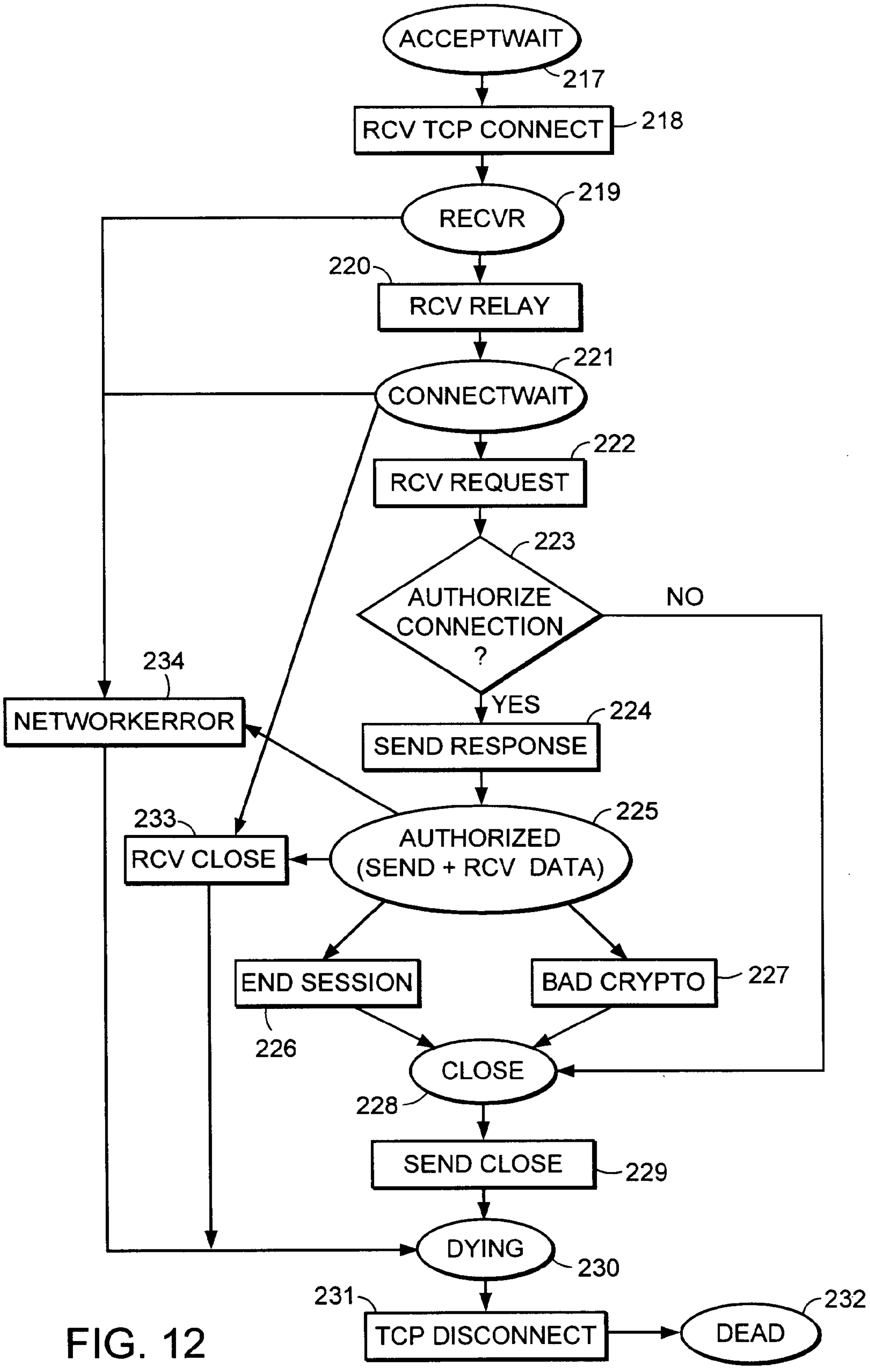

ACCEPTWAIT
217
RCV TCP CONNECT
218
RECVR
219
220
RCV RELAY
CONNECTWAIT
221
RCV REQUEST
222
223
AUTHORIZE CONNECTION ?
NO
YES
234
NETWORKERROR
SEND RESPONSE
224
233
RCV CLOSE
AUTHORIZED (SEND + RCV DATA)
225
END SESSION
BAD CRYPTO
227
226
CLOSE
228
SEND CLOSE
229
DYING
230
231
TCP DISCONNECT
DEAD
232

FIG. 12

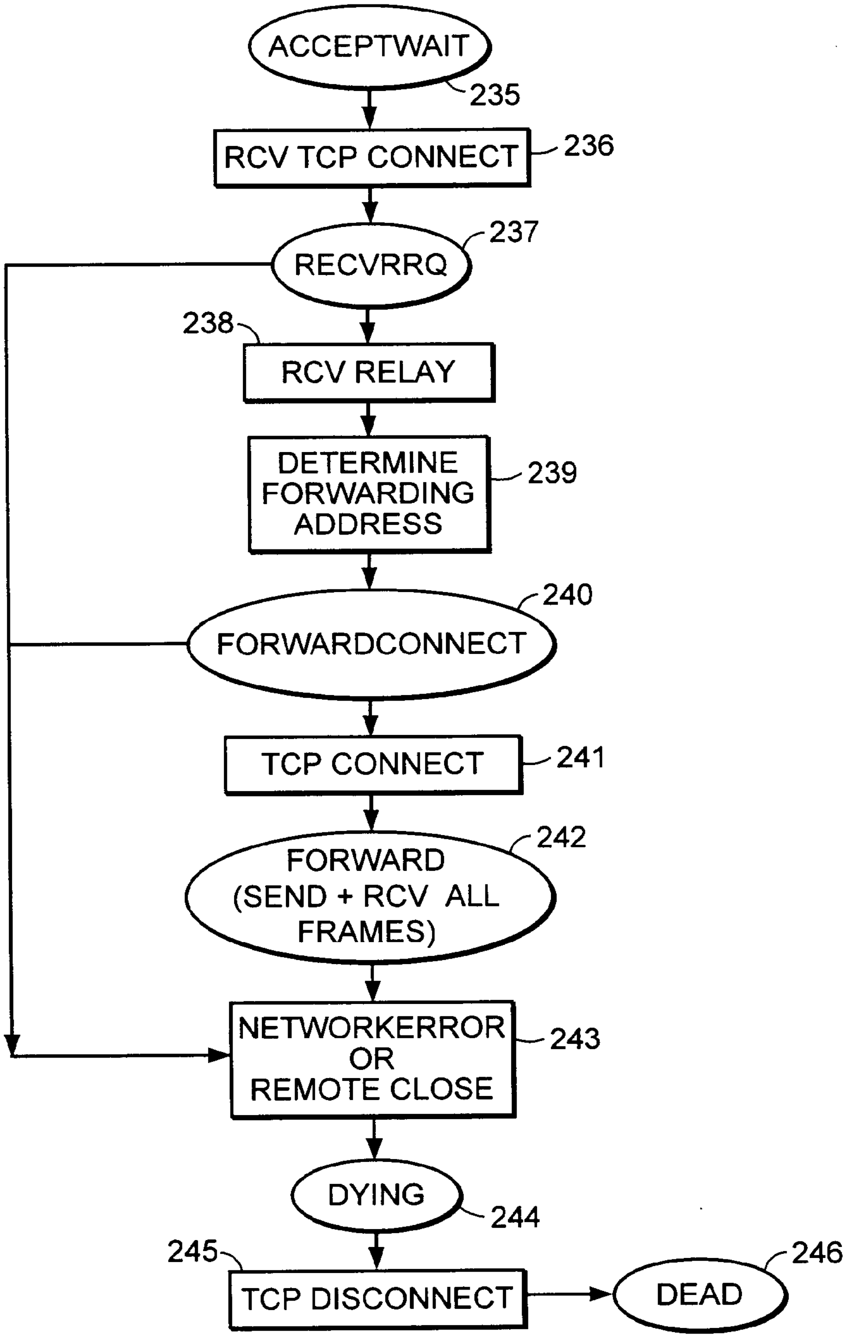

ACCEPTWAIT
235
RCV TCP CONNECT
236
RECVRRQ
237
238
RCV RELAY
DETERMINE FORWARDING ADDRESS
239
FORWARDCONNECT
240
TCP CONNECT
241
FORWARD (SEND + RCV ALL FRAMES)
242
NETWORKERROR OR REMOTE CLOSE
243
DYING
244
245
TCP DISCONNECT
DEAD
246

FIG. 13

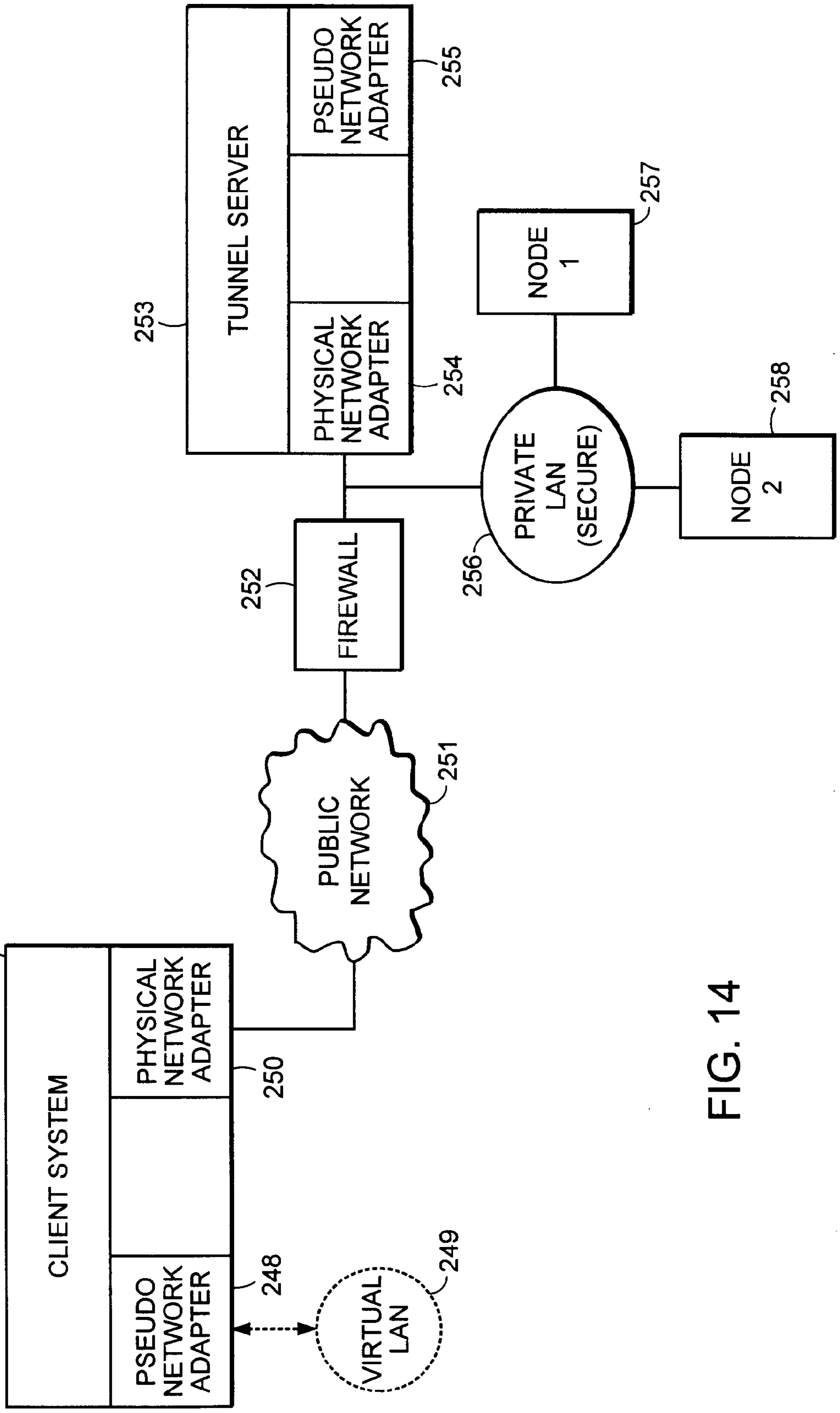

CLIENT SYSTEM
247
PSEUDO NETWORK ADAPTER
248
VIRTUAL LAN
249
PHYSICAL NETWORK ADAPTER
250
PUBLIC NETWORK
251
FIREWALL
252
TUNNEL SERVER
253
PHYSICAL NETWORK ADAPTER
254
PSEUDO NETWORK ADAPTER
255
PRIVATE LAN (SECURE)
256
NODE 1
257
NODE 2
258

FIG. 14

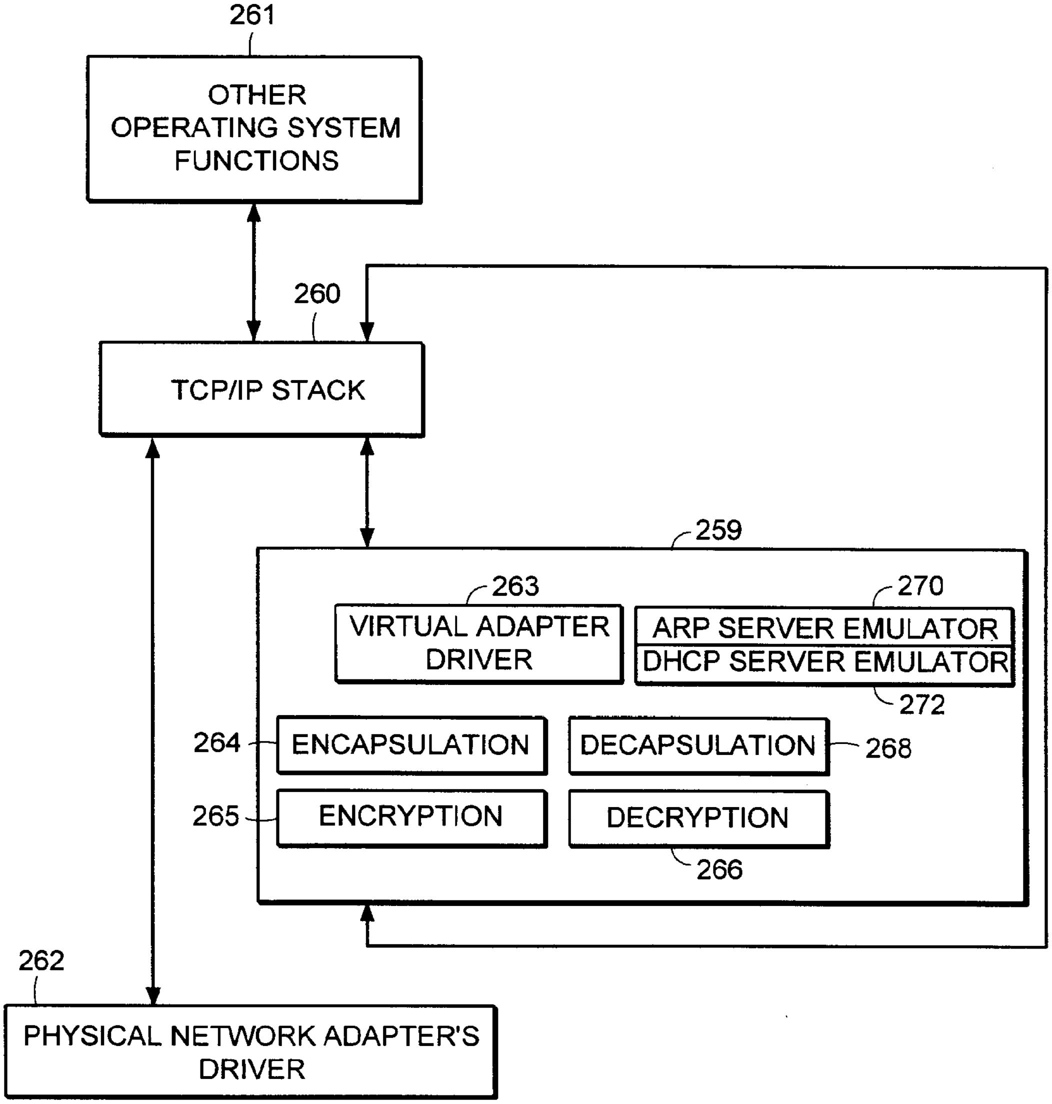

261
OTHER OPERATING SYSTEM FUNCTIONS
260
TCP/IP STACK
259
263
VIRTUAL ADAPTER DRIVER
270
ARP SERVER EMULATOR
DHCP SERVER EMULATOR
272
264
ENCAPSULATION
DECAPSULATION
268
265
ENCRYPTION
DECRYPTION
266
262
PHYSICAL NETWORK ADAPTER'S DRIVER

FIG. 15

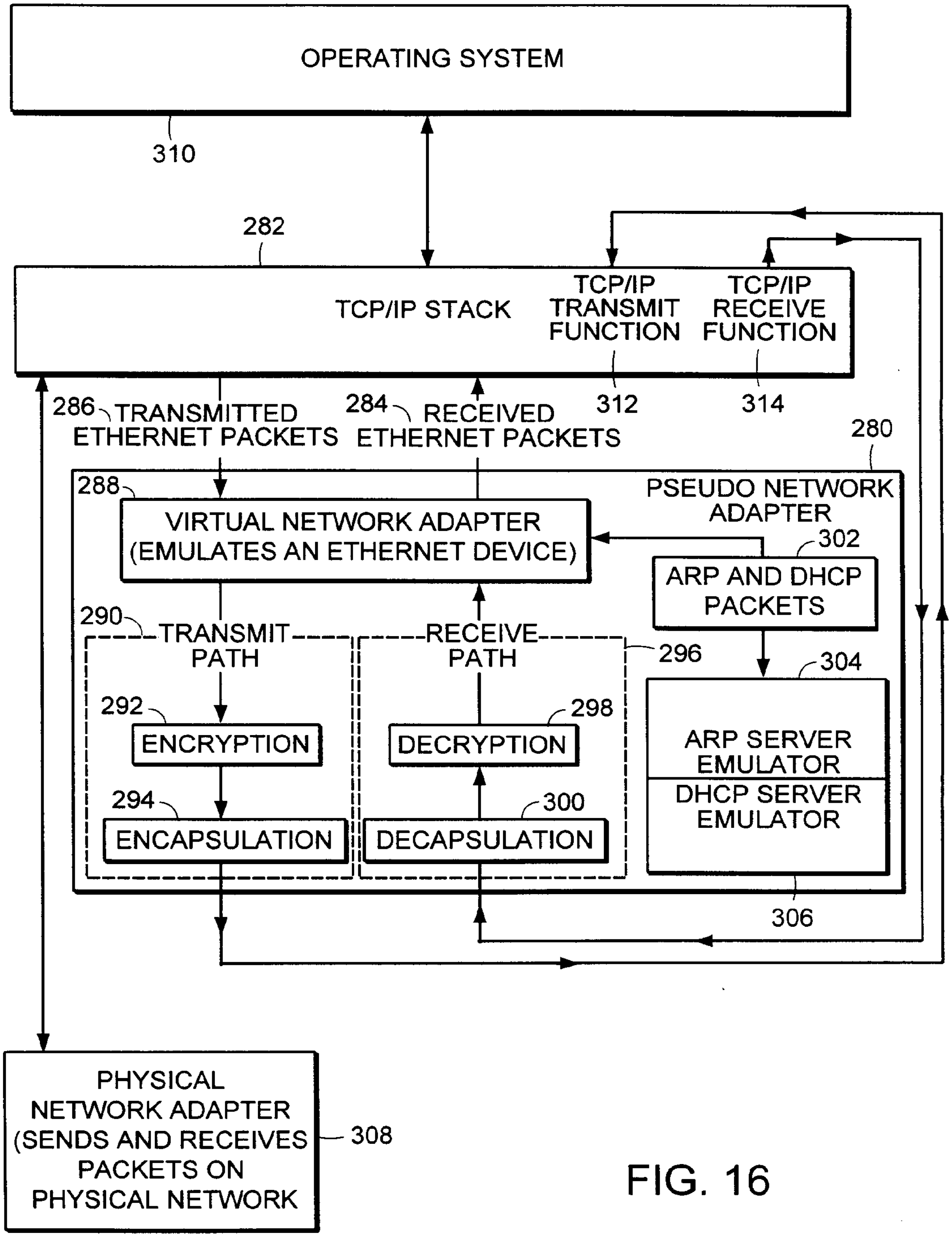

OPERATING SYSTEM
310
282
TCP/IP STACK
TCP/IP TRANSMIT FUNCTION
TCP/IP RECEIVE FUNCTION
312
314
286 TRANSMITTED ETHERNET PACKETS
284 RECEIVED ETHERNET PACKETS
280
PSEUDO NETWORK ADAPTER
288
VIRTUAL NETWORK ADAPTER (EMULATES AN ETHERNET DEVICE)
302
ARP AND DHCP PACKETS
290 TRANSMIT PATH
RECEIVE PATH
296
304
292
ENCRYPTION
298
DECRYPTION
ARP SERVER EMULATOR
DHCP SERVER EMULATOR
294
ENCAPSULATION
300
DECAPSULATION
306
PHYSICAL NETWORK ADAPTER (SENDS AND RECEIVES PACKETS ON PHYSICAL NETWORK
308

FIG. 16

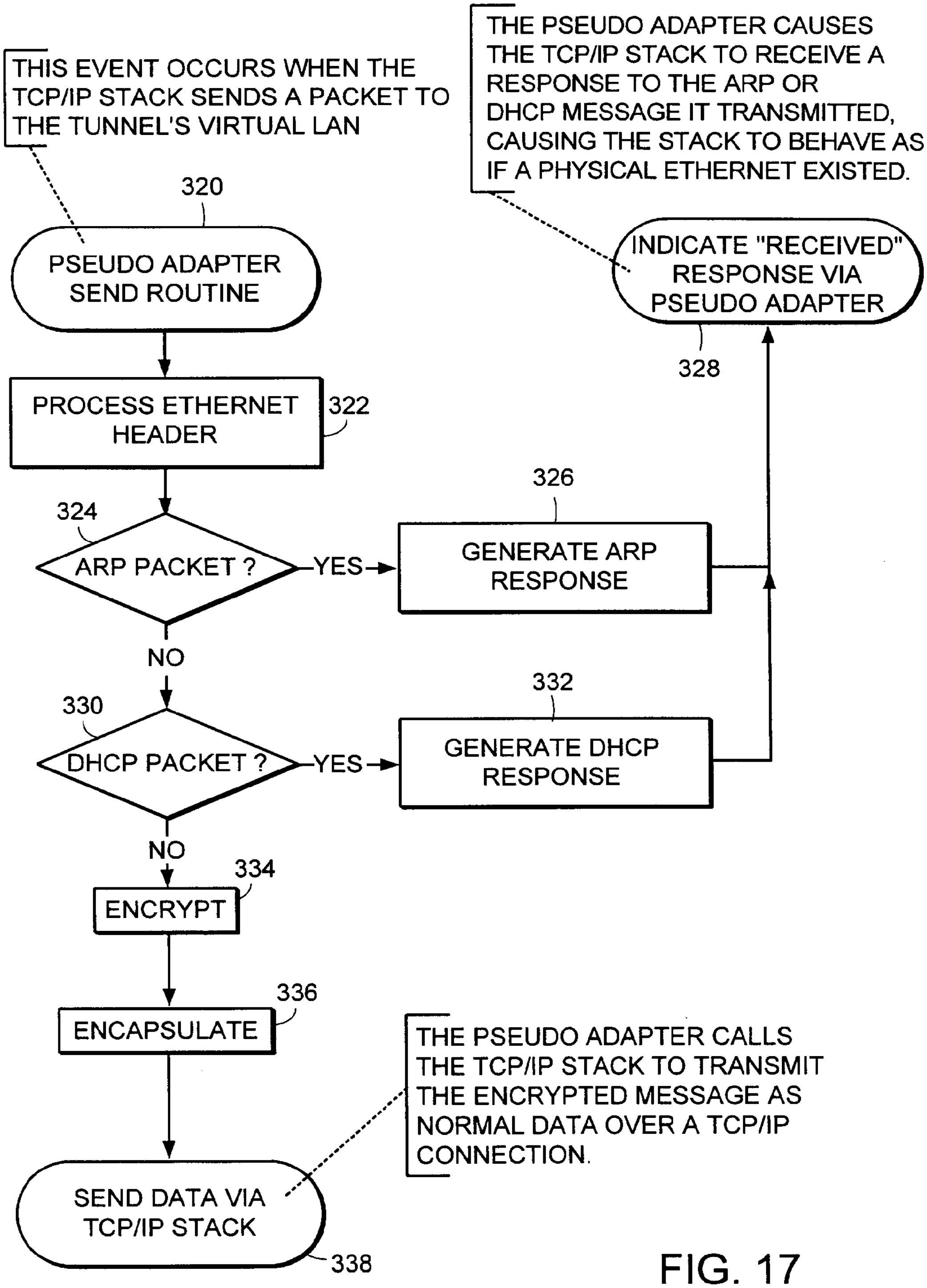

THE PSEUDO ADAPTER CAUSES THE TCP/IP STACK TO RECEIVE A RESPONSE TO THE ARP OR DHCP MESSAGE IT TRANSMITTED, CAUSING THE STACK TO BEHAVE AS IF A PHYSICAL ETHERNET EXISTED.
THIS EVENT OCCURS WHEN THE TCP/IP STACK SENDS A PACKET TO THE TUNNEL'S VIRTUAL LAN
320
PSEUDO ADAPTER SEND ROUTINE
INDICATE "RECEIVED" RESPONSE VIA PSEUDO ADAPTER
328
PROCESS ETHERNET HEADER
322
326
324
ARP PACKET ?
YES
GENERATE ARP RESPONSE
NO
332
330
DHCP PACKET ?
YES
GENERATE DHCP RESPONSE
NO
334
ENCRYPT
336
ENCAPSULATE
THE PSEUDO ADAPTER CALLS THE TCP/IP STACK TO TRANSMIT THE ENCRYPTED MESSAGE AS NORMAL DATA OVER A TCP/IP CONNECTION.
SEND DATA VIA TCP/IP STACK
338

FIG. 17

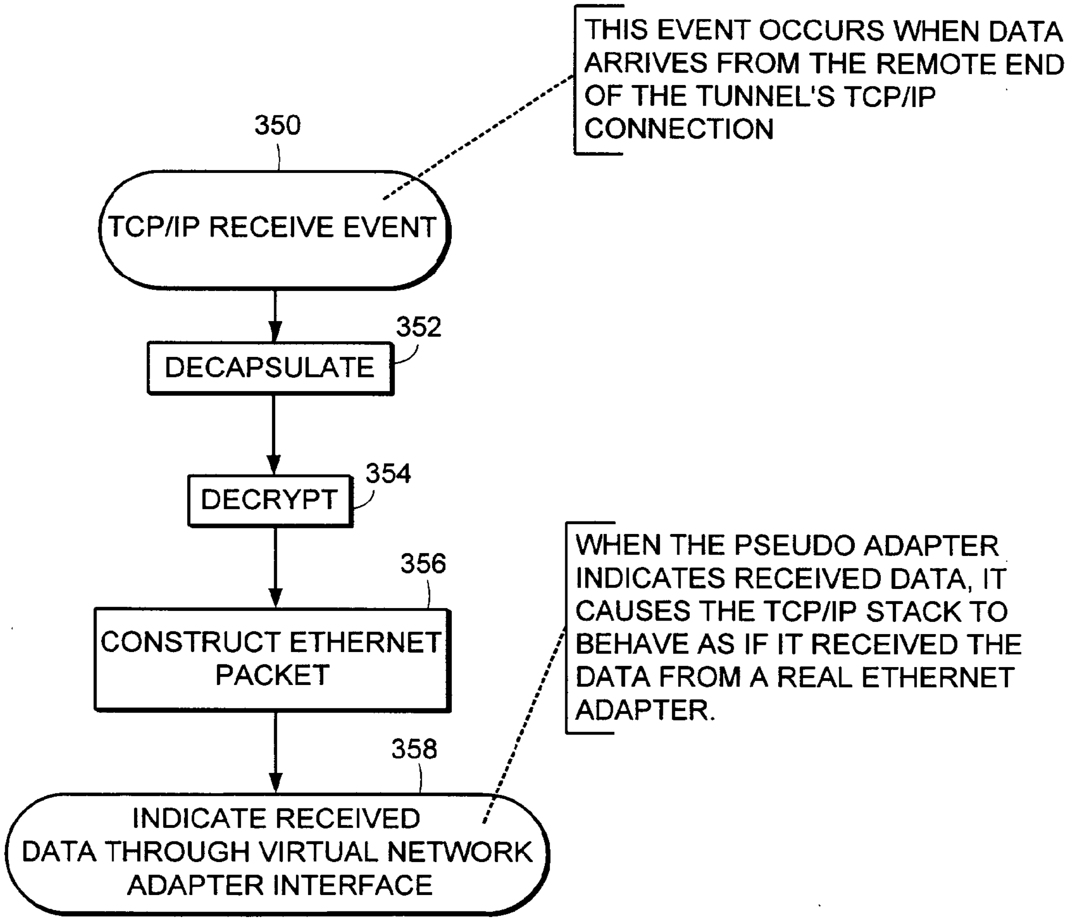

THIS EVENT OCCURS WHEN DATA ARRIVES FROM THE REMOTE END OF THE TUNNEL'S TCP/IP CONNECTION
350
TCP/IP RECEIVE EVENT
352
DECAPSULATE
354
DECRYPT
356
CONSTRUCT ETHERNET PACKET
WHEN THE PSEUDO ADAPTER INDICATES RECEIVED DATA, IT CAUSES THE TCP/IP STACK TO BEHAVE AS IF IT RECEIVED THE DATA FROM A REAL ETHERNET ADAPTER.
358
INDICATE RECEIVED DATA THROUGH VIRTUAL NETWORK ADAPTER INTERFACE

FIG. 18

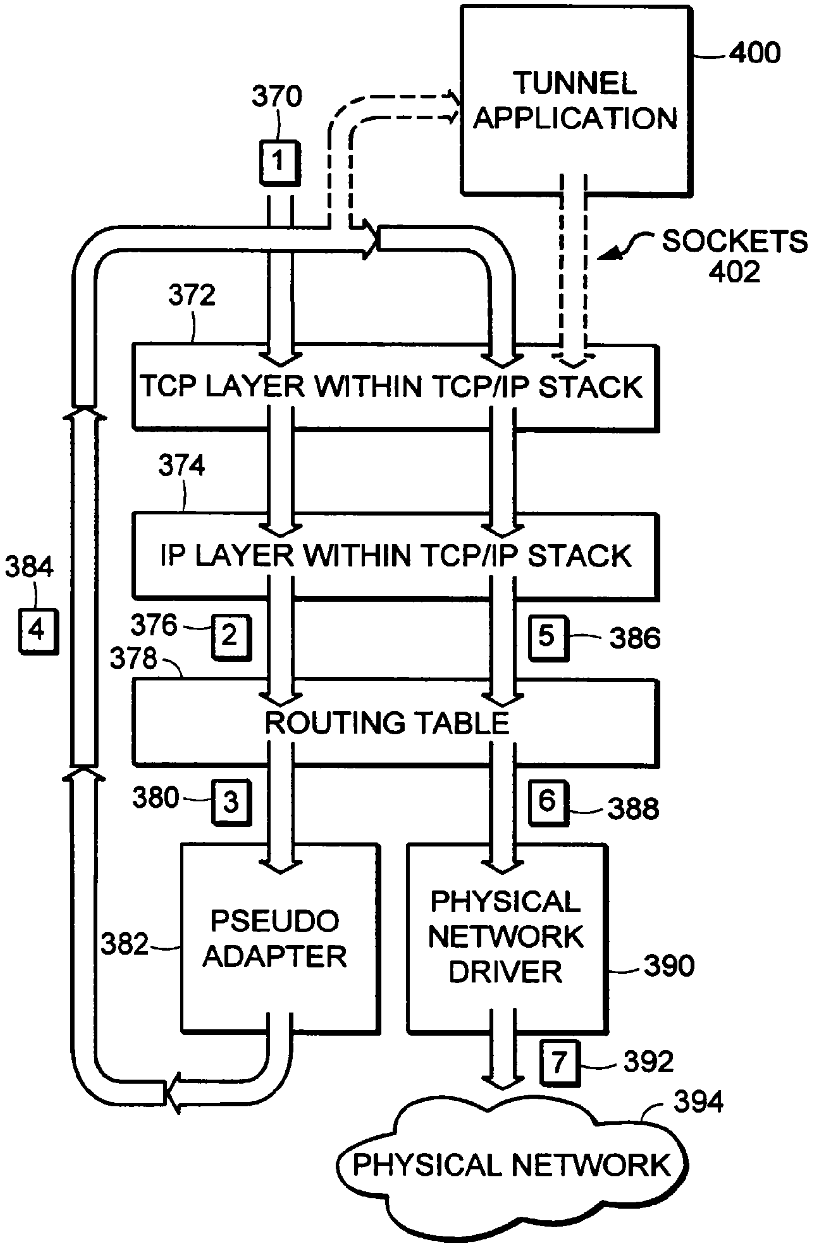

TUNNEL APPLICATION
400
370
1
SOCKETS 402
372
TCP LAYER WITHIN TCP/IP STACK
374
IP LAYER WITHIN TCP/IP STACK
384
4
376
2
378
5
386
ROUTING TABLE
380
3
6
388
382
PSEUDO ADAPTER
PHYSICAL NETWORK DRIVER
390
7
392
394
PHYSICAL NETWORK

FIG. 19

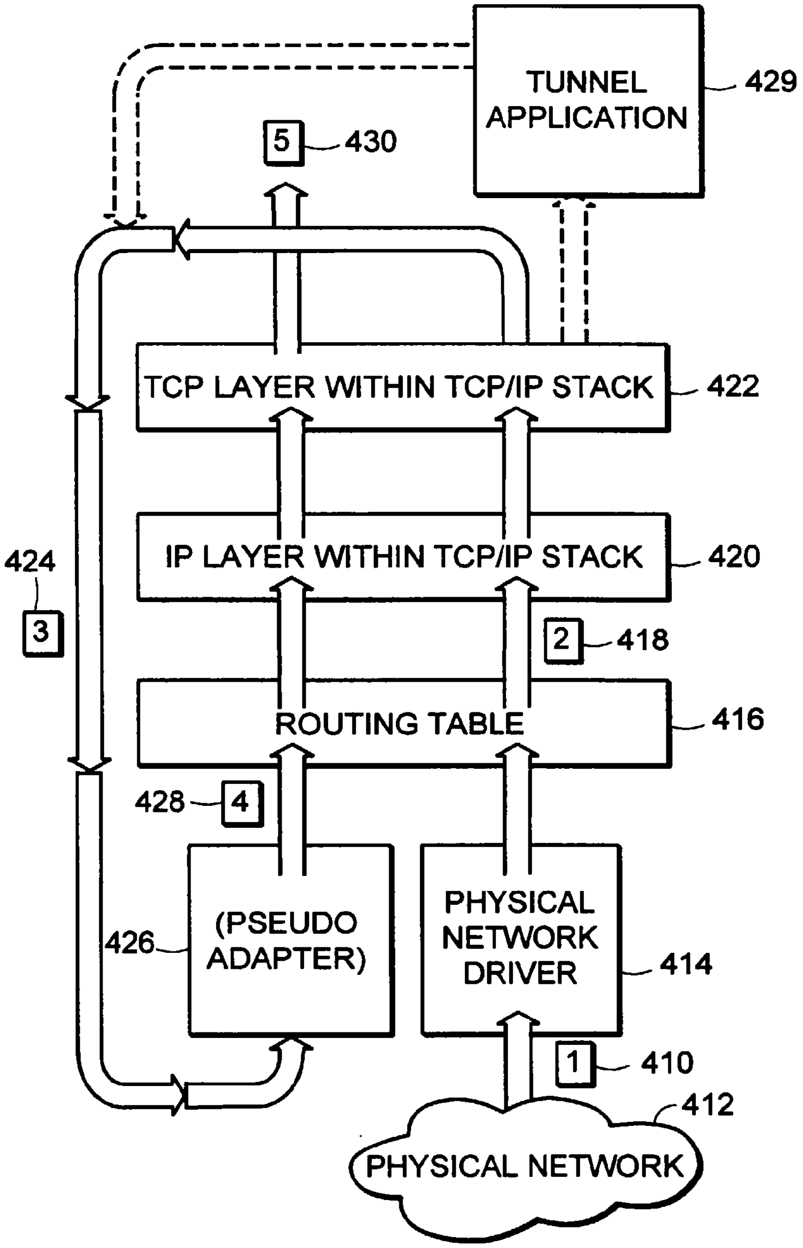

TUNNEL APPLICATION
429
5
430
TCP LAYER WITHIN TCP/IP STACK
422
IP LAYER WITHIN TCP/IP STACK
420
424
3
2
418
ROUTING TABLE
416
428
4
(PSEUDO ADAPTER)
426
PHYSICAL NETWORK DRIVER
414
1
410
412
PHYSICAL NETWORK

FIG. 20

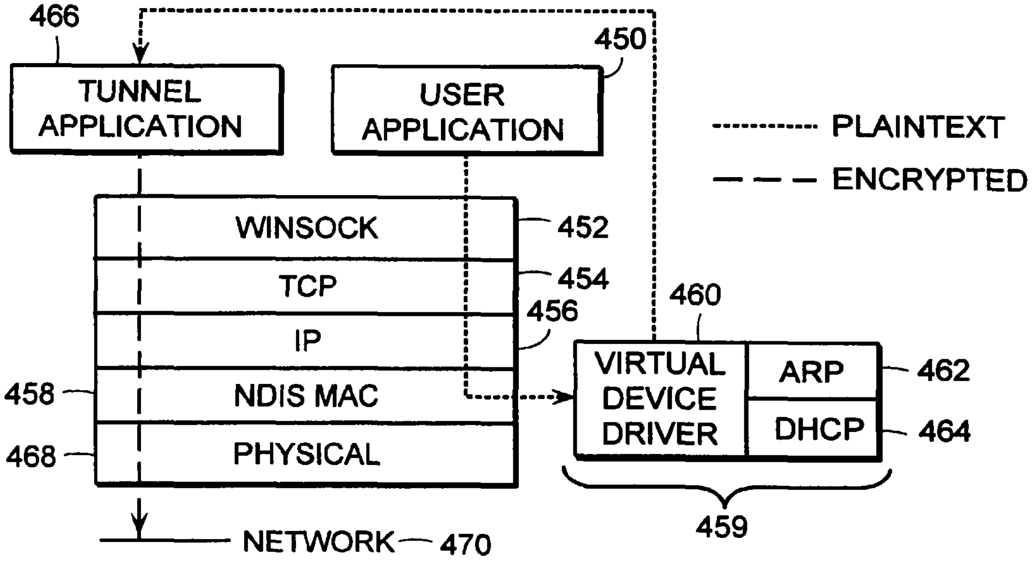

466
TUNNEL APPLICATION
USER APPLICATION
450
PLAINTEXT
ENCRYPTED
WINSOCK
452
TCP
454
IP
456
460
458
NDIS MAC
VIRTUAL DEVICE DRIVER
ARP
462
DHCP
464
468
PHYSICAL
NETWORK
470
459

FIG. 21

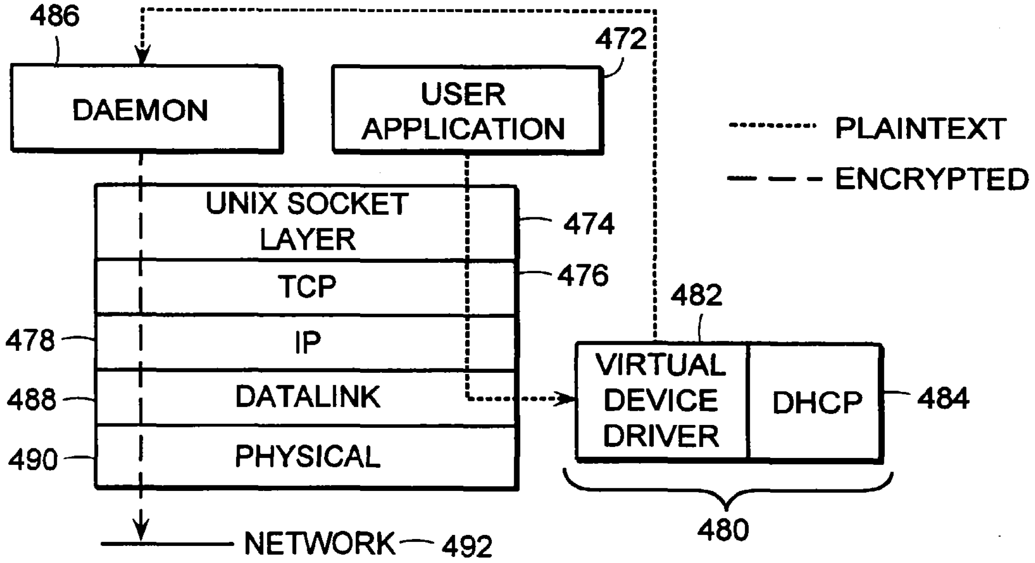

486
DAEMON
USER APPLICATION
472
PLAINTEXT
ENCRYPTED
UNIX SOCKET LAYER
474
TCP
476
482
478
IP
488
DATALINK
VIRTUAL DEVICE DRIVER
DHCP
484
490
PHYSICAL
NETWORK
492
480

FIG. 22

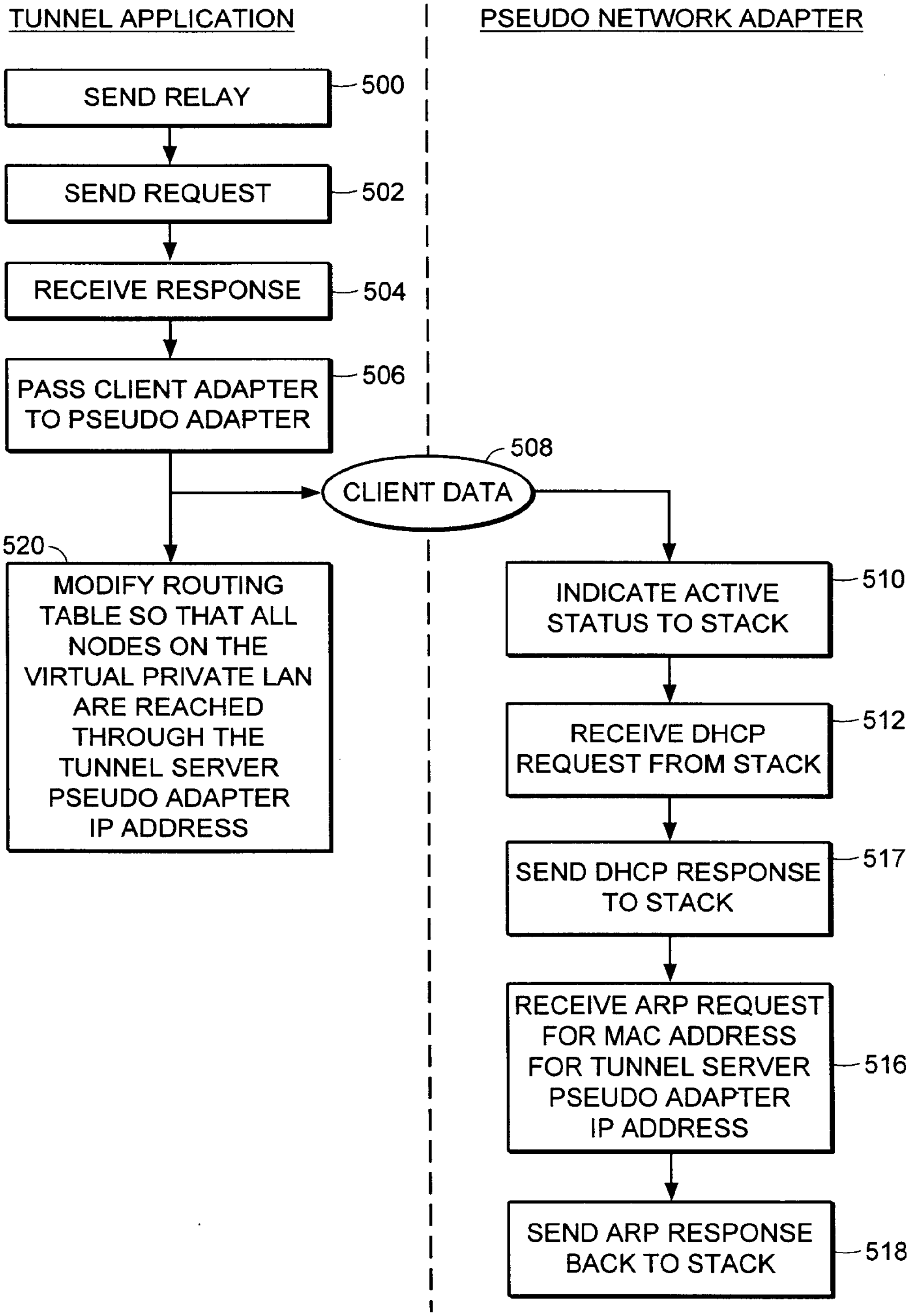

TUNNEL APPLICATION
PSEUDO NETWORK ADAPTER
SEND RELAY
500
SEND REQUEST
502
RECEIVE RESPONSE
504
PASS CLIENT ADAPTER TO PSEUDO ADAPTER
506
CLIENT DATA
508
520
MODIFY ROUTING TABLE SO THAT ALL NODES ON THE VIRTUAL PRIVATE LAN ARE REACHED THROUGH THE TUNNEL SERVER PSEUDO ADAPTER IP ADDRESS
INDICATE ACTIVE STATUS TO STACK
510
RECEIVE DHCP REQUEST FROM STACK
512
SEND DHCP RESPONSE TO STACK
517
RECEIVE ARP REQUEST FOR MAC ADDRESS FOR TUNNEL SERVER PSEUDO ADAPTER IP ADDRESS
516
SEND ARP RESPONSE BACK TO STACK
518

FIG. 23

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 814 589 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
29.12.1997 Bulletin 1997/52

(51) Int. Cl.6: H04L 29/06

(21) Application number: 97109792.8

(22) Date of filing: 16.06.1997

(84) Designated Contracting States:
AT BE CH DE DK ES FR GB GR IE IT LI LU MC NL
PT SE

(30) Priority: 19.06.1996 US 667524

(71) Applicant: AT&T Corp.
New York, NY 10013-2412 (US)

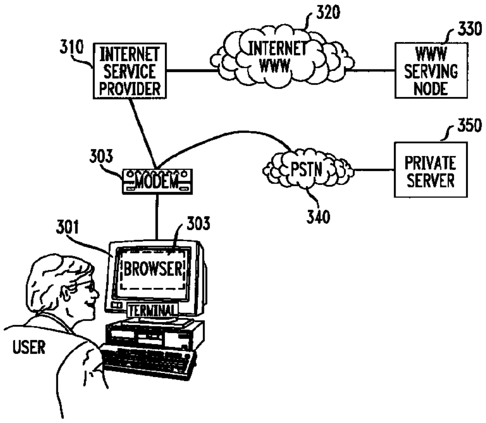
(72) Inventors:
- Harwood, Jonathan P.
  Morganville, N.J. 07751 (US)
- Kimmeth, Thomas
  Gladstone, N.J. 07977 (US)
- Nusbaum, Kurt
  Downers Grove, Illinois 60515 (US)

(74) Representative:
KUHNEN, WACKER & PARTNER
Alois-Steinecker-Strasse 22
85354 Freising (DE)

(54) System and method for automated network reconfiguration

(57) A method is disclosed for providing an enhanced level of security for sensitive or proprietary information associated with information transactions in a public network, such as the Internet. In carrying out that method, an on-line information transaction is bifurcated between a generalized information access portion of such a transaction and an exchange of sensitive user information. With such a bifurcation, the generalized information access portion of the transaction, which generally would constitute the more substantial (in terms of network resources) portion of the transaction, would be handled via a non-secure network, usually a public network such as the Internet. The portion of the transaction involving sensitive user information, on the other hand, would be handled by a separate secure connection, such as a private network, or intranetwork. An important characteristic of this bifurcation arrangement is the provision of a means for automated reconfiguration of a user terminal as between accessing the Ageneralized information via the non-secure network and access to the secure communications network for the exchange of sensitive user information. Such an automated reconfiguration will be carried out without the necessity for any action on the part of the user, and indeed will be largely invisible to the user.

FIG. 3



Printed by Xerox (UK) Business Services
2.15.6/3.4

Description

FIELD OF THE INVENTION

This invention is related to the field of data communications, and more particularly to a method and means for establishing an automatic reconfiguration of a user terminal among alternative tasks.

BACKGROUND OF THE INVENTION

With the increasing popularity of personal computers over the last several years has come a striking growth in transaction-oriented computer-to-computer communications (as opposed to bulk-data transfers among such computers). For convenience herein such transaction-oriented computer-to-computer communications will be described by the shorthand term "information transaction". That growth in the use of computers for such information transactions has unquestionably been fueled by the existence of an international infrastructure for implementing such data communications, known as the Internet. And, driven by the burgeoning demand for such information transaction services, the Internet has itself experienced explosive growth in the amount of traffic handled.

At least partly in response to that demand, a new level of accessibility to various information sources has recently been introduced to the Internet, known as the World Wide Web ("WWW"). The WWW allows a user to access a universe of information which combines text, audio, graphics and animation within a hypermedia document. Links are contained within a WWW document which allow simple and rapid access to related documents. Using a system known as the HyperText Markup Language ("HTML"), pages of information in the WWW contain pointers to other pages, those pointers typically being a key word (commonly known as a hyperlink word). When a user selects one of those key words, a hyperlink is created to another information layer (which may be in the same, or a different information server), where typically additional detail related to that key word will be found.

