Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2545496 A1 2005/05/26

(21) 2 545 496

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2004/11/12
(87) Date publication PCT/PCT Publication Date: 2005/05/26
(85) Entrée phase nationale/National Entry: 2006/05/10
(86) N° demande PCT/PCT Application No.: US 2004/037918
(87) N° publication PCT/PCT Publication No.: 2005/048106
(30) Priorités/Priorities: 2003/11/11 (US60/518,305); 2003/11/24 (US60/524,999)

(51) Cl.Int./Int.Cl. *G06F 11/30* (2006.01), *G06F 12/14* (2006.01)

(71) Demandeur/Applicant:
CITRIX GATEWAYS, INC., US

(72) Inventeurs/Inventors:
RAO, GOUTHMAN P., US;
RODRIGUEZ, ROBERT, US;
BRUEGGEMANN, ERIC, US

(74) Agent: BERESKIN & PARR

(54) Titre : SYSTEME, APPAREIL ET PROCEDE POUR L'ETABLISSEMENT D'UNE LIAISON DE COMMUNICATION SECURISEE POUR LA FORMATION D'UN RESEAU PRIVE VIRTUEL AU NIVEAU D'UNE COUCHE DE PROTOCOLE DE RESEAU AUTRE QUE CELLE A LAQUELLE DES PAQUETS SONT FILTRES

(54) Title: VIRTUAL PRIVATE NETWORK WITH PSEUDO SERVER

(57) Abrégé/Abstract:
A system, apparatus and a method for implementing a secured communications link at a layer other than that at which packets are filtered are disclosed. In one embodiment, a computer system is configured to form a virtual private network ("VPN") and comprises an address inspection driver to identify initial target packet traffic addressed to a target server. Also, the computer system includes a pseudo server module to receive rerouted initial target packet traffic from the address inspection driver. The pseudo server module is configured to convey packet regeneration instructions to a VPN gateway. The address inspection driver functions to identify additional target packet traffic addressed to the target server and routes the additional target packet traffic to the pseudo server. In one embodiment, the pseudo server is configured to strip header information from the additional target packet traffic to form a payload, and thereafter, to route the payload to the target server.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 May 2005 (26.05.2005)

PCT

(10) International Publication Number
WO 2005/048106 A3

(51) International Patent Classification[7]: G06F 11/30, 12/14

(21) International Application Number:
PCT/US2004/037918

(22) International Filing Date:
12 November 2004 (12.11.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/518,305	11 November 2003 (11.11.2003)	US
60/524,999	24 November 2003 (24.11.2003)	US

(71) Applicant *(for all designated States except US)*: **NET6, INC.** [US/US]; 2740 Zanker Road, Suite 201, San Jose, CA 95134 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **RAO, Gouthman, P.** [US/US]; 322 Granville Ct., San Jose, CA 95139 (US). **RODRIGUEZ, Robert** [US/US]; 5647 Wells Court, San Jose, CA 95123 (US). **BRUEGGEMANN, Eric** [US/US]; 5642 Stevens Creek Blvd., No.606, Cupertino, CA 95014 (US).

(74) Agent: **BACKUS, Kenneth, R., Jr.**; Cooley Godward LLP, 3000 El Camino Real, Five Palo Alto Square, Palo, Alto, CA 94306-2155 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(88) Date of publication of the international search report:
23 June 2005

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 2005/048106 A3

(54) Title: VIRTUAL PRIVATE NETWORK WITH PSEUDO SERVER

(57) Abstract: A system, apparatus and a method for implementing a secured communications link at a layer other than that at which packets are filtered are disclosed. In one embodiment, a computer system is configured to form a virtual private network ("VPN") and comprises an address inspection driver to identify initial target packet traffic addressed to a target server. Also, the computer system includes a pseudo server module to receive rerouted initial target packet traffic from the address inspection driver. The pseudo server module is configured to convey packet regeneration instructions to a VPN gateway. The address inspection driver functions to identify additional target packet traffic addressed to the target server and routes the additional target packet traffic to the pseudo server. In one embodiment, the pseudo server is configured to strip header information from the additional target packet traffic to form a payload, and thereafter, to route the payload to the target server.

SYSTEM, APPARATUS AND METHOD FOR ESTABLISHING A SECURED COMMUNICATIONS LINK TO FORM A VIRTUAL PRIVATE NETWORK AT A NETWORK PROTOCOL LAYER OTHER THAN THAT AT WHICH PACKETS ARE FILTERED

FIELD OF THE INVENTION

[0001]The present invention relates generally to secured communication networks. More particularly, the present invention relates to a system, apparatus and a method for establishing a secured communications link between a remote device and a gateway device, whereby at least the remote device (e.g., such as a remote computing device) is configured to capture and redirect packet traffic at that remote device, and to modify the packets for minimizing latency of encrypted packet traffic for real-time applications.

BACKGROUND OF THE INVENTION

[0002]Internet Protocol Security ("IPsec") and Secure Sockets Layer ("SSL") are examples of conventional encryption protocols that are used to establish virtual private networks ("VPNs") over a public communications network, such as the Internet, to ensure that only authorized users can access data in the VPNs. While functional, traditional VPNs implementing these and other conventional encryption protocols have several drawbacks.

[0003]A drawback to implementing IPsec, for example, is that most firewalls cannot effectively route IPsec-encrypted packet traffic with minimal effort, especially those performing network address translation ("NAT"). Although NAT traversal techniques exist to pass IPsec-encrypted packets through NAT firewalls, these techniques limit IPsec-encrypted packets to a couple of ports (e.g., port 80 and 443), thereby forming bottlenecks. Another drawback is that VPNs implementing IPsec require that an address assigned to a remote computing device be visible by a private network to which that remote device is connected, giving rise to a vulnerability to certain breaches in security. For example, a worm infecting a client in the private network can use the visible address of the remote device to propagate itself into a private network including that remote device. At least some of the drawbacks of IPsec-based VPNs are due to performing both packet inspection and encryption at the network layer, such as at the Ethernet frame-level.

[0004]One drawback to implementing SSL, for example, is that this protocol is typically limited to web applications, thereby precluding the use of numerous other applications that are not browser-based. Another drawback is that SSL-based VPNs do support a wide range of routing protocols. Consequently, SSL-based VPNs cannot generally support real-time applications, such as voice over IP, or "VoIP," and peer-to-peer applications. At least some of the drawbacks of SSL-based VPNs are due to performing both packet inspection and encryption at the transport layer (or the applications layer), which limits routing protocols to, for example, User Data Protocol ("UDP") and Transmission Control Protocol ("TCP").

[0005]Thus, there is a need for a system, an apparatus and a method to overcome the drawbacks of the above-mentioned implementations of encryption protocols in VPNs, and in particular, to establish a secured communications link from a remote computing device to a private network by capturing and redirecting packet traffic at the remote device and by modifying the packets to at least minimize the latency of encrypted packet traffic for real-time applications.

SUMMARY OF THE INVENTION

[0006]A system, apparatus and a method for implementing a secured communications link at a layer other than that at which packets are filtered are disclosed. In one embodiment, a computer system is configured to form a virtual private network ("VPN") and comprises an address inspection driver to identify initial target packet traffic addressed to a target server. Also, the computer system includes a pseudo server module to receive rerouted initial target packet traffic from the address inspection driver. The pseudo server module is configured to convey packet regeneration instructions to a VPN gateway. The address inspection driver functions to identify additional target packet traffic addressed to the target server and routes the additional target packet traffic to the pseudo server. In one embodiment, the pseudo server is configured to strip header information from the additional target packet traffic to form a payload, and thereafter, to route the payload to the target server.

[0007]A method is disclosed, according to another embodiment of the present invention, whereby the method secures communications with a remote client computing device by establishing a virtual private network. The method comprises generating packet traffic with a communication application running on a client computing device, identifying

at the client computing device target packet traffic of the packet traffic that is addressed to a target server, forming a secure communications link between a pseudo server module on the computing device and the target server, directing additional packet traffic addressed to the target server to the pseudo server module, sending an acknowledgment to the communication application upon receipt of the additional packet traffic rerouted to the pseudo server module, and routing a payload to the target server.

