

(43) International Publication Date
5 January 2006 (05.01.2006)

PCT

(10) International Publication Number
WO 2006/000802 A2

(51) International Patent Classification:

H04L 29/08 (2006.01) **H04L 29/06** (2006.01)
H04L 29/12 (2006.01)

(21) International Application Number:

PCT/GB2005/002509

(22) International Filing Date: 28 June 2005 (28.06.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

0414415.0 28 June 2004 (28.06.2004) GB

(71) Applicant (for all designated States except US): **AMTEUS LTD** [GB/GB]; Concept House, 57 Cardigan Lane, Leeds, West Yorkshire LS4 2LE (GB).

(72) Inventors; and

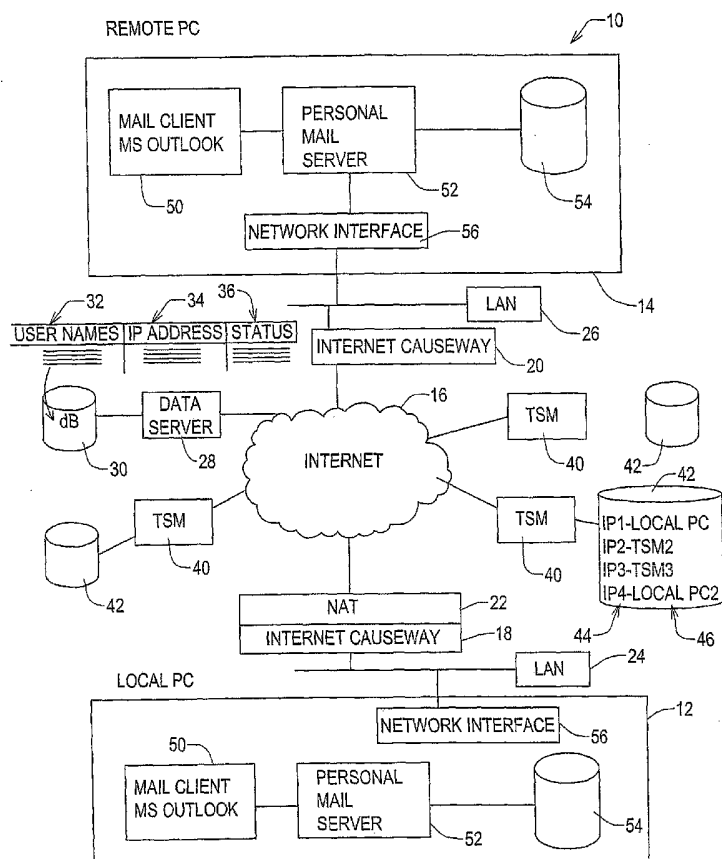
(75) Inventors/Applicants (for US only): **MORRIS, Jeffrey** [GB/GB]; Amteus Ltd., Concept House, 57 Cardigan Lane, Leeds, West Yorkshire LS4 2LE (GB). **FOOT, Eric** [GB/GB]; Amteus Ltd, Concept House, 57 Cardigan Lane,Leeds, West Yorkshire LS4 2LE (GB). **BARR, Robert** [GB/GB]; Amteus Ltd, Concept House, 57 Cardigan Lane, Leeds, West Yorkshire LS4 2LE (GB). **WARBURTON, Randal** [GB/GB]; Amteus Ltd, Concept House, 57 Cardigan Lane, Leeds, West Yorkshire LS4 2LE (GB).(74) Agents: **AHMAD, Sheikh, Shakeel** et al.; David Keltie Associates, Fleet Place House, 2 Fleet Place, London EC4M 7ET (GB).

(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, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, 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,

[Continued on next page])

(54) Title: IMPROVEMENTS RELATING TO SECURE TELECOMMUNICATIONS



(57) Abstract: A method of carrying out a secure peer-to-peer data communication, such as an e-mail communication, between a first remote party computer and a second remote party computer over a data communications medium is described. The method comprises: receiving the address details and current status of a connection to the data communications medium of each remote party computer; creating the data communication at the first remote party computer; checking the current connection status of the second remote party computer; and sending the data communication from the first remote party computer directly to the second remote party computer without any storage of the data communication en route, only when the connection status of the second remote party computer indicates that it is currently connected to the data communication medium.



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, LT, 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:

— *without international search report and to be republished upon receipt of that report*

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.

Improvements Relating to Secure TelecommunicationsField of the Invention

The present invention concerns improvements relating to secure telecommunications and more particularly, though not exclusively, to secure peer-to-peer e-mail and data communications as well as Voice over IP. A further aspect of the present invention is directed towards a method of supporting direct peer-to-peer communications even when Network Address Translators (NATs) are provided to protect a communications network and amplify its IP addressing range. The present invention also extends to effecting the very high levels of security in such communications systems and networks using such communications systems.

Background to the Invention

There are many different ways of transporting messages over a communications network when indirect communications are desired. However there are inherent problems associated with this form of messaging in that it is inherently insecure with copies of the messages being stored and available for hackers to read. Some of these problems which are described in WO 03/014955 A1 are addressed by the use of peer-to-peer communications. However, the main issue with the solutions proposed and used before is that of the complication of setting up such communications. The solution presented by this prior art document is inherently insecure and cumbersome in that it still requires the action of a central server to facilitate the communications. This makes the whole process laborious and open to security abuse.