In order to facilitate implementation of the WWW on the Internet, new software tools have been developed for user terminals, usually known as Web Browsers, which provide a user with a graphical user interface means for accessing information on the Web, and navigating among information layers therein. A commonly used such Web Browser is that provided by Netscape.

The substantial growth in the use of computer networks, and particularly the WWW, for such information transactions, has predictably led to significant commercialization of this communications medium. For example, with the WWW, a user is not only able to access numerous information sources, some public and some commercial, but is also able to access "catalogs" of merchandise, where individual items from such a catalog can be identified and ordered, and is able to carry out a number of banking and other financial transactions. As will be obvious, such commercial transactions will typically involve sensitive and proprietary information, such as credit card numbers and financial information of a user. Thus, with the growth of commercial activity in the Internet, has also come a heightened concern with security.

It is well known that there are persons with a high level of skill in the computer arts, commonly known as "hackers", who have both the ability and the will to intercept communications via the Internet. Such persons are thereby able to gain unauthorized access to various sensitive user information, potentially compromising or misappropriating such information.

The vulnerability of such sensitive user information to misuse when so transmitted via the Internet is a phenomena which has only recently received wide public attention. Unless such security concerns can be quickly addressed and alleviated, the commercial development of this new communications medium may be slowed or even stalled altogether.

SUMMARY OF THE INVENTION

Accordingly, it is an object of the invention to provide an acceptable level of security for sensitive or proprietary information associated with information transactions in a public network, such as the Internet. That object is realized through an arrangement whereby an on-line information transaction is bifurcated between a generalized information access portion of such a transaction and an exchange of sensitive user information. With such a bifurcation, the generalized information access portion of the transaction, which generally would constitute the more substantial (in terms of network resources) portion of the transaction would be handled via a non-secure network, usually a public network such as the Internet. The portion of the transaction involving sensitive user information, on the other hand, would be handled by a separate secure connection, such as a private network, or intranetwork. An important characteristic of this bifurcation arrangement is the provision of a means for automated reconfiguration of a user terminal as between accessing the generalized information via the non-secure network and access to the secure communications network for the exchange of sensitive user information. Such an automated reconfiguration will be carried out without the necessity for any action on the part of the user, and indeed will be largely invisible to the user. In a further embodiment of the invention, a transfer of data is provided from a public to a private network, wherein data selected by a user from a public network site may be arranged and displayed at a user terminal and, subject to further user selection/confirmation activity, thereafter transferred to a private network.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 depicts an illustrative case of information transactions carried out via a public network such as the Internet.

Figure 2 shows the architecture of a browser as would typically be applied for accessing a hypermedia web page.

Figure 3 illustrates the primary elements of the reconfigurable dual-path method of the invention.

Figure 4 depicts in flow chart form the basic jump capability of the methodology of the invention.

Figures 5A & 5B (generally designated collectively herein as "Figure 5") depict in flow chart form the "shopping cart" capability of the methodology of the invention.

Figure 6A & 6B (generally designated collectively herein as "Figure 6") depict in flow chart form the stored configuration capability of the methodology of the invention.

Figure 7A & 7B (generally designated collectively herein as "Figure 7") depict in flow chart form the off-line form capability of the methodology of the invention.

DETAILED DESCRIPTION

For clarity of explanation, the illustrative embodiment of the present invention is presented as comprising individual functional blocks. The functions these blocks represent may be provided through the use of either shared or dedicated hardware, including, but not limited to, hardware capable of executing software.

Figure 1 depicts an illustrative case of information transactions carried out via the Internet. As seen in the figure, an exemplary user obtains access to the Internet by First connecting, via a Terminal **110** having an associated Browser **111**, to an Internet Service Provider **112** selected by the user. That connection between the user and the Internet Service Provider will typically be made via the Public Switched Telephone Network (PSTN) from a modem associated with the user's Terminal to a network node in the Internet maintained by the selected Internet Service Provider.

Once the user has obtained access to the selected Internet Service Provider, an address is provided for connection to another user or other termination site and such a connection is made via the Internet to that destination location. As can be seen from the figure, communication via the Internet may be either user-to-user, as from Terminal **110** to Terminal **130**, or from a user to a node representing an information source accessed via the Internet, such as Public Site **120**.

It will of course be understood that the Internet provides service to a large number of users and includes a large number of such Public Sites, but the illustration provides the essential idea of the communication paths established for such Internet communication. It will also be understood that a number of service classifications are supported by the Internet, with the World Wide Web service, which represents a preferred embodiment for the public network aspect of the method of the invention, being one of the currently most heavily trafficked of such services.

The Web Browser, such as depicted at **111**, can be seen as a software application operating in conjunction with a user terminal (such as Terminal **110**) which provides an interface between such a user terminal and the particular functionality of the WWW information site. The architecture of such a browser is generally described in terms of three main components, as illustrated in Figure 2. At the top level is the Browser **201**, which enables the acquisition of information pages from a WWW server (beginning, in all cases, with the "home page" for that server), for display at a display device associated with the terminal. The Browser also provides the necessary interface for the terminal with the HTML functionality used by the server to provide access to other linked information layers.

The second level of the browser architecture is the TCP/IP Stack **202**, which handles the communications protocols used for connecting the terminal to the WWW server. The bottom level of this architecture is the Dialer **203**, which typically handles the function of providing dialing and setup digits to a modem, as illustrated at **204**, such a modem generally being a part of the terminal. Normally, upon receiving dialing and other setup information from the dialer, the modem would cause a connection to be made via the PSTN to the Internet Service Provider selected for that terminal.

After a connection is established in this manner to the Internet Service Provider, an address would be provided for the WWW information node sought to be contacted, a connection to that node made through the Internet, and the home page for that node caused to be displayed at the terminal's display device. A user would then select a key word in that home page, typically by clicking on the word with a mouse or similar device, and, upon transmission of that selection signal to the WWW server, a hyperlink would be created to the linked information layer and the open page of that layer would be caused to be displayed at the user terminal.

As explained above, serious questions have been raised in respect to the security of communications via the public Internet. (Note, that the discussion herein is focused on the Internet, and particularly the WWW functionality of the Internet, as a preferred embodiment of such public data communication networks generally, but the methodology of the invention will be applicable to any such network.) To address this problem, the methodology of the invention begins with a bifurcation of the information transaction between a user and the selected information transaction provider into a portion related to sensitive or proprietary user information, and other information comprising that transaction. With such a bifurcation, it becomes possible to provide substantial security for that proprietary information by use of an alternative communications path for that separated portion of the transaction via a private network, or intranetwork -- *i.e.*, a connection between a user's terminal and a secure serving node on that private network. It is anticipated that a coordination means will be established in respect to the management of information among the public and private network elements of the bifurcated information transaction.

In its basic form, this methodology may be carried out by the user terminal initiating a call via the Internet to a selected WWW node, and upon establishing connection to that node, proceeding with the desired information transaction up to the point where an exchange of sensitive or proprietary information were required. At that point the user terminal would be instructed by the WWW server to terminate that connection (*i.e.*, hangup) and to place a new call to an identified private network server for the necessary exchange of sensitive information.

However, in order to accomplish such a dual-path transaction, it is necessary that the browser at the user terminal be reconfigured to provide the dialing, authorization (*i.e.*, login and password), and other needed information for accessing the alternative private network, in order to implement the proprietary portion of the transaction. It will also usually be the case that, upon completion of that private-network transaction, the original dialer, stack and browser configurations will need to be restored, in order for the terminal to retain its normal Internet access functionality. Such a reconfiguration and subsequent restoral of the necessary parameters in the browser, stack and dialer is likely to be well beyond the capabilities of the average user.

Accordingly, as a further embodiment of the inventive methodology, an automated browser reconfiguration means is provided which interoperates with the browser. This browser reconfiguration means is described in detail hereafter and will be referred to as the "Bridging Software".

Figure 3 provides an illustration of the primary elements of the reconfigurable dual-path method of the invention. As seen in the figure, a first path comparable to the Internet link shown in Figure 1, between User Terminal **301** and WWW Serving Node **330** (via Browser **302**, Modem **303**, Internet Service Provider **310**, and Internet **320**) is provided. However, an alternative path is now provided from the output of Modem **303** to Private Server **350**. That path is illustrated as being via the PSTN, which is generally regarded as being highly secure, but an alternative dedicated or other more-secure path between the User Terminal **301** and the Private Server **350** could as well be provided. In keeping with the discussion above, Browser **302** shown in Figure 3 would also include the Bridging Software installed as a helper application for implementing the automatic reconfiguration of the Browser.

In the operation of this system, a user would normally make an initial connection to an Internet application, such as the application represented by WWW Serving Node **330**, which, *e.g.*, might be a shopping application, a financial transaction, or the provision of an enrollment form for off-line preparation. After conducting all, or some portion of an information transaction short of an exchange of sensitive or proprietary information, including a capture by the user's terminal of needed information from the public site, a user provides a signal indicative of an end to that portion of that transaction. During the course of the public portion of the information transaction, specially configured files are sent from the WWW serving node to the Bridging Software associated with Browser **302**. Such files contain instructions for the Bridging Software to store information-like products -- e.g., for selected items from a catalog, forms for enrollment, or non-secure portions of a financial transaction, and reconfiguration information for dialing and logging into the private portion of the transaction. The Bridging Software then hangs up the Internet connection, edits the user terminal's browser, stack and dialer files to reconfigure the terminal to connect to the private server. Prior to automatic redialing of the new private site for the user, the Bridging Software may be instructed by the application operating at WWW Server Node **330** to display items chosen for purchase, or to display a form for the end-user to complete off-line before dialing the private application. Upon connecting to the private application and completing the transaction as to the user sensitive information in a private environment, the Bridging Software then restores the end-user software to the dialing and authorization parameters required to dial to the public Internet.

A particularly advantageous application of the automated reconfiguration and information transfer methodology of the Bridging Software is that it adds value to certain WWW servers which do not possess the Common Gateway Interface ("CGI") capability -- *i.e.*, a provision of specialized functions on the server beyond just displaying HTML files, and are accordingly unable to accomplish any transactional processing in respect to items selected by a user. In effect, such a non-CGI server, on its own, can only serve as a "billboard" for the items represented in its database.

However, with the collection and redelivery process of the Bridging Software, a data capture and processing mechanism can be implemented for servers operating in a non-CGI environment -- such servers being incapable of more than the simple delivery of static data packets corresponding to available items. The data set enabled by the Bridging Software is a mechanism for augmenting such limited server capabilities by defining a flexible mechanism for the receipt, display, and delivery of arbitrary data from one site to another.

In such a scenario, the Bridging Software receives a "shopping cart" item list from the host as a data-set defined with a static MIME data packet associated with the Bridging Software. This information comprising the data-set may be updated, displayed to the user in a "read-only" fashion, or presented to the user for order selection.

During the process of interacting with the WWW server, a user may trigger HTML links resulting in additional MIME packets for the Bridging Software being delivered to the client. These packets allow items to be added and/or removed from the specified data set or presented to the user for local confirmation. The user will interact with a pop-up screen provided by the Bridging Software which presents the items available with product information, such as part number, description, unit cost, etc. The user identifies those items which are to be placed into the "shopping cart" and the quantity of items desired. Upon completion of the form, the Bridging Software stores the order in a format suitable for subsequent delivery to the private server site.

An additional feature provided by the methodology of the Bridging Software is an automated mechanism for providing compatibility with user terminals not previously having the Bridging Software included with the terminal's browser. To that end, the Bridging Software located at an accessed public network site initially checks to see if the browser counterpart for that software is loaded at the calling user terminal. If yes, the heretofore described processes of the Bridging Software go forward. If not however, a request is sent through the public host to download the Bridging Software to the calling terminal. After such a download, a helper application loads the Bridging Software to the terminal's browser.

I. Illustrative Embodiments

A variety of browser reconfiguration applications are supported by the automated browser reconfiguration means of the invention. Four essentially diverse capabilities of this invention, which support such applications, are described hereafter as illustrative embodiments of the invention.

A. Basic Jump Capabilities

In this configuration, which is illustrated in flow chart form in Figure 4, an end-user is connected to a chosen WWW serving node (where a desired information product is made available) via a modem and an Internet browser associated with the user's terminal (**Step 401** of Figure 4). After conducting an information transaction with the selected WWW serving node for some interval (determined in relation to the specific application accessed), the user clicks on a hypertext link, or picture, to begin an automated process which will cause that public session to be terminated and a new connection established to an alternate private data network (**Step 402**).

In response to that user action, a data message containing parameter reconfiguration instructions is passed from the WWW server application to the Bridging Software at the user's terminal (**Step 403**). Upon receiving such instructions, the Bridging Software edits the user's on-line communications software parameters, reconfiguring that software to dial the alternate data network (**Step 404**). This reconfiguration is fully automatic and transparent to the user, and includes parameters such as modem dial number, login, password, and TCP/IP addresses. At that point, the Bridging Software causes the modem to disconnect the current data network connection, shutting down the browser, and to then dial the alternate private data network (**Step 405**).

With the establishment of a connection to the private server on the alternate data network, the user interacts with the alternate data network application as appropriate (**Step 406**), and after an interval completes his activity with the alternate data network and provides an indication of such completion (**Step 407**). A data message containing parameter reconfiguration instructions is then passed from the alternate data network application to the Bridging Software (**Step 408**).

At that point, the Bridging Software again edits the user's on-line communications software parameters, reconfiguring them to dial the original public data network, or another preselected network (**Step 409**). As with the first reconfiguration, this configuration is automatic and includes parameters such as modem dial number, login, password, and TCP/IP addresses. The Bridging Software automatically causes the current private data network to be disconnected by the modem (**Step 410**), and if appropriate, causes the original public data network to be redialed (**Step 411**). When such a reconnection to the public data network is established, the end-user would then continue his application in the public data network.

B. "Shopping Cart" Capability

With this configuration, illustrated in flow chart form in Figure 5, a user begins by establishing a connection to a WWW application (assuming for the moment that the application is non-CGI enabled) at a serving node for that application, using the Internet browser and modem associated with the user's terminal (**Step 501** of Figure 5). Upon finding an item in that application to be saved, or remembered for later consideration, or purchase, the user clicks on a hypertext link, or picture, representing that item (**Step 502**). That application then sends a data message to the Bridging Software containing information about the items selected (**Step 503**) and such information is stored by the Bridging Software in the "shopping cart" file in the user's terminal (**Step 504**). Such selection download and storage steps (*i.e.*, steps 502, 503 & 504) are repeated for as many items as the user chooses to select. At any point after the Bridging Software has received the first set of item selection information, the user can instruct the Bridging Software to cause those selected items about which such information has been received to be displayed locally (at the user's terminal), where the user may review or edit (including deletion if desired) the collection of items theretofore selected. The application may also control display characteristics such as color and font for such locally displayed items. Note that in the case of a CGI-enabled application, the application itself will keep track of the items selected by the user and only download the totality of the selected items at the end of the selection process, and accordingly, the described local display option will not be applicable to such a CGI-enabled application.

At the point of completion of his "shopping", the user clicks on a hyper-text link or picture to "check out" (**Step 505**), which will begin a process of causing a jump to an alternate data network for the completion of sensitive portions of the transaction. To that end, a data message containing parameter reconfiguration instructions is passed from the WWW application to the Bridging Software (**Step 506**). It is to be noted that, as a security measure, information such as the new dial number, IP address, home page, configuration data (*e.g.*, login, password, DNS address) may be passed over the public network in encrypted form.

Upon receiving such reconfiguration instructions, the Bridging Software edits the user's on-line communications software parameters, reconfiguring that software to dial the alternate data network (**Step 507**). This reconfiguration is fully automatic and transparent to the user, and includes parameters such as modem dial number, login, password, and TCP/IP addresses. At that point, the Bridging Software causes the modem to disconnect the current data network connection, shutting down the browser, and to then dial the alternate data network (**Step 508**).

The Bridging Software passes the stored "shopping cart" data captured from the WWW application to the alternate network application (**Step 509**), where that data may be displayed for the user, permitting the user to confirm and/or modify the data (**Step 510**). The user interacts with the alternate data network application as appropriate, and after an interval completes his activity with the alternate data network (**Step 511**) and thus, by providing an appropriate completion signal to the application, completing the private portion of the information transaction (**Step 512**). A data message containing parameter reconfiguration instructions is then passed from the alternate data network application to the Bridging Software (**Step 513**).

The Bridging Software, at this point, again edits the user's on-line communications software parameters, reconfiguring them to dial the original (or another pre-defined) data network (**Step 514**). As with the first reconfiguration, this configuration is automatic and includes parameters such as modem dial number, login, password, and TCP/IP addresses. The Bridging Software automatically causes the current private data network to be disconnected by the modem (**Step 515**), and if appropriate, causes the original public data network to be redialed (**Step 516**). When such a reconnection is established to the point in the public data network where the user had left off to handle the secured aspects of his information transaction, the user would then continue his application in the public data network.

C. Stored Configuration Capabilities

For this configuration, depicted in flow chart form in Figure 6, an end-user is connected to a chosen WWW serving node (where a desired information product is made available) via a modem and an Internet browser associated with the user's terminal (**Step 601** of Figure 6). The user selects a hypertext link or picture associated with the WWW application by clicking on such link or picture (**Step 602**). A data message containing parameter reconfiguration instructions and an application icon (related to the selected hypertext link or picture) is passed from the WWW application to the Bridging Software (**Step 603**).

The Bridging Software creates an icon for display at the user's terminal, and saves a Bridging Software configuration file that is associated with that icon (**Step 604**). Such Bridging Software actions are automatic and multiple selections may he captured in this manner. At this point the user may continue the on-line session, or, if all desired selections have been made, a signal is provided from the user that the session should be discontinued (**Step 605**). The Bridging Software then automatically disconnects the current data network connection (**Step 606**).

After disconnecting from the WWW application, and following an interval determined by the user, a new application is selected by the user by clicking on the appropriate new icon displayed at the user's terminal (**Step 607**). The Bridging Software receives the reconfiguration instructions from the file associated with the selected icon (**Step 608**).

The Bridging Software edits the user's on-line communications software parameters, reconfiguring that software to dial the alternate data network (**Step 609**). The Bridging Software then automatically starts the user's Internet browser software and causes the alternate network application to be dialed by the modem associated with that terminal (**Step 610**). Upon establishing a connection to the alternate network, the user interacts with that application and completes the transaction to the user's satisfaction (**Step 611**). After a signal is sent to the alternate network indicating such completion of the user's activity (**Step 612**), a data message containing parameter reconfiguration instructions is passed from the alternate data network application to the Bridging Software (**Step 613**). That Software then causes the user's terminal configuration parameters to be reset (**Step 614**) and the alternate data network to be automatically disconnected (**Step 615**).

D. Off-Line Form Capability

In this configuration, depicted in flow chart form in Figure 7, an end-user is connected to a chosen WWW serving node (where a desired information product is made available) via a modem and an Internet browser associated with the user's terminal (**Step 701** of Figure 7). The user selects a hypertext link or picture associated with an off-line form application -- an exemplary such form being an HTML-based form -- by clicking on such link or picture (**Step 702**). A data message containing parameter reconfiguration instructions for the Bridging Software, the selected off-line-form application, and an optional icon (related to the selected hypertext link or picture) is passed from the WWW application to the Bridging Software (**Step 703**). Note that the selected off-line form may be for either single or multiple use.

In the case of a delayed or multiple use of the selected form, the Bridging Software may create an icon for display at the user's terminal, and will save a Bridging Software configuration file that is associated with that icon (**Step 704**). The form in question is also saved on the user's terminal. Such Bridging Software actions are automatic. At this point the user may continue the on-line session, or, if all desired selections have been made, a signal is provided from the user that the session should be discontinued (**Step 705**). The Bridging Software then automatically disconnects the current data network connection (**Step 706**).

After disconnecting from the WWW application, two cases are to be considered as to the further processing of the selected form: (1) an immediate single use of the form and (2) either a delayed or multiple use of the form. In the first case, the Bridging Software edits the user's on-line communications software parameters, reconfiguring that software to dial the alternate data network. The Bridging Software then automatically starts the user's Internet browser software which is caused to display the off-line form. The user then completes the off-line form and chooses a "Submit Form" button displayed at his terminal.

In the second case, the Bridging Software will have created an icon for display at the user's terminal and saved a Bridging Software configuration file associated with that icon. Following an interval determined by the user, the off-line-form application is started by the user by clicking on the new form icon displayed at the user's terminal (**Step 707**). The Bridging Software receives the reconfiguration instructions from the file associated with the selected icon (**Step 708**).

The Bridging Software edits the user's on-line communications software parameters, reconfiguring that software to dial the alternate data network (**Step 709**). The Bridging Software then automatically starts the user's Internet browser software which is caused to display the off-line form (**Step 710**). The user then completes the off-line form and chooses a "Submit Form" button displayed at his terminal (**Step 711**).

In either the first or second case, following activation of the "Submit Form" button, the alternate network application is then caused to be dialed by the Bridging Software. Upon establishing a connection to the alternate network, the form data is passed to the alternate network (**Step 712**). The user then interacts with that application and completes the application (**Step 713**). After a signal is sent to the alternate network indicating such completion of the user's activity (**Step 714**),a data message containing parameter reconfiguration instructions is passed from the alternate data network application to the Bridging Software (**Step 715**). That Software then causes the user's terminal configuration parameters to be reset (**Step 716**) and the alternate data network to be automatically disconnected (**Step 717**).

CONCLUSION

A system and method has been described for the automatic switching of an information transaction between two or more alternate networks. This functionality, which incorporates a reconfiguration means designated herein as the Bridging Software, supports the movement of application specific data from one on-line environment to another. Among potential applications of this process for passing data between different environments are: selected items for purchase ("shopping cart"), captured data from forms, and other server captured data such as web pages visited.

The Bridging Software reconfiguration means is intended to work with various Web Browser software implementations, including the Netscape Personal Edition (NPE) Software for Windows 3.1 and 3.11, and which represents a working embodiment for the invention. The Bridging Software installs itself as a helper application within the browser application and utilizes a special MIME type configuration file to pass reconfiguration and "shopping cart" information from the server to the client software.

When an application requires a user to re-connect to a private application, a reconfiguration file is passed to the Bridging Software helper application via a CGI script or simple hyper-text link. The helper application disconnects the current data connection, reconfigures the dial parameters (dial #, login password, DNS address, and home page) and initiates the dial program so the end-user can access the private application.

When the end-user connects to the private application, the Bridging Software reconfiguration means provides the new "private server" application with data collected from the "public server", and the application resumes in a private, secure environment.

The Bridging Software allows both short term and long term storage of dial configurations. Configurations passed to the Bridging Software can be designated as single use configurations and discarded after the application has terminated, or saved and displayed to the end-user as a dial choice by the Bridging Software.

Although the present embodiment of the invention has been described in detail, it should be understood that various changes, alterations and substitutions can be made therein without departing from the spirit and scope of the invention as defined by the appended claims. In particular, it is noted that, while the invention has been primarily described in terms of a preferred embodiment based on an automatic reconfiguration between a public and a private data network, any the methodology of the invention will be equally applicable to any set of alternate networks.

Claims

1. A method for managing a transaction via a communications path between a terminal device and a serving node in a data network, said method comprising the steps of:

establishing an initial communications path via a first connection between said terminal device and a serving node in a first data network;
receiving information from said serving node in said first data network for effecting a reconfiguration of said communications path for said transaction from said first connection in said first data network to a second connection in a second data network; and
automatically connecting said terminal device to a serving node in said second data network via said second connection.

2. A method for managing a transaction via a communications path between a terminal device and a serving node in a data network, said method comprising the steps of:

establishing an initial communications path via a first connection between said terminal device and a serving node in a first data network;
selecting at least one information item from a data base of said information items provided at said serving node in said first data network;
causing said selected information items to be downloaded to said terminal device via said first connection;
receiving information from said serving node in said first data network for effecting a reconfiguration of said communications path for said transaction from said first connection in said first data network to a second connection in a second data network; and
automatically connecting said terminal device to a serving node in said second data network via said second connection.

3. A method for managing a transaction via a communications path between a terminal device and a serving node in a data network, said method comprising the steps of:

establishing an initial communications path via a first connection between said terminal device and a serving node in a first data network;
identifying at least one data network application from a data base of said data network applications provided at said serving node in said first data network;
receiving information from said serving node in said first data network for reconfiguring said terminal device for implementation of a communication path via an alternate connection between said terminal device and at least one of said identified data network applications in a second data network; and
in response to a selection signal from a user, automatically connecting said terminal device to a selected one of said identified data network applications via said alternate connection.

4. A method for managing a transaction via a communications path between a terminal device and a serving node in a data network, said method comprising the steps of:

establishing an initial communications path via a first connection between said terminal device and a serving node in a first data network;
selecting an off-line form application from a data base provided at said serving node in said first data network;
receiving information from said serving node in said first data network for reconfiguring said terminal device for implementation of a communication path via a second connection between said terminal device and said

selected off-line form application in a second data network; and

in response to, a selection signal from a user, automatically connecting said terminal device to said selected off-line form application.

5. The method for managing a transaction of Claim 1 or 2 including the further step of recognizing a signal to reconfigure said communications path from said first connection to said second connection.

6. The method for managing a transaction of Claim 3 wherein said selected data network application is operated at a serving node in said second data network.

7. The method for managing a transaction of Claim 4 wherein said selected off-line form application is operated at a serving node in said second data network.

8. The method for managing a transaction of one of the Claims 1, 2, 6 or 7 wherein said serving nodes in said first and said second data networks are manifested in a common node.

9. The method for managing a transaction of Claim 1 or 2 wherein said step of receiving information includes the further step of effecting said reconfiguration of said communications path.

10. The method for managing a transaction of Claim 1 or 2 wherein said step of automatically connecting includes the step of automatically disconnecting said first connection prior to implementation of said second connection.

11. The method for managing a transaction of Claim 1 or 2 including the further steps of:

automatically disconnecting said second connection in response to a user signal; and

reconfiguring said terminal device to enable, in response to user instruction, an implementation of a connection via an identified data network.

12. The method for managing a transaction of Claim 11 wherein said step of automatically reconfiguring said terminal device includes the step of effecting said implementation of said connection via said identified data network.

13. The method for managing a transaction of Claim 2 wherein said step of causing said selected information items to be downloaded includes the further step of causing said selected information items to be displayed at said terminal device.

14. The method for managing a transaction of Claim 13 wherein said displayed selected items can be edited by a user at said terminal device.

15. The method for managing a transaction of Claim 13 wherein display characteristics for said displayed selected items can be controlled at said terminal device.

16. The method for managing a transaction of Claim 2 wherein said step of automatically connecting includes the step of uploading said selected information items from said terminal device to said service provider via said second connection.

17. The method for managing a transaction of Claim 3 including the further steps of:

automatically disconnecting said alternate connection in response to a user signal; and

reconfiguring said terminal device to enable implementation of a pre-selected connection between said terminal device and an identified data network.

18. The method for managing a transaction of Claim 17 wherein said step of automatically reconfiguring said terminal device includes the further step of effecting said implementation of said pre-selected connection.

19. The method for managing a transaction of Claim 4 including the further step of downloading from said serving node in said first data network to said terminal device of an off-line form related to said off-line form application.

20. The method for managing a transaction of Claim 4 including the further step of uploading said downloaded off-line form from said terminal device to said selected off-line form application, after processing by a user.

21. The method for managing a transaction of Claim 4 including the further steps of:

automatically disconnecting said connection to said selected off-line form application in response to a user signal; and
reconfiguring said terminal device to enable implementation of a pre-selected connection between said terminal device and an identified data network.

22. The method for managing a transaction of Claim 21 wherein said step of automatically reconfiguring said terminal device includes the further step of effecting said implementation of said pre-selected connection.

23. A method for managing connections between a terminal device and at least one information source/processor wherein at least two of said connections are implemented via separate communications networks, comprising the steps of:

recognizing a signal for connection to an information source/processor via a communications network other than a communications network for which a predetermined connection is configured;
causing said terminal device to implement a connection to said information source/processor via said other communications network; and
upon termination of said information source/processor connection via said other communications network, automatically reconfiguring a connection criteria in said terminal device to enable said terminal device to implement, in response to user instruction, a connection via an alternative one of said communications networks.

24. The method for managing connections of Claim 23 wherein said recognizing step occurs at a point when said terminal device is connected to a given source/processor.

25. The method for managing connections of Claim 23 wherein information items may be selected by a user at said terminal device from said given source/processor, and including the further step of causing said selected information items to be downloaded from said source/processor to said terminal device.

26. The method for managing connections of Claim 25 wherein said step of effecting connection includes the further step of uploading said selected information items from said terminal device to said other information source/processor.

27. The method for managing connections of Claim 26 wherein said selected information items are processed by said user at said terminal device prior to uploading to said other information source/processor.

28. The method for managing connections of Claim 24 including the further step of causing said given source/processor to download to said terminal device configuration data for enabling said step of effecting connection to said other information source/processor.

29. The method for managing connections of Claim 24 including the further step of causing said other source/processor to download to said terminal device configuration data for enabling said step of automatically restoring a prior connection criteria in said terminal device.

30. A method for enhancing security of certain data in an on-line information transaction comprising the steps of:

bifurcating said information transaction into a first portion comprising said certain data and a remaining portion, wherein said remaining portion is carried out via a public on-line communications connection between a terminal device and a public information server;
causing said first portion to be carried out via a secure private on-line communications connection between said terminal device and a private information server; and
automatically reconfiguring network access means in said terminal device to switch between said public connection and said private connection.

FIG. 1

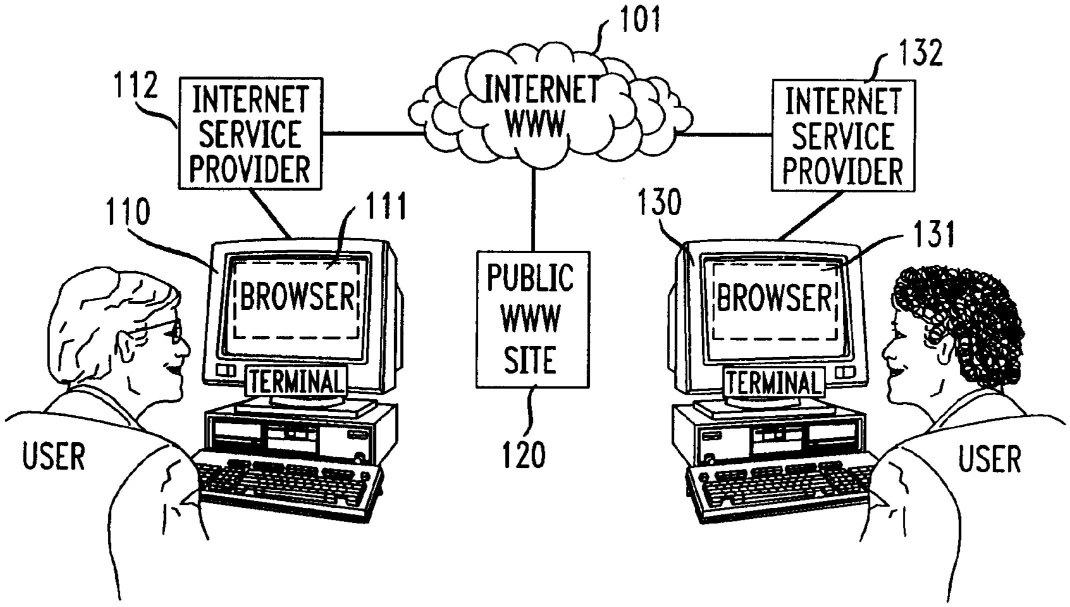

101
INTERNET WWW
112
INTERNET SERVICE PROVIDER
132
INTERNET SERVICE PROVIDER
110
111
BROWSER
TERMINAL
130
131
BROWSER
TERMINAL
PUBLIC WWW SITE
120
USER
USER

FIG. 2

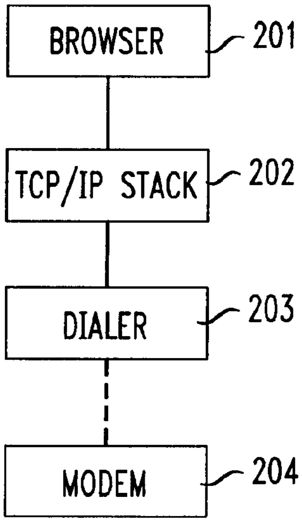

BROWSER
201
TCP/IP STACK
202
DIALER
203
MODEM
204

FIG. 3

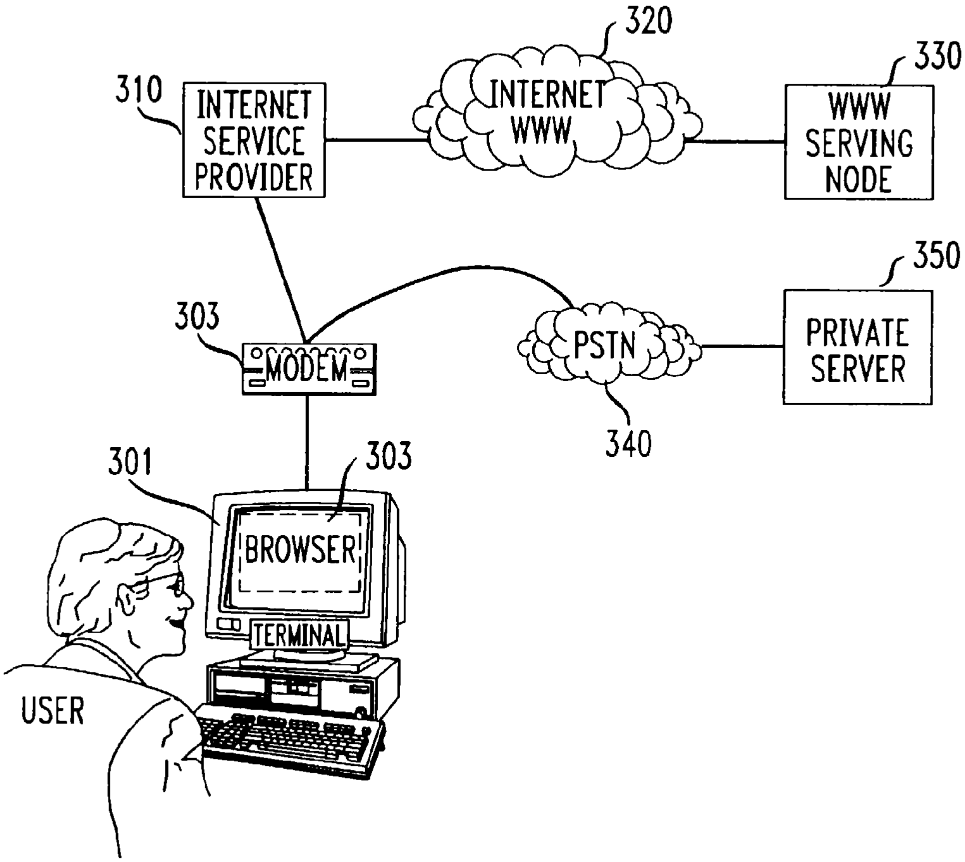

320
330
310
INTERNET SERVICE PROVIDER
INTERNET WWW
WWW SERVING NODE
350
303
PRIVATE SERVER
MODEM
PSTN
340
301
303
BROWSER
TERMINAL
USER

FIG. 4

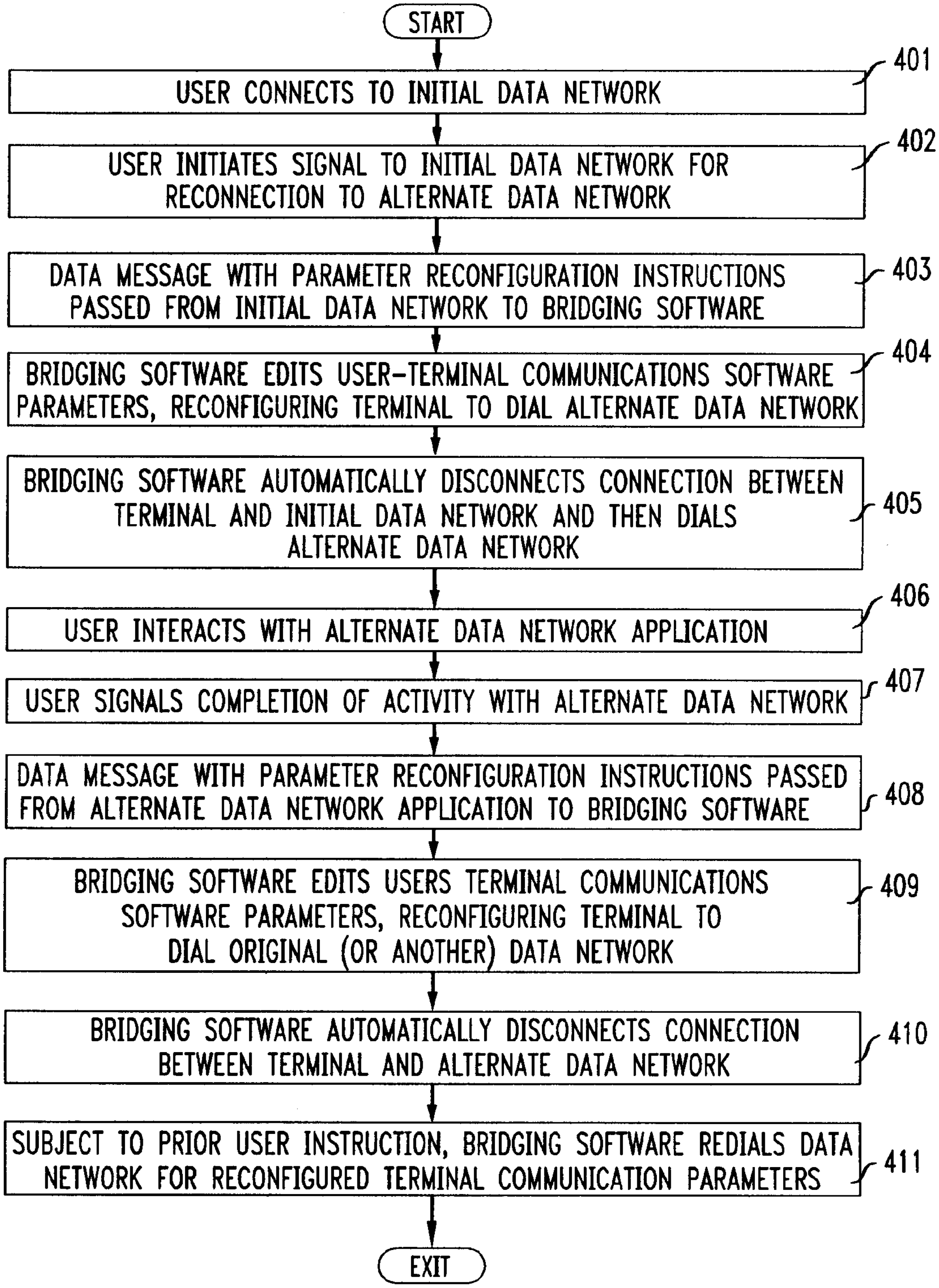

START
USER CONNECTS TO INITIAL DATA NETWORK
401
USER INITIATES SIGNAL TO INITIAL DATA NETWORK FOR RECONNECTION TO ALTERNATE DATA NETWORK
402
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM INITIAL DATA NETWORK TO BRIDGING SOFTWARE
403
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL ALTERNATE DATA NETWORK
404
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND INITIAL DATA NETWORK AND THEN DIALS ALTERNATE DATA NETWORK
405
USER INTERACTS WITH ALTERNATE DATA NETWORK APPLICATION
406
USER SIGNALS COMPLETION OF ACTIVITY WITH ALTERNATE DATA NETWORK
407
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM ALTERNATE DATA NETWORK APPLICATION TO BRIDGING SOFTWARE
408
BRIDGING SOFTWARE EDITS USERS TERMINAL COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL ORIGINAL (OR ANOTHER) DATA NETWORK
409
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND ALTERNATE DATA NETWORK
410
SUBJECT TO PRIOR USER INSTRUCTION, BRIDGING SOFTWARE REDIALS DATA NETWORK FOR RECONFIGURED TERMINAL COMMUNICATION PARAMETERS
411
EXIT

FIG. 5A

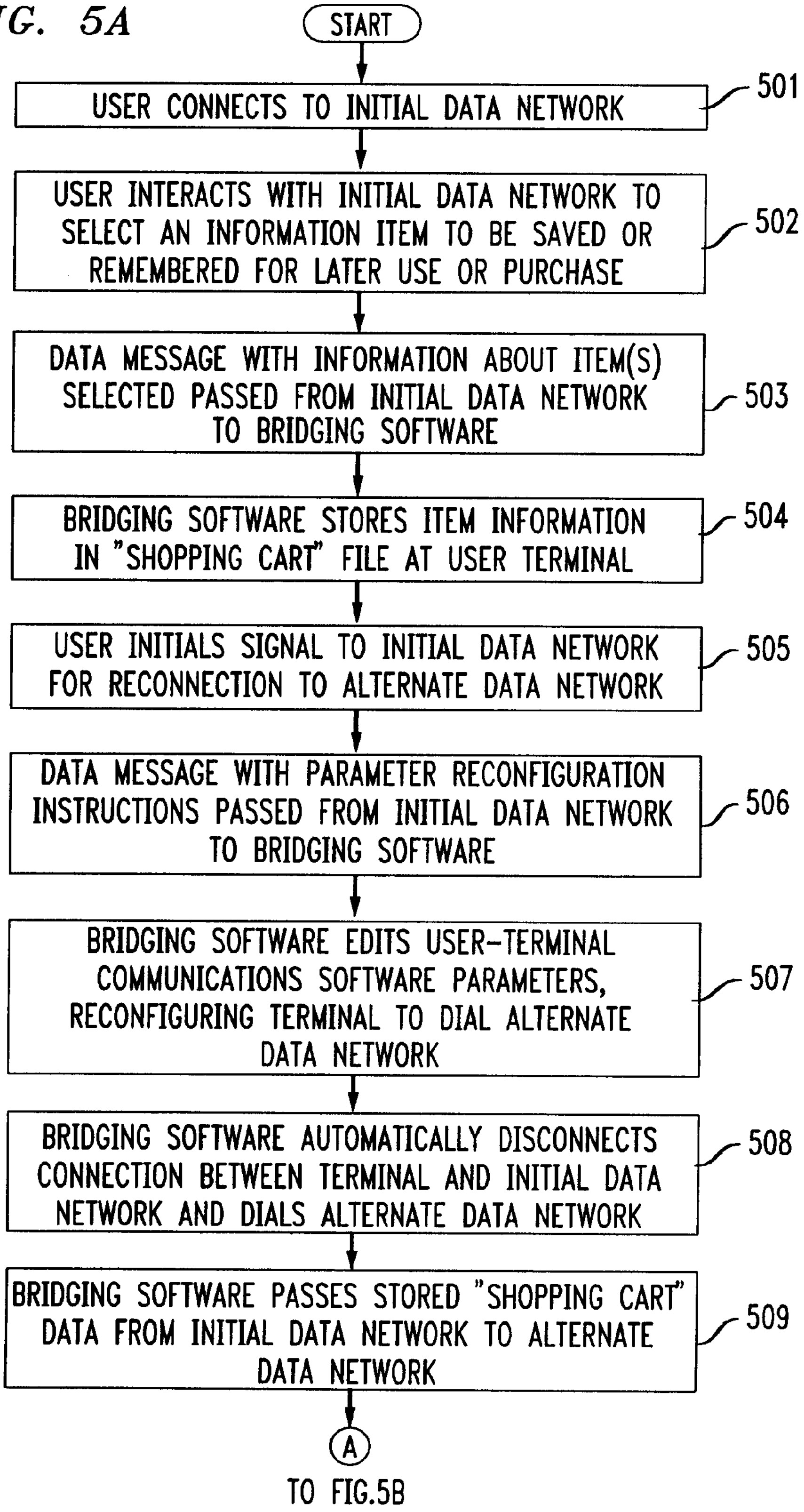

START
USER CONNECTS TO INITIAL DATA NETWORK
501
USER INTERACTS WITH INITIAL DATA NETWORK TO SELECT AN INFORMATION ITEM TO BE SAVED OR REMEMBERED FOR LATER USE OR PURCHASE
502
DATA MESSAGE WITH INFORMATION ABOUT ITEM(S) SELECTED PASSED FROM INITIAL DATA NETWORK TO BRIDGING SOFTWARE
503
BRIDGING SOFTWARE STORES ITEM INFORMATION IN "SHOPPING CART" FILE AT USER TERMINAL
504
USER INITIALS SIGNAL TO INITIAL DATA NETWORK FOR RECONNECTION TO ALTERNATE DATA NETWORK
505
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM INITIAL DATA NETWORK TO BRIDGING SOFTWARE
506
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL ALTERNATE DATA NETWORK
507
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND INITIAL DATA NETWORK AND DIALS ALTERNATE DATA NETWORK
508
BRIDGING SOFTWARE PASSES STORED "SHOPPING CART" DATA FROM INITIAL DATA NETWORK TO ALTERNATE DATA NETWORK
509
A
TO FIG.5B

FIG. 5B

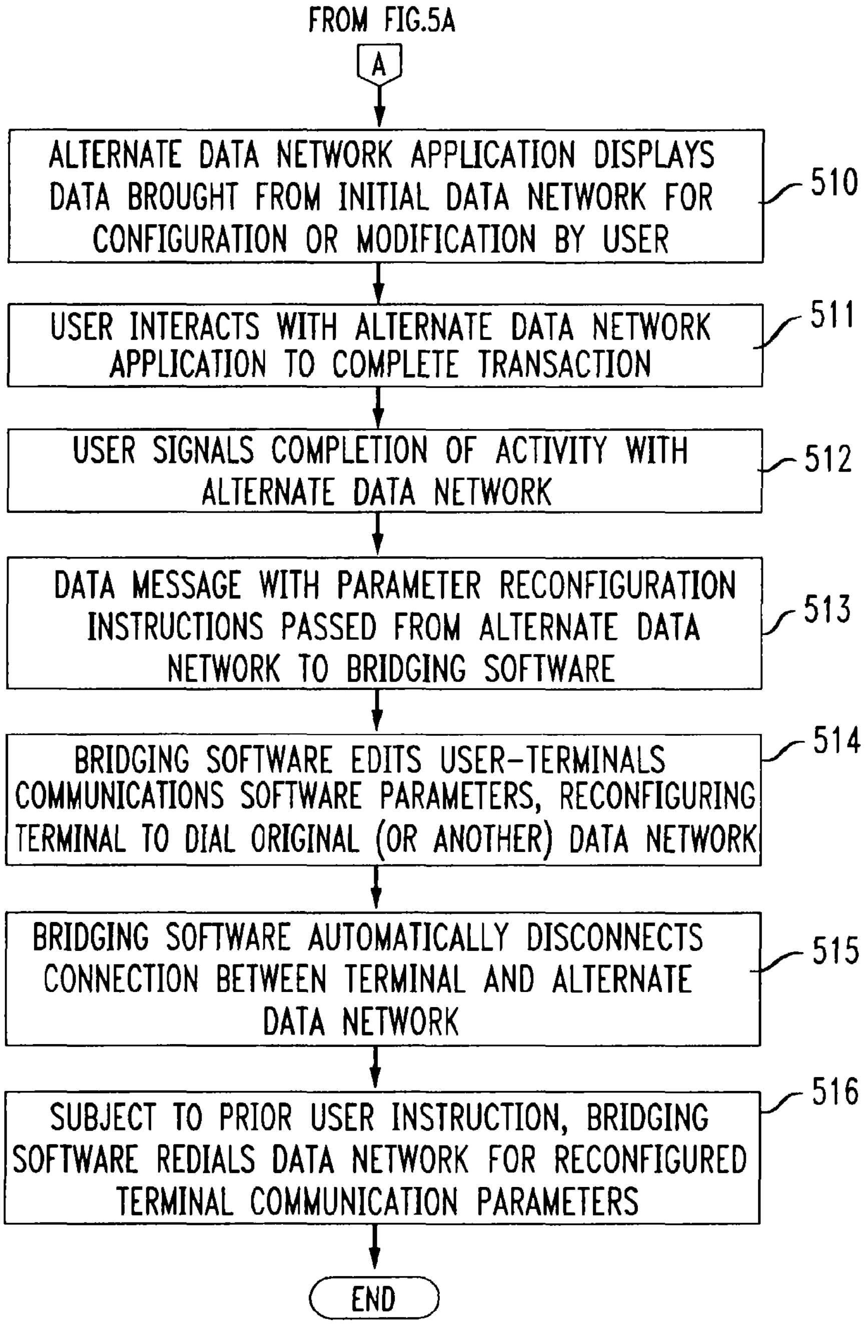

FROM FIG.5A
A
ALTERNATE DATA NETWORK APPLICATION DISPLAYS DATA BROUGHT FROM INITIAL DATA NETWORK FOR CONFIGURATION OR MODIFICATION BY USER
510
USER INTERACTS WITH ALTERNATE DATA NETWORK APPLICATION TO COMPLETE TRANSACTION
511
USER SIGNALS COMPLETION OF ACTIVITY WITH ALTERNATE DATA NETWORK
512
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM ALTERNATE DATA NETWORK TO BRIDGING SOFTWARE
513
BRIDGING SOFTWARE EDITS USER-TERMINALS COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL ORIGINAL (OR ANOTHER) DATA NETWORK
514
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND ALTERNATE DATA NETWORK
515
SUBJECT TO PRIOR USER INSTRUCTION, BRIDGING SOFTWARE REDIALS DATA NETWORK FOR RECONFIGURED TERMINAL COMMUNICATION PARAMETERS
516
END