[0008]In yet another embodiment, a virtual private network comprises a client machine configured as a pseudo server machine with respect to a communication application running on the client machine. The communication application is configured to receive packet traffic acknowledgements from the pseudo server machine. A virtual private network gateway is included and is operative with a server machine to function as a client machine with respect to the pseudo server machine, thereby facilitating secure communications between the client machine and the server machine.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009]A more complete understanding of the present invention is apparent and more readily appreciated by reference to the following Detailed Description and to the appended claims when taken in conjunction with the accompanying Drawings wherein:

[0010]FIG. 1 is a diagram illustrating a virtual private network ("VPN") system for establishing a secured communications link between a remote computing device and a VPN gateway computing device, according to one embodiment of the present invention;

[0011]FIG. 2 is a flow diagram depicting an exemplary method of communicating packets over a secured communications link, according to one embodiment of the present invention;

[0012]FIG. 3 is a block diagram for describing a remote client computing device in accordance with one embodiment of the present invention;

[0013]FIG. 4 is a functional block diagram illustrating the interaction between a pseudo server and an address inspection driver when transmitting target packet traffic from a remote client to a private network, according to a specific embodiment of the present invention;

[0014]FIG. 5 is a functional block diagram illustrating the interaction between a pseudo server and an address inspection driver after receipt of encrypted packet traffic into

a remote client from a private network, according to a specific embodiment of the present invention; and

[0015]FIG. 6 is a block diagram illustrating various modules of a pseudo server for communicating encrypted packets for real-time and other applications, according to various embodiments of the present invention.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0016]FIG. 1 is a diagram illustrating a virtual private network ("VPN") system for establishing a secured communications link between a remote computing device and a VPN gateway computing device, according to one embodiment of the present invention. A virtual private network 100 includes a remote client computing device ("client") 110 coupled via a secured communications link ("secured comm. link") 190 to private network 150 for exchanging encrypted data. Remote client computing device 110 is configured to capture and reroute packet traffic associated with one or more virtual private networks private networks ("VPNs") at or near the network layer (i.e., Layer 2 of the Open System Interconnection model, or "OSI" model). By capturing and inspecting packets at the network layer, remote client computing device 110 is able to inspect a wide range of network traffic including, for example, Internet Protocol ("IP"), TCP, UDP, Internet Control Message Protocol ("ICMP"), Generic Routing Encapsulation ("GRE") techniques, Apple talk, Netbios, etc. Further, remote client computing device 110 can generate secured communications link 190 (or "tunnel") at or near the transport layer (i.e., Layer 4), thereby permitting encrypted packets to pass through network address translation ("NAT")-based firewalls and network devices. In at least one embodiment, private network 150 assigns an address to remote client computing device 110 that can be concealed from computing devices (e.g., target server 154) in that private network, thereby reducing exposure of remote client computing device 110 to security threats, such as worms. In a specific embodiment, remote client computing device 110 is configured to modify packets by, for example, stripping header information prior to transport via secured communications link 190, thereby minimizing latency of encrypted packet traffic in real-time applications.

[0017]Although not shown, remote client computing device 110 includes a processor and a memory for executing and storing programs instructions, respectively, for running various user-level computer software applications (e.g., Microsoft Outlook®). Remote client computing device 110 includes a communication application 112 as an

intermediary to exchange data between the computer software applications to private network 150. Examples of communication application 112 are telnet, File Transfer Protocol ("FTP"), Simple Mail Transfer Protocol ("SMTP"), Hypertext Transfer Protocol ("HTTP") and the like.

[0018]Also, remote client computing device 110 includes a tunnel generator 116 configured to generate at least one end of secured communications link 190. Tunnel generator 116 includes an address inspection driver ("AID") 122, a pseudo server ("PS") 120 and an encryptor 124, each of which is composed of either hardware or software, or both. Address inspection driver ("AID") 122 is disposed at or near the network layer to capture and inspect packet traffic, such as network (e.g., Ethernet) frames, traversing the one or more network adapters of remote client computing device 110. During inspection of, for example, the IP headers of captured packets, address inspection driver 122 determines whether the captured packets are destined for private network 150. If a packet is not bound for private network 150, then address inspection driver 122 forwards the packet as an unencrypted packet via path 114 out into Internet 102.

[0019]But when packet traffic is identified as being destined to private network 150 (i.e., "target packet traffic"), address inspection driver 122 filters that packet traffic from passing out onto path 114. Address inspection driver 122 reconfigures the filtered packets (i.e., the target packet traffic) as "incoming packets" to reroute them to a traffic port on pseudo server 120. In some embodiments, that traffic port can be a "well known port" on remote client computing device 110, where a well known port can be any of port numbers 0 to 1024, or the like. In addition, address inspection driver 122 is configured to also send control information encapsulated as control packets along with the rerouted filtered packets to pseudo server 120. Note that it is not necessary to generate a control packet for every rerouted filtered packet as pseudo server 120 can detect other packets to which the same control information will be applicable. While address inspection driver 122 can be implemented in accordance with the Network Driver Interface Specification ("NDIS"), it can also be implemented in program instructions operable with any known operating system, such as UNIX®, Linux, Microsoft Windows™ and the like.

[0020]Pseudo server ("PS") 120 is disposed at or near the transport layer to receive encrypted packet traffic from secured communications link 190 and to transmit (i.e., redirect) encrypted packet traffic that is rerouted from address inspection driver 122. In some embodiments, pseudo server 120 is configured to modify packets by, for example,

stripping header information prior to transport via secured communications link 190. In operation, pseudo server 120 monitors (or "listens" to) its traffic ports waiting to accept incoming rerouted packets and any control packets that get passed from address inspection driver 122. Pseudo server 120 associates the control packets with respective rerouted packets, and then creates a message frame 132 for transmission to private network 150. Message frame 132 includes, among other things, regeneration instructions for reconstructing the packets at private network 150. Note that message frame 132 is generally then encrypted and sent over secured communications link 190 to private network 150.

[0021]Note that when pseudo server 120 receives encrypted packet traffic from secured communications link 190 rather than transmitting it, pseudo server 120 provides for the decryption of those packets by passing them to encryptor 124. Then, pseudo server 120 passes the decrypted packets to address inspection driver 122, along with control information, if any. In response, address inspection driver 122 reconfigures those decrypted packets signals as "incoming packets" to reroute them to communication application 112.

[0022]In at least one embodiment, pseudo server 120 is configured to modify outgoing packets to form modified packets. In this example, pseudo server 120 can strip header information from the outgoing packets bound for private network 150. Examples of header information that can be stripped include TCP headers, IP headers, link layer headers, and the like. The residual data of the packets from which the header information is stripped is referred to as "modified packets," each including a payload. A modified packet is depicted in FIG. 1 as "payload" 138. Further, message frame 132 includes regeneration instructions to reconstruct the stripped header information for regenerating the pre-modified packets in private network 150. In some cases, message frame 132 can include authentication information. Once message frame 132 is understood by at least one entity of private network 150, a link acknowledgment ("ACK") 134 is returned to tunnel generator 116. In a specific embodiment of the present invention, pseudo server 120 forms modified packets as pseudo-UDP packets, which constitute additional traffic 136 composed of modified packets 138 to be conveyed to private network 150. As such, tunnel generator 116 generates an acknowledgement 130 when sending modified packet 138 to prevent delays associated with acknowledgements required by TCP standards. Acknowledgement 130 can be implemented as a "false acknowledgement" so that remote client computing device need not wait for an acknowledgment (e.g., a TCP acknowledgement) when sending a modified packet 138. Accordingly, modified packets 138 are TCP packets that can behave as UDP

packets. As such, secured communications link 190 can be referred to as a "virtual TCP connection" rather than a standard TCP connection as the packets traversing link 190 are UDP packets masquerading as TCP packets. In one embodiment, tunnel generator 116 determines that traffic target packets includes a certain type of data, such as video or audio data, that is time sensitive (i.e., part of a real-time application) and selectively modifies those traffic target packets to form modified packets 138.