Encryption techniques are known to improve the security of e-mails which are stored during the communications process. However, no encryption technique is completely secure and so with the sophistication of hacking techniques, this does not in itself present a viable alternative.

NATs are devices that are used to allow multiple computers to share an address space for

communications, such as connecting to the Internet. They have a disadvantage which effects peer-to-peer communications in that such communications do not normally work behind a NAT without complex set ups, which are well beyond the capabilities of the average PC user. The problems associated with NATs in the context of peer-to-peer communications are described in US 2004/0064584 A1.

Summary of the Present Invention

It is desired to improve on the above-described systems and more particularly there is a need for a more secure system. It is also desired to provide a system which is easier to manage and simpler for the user to use. In particular, a system for communicating which does not require the user to log onto a central server for facilitating the peer-to-peer communications or one, which involves even temporary storage of the message at a node during a peer-to-peer communications. Therefore, a 'pure' peer-to-peer communication system is ideally required.

One aspect of the present invention resides in the appreciation that the most secure peer-to-peer e-mail communications, for example, would not require any storage of the message in an insecure environment, namely at any place in the communications route other than the source or destination. In this way, there is only very small chance of a hacker reading the e-mail, namely as it is being transmitted. By encrypting the e-mail for transmission say by using PKI (Public Key Infrastructure) one can minimise the risk of hacking during transmission.

More specifically, according to one aspect of the present invention, there is provided a method of carrying out a secure peer-to-peer data communication, such as an e-mail communication, between a first remote party computer and a second remote party computer over a data communications medium, the method comprising: receiving the address details and current status of a connection to the data communications medium of each remote party computer; creating the data communication at the first remote party computer; checking the current connection status of the second remote party computer; and sending the data

communication from the first remote party computer directly to the second remote party computer without any storage of the data communication en route, only when the connection status of the second remote party computer indicates that it is currently connected to the data communication medium.

5

It is to be appreciated that the term 'storage' as used herein is a reference to storage of the complete message. Such storage would provide an opportunity for an unauthorised third party to obtain illegally a complete copy of the message. Communication protocols often include some temporary storage of part of a message, for example one which is broken
10 down into packets. Such temporary and partial storage of part of the message is permissible with the present aspect of the present invention as it does not permit a third party to obtain a complete copy of the message being transmitted.

Thus, the present invention provides a significant improvement over the prior art in that
15 there is no insecure storage of the complete data message en-route from the source to the destination. Rather, even if the destination is not on-line, the message is always stored securely and temporarily at the source until such time as the peer-to-peer communications channel has been established with the destination, such that no insecure storage of the data message is required en route.

20

The present invention is embodied in a system where there is a central repository of usernames and their IP addresses. However, the central repository is not used in the actual peer-to-peer communication itself, but rather serves to update the communications means provided at the sender's location. Users register with the central repository and are assigned
25 names for their specific IP address. The registered user then downloads and installs a personal communications server onto their computer for facilitating peer-to-peer communications. Whenever a registered user comes on-line as determined by use of their personal communications server, the central repository is notified and the status of the user can be changed from off-line to on-line. This change in status can also be communicated to
30 all other users via a push mechanism such that any peer-to-peer communications that need

to be carried out between existing on-line users and the user who has just connected can be done so secure in the knowledge that both recipient and sender should at this instant be able to send and receive the communication.

- 5 Furthermore, the system described above has downloaded software components resident on the client personal computer which control the messaging function. These are described in greater detail later but include for an e-mail messaging embodiment, an e-mail server which co-operates with a local e-mail client such as Microsoft Outlook™ to handle the sending and receipt of all data messages without the requirement for the user to access any central
10 server, or in fact any user interaction. In other words, the client's personal computer is able to control the desired peer-to-peer communications and the process is very easy from the user's point of view.

- Further advantages of the present invention, particularly in relation to e-mail messaging,
15 include the ability to prevent spam e-mails from being received. As the e-mail addresses are all qualified by the central database, any source of spam can be immediately barred. Furthermore, the address of the recipient is not readily available with the present invention unlike conventional e-mail addresses.

- 20 It is also to be appreciated that VOIP can be provided relatively easily because the user can instantaneously be notified of any people who are available for a phone conversation, namely on-line at any time as this status information is pushed-to the person wishing to make the call. As the VOIP connection is via the Internet, it is free, such that long distance and local calls can be made without any additional charge to the standard internet
25 connection charge paid by the user.

- Thus the present invention allows users to send and received confidential business and personal information in total privacy. Also if the present invention is used in conjunction with conventional e-mail, the present invention provides a back up system to the
30 conventional e-mail server, providing continuity of service in the event of a fault, disaster,

physical or cyber attack.

The present invention means that third parties such as ISPs can no longer glean private information from received e-mails about the sender and their business for advertising or other purposes. Also, no additional equipment is required such as expensive e-mail servers and the expense of technical support is also mitigated which can considerably reduce costs.

As will be highlighted below, the present invention can work through firewalls, and requires no additional configuration. The present invention can integrate seamlessly with existing e-mail applications, such as Microsoft Outlook™ and Novell Groupwise™.