FIG. 6A

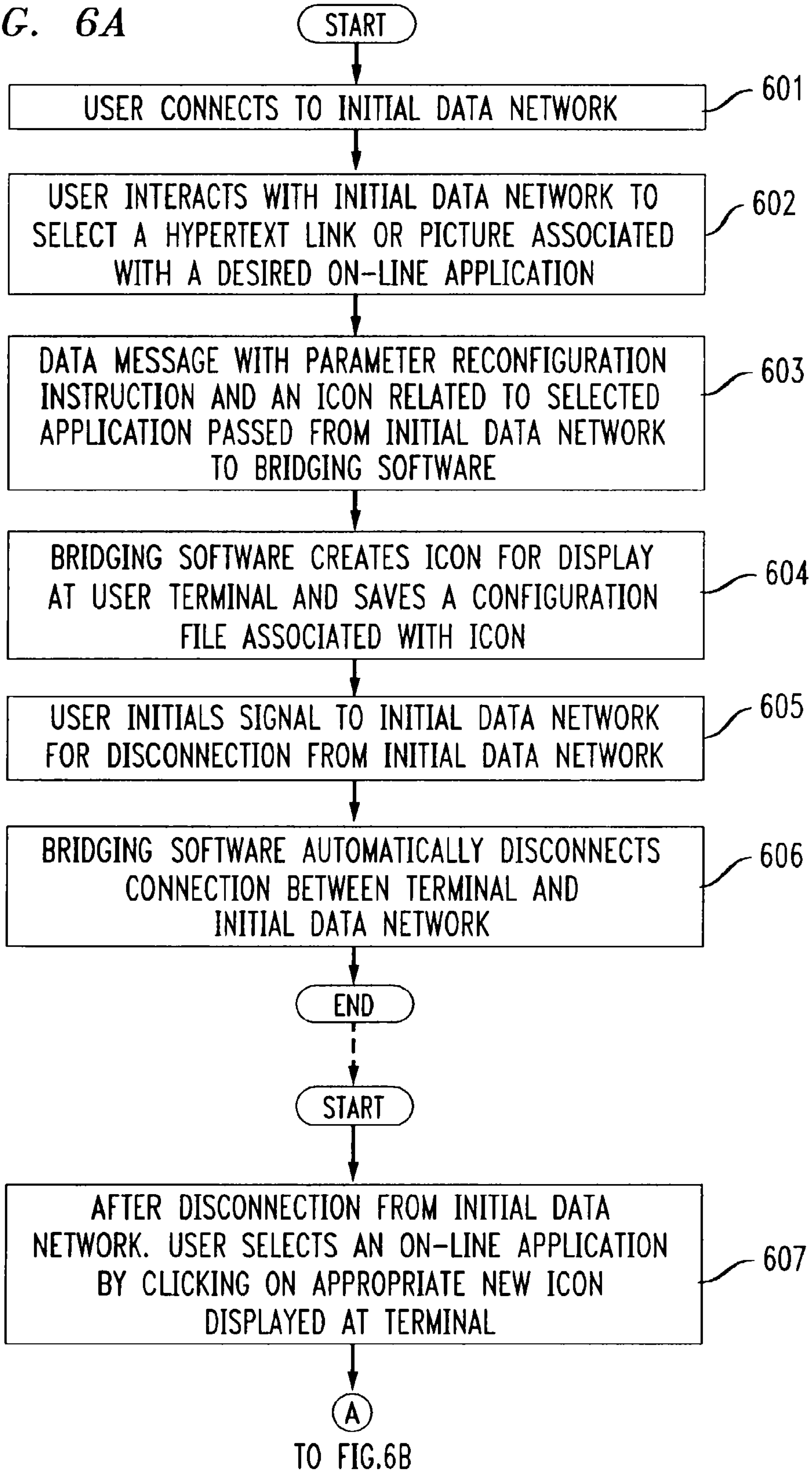

START
USER CONNECTS TO INITIAL DATA NETWORK
601
USER INTERACTS WITH INITIAL DATA NETWORK TO SELECT A HYPERTEXT LINK OR PICTURE ASSOCIATED WITH A DESIRED ON-LINE APPLICATION
602
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTION AND AN ICON RELATED TO SELECTED APPLICATION PASSED FROM INITIAL DATA NETWORK TO BRIDGING SOFTWARE
603
BRIDGING SOFTWARE CREATES ICON FOR DISPLAY AT USER TERMINAL AND SAVES A CONFIGURATION FILE ASSOCIATED WITH ICON
604
USER INITIALS SIGNAL TO INITIAL DATA NETWORK FOR DISCONNECTION FROM INITIAL DATA NETWORK
605
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND INITIAL DATA NETWORK
606
END
START
AFTER DISCONNECTION FROM INITIAL DATA NETWORK. USER SELECTS AN ON-LINE APPLICATION BY CLICKING ON APPROPRIATE NEW ICON DISPLAYED AT TERMINAL
607
A
TO FIG.6B

FIG. 6B

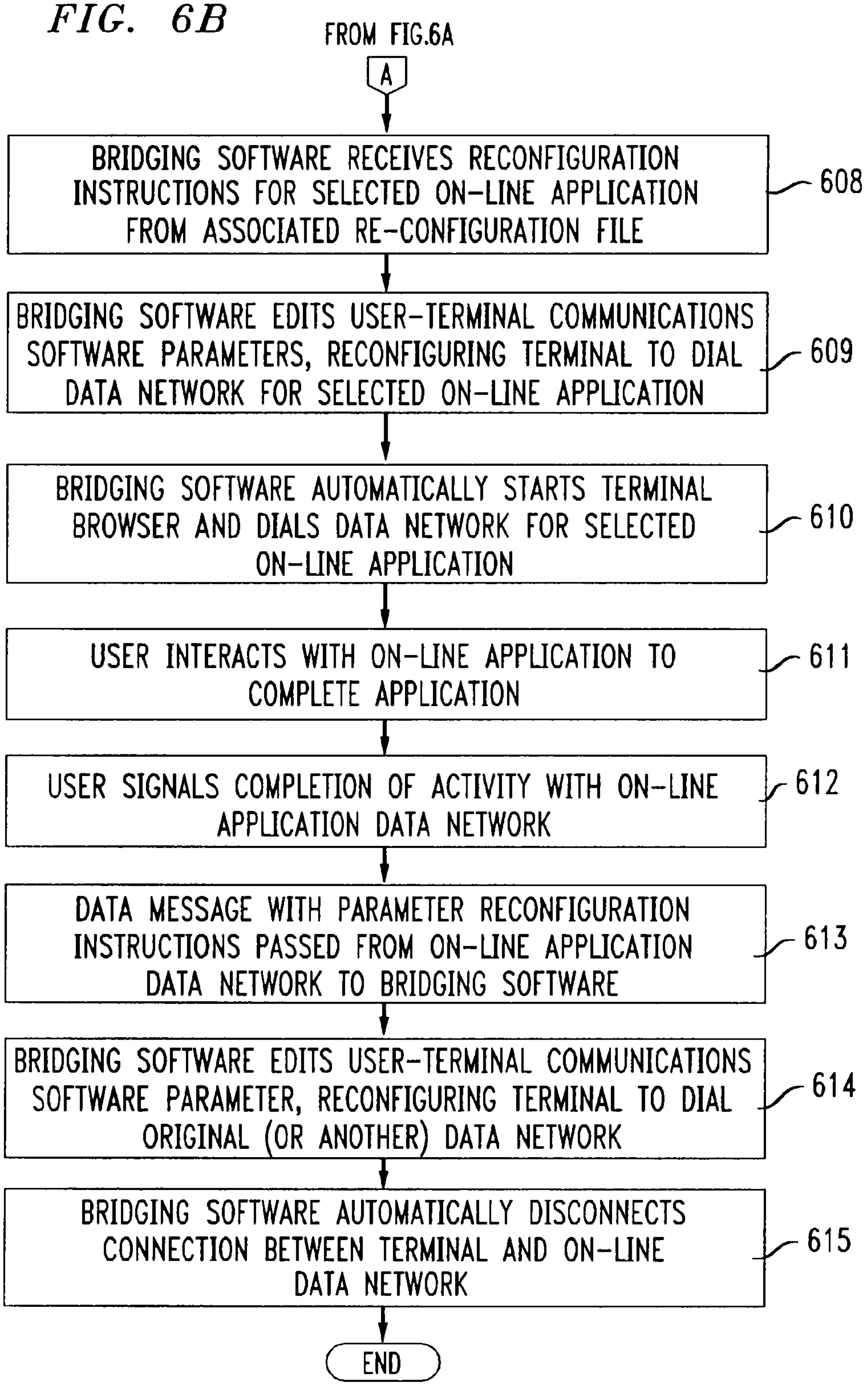

FROM FIG.6A
A
BRIDGING SOFTWARE RECEIVES RECONFIGURATION INSTRUCTIONS FOR SELECTED ON-LINE APPLICATION FROM ASSOCIATED RE-CONFIGURATION FILE
608
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL DATA NETWORK FOR SELECTED ON-LINE APPLICATION
609
BRIDGING SOFTWARE AUTOMATICALLY STARTS TERMINAL BROWSER AND DIALS DATA NETWORK FOR SELECTED ON-LINE APPLICATION
610
USER INTERACTS WITH ON-LINE APPLICATION TO COMPLETE APPLICATION
611
USER SIGNALS COMPLETION OF ACTIVITY WITH ON-LINE APPLICATION DATA NETWORK
612
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM ON-LINE APPLICATION DATA NETWORK TO BRIDGING SOFTWARE
613
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETER, RECONFIGURING TERMINAL TO DIAL ORIGINAL (OR ANOTHER) DATA NETWORK
614
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND ON-LINE DATA NETWORK
615
END

FIG. 7A

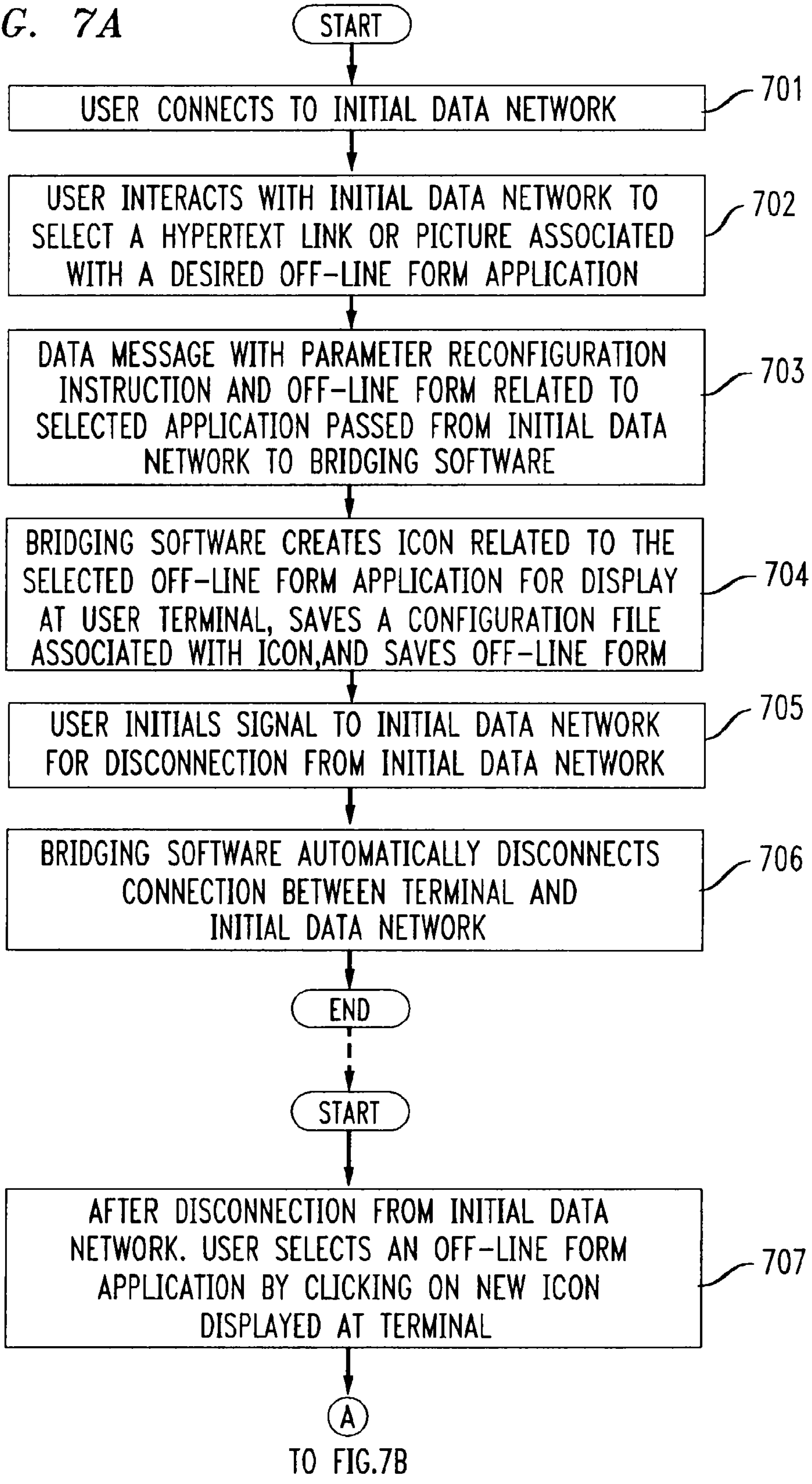
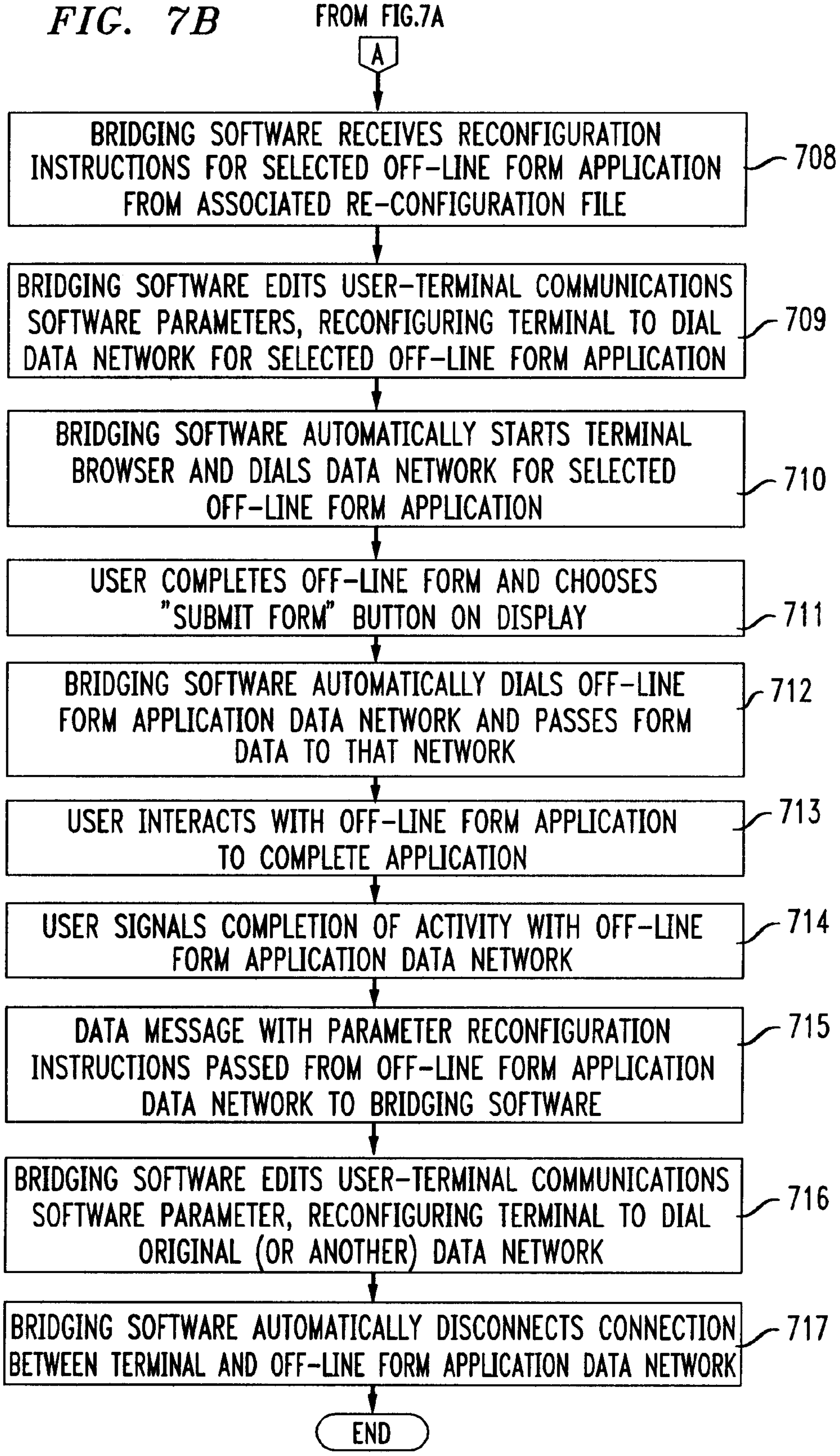

START
USER CONNECTS TO INITIAL DATA NETWORK
701
USER INTERACTS WITH INITIAL DATA NETWORK TO SELECT A HYPERTEXT LINK OR PICTURE ASSOCIATED WITH A DESIRED OFF-LINE FORM APPLICATION
702
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTION AND OFF-LINE FORM RELATED TO SELECTED APPLICATION PASSED FROM INITIAL DATA NETWORK TO BRIDGING SOFTWARE
703
BRIDGING SOFTWARE CREATES ICON RELATED TO THE SELECTED OFF-LINE FORM APPLICATION FOR DISPLAY AT USER TERMINAL, SAVES A CONFIGURATION FILE ASSOCIATED WITH ICON,AND SAVES OFF-LINE FORM
704
USER INITIALS SIGNAL TO INITIAL DATA NETWORK FOR DISCONNECTION FROM INITIAL DATA NETWORK
705
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND INITIAL DATA NETWORK
706
END
START
AFTER DISCONNECTION FROM INITIAL DATA NETWORK. USER SELECTS AN OFF-LINE FORM APPLICATION BY CLICKING ON NEW ICON DISPLAYED AT TERMINAL
707
A
TO FIG.7B

FIG. 7B


FROM FIG.7A
A
BRIDGING SOFTWARE RECEIVES RECONFIGURATION INSTRUCTIONS FOR SELECTED OFF-LINE FORM APPLICATION FROM ASSOCIATED RE-CONFIGURATION FILE
708
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETERS, RECONFIGURING TERMINAL TO DIAL DATA NETWORK FOR SELECTED OFF-LINE FORM APPLICATION
709
BRIDGING SOFTWARE AUTOMATICALLY STARTS TERMINAL BROWSER AND DIALS DATA NETWORK FOR SELECTED OFF-LINE FORM APPLICATION
710
USER COMPLETES OFF-LINE FORM AND CHOOSES "SUBMIT FORM" BUTTON ON DISPLAY
711
BRIDGING SOFTWARE AUTOMATICALLY DIALS OFF-LINE FORM APPLICATION DATA NETWORK AND PASSES FORM DATA TO THAT NETWORK
712
USER INTERACTS WITH OFF-LINE FORM APPLICATION TO COMPLETE APPLICATION
713
USER SIGNALS COMPLETION OF ACTIVITY WITH OFF-LINE FORM APPLICATION DATA NETWORK
714
DATA MESSAGE WITH PARAMETER RECONFIGURATION INSTRUCTIONS PASSED FROM OFF-LINE FORM APPLICATION DATA NETWORK TO BRIDGING SOFTWARE
715
BRIDGING SOFTWARE EDITS USER-TERMINAL COMMUNICATIONS SOFTWARE PARAMETER, RECONFIGURING TERMINAL TO DIAL ORIGINAL (OR ANOTHER) DATA NETWORK
716
BRIDGING SOFTWARE AUTOMATICALLY DISCONNECTS CONNECTION BETWEEN TERMINAL AND OFF-LINE FORM APPLICATION DATA NETWORK
717
END

(12) UK Patent Application (19) GB (11) 2 317 792 (13) A

(43) Date of A Publication 01.04.1998

(21) Application No 9719816.2

(22) Date of Filing 17.09.1997

(30) Priority Data
(31) 08715343 (32) 18.09.1996 (33) US
08715668 18.09.1996

(71) Applicant(s)
Secure Computing Corporation

(Incorporated in USA - Delaware)

2675 Long Lake Road, Roseville,
Minnesota 55113-2536, United States of America

(72) Inventor(s)
Spence Minear
Edward B Stockwell
Troy De Jongh

(74) Agent and/or Address for Service
Beresford & Co
2-5 Warwick Court, High Holborn, LONDON,
WC1R 5DJ, United Kingdom

(51) INT CL^6
H04L 9/00

(52) UK CL (Edition P)
H4P PPEB
U1S S2124 S2209

(56) Documents Cited
WO 97/26735 A1 WO 97/26734 A1 WO 97/26731 A1
WO 97/23972 A1 WO 97/13340 A1

(58) Field of Search
UK CL (Edition P) H4P PDCSA PDCSC PPEB
INT CL^6 H04L 9/00 9/32 29/06 29/08
Online: WPI, INSPEC

(54) **Virtual Private Network for encrypted firewall**

(57) A system (10) for regulating the flow of messages through a firewall (18) having a network protocol stack, wherein the network protocol stack includes an Internet Protocol (IP) layer where if the message is not encrypted, it passes the unencrypted message up the network protocol stack to an application level proxy (50), and if the message is encrypted, it decrypts the message and passes the decrypted message up the network protocol stack to the application level proxy. The step of decrypting the message includes the step of executing a process at the IP layer to decrypt the message.

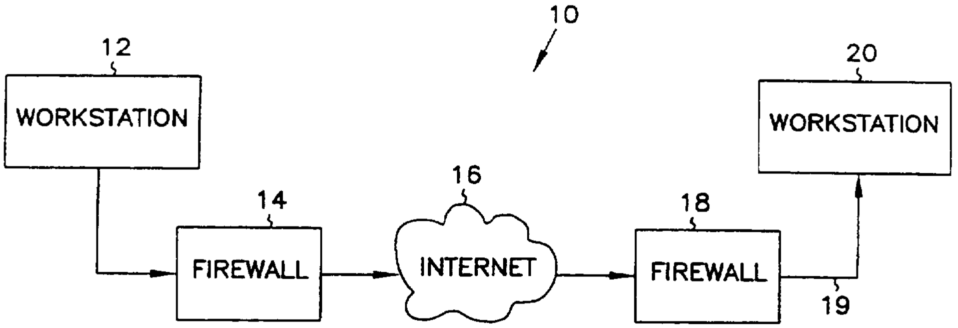


FIG. 1

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

GB 2 317 792 A

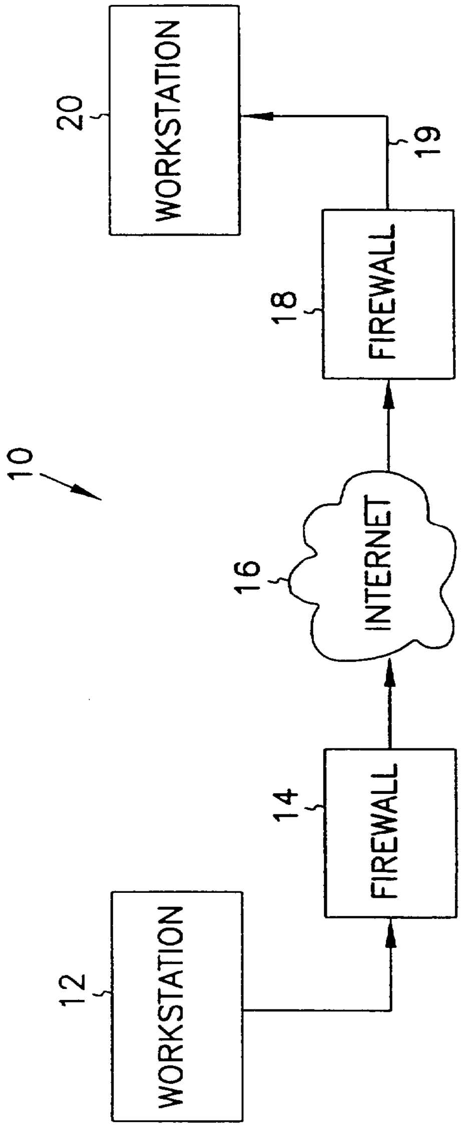
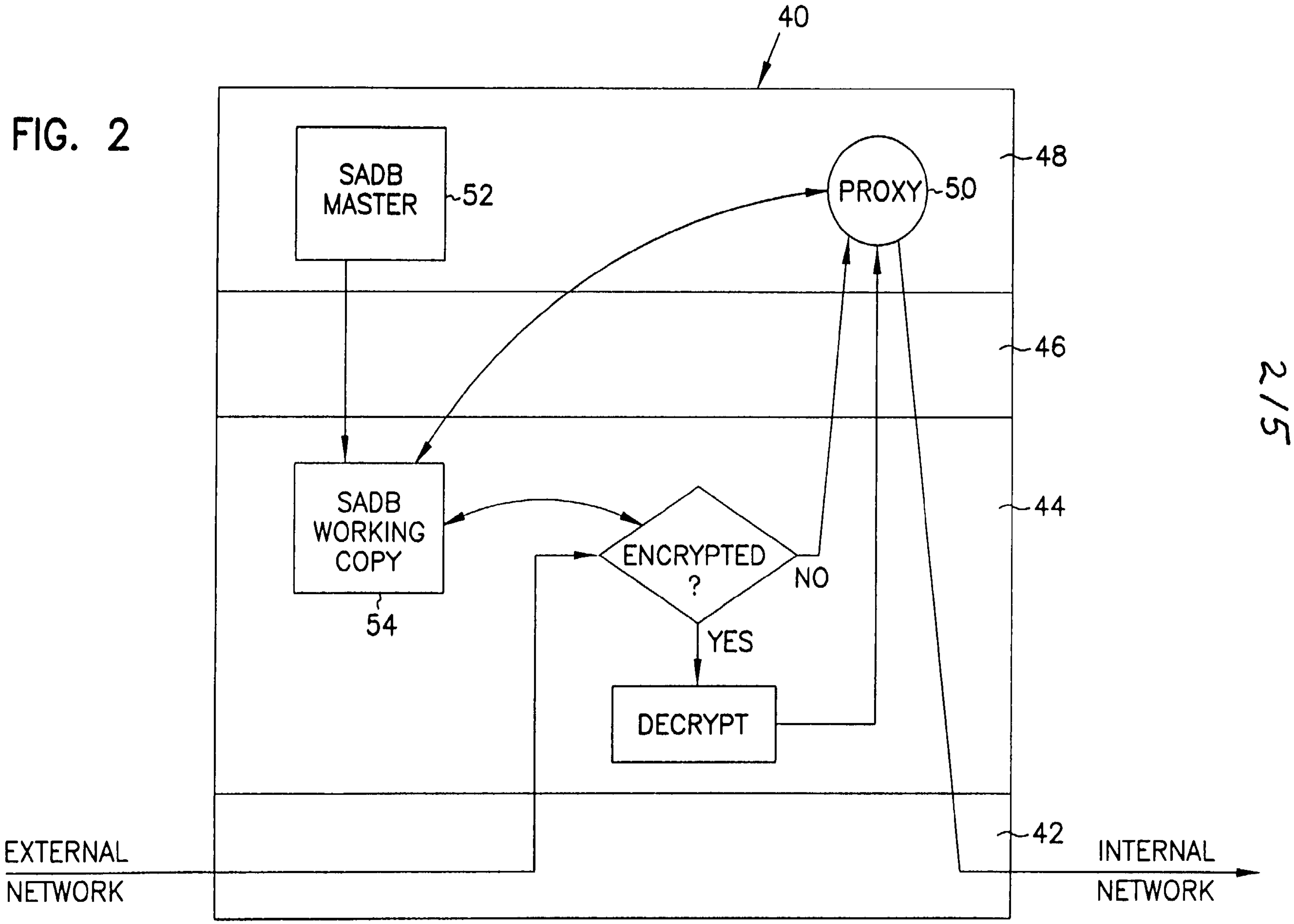

10
WORKSTATION
12
FIREWALL
14
INTERNET
16
FIREWALL
18
19
WORKSTATION
20

FIG. 1

FIG. 2


40
SADB MASTER
52
PROXY
50
48
46
SADB WORKING COPY
54
44
ENCRYPTED ?
NO
YES
DECRYPT
42
EXTERNAL NETWORK
INTERNAL NETWORK

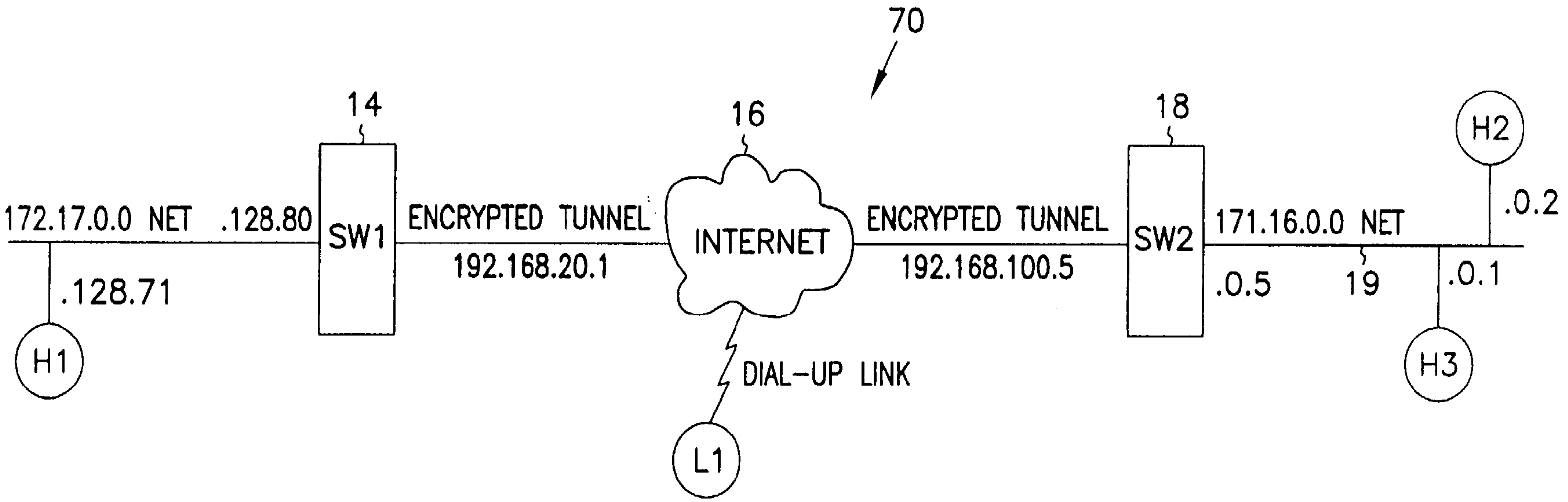

70
14
16
18
H2
H1
H3
L1
172.17.0.0 NET
.128.80
.128.71
SW1
ENCRYPTED TUNNEL
192.168.20.1
INTERNET
DIAL-UP LINK
ENCRYPTED TUNNEL
192.168.100.5
SW2
171.16.0.0 NET
.0.5
19
.0.2
.0.1

FIG. 3

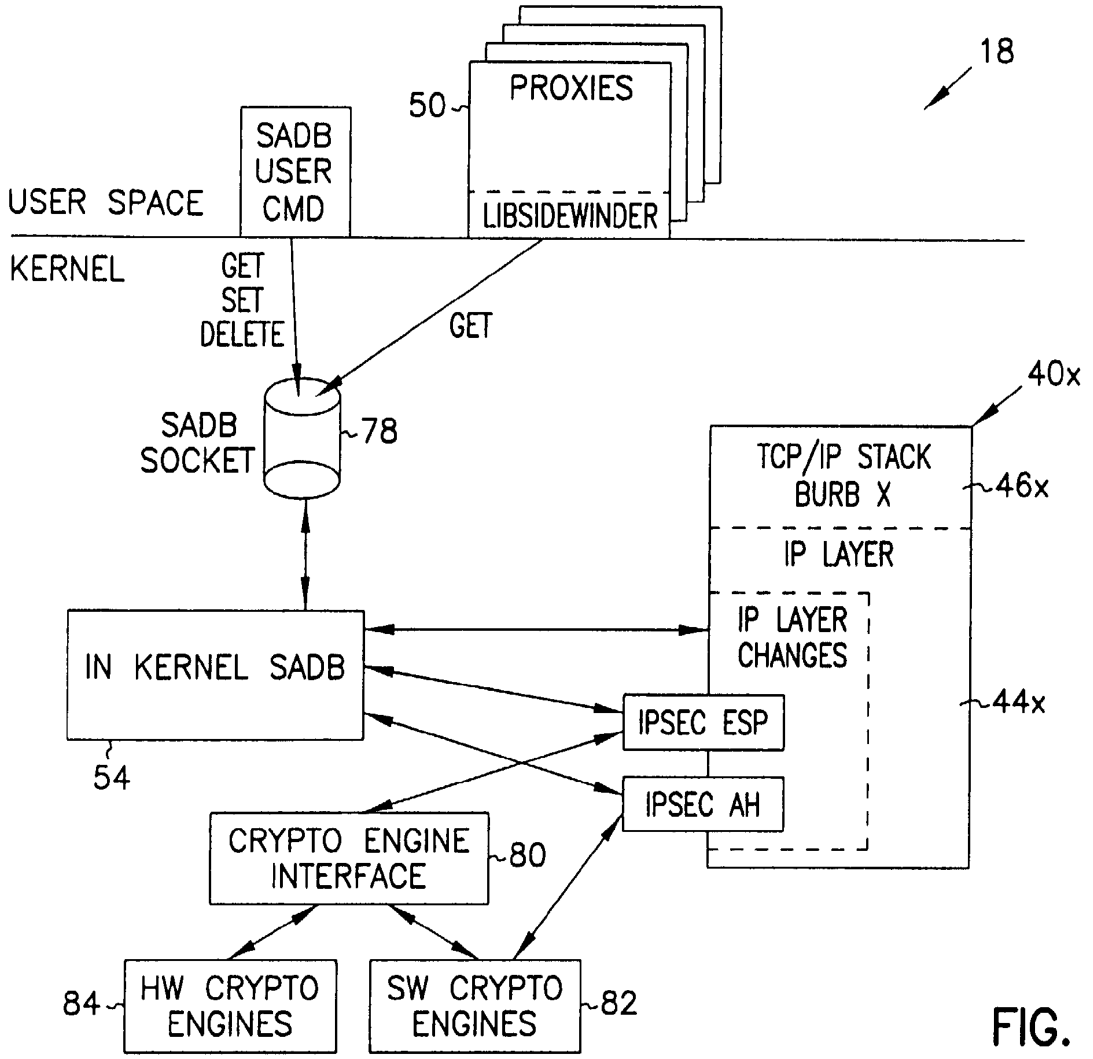

18
USER SPACE
KERNEL
SADB USER CMD
50
PROXIES
LIBSIDEWINDER
GET SET DELETE
GET
SADB SOCKET
78
40x
TCP/IP STACK BURB X
46x
IP LAYER
IP LAYER CHANGES
44x
IN KERNEL SADB
54
IPSEC ESP
IPSEC AH
CRYPTO ENGINE INTERFACE
80
HW CRYPTO ENGINES
84
SW CRYPTO ENGINES
82

FIG. 4

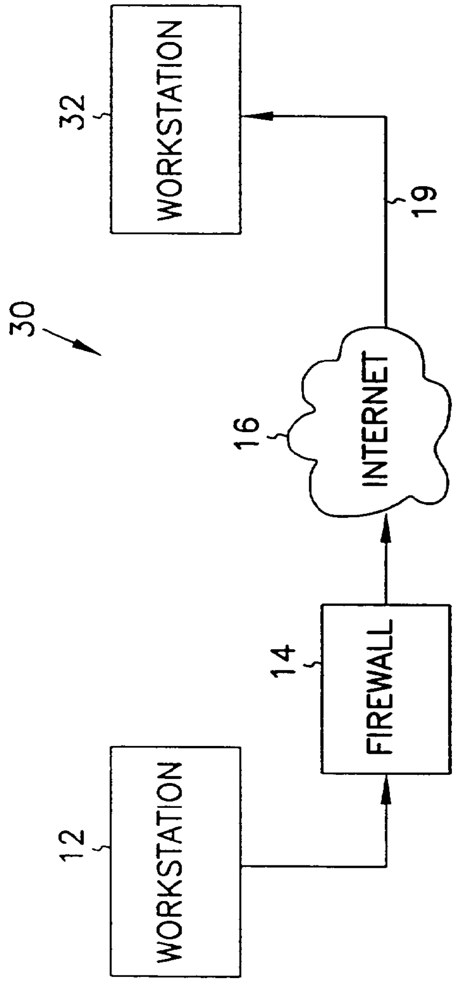

30
WORKSTATION
12
FIREWALL
14
INTERNET
16
19
WORKSTATION
32

FIG. 5

2317792

VIRTUAL PRIVATE NETWORK ON APPLICATION GATEWAY

Background of the Invention

Field of the Invention

The present invention pertains generally to network communications, and in particular to a system and method for securely transferring information between firewalls over an unprotected network.

Background Information

Firewalls have become an increasingly important part of network design. Firewalls provide protection of valuable resources on a private network while allowing communication and access with systems located on an unprotected network such as the Internet. In addition, they operate to block attacks on a private network arriving from the unprotected network by providing a single connection with limited services. A well designed firewall limits the security problems of an Internet connection to a single firewall computer system. This allows an organization to focus their network security efforts on the definition of the security policy enforced by the firewall. An example of a firewall is given in "SYSTEM AND METHOD FOR PROVIDING SECURE INTERNETWORK SERVICES" by Boebert et al. (PCT Published Application No. WO 96/13113, published on May 2, 1996), the description of which is hereby incorporated by reference. Another description of a firewall is provided by Dan Thomsen in "Type Enforcement: the new security model", *Proceedings: Multimedia: Full-Service Impact on Business, Education, and the Home*, SPIE Vol. 2617, p. 143, August 1996. Yet another such system is described in "SYSTEM AND METHOD FOR ACHIEVING NETWORK SEPARATION" by Gooderum et al. (PCT Published Application No. WO 97/29413, published on August 14, 1997), the description of which is hereby incorporated by reference. All the above systems are examples of application level gateways. Application level gateways use proxies or other such mechanisms operating at the application layer to process traffic through the firewall. As such, they can review not only the message traffic but also message content. In addition, they provide authentication and identification services, access control and auditing.

Data to be transferred on unprotected networks like the Internet is susceptible to electronic eavesdropping and accidental (or deliberate) corruption. Although a firewall can protect data within a private network from attacks launched from the unprotected network, even that data is vulnerable to both eavesdropping and corruption when transferred from the private network to an external machine. To address this danger, the Internet Engineering Task Force (IETF) developed a standard for protecting data transferred between firewalls over an unprotected network. The Internet Protocol Security (IPSEC) standard calls for encrypting data before it leaves the first firewall, and then decrypting the data when it is received by the second firewall. The decrypted data is then delivered to its destination, usually a user workstation connected to the second firewall. For this reason IPSEC encryption is sometimes called *firewall-to-firewall encryption* (FFE) and the connection between a workstation connected to the first firewall and a client or server connected to the second firewall is termed a *virtual private network*, or VPN.

The two main components of IPSEC security are data encryption and sender authentication. Data encryption increases the cost and time required for the eavesdropping party to read the transmitted data. Sender authentication ensures that the destination system can verify whether or not the encrypted data was actually sent from the workstation that it was supposed to be sent from. The IPSEC standard defines an encapsulated payload (ESP) as the mechanism used to transfer encrypted data. The standard defines an authentication header (AH) as the mechanism for establishing the sending workstation's identity.

Through the proper use of encryption, the problems of eavesdropping and corruption can be avoided; in effect, a protected connection is established from the internal network connected to one firewall through to an internal network connected to the second firewall. In addition, IPSEC can be used to provide a protected connection to an external computing system such as a portable personal computer.

IPSEC encryption and decryption work within the IP layer of the network protocol stack. This means that all communication between two IP addresses will be protected because all interfirewall communication must go through the IP layer. Such an approach is preferable over encryption and decryption at higher levels in the network protocol stack since when encryption is performed at layers higher than the IP layer more work is required to ensure that all supported communication is properly protected. In addition, since IPSEC encryption is handled below the Transport layer, IPSEC can encrypt data sent by any application. IPSEC therefore becomes a transparent add-on to such protocols as TCP and UDP.

Since, however, IPSEC decryption occurs at the IP layer, it can be difficult to port IPSEC to an application level gateway while still maintaining control at the proxy over authentication, message content, access control and auditing. Although the IPSEC specification in RFC 1825 suggests the use of a mandatory access control mechanism in a multi-level secure (MLS) network to compare a security level associated with the message with the security level of the receiving process, such an approach provides only limited utility in an application level gateway environment. In fact, implementations on application level gateways to date have simply relied on the fact that the message was IPSEC-encrypted as assurance that the message is legitimate and have simply decoded and forwarded the message to its destination. This creates, however, a potential chink in the firewall by assuming that the encrypted communication has access to all services.

What is needed is a method of handling IPSEC messages within an application level gateway which overcomes the above deficiencies. The method should allow control over access by an IPSEC connection to individual services within the internal network.

Summary of the Invention

The present invention is a system and method for regulating the flow of messages through a firewall having a network protocol stack, wherein the network protocol stack includes an Internet Protocol (IP) layer, the method comprising the steps of determining, at the IP layer, if a message is encrypted, if the message is not encrypted, passing the unencrypted message up the network protocol stack to an application level proxy, and if the message is encrypted, decrypting the message and passing the decrypted message up the network protocol stack to the application level proxy, wherein the step of decrypting the message includes the step of executing a procedure at the IP layer to decrypt the message.

According to another aspect of the present invention, a system and method is described for authenticating the sender of a message within a computer system having a network protocol stack, wherein the network protocol stack includes an Internet Protocol (IP) layer, the method comprising the steps of determining, at the IP layer, if the message is encrypted, if the message is encrypted, decrypting the message, wherein the step of decrypting the message includes the step of executing a procedure at the IP layer to decrypt the message, passing the decrypted message up the network protocol stack to an application level proxy, determining an authentication protocol appropriate for the message, and executing the authentication protocol to authenticate the sender of the message.

Brief Description of the Drawings

In the following detailed description of example embodiments of the invention, reference is made to the accompanying drawings which form a part hereof, and which is shown by way of illustration only, specific embodiments in which the invention may be practiced. It is to be understood that other embodiments may be utilized and structural changes may be made without departing from the scope of the present invention.

In the drawings, where like numerals refer to like components throughout the several views:

Figure 1 is a functional block diagram of an application level gateway-implemented firewall-to-firewall encryption scheme according to the present invention;

Figure 2 is a block diagram showing access control checking of both encrypted and unencrypted messages in network protocol stack according to the present invention;

Figure 3 is a block diagram of a representative application level gateway-implemented firewall-to-firewall encryption scheme;

Figure 4 is a block diagram of one embodiment of a network-separated protocol stack implementing IPSEC according to the present invention; and

Figure 5 is a functional block diagram of a firewall-to-workstation encryption scheme according to the present invention.

Description of the Preferred Embodiments

In the following detailed description of the preferred embodiment, references made to the accompanying drawings which form a part hereof, and in which is shown by way of illustration specific preferred embodiments in which the invention may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the invention, and it is to be understood that other embodiments may be utilized and that structural, logical, physical, architectural, and electrical changes may be made without departing from the spirit and scope of the present invention. The following detailed description is, therefore, not to be taken in a limiting sense, and the scope of the present invention is defined only by the appended claims and their equivalents.

A system 10 which can be used for firewall-to-firewall encryption (FFE) is shown in Figure 1. In Figure 1, system 10 includes a workstation 12 communicating through a firewall 14 to an unprotected network 16 such as the Internet. System 10 also includes a workstation 20 communicating through a firewall 18 to unprotected network 16. In one embodiment, firewall 18 is an application level gateway.

As noted above, IPSEC encryption and decryption work within the IP layer of the network protocol stack. This means that all communications between two IP addresses will be protected because all interfirewall communication must pass through the IP layer. IPSEC takes the standard

Internet packet and converts it into a carrier packet. The carrier packet is designed to do two things: to conceal the contents of the original packet (encryption) and to provide a mechanism by which the receiving firewall can verify the source of the packet (authentication). In one embodiment of the present invention, each IPSEC carrier packet includes both an authentication header used to authenticate the sending machine and an encapsulated payload containing encrypted data. The authentication header and the encapsulated payload features of IPSEC can, however, be used independently. As required in RFC 1825, DES-CBC is provided for use in encrypting the encapsulated payload while the authentication header uses keyed MD5.

To use IPSEC, you must create a *security association* (SA) for each destination IP address. In one embodiment, each SA contains the following information:

- Security Parameters Index (SPI) - The index used to find a SA on receipt of an IPSEC datagram.
- Destination IP address - The address used to find the SA and trigger use of IPSEC processing on output.
- The peer SPI - The SPI value to put on a IPSEC datagram on output.
- The peer IP address - The destination IP address to be put into the packet header if IPSEC Tunnel mode is used.
- The Encryption Security Payload (ESP) algorithm to be used.
- The ESP key to used for decryption of input datagrams.
- The ESP key to used for encryption of output datagrams.
- The authentication (AH) algorithm to be used.
- The AH key to be used for validation of input packets.
- The AH key to be used for generation of the authentication data for output datagrams.

The combination of a given Security Parameter Index and Destination IP address uniquely identifies a particular "Security Association." In one embodiment, the sending firewall uses the sending userid and Destination Address to select an appropriate Security Association (and hence SPI value). The receiving firewall uses the combination of SPI value and Source address to obtain the appropriate Security Association.

A security association is normally one-way. An authenticated communications session between two firewalls will normally have two Security Parameter Indexes in use (one in each direction). The combination of a particular Security Parameter Index and a particular Destination Address uniquely identifies the Security Association.

More information on the specifics of an IPSEC FFE implementation can be obtained from the standards developed by the IPSEC work group and documented in *Security Architecture for IP* (RFC 1825) and in RFC's 1826-1829.

When a datagram is received from unprotected network 16 or is to be transmitted to a destination across unprotected network 16, the firewall must be able to determine the algorithms, keys, etc. that must be used to process the datagram correctly. In one embodiment, this information is obtained via a security association lookup. In one such embodiment, the lookup routine is passed several arguments: the source IP address if the datagram is being received from network 16 or the destination IP address if the datagram is to be transmitted across network 16, the SPI, and a flag that is used to indicate whether the lookup is being done to receive or transmit a datagram.

When an IPSEC datagram is received by firewall 18 from unprotected network 16, the SPI and source IP address are determined by looking in the datagram. In one embodiment a Security Association Database (SADB) stored within firewall 18 is searched for the entry with a matching SPI. In one such embodiment, security associations can be set up based on network address as well as a more granular host address. This allows the network administrator to create a security association between two firewalls with only a couple of lines in a configuration file on each machine. For such embodiments, the entry in the Security Association Database that has both the matching SPI and the longest address match is selected as the SA entry. In another such embodiment, each SA has a prefix length value associated with the address. An address match on a SA entry means that the addresses match for the number of bits specified by the prefix length value.

There are two exceptions to this search process. First, when an SA entry is set marked as being dynamic it implies that the user of this SA may not have a fixed IP address. In this case the match is fully determined by the SPI value. Thus it is necessary that the SPI values for such SA entries be unique in the SADB. The second exception is for SA entries marked as tunnel mode entries. In this case it is normally the case that the sending entity will hide its source address so that all that is visible on the public wire is the destination address. In this case, like in the case where the SA entries are for dynamic IP addresses, the search is done exclusively on the basis of the SPI.

When transmitting a datagram across unprotected network 16 the SADB is searched using only the destination address as an input. In this case the entry which has the longest address match is selected and returned to the calling routine.

In one embodiment, if firewall 18 receives datagrams which are identified as either an IP_PROTO_IPSEC_ESP or IP_PROTO_IPSEC_AH protocol datagram, there must be a corresponding SA in the SADB or else firewall 18 will drop the packet and an audit message will be generated. Such an occurrence might indicate a possible attack or it might simply be a symptom of an erroneous key entry in the Security Association Database.

In a system such as system 10, application level gateway firewall 18 acts as a buffer between unprotected network 16 and workstations such as workstation 20. Messages coming from unprotected network 16 are reviewed and a determination is made as to whether execution of an authentication and identification protocol is warranted. In contrast to previous systems, system 10 also performs this same determination on IPSEC-encrypted messages. If desired, the same authentication and identification can be made on messages to be transferred from workstation 20 to unprotected network 16. Figure 2 illustrates one way of authenticating both encrypted and unencrypted messages in a system such as system 10.

In the system of Figure 2 a network protocol stack 40 includes a physical layer 42, an Internet protocol (IP) layer 44, a Transport layer 46 and an application layer 48. Such a protocol stack exists, for instance on application level gateway firewall 18 of Figure 1. An application executing in application layer 48 can communicate to an application executing on another system by preparing a message and transmitting it through one of the existing transport services executing on transport layer 46. Transport layer 46 in turn uses a process executing in IP layer 44 to continue the transfer. Physical layer 42 provides the software needed to transfer data through the communication hardware (e.g., a network interface card or a modem). As noted above, IPSEC executes within IP layer 44. Encryption and authentication is transparent to the host as long as the network administrator has the Security Association Database correctly configured and a key management mechanism is in place on the firewall.