[0023]Encryptor 124 is configured to establish a connection with private network 150 and to encrypt and decrypt packets exiting and entering, respectively, remote client computing device 110. For example, encryptor 124 can establish a connection using Hyper Text Transfer Protocol over Secure Socket Layer ("HTTPS"), Proxy HTTPS and like connection protocols. With these connection protocols being operative generally at a transportation layer (or a higher layer), encryptor 124 establishes a connection that is suitable for traversing NAT-based firewalls and bridges. Once a connection (e.g., HTTPS) is established, the payload of the packets bound for private network 150 is encrypted using, for example, Secure Socket Layer ("SSL"), Transport Layer Security ("TLS") Protocol, or the like. Encryptor 124 can encrypt an entire packet including header information, such as an IP header, if not stripped.

[0024]Private network 150 includes a VPN Gateway 152 and a target server 154, which represents any computing device (as either a server or a client) with which remote client computing device 110 establishes communications. VPN Gateway 152 is an intermediary computing device that coordinates establishment of secured communications link 190 with remote client computing device 110. VPN Gateway 152 exchanges communications between remote client computing device 110 and target server 154. Further, VPN Gateway 152 is similar, at least in some respects, to remote client computing device 110. Namely, VPN Gateway 152 includes a processor, a memory and an encryptor, all of which are not shown, as well as address inspection driver ("AID") 122 and a pseudo server ("PS") 120. AID 122 and PS 120 have similar functionality and/or structure as those described in relation to remote client computing device 110.

[0025]VPN Gateway 152 also includes a tunnel manager ("Tunnel Mgr") 160 and an address translator ("Addr Trans") 162. Tunnel manager 160 is configured to download as a software program at least pseudo server 120 and address inspection driver 122. Also, tunnel manager 160 is configured to provide configuration information. The configuration information can include a range of addresses that are associated with private network 150 so

that remote client computing device 110 can select which packets to filter out as target packet traffic. Further, tunnel manager 160 is also configured to receive message frame 132 and to regenerate packets to, for example, include IP header information and/or the assigned address of remote client computing device 110.

[0026]Address translator 162 is configured to provide a NAT process, and specifically, a reverse NAT process to hide the assigned address of remote client computing device 110 from target server 154. To illustrate, consider the following example in which a TCP connection is created from remote client computing device 110 to target server 154, which has a destination address of 192.168.1.100. First, a TCP SYN packet is generated for address 192.168.1.100. Tunnel generator 116 passes this SYN packet over secured communications link 190. VPN Gateway 152 examines the packet as it arrives and determines that it is a SYN packet for 192.168.1.100. In turn, VPN Gateway 152 generates a new SYN (i.e., replays or regenerates that packet) destined for 192.168.1.100, with a source address appearing to indicated that the new SYN packet originated from 192.168.1.2, which is the private address for VPN Gateway 152. After target server 154 at address 192.168.1.100 generates a SYN-ACK packet, VPN Gateway 152 then receives this packet. Then, a new SYN-ACK packet is in turn conveyed over secured communications link 190 back to tunnel generator 116, which then generates a SYN-ACK packet. This packet appears to originate from target server 154 at address 192.168.1.100, as viewed by remote client computing device 110. In short, VPN Gateway 152 is able to reverse map reply packets and acknowledgments or any other packet as part of that protocol by using unique source port numbers of VPN Gateway 152. In this manner, remote client computing device 110 is able to connect to any foreign private network and still maintain IP invisibility. Such invisibility can be on an application-by-application basis. In some cases, VPN Gateway 152 can optionally enable address visibility by sending an assigned private address for a successfully established secured communications link to a tunnel generator, which in turn, assigns that private address to the remote client computing device in which it resides. But note that the visibility of the address of remote client computing device is not mandatory, but can be optionally enabled, for example, to facilitate certain applications, such as voice applications or any other peer-to-peer applications.

[0027]In a specific embodiment, remote client computing device 110 can establish another secured communications link 192 (which is similar to that of link 190) to another private network ("n") 198 simultaneous to the pendency of secured communications link

190. As such, remote client computing device 110 can simultaneously establish multiple VPN tunnels or secured communications links to different private subnets or networks, especially in cases where destination network addresses overlap partially or completely. Note that while Internet 102 is exemplified as a communications network through which secured communications link 190 can be established in accordance with an embodiment of the present invention, remote client computing device 110 can employ tunnel generator 116 to form tunnels to any type of communications networks, such as a wireless network. It will also be understood that the embodiments of the present invention may be implemented using any routing protocol (e.g., Internet Protocol version 6, "IPv6"), in any packet switching technology (e.g., an Ethernet network), over any communications media (e.g., Ethernet cabling, wireless, optical fibers, etc.) and for use with any computing device (e.g., a wireless station) as an end station, without deviating from the scope and the spirit of the present invention.

[0028]FIG. 2 is a flow diagram 200 depicting an exemplary method of communicating packets over a secured communications link, according to one embodiment of the present invention. At 202, a communications application running on a remote client computing device, such a telnet application, generates packet traffic in response to a request by a user-level application to access a private network. The client computing device at 204 identifies target packet traffic bound for a target server. At 206, the target packet traffic is rerouted to a pseudo server module to at least convey packet regeneration instructions to, for example, a VPN Gateway. The client computing device, receives a link acknowledgement sent from the VPN Gateway at 208, thereby signaling, for example, that a secured communications link between the client and the private network is operational. In turn, the link acknowledgment is conveyed at 210 to the communications application to initiate packet transfer. At 212, additional packet traffic addressed to the target server can be directed to the pseudo server module from, for example, an address inspection driver. Thereafter, at 214, an acknowledgement can be sent to the communications application upon receipt of the additional packet traffic at the pseudo server module prior to transmission to the target server, according to at least one embodiment of the present invention. In some embodiments, header information is stripped form the additional packet traffic to form a payload at 216. Then, the payload at 218 is routed to the VPN Gateway.

[0029]FIG. 3 is a block diagram for describing a remote client computing device in accordance with one embodiment of the present invention. Computing device 302 in this

example is capable of exchanging encrypted packet traffic 390 with another computing device located, for example, on a private network via a secured communications link 380. In the example shown in FIG. 3, computing device 302 includes an operating system 304 coupled to a network interface card ("NIC") 324, which can be, for example, an Ethernet network adapter. Operating system 304 also includes a protocol stack 310, which can be any set of network protocols for binding high-level protocol layers, such as an application layer, to lower-level protocol layers, such as a physical layer including NIC 324. As shown, protocol stack includes a pseudo server 317, an address inspection driver 323 and an encryption protocol 310 in accordance with a specific embodiment of the present invention.

[0030]Protocol stack 310 is shown to include at least a transport layer, a network layer and a link layer. The transport layer includes at least one transport protocol, such as a UDP process 312, a TCP process 314 (i.e., a TCP service) or an optional another type of transport protocol, "other transport" protocol 316, such as "ICMP." FIG. 3 shows that the network layer, which includes an IP process 318 (i.e., an IPv4 or IPv6 service), can be at a next higher layer over the link layer. In this example, pseudo server 317 is disposed at the transport layer and address inspection driver 323 is disposed near the network layer. In particular, address inspection driver 323 is disposed at the data link layer. An encryption protocol 310, such as SSL, can be disposed along side or above pseudo server 317, and is suitable to implement encryptor 124 of FIG. 1. In FIG. 3, encryption protocol 310 is in a layer above TCP process 314.

[0031]According to one embodiment of the present invention, protocol stack 310 is a collection of processes embodied in software. In another embodiment, protocol stack 310 and/or its constituents can be embodied in software or hardware, or both. Each process (e.g., TCP 314, IP 318, etc.) of protocol stack 310 is configured to communicate with each other process, such as across layers of protocol stack 310. A higher-level layer, such as the transport layer, can be configured to communicate, for example, via a Winsock API 308 (or any other socket layer program to establish, for example, raw sockets) with an application 306. Winsock API 308 provides an interface with application 306, which can be a telnet application. A lower-level layer, such as either a network layer or a link layer, can be configured to communicate, for example, via a MAC driver 322 with NIC 324. Exemplary interactions between pseudo server 317 and address inspection driver 323 to establish a secured communications link are described next in FIGs. 4 and 5.