Peer-to-peer communication with an IP address is relatively easy when the IP address is the actual IP address of the recipient. However, when a NAT is employed, peer-to-peer communication becomes more difficult because of the inability of the sender to uniquely identify the recipient and also because of the possible security function of the NAT as a firewall. Another aspect of the present invention is directed to this specific issue (described later).

The present aspect of the present invention may also be considered to provide a communications server arranged to carry out a secure peer-to-peer data communication, such as an e-mail communication, between a first remote party computer including the server and a second remote party computer over a data communications network, the server comprising: receiving means for receiving the address details and current status of a connection to the data communications network of each remote party computer; a message generation module for creating the data communication at the first remote party computer; a checking module for checking the current connection status of the second remote party computer; and a transport module arranged to send the data communication from the first remote party computer directly to the second remote party computer without storage of the complete data communication en route, only when the connection status of the second remote party computer indicates that it is currently connected to the data communication

network.

The present aspect of the present invention also extends to a communications system comprising: a plurality of communication servers as described above; and a data server
5 connectable to the plurality of communications servers by the data communications network, wherein the data server is arranged to receive, collate, and store the current status of the connection of each of the communication servers to the communications network together with the current network address of each of the communications servers and to forward at least part of this information to the plurality of communications servers to enable
10 them to effect peer-to-peer communications between ones of the plurality of communications servers.

According to a second aspect of the present invention, there is provided a communications server arranged to assist in establishing a secure peer-to-peer data communication, such as
15 an e-mail communication, between first and second user computers over a data communications network, the server being provided at a given hierarchical level within a server network comprised of a plurality of the communications servers, the server comprising: connection means enabling the communications server to be able to connect operably to other communications servers at other hierarchical levels within the server
20 network; registration means for registering a plurality of local user computers with the communications server; and a data store for storing registration details of each registered local user computer, the registration details including address information and current status of a connection to the data communications network of each local user computer; wherein the connection means is arranged to forward the stored registration details to an adjacent
25 communications server in the next higher hierarchical level of the server network and to receive and store registration details of any local user computers for a connected communications server at a lower level in the hierarchical network.

Use of a hierarchical network of transport servers is highly advantageous because it enables
30 a messaging network to operate more efficiently. In particular, when high demands are

placed on a communications network by the high volume and large payloads of messaging traffic, the use of a hierarchical structure in the communications medium enables much of the traffic to be handled locally without placing an undue burden on the whole of the communications system. VoIP message traffic can have a particularly heavy payload and can use up much of the available bandwidth. Use of the hierarchical structure and the distributed recording of addressing details can help to localise the traffic to particular parts of the communication network thereby not effecting the performance characteristics of other parts of the network. This has a major performance benefit on, for example, VOIP communications as there most VOIP calls are between geographically local source and destinations.

The present second aspect of the present invention also extends to a method of assisting in establishing a secure peer-to-peer data communication, such as an e-mail communication, between first and second user computers over a data communications network, the method being implemented on a communications server at a given hierarchical level within a server network comprised of a plurality of the communications servers, the method comprising: establishing operable network connections to communications servers at other hierarchical levels within the server network; registering a plurality of local user computers with the communications server; and storing registration details of each registered local user computer, the registration details including address information and current status of a connection to the data communications network of each local user computer; wherein the establishing step comprises forwarding the stored registration details to an adjacent communications server in the next higher hierarchical level of the server network and receiving and storing registration details of any local user computers for a connected communications server at a lower level in the hierarchical network.

According to a third aspect of the present invention there is provided a method of searching for an intended recipient computer of a peer-to-peer communications message to assist in establishing a secure peer-to-peer data communication, such as an e-mail communication, between a sender computer and the intended recipient computer over a data

communications network, the method being implemented within a hierarchical network of communications servers and comprising: sending to a local server a request for data communication, the request including the intended recipient computer's identity and the sender computer's identity; determining whether the intended recipient is known to the local server; retrieving stored details regarding the intended recipient if the same is known to the local server, and transmitting these details back to the sender computer; forwarding the request to an adjacent communications server in a next higher hierarchical level of the server network if the intended recipient is not known locally, the adjacent communications server then becoming the local server; and repeating the determining, retrieving and forwarding steps until the intended recipient is found or the server at the summit of the hierarchical network has been checked.

This manner of using a hierarchical communications network is advantageous in that the procedure for finding of a destination address can be considerably faster than in the prior art. The reason of this is that geographically local addresses to the source are checked first and gradually more and more remote addresses are checked at higher and higher positions in the hierarchical network until the address is found or the summit reached. Again as many communications are actually directed to local destinations, these can be found quickly and the potential administration burden on the network of communications servers can be avoided.

The present third aspect of the present invention also extends to a method of establishing a secure peer-to-peer data communication, such as an e-mail communication, between a sender computer and an intended recipient computer over a data communications network, the establishing method being implemented within hierarchical server network comprised of a plurality of communications servers, the establishing method comprising: a method of searching for an intended recipient computer as described above; communicating the current global communications address of the intended recipient computer to the sender computer; communicating the current global communications address of the sender computer to the intended recipient computer; and using the global communications

addresses to set up a peer-to-peer communications channel between the sender and the intended recipient computers.