In application level gateway firewall 18, a proxy 50 operating within application layer 48 processes messages transferred between internal and external networks. All network-to-network traffic must pass through one of the proxies within application layer 48 before being the transfer across networks is allowed. A message arriving from external network 16 is examined at IP layer 44 and an SADB is queried to determine if the source address and SPI are associated with an SA. In the embodiment shown in Figure 2, an SADB Master copy 52 is maintained in persistent memory at application layer 48 while a copy 54 of SADB is maintained in volatile memory within the kernel. If the message is supposed to be encrypted, the message is decrypted based on the algorithm and key associated with the particular SA and the message is transferred up through transport layer 46 to proxy 50. Proxy 50 examines the source and destination addresses and the type of service desired and decides whether authentication of the sender is warranted. If so, proxy 50 initiates an authentication protocol. The protocol may be as simple as requesting a user name and password or it may include a challenge/response authentication process. Proxy 50 also looks to see whether the message coming in was encrypted or not and may factor that into whether a particular type of authentication is needed. In Telnet, for instance, user name/password authentication may be sufficient for an FFE link while the security policy may dictate that a more stringent challenge/response protocol is needed for unencrypted links. In that case, proxy 50 will be a Telnet proxy and it will base its authentication protocol on whether the link was encrypted or not.

Since IPSEC executes within IP layer 44 there is no need for host firewalls to update their applications. Users that already have IPSEC available on their own host machine will, however, have to request that the firewall administrator set up SA's in the SADB for their traffic.

In the embodiment shown in Figure 2, a working copy 54 of the Security Association Database consisting of all currently active SA's is kept resident in memory for ready access by IP layer processing as datagrams are received and transmitted. In addition, a working master copy 52 of the SADB is maintained in a file in nonvolatile memory. During system startup and initialization processing the content of all of the required SA's in master SADB 52 is added to the working copy 54 stored in kernel memory.

In one embodiment, firewall 18 maintains different levels of security on internal and external network interfaces. It is desirable for a firewall to have different levels of security on both the internal and external interfaces. In one embodiment, firewall 18 supports three different levels, numbered 0 through 2. These levels provide a simple policy mechanism that controls permission for both in-bound and out-bound packets.

- Level 0 - do not allow any in-bound or out-bound traffic unless there is a security association between the source and destination.

- Level 1 - Allow both in-bound and out-bound non-IPSEC traffic but force the use of IPSEC if a SA exists for the address. (To support this firewall 18 must look for a SA for each in-bound datagram.)
- Level 2 - allow NULL security associations to exist. NULL associations are just like normal security associations, except no encryption or authentication transform is performed on in-bound or out-bound packets that correspond to this NULL association. With Level 2 enabled, the machine will still receive unprotected traffic, but it will not transmit unless Level 1 is enabled.

The default protection level established when the Security Association Database (SADB) is initialized at boot time is 1 for in-bound traffic and 2 for out-bound traffic.

An Access Control List, or ACL, is a list of rules that regulate the flow of Internet connections through a firewall. These rules control how a firewall's servers and proxies will react to connection attempts. When a server or proxy receives an incoming connection, it performs an ACL check on that connection.

An ACL check compares a set of parameters associated with the connection against a list of ACL rules. The rules determine whether the connection is allowed or denied. A rule can also have one or more side effects. A side effect causes the proxy to change its behavior in some fashion. For example, a common side effect is to redirect the destination IP address to an alternate machine. In addition to IP connection attempts, ACL checks can also made on the console logins and on logins made from serial ports. Finally, ACL checks can also be made on behalf of IP access devices, such as a Cisco box, through the use of the industry standard TACACS+ protocol.

In one embodiment, the ACL is managed by an acld daemon running in the kernel of firewalls 10 and 30. The acld daemon receives two types of requests, one to query the ACL and one to administer it. In one such embodiment, the ACL is stored in a relational database such as the Oracle database for fast access. By using such a database, query execution is asynchronous and many queries can be executing concurrently. In addition, these types of databases are designed to manipulate long lists of rules quickly and efficiently. These qualities ensure that a given query cannot hang up the process that issued the query for any appreciable time (> 1-2 seconds).

In one such embodiment, the database can hold up to 100,000 users and up to 10,000 hosts but can be scaled up to the capacity of the underlying database engine. The results of an ACL check is cached, allowing repeated checks to be turned around very quickly.

Applications on firewalls 10 and 30 can query acld to determine if a given connection attempt should be allowed to succeed. In one embodiment, the types of applications (i.e. "agents") that can make ACL queries can be divided into four classes:

1) Proxies. These allow connections to pass through firewall 10 or 30 in order to provide access to a remote service. They include tnauthp (authenticated telnet proxy), pftp (FTP proxy), httpp (HTTP proxy), and tcpgsp (TCP generic service proxy).

2) Servers. These provide a service on the firewall itself. They include ftpd and httpd.

3) Login agents. Login agent is a program on the firewall that can create a Unix shell. It is not considered a server because it cannot receive IP connections. One example is /usr/bin/login when used to create a dialup session or a console session on firewall 10 or 30. Another example is the command *srole*.

4) Network Access Servers (NAS). NAS is a remote IP access device, typically a dialup box manufactured by such companies as Cisco or Bridge. The NAS usually provides dialup telnet service and may also provide SLIP or PPP service.

Proxies, servers, login agents, and NASes make queries to acld to determine if a given connection attempt should be allowed to succeed. All of the agents except NAS make their queries directly. NAS, because it is remote, must communicate via an auxiliary daemon that typically uses an industry standard protocol such as RADIUS or TACACS+. The auxiliary daemon (e.g., tacradd) in turn forwards the query to local acld.

As a side effect of the query, acld tells the agent if authentication is needed. If no authentication is needed, the connection proceeds immediately. Otherwise acld provides (as another side effect) a list of allowed authentication methods that the user can choose from. The agent can present a menu of choices or simply pick the first authentication method by default. Typical authentication methods include plain password, SNK DSS, SDI SecurID, LOCKout DES, and LOCKout FORTEZZA. In one embodiment, the list of allowed authentication methods varies depending on the host name, user name, time of day, or any combination thereof.

In the case of a Level 0 policy, it would be safe to assume that all incoming traffic is encrypted or authenticated. In the case of Levels 1 through 2, a determination must be made whether or not a security association exists for a given peer. Otherwise an application may believe that in-bound traffic has been authenticated when it really has not. (That is why it is necessary to look for an SA on input of each non-IPSEC datagram.)

In one embodiment, a flag which accompanies the message as it is sent from IP layer 44 to proxy 50 indicates whether the incoming message was or was not encrypted. In another embodiment, proxy 50 accesses Security Association Database 54 (the table in the kernel can be queried via an SADB routing socket (PF-SADB)) to determine whether or not a security association exists for a given peer.. The SADB socket is much like a routing socket found in the stock BSD 4.4 kernel (protocol family PF-ROUTE) except that PF-SADB sockets are used to maintain the Security Association Database (SADB) instead of the routing table. Because the private keys used for encryption, decryption, and keyed authentication are stored in this table, access must be strictly prohibited and allowed to only administrators and key management daemons. Care must be taken when allowing user-level daemons access to `/dev/mem` or `/dev/kmem` as well, since the keys are stored in kernel memory and could be exposed with some creative hacking.

In one embodiment, a command-line tool called `sadb` is used to support the generation and maintenance of in-kernel version 54 of SADB. The primary interface between this tool and the SADB is the PF-SADB socket. The kernel provides socket processing to receive client requests to add, update, or change entries in in-kernel SADB 54. As noted above, the default protection level established when the Security Association Database (SADB) is initialized at boot time is 1 for in-bound traffic and 2 for out-bound traffic. This may be changed by the use of the sadb command.

The existing sadb command was derived from the NIST implementation of IPSEC. As noted above, this tool is much like route in that it uses a special socket to pass data structures in and out of the kernel. There are three commands recognized by the sadb command: *get, set, delete*. The following simple shell script supports adding and removing a single SA entry to SADB 54. It shows one embodiment of a parameter order for adding a SA to the SADB.

```
# ! /bin/sh
if [ $# -ne 1 ]
then
     echo "usage: $0 <on>|<off>" >&2
     exit 1
fi
ONOFF=$1

addsa ()
{
IPADDRESS=$2
PEERADDRESS=0.0.0.0
PREFIXLEN=0                # Num of bits, 0 => full 32
bit match
LOCALADDRESS=0.0.0.0
REALADDRESS=0.0.0.0
PORT=0
PROTOCOL=0
UID=0
DESALG=1                   # I = DES-CBC
IVLEN=4                    # bytes
DESKEY=0b0b0b0b0b0b0b0b
DESKEYLEN=8                # bytes
AHALG=1                    # 1 = MD5
AHKEY=30313233343536373031323334353637
AHKEYLEN=16                # bytes
LOCAL_SPI=$1
```

```
PEER_SPI=$1
TUNNEL_MODE=0
AHRESULTLEN=4
COMBINED_MODE=1           # On output, 1 = ESP, then
AH; 0 = AH, then ESP
DYNAMIC_FLAG=0

if [ "$ONOFF" = "on"
then
     ./sadb add dst $IPADDRESS $PREFIXLEN $LOCAL_SPI
$UID $PEERADDRESS $PEER_SPI $TUNNEL_MODE $LOCALADDRESS
$REALADDRESS $PROTOCOL $PORT $DESALG $IVLEN $DESKEYLEN
$DESKEY $DESKEYLEN $DESKEY $AHALG $AHKEYLEN $AHKEY
$AHKEYLEN $AHKEY $AHRESULTLEN $COMBINED_MODE
$DYNAMIC_FLAG
else
     ./sadb delete dst $IPADDRESS $LOCAL-SPI
fi
}

#    Get down to work:
addsa 500 172.17.128.115          # number6.sctc.com
```

The current status of in-kernel SADB 54 can be obtained with the sadb command. The get option allows dumping the entire SADB or a single entry. In one embodiment, the complete dump approach uses /dev/kmem to find the information. The information may be presented as follows:

```
# sadb get dst

Local-SPI Address-Family Destination-Addr
Preflx_length UID
     Peer-Address Peer-SPI Transport-Type
     Local-Address Real-Address
     Protocol Port
     ESP_Alg_ID ESP_IVEC_Length
          ESP_Enc_Key_length ESP_Enc_ESP_Key
          ESP_Dec_Key_length ESP_Dec_ESP_Key
     AH_Alg_ID AH_Data_Length
          AH_Gen_Key_Length AH_Gen_Key
          AH_Check_Key_Length AH_Check_Key
     Combined_Mode  Dynamic_Flag
```

```
-----------------------------------------------------
-
500 INET: number6.sctc.com 0 0
     0.0.0.0   500 Transport(0) 0
     0.0.0.0 0.0.0.0
     None None
     DES/CBC-RFC1829(1) 4
          8 0b0b0b0b0b0b0b0b
          8 0b0b0b0b0b0b0b0b
     MD5-RFC1828(1) 4
          16 30313233343536373031323334353637
          16 30313233343536373031323334353637
     ESP+AH(1) 0
501 INET: spokes.sctc.com 0 0
     0.0.0.0   501 Transport(0) 0
     0.0.0.0.0.0.0.0
     None None
     DES/CBC-RFC1829(1) 4
          8 0b0b0b0b0b0b0b0b
          8 0b0b0b0b0b0b0b0b
     MD5-RFC1828(1) 4
          16 30313233343536373031323334353637
          16 30313233343536373031323334353637
     ESP+AH(1) 0

End of list.
```

When a new entry is added to in-kernel SADB 54, the add process first checks to see that no existing entry will match the values provided in the new entry. If no match is found then the entry is added to the end of the existing SADB list.

To illustrate the use and administration of an FFE, we'll go through an example using FFE 70 in Figure 3. Firewalls 14 and 18 are both application level gateway firewalls implemented according to the present invention. Workstations H2 and H3 both want to communicate with Hl. For the administrator of firewalls 14 and 18, this is easy to accomplish. The administrator sets up a line something like this (we'll only show the IP address part and SPI parts of the SA, since they're the trickiest values to configure. Also, assume that we are using tunnel mode):

```
#  Hypothetical SW1 Config File
```

```
#
#  Fields are laid out in the following manner:
#  srcaddrornet= localSPI= peeraddr= peerSPI=
realsrcaddr= localaddr= key=

# The following entry sets up a tunnel between hosts
behind SW1
# and hosts behind SW2.
src=172.16.0.0 localSPI=666 peer=192.168.100.5
peerSPI=777 \
     realsrcaddr=192.168.100.5 localaddrs=0.0.0.0
     key=0xdeadbeeffadebabe

#  Hypothetical SW2 Config File
#
#  Fields are laid out in the following manner:
#  srcaddrornet= localSPI= peeraddr= peerSPI=
     realsrcaddr= localaddr= key=

#  The following entry sets up a tunnel between hosts
behind SW1 and
#  hosts behind SW2.
src=172.17.0.0 localSPI=777 peer=192.168.20.1
peerSPI=666 \
     realsrcaddr=192.168.20.1 localaddr=0.0.0.0 \
     key=0xdeadbeeffadebabe
```

With this setup, all traffic is encrypted using one key, no matter who is talking to whom. For example, traffic from H2 to H1 as well as traffic from H3 to H1 will be encrypted with one key. Although this setup is small and simple, it may not be enough.

What happens if H2 cannot trust H3? In this case, the administrator can set up security associations at the host level. In this case, we have to rely on the SPI field of the SA, since the receiving firewall cannot tell from the datagram header which host behind the sending firewall sent the packet. Since the SPI is stored in IPSEC datagrams, we can do a lookup to obtain its value. Below are the sample configuration files for both firewalls again, but this time, each host combination communicates with a different key. Moreover, H2 excludes H3 from communications with H1, and H3 excludes H2 in the same way.

```
#  Hypothetical SW1 Config File
#
#  Fields are laid out in the following manner:
#  srcaddrornet= localSPI= peeraddr= peerSPI=
realsrcaddr= localaddr= key=

# The following entry sets up a secure link between H2
and H1
src=172.16.0.2 localSPI=666 peer=192.168.100.5
peerSPI=777 \
     realsrcaddr=192.168.100.5
localaddrs=178.17.128.71 \
     key=0x0a0a0a0a0a0a0a0a

# The following entry sets up a secure link between H3
and H1
src=172.16.0.1 localSPI=555 peer=192.168.100.5
peerSPI=888 \
     realsrcaddr=192.168.100.5
localaddrs=178.17.128.71 \
     key=0x0b0b0b0b0b0b0b0b

#  Hypothetical SW2 Config File
#
#  Fields are laid out in the following manner:
#  srcaddrornet= localSPI= peeraddr= peerSPI=
realsrcaddr= localaddr= key=

# The following entry sets up a secure link between H2
and H1
src=172.17.128.71 localSPI=777 peer=192.168.20.1
peerSPI=666 \
     realsrcaddr=192.168.20.1 localaddrs=172.16.0.2 \
     key=0x0a0a0a0a0a0a0a0a

# The following entry sets up a secure link between H3
and H1
src=172.17.128.71 localSPI=888 peer=192.168.20.1
peerSPI=555 \
     realsrcaddr=192.168.20.1 localaddrs=172.16.0.1 \
     key=0x0b0b0b0b0b0b0b0b
```

Figure 4 is a block diagram showing in more detail one embodiment of an IPSEC-enabled application level gateway firewall 18. Application level gateway firewall 18 provides access control checking of both encrypted and unencrypted messages in a more secure environment due to its network-separated architecture. Network separation divides a system into a set of independent regions or burbs, with a domain and a protocol stack assigned to each burb. Each protocol stack 40x has its own independent set of data structures, including routing information and protocol information. A given socket will be bound to a single protocol stack at creation time and no data can pass between protocol stacks 40 without going through proxy space. A proxy 50 therefore acts as the go-between for transfers between domains. Because of this, a malicious attacker who gains control of one of the regions is prevented from being able to compromise processes executing in other regions. Network separation and its application to an application level gateway is described in "SYSTEM AND METHOD FOR ACHIEVING NETWORK SEPARATION", U.S. Application No. 08/599,232, filed February 9, 1996 by Gooderum et al.

In the system shown in Figure 4, the in-bound and out-bound datagram processing of a security association continues to follow the conventions defined by the network separation model. Thus all datagrams received on or sent to a given burb remain in that burb once decrypted. In one such embodiment SADB socket 78 has been defined to have the type 'sadb'. Each proxy 50 that requires access to SADB socket 78 to execute its query as to whether the received message was encrypted must have create permission to the sadb type.

The following is list of specific requirements that a system such as is shown in Figure 4 must provide. Many of the requirements were discussed in the information provided earlier in this document.

1. Firewall applications may query the IPSEC subsystem to determine if traffic with a given address is guaranteed to be encrypted.
2. Receipt of an unencrypted datagram from an address that has a SA results in the datagram being dropped and an audit message being generated.
3. On receipt of encrypted protocol datagrams the SADB searches will be done using the SPI as the primary key. The source address will a secondary key. The SA returned by the search will be the SA which matches the SPI exactly and has the longest match with the address.

4. A search of the SADB for a SPI that finds an entry that is marked as SA for a dynamic IP will not consider the address in the search process.

5. A search of the SADB for a SPI that finds an entry that is marked as a SA for a tunnel mode connection will to consider the address if it is (0.0.0.0) i.e INADDR.

6. On receipt of a non-IPSEC datagram the SADB will be searched for an entry that matches the src address. If a SA is found the datagram will be dropped and an audit message sent.

7. SADB searches on output will be done using the DST address as key. If morc than one SA entry in the SADB has that address the first one with the maximum address match will be returned.

8. The SADB must be structured so that searches are fast regardless if the search is done by SPI or by address.

9. The SADB must provide support for connections to a site with a fixed SPI but changing IP address. SA entries for such connections will be referred to as Dynamic Address Sites, or just Dynamic entries.

10. When a dynamic entry is found by a SPI search, the current datagram's SRC address, which is required to ensure that the return datagrams are properly encrypted, will be recorded in the SA only after the AH checking has passed successfully. (This is because if the address is recorded before AH passes then an attacker can cause return packets of an outgoing connection to be transmitted in the clear.)

11. A failure of an AH check on a dynamic entry results in an audit message.

12. In an embodiment where the firewall requires that all connections use both AH and ESP, on receipt the order should bc AH first ESP second.

13. The processing structure on both input and output should try to minimize the number of SADB required lookups.

Returning to Figure 4, in one embodiment firewall 18 includes a crypto engine interface 80 used to encrypt an IPSEC payload. Crypto engine interface 80 may be connected to a software encryption engine 82 or to a hardware encryption engine 84. Engines 82 and 84 perform the actual encryption function using, for example, DES-CBC. In addition, software encryption engine 82 may include the keyed MD5 algorithm used for AH.

In one embodiment, crypto engine interface 80 is a utility which provides a consistent interface between the software and hardware encryption engines. As shown in Figure 4, in one such embodiment interface 80 only supports the use of the use of hardware cryptographic engine 84 for IPSEC ESP processing. The significant design issue that interface 80 must deal with is that use of a hardware encryption engine requires that the processing be down in disjoint steps operating in different interrupt contexts as engine 84 completes the various processing steps.

The required information is stored in a request structure that is bound to the IP datagram being processed. The request is of type `crypto_request_t`. This structure is quite large and definitely does not contain a minimum state set.

In addition to the definition of the request data structure, this software implementing interface 80 provides two functions which isolate the decision of which cryptographic engine to use. The `crypt_des_encrypt` function is for use by the IP output processing to encrypt a datagram. The `crypt_des_decrypt` function is for use by the IP input processing to decrypt a datagram. If hardware encryption engine 84 is present and other hardware usage criteria are met the request is enqueued on a hardware processing queue and a return code indicating that the cryptographic processing is in progress is returned. If software engine 82 is used, the return code indicates that the cryptographic processing is complete. In the former case, the continuation of the IP processing is delayed until after hardware encryption is done. Otherwise it is completed as immediately in the same processing stream.

There are two software cryptographic engines 82 provided in the IPSEC software. One provides the MD5 algorithm used by the IPSEC AH processing, and the other provides the DES algorithm used by the IPSEC ESP processing. This software can be obtained from the US Government IPSEC implementation.

In one embodiment hardware cryptographic engine 84 is provided by a Cylink SafeNode processing board. The interface to this hardware card is provided by the Cylink device driver. A significant aspect of the Cylink card that plays a major part in the design of the IPSEC Cylink driver is that the card functions much like a low level subroutine interface and requires software support to initiate each processing step. Thus to encrypt or decrypt an individual datagram there are a minimum of two steps, one to set the DES initialization vector and one to do the encryption. Since the IP processing can not suspend itself and wait while the hardware completes and then be rescheduled by the hardware interrupt handler, in one embodiment a finite state machine is used to tie sequences of hardware processing elements together. In one such embodiment the interrupt handler looks at the current state, executes a defined after state function, transitions to the state and then executes that state's start function.

One function, `cyl_enqueue_request`, is used to initiate either an encrypt or a decrypt action. This function is designed to be called by cryptographic engine interface 80. All of the information required to initiate the processing as well as the function to be performed after the encryption operation is completed is provided in the request structure. This function will enqueue the request on the hardware request queue and start the hardware processing if necessary.

A system 30 which can be used for firewall-to-workstation encryption is shown in Figure 5. In Figure 5, system 30 includes a workstation 12 communicating through a firewall 14 to an unprotected network 16 such as the Internet. System 30 also includes a workstation 32 communicating directly with firewall 14 through unprotected network 16. Firewall 14 is an application level gateway incorporating IPSEC handling as described above. (It should be noted that IPSEC security cannot be used to authenticate the personal identity of the sender for a firewall to firewall transfer. When IPSEC is used, however, on a single user machine such as a portable personal computer, IPSEC usage should be protected with a personal identification number (PIN). In these cases IPSEC can be used to help with user identification to the firewall.)

According to the IPSEC RFC's, you can use either tunnel or transport mode with this embodiment based on your security needs. In certain situations, the communications must be sent in tunnel mode to hide unregistered addresses.

Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiment shown. This application is intended to cover any adaptations or variations of the present invention. Therefore, it is intended that this invention be limited only by the claims and the equivalents thereof.

What is claimed is:

1. A method of regulating the flow of messages through a firewall having a network protocol stack, wherein the network protocol stack includes an Internet Protocol (IP) layer, the method comprising the steps of:

determining, at the IP layer, if a message is encrypted;

if the message is not encrypted, passing the unencrypted message up the network protocol stack to an application level proxy; and

if the message is encrypted, decrypting the message and passing the decrypted message up the network protocol stack to the application level proxy, wherein the step of decrypting the message includes the step of executing a procedure at the IP layer to decrypt the message.

2. A method of authenticating the sender of a message within a computer system having a network protocol stack, wherein the network protocol stack includes an Internet Protocol (IP) layer, the method comprising the steps of:

determining, at the IP layer, if the message is encrypted;

if the message is encrypted, decrypting the message, wherein the step of decrypting the message includes the step of executing a process at the IP layer to decrypt the message;

passing the decrypted message up the network protocol stack to an application level proxy;

determining an authentication protocol appropriate for the message; and

executing the authentication protocol to authenticate the sender of the message.

3. The method according to claim 2 wherein the step of determining an authentication protocol appropriate for the message includes the steps of:

determining a source IP address associated with the message; and

determining the authentication protocol associated with the source IP address.

4. The method according to claim 2 wherein the message includes security parameters index and wherein the step of determining an authentication protocol appropriate for the message includes the steps of:

determining the authentication protocol associated with a dynamic IP address, wherein the step of determining the authentication protocol includes the step of looking up a security association based on the security parameters index;

determining a current address associated with the dynamic source IP address; and

binding the current address to the security parameters index.

5. A firewall, comprising:

a first communications interface;

a second communications interface;

a network protocol stack connected to the first and the second communications interfaces, wherein the network protocol stack includes an Internet Protocol (IP) layer and a transport layer;

a decryption procedure, operating at the IP layer, wherein the decryption procedure decrypts encrypted messages received at one of said first and second communications interfaces and outputs decrypted messages; and

a proxy, connected to the transport layer of said network protocol stack, wherein the proxy receives decrypted messages from the decryption procedure and executes an authentication protocol based on the content of the decrypted message.

6. A firewall, comprising:

a first communications interface;

a second communications interface;

a first network protocol stack connected to the first communications interface, wherein the first network protocol stack includes an Internet Protocol (IP) layer and a transport layer;

a second network protocol stack connected to the second communications interface, wherein the second network protocol stack includes an Internet Protocol (IP) layer and a transport layer;

a decryption procedure, operating at the IP layer of the first network protocol stack, the decryption procedure receiving encrypted messages received by said first communications interface and outputting decrypted messages; and

a proxy, connected to the transport layers of said first and second network protocol stacks, the proxy receiving decrypted messages from the decryption procedure and executing an authentication protocol based on the content of the decrypted message.

7. The firewall according to claim 6 wherein the firewall further includes:

a third communications interface; and

a third network protocol stack connected to the third communications interface and to the proxy, wherein the third network protocol stack includes an Internet Protocol (IP) layer and a transport layer and wherein the second and third network protocol stacks are restricted to first and second burbs, respectively.

8. A method of establishing a virtual private network between a first and a second network, wherein each network includes an application level gateway firewall which uses a proxy operating at the application layer to process traffic through the firewall, wherein each firewall includes a network protocol stack and wherein each network protocol stack includes an Internet Protocol (IP) layer, the method comprising the steps of:

transferring a connection request from the first network to the second network;

determining, at the IP layer of the network protocol stack of the second network's firewall, if the connection request is encrypted;

if the connection request is encrypted, decrypting the request, wherein the step of decrypting the request includes the step of executing a procedure at the IP layer of the second network's firewall to decrypt the message;

passing the connection request up the network protocol stack to an application level proxy;

determining an authentication protocol appropriate for the connection request;

executing the authentication protocol to authenticate the connection request; and

if the connection request is authentic, establishing an active connection between the first and second networks.

9. The method according to claim 8 wherein the step of executing the authentication protocol includes the step of executing program code within the firewall of the second network to mimic a challenge/response protocol executing on a server internal to the second network.

10. The method according to claim 8 wherein the step of executing the authentication protocol includes the step of executing program code to execute the authentication protocol in line to the session.

11. The method according to claim 8 wherein the step of determining an authentication protocol includes the step of determining if the connection request arrived encrypted and selecting the authentication protocol based on whether the connection request was encrypted or not encrypted.

The Patent Office

28

Application No:	GB 9719816.2	**Examiner:**	B.J.SPEAR
Claims searched:	1-11	**Date of search:**	21 January 1998

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.P): H4P (PPEB,PDCSA,PDCSC)

Int Cl (Ed.6): H04L 9/00, 9/32, 29/06, 29/08

Other: Online:WPI, INSPEC

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
XP	WO97/26734A1	(Raptor Systems) Whole document, eg Figs 1,3 and pages 6-12	1,2.5,6,8 at least
XP	WO97/26731A1	(Raptor Systems) Whole document, eg Figs 1,3 and pages 7-12	1,2.5,6,8 at least
XP	WO97/26735A1	(Raptor Systems) Whole document, eg Figs 1,3 and pages 4-10	1,2.5,6,8 at least
XP	WO97/23972A1	(V-ONE Corp) Whole document, eg Figs 1,2 and claim 1.	1,2.5,6,8 at least
XP	WO97/13340A1	(Digital Secured Networks) Whole document, eg pages 7-13	1,2.5,6,8 at least

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ^{6} :
H04Q 11/04, H04L 12/22

A1

(11) International Publication Number: **WO 98/27783**

(43) International Publication Date: 25 June 1998 (25.06.98)

(21) International Application Number: PCT/IB97/01563

(22) International Filing Date: 12 December 1997 (12.12.97)

(30) Priority Data:
08/769,649 19 December 1996 (19.12.96) US

(71) Applicant *(for all designated States except US)*: NORTHERN TELECOM LIMITED [CA/CA]; World Trade Center of Montreal, 8th floor, 380 St. Antoine Street West, Montreal, Quebec H2Y 3Y4 (CA).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: TELLO, Antonio, G. [US/US]; 114 Fountain Hills Drive, Garland, TX 75044 (US). HUI, Margaret [US/US]; 9920 Forest Lane #208, Dallas, TX 75243 (US). HOLMES, Kim [US/US]; 5409 Scenic Drive, Rowlett, TX 75088 (US).

(74) Agents: MCCOMBS, David et al.; Haynes and Boone, L.L.P., Suite 3100, 901 Main Street, Dallas, TX 75202-3789 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: VIRTUAL PRIVATE NETWORK SERVICE PROVIDER FOR ASYNCHRONOUS TRANSFER MODE NETWORK

(57) Abstract

A virtual private network service provider is used to transfer data over a data network to a final destination, with third-party billing. The method comprises the steps of: prompting the user at a data terminal to select a destination, password, and call type; sending a set-up message to the data network; selecting a virtual private network provider through the data network; the virtual private network provider giving an encryption key to the user, and then prompting the user for a password and a user identification; encrypting the password, and sending the user identification and the encrypted password to the virtual private network provider; the virtual private network provider decrypting the encrypted password, and verifying the password; the virtual private network provider providing an authorization code; and the data terminal transferring the data through the data network to the final destination, using the authorization code.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

VIRTUAL PRIVATE NETWORK SERVICE PROVIDER FOR ASYNCHRONOUS TRANSFER MODE NETWORK

Technical Field

The invention relates generally to asynchronous transfer mode ("ATM") networks and virtual private networks ("VPN"), such as those offered by MCI and Sprint, and, more particularly, to a method of using a VPN to transfer data over a data network, with third-party billing.

Background of the Invention

Telephone service providers offer third-party billing. For example, local and long distance telephone companies offer calling cards for third party billing.

VPNs exist to provide the sense of a private network among a company's locations. The lines/trunks of a VPN are actually shared among several companies, to reduce costs, yet to each company the VPN appears to be that company's own private network. However, a user at a remote data terminal, such as a portable computer in a hotel room, can not immediately charge his company for the access time to a data net, such as the Internet. Instead, his access time is charged to his hotel room, and so he must pay the inflated rates that hotels charge for phone service.

What is needed is a VPN service provider that offers remote access for users belonging to a VPN, user authorizations to prevent delinquent access into the VPN, and convenient third-party billing.

Summary of the Invention

The present invention, accordingly, provides a system and method for using a VPN service provider to transfer data over a data network to a final destination, with third-party billing. The method comprises the steps of: prompting the user at a data terminal to select a destination, password, and call type; selecting a VPN through the data network; giving an encryption key to the user, and then prompting the user for a password and a user identification; verifying the password, and providing an authorization code to the user; and allowing the user to transfer the data through the data network to the final destination, using the authorization code.

In another feature of the invention, the method further comprises negotiating for more bandwidth for the user, and including within the authorization code a grant of additional bandwidth.

In another feature of the invention, the method further comprises encrypting the user's password, and sending the user identification and the encrypted password to the VPN service provider.

In another feature of the invention, the method further comprises a step of sending a set-up message to the data network.

In another feature of the invention, the method further comprises a step of the VPN service provider decrypting the encrypted password.

A technical advantage achieved with the invention is that it shifts or defers costs from an end user to a bulk purchaser of data network services. Another technical advantage achieved with the invention is that it permits end users mobility while attaining a virtual appearance on a corporate intranet.

Brief Description of the Drawings

Fig. 1 is a system block diagram of a VPN service provider of the present invention.

Fig. 2 is a flow chart depicting the method of the present invention, as implemented by application software on a user terminal.

Fig. 3 is the initial screen display of the user interface of the application software.

Figs. 4A and 4B are call flow diagrams, illustrating the preferred sequence of steps of the method of the present invention.

Figs. 5A, 5B, 5C, 5D, 5E, and 5F comprise a flow chart depicting the method of the present invention, as implemented by switching control point software.

Description of the Preferred Embodiment

In Fig. 1, the VPN service provider system of the present invention is designated generally by a reference numeral 10. The VPN service provider system 10 includes a VPN 12. The VPN 12 may be a corporate, government, association, or other organization's telephone/data line `network. The VPN service provider system 10 also includes access lines 13 from the VPN 12 to a data network 14, such as the Internet, or an ATM network. The VPN service provider system 10 also includes access lines 16 from the data network 14 to a long distance phone company 18, such as AT&T, MCI, or Sprint. The VPN service provider system 10 also includes access lines 20 from the data network 14 to a called party 22, such as, for example, American Express reservations service. The VPN service provider system 10 also includes access lines 24 from the data network 14 to a remote user terminal 26, such as a portable computer in a hotel room. The user terminal 26 includes user application software 28, which provides the interface for the user to enter the number to be called, the user identification number, and the user's authorization code. The VPN service provider system 10 also includes VPN service provider software 30, located in a switching control point (SCP) device 32, which, in the preferred embodiment may be physically located anywhere. The SCP 32 connects to the data network 14 via access lines 36. One possible physical location for the SCP 32 is on the premises of a local phone company central switch building 34. However, even when located within the building 34, the SCP 32 connects to the local phone company switches via the data network 14. The local phone company switches connect to the data network 14 via access lines 38.

In an alternate embodiment, the VPN service provider software 30 and the SCP device 32 may be located on the premises of an independent provider of local phone service, or on the premises of an independent VPN service provider.

Referring now to Fig. 2, the application software 28 begins the data transfer process in step 50. In step 52, the user is presented with a screen display.

Referring now to Fig. 3, a screen display 100 displays the following information requests: whether the call is a direct call 102 or a VPN call 104, the number the user desires to call 106, the VPN user ID 108, and the user password 110. The user is also presented with the option to make the call 112, or to quit 114.

Referring back to Fig. 2, in step 54 the user terminal sends to the SCP 32 the information captured through the graphical user interface ("GUI") in step 52 within a user network interface ("UNI") setup message. In step 56 the user terminal 26 waits for a connect message from the SCP 32. In step 58 the user terminal 26 determines if a connection was made. If no connection was made, then in step 60 the user application software 28 displays an error message to the user, and returns to step 50 to begin again the data transfer process.

If a connection was made, then in step 62 the user terminal 26 sends the VPN user ID to the SCP 32. In step 64 the user terminal 26 waits for an encryption key from the SCP 32. In step 66, having received the encryption key from the SCP 32, the user application software 28 encrypts the user's password, and sends it to the SCP 32. In step 68 the user terminal 26 waits for authentication of the user. In step 70 the user application software 28 determines if the SCP 32 authorizes the user to make the call.

If the user is not authorized, then in step 72 the user terminal 26 displays an error message, terminates the connection, blanks the screen display 100, and returns to step 50 to begin again the data transfer process. If the user is authorized, then in step 74 the VPN service provider software 30 sets up the billing, and authorizes it. In step 76 the user terminal 26 sends a "release", meaning to terminate or disconnect the connection, to the SCP 32. In step 78 the user terminal 26 sends a setup message to the number listed by the user as the "number to call", that is, to the final destination. In step 80 the user terminal 26 waits for a connection. In step 82 the user terminal 26 determines if a connection was made.

If a connection to the final destination was not made, then the user application software 28 returns to step 72, in which step the user terminal 26 displays an error message, terminates the connection, blanks the screen display 100, and returns to step 50 to begin again the data transfer process. If a connection to the final destination was made, then in step 84 the user terminal 26 exchanges user data, services, and/or value added or user specific applications with the computer at the address, that is, the telephone number, of the final destination. In step 86 the user selects the option presented to him to release, or terminate, the call. In step 88 the user terminal 26 sends a release message to the final destination. In step 90 the data network 14 sends billing information to the SCP 32. In step 92 the application software 28 ends the data transfer process.

Fig. 4A and Fig. 4B are call flow diagrams, showing the sequence of messages in the method of the preferred embodiment. These diagrams present the same method as the flow chart of Fig. 2. The horizontal arrows represent the messages sent and received. The vertical lines represent the various devices involved in sending and receiving the messages. For example, the top left arrow in Fig. 4A represents a message sent from the user terminal 26, labeled "Macintosh" in Fig.4A, to an interface with a public network. The user terminal 26 can be any brand of a work station computer, a desktop computer, a laptop computer, or even a notebook computer. The interface could be any interface, but in the example of Fig. 4A and Fig. 4B, the interface is imagined to be at a hotel, where a business traveler is using the method of the present invention. Thus, the interface is labeled "Hotel ATM Interface", which is not shown in Fig. 1. The vertical line labeled "Public ATM Network" is the same as the data network 14 in Fig. 1. The vertical line labeled "Moe's VPN Service" represents the VPN service provider software 30 within the SCP 32. The vertical line labeled "Travel ATM Interface" is not shown in Fig. 1, but is located between the called party 22 and the data network 14. The vertical line labeled "Travel Service" is one example of the called party 22 shown in Fig. 1. In the example of Fig. 4A and Fig. 4B, the business traveler is imagined to be using the method of the present invention to contact a travel service to make reservations for his next airline flight. In Figs. 4A and 4B the designation "Ack" represents "acknowledge", and the designation "Cmp" represents "complete".

Referring now to Fig. 5, the VPN service provider software 30 begins the data transfer process in step 300 by waiting for an event. The event it waits for is a setup message on a signaling port of the SCP 32, to be received from the user terminal 26. In step 302, having monitored the signaling ports, and the SCP 32 having received a setup message, the VPN service provider software 30 assigns a call condense block ("CCB") to the setup message, based on a call reference number. The CCB is a software data structure for tracking resources associated with the call. The call reference number is a number, internal to the SCP, for tracking calls. In step 304 the VPN service provider software 30 compiles the connect message. In step 306 the VPN service provider software 30 sends a connect message to the calling address, that is, the hotel room from which the user is calling. In step 308 the VPN service provider software 30 condenses, that is, it remains in a wait state for that call.

Referring now to Fig. 5B, in step 310 the VPN service provider software 30 waits for an event by monitoring the signaling ports of the SCP 32. After the SCP 32 receives a connect acknowledge message from the user terminal 26, then in step 312 the VPN service provider software 30 accesses the CCB, based on the call reference number. In step 314 the VPN service provider software 30 condenses.

Referring now to Fig. 5C, in step 316 the VPN service provider software 30 waits for dialog on a data port of the SCP 32. After the SCP 32 receives a

VPN ID on a data port, the VPN service provider software 30 verifies the VPN ID in step 318. In step 320 the VPN service provider software 30 determines if the VPN ID is valid. If the VPN ID is not valid, then in step 322 the SCP 32 sends a reject message over an assigned switch virtual circuit ("SVC"). The SVC is a channel over the data network 14. In step 324 the VPN service provider software 30 waits for dialog. In step 326, because the VPN ID is valid, the VPN service provider software 30 assigns an encryption key to the user terminal 26, in step 328 sends the encryption key over the assigned SVC to the user terminal 26, and in step 330 waits for dialog.

Referring now to Fig. 5D, in step 332 the VPN service provider software 30 waits for dialog. When the SCP 32 receives the encrypted password from the user terminal 26 at a data port, then in step 334 the VPN service provider software 30 verifies the password, and determines in step 336 if the password is valid. If the password is not valid, then in step 338 the SCP 32 sends a reject message over the assigned SVC to the user terminal, and in step 340 waits for dialog. If the password is valid, then in step 342 the VPN service provider software 30 assigns an authorization token to the user terminal 26, in step 344 sends the token over an assigned SVC to the user terminal 26, and in step 346 waits for dialog.

Referring now to Fig. 5E, in step 348 the VPN service provider software 30 waits for an event. When the VPN service provider software 30 senses that the SCP 32 has received on a signaling port a release message from the user terminal 26, then in step 350 the VPN service provider software 30 accesses the CCB, based on the call reference number of the user terminal 26, in step 352 compiles a release complete message, in step 354 sends a release complete message to the user terminal 26, and in step 356 condenses.

Referring now to Fig. 5F, in step 358 the VPN service provider software 30 waits for an event. When the VPN service provider software 30 senses that the SCP 32 has received on a signaling port a third-party billing setup message from the user terminal 26, then in step 360 the VPN service provider software 30 verifies the token just received from the user terminal 26, to determine, in step 362, if it is the same token that the VPN service provider software 30 sent to the user terminal 26 in step 344. If the token is not valid, then in step 364 the SCP 32 sends a release message to the terminal 26, and in step 366 condenses. If the token is valid, then in step 368 the SCP 32 sends a modified third-party billing setup message to the data network 14, and in step 370 condenses.

Although an illustrative embodiment of the invention has been shown and described, other modifications, changes, and substitutions are intended in the foregoing disclosure. Accordingly, it is appropriate that the appended claims be construed broadly and in a manner consistent with the scope of the invention.

WHAT IS CLAIMED IS:

1. A computerized method of a virtual private network service provider with third party billing, using a virtual private network to transfer data over a data network to a final destination, the method comprising the steps of:

   a. prompting the user at a data terminal to select a destination, password, and call type;

   b. selecting a virtual private network through the data network;

   c. giving an encryption key to the user, and then prompting the user for a password and a user identification;

   d. verifying the password, and providing an authorization code to the user; and

   e. allowing the user to transfer the data through the data network to the final destination, using the authorization code.

2. The method of claim 1, wherein step (d) further comprises negotiating for more bandwidth for the user, and including within the authorization code a grant of additional bandwidth.

3. The method of claim 2, wherein step (c) further comprises encrypting the user's password, and sending the user identification and the encrypted password to the virtual private network service provider.

4. The method of claim 3, further comprising, after step (a), the step of sending a set-up message to the data network.

5. The method of claim 4, further comprising, after step (c), the step of the virtual private network service provider decrypting the encrypted password.

6. An apparatus for providing a datalink connection from a user terminal to a data network and to a virtual private network, with third party billing, comprising:

   a. an interface between the user terminal and the data network;

b. a switching control point device connected to the data network, the switching control point device connected to a computer; and

c. a computer-readable medium encoded with a method of using the virtual private network and the data network, with third party billing, the computer-readable medium accessible by the computer.

7. The apparatus of claim 6, wherein the method comprises negotiating for more bandwidth for the user, and including within an authorization code a grant of additional bandwidth.

8. The apparatus of claim 7, wherein the method further comprises encrypting a user's password, and temporarily storing the user identification and the encrypted password.

9. The apparatus of claim 8, wherein the method further comprises sending a set-up message to the data network.

10. The apparatus of claim 9, wherein the method further comprises decrypting the encrypted password.

11. A computer-readable medium encoded with a method of using a virtual private network, with third party billing, the method comprising the steps of:

a. prompting the user at a data terminal to select a destination, password, and call type;

b. selecting a virtual private network through the data network;

c. giving an encryption key to the user, and then prompting the user for a password and a user identification;

d. verifying the password, and providing an authorization code to the user; and

e. allowing the user to transfer the data through the data network to the final destination, using the authorization code.

12. The computer-readable medium of claim 11 wherein step (d) further comprises negotiating for more bandwidth for the user, and including within the authorization code a grant of additional bandwidth.

13. The computer-readable medium of claim 12 wherein step (c) further comprises encrypting the user's password, and sending the user identification and the encrypted password to the virtual private network service provider.

14. The computer-readable medium of claim 13 further comprising, after step (a), the step of sending a set-up message to the data network.

15. The computer-readable medium of claim 14 further comprising, after step (c), the step of the virtual private network service provider decrypting the encrypted password.

16. An apparatus for providing a datalink connection from a user terminal to a data network and to a virtual private network, with third party billing, comprising:

a. means for prompting a user at the data terminal to select a destination, password, and call type;

b. means for selecting the virtual private network through the data network;

c. means for giving an encryption key to the user, and then prompting the user for a password and a user identification;

d. means for verifying the password, and providing an authorization code to the user; and

e. means for allowing the user to transfer data through the data network to a final destination, using the authorization code.

17. The apparatus of claim 16, further comprising means for negotiating for more bandwidth for the user, and including within the authorization code a grant of additional bandwidth.

18. The apparatus of claim 17, further comprising means for encrypting the user's password, and sending the user identification and the encrypted password to the virtual private network service provider.