[0032]FIG. 4 is a functional block diagram 400 illustrating the interaction between a pseudo server and an address inspection driver when transmitting target packet traffic from a remote client to a private network, according to a specific embodiment of the present invention. In this example, encryptor 124 and pseudo server 120 are disposed at transport layer 404 and address inspection driver ("AI Drvr") 122 is disposed at network layer 408. Again, note that in some embodiments address inspection driver resides at the data link layer. Pseudo server 120 is coupled to a port forwarding mapping data structure 440 that maintains packet information, such as a "key," a source address ("SA"), a source port ("SP"), a destination address ("DA"), and a destination port ("DP"). Similarly, address inspection driver 122 maintains a similar data structure depicted as a driver mapping table data structure 422. Further, address inspection driver 122 is also coupled to a filter table 420 that includes configuration information provided by a VPN Gateway. Filter table 420 includes network addresses, such as a source address and a destination address (e.g., 198.0.0.80), an optional subnet mask (not shown), a protocol, such as TCP, UDP, ICMP, etc. (not shown), port information, such as a source port and a destination port, and a unique mapping key to uniquely identify destination information associated with target packet traffic. Pseudo server 120 and address inspection driver 122 synchronize these data structures by exchanging control information, such as in a control packet 434, when a change is made to one of those data structures. An exemplary control packet 434 can be a UDP packet or a packet of any other protocol, and is typically sent with rerouted data packets to pseudo server 120. If some of the control information includes updates to entry 442, such as a change in destination port, than that change is entered. In some cases, the control information includes how the packet should be handled or regenerated at the VPN Gateway.

[0033]Consider that application 112 resides on a remote client computing device and is identifiable by a source address of 10.0.02 and a source port of 8678, and a target server (not shown) resides at destination address 198.0.0.80 and destination port 445. If address inspection driver 122 has yet to detect the destination address or port in packet traffic 462, then a destination address and a destination port for that target server is stored in driver mapping table data structure 422. In this case, an entry 424 is made in data structure 422 to include a source address ("SA") as 10.0.0.2, a source port ("SP") as 8678, a destination address ("DA") as 198.0.0.80, and a destination port ("DP") as 445, as well as a "key" that is generated and assigned to the packet traffic by address inspection driver 122. Note that

entry 426 signifies that application 112 has formed another secured communications link to another VPN Gateway and that address inspection driver 112 is configured to inspect packet traffic relating to both entries 424 and 426. As such, multiple VPNs can be established concurrently with application 112.

[0034]Next, consider that application 112 is generating target packet traffic 464 that is destined for destination address 198.0.0.80 and destination port 445. This target packet traffic 464 passes through a socket layer 402 to pseudo server 120. Socket layer 402 can include a Winsock API, a Winsock provider or any other socket connection provider process (e.g., a programming interface that provides for raw sockets), regardless of operating system. Pseudo server 120 matches entries of data structure 440 to information in target packet traffic 464 to determine whether that packet traffic is part of a VPN. Since an entry in data structure 440 includes a DA and a DP that respectively correspond to 198.0.0.80 and 445, a match is made and pseudo server 120 concludes that packet traffic 464 is to be routed via a secured communications link. Pseudo server 120 then passes target packet traffic 466 to address inspection driver 122, whereby target packet traffic 466 is characterized by source address ("SA") 450, source port ("SP") 452, destination address ("DA") 454 and destination port ("DP") 456. Note that FIG. 4 shows packet 466 and other packets with select address and port information; other packet data, including payload, is omitted for discussion purposes.

[0035]Address inspection driver 122 then reconfigures target packet traffic 466 and reroutes it back to pseudo server 120 as rerouted packet 432. In at least one embodiment, address inspection driver 122 reconfigures SP 452 to include a "key," which in this example, is "54321." Also, DA and DP are respectively reconfigured to include a local host or a local machine ("LM") address 454 and a traffic port ("TP") 456. In a specific embodiment, local machine address 454 is 127.0.0.1, which is a loop back address causing rerouted packet 432 to be sent up the OSI protocol stack. Address inspection driver 122 sends rerouted packet 432 up to traffic port ("TP") 430 of pseudo server 120, where TP 430 is a listening port for detecting, for example, TCP packets. In some embodiments, rerouted packet 432 is sent to a TCP traffic port of pseudo server 120 regardless of whether rerouted packet 432 is a UDP packet, such as in the case where pseudo server 120 generates a pseudo-UDP packet as a modified packet. Concurrently (or nearly so), control packet 434 includes a local machine address (not shown) so that it can be sent up the OSI protocol stack to a control port (not shown) of pseudo server 120. In such a case, control packet 434

includes information describing the modifications to a packet to form rerouted packet 432. Thereafter, pseudo server 120 then redirects rerouted packet 432 to encryptor 124 to form an encrypted packet 468 that is passed through a secured communications link.

[0036]FIG. 5 is a functional block diagram 500 illustrating the interaction between a pseudo server and an address inspection driver after the receipt of encrypted packet traffic into a remote client from a private network, according to a specific embodiment of the present invention. To illustrate the interaction, consider that an encrypted packet 502 is passed through to encryptor 124 for decryption. Then, the decrypted packet is passed to pseudo server 120, which matches at least some of the contents of the decrypted packet against data in data structure ("port fwd table") 440. Consider that a match is made, thereby signifying that the decrypted packet is part of an established VPN. As such, pseudo server 120 provides decrypted packet 504 and an attendant control packet 505, which in this case includes the key associated with packet 504, to a well known port ("WKP") 506 of address inspection driver 122. Thereafter, address inspection driver 122 reconfigures decrypted packet 504 in accordance with information indexed by the key into driver table 422, which is similar to the data structure of FIG. 4. As such, the reconfigured packet will include destination information that identifies application 112. As such, rerouted packet 432 is signaled as an "incoming" (or received) packet 520 and is passed up the protocol stack to application 112.

[0037]FIG. 6 is a block diagram 600 illustrating various modules of a pseudo server to modify packets for real-time applications, according to at least one embodiment of the present invention. As shown, pseudo server 604 includes a flag-UDP-as-TCP module 605, a packet modifier module 607, and an acknowledgement generator ("ack gen") 609, one or more of which can be simultaneously operative when sending packets through a secure communications link of the present invention. Although real-time packet traffic, such as voice and video, benefits from the performance advantages of a session-less protocol, such as UDP, standard UDP packets are generally difficult to traverse through many firewalls, whereas TCP traffic is not so disadvantaged. In at least one specific embodiment of the present invention, pseudo server 604 is configured to form "pseudo-UDP" packets using modified TCP packets.

[0038]Flag-UDP-as-TCP module 605 is configured to flag a UDP packet as a TCP packet in the IP header, which fools the communications network into thinking that the packets are part of a TCP session. Packet modifier 607 is configured to operate with raw

socket connection process 603 of socket layer 602. In particular, packet modifier 607 strips header information, such as IP header information, and sends the remaining payload via raw socket connections formed by raw socket connection process 603. As such, regeneration instructions are also sent to describe how to reconstruct packets after those packets pass through a secured communications link with header information stripped out. In one embodiment, the regeneration instructions include information for regenerating header information at the target server so that the target packet traffic can be converted from a first format to a second format. In cases where the first format is associated with Transmission Control Protocol ("TCP") and the a second format is associated with User Data Protocol ("UDP"), then the first packet is formatted as a pseudo-UDP (e.g., a UDP packet flagged as a TCP packet), and the second packet is formatted as UDP packet for transmission of, for example, real-time applications.

[0039]Acknowledgement generator ("ack gen") 609 is configured to issue "false acknowledgments" in response to TCP representations of UDP packets (i.e., pseudo-UDP packets) being transmitted over the secure communications link. This allows for UDP-like behavior to TCP traffic, in that if the TCP packet (i.e., the pseudo-UDP packet) was lost, no attempt is made by either the transmitting end or the receiving end of the VPN tunnel to synchronize sequence numbers and retransmit that packet. Consequently, the VPN interprets the forwarding of pseudo-UDP packets as the forwarding TCP packets, but with raw sockets on either end of the secured communications link interpreting whether these packets are UDP packets carrying voice (such as RTP) or video.