According to a fourth aspect of the present invention there is provided a transport server for use in establishing a peer-to-peer data communication, such as an e-mail communication, between a first and second user computers, the transport server comprising: receiving means for receiving from the first user computer a request for connection to the second user computer which is registered with the transport server; verifying means for verifying the current connection status of a connection to the second user computer; a data store for storing details of the request as a watch if the current connection status of the second user computer indicates that a peer-to-peer communication cannot at present be established with the second user computer; and response means responsive to the verifying means to send a message to the first user computer indicating the on-line status of the second user computer if the status of the same changes to indicate that a peer-to-peer communication can now be established with the second user computer; wherein the verifying means is arranged to periodically check and update the current connection status to the second user computer, and when an update indicates that the second user status has changed to on-line, to check for the existence of a corresponding watch and if found to activate the response means to send the message.

20

Transport servers, according to this fourth aspect of the present invention, are provided with a mechanism for monitoring the connection status of an intended recipient, which advantageously does not involve the message sender's resources. Rather, they can effectively watch the connection status of the intended recipient, and when it changes such that a peer-to-peer communications then becomes possible, the message sender can be notified. This also supports data messaging without any storage of the data message on-route at a potentially insecure location.

25

The fourth aspect of the present invention is also realised in a method of assisting the establishment of a peer-to-peer data communication, such as an e-mail communication,

30

between a first and second user computers, the method comprising: receiving from the first user computer a request for connection to the second user computer which is locally registered; verifying the current connection status of a connection to the second user computer; storing details of the request as a watch if the current connection status of the second user computer indicates that a peer-to-peer communication cannot at present be established with the second user computer; and sending a message to the first user computer indicating the on-line status of the second user computer if in response to the verifying step the status of the second user computer changes to indicate that a peer-to-peer communication can now be established with the second user computer; wherein the verifying step comprises periodically checking and updating the current connection status to the second user computer, and when an update indicates that the second user status has changed to on-line, checking for the existence of a corresponding watch and if found activating the response means to send the message.

A fifth aspect of the present invention addresses the issue of overcoming the problems associated with NATs and firewalls such that the peer-to-peer communications can be supported. This aspect of the present invention resides in the appreciation that the mapping function of a NAT can be determined by sending the implementation of a series of communications (investigations) of the capability of the intended recipient including the determination of the actual local address of the recipient even though this is not normally visible on the remote communications side of the NAT. These investigations include use of a communications control channel and the important function of the intended recipient sending its local address as data within a message through the NAT such that the translation function of the NAT can be extrapolated. Other investigations include the ability to provide the results of investigations on the potential intended recipient.

The benefits of effectively determining the mapping function of the NAT is that this can then be used to send the NAT an appropriately addressed communication which will be mapped back by the NAT to the desired single recipient location. The NAT is completely unaware that its mapping function has been determined and as a result direct peer-to-peer

communications can be supported.

An alternative way of overcoming the problems associated with NATs and firewalls is to use established TCP/IP links to transport servers to help in creating UDP communications channels for supporting peer-to-peer communications. More specifically, according to a fifth aspect of the present invention, there is provided a method of establishing a secure peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where at least the first user computer has its communications handled by a first Network Address Translator (NAT), the method comprising: requesting a direct connection over an established Transmission Control Protocol/Internet Protocol (TCP/IP) communications link between the first user computer and a transport server through the first NAT; establishing a first User Datagram Protocol (UDP) port at the transport server; reporting the address of the first UDP port to the first user computer via the TCP/IP communications link; opening a second UDP port at the first user computer; transmitting data packets from the second UDP port via the first NAT to the first UDP port such that the transport server is able to determine the first NAT address of the second UDP port; obtaining at the transport server a third UDP port address of the second user computer; and notifying each of the first and second user computers of the other user computer's UDP port address such that they can establish a secure peer-to-peer communication therebetween.

It is to be appreciated that NAT traversal is a significant issue when seeking to establish peer-to-peer communications. This fifth aspect of the present invention provides a simple workable solution which enables NAT traversal for peer-to-peer communications, even in the case when one of the NATs is asymmetric.

The fifth aspect of the present invention also extends to a transport server for assisting in establishing a secure peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where at least the first user computer has its communications

handled by a first Network Address Translator (NAT), the method comprising: request receiving means for receiving a request for a direct connection over an established Transmission Control Protocol/Internet Protocol (TCP/IP) communications link between the first user computer and the transport server through the first NAT; establishing means
5 for establishing a first User Datagram Protocol (UDP) port at the transport server; reporting means for reporting the address of the first UDP port to the first user computer via the TCP/IP communications link; data packet receiving means for receiving data packets transmitted from a second UDP port set up at the first user computer via the first NAT to the first UDP port, the transport server being arranged to determine the first NAT address of
10 the second UDP port; obtaining means for obtaining at the transport server a third UDP port address of the second user computer; and notifying means for notifying each of the first and second user computers of the other user computer's UDP port address such that they can establish a secure peer-to-peer communication therebetween.