19. The apparatus of claim 18, further comprising means for sending a set-up message to the data network.

20. The apparatus of claim 19, further comprising means for decrypting the encrypted password.

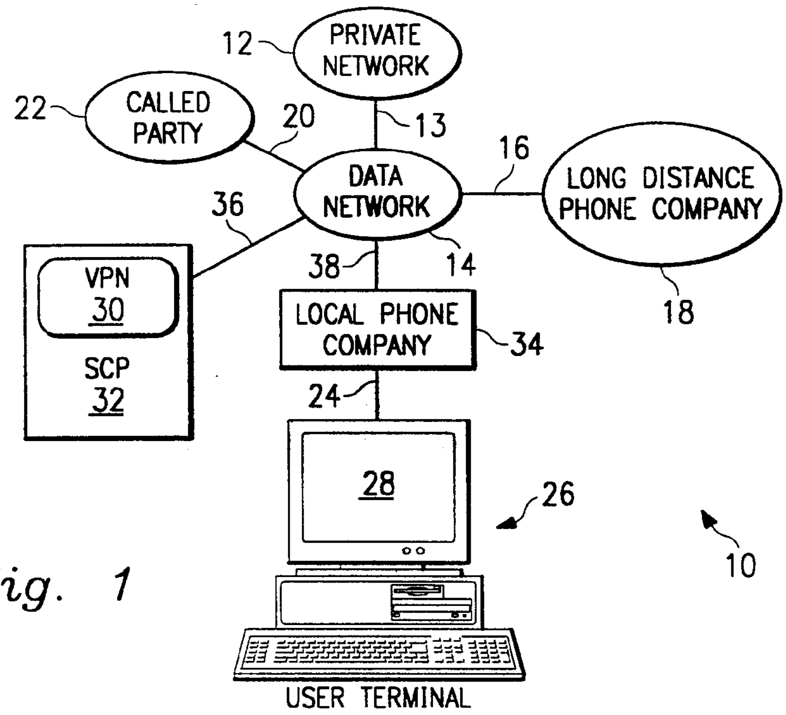

PRIVATE NETWORK
12
CALLED PARTY
22
20
13
16
DATA NETWORK
LONG DISTANCE PHONE COMPANY
36
38
14
VPN
30
SCP
32
LOCAL PHONE COMPANY
34
18
24
28
26
10
USER TERMINAL

Fig. 1

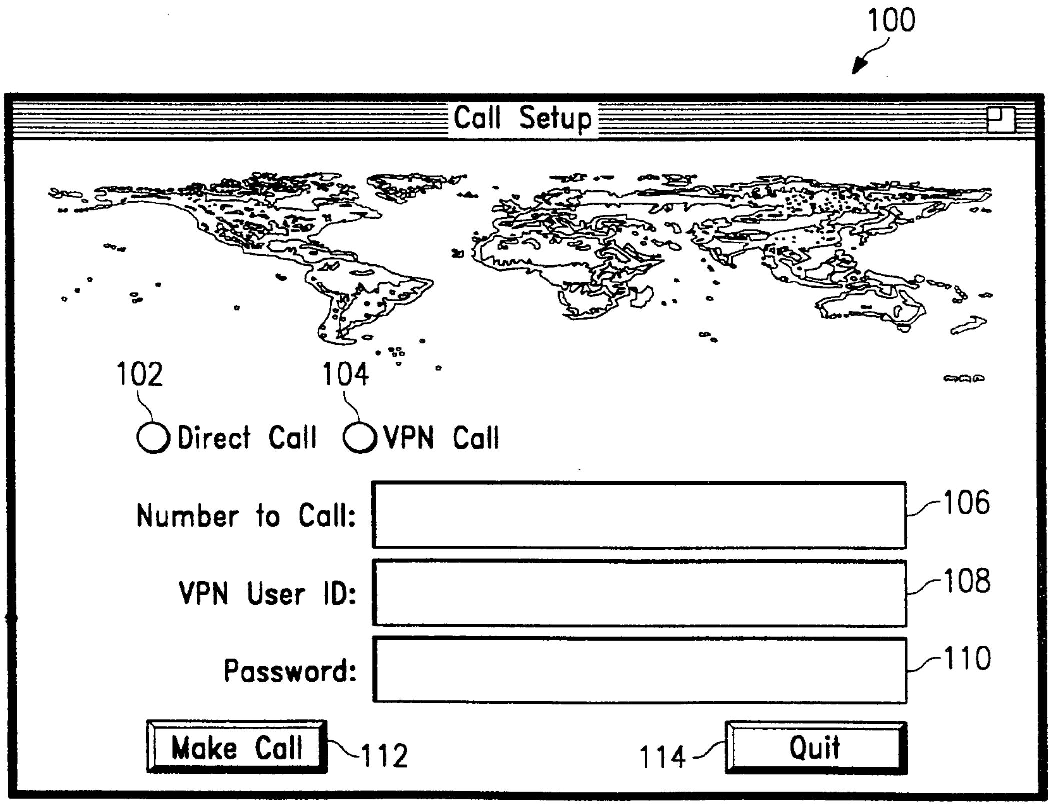

100
Call Setup
102
104
○ Direct Call
○ VPN Call
Number to Call:
106
VPN User ID:
108
Password:
110
Make Call
112
114
Quit

Fig. 3

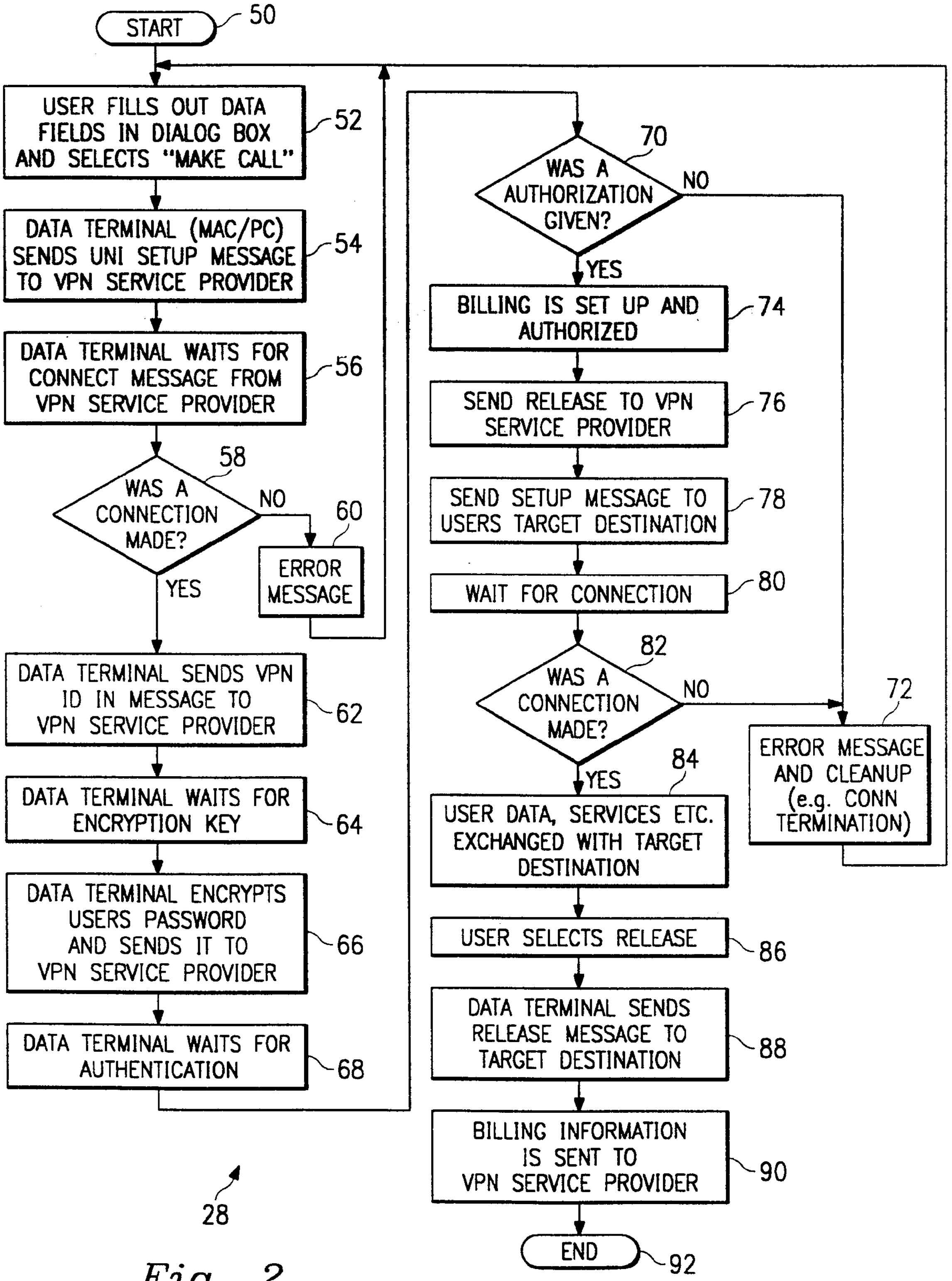

START
50
USER FILLS OUT DATA FIELDS IN DIALOG BOX AND SELECTS "MAKE CALL"
52
DATA TERMINAL (MAC/PC) SENDS UNI SETUP MESSAGE TO VPN SERVICE PROVIDER
54
DATA TERMINAL WAITS FOR CONNECT MESSAGE FROM VPN SERVICE PROVIDER
56
WAS A CONNECTION MADE?
58
NO
60
ERROR MESSAGE
YES
DATA TERMINAL SENDS VPN ID IN MESSAGE TO VPN SERVICE PROVIDER
62
DATA TERMINAL WAITS FOR ENCRYPTION KEY
64
DATA TERMINAL ENCRYPTS USERS PASSWORD AND SENDS IT TO VPN SERVICE PROVIDER
66
DATA TERMINAL WAITS FOR AUTHENTICATION
68
WAS A AUTHORIZATION GIVEN?
70
NO
YES
BILLING IS SET UP AND AUTHORIZED
74
SEND RELEASE TO VPN SERVICE PROVIDER
76
SEND SETUP MESSAGE TO USERS TARGET DESTINATION
78
WAIT FOR CONNECTION
80
WAS A CONNECTION MADE?
82
NO
72
ERROR MESSAGE AND CLEANUP (e.g. CONN TERMINATION)
YES
84
USER DATA, SERVICES ETC. EXCHANGED WITH TARGET DESTINATION
USER SELECTS RELEASE
86
DATA TERMINAL SENDS RELEASE MESSAGE TO TARGET DESTINATION
88
BILLING INFORMATION IS SENT TO VPN SERVICE PROVIDER
90
END
92
28

Fig. 2

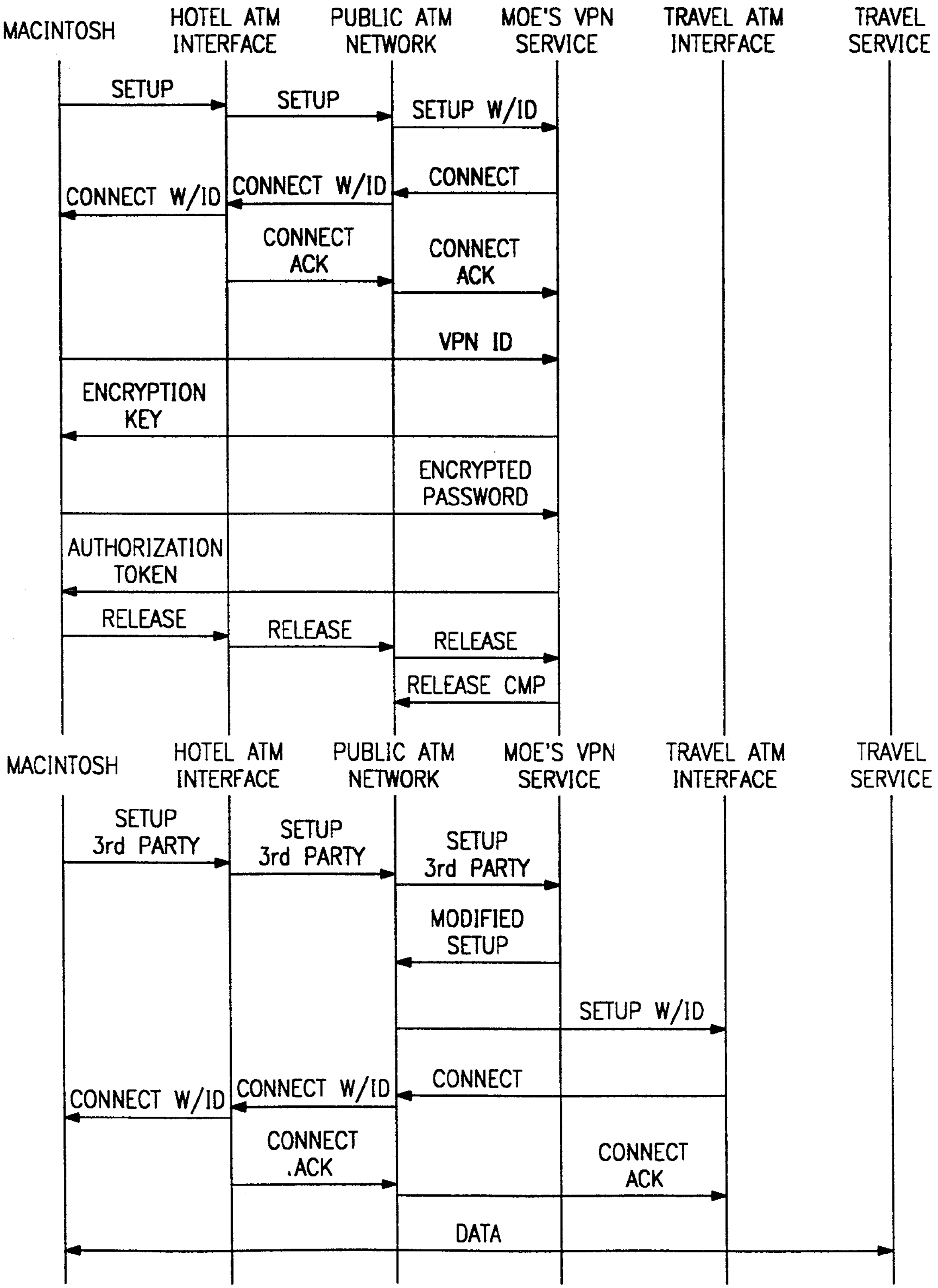

MACINTOSH
HOTEL ATM INTERFACE
PUBLIC ATM NETWORK
MOE'S VPN SERVICE
TRAVEL ATM INTERFACE
TRAVEL SERVICE
SETUP
SETUP
SETUP W/ID
CONNECT W/ID
CONNECT W/ID
CONNECT
CONNECT ACK
CONNECT ACK
VPN ID
ENCRYPTION KEY
ENCRYPTED PASSWORD
AUTHORIZATION TOKEN
RELEASE
RELEASE
RELEASE
RELEASE CMP
MACINTOSH
HOTEL ATM INTERFACE
PUBLIC ATM NETWORK
MOE'S VPN SERVICE
TRAVEL ATM INTERFACE
TRAVEL SERVICE
SETUP 3rd PARTY
SETUP 3rd PARTY
SETUP 3rd PARTY
MODIFIED SETUP
SETUP W/ID
CONNECT W/ID
CONNECT W/ID
CONNECT
CONNECT .ACK
CONNECT ACK
DATA

Fig. 4A

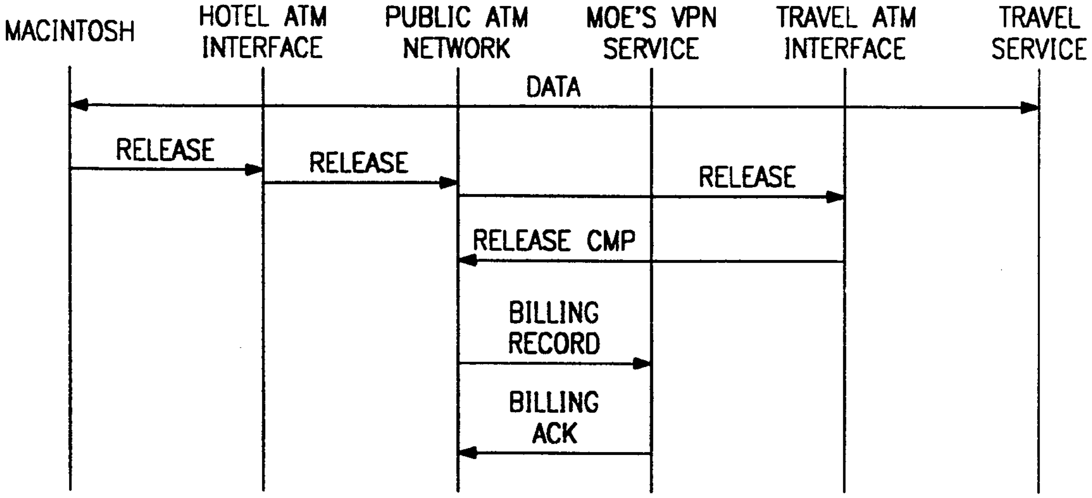


Fig. 4B

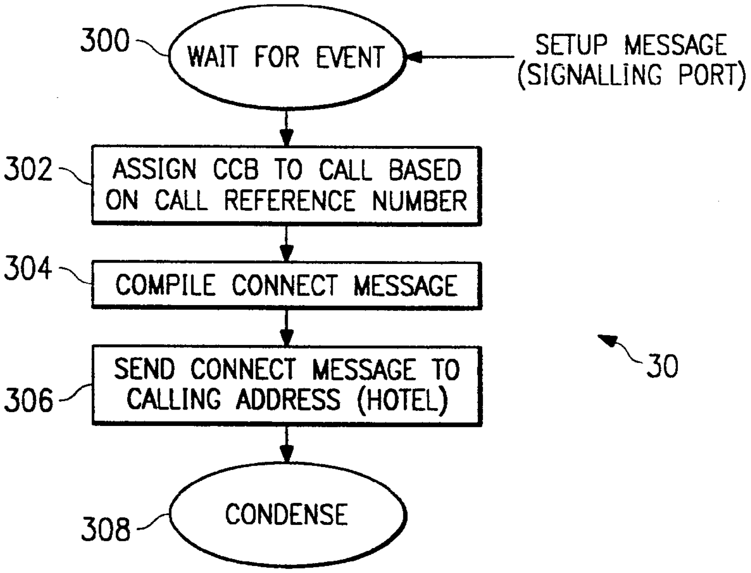

300
WAIT FOR EVENT
SETUP MESSAGE (SIGNALLING PORT)
302
ASSIGN CCB TO CALL BASED ON CALL REFERENCE NUMBER
304
COMPILE CONNECT MESSAGE
306
SEND CONNECT MESSAGE TO CALLING ADDRESS (HOTEL)
308
CONDENSE
30

Fig. 5A

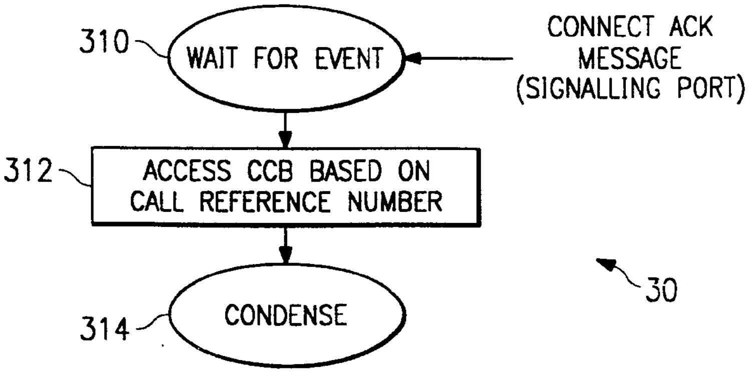

310
WAIT FOR EVENT
CONNECT ACK MESSAGE (SIGNALLING PORT)
312
ACCESS CCB BASED ON CALL REFERENCE NUMBER
314
CONDENSE
30

Fig. 5B

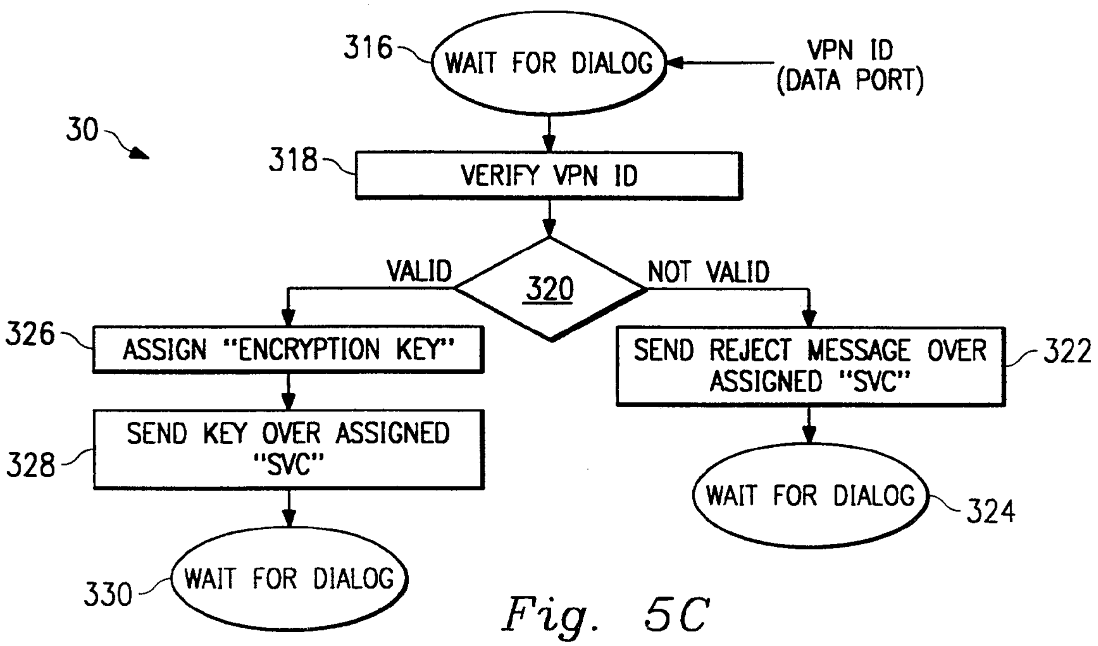

30
316
WAIT FOR DIALOG
VPN ID
(DATA PORT)
318
VERIFY VPN ID
VALID
320
NOT VALID
326
ASSIGN "ENCRYPTION KEY"
328
SEND KEY OVER ASSIGNED "SVC"
330
WAIT FOR DIALOG
SEND REJECT MESSAGE OVER ASSIGNED "SVC"
322
WAIT FOR DIALOG
324

Fig. 5C

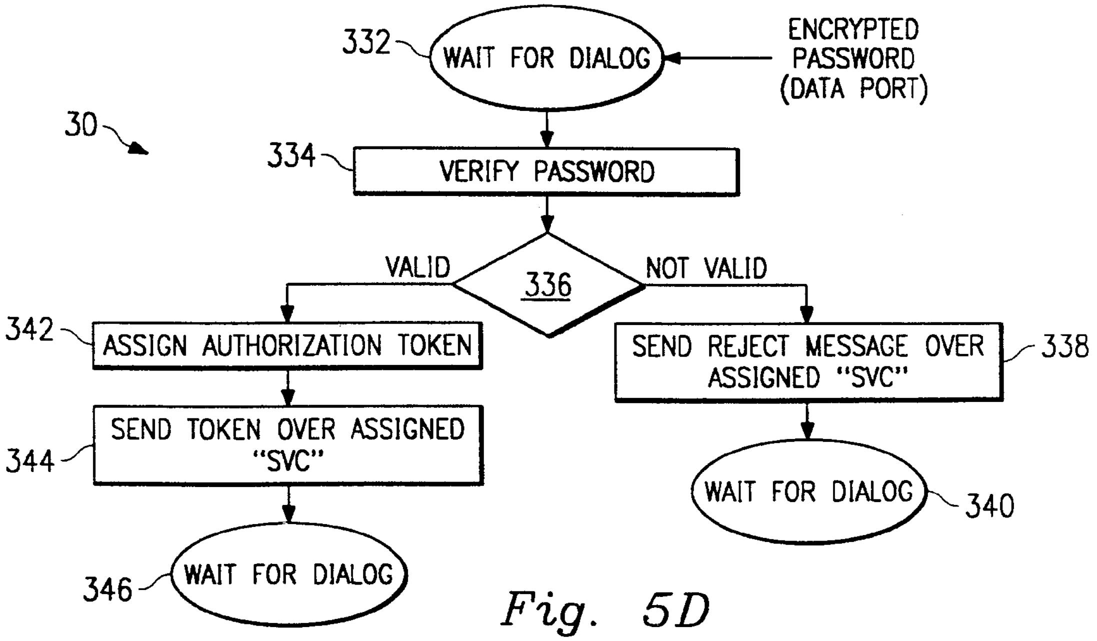

30
332
WAIT FOR DIALOG
ENCRYPTED
PASSWORD
(DATA PORT)
334
VERIFY PASSWORD
VALID
336
NOT VALID
342
ASSIGN AUTHORIZATION TOKEN
344
SEND TOKEN OVER ASSIGNED "SVC"
346
WAIT FOR DIALOG
SEND REJECT MESSAGE OVER ASSIGNED "SVC"
338
WAIT FOR DIALOG
340

Fig. 5D

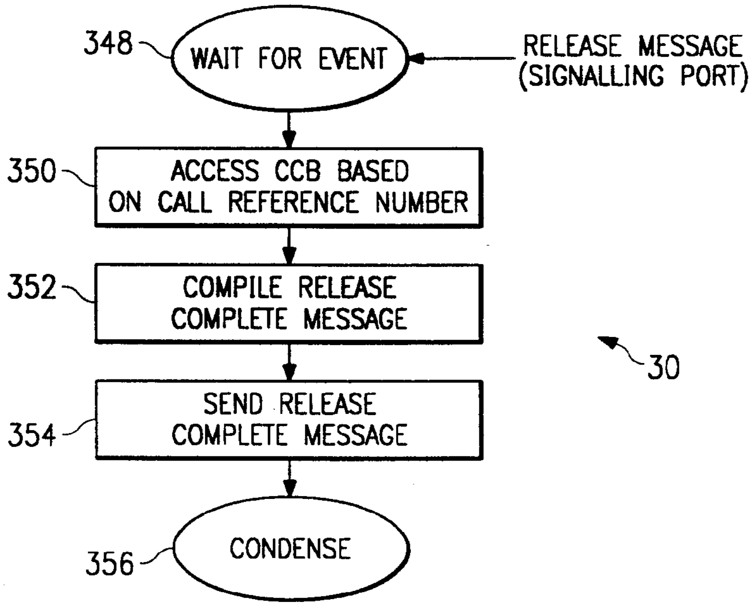

348
WAIT FOR EVENT
RELEASE MESSAGE (SIGNALLING PORT)
350
ACCESS CCB BASED ON CALL REFERENCE NUMBER
352
COMPILE RELEASE COMPLETE MESSAGE
354
SEND RELEASE COMPLETE MESSAGE
356
CONDENSE
30

Fig. 5E

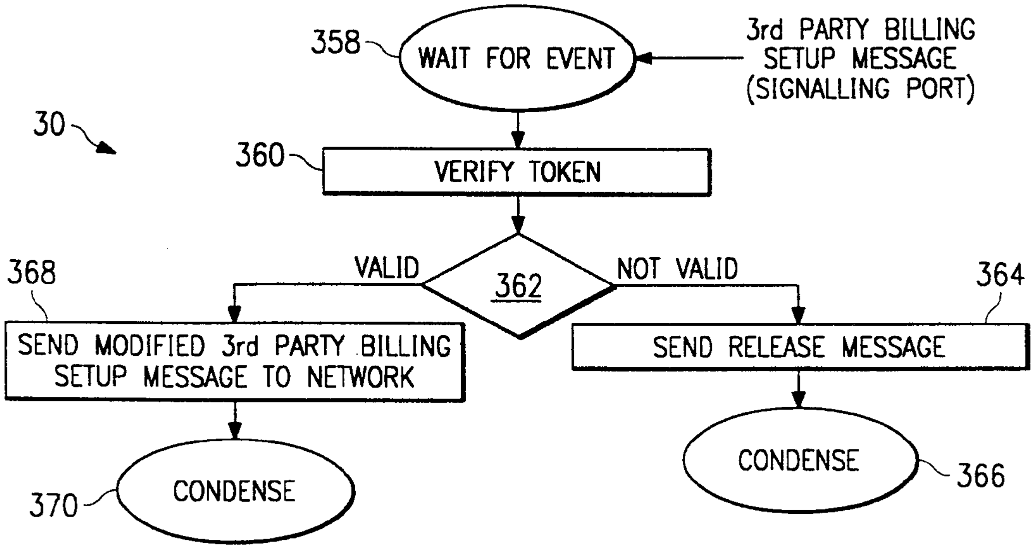

358
WAIT FOR EVENT
3rd PARTY BILLING SETUP MESSAGE (SIGNALLING PORT)
30
360
VERIFY TOKEN
362
VALID
NOT VALID
368
SEND MODIFIED 3rd PARTY BILLING SETUP MESSAGE TO NETWORK
364
SEND RELEASE MESSAGE
370
CONDENSE
CONDENSE
366

Fig. 5F

INTERNATIONAL SEARCH REPORT

Inte ɔnal Application No
PCT/IB 97/01563

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04Q11/04 H04L12/22

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04Q H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	MUN CHOON CHAN ET AL: "AN ARCHITECTURE FOR BROADBAND VIRTUAL NETWORKS UNDER CUSTOMER CONTROL" NOMS '96 IEEE NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM , vol. 1, 15 April 1996, KYOTO, JP, pages 135-144, XP000641086 see abstract ---	1-20
A	BIC V: "VOICE PERIPHERALS IN THE INTELLIGENT NETWORK" TELECOMMUNICATIONS, vol. 28, no. 6, June 1994, page 29/30, 32, 34 XP000600293 see the whole document --- -/--	1-20

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

1

Date of the actual completion of the international search 19 March 1998	Date of mailing of the international search report 02/04/1998
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Staessen, B

INTERNATIONAL SEARCH REPORT

International Application No
PCT/IB 97/01563

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0 729 256 A (NEDERLAND PTT) 28 August 1996 see abstract figures of pages 136 and 140 ---	1-20
A	CROCETTI P ET AL: "ATM VIRTUAL PRIVATE NETWORKS: ALTERNATIVES AND PERFORMANCES COMPARISONS" SUPERCOMM/ICC '94, 1 May 1994, NEW ORLEANS, US, pages 608-612, XP000438985 see abstract -----	1-20

1

New Bay Capital, LLC
Ex. 1007

INTERNATIONAL SEARCH REPORT

ınformation on patent family members

Inte ɔnal Application No
PCT/IB 97/01563

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0729256 A	28-08-96	NL 9500339 A	01-10-96

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
H04L 9/00

A1

(11) International Publication Number: **WO 99/11019**

(43) International Publication Date: 4 March 1999 (04.03.99)

(21) International Application Number: PCT/US98/17198

(22) International Filing Date: 24 August 1998 (24.08.98)

(30) Priority Data:
08/917,341 26 August 1997 (26.08.97) US

(71) Applicant: V-ONE CORPORATION [US/US]; Suite 300, 20250 Century Boulevard, Germantown, MD 20874 (US).

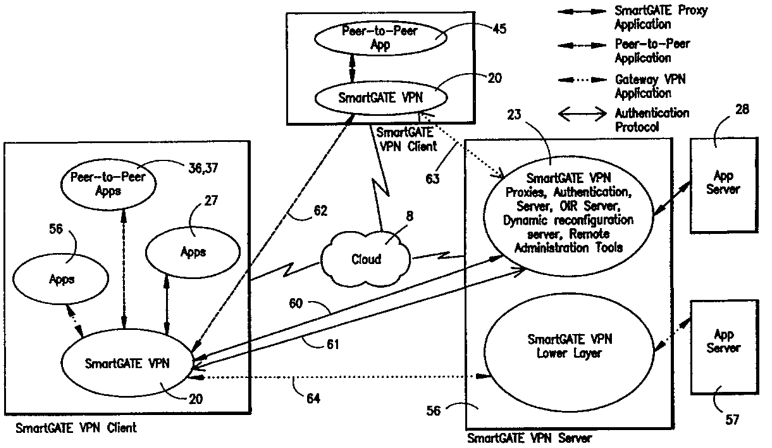
(72) Inventors: CHEN, James, F.; 12648 Tavilah Road, Potomac, MD 20854 (US). WANG, Jieh-Shan; 10903 Silent Wood Place, N. Potomac, MD 20878 (US). BROOK, Christopher, T.; 7308 Pomander Lane, Chevy Chase, MD 20815 (US). GARVEY, Francis; 2908 S. Buchanan Street, Arlington, VA 22206 (US).

(74) Agents: URCIA, Benjamin, E. et al.; Bacon & Thomas, PLLC, 4th floor, 625 Slaters Lane, Alexandria, VA 22314 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: MULTI-ACCESS VIRTUAL PRIVATE NETWORK



(57) Abstract

A virtual private network for communicating between a server and clients over an open network uses an applications level encryption and mutual authentication program (20) and at least one shim (50, 53) positioned above either the layers of a client computer to intercept function calls, communicate with the server and authenticate the parties to a communication and enable the parties to the communication to establish a common session key. Where the parties to the communication are peer-to-peer applications (36, 37, 45), the intercepted function calls, request for service, or data packets include the destination address of the peer application, which is supplied to the server so that the server can authenticate the peer and enable the peer to decrypt further direct peer-to-peer communications (62).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

MULTI-ACCESS VIRTUAL PRIVATE NETWORK

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates a system and method for allowing private communications over an open network, and in particular to a virtual private network which provides data encryption and mutual authentication services for both client/server and peer-to-peer applications at the applications, transport driver, and network driver levels.

2. Discussion of Related Art

A virtual private network (VPN) is a system for securing communications between computers over an open network such as the Internet. By securing communications between the computers, the computers are linked together as if they were on a private local area network (LAN), effectively extending the reach of the network to remote sites without the infrastructure costs of constructing a private network. As a result, physically separate LANs can work together as if they were a single LAN, remote computers can be temporarily connected to the LAN for communications with mobile workers or telecommuting, and electronic commerce can be carried out without the risks inherent in using an open network.

In general, there are two approaches to virtual private networking, illustrated in Figs. 1A and 1B. The first is to use a dedicated server 1, which may also function as a gateway to a secured network 2, to provide encryption and authentication services for establishment of secured links 3 between the server 1 and multiple clients 4-6 over the open network 7, represented in Fig. 1A as a cloud, while the second is to permit private communications links 8 to be established between any two computers or computer systems 9-12 on network 7, as illustrated in Fig. 1B.

The advantages of a client/server arrangement such as the one shown in Fig. 1A are that the server can handle functions requiring the majority of the computing resources, increasing the number of potential clients, and that management of the network, including key management is centralized. The disadvantage of a client/server network of this type is that peer-to-peer communications links between applications on the client computers cannot utilize the security and management functions provided by the server, leaving such communications unprotected. On the other hand, the advantage of the direct peer-to-peer approach illustrated in Fig. 1B is that it permits secured links to be established between any computers capable of carrying out the required security functions, with the disadvantages being the cost of configuring each computer to carry-out encryption, authentication, and key management functions, and the lack of central control.

In both the client/server and peer-to-peer approaches, a virtual private network can in theory be based either on applications level technology or can operate at a lower level. Generally, however, peer-to-peer "tunneling" arrangements require modification of the lower layers of a computer's communications architecture, while client/server arrangements can use the applications level approach because less modification of the clients is required, and thus the two approaches are in practice mutually exclusive. The present invention, on the other hand, seeks to provide a virtual private network which utilizes a client/server approach, including centralized control of encryption, authentication, and key management functions, while at the same time enabling secured peer-to-peer communications between applications, by utilizing the server to provide authentication and session key generation functions for both client to server communications and peer-to-peer communications, providing a virtual private network capable of serving both as an extended intranet or wide area network (WAN), and as a commercial mass marketing network, with high level mutual authentication and encryption provided for all communications.

In order to completely integrate the two approaches and maximize the advantage of each approach, the invention maintains the applications level infrastructure of prior client server private networking arrangements, while adding shims to lower levels in order to accommodate a variety of peer-to-peer communications applications while utilizing the applications level infrastructure for authentication and session key generation purposes. This results in the synergistic effect that not only are existing peer-to-peer tunneling schemes and applications level client server security arrangements combined, but they are combined in a way which greatly reduces implementation costs

In order to understand the present invention, it is necessary to understand a few basic concepts about computer to computer communications, including the concepts of "layers" and communications protocols, and of mutual authentication and file encryption. Further information about layers and protocols can be found in numerous sources available on the Internet, a few of which are listed at the end of this section, while a detailed description of a mutual authentication and encryption system and method suitable for use in connection with the present invention can be found in U.S. Patent No. 5,602,918, which is incorporated herein by reference. In general, the basic communications protocols and architecture used by the present invention, as well as authentication, encryption, and key management schemes, are already well-known, and can be implemented as a matter of routine programming once the basic nature of the invention is understood. The changes made by the present invention to the conventional client server virtual private network may be thought of as, essentially, the addition of means, most conveniently implemented as shims, which add a secured mutual authentication and session key generation channel between the server and all parties to a communication, at all levels at which a communication can be carried out.

Having explained the key differences between the present invention and existing systems, the basic concepts of layers and so forth will now be briefly explained by way of background. First, the concept of "layers," "tiers," and "levels," which essential to an understanding of the invention, simply refers to libraries or sets of software routines for carrying out a group of related functions, and which can conveniently be shared or called on by different programs at a higher level to facilitate programming, avoiding duplication and maximizing computer resources. For example, the Windows NT device driver architecture is made up of three basic layers, the first of which is the Network Driver Interface Specification (NDIS 3.0) layer, the second of which is called the Transport Driver Interface (TDI) layer, and the third being the file systems. These layers are generically referred to as the network driver layer, the transport or transport driver layer, and the applications layer.

In the Windows NT architecture, the TDI layer formats data received from the various file systems or applications into packets or datagrams for transmission to a selected destination over the open network, while the NDIS layer controls the device drivers that send the data, packets, or IP datagrams, for example by converting the stream of data into a waveform suitable for transmission over a telephone line or a twisted pair cable of the type known as an Ethernet.

By providing layers in this manner, an applications software programmer can design an application program to supply data to the TDI layer without having to re-program any of the specific functions carried out by that layer, and all of the transmission, verification, and other functions required to send a message will be taken care of the TDI layer without further involvement by the applications software. In a sense, each "layer" simply accepts data from the higher layer and formats it by adding a header or converting the data in a manner which is content independent, with retrieval of the data simply involving reverse conversion or stripping of the headers, the receiving software receiving the data as if the intervening layers did not exist.

In the case of Internet communications, the most commonly used set of software routines for the transport or TDI layer, which takes care of the data formatting and addressing, is the TCP/IP protocol, in which the transport control protocol (TCP) packages the data into datagrams and provides addressing, acknowledgements, and checksum functions, and the internet protocol (IP) further packages the TCP datagrams into packets by adding additional headers used in routing the packets to a destination address. Other transport protocols which can be included in the TDI layer include the user diagram protocol (UDP), the internet control message protocol (ICMP), and non-IP based protocols such as Netbeui or IPX.

Additional "protocols" are may be used at the applications level, although these protocols have nothing to do with the present invention except that they may be included in the applications programs served by the network. Common applications level protocols which utilize the TCP/IP protocol include hypertext transfer protocol (HTTP), simple mail transfer protocol (SMTP), and file transfer protocol (FTP), all of which operate at the layer above the transport layer.

Some applications are written to directly call upon the TCP functions. However, for most applications utilizing a graphical user interface conveniently rely on a set of software routines which are considered to operate above the TDI layer, and are known as sockets. Sockets serve as an interface between the TCP set of functions, or stack, and various applications, by providing libraries of routines which facilitate TCP function calls, so that the application simply has to refer to the socket library in order to carry out the appropriate function calls. For Windows applications, a commonly used non-proprietary socket is the Windows socket, known as Winsock, although sockets exist for other operating systems or platforms, and alternative sockets are also available for Windows, including the Winsock 2 socket currently under development.

In order to implement a virtual private network, the encryption and authentication functions must be carried out at one of the above "levels," for example by modifying the network drivers to encrypt the IP datagrams, by inserting authentication headers into the TCP/IP stacks, or by writing applications to perform these functions using the existing drivers. If possible, it is generally desirable to minimize modification of the existing levels by adding a layer to perform the desired functions, calling upon the services of the layer below, while utilizing the same function calls so that the higher layer also does not need to be modified. Such a layer is commonly referred to as a "shim."

As indicated above, the preferred approach to implementing client/server virtual private networks is to use an applications level security system to encrypt files to be transmitted, and to then utilize existing communications layers such as Winsock, or TCP/IP directly. This is the approach taken by the commercially available access control system known as SmartGATE™, developed by V-One Corp. of Germantown, Md., which provides both encryption and mutual authentication at the applications level utilizing a dedicated server known as an authentication server and authentication client software installed at the applications level on the client computers. A description of the manner in which encryption and mutual authentication is carried out may be found in the above-cited U.S. Patent No. 5,602,918. While the principles of the invention are applicable to other client/server based virtual private networks, SmartGATE™ is used as an example because it provides the most complete range of mutual authentication and encryption services currently available.

The present invention can be implemented using the existing SmartGATE™ system, but adds mutual authentication and encryption services to lower layers by intercepting function calls or data packets and, during initialization of a communications link, establishing separate channels between the party initiating the communication and the authentication server, and between the authentication server and the party which is to share in the communication, so as to mutually authenticate the parties with respect to the server, and so as to establish a session key which can be used for further direct communications between the parties.

A number of protocols exist which can be used, in total or in part, to implement the mutual authentication and encryption services at the lower layers, using the same basic authentication and encryption scheme currently implemented by SmartGATE™ at the applications level. These include, by way of example, the SOCKS protocol, which places a shim between the TDI or transport layer and the applications, and the commercially available program, known as SnareNet, which operates at the network driver level and can be directly utilized in connection with the present invention.

On the other hand, a network level implementation such as the SKIP protocol, which operates below the TDI layer to encrypt the datagrams, and which in its description explicitly precludes the generation of session keys (see the above cited U.S. Patent No. 5,602,918), is fundamentally different in concept than the present invention. Similarly, alternative implementations such as Point-to-Point Tunneling Protocol (PPTP) which involve modifying the TCP/IP stack and/or hardware to provide encryption, as opposed to inserting shims, are not utilized by the preferred embodiment of the present invention, although individual aspects of the protocol could perhaps be used, and the present system could be added to computers also configured to accept PPTP communications.

The SmartGATE™ system uses public key and DES encryption to provide two-way authentication and 56-bit encrypted communications between a server equipped with the SmartGATE program and client computers equipped with a separate program. Currently, SmartGATE™ operates at the highest level, or applications level, by using shared secret keys to generate a session key for use in further communications between the authentication server or gateway and the client program. Since the session key depends on the secret keys at the gateway and client sides of the communication, mutual authentication is established during generation of the session key, which can then be used to encrypt further communications.

When installed on a client system, the SmartGATE™ client software reads a request for communications by an applications program, such as a browser program, and then proceeds to establish its own communications link with the destination server to determine if the server is an authentication server. If it is not, control of communications is relinquished, but if it is, then the security program and the server carry out a challenge/response routine in order to generate the session key, and all further communications are encrypted by the security program. Although this program is placed between the Winsock layer and the applications, it does not function as a shim, however, because it only affects communications directed to the authentication server.

Having briefly summarized the concepts used by the present invention, including the concepts of layers, protocols, and shims, and having described a specific applications level security program which is to be modified according to the present invention by adding shims in a way which enables secured authentication and session key generation channels to be set up from the lower layers, it should now be possible to understand the nature of the invention, and in particular how it integrates the two approaches to virtual private networking in a way which greatly expands the concept and yet can easily be implemented. More details will be given below, but as a final observation in this background portion of the patent specification, it should be noted that while the overall concept of the invention is in a sense very simple, it is fundamentally at odds with present approaches. For example, the literature is replete with references to conflicts between VPN standards and implementations, as exemplified by the title of an article from LAN Times On-Line, 9/96, (http://www.wcmh.com/), which reads *Clash Over VPN Supremacy*. Even a cursory search of the available literature indicates that the amount of information and choices available to those wishing to set up a virtual private network is overwhelming. One can choose between

Netscape Communications Secure Socket Layer, Open Market Inc.'s Secure HTTP, Microsoft's PPTP, among others. However, all of these approaches operate at a single level, and force a choice between establishing a network of the type shown in Fig. 1A and a network of the type shown in Fig. 1B. Only the present invention offer the advantages of both approaches, without the inflexibility of client/server arrangements or the costs of more distributed architectures.

For further information on the various competing VPN protocols and systems, see also *The Development of Network Security Technologies*, Internet Smartsec, 2/97 (http://www.smartsec.se), which compares SmartGATE™ to other application level security systems, including PPTP, SSL, and S-HTTP; *Point-To-Point Tunneling Protocol (PPTP) Frequently Asked Questions*, Microsoft Corp., date unknown, (http://www.microsoft.com), *Simple Key-Management for Internet Protocols (SKIP)*, Aziz *et al.*, date unknown, (http://skip.incog.com), and *SOCKS Protocol Version 5, RFC 1928*, Leech *et al.*, 3/96 (http://andrew2.andrew.cmu.edu) (this document describes a protocol involving a TDI shim). For more general information on security problems, Internet protocols, and sockets, see *Introduction to the Internet Protocols*, Charles L. Hedrick, Rutgers University, 1987 (http://oac3.hsc.uth.tmc.edu); *Windows Sockets - Where Necessity is the Mother of Reinvention*, Stardust

Technologies, Inc., 1996, (http//www.stardust.com), and *Secure Internet Connections*, LAN Times, 6/17/96 (Ibid).

SUMMARY OF THE INVENTION

It is accordingly a principal objective of the invention to provide a client/server virtual private network which is capable not only of carrying out authenticated secure communications over an open network between an authentication server and clients, but also authenticated secure peer-to-peer communications.

It is also an objective the invention to provide a virtual private network that provides data encryption and mutual authentication for both client/server and peer-to-peer communications for different-types of applications, using both the applications level and lower levels of a communications hierarchy.

It is a further objective of the invention to provide a client/server virtual private network which can provide both client/server and peer-to-peer encryption and authentication services for any application sharing a specified socket or sockets, whether or not the application is recognized by the encryption and authentication program.

It is a still further objective of the invention to provide a client/server virtual private network which can provide encryption and authentication services at the applications level, transport driver interface level, and network interface level, without the need for modifying either the communication driver or network driver, or any sockets utilizing the communications driver interface.

It is yet another objective of the invention to provide a virtual private network which provides encryption and authentication services for peer-to-peer communications while maintaining centralized control of key distribution and management functions.

Finally, it is also an objective of the invention to provide a virtual private network which provides encryption and authentication services for peer-to-peer communications and in which registration is carried out by a central gateway server.

These objectives of the invention are accomplished by providing a virtual private network for communicating between a server and clients over an open network and in which the clients are equipped with an applications level encryption and mutual authentication program which includes at least one shim positioned above either the socket, transport driver interface, or network interface layers of a client computers communications hierarchy, and which intercepts function calls or data packets in order to authenticate the parties to the communication by establishing secured channels between the server and the parties to the communication, prior to establishment of the secured communications link between the parties, in order to carry out mutual authentication and session key generation functions.

More particularly, according to the principles of a preferred embodiment of the invention, client communications software is provided which, at the socket or transport driver interface levels, intercepts function calls to the socket or transport driver and directs calls to the authentication server in order to perform encryption and authentication routines, and at the network driver interface, performs encryption and authentication functions by intercepting the datagrams or data portions of the packets transmitted by the transport driver interface based on communications between the authentication server and the client. According to this aspect of the invention, a system of providing authentication and encryption services for the purpose of establishing a virtual private network includes a plurality of shims arranged to operate at different protocol levels in order to establish a common secure communications link to an authentication server.

In one especially preferred embodiment of the invention, the client software includes a Winsock shim arranged to intercept function calls to the Winsock library on a client machine and redirect initial communications through the authentication client software to the authentication server, so that any function calls to the Winsock library of programs are intercepted by the shim and carried out by the applications level security program. In this embodiment, the client authentication software substitutes its own function calls for the original function calls in order to establish a secured communications link to the authentication server over which such functions as mutual authentication between the client and server, indirect authentication of peer applications by the now trusted server, session key generation, are carried out, as well as ancillary functions such as on-line registration (OLR), utilizing the unmodified original Winsock library and TCP/IP communications stacks.

By inserting a shim at the Winsock level, an applications level client/server based security program such as SmartGATE™ can be used to provide secure communications for any application which utilizes the Winsock library. In addition, by including analogous shims at other levels, the invention can be used to secure virtually any communications application, including those which by-pass the TDI layer and communicate directly with the network driver level.

Instead of the current array of mutually exclusive alternative methods and systems of establishing secured communications over an open network, the invention thus provides a single integrated method and system capable of carrying out both client/server communications and peer-to-peer communications between a wide variety of communications applications regardless of whether the applications use a socket or even commonly accepted internet protocols, with complete mutual authentication and encryption of data files at all levels and between all parties to the network.

It will be appreciated that the term "virtual private network" is not to be taken as limiting, and that the principles of the invention can be applied to any remote access schemes which utilize the Internet or other relatively insecure networks to provide access for remote users, corporate intranets, and electronic commerce.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1A is a schematic diagram of a client/server virtual private network.

Fig. 1B is a schematic diagram of an alternative virtual private network based on peer-to-peer communications.

Fig. 2 is a functional block diagram showing the operation of an applications level security program in a conventional communications network hierarchy.

Fig. 3 is a functional block diagram showing the communications network hierarchy of Fig. 1, modified to provide a second layer of service in accordance with the principles of a preferred embodiment of the invention.

Fig. 4 is a functional block diagram showing the communications network hierarchy of Fig. 2, modified to provide a third layer of service in accordance with the principles of the preferred embodiment.

Fig. 5 is a functional block diagram showing the communication network hierarchy of Fig. 3, modified to provide a fourth layer of service in accordance with the principles of the preferred embodiment.

Fig. 6 is a schematic diagram of a virtual private network utilizing the principles of the preferred embodiment of the invention.

Fig. 7 is a flowchart illustrating a method of implementing the system of the preferred embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Fig. 2 illustrates the operation of a client authentication program which is utilized in the present invention. An example of such a program is the SmartGATE™ program discussed briefly above, although other applications level security programs, whether or not token based, could be modified in a manner similar to that discussed in the following description. The illustrated hierarchy is the Windows NT architecture, although versions of SmartGATE™ exist for other architectures, and the invention could easily be adapted for use with any version of SmartGATE™, including UNIX and MacIntosh versions, as well as for use with applications level security programs designed for communications architectures other than those supported by SmartGATE™. Conversely, it is intended that the present invention can be used with authentication and encryption schemes other than that used by SmartGATE™ and disclosed in U.S. Patent No. 5,602,918. For purposes of convenience, therefore, the software represented by SmartGATE™ is simply referred to as client authentication software.

In addition, it noted that the client computer architectures illustrated in Figs. 3-6, which are modified versions of the architecture of Fig. 2, is to be used with an overall network layout such as the one illustrated in Fig. 6, which includes an authentication server that may be a SmartGATE™ server, or another server depending on the client authentication software. The invention is not merely the addition of shims to the client software, but involves the manner in which the shims are used in the establishment of the authentications and key generation links to the server.

Turning to Fig. 2, which provides background for the description of the invention illustrated in Figs. 3-6, the client authentication software 20 is situated above the boundary of the transport or TDI layer 21 and is designed to utilize a socket 22, such as Winsock, to carry out communications with the authentication server 23 shown in Fig. 6 by means of a transport protocol such as TCP/IP, UDP, or the like, which in turn supply datagrams or packets to a hardware driver layer 24, such as NDIS 3.0, of a network or modem connection 25.

In operation, the client authentication software 20 intercepts interconnect calls 26 form client authentication software supported applications 27 and, if the calls are directed to the authentication server 23, or to a server 28 situated on a secured network whose access is controlled by the authentication server, establishes a secured communications link to the server by executing appropriate function calls 29 to the socket library, which in turn transmits function calls 30 to the TDI layer, causing the TDI layer to form datagrams or packets 31. Datagrams or packets 31 are then formatted over packaged for transmission by the hardware drivers 24 and sent to the communications network in the form of Ethernet packets or analog signals 32 containing the original datagrams from the TDI layer. Once the secured communications link has been established, client authentication software 20 encrypts all further data communications 34 from applications 27, which are indicated by dashed lines, before handing them off to the next lower layer in the form of encrypted files 35. The dashed lines are shown in Fig. 2 as extending only to the TDI layer 21, because the datagrams formed by the TDI layer are indistinguishable as to content, but it is to be understood that datagrams or packets 31 carry both the communications used to establish the secure channel, and the encrypted files subsequently sent therethrough.

Finally, in the case of SmartGATE™, the authentication client software utilizes either a smart card or secured file to supply the secret keys used during authentication to generate a session key for encryption of further communications, and also to carry out certain other encryption and authentication functions, although it is of course within the scope of the invention to use key distribution and authentication methods which do not rely on smartcards or tokens, and the tokens are not involved in any of the basic communications functions of the client authentication software 20.

In addition to the applications 27 which communicate with the server via the authentication/encryption software 20, a typical system will have a number of additional software applications 36 and 37 capable of carrying out communications over the open network, but which the authentication client software is not configured to handle, and which are not specifically adapted or intended to carry out communications with the authentication server. These are referred to herein as peer-to-peer applications, and can include applications which use the same sockets as the authentication client software, applications which directly call upon a transport driver interface stack, whether using the same protocol as the authentication client software or another protocol, all of which are intended to be represented by the TDI layer, and applications which are written to call directly upon the hardware drivers. These peer-to-peer applications may have their own encryption and authentication capabilities, but cannot utilize the services of the authentication server or client software, and therefore the function calls made by the applications and the files transmitted are indicated by separate reference numerals 40-43.

It will be appreciated by those skilled in the art that lower layer application programs which generate packets in forms other than those represented by the TDI layer are also possible, and should be considered within the scope of the invention, but at present virtually all open network applications use at least one of the TDI protocols, and thus while these programs may interact directly with the network driver layer, and require a network driver layer shim, as will be discussed below, are illustrated for purposes of convenience as part of the TDI layer applications.

Turning now to a preferred embodiment of the invention, the arrangement shown in Fig. 3 modifies the arrangement of Fig. 2 by adding a socket shim 50 between the socket 22 utilized by the authentication client software 20, the peer-to-peer applications 36 which also utilize the socket 20, and the authentication client software itself. The shim 50 operates by hooking or intercepting call initiation function calls 40 made to the socket and, in response thereto, having the authentication client software initiate communications with the authentication server 23, shown in Fig. 6, in order to carry out the authentication protocol, as will be discussed in more detail below. Shim 50 also causes files 41 intended for the TDI layer to be diverted to the authentication software for encryption based on the session keys generated during the initial communications with the authentication server, and transmission as encrypted files 51 addressed to the peer application, also shown in Fig. 6, which could also be an application on the application server 28.