[0040]Various structures and methods for establishing a secured communications link, such as with a pseudo server and an address inspection driver, are described herein. The methods can be governed by or include software processes, for example, as part of a software program. In one embodiment, a pseudo server module and an address inspection driver module are disposed in a software program embedded in a computer readable medium that contains instructions for execution on a computer to implement a secured communications link, according to the present invention.

[0041]An embodiment of the present invention relates to a computer storage product with a computer-readable medium having computer code thereon for performing various computer-implemented operations. The media and computer code may be those specially designed and constructed for the purposes of the present invention, or they may be of the kind well known and available to those having skill in the computer software arts.

Examples of computer-readable media include, but are not limited to: magnetic media such as hard disks, floppy disks, and magnetic tape; optical media such as CD-ROMs and holographic devices; magneto-optical media such as floptical disks; and hardware devices that are specially configured to store and execute program code, such as application-specific integrated circuits ("ASICs"), programmable logic devices ("PLDs") and ROM and RAM devices. Examples of computer code include machine code, such as produced by a compiler, and files containing higher-level code that are executed by a computer using an interpreter. For example, an embodiment of the invention may be implemented using Java, C++, or other programming language and development tools. Another embodiment of the invention may be implemented in hardwired circuitry in place of, or in combination with, machine-executable software instructions.

[0042]The foregoing description, for purposes of explanation, used specific nomenclature to provide a thorough understanding of the invention. However, it will be apparent to one skilled in the art that nomenclature selected herein is presented to teach certain aspects of the present invention and is not intended to restrict the implementations of the various embodiments. Thus, the foregoing descriptions of specific embodiments of the invention are presented for purposes of illustration and description. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed; obviously, many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, they thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated. It is intended that the following claims and their equivalents define the scope of the invention.

IN THE CLAIMS:

1. A method for securing communications with a remote client computing device by establishing a virtual private network, comprising:

generating packet traffic with a communication application running on a client computing device;

identifying at said client computing device target packet traffic of said packet traffic that is addressed to a target server;

forming a secure communications link between a pseudo server module on said computing device and said target server;

directing additional packet traffic addressed to said target server to said pseudo server module;

sending an acknowledgment to said communication application upon receipt of said additional packet traffic rerouted to said pseudo server module; and

routing a payload to said target server.

2. The method of claim 1 wherein identifying at said client computing device said target packet traffic comprises:

inspecting said packet traffic at an address inspection driver;

matching information of said packet traffic to an address representing said target server; and

filtering a subset of said packet traffic bound for said address representing said target server as said traffic packet traffic to be rerouted to said pseudo server module.

3. The method of claim 1 wherein forming said secure communications link between said pseudo server and said target server comprises:

rerouting said target packet traffic to said pseudo server module on said client computing device, said pseudo server module conveying packet regeneration instructions to said target server;

receiving a link acknowledgment from said target server in response to receipt of said packet regeneration instructions at said target server; and

conveying said link acknowledgment to said communication application.

4. The method of claim 3 wherein conveying said packet regeneration instructions comprises including information for regenerating header information at said target server.

5. The method of claim 4 wherein including information further comprises including conversion information from converting said target packet traffic from a first format to a second format.

6. The method of claim 5 wherein including conversion information includes information for converting said first format associated with the Transmission Control Protocol ("TCP") to a second format associated with the User Data Protocol ("UDP").

7. The method of claim 1 wherein routing said payload to said target server comprises stripping header information from said additional packet traffic to form a payload.

8. The method of claim 1 wherein said acknowledgement is a false acknowledgment.

9. A computer system for forming a virtual private network, comprising:

an address inspection driver to identify initial target packet traffic addressed to a target server; and

a pseudo server module to receive rerouted initial target packet traffic from said address inspection driver, said pseudo server module conveying packet regeneration instructions to said target server;

wherein said address inspection driver identifies additional target packet traffic addressed to said target server and routes said additional target packet traffic to said pseudo server; and

wherein said pseudo server strips header information from said additional target packet traffic to form a payload and thereafter routes said payload to said target server.

10. The computer system of claim 9 further comprising a driver mapping data structure configured to include source information and destination information against which said address inspection driver compares packet information from said initial packet traffic.

11. The computer system of claim 9 wherein said address inspection driver is configured to filter said additional target packet traffic from passing unencrypted to said target server when at least a portion of said packet information matches at least a portion of said destination information.

12. The computer system of claim 9 wherein said address inspection driver is configured to generate a control packet that is rerouted to said pseudo server in association with said initial target packet traffic.

13. The computer system of claim 12 wherein said control packet includes source and destination information of said initial target packet traffic for detecting packet traffic originating at said target server.

14. The computer system of claim 9 wherein said packet regeneration instructions are configured to direct said target server to regenerate said target packet traffic to form regenerated packet traffic including regenerated header information and said payload.

15. The computer system of claim 9 further comprising a protocol stack in which said address inspection driver resides at or near the network layer and said pseudo server module resides at or near said transportation layer, said network layer and transportation layer being layers in accordance with the Open System Interconnection model.

16. The computer system of claim 15 wherein said address inspection driver inspects and filters packets at or near said network layer, thereby supporting any routing protocol with which to establish a secured communications link in said virtual private network.

17. The computer system of claim 17 wherein said pseudo server passes encrypted packets onto a secured communications link originating at or near said transportation layer, thereby enabling said encrypted packets to pass through network address translation ("NAT")-enabled network devices.

18. A virtual private network, comprising:

a client machine configured as a pseudo server machine with respect to a communication application running on said client machine, such that said communication application receives packet traffic acknowledgements from said pseudo server machine; and

a virtual private network gateway operative with a server machine to function as a client machine with respect to said pseudo server machine to facilitate secure communications between said client machine and said server machine.

19. The virtual private network of claim 17 wherein said virtual private network gateway selectably conceals from said server machine an address of said client machine running said communication application.

20. The virtual private network of claim 17 wherein said pseudo server machine includes an acknowledgement generator module for generating said packet traffic acknowledgements as false acknowledgments.

21. The virtual private network of claim 17 wherein said pseudo server machine includes a packet modifier module configured to modify packets representing said secure communications to form modified packets by stripping header information therefrom.

22. The virtual private network of claim 21 wherein said pseudo server machine is configured to form a raw socket at a socket layer with which to pass said secure communications.

23. The virtual private network of claim 17 wherein said pseudo server machine includes a flag-UDP-as-TCP module configured to modify a flag in a UDP packet to indicate said UDP packet is a TCP packet, thereby forming a pseudo-UDP packet.