15 The most complicated situation regarding NAT traversal arises when it is simply not possible to implement a direct peer-to-peer connection either because of severe firewall implementations or because both parties have asymmetric NATs. This situation can be resolved by establishing a 'pseudo' peer-to-peer communication to traverse the asymmetric NATs. More particularly, according to a sixth aspect of the present invention, there is
20 provided a method of establishing a secure pseudo peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where both the user computers have their communications handled by respective first and second asymmetric Network Address Translators (NATs), the method comprising: creating a Transmission Control
25 Protocol/Internet Protocol (TCP/IP) communications link from each of the user computers to a transport server through the respective first and second asymmetric NATs; establishing a first and a second User Datagram Protocol (UDP) port at the transport server in response to receiving a request for a direct connection, the request being received over the TCP/IP communications link between the first user computer and a transport server through the first
30 NAT; reporting the addresses of the first and second UDP ports to the first and second user

computers respectively via their respective TCP/IP communications links; opening a third UDP port at the first user computer and a fourth UDP port at the second user computer; transmitting data packets from the third UDP port via the first NAT to the first UDP port and from the fourth UDP port via the second NAT to the second UDP port such that the transport server is able to determine the first NAT address of the third UDP port and the second NAT address of the fourth UDP port; and retransmitting data packets received at the first UDP port to the NAT address of the fourth UDP port via the second UDP port and data packets received at the second UDP port to the NAT address of the third UDP port via the first UDP port, such that pseudo peer-to-peer communications between the first and second user computers is established by effectively bouncing data packets off of the transport server.

As described above, the pseudo peer-to-peer communication is provided by the ability of the present aspect of the invention to 'bounce' out messages off a local transport server that acts as a trusted intermediary. This mitigates issues of unknown sender as the local transport server will always be known to the local asymmetric such that even the most severe rules at a firewall, for example, will not prevent such indirect pseudo peer-to-peer communications. Furthermore, the possible reduction in security of such an indirect communications channel is minimised by ensuring that the bounced out packets are forwarded by the transport server acting as the trusted intermediary with minimal analysis, and certainly no storage.

In the case where the idiotic (very severe) interpretation of firewall action is encountered, the present aspect of the present invention is arranged to implement a true peer-to-peer communication in one direction and to bounce off a local transport server in the other direction. This variation is worth doing because it minimises the reduction in security. For example, in the case of e-mail transmission, it is advantageous to arrange that the e-mail is sent directly, and that the acknowledgements are bounced back.

This sixth aspect of the present invention also extends to a transport server for assisting in

establishing a secure pseudo peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where both the user computers have their communications handled by respective first and second asymmetric Network Address Translators (NATs),
5 the transport server comprising: creating means for creating a Transmission Control Protocol/Internet Protocol (TCP/IP) communications link from each of the user computers to a transport server through the respective first and second asymmetric NATs; establishing means for establishing a first and a second User Datagram Protocol (UDP) port at the transport server in response to receiving a request for a direct connection, the request being
10 received over the TCP/IP communications link between the first user computer and a transport server through the first NAT; reporting means for reporting the addresses of the first and second UDP ports to the first and second user computers respectively via their respective TCP/IP communications links; receiving means for receiving data packets transmitted from a third UDP port set up at the first computer via the first NAT to the first
15 UDP port and from the fourth UDP port set up at the second user computer via the second NAT to the second UDP port such that the transport server is able to determine the first NAT address of the third UDP port and the second NAT address of the fourth UDP port; and retransmitting means for retransmitting data packets received at the first UDP port to the NAT address of the fourth UDP port via the second UDP port and data packets received
20 at the second UDP port to the NAT address of the third UDP port via the first UDP port, such that pseudo peer-to-peer communications between the first and second user computers is established by effectively bouncing data packets off of the transport server.

According to a seventh aspect of the present invention, there is provided a method of
25 connecting a user computer to a hierarchical connection network of transport servers for assisting in establishing a peer-to-peer communication between a sender computer and an intended recipient computer over a data communications network, the method comprising: receiving information describing the current loading of each of a plurality of peer transport servers at the same hierarchical level in the connection network as a local transport server;
30 receiving a request to connect a local user computer to the local transport server; comparing

the current loading of the local server with each of the peer servers; sending a response to the local user computer indicating that it should connect to the peer server having the lowest loading if the loading of the local transport server is significantly greater than the loading of any one of the peer servers; and accepting the request to connect if the loading of the local transport server is not significantly greater than the loading of any one of the peer servers, and updating the current loading of the local transport server.

This aspect of the present invention addresses the problem of load balancing in a communications network. By keeping a measure of the current loading of the local servers within a hierarchy of transport servers, it is always possible to redistribute requests for establishing peer-to-peer connections efficiently. By keeping a communications balanced in terms of loading, the network itself becomes more efficient and any reduction of performance is kept to a minimum.

According to a eighth aspect of the present invention, there is provided a method of joining a node to a network of authenticated communications servers set up to be used in establishing a peer-to-peer data communications between user computers registered with the communications servers, the method comprising: using a user identity and password received from an authentication server to authenticate the node to the authentication server; being notified of the identity of a selected communications server to which the node needs to connect to join the network; requesting selected communications server and node specific data from the authentication server for a connection to the selected communications server; receiving selected communications server and node specific data and a shared encryption key shared by the node and the selected communications server; transmitting the selected communications server and node specific data and global data encrypted by the shared encryption key to the selected communications server, such that the selected communications server has the tools by which to authenticate the node to the network and thereby join to the same without the need to seek verification from the authentication server.