Since the basic authentication client software is designed to send all communications directly to the authentication server, while the peer-to-peer applications are designed only to communicate with "peers" 45 and not with the authentication server, the principal function of shim 50 is to arrange for the destination of address of the communication to be supplied to both the authentication client software and to authentication server, even though the peer application assumes that it is communicating only with the peer application. This function permits session key encrypted communications to be forwarded directly to the peer application, as illustrated in Fig. 6, while the latter function provides the authentication server with the client address so that the authentication server can establish a secured and authenticated link with the peer application, via authentication client software on the peer computer, and transmit the session key to the peer application or at least enable the peer application to recreate the session so that it can decrypt the encrypted files received directly from the client application.

Thus, while it is appreciated that the use of socket shims is well-known, as mentioned above, the socket shim shown in Fig. 2 has the unique function of enabling direct peer-to-peer communications with mediation by the authentication server, permitting the highest level of authentication service and collateral functions. In addition, because of the mediation by the key server, the peer applications do not need to have a shared secret key, allowing centralized key management, with only the authentication server having access to all of the client's secret keys.

Figs. 4 shows the variation of the client authentication software 20 in which a TDI shim 52 similar in function to the socket shim 50 is provided above the TDI layer. Like the socket shim, implementation of the TDI shim essentially simply involves diverting certain information to the client software in order to establish a communications link with the authentication server, and subsequently perform encryption to obtain encrypted files 54 for transmission directly through the TDI layer in the usual manner. As with the socket shim, TDI shims are not new and can be implemented in known manner, by intercepting TDI service requests, but with the difference from prior TDI shims that the TDI shim works with the authentication software 20 and authentication server to authenticate communications and generate a session key.

Finally, as shown in Fig. 5, a further layer of authentication and encryption may be added by adding a network driver shim 55, either to the arrangement shown in Fig. 3 without the TDI shim, in combination with the TDI shim shown in Fig. 4, or in combination with the TDI shim of Fig. 4 but not the socket shim, to provide for authentication of communications at the network driver layer. At this layer, the shim 55 intercepts IP packets from applications 56, but instead of referring back to the applications level routine, checks the destination address (which can be in TCP format, UDP format, and so forth), establishes a session key by communications with the authentication server, converts the session key into a format which can be used to encrypt the IP packet, and sends the IP packet towards the destination, all by carrying out the necessary operations at the network driver level, in a manner similar to that utilized by the above-mentioned SnareNet software program, but with the difference that the authenticating communications link and key generation is carried out by packets addressed to a corresponding layer 56 of the authentication server, which may be further connected to an applications server 57.

It will be noted that since the IP packets are not distinguishable by content, the network driver layer shim could be used as an additional level of security, rather than as an alternative to applications level encryption, with the encrypted files generated by software 20 being further encrypted by shim 55 before transmission to the authentication server or associated gateway.

The overall system utilizing the authentication client software illustrated in Figs. 3-5 is schematically illustrated in Fig. 6. The principal components of the overall system are the client computers containing software of the type illustrated in Figs. 2-5, including client authentication software 20 and shims 50, 53, and/or 55, and applications with communications capabilities (represented by applications 27, 36, 37, and 56 on one client, and application 45 on the other). For purposes of illustration, the client of Figs. 6 is thus depicted as including applications for communicating at the highest levels, such as the SmartGATE™ proxy application, applications for communicating at the network driver level with corresponding applications connected to the lower layer of the authentication server, and peer-to-peer applications with no capability of communicating with SmartGATE™, but which use sockets or TDI protocols recognized by the shims.

In the case of the SmartGATE™ proxy application, communications are established in the same manner as in the currently available version of the SmartGATE™ authentication client software, and as described in U.S. Patent No. 5,602,918, the communications link being indicated by arrows 60 and 61, with arrow 60 representing the client/server response channel used to authenticate the parties and generate the session key.

In the case of a peer-to-peer application, in which the clients wish to communicate over a direct link 62, the invention provides for the function calls establishing the communications to be intercepted and the initialization procedure routed through channel 61 to the authentication server 23. Server 23 then opens a secured channel 63 to the authentication client software 20 associated with peer application 45 by performing the same mutual authentication procedure performed for the purpose of establishing channel 63, and once the channel is established with its own session key, transmits information using the channel 63 session key which allows the client to recreate the channel 60 session key for use in decrypting communications sent over channel 62. Alternatively, after establishing channel 63, the channel 60 session key could be used to transmit back to the original sending party information necessary to recreate the channel 63 session key. In either case, the authentication server is thus used to establish a fully authenticated "tunnel" between the peer applications without the need to modify any of the sockets, TDI protocols, or hardware drivers on either of the client computers. While the transmitting peer application has no way of directly authenticating the receiving peer, only a receiving peer authenticated by the authentication server will be able to generate the necessary session keys, and thus each of the parties to the communication is effectively authenticated.

For the lower layer application 56, a similar protocol may be employed, in which the attempted communication between lower layer applications is intercepted, and the communications link to the authentication server is used to generate a session key, which is then used to encrypt the packets or datagrams being sent. In this case, the destination must be the lower layer of the authentication server, and thus the communications link is indicated by a separate channel 67.

Finally, the procedures associated with the network illustrated in Fig. 6 are summarized in the flowchart of Fig. 7. For communications directly with the applications level portion of the server 23, steps 100-103 are used, while for peer-to-peer communications, steps 104-109 are used, and for network driver level communications, steps 110-114 are used.

In particular, step 100 by which the applications level authentication program 20 illustrated in Figs. 3-5 receives a call initiation request, either directly from a supported applications program 27 or from a programs 36 and 37 via one of the shims 50 and 53, step 101 is step by which the program 20 addresses the authentication server, step 102 is the step by which the client and server are mutually authenticated and the session keys generated using, for example, the procedure described in U.S. Patent No. 5,602,918, and step 103 is the step by which program 20 encrypts further communications received directly or via shims 50 and 53 from the applications programs 27, 36, and 37.

For peer-to-peer communications, step 105, which is part of step 100, is the step by which the peer address is supplied to program 20, steps 106 and 107 are identical to steps 101 and 102, step 108 is the step by which communications channel 63 shown in Figure 6 is established, step 109 is the step by which the destination computer authenticated by the server is enabled to decrypt communications received over channel 62, and step 110 is the step by which program 20 encrypts the communications. It will of course be appreciated that these steps represent only a summary of the steps involved in carrying out the present invention, and that further steps will be apparent to those skilled in the art based on the above description of the apparatus and software portions of the preferred embodiment of the invention.

Having thus described various preferred embodiments of the invention, those skilled in the art will appreciate that variations and modifications of the preferred embodiment may be made without departing from the scope of the invention. It is accordingly intended that the invention not be limited by the above description or accompanying drawings, but that it be defined solely in accordance with the appended claims.

I claim:

1. Apparatus for carrying out communications over a multi-tier virtual private network, said network including a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network, comprising:

means for intercepting function calls and requests for service sent by an applications program on one of said client computers to a lower level set of communications drivers; and

means for causing an applications level authentication and encryption program in said one of said client computers to communicate with the server, generate said session key, and encrypt files sent by the applications program before transmittal over said open network.

2. Apparatus as claimed in claim 1, further comprising means for intercepting files packaged by a transport driver interface layer to form packets and encrypting the packets using a session key generated during communications with a lower layer of the server.

3. A method as claimed in claim 1, further comprising

means for intercepting a destination address during initialization of communications between said one of said client computers and a second of said client computers on said virtual private network;

means for causing said applications level authentication and encryption program to communicate with the server to carry out functions a.) and b.);

means for transmitting said destination address to said server;

means for causing said server to carry-out functions a.) and b.) with respect to the second of said two client computers;

means for enabling said second of said two client computers to recreate the session key;

means for causing said authentication software to encrypt files to be sent to the destination address using the session key; and

means for transmitting the encrypted files directly to the destination address.

4. Apparatus as claimed in claim 3, wherein said means for intercepting the destination address is carried out by a shim positioned between a peer-to-peer applications program and a layer of a communications driver architecture of said one of the two client computers.

5. A multi-tier virtual private network, comprising:

a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said means for transmitting data to and receiving data from the open network includes, in any client computer initiating communications with the server:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files;

at least one lower level set of communications drivers;

and a shim arranged to intercept function calls and requests for service sent by an applications program to the lower level set of communications drivers in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before transmittal over said open network.

6. A multi-tier virtual private network as claimed in claim 5, wherein said lower level set of communications drivers includes a network driver layer, a transport driver

interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and an applications socket for facilitating service requests by said applications program to the transport driver interface layer, and wherein said shim is a socket shim positioned between the applications program and the socket to intercept function calls to the socket in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

7. A multi-tier virtual private network as claimed in claim 6, wherein said applications program is a peer-to-peer communications program, and wherein a peer application destination address, included in said function calls to the socket, is diverted by the socket shim and wherein a destination address including said intercepted function calls is supplied to the server during communications with the server, causing the service to establish a communications link with a peer application, mutually authenticate the peer application, and enable the peer application to reconstruct the session key in order to receive encrypted files sent by the peer-to-peer communications program over the open network.

8. A multi-tier virtual private network as claimed in claim 6, further including a transport driver interface shim positioned between the transport driver interface layer and a second applications program, for intercepting requests from the second applications program for service by the transport driver interface layer in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

9. A multi-tier virtual private network as claimed in claim 8, further comprising a network driver layer shim positioned between the network driver layer and the transport driver interface layer and arranged to intercept files packaged by the transport driver interface layer and encrypt the files using a session key generated during communications with a lower layer of the server.

10. A multi-tier virtual private network as claimed in claim 5, wherein said lower level set of communications drivers includes a network driver layer, and a transport driver interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and wherein said shim is a transport driver interface layer shim positioned

between the applications program and the transport driver interface layer to intercept service requests by the applications program to the transport driver interface layer in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

11. A multi-tier virtual private network as claimed in claim 10, wherein said applications program is a peer-to-peer communications program, and wherein a peer application destination address, included in said intercepted requests for service, is diverted by the transport driver interface layer shim and supplied to the server during communications with the server, causing the service to establish a communications link with a peer application, mutually authenticate the peer application, and enable the peer application to reconstruct the session key in order to receive encrypted files sent by the peer-to-peer communications program over the open network.

12. A multi-tier virtual private network as claimed in claim 10, further comprising a network driver layer shim positioned between the network driver layer and the transport driver interface layer and arranged to intercept files packaged by the transport driver interface layer and

encrypt the files using a session key generated during communications with a lower layer of the server.

13. A multi-tier virtual private network, comprising:

a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said means for transmitting data to and receiving data from the open network includes, in any client computer initiating communications with the server:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files; and

at least one lower level set of communications drivers,

wherein said lower level set of communications drivers includes a network driver layer, a transport driver interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and a network driver layer shim positioned between the transport driver interface layer and the network driver layer and arranged to intercept files packaged by the transport driver interface layer and encrypt the files using a session key generated during communications with a lower layer of the server.

14. A multi-tier virtual private network, comprising:

a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said means for transmitting data to and receiving data from the open network includes, in any client computer initiating communications with the server:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files; and

further comprising means for securing peer-to-peer communications between applications on two of said client computers, said peer-to-peer communications securing means comprising:

means for intercepting a destination address during initialization of communications by a first of said two client computers;

means for causing said authentication software to communicate with the server to carry out functions a.) and b.);

means for transmitting said destination address to said server;

means for causing said server to carry-out functions a.) and b.) with respect to the second of said two client computers;

means for enabling said second of said two client computers to recreate the session key;

means for causing said authentication software to encrypt files to be sent to the destination address using the session key;

means for transmitting the encrypted files directly to the destination address.

15. A multi-tier virtual private network as claimed in claim 14, wherein said means for intercepting the destination address comprises a shim positioned between the peer-to-peer applications program and a layer of a communications driver architecture of said first of the two client computers.

16. A multi-tier virtual private network as claimed in claim 5, wherein said shim is positioned above a socket,

the socket being positioned above a transport driver layer of said communications driver architecture.

17. A multi-tier virtual private network as claimed in claim 5, wherein said shim is positioned above a transport driver layer of said communications driver architecture.

18. Computer software for installation on a client computer of a multi-tier virtual private network, said network including a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said computer software includes:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files;

and a shim arranged to intercept function calls and requests for service sent by an applications program to a lower level set of communications drivers in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before transmittal over said open network.

19. Computer software as claimed in claim 18, wherein said lower level set of communications drivers includes a network driver layer, a transport driver interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and an applications socket for facilitating service requests by said applications program to the transport driver interface layer, and wherein said shim is a socket shim positioned between the applications program and the socket to intercept function calls to the socket in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

20. Computer software as claimed in claim 19, wherein said applications program is a peer-to-peer communications program, and wherein a peer application destination address, included in said function calls to the socket, is diverted by the socket shim and wherein a destination address including said intercepted function calls is supplied to the server during communications with the

server, causing the service to establish a communications link with a peer application, mutually authenticate the peer application, and enable the peer application to reconstruct the session key in order to receive encrypted files sent by the peer-to-peer communications program over the open network.

21. Computer software as claimed in claim 19, further including a transport driver interface shim positioned between the transport driver interface layer and a second applications program, for intercepting requests from the second applications program for service by the transport driver interface layer in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

22. Computer software as claimed in claim 21, further comprising a network driver layer shim positioned between the network driver layer and the transport driver interface layer and arranged to intercept files packaged by the transport driver interface layer and encrypt the files using a session key generated during communications with a lower layer of the server.

23. Computer software as claimed in claim 18, wherein said lower level set of communications drivers includes a

network driver layer, and a transport driver interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and wherein said shim is a transport driver interface layer shim positioned between the applications program and the transport driver interface layer to intercept service requests by the applications program to the transport driver interface layer in order to cause the applications level authentication and encryption program to communicate with the server, generate said session key, and encrypt files sent by the applications program before the files are packaged by the transport driver interface layer.

24. Computer software as claimed in claim 23, wherein said applications program is a peer-to-peer communications program, and wherein a peer application destination address, included in said intercepted requests for service, is diverted by the transport driver interface layer shim and supplied to the server during communications with the server, causing the service to establish a communications link with a peer application, mutually authenticate the peer application, and enable the peer application to reconstruct the session key in order to receive encrypted files sent by the peer-to-peer communications program over the open network.

25. Computer software as claimed in claim 23, further comprising a network driver layer shim positioned between the network driver layer and the transport driver interface layer and arranged to intercept files packaged by the transport driver interface layer and encrypt the files using a session key generated during communications with a lower layer of the server.

26. Computer software for installation on a client computer of a multi-tier virtual private network, said network including a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said computer software includes:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files; and

at least one lower level set of communications drivers,

wherein said lower level set of communications drivers includes a network driver layer, a transport driver interface layer arranged to package applications files as packets capable of being routed over the open network and supply the packets to the network driver layer for transmission to the open network, and a network driver layer shim positioned between the transport driver interface layer and the network driver layer and arranged to intercept files packaged by the transport driver interface layer and encrypt the files using a session key generated during communications with a lower layer of the server.

27. Computer software for installation on a client computer of a multi-tier virtual private network, said network including a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network,

wherein said computer software includes:

applications level encryption and authentication software arranged to communicate with the server in order to: a.) mutually authenticate the server and the client computer initiating communications with the server and b.) generate a session key for use by the client computer initiating communications to encrypt files; and

further comprising means for securing peer-to-peer communications between applications on two of said client computers, said peer-to-peer communications securing means comprising:

means for intercepting a destination address during initialization of communications by a first of said two client computers;

means for causing said authentication software to communicate with the server to carry out functions a.) and b.);

means for transmitting said destination address to said server;

means for causing said server to carry-out functions a.) and b.) with respect to the second of said two client computers;

means for enabling said second of said two client computers to recreate the session key;

means for causing said authentication software to encrypt files to be sent to the destination address using the session key;

means for transmitting the encrypted files directly to the destination address.

28. Computer software as claimed in claim 27, wherein said means for intercepting the destination address comprises a shim positioned between the peer-to-peer applications program and a layer of a communications driver architecture of said first of the two client computers.

29. Computer software as claimed in claim 27, wherein said shim is positioned above a socket, the socket being positioned above a transport driver layer of said communications driver architecture.

30. Computer software as claimed in claim 27, wherein said shim is positioned above a transport driver layer of said communications driver architecture.

31. A method of carrying out communications over a multi-tier virtual private network, said network including a server and a plurality of client computers, the server and client computers each including means for transmitting data to and receiving data from an open network, comprising the steps of:

intercepting function calls and requests for service sent by an applications program in one of said client computers to a lower level set of communications drivers;

causing an applications level authentication and encryption program said one of said client computers to communicate with the server, generate said session key, and encrypt files sent by the applications program before transmittal over said open network.

32. A method as claimed in claim 31, further comprising the step of intercepting files packaged by a transport driver interface layer to form packets and encrypting the

packets using a session key generated during communications with a lower layer of the server.

33. A method as claimed in claim 31, further comprising the step of intercepting a destination address during initialization of communications between said one of said client computers and a second of said client computers on said virtual private network;

causing said applications level authentication and encryption program to communicate with the server to carry out functions a.) and b.);

transmitting said destination address to said server;

causing said server to carry-out functions a.) and b.) with respect to the second of said two client computers;

enabling said second of said two client computers to recreate the session key;

causing said authentication software to encrypt files to be sent to the destination address using the session key; and

transmitting the encrypted files directly to the destination address.

34. A method as claimed in claim 33, wherein said step of intercepting the destination address is carried out by a shim positioned between a peer-to-peer applications program

and a layer of a communications driver architecture of said one of the two client computers.

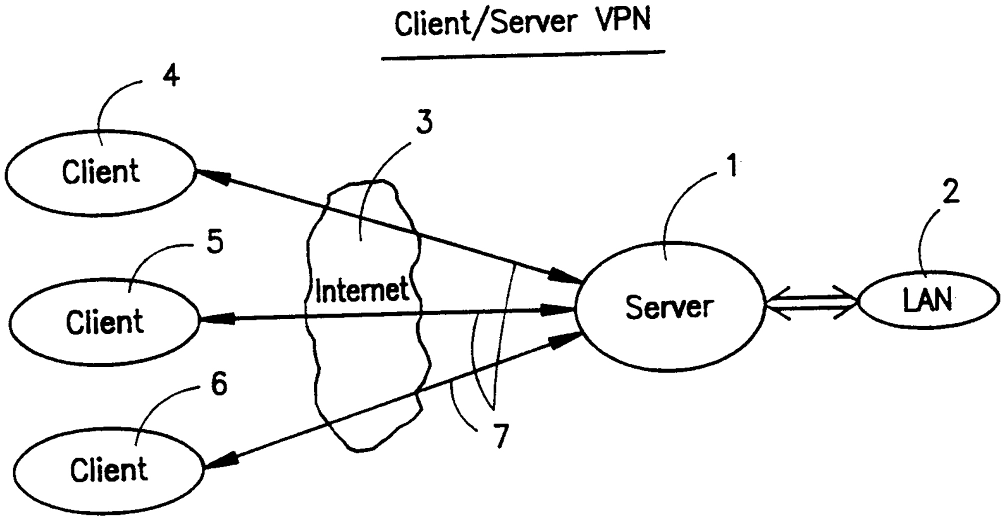


FIG. 1A
(PRIOR ART)

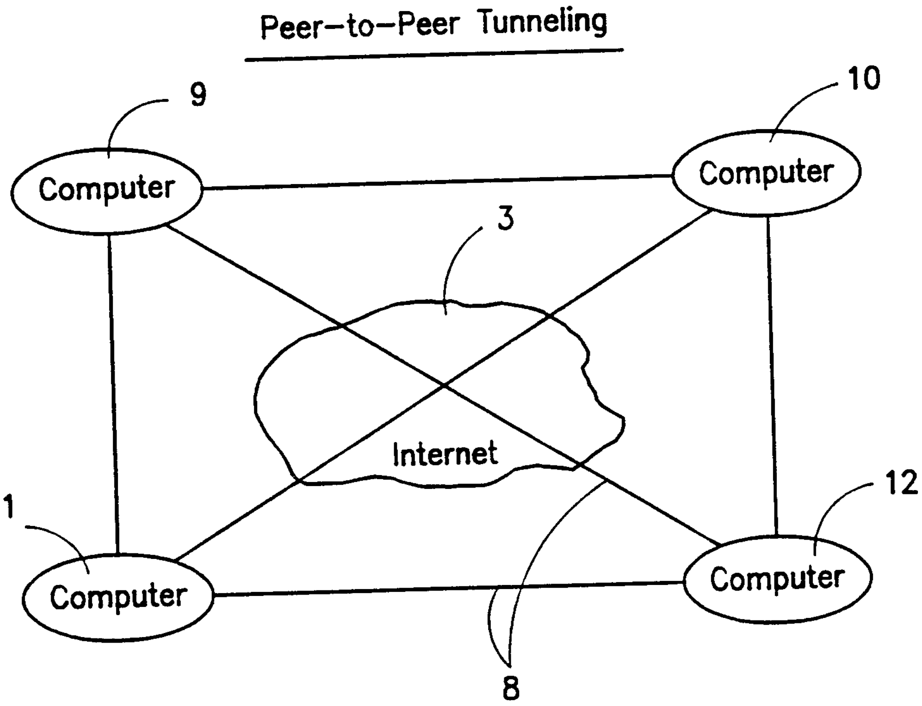


FIG. 1B
(PRIOR ART)

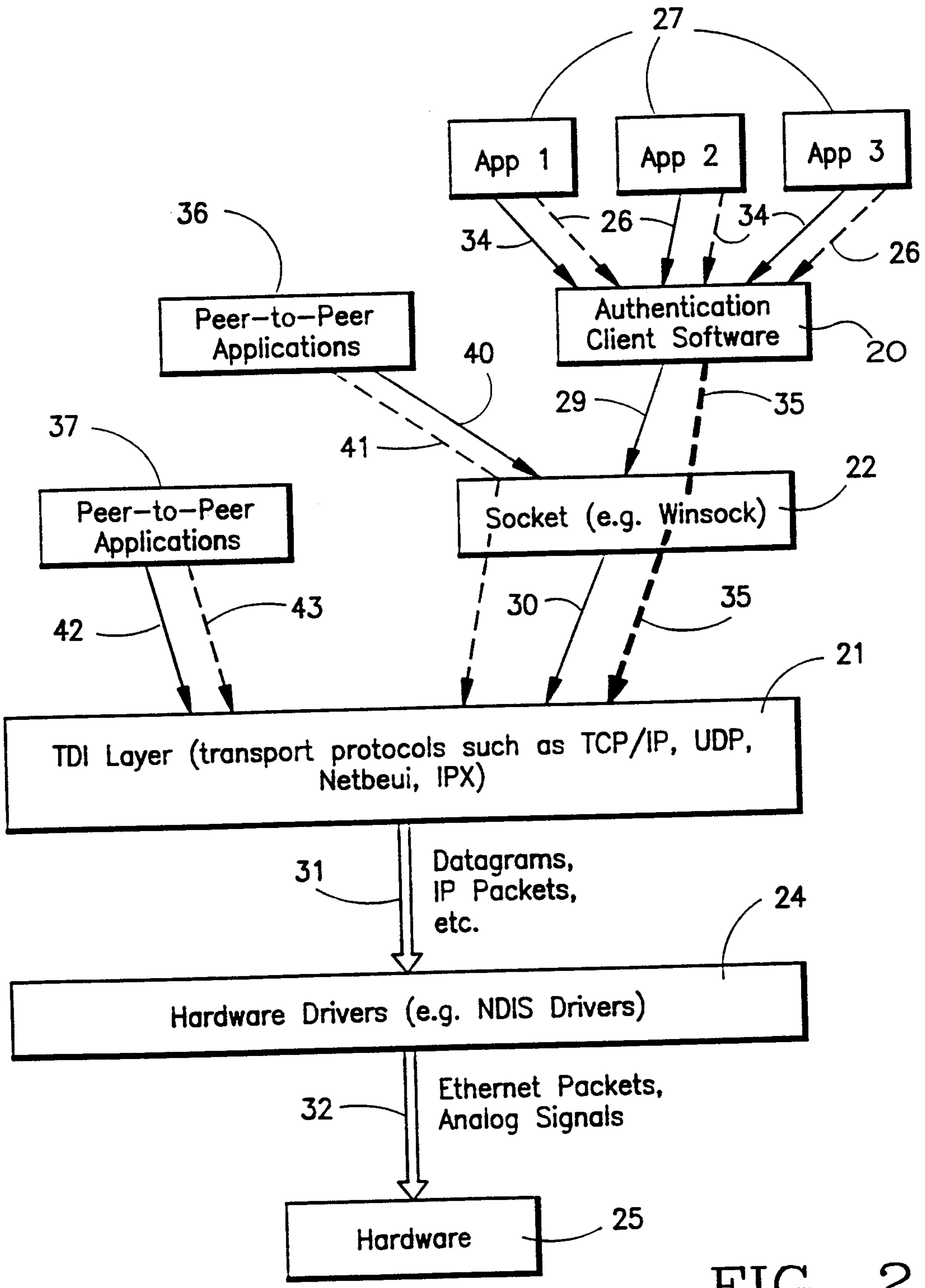

27
App 1
App 2
App 3
36
34
26
34
26
Peer–to–Peer Applications
Authentication Client Software
20
40
29
35
37
41
22
Peer–to–Peer Applications
Socket (e.g. Winsock)
42
43
30
35
21
TDI Layer (transport protocols such as TCP/IP, UDP, Netbeui, IPX)
31
Datagrams, IP Packets, etc.
24
Hardware Drivers (e.g. NDIS Drivers)
32
Ethernet Packets, Analog Signals
Hardware
25

FIG. 2
(PRIOR ART)

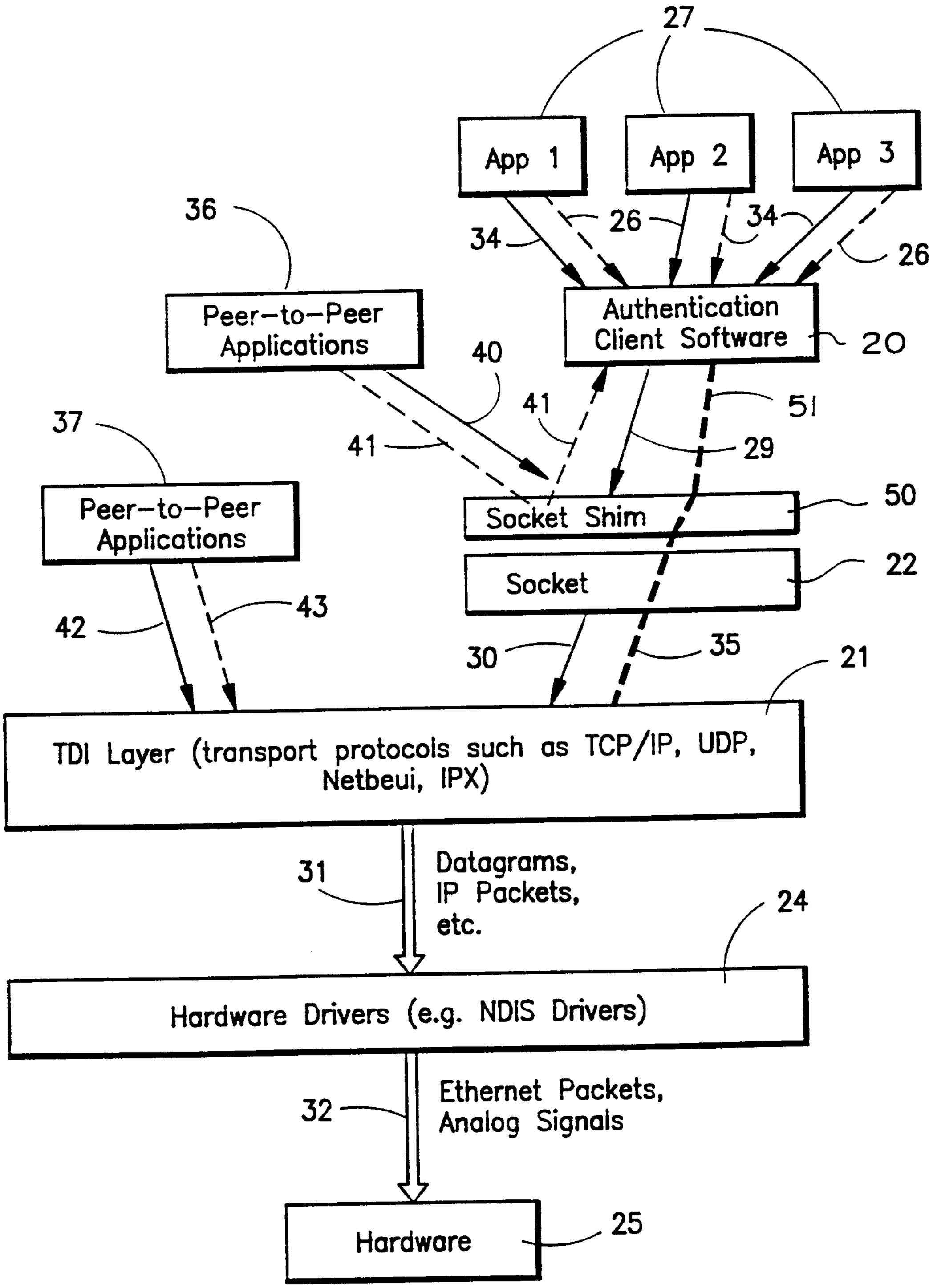

27
App 1
App 2
App 3
34
26
34
26
36
Peer–to–Peer Applications
Authentication Client Software
20
40
41
51
41
29
37
Peer–to–Peer Applications
Socket Shim
50
Socket
22
42
43
30
35
21
TDI Layer (transport protocols such as TCP/IP, UDP, Netbeui, IPX)
31
Datagrams, IP Packets, etc.
24
Hardware Drivers (e.g. NDIS Drivers)
32
Ethernet Packets, Analog Signals
Hardware
25

FIG. 3

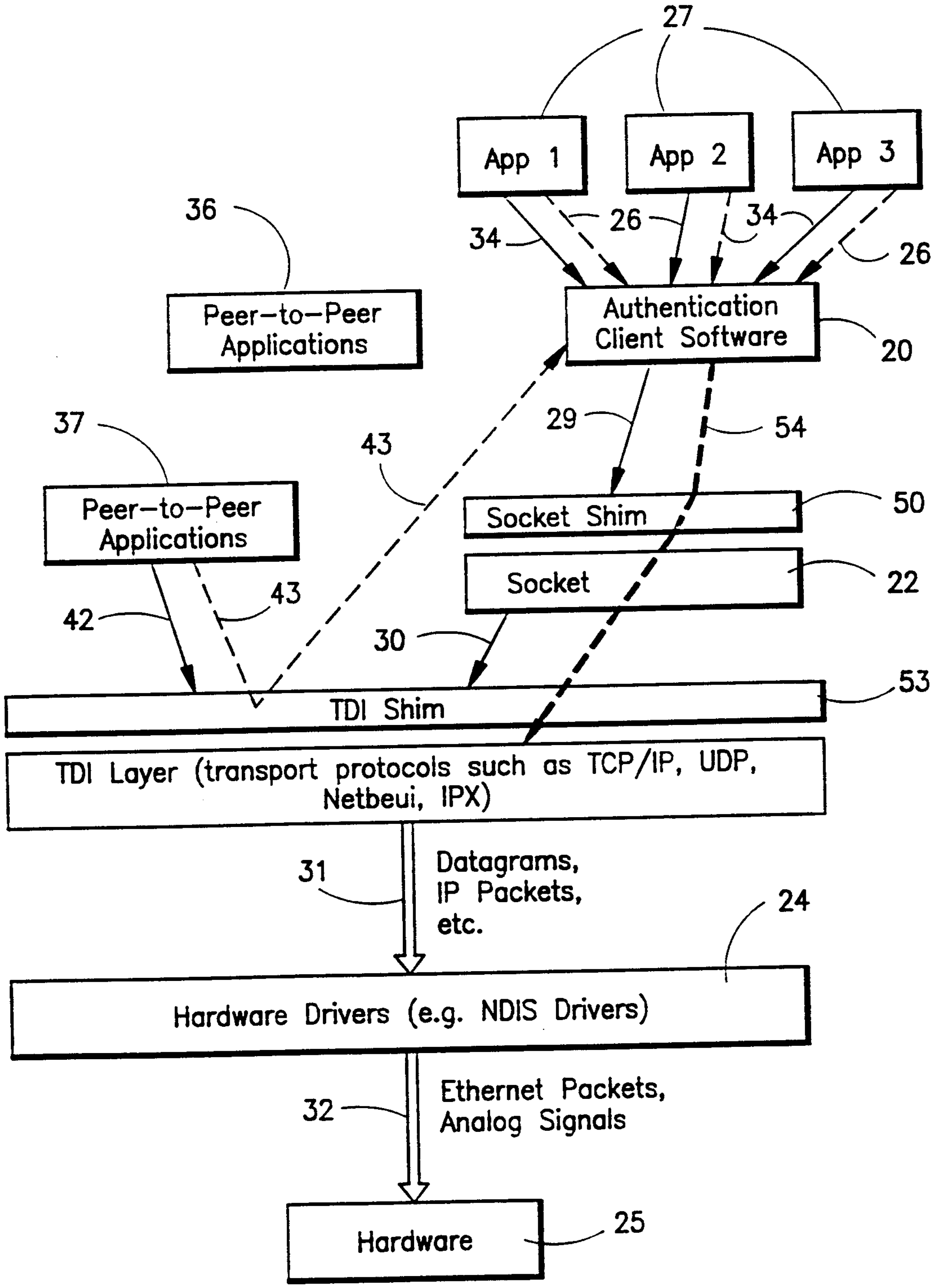

27
App 1
App 2
App 3
36
34
26
34
26
Peer–to–Peer Applications
Authentication Client Software
20
37
43
29
54
Socket Shim
50
Peer–to–Peer Applications
Socket
22
42
43
30
53
TDI Shim
TDI Layer (transport protocols such as TCP/IP, UDP, Netbeui, IPX)
31
Datagrams, IP Packets, etc.
24
Hardware Drivers (e.g. NDIS Drivers)
32
Ethernet Packets, Analog Signals
Hardware
25

FIG. 4

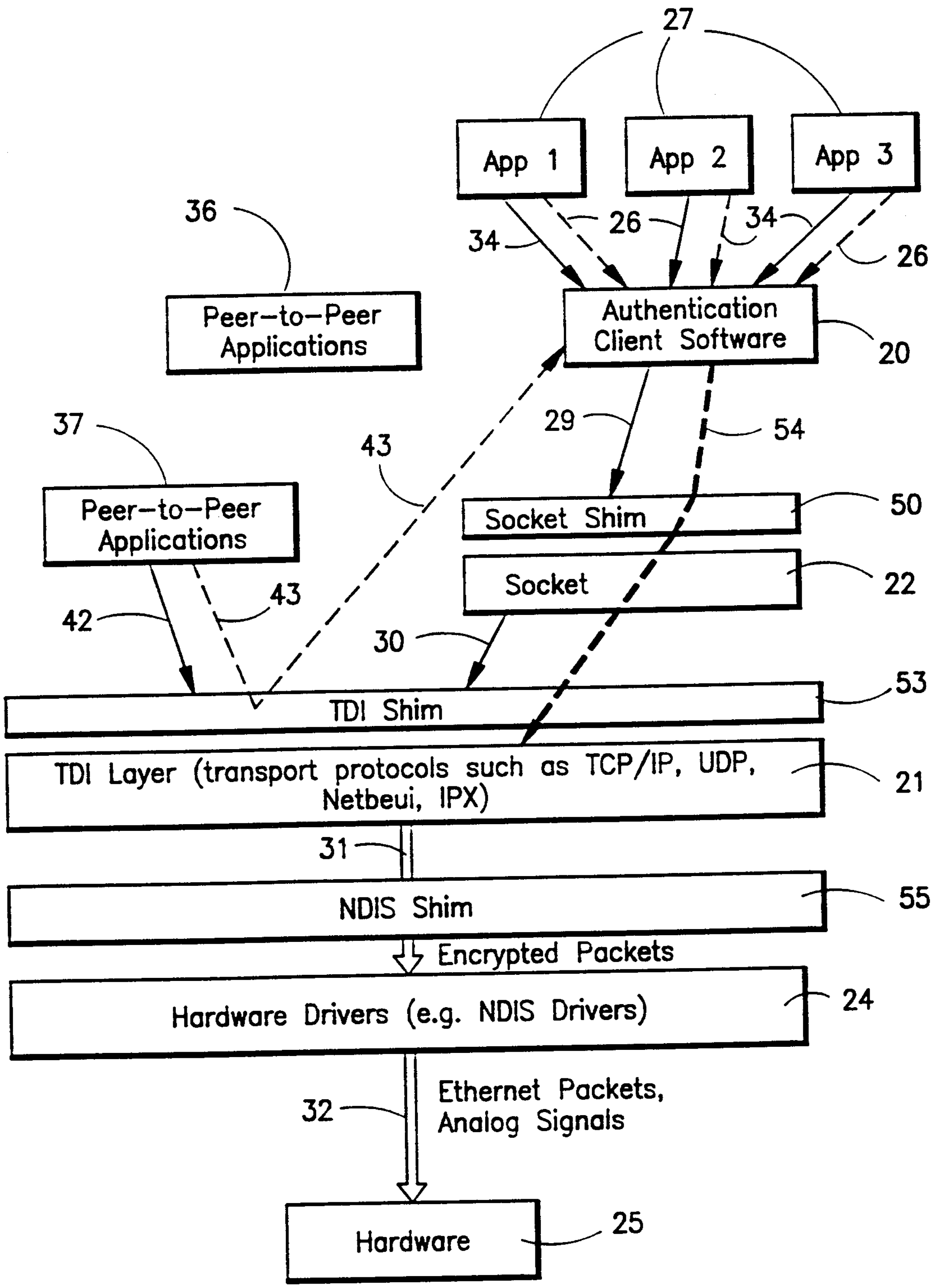

27
App 1
App 2
App 3
34
26
34
26
36
Peer–to–Peer Applications
Authentication Client Software
20
37
43
29
54
Peer–to–Peer Applications
Socket Shim
50
Socket
22
42
43
30
TDI Shim
53
TDI Layer (transport protocols such as TCP/IP, UDP, Netbeui, IPX)
21
31
NDIS Shim
55
Encrypted Packets
Hardware Drivers (e.g. NDIS Drivers)
24
32
Ethernet Packets, Analog Signals
Hardware
25

FIG. 5

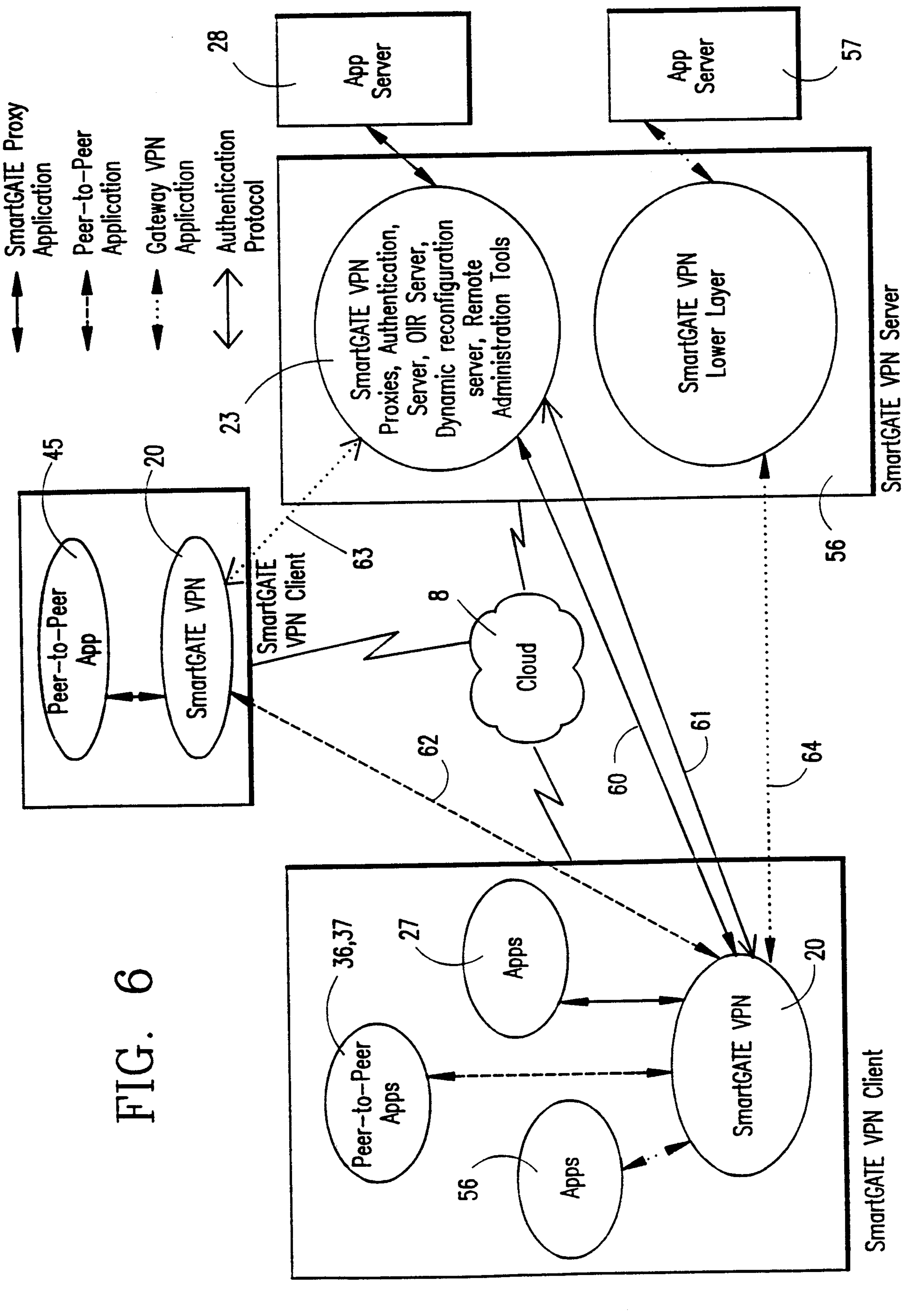

SmartGATE Proxy Application
Peer-to-Peer Application
Gateway VPN Application
Authentication Protocol
28
App Server
57
App Server
23
SmartGATE VPN Proxies, Authentication, Server, OIR Server, Dynamic reconfiguration server, Remote Administration Tools
SmartGATE VPN Lower Layer
SmartGATE VPN Server
56
45
20
Peer-to-Peer App
SmartGATE VPN
SmartGATE VPN Client
63
8
Cloud
62
60
61
64
36,37
27
Apps
20
Peer-to-Peer Apps
SmartGATE VPN
56
Apps
SmartGATE VPN Client

FIG. 6

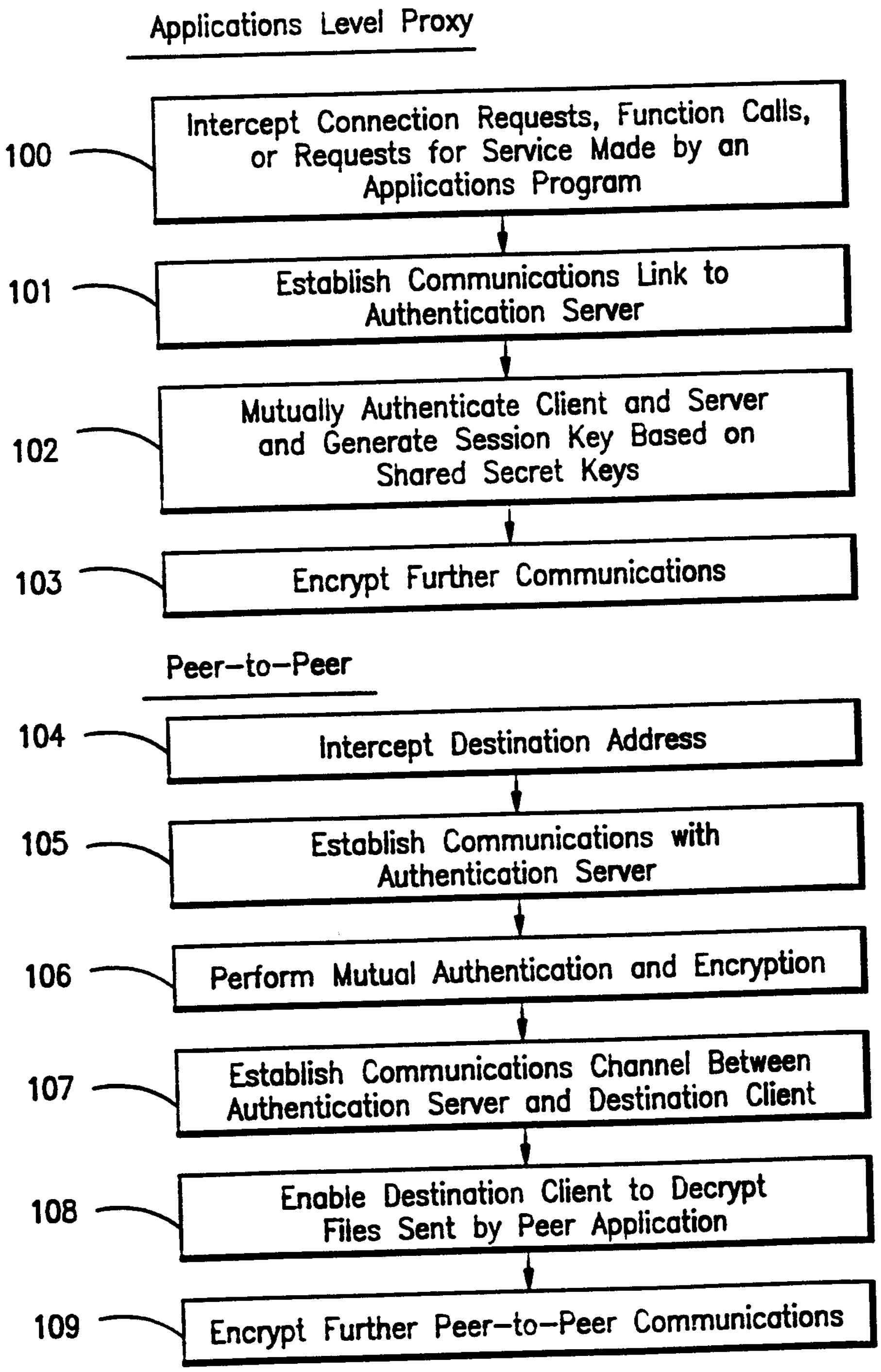

Applications Level Proxy
100
Intercept Connection Requests, Function Calls, or Requests for Service Made by an Applications Program
101
Establish Communications Link to Authentication Server
102
Mutually Authenticate Client and Server and Generate Session Key Based on Shared Secret Keys
103
Encrypt Further Communications
Peer-to-Peer
104
Intercept Destination Address
105
Establish Communications with Authentication Server
106
Perform Mutual Authentication and Encryption
107
Establish Communications Channel Between Authentication Server and Destination Client
108
Enable Destination Client to Decrypt Files Sent by Peer Application
109
Encrypt Further Peer-to-Peer Communications

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/17198

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :H04L 9/00
US CL :395/187.01
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/187.01, 186, 188.01, 200.17, 200.12; 380/49, 21, 25, 4

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS, STN, IEEE ProQuest
search terms: virtual private network, shims, DLLs, protocol layers, Winsock, sockets, encryption, authentication.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,657,390 A (ELGAMAL ET AL) 12 AUGUST 1997, FIGURES 1-8, COL. 3, LINES 20-55, COL. 5, LINE 15 TO COL. 8, LINE 32, COL. 11, LINE 1 TO COL. 16, LINE 49.	1,5,6,16,17,18,19,23, 31
A	US 5,602,918 A (CHEN ET AL) 11 FEBRUARY 1997, SEE ENTIRE PATENT.	1-34
A	US 5,550,984 A (GELB) 27 AUGUST 1996, ABSTRACT, COL. 3, LINE 52 TO COL. 4, LINE 45, COL.6, LINES 27-55.	1-34
Y	HURWICZ, A VIRTUAL PRIVATE AFFAIR, BYTE MAGAZINE, JULY 1997, PAGES 79-87.	1,5,6,16,17,18,19,23, 31

□ Further documents are listed in the continuation of Box C. □ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search 22 OCTOBER 1998	Date of mailing of the international search report 12 NOV 1998
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer JOSEPH PALYS Telephone No. (703) 305-9600

(12) UK Patent Application (19) GB (11) 2 334 181 (13) A

(43) Date of A Publication 11.08.1999

(21) Application No 9802545.5

(22) Date of Filing 06.02.1998

(71) Applicant(s)
NEC Technologies (UK) Ltd
(Incorporated in the United Kingdom)
Castle Farm Campus, Priorslee, TELFORD, Shropshire,
TF2 9SA, United Kingdom

(72) Inventor(s)
Charles Marie Herve Noblet

(74) Agent and/or Address for Service
J W White
NEC Technologies (UK) Ltd, Level 3, The Imperium,
Imperial Way, READING, Berks, RG2 0TD,
United Kingdom

(51) INT CL^6
H04Q 7/32

(52) UK CL (Edition Q)
H4L LDSC

(56) Documents Cited
US 5613204 A US 5109403 A

(58) Field of Search
UK CL (Edition P) H4L LDSC LDSU LECC LECX
INT CL^6 H04Q 7/32 7/38
Online: WPI

(54) Abstract Title

Over-the-air re-programming of radio transceivers

(57) A method of downloading reprogramming data from a network for installation in a mobile station makes use of a dedicated small bandwidth pilot channel. The mobile station obtains from the base station the radio access parameters of a second channel. The second channel is a large bandwidth (bootstrap) channel suitable for fast transfer of data. The bootstrap channel is logically mapped onto a local transmission mode such as DECT or GSM by the mobile station and re-programming data may be downloaded from the base station via the bootstrap channel.