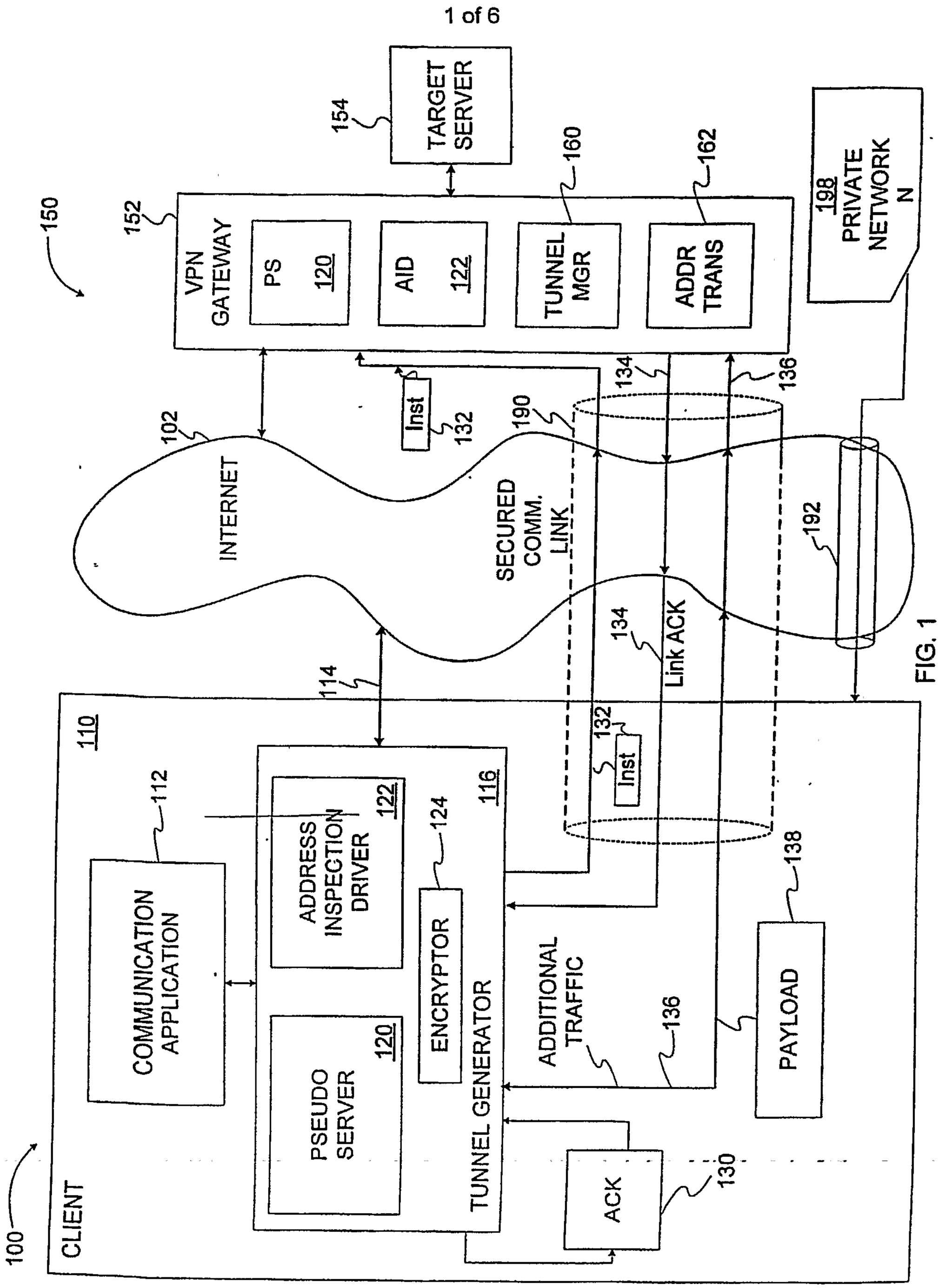
150
152
VPN GATEWAY
PS 120
AID 122
TUNNEL MGR
ADDR TRANS
154
TARGET SERVER
198
PRIVATE NETWORK N
Inst
132
102
INTERNET
190
SECURED COMM. LINK
134
136
192
114
Link ACK
110
COMMUNICATION APPLICATION
112
ADDRESS INSPECTION DRIVER
122
ENCRYPTOR
124
116
TUNNEL GENERATOR
ADDITIONAL TRAFFIC
PAYLOAD
138
PSEUDO SERVER
120
ACK
130
CLIENT
100

FIG. 1

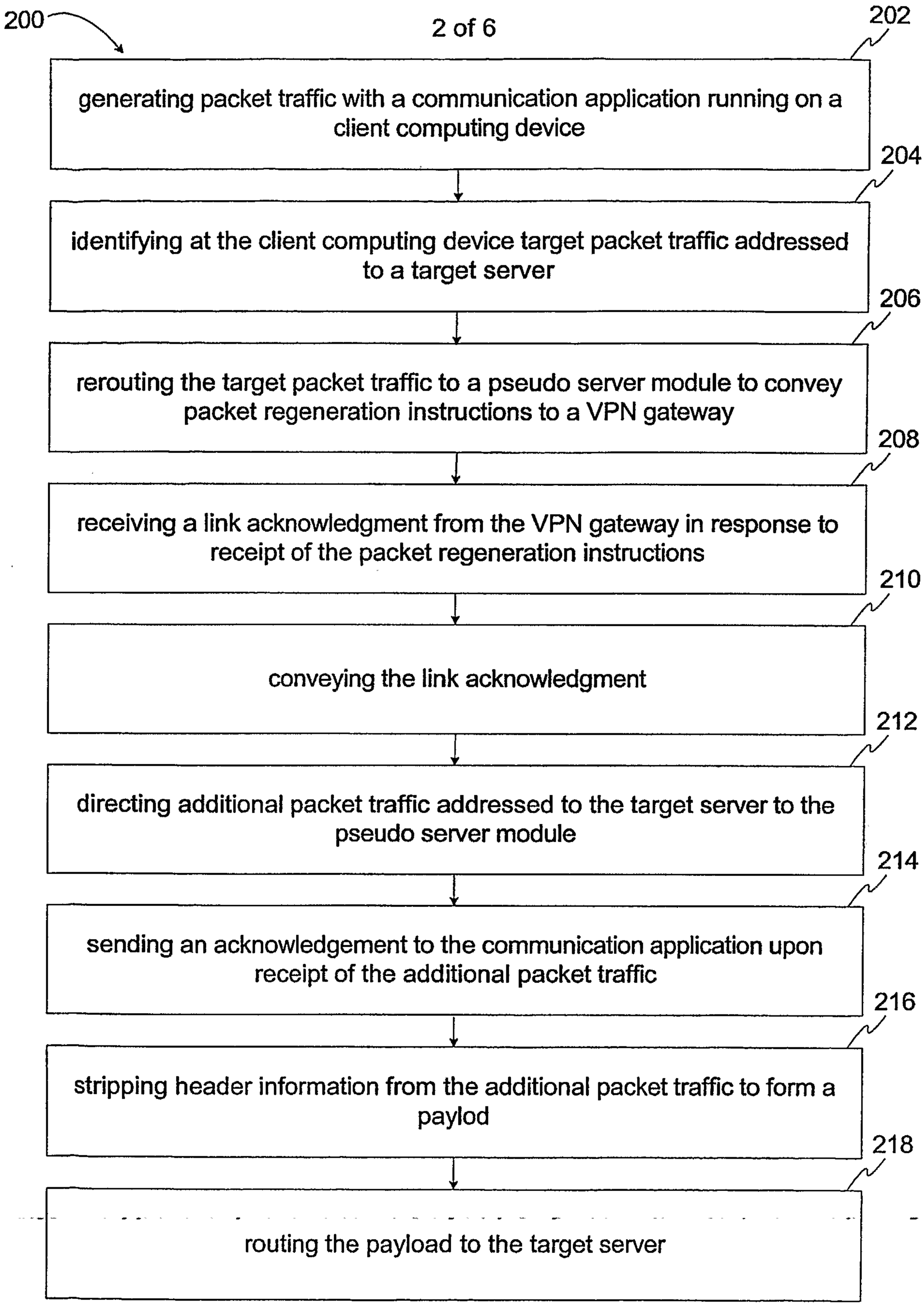
200
202
generating packet traffic with a communication application running on a client computing device
204
identifying at the client computing device target packet traffic addressed to a target server
206
rerouting the target packet traffic to a pseudo server module to convey packet regeneration instructions to a VPN gateway
208
receiving a link acknowledgment from the VPN gateway in response to receipt of the packet regeneration instructions
210
conveying the link acknowledgment
212
directing additional packet traffic addressed to the target server to the pseudo server module
214
sending an acknowledgement to the communication application upon receipt of the additional packet traffic
216
stripping header information from the additional packet traffic to form a paylod
218
routing the payload to the target server