This aspect of the present invention provides a very secure way of implementing an authentication procedure for a new node wishing to join a network. This is particularly important in the context of a messaging source wishing to join a peer-to-peer communications facilitating network where access to the network bestows privileges upon the messaging source, for example being allowed to communicate freely with other members of the communications network.

According to a ninth aspect of the present invention, there is provided a method of connecting a first hierarchical realm of interconnected transport server nodes to a second hierarchical realm of interconnected transport server nodes, the method comprising: providing a local authenticating server within each realm, the authenticating server being connected to a primary node at the highest level of the respective hierarchical realm and being arranged to control the authentication issues related to all servers within the realm; registering the primary node of the first hierarchical realm to the authenticating server of the second hierarchical realm; authenticating the primary node of the first hierarchical realm to the authenticating server of the second hierarchical realm; being notified of the identity of a lowermost node server to which the primary node of the first hierarchical realm needs to connect to join the hierarchical realms together; receiving shared data and shared encryption keys for both of the lowermost node server of the second hierarchical realm and the primary node of the first hierarchical realm; using the receiving shared data and shared encryption keys to authenticate the primary node of the first hierarchical realm to the lowermost node server of the second hierarchical realm and thereby to join the first and second realms together without the need to seek verification from the authenticating server of the second realm.

25

This aspect of the present invention is advantageous in that it enables corporations with their own communications networks (realms) to join together without loss of security as a result. Also the integrity of each realm remains intact as only the permissions in higher realms need to be modified in the above simple manner in order to effect the joining together of what could be very significant sized existing networks. This means that

30

communications between people in completely different organisations becomes possible in a very secure way using peer-to-peer technology.

Brief Description of the Drawings

5 Figure 1 is a schematic block diagram of a system according to an exemplary embodiment of the present invention;

Figure 2 is a schematic diagram showing an example of how the system of Figure 1 can be used to enable peer-to-peer communications between two users, Bob and Alice;

10

Figure 3 is a block diagram showing a method of overall operation of the personal e-mail server of Figure 1;

Figure 4 is a block diagram of a ConnectfromClient subroutine of the method of Figure 3;

15

Figure 5 is a block diagram of a MessageReceive subroutine of the method of Figure 3;

Figure 6 is a block diagram of a MessageWaitingToSend subroutine of the method of Figure 3;

20

Figure 7 is a flow diagram showing the interaction between the Transport server mechanisms and the mails server for determining the unique address of an intended recipient;

25 Figure 8 is a schematic diagram illustrating the different uses of the system by different entities;

Figures 9a to 9f are a series of schematic block diagrams illustrating the investigations which are carried out by the TSMs in determining the unique IP address for the intended
30 recipient;

Figures 9g to 9i show how the investigations carried out in Figures 9a to 9f can be applied in the case of symmetric NATs and also in the case of asymmetric NATs;

5 Figure 10 is a schematic block diagram showing the overall functional tree architecture of a second embodiment of the present invention;

Figure 11 is a schematic block diagram showing the hierarchical nature of the administration level of the tree architecture of Figure 10;

10 Figure 12 is a schematic block diagram showing the hierarchical nature of the transport server layer of the tree architecture of Figure 10;

Figure 13 is a schematic block diagram showing the functional elements of a generic transport server of the transport server layer of Figure 12;

15

Figure 14 is a flow diagram showing the process of load balancing new connections to the administration layer of Figure 10;

20 Figure 15 is a flow diagram showing the first stage of establishing a peer-to-peer connection using the network of the second embodiment of Figure 10;

Figure 16 is a flow diagram showing the second stage of establishing a peer-to-peer connection using the network of the second embodiment of Figure 10;

25 Figure 17 is a schematic block diagram showing how the principles of the Kerberos technique are applied in a third embodiment of the present invention;

Figure 18 is a block flow diagram showing the main steps of the third embodiment in seeking a connection to a network;

30

Figure 19 is a schematic block diagram showing the details of the registering with a central data server and thereafter how a peer-to-peer communications is supported from a security angle by the network;

- 5 Figure 20a is a schematic block diagram of a parent node within the hierarchical network of Figure 19;

Figure 20b is a schematic block diagram of a child node within the hierarchical network of Figure 19; and

10

Figure 21 is a schematic block diagram showing how cross-realm linking is achieved in effecting a peer-to-peer call between user computers X and Y.

Detailed Descriptions of the Preferred Embodiments

- 15 Referring to Figure 1, a system 10 embodying the present invention is shown. This embodiment is described with reference to the presently preferred e-mail communication. However, it is to be appreciated that any form of electronic communication transmission can be used such as Voice over IP (VoIP) or instant messaging for example. Furthermore, even though the present embodiment is described in relation to communications over a
- 20 Wide Area Network such as the Internet, the invention can be implemented over a Local Area Network, or could even be carried out via mobile telecommunications.

- The system 10 comprises two personal computers (PCs) 12, 14 provided at different locations. The first PC is a local PC 12 and the second PC is a remote PC 14 and the system
- 25 10 supports peer-to-peer communications between these two computers 12, 14. Each PC 12, 14 is identical in its communication function and therefore it is only necessary to describe one in detail.