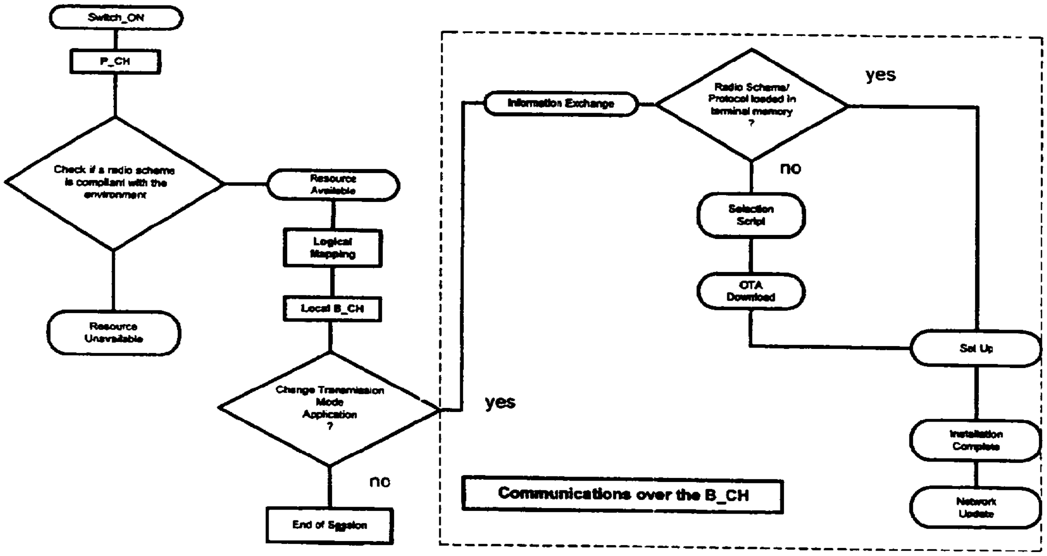


Figure 2

GB 2 334 181 A

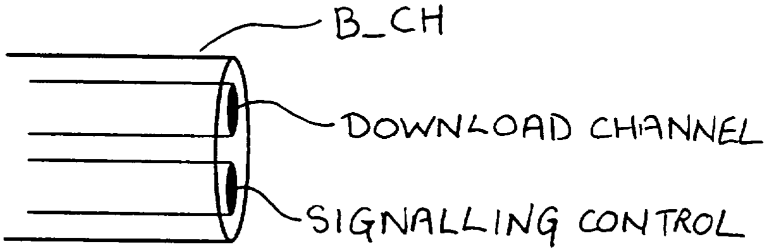


Figure: 1

2/3

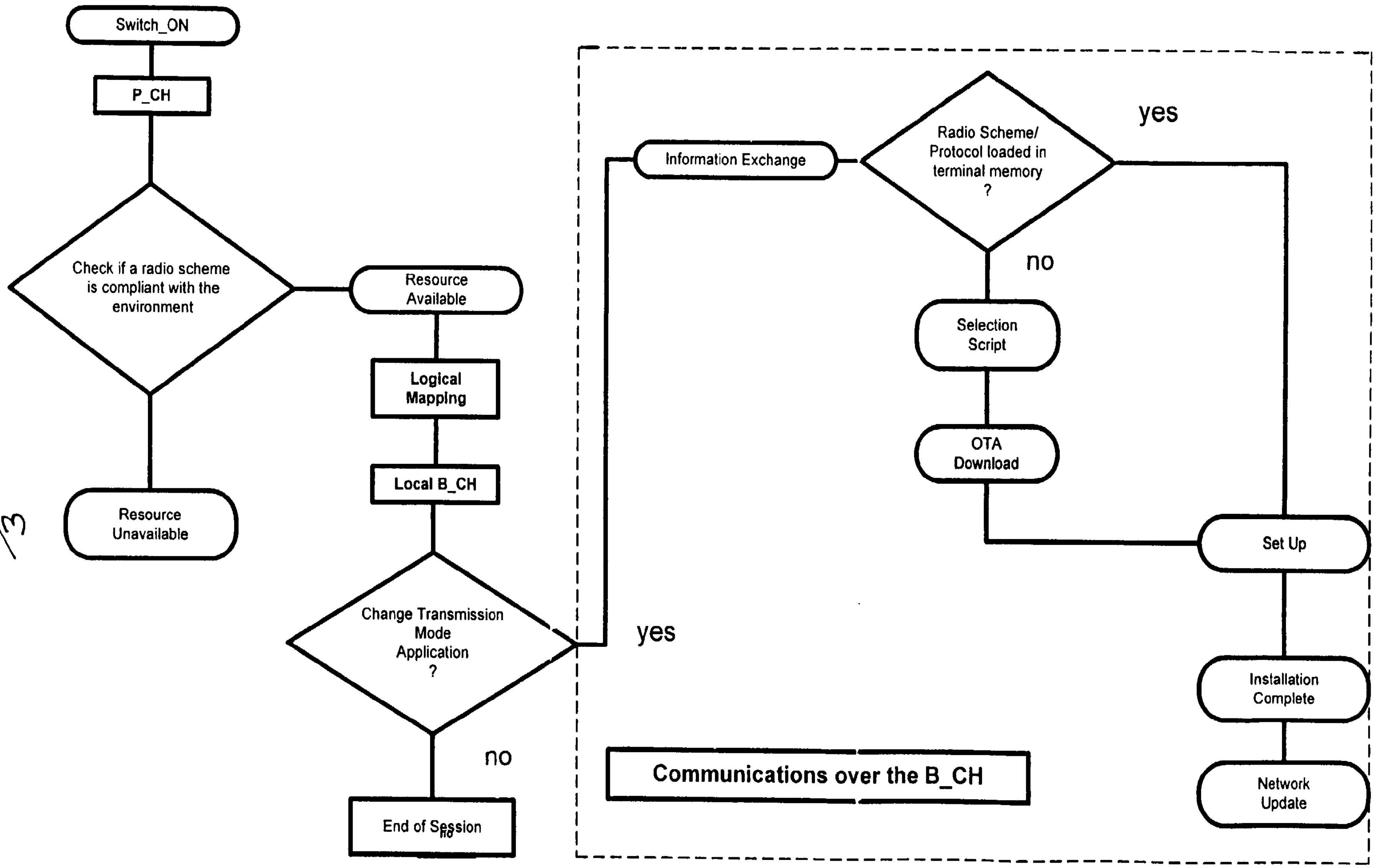

Switch_ON
P_CH
Check if a radio scheme is compliant with the environment
Resource Unavailable
Resource Available
Logical Mapping
Local B_CH
Change Transmission Mode Application ?
no
End of Session
yes
Information Exchange
Radio Scheme/ Protocol loaded in terminal memory ?
yes
no
Selection Script
OTA Download
Set Up
Installation Complete
Network Update
Communications over the B_CH

Figure 2

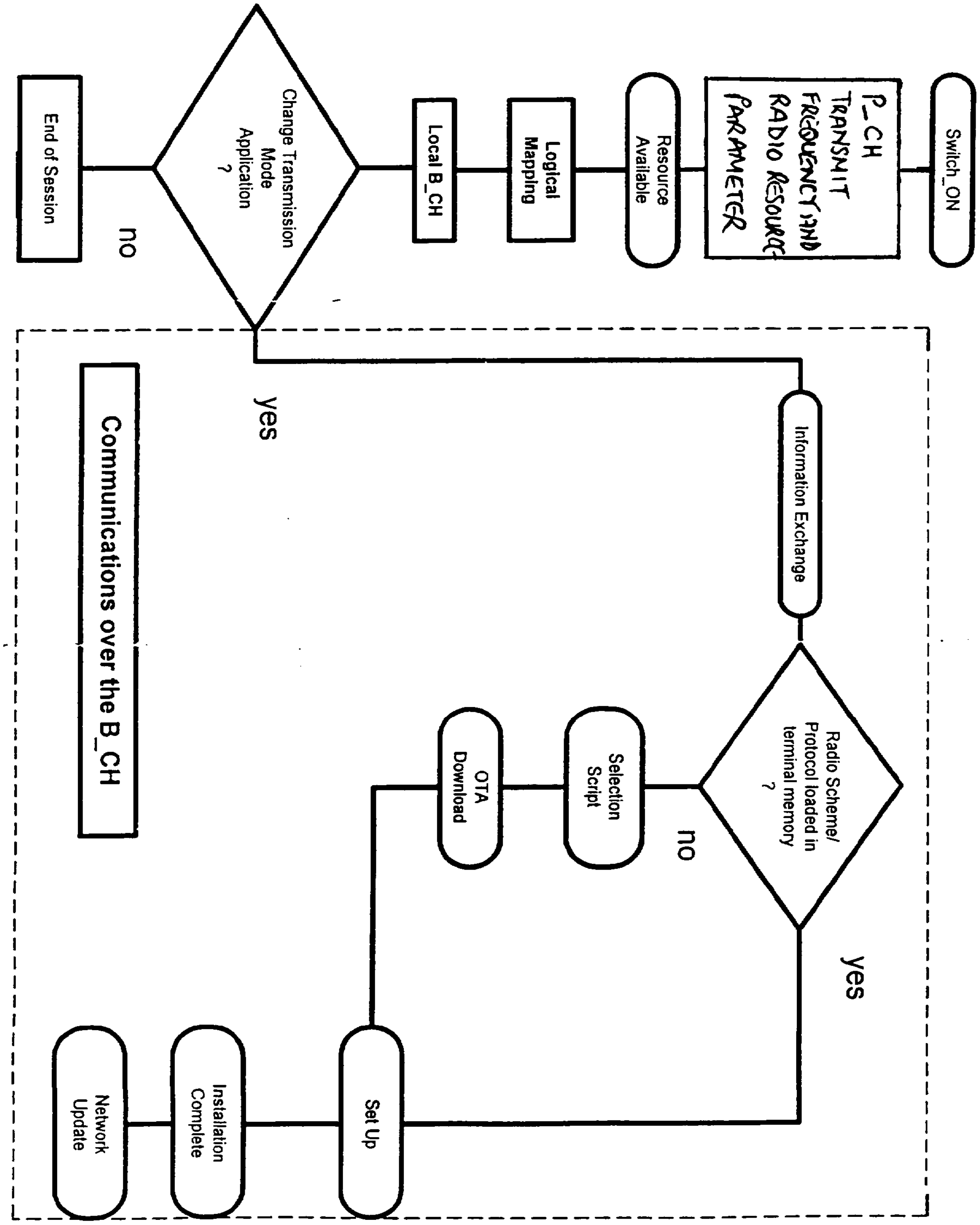

Switch_ON
P_CH TRANSMIT FREQUENCY AND RADIO RESOURCE PARAMETER
Resource Available
Logical Mapping
Local B_CH
Change Transmission Mode Application ?
no
End of Session
yes
Information Exchange
Radio Scheme/ Protocol loaded in terminal memory ?
no
Selection Script
OTA Download
yes
Set Up
Installation Complete
Network Update
Communications over the B_CH

Figure 3

2334181

Over-the-air re-programming of radio transceivers

This invention relates to radio transmitter/receivers and in particular it relates to a method of re-programming radio transmitter/receivers over-the - air.

A radio transmitter/receiver (transceiver) such as a radiotelephone is designed for operation with particular types of networks such as GSM 900 or DCS 1800. Intended use of the radiotelephone with a particular network(s) in a restricted geographical area, however, requires that the telephone be configured so as properly to communicate with the particular network (s). The user of a radiotelephone will usually have a telephone which has been configured for communication with a so called "home network". The home network is the local network usually most used by the subscriber.

The area within which a user of e.g. a GSM radiotelephone may operate, however, is considerable and is not limited to the home network but may be used on many other networks throughout the world. Use of a handset outside the home network is known as "roaming".

When the radiotelephone is to be used in roaming it is often necessary for it to have a configuration different to that for use with the home network. It is possible for re-configuration of radio transmitter/receivers to be effected by means of signals received across the air interface.

It is also convenient for the radio to be re-configurable over the air interface so as to support different types of communication and user applications e.g. addition of address book manager, whether or not it is located in the home network.

Over the air re-programming of radio receivers is well known in the art and reference may be made to US patent 5 381 138 for example. The capability to obtain programming data from a network is particularly useful for a roaming radio transmitter/receiver.

When beginning operation in an area for which the radiotelephone is not configured and it is required to download the data for reconfiguration from one of the available networks, a communication link must first be established with the network of interest. It has been proposed that a pilot channel be established in all areas from which the roaming radiotelephone may obtain the data necessary for reconfiguration.

A pilot channel of this type, however, will require a relatively large bandwidth to allow a sufficiently fast transfer of the data required.

According to the invention there is provided a method of downloading reprogramming data from a network for installation in a radio transmitter/receiver comprising initial communication from a first dedicated channel of relatively small bandwidth broadcasting at least the frequency and radio access parameters of a second channel of relatively large bandwidth from which reprogramming data may be downloaded.

Examples of the invention will now be described in more detail with reference to the accompanying figures in which

figure 1 Illustrates the logical structure of the bootstrap channel

.igure 2 Is a flow diagram of a reconfiguration process

figure 3 Is a flow diagram of an alternative reconfiguration process

A roaming radio transmitter/receiver (mobile) is located in a region served by one or more networks and the user wishes to communicate with a network from which he can obtain reprogramming data and subsequently begin communicating with the network in the communication mode selected.

A pilot channel broadcast is maintained in the region and contained in the pilot channel broadcast there is at least sufficient information for the mobile to connect to a second channel which we shall call the bootstrap channel. Conveniently the pilot channel will be broadcast in all regions over a standardised radio interface. Only a small bandwidth is required for the pilot channel because of the small amount of information contained in the broadcast.

The small bandwidth requirement makes the task of standardisation much easier with respect to the pilot channel. The wider bandwidth channels are more conveniently assigned locally for ease of implementation.

The Pilot Channel (P_CH) broadcasts a list of sets of parameters corresponding to networks available in the region. The mobile receives the network transmission through the P_CH. If the existing configuration of the mobile is matched to the available regional radio schemes, then a second channel the bootstrap channel (B_CH) is logically mapped onto the selected transmission mode. The base station and mobile exchange information over this dedicated logical channel.

The Bootstrap channel is logically mapped on top of one of the default modes of the terminal; a mapping of a logical B_CH onto the physical GSM channel for instance may be implemented. Once the mapping has been effected the terminal may download data from the base station. The bootstrap channels provided by each operator may accommodate differing services with regard to the applications available for downloading.

The flow diagram shown at fig 3 depicts a reconfiguration procedure.

When the mobile is switched on, it reads the Pilot Channel broadcast. The mobile must be configured to support the (standardised) radio interface of the Pilot Channel. The Pilot Channel carries local radio parameters (standards supported in the regional environment in which the mobile is located). After processing the received information, the mobile communicates with the base station through the Bootstrap Channel, provided that the mobile has the minimum resources required by its local radio environment. Prior to the change of channel, P_CH to B_CH, a logical mapping of the Bootstrap Channel is performed within the mobile on the selected air interface.

When operation on a local B_CH transmission has been established, the user may wish to change some properties or the performance of his mobile and can request supply of the desired services from the network. If no changes are required then the mobile adopts the default transmission mode in stand-by and releases the allocated B_CH.

If the user requests a change then communication between the base station and mobile is maintained for the exchange, the nature of which will depend on the capabilities of both mobile and network. At least 3 conditions can affect the nature of this information exchange.

Firstly, the mobile may not be able to support the required software. Where the mobile is not able to support the required software, no communication channel is available to the mobile from the existing network resources and use of the mobile within the region will therefore not be possible.

Secondly, the required software may be stored already in the mobile's memory. In this situation there is no need to download a software module but the allocated B_CH connection is maintained for further operations as described.

Thirdly, the software module required to support a different type of communication or user application may need to be downloaded from the base station. Where the download of a software module is required, initially a selection script is downloaded to the mobile followed by downloading and installation of the required software.

When the installation of the required software into the mobile has been completed, the mobile signals to the network the achievement of correct reconfiguration. On receipt of the "correct reconfiguration" signal from the mobile details of the mobile identity and its present configuration are entered on the network database (to license the product for instance) .

With reference to figure 1, the logical structure of the bootstrap channel will include 2 logical sub-channels : a download channel and a signalling control channel (S_CH). The signalling control channel assists in the reduction of errors in transmission so as to allow correct software download.

In the above example, the first channel, the Pilot Channel, is standardised and the mobile must be configured to support the radio interface for the Pilot Channel. The second (bootstrap) channel may be subject to local definition through logical mapping on a local transmission mode e.g. GSM, DECT and the mobile is not initially configured to support the radio interface for the bootstrap channel..

An example of a method of reprogramming providing greater flexibility will now be given. In this example the mobile is configured to support the radio interfaces for both the first, dedicated relatively small bandwidth (Pilot) channel and the second relatively large bandwidth (bootstrap) channel. That is to say that when the mobile is switched on in most and preferably all regions, the network can communicate with the mobile via both pilot and bootstrap channels.

In order for the mobile always to have the appropriate radio interface for the bootstrap channel then this channel would need also to be standardised (in addition to the Pilot Channel). The parameters of the bootstrap channels provided in different regions may have local variations in terms of e.g. allocated frequency, data rate and available user applications.

With reference to figure 3 which is a flow diagram of the reconfiguration process for this example, the mobile when switched on reads the Pilot Channel broadcast. The allocated frequency and radio resource parameters for the bootstrap channel contained in the pilot channel broadcast are processed and any required logical mapping effected. After processing the received information, the mobile communicates with the base station through the Bootstrap Channel.

The condition likely to be experienced in the previous example whereby the mobile is not able to support the required software and no communication channel is available to the mobile from the existing network resources does not apply in this arrangement. The communication via the bootstrap channel allows the request for and supply of the software module necessary to establish communication with the network. The transfer to the bootstrap channel does not depend on the existing configuration of the mobile since the bootstrap channel is standardised in this example and the mobile is equipped to interface, via the pilot channel, with the bootstrap channel.

The services and structure offered by the Bootstrap Channel are common for both of the above examples, however, the requirements on the terminals and networks differ.

The bootstrap channel will provide the following services by means of over -the-air (OTA) reconfiguration :
capability Exchange - the terminal provides some information to the network on its current configuration and capabilities.
module Selection : at this stage the user specifies the software that his terminal requires to download. This operation could be compared to an installation script.
data download : transfer of the data. In some cases software code will have to be downloaded whilst in other cases the software may already be implemented in the mobile. In the latter case, a set-up mechanism would be sufficient to initiate the reconfiguration.

Once the mobile and the base station are synchronised on the bootstrap channel, information exchange can begin.

Claims

1. A method of downloading reprogramming data from a network for installation in a radio transmitter/receiver comprising initial communication from a first dedicated channel of relatively small bandwidth broadcasting at least the frequency and radio access parameters of a second channel of relatively large bandwidth from which reprogramming data may be downloaded.

2. A method of downloading reprogramming data from a network as in claim 1 where first, dedicated relatively small bandwidth channel has a standard radio interface common to many network locations.

3. A method of downloading reprogramming data from a network as in claim 2 where second relatively large bandwidth channel has a standard radio interface common to many network locations.

4. A method of downloading reprogramming data from a network as in claims 1 to 3 where first, dedicated relatively small bandwidth channel broadcasts a list of sets of parameters corresponding to networks available in the region.

5. A method of downloading reprogramming data from a network as in claim 1 where the radio transmitter/receiver is configured to support the radio interfaces for both the first, dedicated relatively small bandwidth channel and the second relatively large bandwidth channel.

The Patent Office

10

Application No:	GB 9802545.5	**Examiner:**	Glyn Hughes
Claims searched:	1 to 5	**Date of search:**	17 August 1998

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.P): H4L (LDSC, LDSU, LECC, LECX)

Int Cl (Ed.6): H04Q 7/32, 7/38

Other: Online: WPI

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
X	US 5613204	(HABERMAN ET AL) see in particular column 15 lines 48 to 50	1
X	US 5109403	(SUTPHIN) see abstract	1

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

New Bay Capital, LLC
Ex. 1007

(12) UK Patent Application (19) GB (11) 2 340 702 (13) A

(43) Date of A Publication 23.02.2000

(21) Application No 9912200.4

(22) Date of Filing 25.05.1999

(30) Priority Data
(31) 09087823 (32) 29.05.1998 (33) US

(71) Applicant(s)
Sun Microsystems Inc
(Incorporated in USA - Delaware)
901 San Antonio Road, MS Palo Alto-521,
California 94303, United States of America

(72) Inventor(s)
Joseph E Provino

(74) Agent and/or Address for Service
D Young & Co
21 New Fetter Lane, LONDON, EC4A 1DA,
United Kingdom

(51) INT CL^7
H04L 29/06 // H04L 9/00 12/22 12/46

(52) UK CL (Edition R)
H4P PPEB

(56) Documents Cited
EP 0887979 A2 EP 0825748 A2 WO 98/31124 A1

(58) Field of Search
UK CL (Edition Q) H4P PPA PPEB PPEC PPG
INT CL^6 H04L 12/22 12/46 12/66 29/06
ONLINE DATABASES: WPI, EPODOC, JAPIO

(54) Abstract Title

Accessing a server in a virtual private network protected by a firewall

(57) A virtual private network 15 has a firewall 30, at least one server 31 and a nameserver 32 each having a network address (eg. an n-bit integer address). The server 31 also has a secondary address (eg. a human readable address) and the nameserver 32 provides an association between the secondary address and the network address. An authorised external device 12 establishes a secure tunnel between itself and the firewall for communication using encryption. When the external device requests connection to server 31 using the secondary address of server 31, the firewall provides external device 12 with the network address of the nameserver 32. The external device 12 transmits a request for resolution of the network address associated with the secondary address to the nameserver through the firewall. The nameserver then transmits the network address of the server 31 through the firewall to the external device using the secure tunnel. The external device can thereafter use the network address of server 31 in subsequent communications.

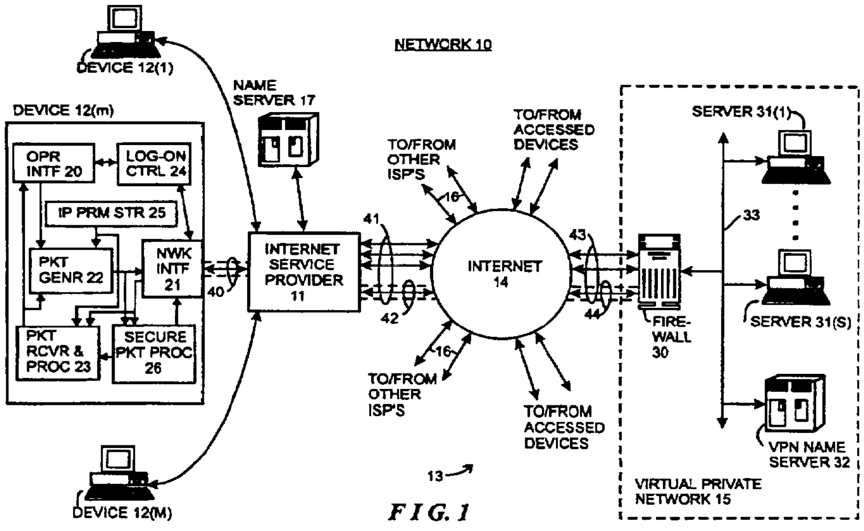


FIG. 1

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

This print takes account of replacement documents submitted after the date of filing to enable the application to comply with the formal requirements of the Patents Rules 1995

GB 2 340 702 A

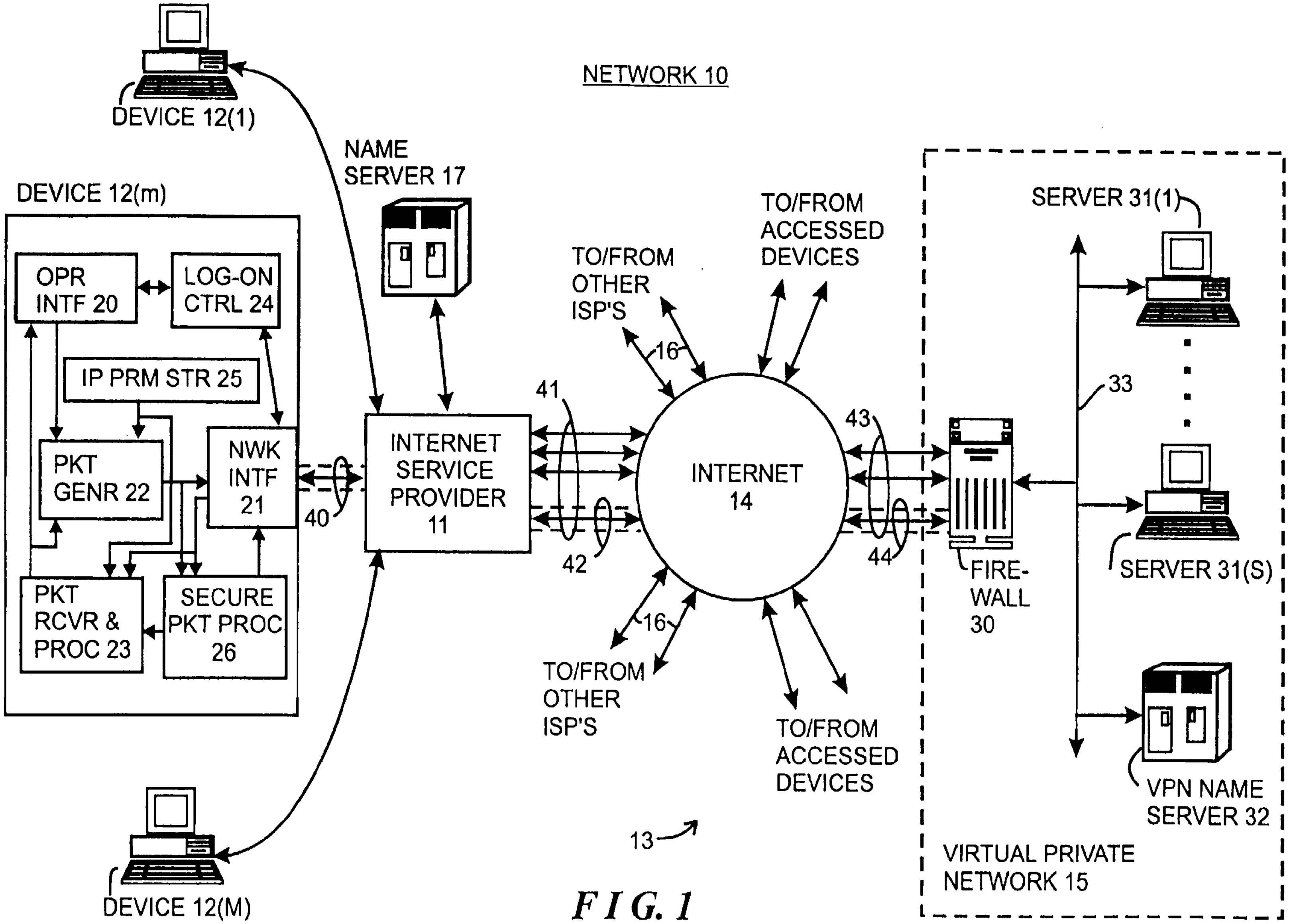

NETWORK 10
DEVICE 12(1)
NAME SERVER 17
DEVICE 12(m)
OPR INTF 20
LOG-ON CTRL 24
IP PRM STR 25
PKT GENR 22
NWK INTF 21
PKT RCVR & PROC 23
SECURE PKT PROC 26
40
INTERNET SERVICE PROVIDER 11
DEVICE 12(M)
41
42
TO/FROM OTHER ISP'S
16
TO/FROM ACCESSED DEVICES
INTERNET 14
16
TO/FROM OTHER ISP'S
TO/FROM ACCESSED DEVICES
13
43
44
FIRE-WALL 30
SERVER 31(1)
33
SERVER 31(S)
VPN NAME SERVER 32
VIRTUAL PRIVATE NETWORK 15

FIG. 1

FIELD OF THE INVENTION

The invention relates generally to the field of digital communications systems and methods, and more particularly to systems and methods for easing communications between devices connected to public networks such as the Internet and devices connected to private networks.

BACKGROUND OF THE INVENTION

Digital networks have been developed to facilitate the transfer of information, including data and programs, among digital computer systems and other digital devices. A variety of types of networks have been developed and implemented, including so-called "wide-area networks" (WAN's) and "iocal area networks" (LAN's), which transfer information using diverse information transfer methodologies. Generally, LAN's are implemented over relatively small geographical areas, such as within an individual office facility or the like, for transferring information within a particular office, company or similar type of organization. On the other hand, WAN's are generally implemented over relatively large geographical areas, and may be used to transfer information between LAN's as well as between devices that are not connected to LAN's. WAN's also include public networks, such as the Internet, which can carry information for a number of companies.

Several problems have arisen in connection with communication over a network, particularly a large public WAN such as the Internet. Generally, information is transferred over a network in message packets, which are transferred from one device, as a source device, to another device as a destination device, through one or more routers or switching nodes (generally, switching nodes) in the network. Each message packet includes a destination address which the switching nodes use to route the respective message packet to the appropriate destination device. Addresses over the Internet are in the form of an "n"-bit integer (where "n" may be thirty two or 128), which are difficult for a person to remember and enter when he or she wishes to enable a message packet to be transmitted. To relieve a user of the necessity of remembering and entering specific integer Internet addresses, the Internet provides second addressing mechanism which is more easily utilized by human operators of the respective devices. In that addressing mechanism, Internet domains, such as LAN's, Internet service providers ("ISP's") and the like which are connected in the Internet, are identified by relatively human-readable names. To accommodate the use of human-readable names, nameservers, also referred to as DNS servers, are provided to resolve the human-readable names to the appropriate Internet addresses. When an operator at one device, wishing to transmit a message packet to another device, enters the other device's human-readable name, the device will initially contact a nameserver. Generally, the nameserver may be part of the ISP itself or it may be a particular device which is accessible through the ISP over the Internet; in any case, the ISP will identify the nameserver to be used to the device when the device logs in to the ISP. If, after being contacted by the device, the nameserver has or can obtain an integer Internet address for the human-readable domain name, it (that is, the nameserver) will provide the integer Internet address corresponding to the human-readable domain name to the operator's device. The device, in turn, can thereafter include the integer Internet address returned by the nameserver in the message packet and provide the message packet to the ISP for transmission over the Internet in a conventional manner. The Internet switching nodes use the integer Internet address to route the message packet to the intended destination device.

Other problems arise, in particular, in connection with the transfer of information over a public WAN such as the Internet. One problem is to ensure that information transferred over the WAN that the source device and the destination device wish to maintain confidential, in fact, remains confidential as against possible eavesdroppers which may intercept the information. To maintain confidentiality, various forms of encryption have been developed and are used to encrypt the information prior to transfer by the source device, and to decrypt the information after it has been received by the destination device. If it is desired that, for example, all information transferred between a particular source device and a particular destination device is maintained confidential, the devices can establish a "secure tunnel" therebetween, which essentially ensures that all information to be transferred by the source device to the destination device is encrypted (except for certain protocol information, such as address information, which controls the flow of network packets through the network between the source and destination devices) prior to transfer, and that the encrypted information will be decrypted prior to utilization by the destination device. The source and destination devices may themselves perform the encryption and decryption, respectively, or the encryption and decryption may be performed by other devices prior to the message packets being transferred over the Internet.

A further problem that arises in particular in connection with companies, government agencies, and private organizations whose private networks, which may be LAN's, WAN's or any combination thereof, are connected to public WAN's such as the Internet, is to ensure that their private networks are secure against others whom the companies do not wish to have access thereto, or to regulate and control access by others whom the respective organizations may wish to have limited access. To accommodate that, the organizations typically connect their private networks to the public WAN's through a limited number of gateways sometimes referred to as "firewalls," through which all network traffic between the internal and public networks pass. Typically, network addresses of domains and devices in the private network "behind" the firewall are known to nameservers which are provided in the private network, but are not available to nameservers or other devices outside of the private network, making communication between a device outside of the private network and a device inside of the private network difficult.

SUMMARY OF THE INVENTION

Particular and preferred aspects of the invention are set out in the accompanying independent and dependent claims. Features of the dependent claims may be combined with those of the independent claims as appropriate and in combinations other than those explicitly set out in the claims.

The invention provides a new and improved system and method for easing communications between devices connected to public networks such as the Internet and devices connected to private networks by facilitating resolution of secondary addresses, such as the Internet's human-readable addresses, to network addresses by nameservers or the like connected to the private networks.

In brief summary, an embodiment of the invention provides a system comprising a virtual private network and an external device interconnected by a digital network. The virtual private network has a firewall, at least one internal device and a nameserver each having a network address. The internal device also has a secondary address, and the nameserver is configured to provide an association between the secondary address and the network address. The firewall, in response to a request from the external device to establish a connection therebetween, provides the external device with the network address of the nameserver. The external device, in response to a request from an operator or the like, including the internal device's secondary address, requesting access to the internal device, generates a network address request message for transmission over the connection to the firewall requesting resolution of the network address associated with the secondary address. The firewall provides the address resolution request to the nameserver, and the nameserver provides the network address associated with the secondary address to the firewall. The firewall, in turn, provides the network address in a network address response message for transmission over the connection to the external device. The external device can thereafter use the network address so provided in subsequent communications with the firewall intended for the internal device.

BRIEF DESCRIPTION OF THE DRAWINGS

Exemplary embodiments of the invention are described hereinafter, by way of example only, with reference to the accompanying drawings, in which:

FIG. 1 is a functional block diagram of a network constructed in accordance with the invention.

DETAILED DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

FIG. 1 is a functional block diagram of a network 10 constructed in accordance with the invention. The network 10 as depicted in FIG. 1 includes an Internet service provider ("ISP") 11 which facilitates the transfer of message packets among one or more devices 12(1) through 12(M) (generally identified by reference numeral 12(m)) connected to ISP 11, and other devices, generally identified by reference numeral 13, over the Internet 14, thereby to facilitate the transfer of information in message packets among the devices 12(m) and 13. The ISP 11 connects to the Internet 14 over one or more logical connections or gateways or the like (generally referred to herein as "connections") generally identified by reference numeral 41. The ISP 11 may be a public ISP, in which case it connects to devices 12(m) which may be controlled by operators who are members of the general public to provide access by those operators to the Internet. Alternatively, ISP 11 may be a private ISP, in which case the devices 12(m) connected thereto are generally operated by, for example, employees of a particular company or governmental agency, members of a private organization or the like, to provide access by those employees or members to the Internet.

As is conventional, the Internet comprises a mesh of switching nodes (not separately shown) which interconnect ISP's 11 and devices 13 to facilitate the transfer of message packets thereamong. The message packets transferred over the Internet 14 conform to that defined by the so-called Internet protocol "IP" and include a header portion, a data portion, and may include a error detection and/or correction portion. The header portion includes information used to transfer the message packet through the Internet 14, including, for example, a destination address that identifies the device that is to receive the message packet as the destination device and a source address that identifies the device which generated the message packet. For each message packet, the destination and source addresses are each in the form of an integer that uniquely identifies the respective destination and source devices. The switching nodes comprising the Internet 14 use at least the destination address of each respective message packet to route it (that is, the respective message packet) to the destination device, if the destination device is connected to the Internet, or to an ISP 11 or other device connected to the Internet 14, which, in turn, will forward the message packet to the appropriate destination. The data portion of each message packet includes the data to be transferred in the message packet, and the error detection and/or correction portion contains error detection and/or correction information which may be used to verify that the message packet was correctly transferred from the source to the destination device (in the case of error detection information), and correct selected types of errors if the message packet was not correctly transferred (in the case of error correction information).

The devices 12(m) connected to ISP 11 may comprise any of a number of types of devices which communicate over the Internet 14, including, for example, personal computers, computer workstations, and the like, with other devices 13. Each device 12(m) communicates with the ISP 11 to transfer message packets thereto for transfer over the Internet 14, or to receive message packets therefrom received by the ISP 11 over the Internet 14, using any convenient protocol such as the well-known point-to-point protocol ("PPP") if the device 12(m) is connected to the ISP 11 using a point-to-point link, any conventional multi-drop network protocol if the device 12(m) is connected to the ISP 11 over a multi-drop network such as the Ethernet, or the like. The devices 12(m) are generally constructed according to the conventional stored-program computer architecture, including, for example, a system unit, a video display unit and operator input devices such as a keyboard and mouse. A system unit generally includes processing, memory, mass storage devices such as disk and/or tape storage elements and other elements (not separately shown), including network and/or telephony interface devices for interfacing the respective device to the ISP 11. The processing devices process programs, including application programs, under control of an operating system, to generate processed data. The video display unit permits the device to display processed data and processing status to the user, and the operator input device enables the user to input data and control processing.

These elements of device 12(m), along with suitable programming, cooperate to provide device 12(m) with a number of functional elements including, for example, an operator interface 20, a network interface 21, a message packet generator 22, a message packet receiver and processor 23, an ISP log-on control 24, an Internet parameter store 25 and, in connection with the invention, a secure message packet processor 26. The operator interface 20 facilitates reception by the device

12(m) of input information from the operator input device(s) of device 12(m) and the display of output information to the operator on the video display device(s) of the device 12(m). The network interface 21 facilitates connection of the device 12(m) to the ISP 11 using the appropriate PPP or network protocol, to transmit message packets to the ISP 11 and receive message packets therefrom. The network interface 21 may facilitate connection to the ISP 11 over the public telephone network to allow for dial-up networking of the device 12(m) over the public telephone system. Alternatively or in addition, the network interface 21 may facilitate connection through the ISP 11 over, for example, a conventional LAN such as the Ethernet. The ISP log on control 24, in response to input provided by the operator interface 20 and/or in response to requests from programs (not shown) being processed by the device 12(m), communicates through the network interface 21 to facilitate the initialization ("log-on") of a communications session between the device 12(m) and the ISP 11, during which communications session the device 12(m) will be able to transfer information, in the form of, message packets with other devices over the Internet 14, as well as other devices 12(m') (m'≠m) connected to the ISP 11 or to other ISP's. During a log-on operation, the ISP log-on control 24 receives the Internet protocol ("IP") parameters which will be used in connection with message packet generation during the communications session.

During a communications session, the message packet generator 22, in response to input provided by the operator through the operator interface 20, and/or in response to requests from programs (not separately shown) being processed by the device 12(m), generates message packets for transmission through the network interface 21. The network interface 21 also receives message packets from the ISP 11 and provides them to message packet receiver and processor 23 for processing and provision to the operator interface 20 and/or other programs (not shown) being processed by the device 12(m). If the received message packets contain information, such as Web pages or the like, which is to be displayed to the operator, the information can be provided to the operator interface 20 to enable the information to be displayed on the device's video display unit. In addition or alternatively, the information may be provided to other programs (not shown) being processed by the device 12(m) for processing.

Generally, elements such as the operator interface 20, message packet generator 22, message packet receiver and processor 23, ISP log-on control 24 and Internet parameter store 25 may comprise elements of a conventional Internet browser, such as Mosaic, Netscape Navigator and Microsoft Internet Explorer.

In connection with the invention, as noted above the device 12(m) also includes a secure message packet processor 26. The secure message packet processor 26 facilitates the establishment and use of a "secure tunnel," which will be described below, between the device 12(m) and another device 12 (m') (m'≠m) or 13. Generally, in a secure tunnel, information in at least the data portion of message packets transferred between device 12(m) and a specific other device 12(m') (m'≠m) or 13 is maintained in secret by, for example, encrypting the data portion prior to transmission by the source device. Information in other portions of such message packets may also be maintained in secret, except for the information that is required to facilitate the transfer of the respective message packet between the devices, including, for example, at least the destination information, so as to allow the Internet's switching nodes and ISP's to identify the device that is to receive the message packet.

In addition to ISP 11, a number of other ISP's may connect to the Internet, as represented by arrows 16, facilitating communications between devices which are connected to those other ISP's with other devices over the Internet, which may include the devices 12(n) connected to ISP 11.

The devices 13 which devices 12(m) access and communicate with may also be any of a number of types of devices, including personal computers, computer workstations, and the like, and also including mini-and mainframe computers, mass storage systems, compute servers, local area networks ("LAN's") and wide area networks ("WAN's") including such devices and numerous other types of devices which may be connected directly or indirectly to the networks. In connection with the invention, at least one of the devices will include at least one private network, identified as virtual private network 15, which may be in the form of a LAN or WAN. The virtual private network 15 may comprise any of the devices 12(m') (m'≠m) (thereby connecting to the Internet 14 through an ISP) or 13 (thereby connecting directly to the Internet 14); in the illustrative embodiment described herein, the virtual private network 15 will be assumed to comprise a device 13. The virtual private network 15 itself includes a plurality of devices, identified herein as a firewall 30, a plurality of servers 31(1) through 31(S) (generally identified by reference numeral 31(s)) and a nameserver 32, all interconnected by a communication link 33. The firewall 30 and servers 31(s) may be similar to any of the various types of devices 12(m) and 13 described herein, and thus may include, for example, personal computers, computer workstations, and the like, and also including mini-and mainframe computers, mass storage systems, compute servers, local area networks ("LAN's") and wide area networks ("WAN's") including such devices and numerous other types of devices which may be connected directly or indirectly to the networks.

As noted above, the devices, including devices 12(m) and devices 13, communicate by transferring message packets over the Internet. The devices 12(m) and 13 can transfer information in a "peer-to-peer" manner, in a "client-server" manner, or both. Generally, in a "peer-to-peer" message packet transfer, a device merely transfers information in one or more message packets to another device. On the other hand, in a "client-server" manner, a device, operating as a client, can transfer a message packet to another device, operating as a server to for example, initiate service by the other device. A number of types of such services will be appreciated by those skilled in the art, including, for example, the retrieval of information from the other device, to enable the other device to perform processing operations, and the like. If the server is to provide information to the client, it (that is, the server) may generally be referred to as a storage server. On the other hand, if the server is to perform processing operations at the request of the client, it (that is, the server) may generally be referred to as a compute server. Other types of servers, for performing other types of services and operations at the request of clients, will be appreciated by those skilled in the art.

In a client/server arrangement, device 12(m) requiring service by, for example, a device 13, generates one or more request message packets requesting the required service, for transfer to the device 13. The request message packet includes the Internet address of the device 13 that is, as the destination device, to receive the message packet and perform the service. The device 12(m) transfers the request message packet(s) to the ISP 11. The ISP 11, in turn, will transfer the message packet over the Internet to the device 13. If the device 13 is in the form of a WAN or LAN, the WAN or LAN will receive the message packet(s) and direct it (them) to a specific device connected therein which is to provide the requested service.

In any case, after the device 13 which is to provide the requested service receives the request message packet (s), it will process the request. If the device 12(m) which generated the request message packet(s), or its operator, has the required permissions to request the service from the device 13 which generated the request message packet, if the requested service is to initiate the transfer of information from the device 13 as a storage server to the device 12(m) as client, the device 13 will generate one or more response message packets including the requested information, and transmit the packet(s) over the Internet 14 to the ISP 11. The ISP 11, in turn, will transfer the message packet(s) to the device 12(m). On the other hand, if the requested service is to initiate processing by the device 13 as a compute server, the device 13 will perform the requested computation service(s). In addition, if the device 13 is to return processed data generated during the computations to the device 12(m) as client, the device 13 will generate one or more response message packet(s) including the processed data and transmit the packet(s) over the Internet 14 to the ISP 11. The ISP 11, in turn, will transfer the message packet(s) to the device 12(m). Corresponding operations may be performed by the devices 12(m) and 13, ISP 11 and Internet 14 in connection with other types of services which may be provided by the server devices 13.

As noted above, each message packet that is generated by devices 12(m) and 13 for transmission over the Internet 14 includes a destination address, which the switching nodes use to route the respective message packet to the appropriate destination device. Addresses over the Internet are in the form of an "n"-bit integer (where "n" currently may be thirty two or 128). To relieve, in particular, an operator of a device 12(m) of the necessity of remembering specific integer Internet addresses and providing them to the device 12(m) to initiate generation of a message packet for transmission over the Internet, the Internet provides a second addressing mechanism which is more easily utilized by human operators of the respective devices. In that addressing mechanism, Internet domains, such as LAN's, Internet service providers ("ISP's") and the like which are connected in the Internet, are identified by relatively human-readable names. To accommodate human-readable domain names, ISP 11 is associated with a nameserver 17 (which may also be referred to as a DNS servers), which can resolve the human-readable domain names to provide the appropriate Internet address for the destination referred to in the respective human-readable name. Generally, the nameserver may be part of or connected directly to the ISP 11, as shown in FIG. 1, or it may be a particular device which is accessible through the ISP over the Internet. In any case, as noted above, when the device 12(m) logs on to the ISP 11 during a communications session, the ISP 11 will assign various Internet protocol ("IP") parameters which the device 12(m) is to use during the communications session, which will be stored in the Internet parameter store 25. These IP parameters include such information as

(a) an Internet address for the device 12(m) which will identify the device 12(m) during the communications session, and

(b) the identification of a nameserver 17 that the device 12(m) is to use during the communications session.

The device 12(m), when it generates message packets for transfer, will include its Internet address (item (a) above) as the source address. The device(s)13 which receives the respective message packets can use the source address from message packets received from the device 12(m) in message packets which they (that is, device(s) 13) generate for transmission to the device 12(m), thereby to enable the Internet to route the message packets generated by the respective device 13 to the device 12(m). If the device 12(m) is to access the nameserver 17 over the Internet 14, the nameserver identification provided by the ISP 11 (item (b) above) will be in the form of an integer Internet address which will allow the device 12(m) to generate messages to the nameserver 17 requesting resolution ofhuman-readable Internet addresses into integer Internet addresses. The ISP 11 may also assign other IP parameters to the device 12(m) when it logs on to the ISP 11, including, for example, the identification of a connection to the Internet 14 that is to be used for messages transmitted by the device 12(m), particularly if the ISP 11 has multiple gateways. Generally, the device 12(m) will store the Internet parameters in the Internet parameter store 25 for use during the communications session.

When an operator operating device 12(m) wishes to enable the device 12(m) to transmit a message packet to a device 13, he or she provides the Internet address for the device 13 to the device 12(m), through the operator interface 20, and information, or the identification of information maintained by the device 12(m) that is to be transmitted in the message. The operator interface 20, in turn, will enable the packet generator 22 to the required packets for transmission through the ISP 11 over the Internet 11. If

(i) the operator has provided the integer Internet address, or

(ii) the operator has provided the human-readable Internet address, but the packet generator 22 already has the integer Internet address which corresponds to the human-readable Internet address provided by the operator,

the packet generator 22 may generate the packets directly upon being enabled by the operator interface 20, and provide them to the network interface 21 for transmission to the ISP 11.

However, if the operator has provided the human-readable Internet address for the device 13 to which the packets are to be transferred, and if the packet generator 22 does not already have the corresponding integer Internet address therefor, the packet generator 22 will enable the network address to be obtained from the nameserver 17 identified in the IP parameter store 25. In that operation, the packet generator 22 will initially contact nameserver 17 to attempt to obtain the appropriate integer Internet address from the nameserver 17. In these operations, the device 12(m) will generate appropriate message packets for transmission to the nameserver 17, using the nameserver's integer Internet address as provided by the ISP 11 when it (that is, the device 12(m)) logs on at the beginning of the communications session. In any case, if the nameserver 17 has or can obtain the integer Internet address for the human-readable name, it (that is, the nameserver 17) will provide the integer Internet address to the device 12(m). The integer Internet address will be received by the packet generator 22 through the network interface 21 and packet receiver and processor 23. After the packet generator 22 receives the integer Internet address, it can generate the necessary message packets for transmission to the device 13 through the network interface 21 and ISP 11.

As noted above, one of the devices 13 connected to the Internet 14 is virtual private network 15, the virtual private network 15 including a firewall 30, a plurality of devices identified as servers 31(s), and a nameserver 32 interconnected by a communication link 33. The servers 31(s), firewall 30 and nameserver 32 can, as devices connected in a LAN or WAN, transfer information in the form of message packets thereamong. Since the firewall 30 is connected to the Internet 14 and can receive message packets thereover it has an Internet address. In addition, at least the servers 31(s) which can be accessed over the Internet also have respective Internet addresses, and in that connection the nameserver 32 serves to resolve human-readable Internet addresses for servers 31(s) internal to the virtual private network 15 to respective integer Internet addresses.

Generally, the virtual private network 15 is maintained by a company, governmental agency, organization or the like, which desires to allow the servers 31(s) to access other devices outside of the virtual private network 15 and transfer information thereto over the Internet 14, but which also desires to limit access to the servers 31(s) by devices 12(m) and other devices over the Internet 14 in a controlled manner. The firewall 30 serves to control access by devices external to the virtual private network 15 to servers 31(s) within the virtual private network 15. In that operation, the firewall 30 also connects to the Internet 14, receives message packets therefrom for transfer to a server 31(s). If the message packet indicates that the source of the message packet is requesting access to the particular server 31(s), and if the source is authorized to access the server 31(s), the firewall 30 will forward the message packet over the communication link 33 to the server 31(s). On the other hand if the source is not authorized to access the server 31(s), the firewall 30 will not forward the message packet to the server 31(s), and may, instead, transmit a response message packet to the source device indicating that the source was not authorized to access the server 31(s). The firewall may be similar to other devices 31(s) in the virtual private network 15, with the addition of one or more connections to the Internet, which are generally identified by reference numeral 43.