FIG. 2

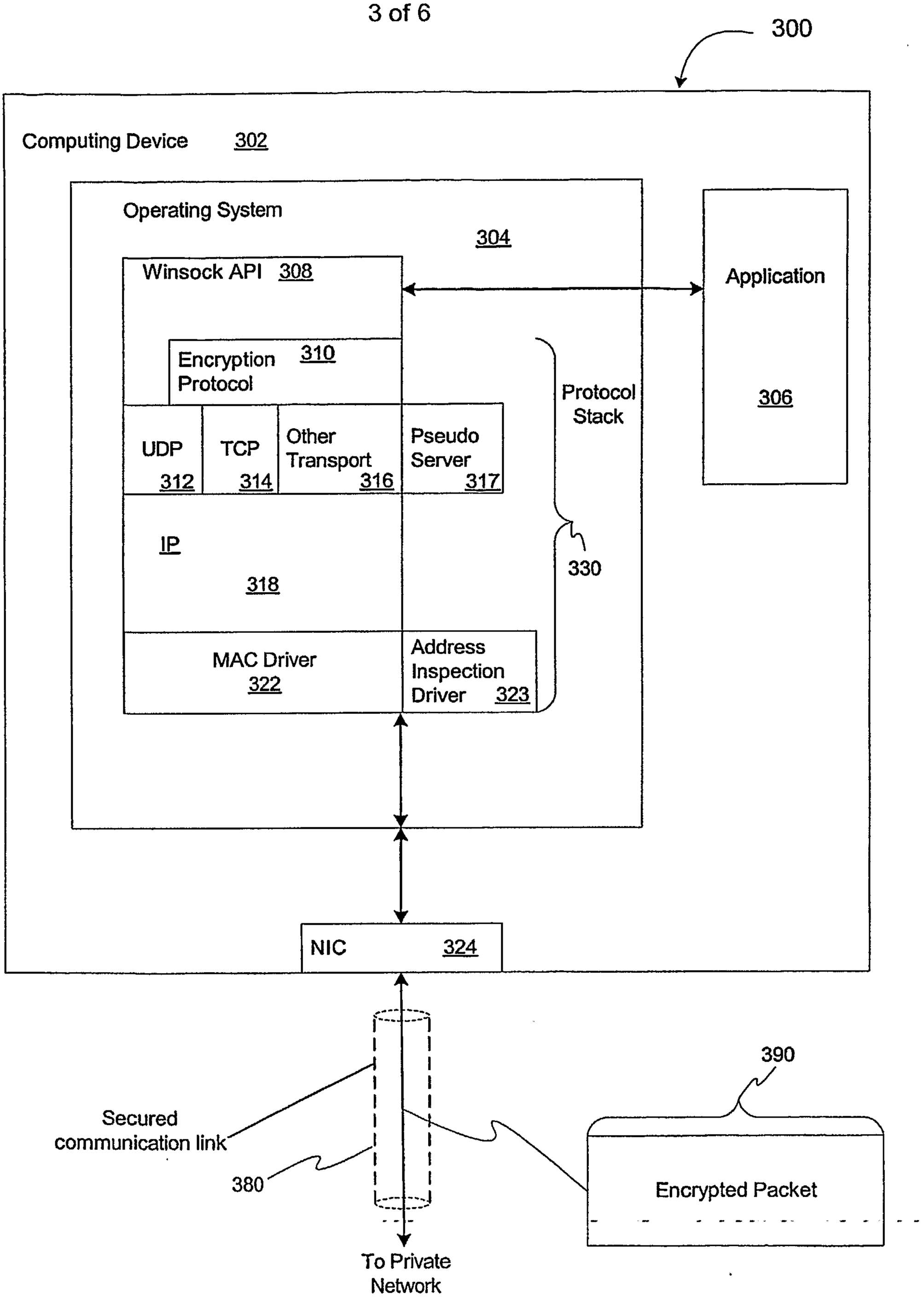
300
Computing Device 302
Operating System
304
Winsock API 308
Encryption Protocol 310
UDP 312
TCP 314
Other Transport 316
Pseudo Server 317
IP 318
MAC Driver 322
Address Inspection Driver 323
Protocol Stack
330
Application 306
NIC 324
Secured communication link
380
To Private Network
390
Encrypted Packet

FIG. 3

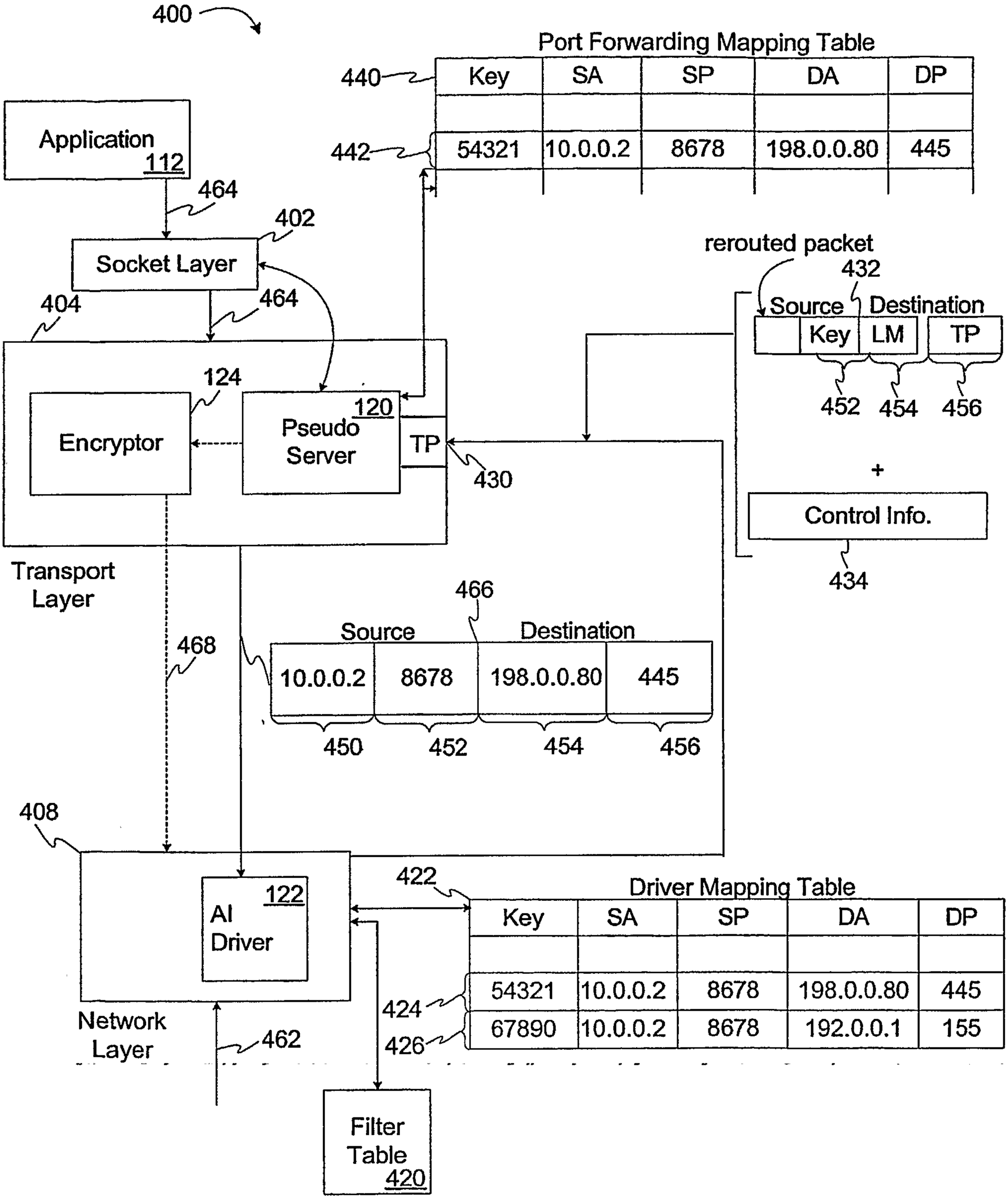
400
Port Forwarding Mapping Table
440
Key
SA
SP
DA
DP
442
54321
10.0.0.2
8678
198.0.0.80
445
Application
112
464
402
Socket Layer
404
464
124
120
Encryptor
Pseudo
Server
TP
430
rerouted packet
432
Source
Destination
Key
LM
TP
452
454
456
+
Control Info.
434
Transport
Layer
466
Source
Destination
468
10.0.0.2
8678
198.0.0.80
445
450
452
454
456
408
422
Driver Mapping Table
122
AI
Driver
Key
SA
SP
DA
DP
54321
10.0.0.2
8678
198.0.0.80
445
67890
10.0.0.2
8678
192.0.0.1
155
424
426
Network
Layer
462
Filter
Table
420