- Each PC 12, 14 is connected to the Internet 16 in this embodiment, via a respective Internet
- 30 gateway 18, 20. The Internet gateway 18 at the local PC 12 also includes a NAT 22 as the local PC 12 is provided as one of a plurality of local PCs 12 on a LAN 24 which all utilise

the local Internet gateway 18. The remote PC 14 is also provided on a LAN 26 but in this case no NAT is provided and the remote PC 14 has a unique IP address.

5 A data server 28 is also provided which is connected to the Internet 16. The data server 28 has a local database 30 of usernames 32 and IP addresses 34 which it manages. The local database 30 also stores the connection status 36 of the registered users. These usernames 32 and addresses 34 have been provided to the database 30, including those of the local and remote PCs 12, 14, by way of a registration procedure which is not described in detail herein as such procedures are commonly known. It is to be appreciated that the data server
10 28 does not take active part in peer-to-peer communications between registered PCs, such as the local and remote PCs 12, 14. It does, however, update all the PCs 12, 14 which are registered with the latest address information 34 and the status 36 of that address, namely whether it is on-line or off-line.

15 A plurality of transport server mechanisms (TSMs) 40 are provided which are all connected to the Internet 16. These TSMs 40 facilitate the establishment of the correct address 34 for a peer-to-peer communication between the local and remote PCs 12, 14. Each TSM 40 has associated with it a local store 42 of information for storing addressing information regarding each of the possible addresses of the registered PCs 12, 14. More specifically,
20 these local stores comprise a list of PC identities 44 and a corresponding list of IP addresses 46. In the list of IP addresses 46, a direct IP address 46 is provided in the cases when the TSM 40 is itself responsible for addressing of a given PC. Otherwise, in cases where it is not responsible, an IP address (or other suitable identification) of the responsible TSM 40 is provided. The operation of the TSMs 40, particularly with reference to NAT traversal, is
25 described in detail later.

Each of the local and remote PCs 12, 14 includes a mail client 50 such as Microsoft Outlook™, which is connected via TCP/IP to a personal mail server 52. The mail server 52 has its own local data store 54 provided and is also connected to a network interface 56. The
30 mail server 52 is provided in the downloaded software and is installed by the user in order

to utilise the peer-to-peer communications protocol. The mail server 52 stores the network address of the data server 28 as it needs to contact it to let it know when it goes on-line. All peer-to-peer communications are routed through the personal mail server 52 in a transparent manner such that the operation of the mail client 50 is unaffected.

5

The network interface 56 and the internet gateways 18,20 used in the present embodiment are entirely conventional and need no further explanation herein.

The general mode of operation of the system 10 is now described. Once the user has registered with the service by providing its details to the data server 28, the personal e-mail server 52 is downloaded and installed locally on the user's PC 12, 14. The personal e-mail server 52 operatively connects to the e-mail client 50 and the network interface 56, both of which are already provided on the PC 12, 14. All e-mail communications to and from the e-mail client 50 are via the personal e-mail server 52.

15

In sending a message, the user creates the e-mail and then sends it to the personal e-mail server 52. If the intended recipient is on-line, as indicated in the current information stored regarding the status of the intended recipient, then the e-mail message is sent directly to the intended recipient's PC 14. Otherwise, the e-mail message is stored in a queue until the intended recipient's PC 14 comes on-line as would be indicated by the receipt of an updated on-line status message from the data server 28.

20

Before sending the message, the actual current IP address 34 of the intended recipient is determined by the TSMs 40. This process is described in detail later. However, on confirmation of the present IP address 34 of the intended recipient, the peer-to-peer communication of the e-mail message is carried out.

25

Referring to Figure 2, an example of how this process 60 is carried out is described with reference to two users Alice 62 and Bob 64. The process 60 commences with Alice's personal e-mail server 52 sending 66 a message to the data server 28 that she has come on-

30

line and indicating 68 that she wants to send a message to Bob 64. The data server 28 then replies 69 with Bob's current address details. Alice's personal e-mail server 52 then uses this information to address the message, encrypt it 70 and store it 72 locally for sending to Bob when he goes on-line.

5

At some point later when Alice is on-line (having sent 66 a message to the data server 28 to indicate this) and Bob goes on-line (sending 74 a message to the data server 28 to indicate this), the data server 28 sends 76 a message out to all users to indicate that Bob 64 has come on-line. This message is received by Alice's personal e-mail server 52 and the stored
10 message can then be sent 78 directly to Bob 64. On receipt of the message, Bob's personal e-mail server 52, stores it 80. When Bob's e-mail client 50 checks 82 for incoming messages (get message), then the message is decrypted 84 and sent 86 to the e-mail client 50 for Bob 64 to read.

15 The actual processes which are carried out at each personal mail server 52 are now described with reference to Figures 3 to 6.

The operation 90 of the personal e-mail server 52 is now described with reference to Figure 3. The operation 90 commences with the start up 92 of the server 52, which leads to a
20 connection being made 94 to the data server (DS) 28. The address of the data server 28 is already known to the mail server 52 and is stored in the local memory 54. The data server 28 provides the location addresses of several TSMs 40, one of which must be available for the system 10 to work. The mail server 52 then tries to establish 96 a connection with a TSM 40. As part of this, a check 98 is carried out to see if the TSM 40 is on-line. If a TSM
25 40 is not on-line and a connection cannot be made, then the mail server 52 is considered to be off-line 100. The process 90 can continue with a subsequent reconnection 102 to the Internet 16 and return to the connection to the DS step 94 described above.