Communications between devices external to the virtual private network 15, such as device 12(m), and a device, such as a server 31(s), inside the virtual private network 15, may be maintained over a secure tunnel between the firewall 30 and the external device as described above to maintain the information transferred therebetween secret while being transferred over the Internet 14 and through the ISP 11. A secure tunnel between device 12(m) and virtual private network 15 is represented in FIG. 1 by logical connections identified by reference numerals 40, 42, and 44; it will be appreciated that the logical connection 42 comprises one of the logical connections 41 between ISP 11 and Internet 14, and logical connection 44 comprises one of the logical connections 43 between the Internet 14 and the firewall 30.

Establishment of a secure tunnel can be initiated by device 12(m) external to the virtual private network 15. In that operation, the device 12(m), in response to a request from its operator, generates a message packet for transfer through the ISP 11 and Internet 14 to the firewall 30 requesting establishment of a secure tunnel between the device 12(m) and firewall 30. The message packet may be directed to a predetermined integer Internet address associated with the firewall 30 which is reserved for secure tunnel establishment requests, and which is known to and provided to the device 12(m) by the nameserver 17. If the device 12(m) is authorized to access a server 31(s) in the virtual private network 15, the client 12(m) and firewall 30 engage in a dialog, comprising one or more message packets transferred therebetween over the Internet 14. During the dialog, the firewall 30 may provide the device 12(m) with the identification of a decryption algorithm and associated decryption key which the device 12(m) is to use in decrypting the encrypted portions of message packets which the virtual private network transmits to the device 12(m). In addition, the firewall 30 may also provide the device 12(m) with the identification of an encryption algorithm and associated encryption key which the device 12(m) is to use in encrypting the portions of message packets which the device 12(m) transmits to the virtual private network 15 which are to be encrypted; alternatively, the device 12(m) can provide the identification of the encryption algorithm and key that it (that is device 12(m)) will use to the firewall 30 during the dialog. The device 12(m) can store in its IP parameter store 25 information concerning the secure tunnel, including information associating the identification of the firewall 30 and the identifications of the encryption and decryption algorithms and associated keys for message packets to be transferred over the secure tunnel.

Thereafter, the device 12(m) and firewall 30 can transfer message packets over the secure tunnel. The device 12(m), in generating message packets for transfer over the secure tunnel, makes use of the secure packet processor 26 to encrypt the portions of the message packets which are to be encrypted prior to transmission by the network interface 21 to the ISP 11 for transfer over the Internet 14 to the firewall 30, and to decrypt the encrypted portions of the message packets received by the device 12(m) which are encrypted. In particular, after the packet generator 22 generates a message packet for transmission to the firewall 30 over the secure tunnel, it will provide the message packet to the secure packet processor 26. The secure packet processor 26, in turn, encrypts the portions of the message packet that are to be encrypted, using the encryption algorithm and key. After the firewall 30 receives a message packet from the device 12(m) over the secure tunnel, it will decrypt it and, if the intended recipient of the message packet is another device, such as a server 31(s), in the virtual private network 14, it (that is, the firewall 30) will transfer the message packet to that other device over the communication link 33.

For a message packet that is to be transferred by a device, such as a server 31(s), in the virtual private network 15 to the device 12(m) over the secure tunnel, the firewall 30 will receive such to the message packet over the communication link 33 and encrypt the message packet for transfer over the Internet 14 to the ISP 11. The ISP 11, in turn, forwards the message packet to the device 12(m), in particular to its network interface 21. The network interface 21 provides the message packet to the secure packet processor 26, which decrypts the encrypted portions of the message packet, using the decryption algorithm and key.

A problem arises in connection with accesses by a device, such as device 12(m), which is external to the virtual private network 15, and a device, such as a server 31(s), which is external to the firewall, namely, that nameserver 17 is not provided with integer Internet addresses for servers 31(s) and other devices which are in the virtual private network 15, except for integer Internet addresses associated with the firewall 30. Thus, the device 12(m), after the operator has entered the human-readable Internet address, will not be able to obtain the integer Internet address of the server 31(s) which is to be accessed from that nameserver 17.

To accommodate this problem, when the device 12(m) and firewall 30 cooperate to establish a secure tunnel therebetween, in addition to possibly providing the device 12(m) with the identifications of the encryption and decryption algorithms and keys which are to be used in connection with the message packets transferred over the secure tunnel, the firewall 30 also provides the device 12(m) with the identification of a nameserver, such as nameserver 32, in the virtual private network 15 which the device 12(m) can access to obtain the appropriate integer Internet addresses for the human-readable Internet addresses which may be provided by the operator of device 12(m). The identification of nameserver 32 is also stored in the IP parameter store 25, along with the identification of nameserver 17 which was provided by the ISP 11 when the device 12(m) logged on to the ISP 11 at the beginning of a communications session. Thus, when the device 12(m) is to transmit a message packet to a device, such as a server 31(s) in the virtual private network 14 using a human-readable Internet address provided by, for example, an operator, the device 12(m) will initially access the nameserver 17, as described above, to attempt to obtain the integer Internet address associated with the human-readable Internet address. Since nameserver 17 is outside of the virtual private network 15 and will not have the information requested by the device 12(m), it will send a response message packet so indicating. The device 12(m) will thereafter generate a request message packet for transmission to the nameserver 32 through the firewall 30 and over the secure tunnel. If the nameserver 32 has an integer Internet address associated with the human-readable Internet address in the request message packet provided by the device 12(m), it will provide the integer Internet address in a manner that is generally similar to that described above in connection with nameserver 18, except that the integer Internet address will be provided by the nameserver 32 in a message packet directed to the firewall 30, and the firewall 30 will thereafter transmit the message packet over the secure tunnel to the device 12(m). In the message packet transmitted by the firewall 30, it will be appreciated that the integer Internet address in the message packet will be in the data portion of the message packet transferred over the secure tunnel and, accordingly, will be in encrypted form. The message packet will be processed by the device 12(m) in a manner similar to that described above in connection with other message packets received by it over the secure tunnel, that is, the message packet will be decrypted by the secure packet processor 26 prior to being provided to the packet receiver and processor 23 for processing. The integer Internet address for the server 31(s) can be cached in an access control list ("ACL") in the IP parameter store 25, along with the association of the human-readable Internet address thereto, an indication that the server 31(s) associated with that human-readable Internet address is to be accessed through the firewall 30 of the virtual private network 15, and the identifications of the encryption and decryption algorithms and keys to be used for encrypting and decrypting the appropriate portions of the message packets transmitted to server 31(s) and received from server 31(s).

It will be appreciated that, if the nameserver 32, in response to a message packet from the device 12(m) requesting the nameserver 32 to provide an integer Internet address for a human-readable Internet address provided by the device 12(m), if the nameserver 32 does not have an association between the human-readable Internet address and an integer Internet address, the nameserver 32 can provide a response message packet so indicating. If the device 12(m) has identification of other nameservers, such as may be associated with other virtual private networks (not shown), to which it (that is, device 12(m)) may have access, then the device 12(m) can attempt to access the other nameservers in a similar manner as described above. If the device 12(m) is unable to obtain an integer Internet address associated with the human-readable Internet address from any of the nameservers to which it has access, and which generally will be identified in its IP parameter store 25, it will generally be unable to access a device having the human-readable Internet address, and may so notify its operator or program which requested the access.

With this background, operations performed by the device 12(m) and virtual private network 15 in connection with the invention will be described in detail. Generally, operations proceed in two phases. In the first phase, the device 12(m) and virtual private network 15 cooperate to establish a secure tunnel through the Internet 14. In that first phase, the virtual private network 15, in particular the firewall 30 provide the identification of a nameserver 32, and may also provide the encryption and decryption algorithm and key information, as described above. In the second phase, after the secure tunnel has been established, the device 12(m) can use the information provided during the first phase in connection with generating and transferring message packets to one or more servers 31(s) in the virtual private network 15, in the process obtaining resolution human-readable Internet addresses to integer Internet addresses as necessary from the nameserver 32 that was identified by the firewall 30 during the first phase.

Thus, in the first (secure tunnel establishment) phase, the device 12(m) initially generates a message packet requesting establishment of a secure tunnel for transfer to the firewall 30. The message packet will include an integer Internet address for the firewall (which may have been provided by the device's operator or a program being processed by the device 12(m) or have been provided by a the nameserver 17 after a human-readable Internet address was provided by the operator or a program), and which, in particular, is to enable the firewall 30 to establish secure tunnels therewith. If the firewall 30 accepts the secure tunnel establishment request, and if the firewall 30 provides the encryption and decryption algorithms and keys as noted above, it (that is, the firewall) will generate a response message packet for transmission to the device 12(m) that identifies the encryption and decryption algorithms and keys; as noted above, this response message packet will not be encrypted. When the device 12(m) receives the response message, the identifications of the encryption and decryption algorithms and keys will be stored in the IP parameter store 25.

At some point later in the first phase, the firewall 30 will also generate a message packet for transmission to the device 12(m) that includes the integer Internet address of the nameserver 32. For this message packet, the portion of the message packet that contains the integer Internet address of the nameserver 32 will be encrypted, using encryption algorithm and key that can be decrypted using the decryption algorithm and key provided in the response message packet described above. This message will generally have a structure

"<IIA(FW),IIA(DEV12(m))><SEC_TUN>

<ENCR<<IIA(FW),IIA(DEV_12(m))><DNS_ADRS:IIA(NS_32)>>>"

where

(i) "IIA(FW)" represents the source address, that is, integer Internet address of the firewall 30,

(ii) "IIA(DEV_12(m))" represents the destination address, that is, the integer Internet address of the device 12(m),

(iii) "DNS_ADRS:IIA(NS) indicates that "IIA(NS_32)" represents the integer Internet address of the nameserver 32, the nameserver which the device 12(m) is authorized to use, and

(iv) "ENCR<...>" indicates that the information between brackets "<" and >" is encrypted.

The initial portion of the message "<IIA(FW),IIA(DEV_12(m))>" forms at least part of the header portion of the message, and "<ENCR<<IIA(FW),IIA(DEV_12(m))><IIA(NS)>>>" represents at least part of the data portion of the message. The "<SEC_TUN>" represents an indicator in the header indicating that the message is being transferred over the secure tunnel, thereby indicating that the data portion of the message contains encrypted information.

After the device 12(m) receives the message from the firewall 30 as described above, since the message packet contains the <SEC_TUN> indicator, its network interface 21 will transfer the encrypted portion "<ENCR<<IIA(FW),IIA(DEV_12(m))><DNS_ADRS:IIA(NS_32)>>>" to the secure packet processor 26 for processing. The secure packet processor will decrypt the encrypted portion, determine that the portion "IIA(NS_32)" is the integer Internet address of a nameserver, in particular nameserver 32, that the device 12(m) is authorized to use, and store that address in the IP parameter store 25, along with an indication that message packets thereto are to be transferred to the firewall 30 and that data in the message packets is to be encrypted using the encryption algorithm and key previously provided by the firewall 30. It will be appreciated that, since the integer Internet address of nameserver 32 is transferred from the firewall to the device 12(m) in encrypted form, it will be maintained in confidence even if the packet is intercepted by a third party.

Depending on the particular protocol used to establish the secure tunnel, the firewall 30 and device 12(m) may also exchange message packets containing other information than that described above.

As noted above, in the second phase, after the secure tunnel has been established, the device 12(m) can use the information provided during the first phase in connection with generating and transferring message packets to one or more of the servers 31(s) in the virtual private network 15. In those operations, if the operator of device 12(m), or a program being processed by device 12(m), wishes to have device 12(m) transmit a message packet to a server 31(s) in the virtual private network 15, if the operator, through the operator interface 20, or the program provides a human-readable Internet address, the device 12(m), in particular the packet generator 22, will initially determine whether the IP parameter store 25 has cached therein an integer Internet address that is associated with the human-readable Internet address. If not, the packet generator 22 will generate a request message packet for transfer to the nameserver 17 requesting it to provide the integer Internet address associated with the human-readable Internet address. If the nameserver 17 has an integer Internet address associated with the human-readable Internet address, it will provide the integer Internet address to the device 12(m). It will be appreciated that this may occur if the human-readable Internet address in the request message packet has been associated with a device 13 external to the virtual private network 15, as well as with a server 32(s) in the virtual private network 15. Thereafter, the device 12(m) can use the integer Internet address to generate message packets for transfer over the Internet as described above.

Assuming, on the other hand, that the nameserver 17 does not have a integer Internet address associated with the human-readable Internet address, it (that is, the nameserver 17) will provide a response message packet so indicating to the device 12(m). Thereafter, the packet generator 22 of device 12(m) will generate a request message packet for transmission to the next nameserver identified in its IP parameter store 25 requesting that nameserver to provide the integer Internet address associated with the human-readable Internet address. If that next nameserver is nameserver 32, the packet generator 22 will provide the message packet to the secure packet processor 26 for processing. The secure packet processor 26, in turn, will generate a request message packet for transfer over the secure tunnel to the firewall 30. This message will generally have a structure

"<IIA(DEV_12(m)),IIA(FW)><SEC_TUN>

<ENCR<<IIA(DEV_12(m)),IIA(NS_32))><IIA_REQ>>>"

where

(i) "IIA(DEV_12(m))" represents the source address, that is, integer Internet address of the device 12(m)

(ii) "IIA(FW)" represents the destination address, that is, the integer Internet address of the firewall 30

(iii) "IIA(NS_32)" represents the address of the nameserver 32

(iii) "<<IIA(DEV_12(m)),IIA(NS_32))><IIA_REQ>>" represents the request message packet generated by the packet generator 22, where "<IIA(DEV_12(m)),IIA(NS_32)> represents the header portion of the request message packet, and "<IIA_REQ>" represents the data portion of the request message packet,

(iv) "ENCR<....>" indicates that the information between brackets "<" and >" is encrypted, and

(v) "<SEC_TUN>" represents an indicator in the header portion of the message packet generated by the secure packet generator 26 indicating that the message is being transferred over the secure tunnel, thereby indicating that the data portion of the message contains encrypted information.

When the firewall 30 receives the request message packet generated by the secure packet processor 26, it will decrypt the encrypted portion of the message packet to obtain <<IIA(DEV_12(m)),IIA(NS_32))><IIA_REQ>>" represents the request message packet as generated by the packet generator 22. After obtaining the request message packet, the firewall 30 will transmit it over the communication link 33 to the nameserver 32. In that process, depending on the protocol for transmission of message packets over the communication link 33, the firewall 30 may need to modify the request message packet to conform to the protocol of communication link 33.

After the nameserver 32 receives the request message packet, it will process it to determine whether it has an integer Internet address associated with the human-readable Internet address provided in the request message packet. If the nameserver determines that it has such an integer Internet address, it will generate a response message packet including the integer Internet address for transmission to the firewall. Generally, the response message packet will have a structure:

<<IIA(NS_32),IIA(DEV_12(m))><IIA_RESP>>

where

(i) "IIA(NS_32)" represents the source address, that is, integer Internet address of the nameserver 32,

(ii) "IIA(DEV_12(m))" represents the destination address, that is, integer Internet address of the device 12(m), and

(iii) "IIA_RESP" represents the integer Internet address associated with the human-readable Internet address.

After the firewall 30 receives the response message packet, since communications with device 12(m) are over the secure tunnel therebetween, it (that is, the firewall 30) will encrypt the response message packet received from the nameserver 32 and generate a message packet for transmission to the device 12(m) including the encrypted response message packet. Generally, the message packet generated by the firewall 30 has the structure:

"<IIA(FW),IIA(DEV12(m))><SEC_TUN>

<ENCR<<IIA(NS_32),IIA(DEV_12(m))><IIA_RESP>>>"

where

(i) "IIA(FW)" represents the source address, that is, integer Internet address of the firewall 30,

(ii) "IIA(DEV_12(m))" represents the destination address, that is, the integer Internet address of the device 12(m),

(iii) "SEC_TUN" represents an indicator in the header portion of the message packet generated by the secure packet generator 26 indicating that the message is being transferred over the secure tunnel, thereby indicating that the data portion of the message contains encrypted information, and

(iv) "ENCR<....>" indicates that the information between brackets "<" and >" (which constitutes the response message packet received from the nameserver 32) is encrypted.

In addition, depending on the protocol for transmission of message packets over the communication link 33, the firewall 30 may need to process and/or modify the message packet to conform to the protocol of Internet 14.

When the device 12(m) receives the message packet from the firewall 30, it (that is, the message packet) will be provided to the secure packet processor 26. The secure packet processor 26, in turn, will decrypt the encrypted portion of the message packet to obtain the integer Internet address associated with the human-readable Internet address, and load that information in the IP parameter store 25. Thereafter, the device can use that integer Internet address in generating message packets for transmission to the server 31(s) which is associated with the human-readable Internet address.

It will be appreciated that, if the nameserver 32 does not have an integer Internet address associated with the human-readable Internet address provided by the device 12(m) in the request message packet, it (that is, nameserver 32) can so indicate in the response message packet generated thereby. The firewall 30 will, in response to the response message packet provided by the nameserver 32, also generate a message packet for transmission to the device 12(m), the message packet including an encrypted portion comprising the response message packet generated by the nameserver 32. After the device 12(m) receives the message packet, the encrypted portion will be decrypted by the secure packet processor 26, which, in turn, will notify the packet generator 22 that the nameserver 32 does not have an integer Internet address associated with the human-readable Internet address. Thereafter, if the IP parameter store 25 contains the identification of another nameserver, the packet generator 22 of device 12(m) will generate a request message packet for transmission to the next nameserver identified in its IP parameter store 25 requesting that nameserver to provide the integer Internet address associated with the human-readable Internet address. On the other hand, if the IP parameter store 25 does not contain the identification of another nameserver, the packet generator 22 can notify the operator interface 20 or program that it is will be unable to generate a message packet for transmission to a device associated with the human-readable Internet address provided thereby.

An embodiment of the invention can provide a number of advantages. For example, it can provide a system for easing communications between devices connected to a public network such as the Internet 14, and devices connected to private networks such as virtual private network 15, by facilitating resolution of human-readable addresses to network addresses by a nameservers connected to the private networks over a secure tunnel.

It will be appreciated that numerous modifications may be made to the arrangement described above in connection with FIG. 1. For example, although the network 10 has been described such that the identification of the encryption and decryption algorithms and keys are exchanged by the device 12(m) and firewall 30 during the dialog during which the secure tunnel is established, it will be appreciated that that information may be provided by the device 12(m) and firewall 30 separately from the establishment of a secure tunnel therebetween.

In addition, although an embodiment of the invention has been described in connection with the Internet, it will be appreciated that an embodiment of the invention can be used in connection with any network. Further, although an embodiment has been described in connection with a network which provides for human-readable network addresses, it will be appreciated that an embodiment can be used in connection with any network which provides for any form of secondary or informal network address arrangements.

It will be appreciated that a system in accordance with the invention can be constructed in whole or in part from special purpose hardware or a general purpose computer system, or any combination thereof, any portion of which may be controlled by a suitable program. Any program may in whole or in part comprise part of or be stored on the system in a conventional manner, or it may in whole or in part be provided in to the system over a network or other mechanism for transferring information in a conventional manner. Thus, such a computer program can form a product operable, when run on a computer, to provide the required functionality of an embodiment of the invention. The computer program product can be provided on a carrier medium, for example, a computer readable medium such as, for example, a memory, disc or other storage medium, or a transmission medium such as a telecommunications channel providing, for example, electrical, optical, wireless or other transmission. In addition, it will be appreciated that the system may be operated and/or otherwise controlled by means of information provided by an operator using operator input elements (not shown) which may be connected directly to the system or which may transfer the information to the system over a network or other mechanism for transferring information in a conventional manner.

The foregoing description has been limited to a specific embodiment of this invention. It will be apparent, however, that various variations and modifications may be made to the invention, with the attainment of some or all of the advantages of the invention.

CLAIMS

1. A system comprising a virtual private network and an external device which communicate over a digital network,

the virtual private network having a firewall, at least one internal device and a nameserver each having a network address, the internal device also having a secondary address, the nameserver being configured to provide an association between the secondary address and the network address,

the firewall, in response to a request from the external device to establish a connection therebetween, being configured to provide the external device with the network address of the nameserver, and

the external device, in response to a request requesting access to the internal device including the internal device's secondary address, being configured to generate a network address request message for transmission over the connection to the firewall requesting resolution of the network address associated with the secondary address, the firewall being configured to provide the address resolution request to the nameserver, the nameserver being configured to provide the network address associated with the secondary address, the firewall in turn being further configured to provide the network address in a network address response message for transmission over the connection to the external device.

2. A system according to claim 1, wherein the external device is further configured to use the network address provided in the network address response message in generating at least one message for transmission to the internal device.

3. A system according to claim 1 or claim 2, wherein the external device is configured to connect to the network through a network service provider.

4. A system according to claim 3, wherein the external device is configured to establish a communications session with the network service provider, the network service provider providing the external device with the identification of a further nameserver, the further nameserver being configured to provide an association between a secondary address and a network address for at least one device.

5. A system according to any preceding claim, wherein the external device is configured to maintain a list of nameservers which have been identified to said external device, the external device being configured to interrogate successive ones of the nameservers in the list in response to a request requesting access to another device, said request including a secondary address for said other device, until said external device receives a network address, in each interrogation the external device being configured to generate a said network address request message for transmission over the network for response by one of said nameservers in said list and to receive a network address response message therefrom.

6. A system according to any preceding claim, wherein the connection between the external device and the firewall is a secure tunnel, in which at least some portion of messages transferred between the external device and the firewall is encrypted.

7. A method of operating a system comprising a virtual private network and an external device interconnected by a digital network, the virtual private network having a firewall, at least one internal device and a nameserver each having a network address, the internal device also having a

secondary address, the nameserver being configured to provide an association between the secondary address and the network address, the method comprising the steps of:

A. enabling the firewall, in response to a request from the external device to establish a connection therebetween, provide the external device with the network address of the nameserver; and

B. enabling

(i) the external device, in response to a request requesting access to the internal device including the internal device's secondary address, to generate a network address request message for transmission over the connection to the firewall requesting resolution of the network address associated with the secondary address,

(ii) the firewall to provide the address resolution request to the nameserver,

(iii) the nameserver to provide the network address associated with the secondary address, and

(iv) the firewall to provide the network address in a network address response message for transmission over the connection to the external device.

8. A method according to claim 7, wherein the external device is further enabled to use the network address provided in the network address response message in generating at least one message for transmission to the internal device.

9. A method according to claim 7 or claim 8, wherein the external device is enabled to connect to the network through a network service provider.

10. A method according to claim 9, wherein the external device is enabled to establish a communications session with the network service provider, the network service provider being enabled to provide the external device with the identification of a further nameserver, the further nameserver being enabled to provide an association between a secondary address and a network address for at least one device.

11. A method according to any one of claims 7 to 10, wherein the external device is enabled to maintain a list of nameservers which have been identified to said external device, the external device being enabled to interrogate successive ones of the nameservers in the list in response to a request requesting access to another device, said request including a secondary address for said other device, until said external device receives a network address, in each interrogation the external device being enabled to generate a said network address request message for transmission over the network for response by one of said nameservers in said list and to receive a network address response message therefrom.

12. A method according to any one of claims 7 to 10, wherein the connection between the external device and the firewall is a secure tunnel, in which at least some portion of messages transferred between the external device and the firewall is encrypted.

13. A computer program product for use in connection with a virtual private network and an external device interconnected by a digital network, the virtual private network having a firewall, at least one internal device and a nameserver each having a network address, the internal device also having a secondary address, the nameserver being configured to provide an association between the secondary

address and the network address, the computer program product comprising :

A. a nameserver identification code module configured to enable the firewall, in response to a request from the external device to establish a connection therebetween, to provide the external device with the network address of the nameserver,

B. a network address request message generating code module for enabling the external device, in response to a request requesting access to the internal device including the internal device's secondary address, to generate a network address request message for transmission over the connection to the firewall requesting resolution of the network address associated with the secondary address,

C an address resolution request forwarding module for enabling the firewall to provide the address resolution request to the nameserver,

D. a nameserver control module for enabling the nameserver to provide the network address associated with the secondary address, and

E. a network address response message forwarding module for enabling the firewall to provide the network address in a network address response message for transmission over the connection to the external device.

14. A computer program product according to claim 13, further comprising a network address utilization module configured to enable the external device to use the network address provided in the network address response message in generating at least one message for transmission to the internal device.

15. A computer program product according to claim 13 or claim 14, further comprising a network service provider control module for enabling the external device to connect to the network through a network service provider.

16. A computer program product according to claim 15, wherein the network service provider control module includes a communications session establishment module for enabling the external device to a communications session with the network service provider and receive therefrom identification of a further nameserver.

17. A computer program product according to any one of claims 13 to 16, further including nameserver interrogation control module for enabling the external device to maintain a list of nameservers which have been identified to said external device, and to interrogate successive ones of the nameservers in the list in response to a request requesting access to another device, said request including a secondary address for said other device, until said external device receives a network address, in each interrogation the external device being enabled to generate a said network address request message for transmission over the network for response by one of said nameservers in said list and to receive a network address response message therefrom.

18. A computer program product according to any one of claims 13 to 16, wherein the connection between the external device and the firewall is a secure tunnel, in which at least some portion of messages transferred between the external device and the firewall is encrypted.

19. A computer program product according to any one of claims 13 to 18 on a carrier medium.

20. A computer program product according to claim 19, wherein the carrier medium is a computer readable medium.

21. A computer program product according to claim 19, wherein the carrier medium is a transmissions medium.

22. A system substantially as hereinbefore described with reference to the accompanying drawings.

23. A method substantially as hereinbefore described with reference to the accompanying drawings.

24. A computer program product substantially as hereinbefore described with reference to the accompanying drawings.

The Patent Office

34

INVESTOR IN PEOPLE

Application No:	GB 9912200.4	**Examiner:**	Gareth Griffiths
Claims searched:	All	**Date of search:**	7 December 1999

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.Q): H4P (PPA, PPEB, PPEC, PPG)

Int Cl (Ed.6): H04L 12/22, 12/46, 12/66, 29/06

Other: Online Databases: WPI, EPODOC, JAPIO

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
X, P	EP0887979 A2	(SUN MICROSYSTEMS) col.15 line 35 - col.17 line 24	1, 2, 5-8, 11-14, 17-21
A	EP0825748 A2	(AT&T) col.6 line 46 - col.11 line 40	
A, P	WO98/31124 A1	(HANSON) p.5 line 2 - p.6 line 25	

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

An Executive Agency of the Department of Trade and Industry

Electronic Patent Application Fee Transmittal

Application Number:	11840560
Filing Date:	17-Aug-2007
Title of Invention:	AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS USING SECURE DOMAIN NAMES
First Named Inventor/Applicant Name:	Victor Larson
Filer:	Atabak R Royaee/Melissa Molchan
Attorney Docket Number:	077580-0063 (VRNK-1CP3CN2

Filed as Large Entity

Utility under 35 USC 111(a) Filing Fees

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Miscellaneous:				
Submission- Information Disclosure Stmt	1806	1	180	180
	Total in USD ($)			**180**

Electronic Acknowledgement Receipt

EFS ID:	8151925
Application Number:	11840560
International Application Number:	
Confirmation Number:	1537
Title of Invention:	AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS USING SECURE DOMAIN NAMES
First Named Inventor/Applicant Name:	Victor Larson
Customer Number:	23630
Filer:	Atabak R Royaee/Melissa Molchan
Filer Authorized By:	Atabak R Royaee
Attorney Docket Number:	077580-0063 (VRNK-1CP3CN2
Receipt Date:	04-AUG-2010
Filing Date:	17-AUG-2007
Time Stamp:	13:03:15
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	Deposit Account
Payment was successfully received in RAM	$180
RAM confirmation Number	8688
Deposit Account	501133
Authorized User	

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)

1	Information Disclosure Statement (IDS) Filed (SB/08)	0063.pdf	65388 ceecf7884a35a2007848134d6dbf923ca54f35a2	no	3
Warnings:					
Information:					
This is not an USPTO supplied IDS fillable form					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
2	Foreign Reference	EP0838930A2.pdf	1724786 9a383b35683829ae78abb4965b90cde255795d93	no	34
Warnings:					
Information:					
3	Foreign Reference	EP0814589A2.pdf	1094602 10c06cd368d846b9f6c82e5622edd22ebc63e401	no	19
Warnings:					
Information:					
4	Foreign Reference	GB2317792A.pdf	1256657 d50989c41fac545a0929025d919331dfbf7136ef	no	34
Warnings:					
Information:					
5	Foreign Reference	WO9827783A1.pdf	846395 2a2ead44cf92a436d19c46f7f35211b7e6ad33cf	no	23
Warnings:					
Information:					
6	Foreign Reference	WO99011019.pdf	2034462 a88bd9be7182a86a8e75ebf9a5aa6e210b0962a5	no	60
Warnings:					
Information:					
7	Foreign Reference	GB2334181A.pdf	431753 98657594c9b37568cbca8e5569b8b1b6fd6f75a0	no	14
Warnings:					
Information:					
8	Foreign Reference	GB2340702A.pdf	1504772 b9d55f72785502abe4081ceb482776f5d62d0f15	no	36
Warnings:					
Information:					

9	NPL Documents	Baumgartner.pdf	535114 e1cfd368a442fe0e98ec5f0b34dc39d0d51aee53	no	20
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
10	NPL Documents	Chapman.pdf	1713700 39c5c492b168aa3e7de9fca2a4031545bed0e957	no	19
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
11	NPL Documents	Davila.pdf	461212 0300df8d7b65f715e893e7e8d5e7985e93b9ed97	no	18
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
12	NPL Documents	DeRaadt.pdf	333587 fdad8832507203c9875d1e55fdc679bf42cecc48	no	10
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
13	NPL Documents	Eastlake.pdf	1007823 7f990c13c14c9426828dd74c35dd10f320f607b2	no	45
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
14	NPL Documents	Gunter.pdf	330364 b806a4f735e709274fa23d1b659fcf93f2e555a2	no	10
Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
15	NPL Documents	Shimizu.pdf	1284498 e3b8ee1a0847be8b7e6a6bc5791dafc815d3ae15	no	15
Warnings:					

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

16	NPL Documents	Stallings.pdf	1451887 b32ca3a1d283b7ceeadc81e075be896ce51656dd	no	42

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

17	NPL Documents	Takata.pdf	109936 009f8475adb2a0557fdfcff79670b4a3112a0119	no	3

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

18	NPL Documents	Wells.pdf	63145 095ce48eaedeaf4359dd456e88732c1e895de5e3	no	1

Warnings:

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:

19	Fee Worksheet (PTO-875)	fee-info.pdf	30429 44729d361de6618a2b0893ab81c890a3a85c18fe	no	2

Warnings:

Information:

Total Files Size (in bytes):	16280510

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111
If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371
If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office
If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Docket No.: 077580-0063 (VRNK-1CP3CN2) **PATENT**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of Victor Larson et al.	:	Customer Number: 23,630
	:	
	:	Confirmation Number: 1537
	:	
Application No.: 11/840,560	:	Group Art Unit: 2453
	:	
Filed: August 17, 2007	:	Examiner: Lim, Krisna
	:	

For: AGILE NETWORK PROTOCOL FOR SECURE COMMUNICATIONS USING SECURE DOMAIN NAMES

REVISED AMENDMENT "A"

MS Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the Advisory Office action dated July 15, 2010, and the substantive Office action on the merits dated March 19, 2010, please amend the above-identified application and consider the accompanying remarks as follows:

Amendments to the specification begin on page 2 of this paper.

Amendments to the claims begin on page 3 of this paper

Remarks begin on page 10 of this paper.

Amendments to the specification:

Please delete paragraph [0001] in its entirety, and substitute therefor:

[0001] This application claims priority from and is a continuation of a co-pending U.S. application serial number 10/714,849, filed November 18, 2003, now U.S. Patent No. 7,418,504, which is a continuation of U.S. application serial number 09/558,210, filed April 26, 2000, now abandoned, which in turn is a continuation-in-part of previously-filed U.S. application serial number 09/504,783, filed on February 15, 2000, now U.S. Patent No. 6,502,135, issued December 31, 2002, which in turn claims priority from and is a continuation-in-part patent application of previously-filed U.S. application serial number 09/429,643, filed on October 29, 1999, now U.S. Patent No. 7,010,604, issued March 07, 2006. The subject matter of U.S. application serial number 09/429,643, now U.S. Patent No. 7,010,604, which is bodily incorporated herein, derives from provisional U.S. application numbers 60/106,261 (filed October 30, 1998) and 60/137,704 (filed June 7, 1999), both now abandoned. The present application is also related to U.S. application serial number 09/558,209, filed April 26, 2000, now abandoned, and which is incorporated by reference herein.

Amendments to the claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

1.-3. (Cancelled)

4. (New) A system for providing a domain name service for establishing a secure communication link, the system comprising:

a domain name service system configured and arranged to be connected to a communication network, store a plurality of domain names and corresponding network addresses, receive a query for a network address, and indicate in response to the query whether the domain name service system supports establishing a secure communication link.

5. (New) The system of claim 4, wherein at least one of the plurality of domain names comprises a top-level domain name.

6. (New) The system of claim 5, wherein the top-level domain name is a non-standard top-level domain name.

7. (New) The system of claim 6, wherein the non-standard top-level domain name is one of .scom, .sorg, .snet, .sgov, .sedu, .smil and .sint.

8. (New) The system of claim 5, wherein the domain name service system is configured to authenticate the query using a cryptographic technique.

9. (New) The system of claim 4, wherein the communication network includes the Internet.

10. (New) The system of claim 4, wherein the domain name service system comprises an edge router.

11. (New) The system of claim 4, wherein the domain name service system is connectable to a virtual private network through the communication network.

12. (New) The system of claim 11, wherein the virtual private network is one of a plurality of secure communication links in a hierarchy of secure communication links.

13. (New) The system of claim 11, wherein the virtual private network is based on inserting into each data packet communicated over a secure communication link one or more data values that vary according to a pseudo-random sequence.

14. (New) The system of claim 11, wherein the virtual private network is based on a network address hopping regime that is used to pseudorandomly change network addresses in packets transmitted between a first device and a second device.

15. (New) The system of claim 11, wherein the virtual private network is based on comparing a value in each data packet transmitted between a first device and a second device to a moving window of valid values.

16. (New) The system of claim 11, wherein the virtual private network is based on a comparison of a discriminator field in a header of each data packet to a table of valid discriminator fields maintained for a first device.

17. (New) The system of claim 4, wherein the domain name service system is configured to respond to the query for the network address.

18. (New) The system of claim 4, wherein the domain name service system is configured to provide, in response to the query, the network address corresponding to a domain name from the plurality of domain names and the corresponding network addresses.

19. (New) The system of claim 4, wherein the domain name service system is configured to receive the query initiated from a first location, the query requesting the network address associated with a domain name, wherein the domain name service system is configured to provide the network address associated with a second location, and wherein the domain name service system is configured to support establishing a secure communication link between the first location and the second location.

20. (New) The system of claim 4, wherein the domain name service system is

connected to a communication network, stores a plurality of domain names and corresponding network addresses, and comprises an indication that the domain name service system supports establishing a secure communication link.

21. (New) The system of claim 4, wherein at least one of the plurality of domain names is reserved for secure communication links.

22. (New) The system of claim 4, wherein the domain name service system comprises a server.

23. (New) The system of claim 22, wherein the domain name service system further comprises a domain name database, and wherein the domain name database stores the plurality of domain names and the corresponding network addresses.

24. (New) The system of claim 4, wherein the domain name service system comprises a server, wherein the server comprises a domain name database, and wherein the domain name database stores the plurality of domain names and the corresponding network addresses.

25. (New) The system of claim 4, wherein the domain name service system is configured to store the corresponding network addresses for use in establishing secure communication links.

26. (New) The system of claim 4, wherein the domain name service system is configured to authenticate the query for the network address.

27. (New) The system of claim 4, wherein at least one of the plurality of domain names comprises an indication that the domain name service system supports establishing a secure communication link.

28. (New) The system of claim 4, wherein at least one of the plurality of domain names comprises a secure name.

29. (New) The system of claim 4, wherein at least one of the plurality of domain names enables establishment of a secure communication link.

30. (New) The system of claim 4, wherein the domain name service system is configured to enable establishment of a secure communication link between a first location and a second location transparently to a user at the first location.

31. (New) The system of claim 4, wherein the secure communication link uses encryption.

32. (New) The system of claim 4, wherein the secure communication link is capable of supporting a plurality of services.

33. (New) The system of claim 32, wherein the plurality of services comprises a plurality of communication protocols, a plurality of application programs, multiple sessions, or a combination thereof.

34. (New) The system of claim 33, wherein the plurality of application programs comprises items selected from a group consisting of the following: video conferencing, e-mail, a word processing program, and telephony.

35. (New) The system of claim 32, wherein the plurality of services comprises audio, video, or a combination thereof.

36. (New) The system of claim 4, wherein the domain name service system is configured to enable establishment of a secure communication link between a first location and a second location.

37. (New) The system of claim 36, wherein the query is initiated from the first location, wherein the second location comprises a computer, and wherein the network address is an address associated with the computer.

38. (New) The system of claim 4, wherein the domain name service system comprises a domain name database connected to a communication network and storing a plurality of domain names and corresponding network addresses for communication, wherein the domain name database is configured so as to provide a network address corresponding to a domain name in response to a query in order to establish a secure communication link.

39. (New) A machine-readable medium comprising instructions executable in a domain name service system, the instructions comprising code for: connecting the domain name service system to a communication network; storing a plurality of domain names and corresponding network addresses; receiving a query for a network address; and indicating in response to the query whether the domain name service system supports establishing a secure communication link.

40. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for storing the plurality of domain names and corresponding network addresses including at least one top-level domain name.

41. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for responding to the query for the network address.

42. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for providing, in response to the query, the network address corresponding to a domain name from the plurality of domain names and the corresponding network addresses.

43. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for receiving the query for a network address associated with a domain name and initiated from a first location, and providing a network address associated with a second location, and establishing a secure communication link between the first location and the second location.

44. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for indicating that the domain name service system supports the establishment of a secure communication link.

45. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for reserving at least one of the plurality of domain names for secure communication links.

46. (New) The machine-readable medium of claim 39, wherein the code resides on a server.

47. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for storing a plurality of domain names and corresponding network addresses so as to define a domain name database.

48. (New) The machine-readable medium of claim 39, wherein the code resides on a server, and the instructions comprise code for creating a domain name database configured to store the plurality of domain names and the corresponding network addresses.

49. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for storing the corresponding network addresses for use in establishing secure communication links.

50. (New) The machine-readable medium of claim 39, wherein the instructions comprise code for authenticating the query for the network address.

51. (New) The machine-readable medium of claim 39, wherein at least one of the plurality of domain names includes an indication that the domain name service system supports the establishment of a secure communication link.

52. (New) The machine-readable medium of claim 39, wherein at least one of the plurality of domain names includes a secure name.

53. (New) The machine-readable medium of claim 39, wherein at least one of the plurality of domain names is configured so as to enable establishment of a secure communication link.

54. (New) The machine-readable medium of claim 39, wherein the domain name service system is configured to enable establishment of a secure communication link between a first location and a second location transparently to a user at the first location.

55. (New) The machine-readable medium of claim 39, wherein the secure communication link uses encryption.

56. (New) The machine-readable medium of claim 39, wherein the secure communication link is capable of supporting a plurality of services.

57. (New) The machine-readable medium of claim 56, wherein the plurality of services comprises a plurality of communication protocols, a plurality of application programs, multiple sessions, or a combination thereof.

58. (New) The machine-readable medium of claim 57, wherein the plurality of application programs comprises items selected from a group consisting of the following: video conferencing, e-mail, a word processing program, and telephony.

59. (New) The machine-readable medium of claim 56, wherein the plurality of services comprises audio, video, or a combination thereof.

60. (New) The machine-readable medium of claim 39, wherein the domain name service system is configured to enable establishment of a secure communication link between a first location and a second location.

61. (New) The machine-readable medium of claim 60, wherein the instructions include code for receiving a query initiated from the first location, wherein the second location comprises a computer, and wherein the network address is an address associated with the computer.

62. (New) The machine-readable medium of claim 39, wherein the domain name service system comprises a domain name database connected to a communication network and storing a plurality of domain names and corresponding network addresses for communication, wherein the domain name database is configured so as to provide a network address corresponding to a domain name is response to the query in order to establish a secure communication link.

63. (New) A method of providing a domain name service for establishing a secure communication link, the method comprising: connecting a domain name service system to a communication network, ; storing a plurality of domain names and corresponding network addresses; and upon receiving a query for a network address for communication, indicating whether the domain name service system supports establishing a secure communication link.

REMARKS

The response filed 28 June 2010 is being resubmitted with the amendment to the specification, paragraph [0001], marked to indicate the changes made. This response is therefore now compliant with the requirements of 37 C.F.R. §§ 1.121 and 1.4.

Claims 4-63 remain in the application. Claims 1-3 have been cancelled, subject to refilling those claims in a continuation application of the present application. Claims 4-63 have been added by this amendment. The Examiner's attention is directed to the parent application, Applicant's patent, U.S. Patent No. 7,418,504 (the "'504 Patent"). Pending claims 4-63 are similar to claims 1-59 of the '504 Patent, except that they have been modified to add the limitation in independent claim 4 (and similar limitations to claims 39 and 63) that there is an indication in response to a query whether the domain name service system supports establishing a secure communication link. New claim 19 (corresponding to claim 16 of the '504 Patent) has also been amended with minor changes.

In the Official action on the merits of March 19, 2010, the Examiner has objected to the disclosure because the first paragraph of the specification needs to be updated. The Examiner has also indicated that the information disclosure statement filed May 19, 2009 fails to comply with 37 C.F.R. § 1.98(a)(3) because it does not include a concise explanation of the relevance of each of the cited references. Claims 1 has been rejected under 35 U.S.C. §112, second paragraph, as indefinite, while claims 2-3 have been rejected under 35 U.S.C. §112, second paragraph, as being incomplete for omitting essential steps. Claim 2 has also been rejected under 35 U.S.C. §112, first paragraph, because one skilled in the art clearly would not know how to use the claimed invention. Claims 1-3 have been rejected on the ground of non-statutory obviousness-type double patenting as being unpatentable over claim 1 of U.S. Patent No. 7,418,504 (from which the current applications claims priority). Finally claims 1-3 have been rejected under 35 U.S.C. § 102(e) as being anticipated by Shrader (US Patent No 5,864,666). The objections and rejections are traversed and reconsideration is respectfully requested in view of the foregoing amendments and following remarks.

Regarding the Examiner's objection to the specification, the Examiner has requested that applicants indicate the current status of the applications identified in paragraph [0001].

Applicants accordingly have amended paragraph [0001] to comply with the Examiner's request. Accordingly, the objection should be withdrawn.

The Examiner has indicated that the information disclosure statement filed May 19, 2009 fails to comply with 37 C.F.R. § 1.98(a)(3) because it does not include a concise explanation of the relevance. As stated in applicants' document:

> All of the documents herein submitted have been produced by Microsoft Corp. in VirnetX Inc. and Science Applications International Corp. v. Microsoft Corp. civil action currently pending before the U.S. District Court for the Eastern District of Texas. Although the undersigned attorney has not reviewed these documents to assess their materiality, these documents are submitted under the assumption that they may be material to the patentability of the claims pending in this application.

While applicants' regret the large number of documents cited in the litigation by the opposing party, the inability to review the documents, and therefore the inability to assess the materiality of the references, nevertheless applicants felt compelled to cite the references in the file history of the present application, because the two patents in the cited litigation (US Patent Nos. 6,502,135 B1 and 7,188,180 B2) and that of the present application all relate to secure communications. The present application claims priority from a chain of applications, including the one that issued as U.S. Patent No. 6,502,135. U.S. Patent No. 7,188,180 claims priority from a chain of applications, including the same one that issued as U.S. Patent No. 6,502, 135. Accordingly, the applicants request that the Examiner reconsider his position, and review and enter the information disclosure statement of May 19, 2010.

Regarding the rejection of Claims 1 has been rejected under 35 U.S.C. §112, second paragraph, as indefinite, the Examiner states that "it is unclear from where a query is sent", "it is unclear from where the query message is requesting a secure computer network address," "it is unclear how a portal authenticates a query," and finally "the interrelationship or interfunction between a portal and a domain name database is unclear." It is submitted that these cited limitations do not make the claims indefinite. The source of the query message can be from any source in communication with the system for providing a domain name service. As recited in new claim 4, the domain name service system is configured and arranged to be (1) connected to a communication network, (2) store a plurality of domain names and corresponding network addresses, (3) receive a query for a network address, and (4) indicate in response to the query whether the domain name service system supports establishing a secure communication link. All four limitations are required by the claim.

The Examiner also finds fault since there is no express recitation of the interrelationship between the portal and domain database. While the applicants disagree with this rejection, none of the currently presented claims include the limitation of "the portal." Accordingly, this objection is believed to be overcome.

The Examiner also believed that original claim 2-3 were incomplete for "omitting essential structural cooperative relationships of elements, such omission amounting to a gap between the necessary structural connections." Claims 4-63 recite adequate limitations to satisfy the statutory requirements of 35 U.S.C. §112.

Claim 4 recites the restrictive limitations of a domain name service system required to be "configured and arranged to be connected to a communication network, store a plurality of domain names and corresponding network addresses, receive a query for a network address, and indicate in response to the query whether the domain name service system supports establishing a secure communication link."

Claim 39 recites "a machine-readable medium comprising instructions executable in a domain name service system, the instructions comprising code for: connecting the domain name service system to a communication network; storing a plurality of domain names and corresponding network addresses; receiving a query for a network address; and indicating in response to the query whether the domain name service system supports establishing a secure communication link." This clearly provides limitations acceptable under 35 U.S.C. §112.

Finally, claim 63 recites "a method of providing a domain name service for establishing a secure communication link, the method comprising: connecting a domain name service system to a communication network; storing a plurality of domain names and corresponding network addresses; and upon receiving a query for a network address for communication, indicating whether the domain name service system supports establishing a secure communication link." Claim 63 recites method steps that clearly recite method limitations within the requirements of

35 U.S.C. §112.

Finally, original claim 2 was also rejected under 35 U.S.C. §112, first paragraph, is not supported by an "undue breadth … asserted utility or a well established utility." Applicants disagree. New claim 4 recites that the system provides "a domain name service for establishing a secure communication link." New claim 39 recites a "machine-readable medium comprising instructions executable in a domain name service system, the instructions comprising code for connecting the domain name service system to a communication network; storing a plurality of domain names and corresponding network addresses; receiving a query for a network address; and indicating in response to the query whether the domain name service system supports establishing a secure communication link."

Claims 1-3 have been rejected on the ground of non-statutory obviousness-type double patenting as being unpatentable over claim 1 of U.S. Patent No. 7,418,504 (from which the current applications claims priority). Applicants agree to submit a terminal disclaimer should the Examiner maintain this rejection against new claims 4-63.

Finally claims 1-3 have been rejected under 35 U.S.C. § 102(e) as being anticipated by Shrader (US Patent No 5,864,666). The latter patent describes a system for administering tunneling on a firewall computer between a secure computer network and a nonsecure computer network (col. 1, lines 40 and 41). The system includes a user interface. The user interface is presented having a first pane in which a tunnel definition can be entered. A query is run on an entered tunnel definition to determine whether any existing tunnel definitions match the entered tunnel definition. The results of the query are then displayed on a scatter bar in another pane in the user interface. Locations of matching tunnel definitions are then indicated by lines through the scatter bar. A small bar is displayed proximate to the scatter bar. The small bar indicates the position of the displayed list of tunnel definitions relative to a complete list of tunnel definitions represented by the scatter bar. At this point, an action may be performed on a selected definition. (col 1., lines 53-65).

The patent reference also describes the use of an Internet firewall product that allows administrators to create a physical firewall between a secure network and an unsecured network. The firewall product is described as providing a number of functions including "specialized domain name services." (col. 2, lines 11-18). However "IP tunneling is a feature provided by internet firewalls which is the primary subject of the present invention."

The "IP Tunnel Query Page" is shown in detail in Figs. 6 and 7, and is described in detail in col. 7, line 34-col. 10, line 46. Clearly the page is used by a human administrator to determine tunnel definitions.

The reference thus describes a system to allow a human network administrator to administer tunneling on a firewall computer between a secure computer network and a nonsecure computer network. The system includes an interface that provides graphical depictions of tunnels between addresses in the networks as lines connecting icons representing network addresses. The system allows the user to display a selected tunnel definition in response to the user input. The only apparent query that the system responds to is a "query on an entered tunnel definition to determine whether any existing tunnel definitions match the entered tunnel definition." Clearly the system of Shrader does not provide a domain name service system "configured and arranged to be connected to a communication network, store a plurality of domain names and corresponding network addresses, receive a query for a network address, and indicate in response to the query whether the domain name service system supports establishing a secure communication link", as recited in applicants' claim 4, nor a "machine-readable medium comprising instructions executable in a domain name service system, the instructions comprising code for: connecting the domain name service system to a communication network; storing a plurality of domain names and corresponding network addresses; receiving a query for a network address; and indicating in response to the query whether the domain name service system supports establishing a secure communication link," as recited in applicants claim 39, nor a "method of providing a domain name service for establishing a secure communication link, the method comprising: connecting a domain name service system to a communication network, ; storing a plurality of domain names and corresponding network addresses; and upon receiving a query for a network address for communication, indicating whether the domain name service system supports establishing a secure communication link," as recited in applicants' claim 63.

In summary therefore, the remaining claims 4-63, the remaining claims in the application, are believed to be in condition for allowance. An early and favorable action thereon is therefore