FIG. 4

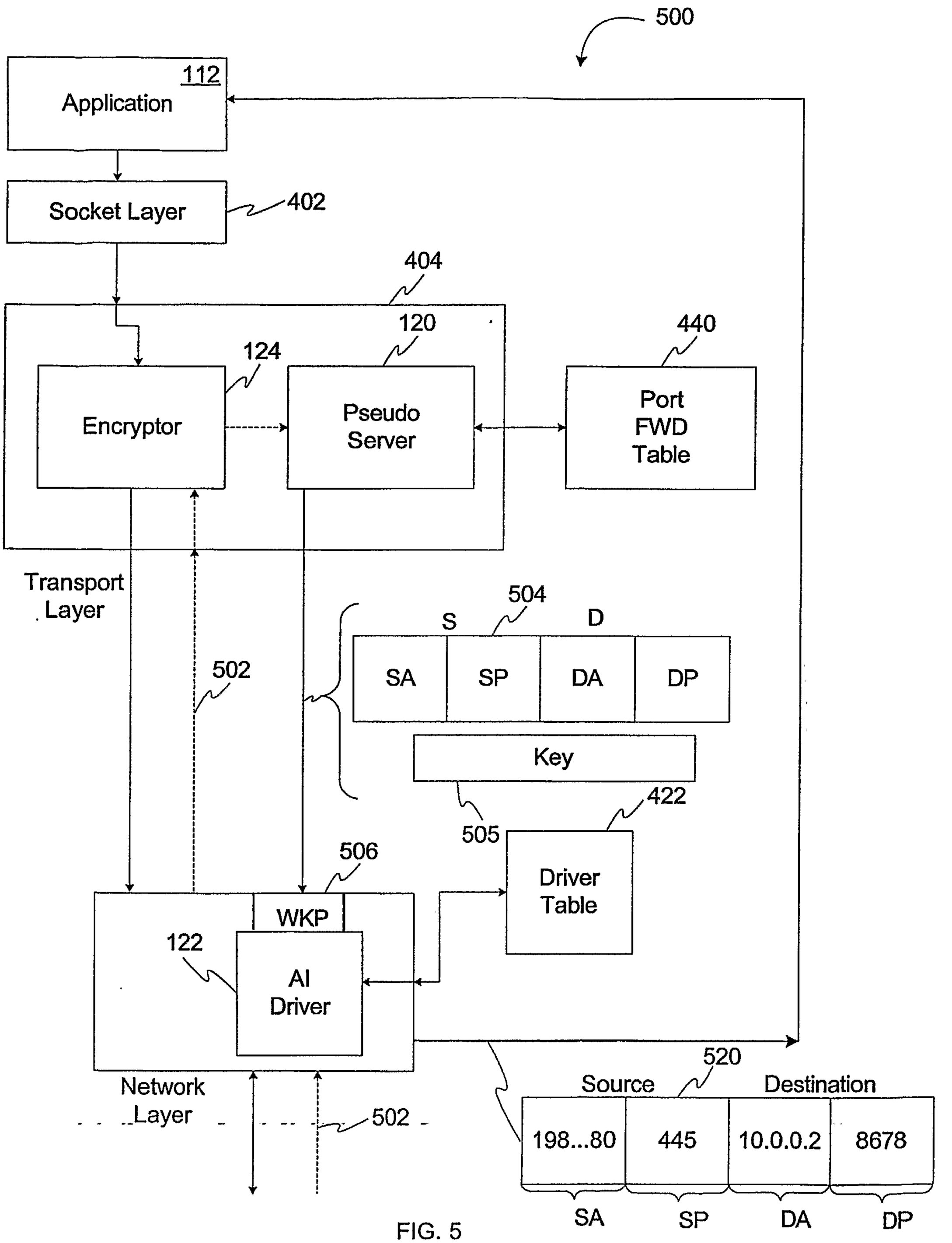
500
112
Application
Socket Layer
402
404
120
124
440
Encryptor
Pseudo Server
Port FWD Table
Transport Layer
504
S
D
SA
SP
DA
DP
502
Key
505
422
Driver Table
506
WKP
122
AI Driver
Network Layer
502
520
Source
Destination
198...80
445
10.0.0.2
8678
SA
SP
DA
DP

FIG. 5

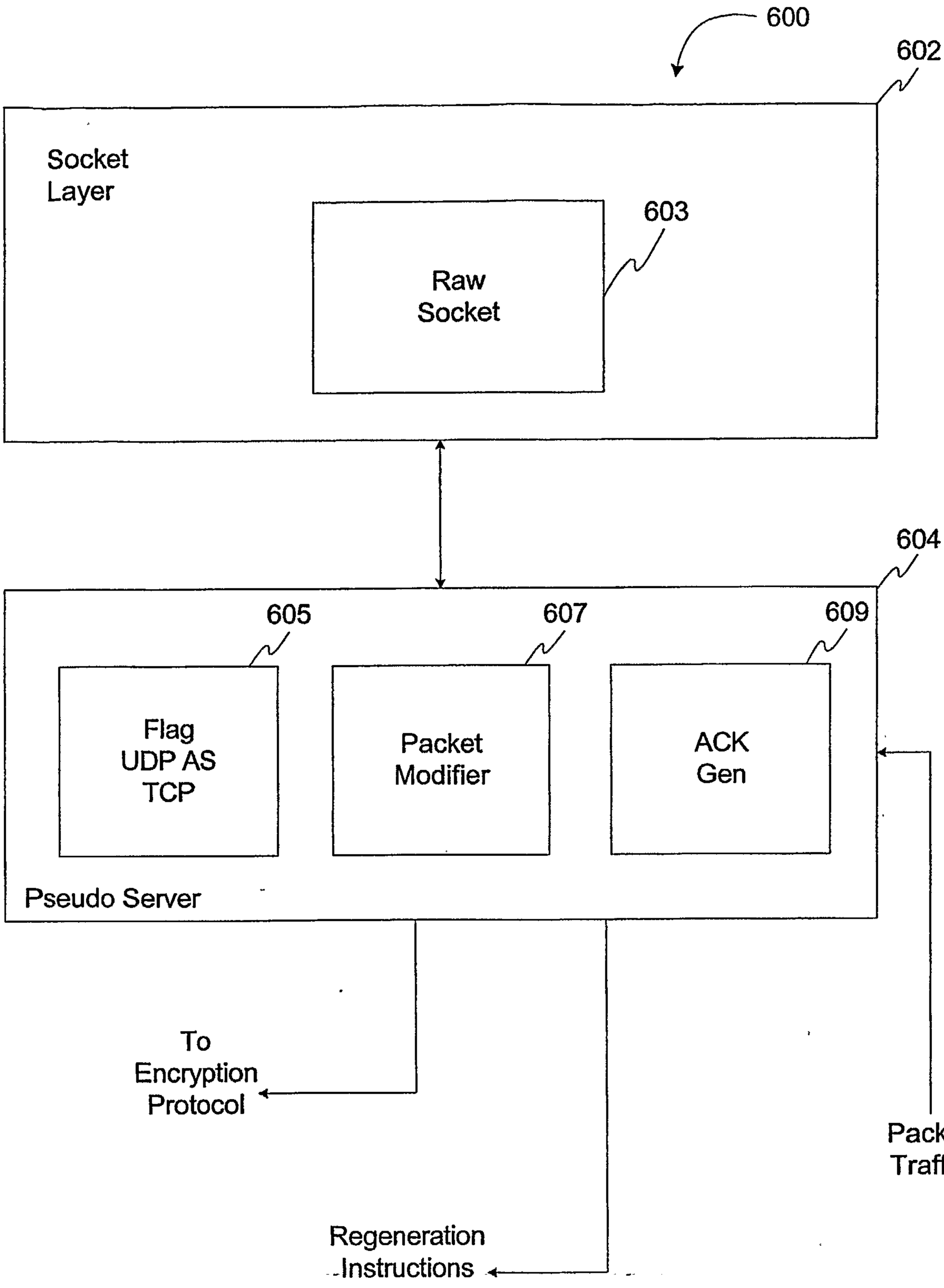
600
602
Socket Layer
603
Raw Socket
604
605
607
609
Flag UDP AS TCP
Packet Modifier
ACK Gen
Pseudo Server
To Encryption Protocol
Packet Traffic
Regeneration Instructions

FIG. 6