However if the TSM 40 is on-line 104, one of five different options (states) become
30 available. Firstly, there is an option if the PC 12, 14 is shut down 106 for the process 90 to

end 108. Secondly, the mail server 52 can be forced to log out 110, thereby taking the process 90 to the off-line condition 100 as mentioned above. Thirdly, if there is an attempt from the mail client 50 to connect 112 to the mail server 52, then the credentials of the mail client 50 are checked 114 against pre-stored information and if acceptable, a connection
5 from the client can be made 116 (as is described in greater detail later with reference to Figure 4). If the client credentials are not verified, then the connection is refused 118 and the mail server 52 either goes back 120 into an on-line state or is forced off-line 122 and has to connect 102 to the Internet 16 and start the process 90 again.

10 A further option (state) is for the mail server 52 to have received 124 a message and for it to be provided to the mail client 52 for presentation to the user. This process 124 is described later in detail with reference to Figure 5.

The last option (state) available once the mail server is on-line, relates to the sending of
15 messages. A check 126 is made to determine if there are messages waiting to be sent. If there are no messages to be sent 128, the process 90 ends and the mail server 52 returns to the on-line state 104. If there are messages to be sent, the MessageWaitingToSend process 130, as described in greater detail later with reference to Figure 6, is then executed. Following the end of this latter process 130, the mail server 52 returns to the on-line state
20 104.

Referring now to Figure 4, the ConnectionFromClient process 116 is now described. The process 116 commences with an SMPT connection request being received 140 from the mail client 52 (for example MS Outlook™). The process 116 determines, for security
25 purposes, whether the mail client 52 is authorised 142. If they are not, then the authorisation fails 144 and a message is sent 146 to the mail client 52 to notify the user. Otherwise, the authorisation succeeds 148 and the e-mail message from the mail client 50 is received 150 by the mail server 52. This is a multi-stage process with elements of the message being transferred successively (for example the 'from' address, followed by the 'to' addresses,
30 followed by the body of the message are sent). This process is carried out until a complete

message has been received. A check 152 is carried out to determine if there are any more messages to be sent and if so, the transfer process 150, 152 is repeated. When there are no further messages 154 to be received, the mail server 52 disconnects 156 from the mail client 50.

5

Before a received message is processed, checks can be made 158 to determine the send permissions associated with the mail client 50. If it is determined 160 that the mail client 50 does not have the relevant permissions to send to a particular intended recipient, then the e-mail message will not be sent 162. However, if the send permissions are in place, then the

10 Destination User Info is obtained 164. This step involves requesting and getting the address information of all registered intended recipients from the data server 28. This data also includes the public key (not shown) of the user for use in encrypting messages for that user. If address information has been received recently in connection with another e-mail message, then this information may be held in cache. Accordingly, a check (not shown) to
15 see if the address information for a particular intended recipient is in cache can be carried out prior to seeking it from the DS 28.

Subsequently, a check is carried out to determine 166 if the intended recipient is a recognised mail user. If he is not 168, then the mail server 52 simply informs the mail client

20 50 that the message could not be sent 170, as the address is unknown to the system. If however the intended recipient is recognised 172 as a registered user, then the e-mail message is encrypted 174 with the public key of the intended recipient, which has been obtained earlier, such that Public Key Infrastructure (PKI) encryption techniques can be used for encryption/decryption.

25

The encrypted message is then passed onto the MessageWaitingToSend process 130, which is described later with reference to Figure 6.

Referring now to Figure 5, the MessageReceived process 124 is now described. In order to
30 receive an e-mail message the mail server 52 does not have to be logged-in but it does have

to be connected to the Internet 16. On receipt 180 of an e-mail message, the user is notified 182 of the message and this can be carried out by the generation of a Notify Recipient pop-up window 184 for example. The received e-mail message is stored in memory 54 pending a 'get' (retrieval) from the mail client. The storage can be to the computer's hard disk but
5 for the most secure solution, the received message is stored in RAM until it is retrieved. Whilst it is waiting 186, if for some reason the mail server 52 has to be closed 188, the undelivered message is stored 190 in semi-permanent memory (typically the hard disk) and the process 124 ends 192.

10 When the mail server 52 receives 194 a get request from the mail client 50, the process 124 checks 196 to determine whether the mail client 50 is authorised. If it is not, the client authorisation fails 198, resulting a failure 200 of the mail client 50 get procedure. This in turn is notified 202 to the recipient user and to Amteus (which is the authority overseeing the operation of the whole system and with whom the users are registered).

15 If the mail client 50 is authorised 204, two actions are performed in parallel. The private key of the recipient, which is stored locally, is retrieved 206 and the undelivered e-mail message is retrieved 208. Then an attempt 210 to decrypt the message is carried out using the retrieved private key. If the decryption is unsuccessful 212, then the sender is notified
20 214. This is achieved by the mail server 52 sending an e-mail back to the sender stating that there is a delivery failure. Then the process 124 continues as if the earlier client authorisation failed 198, namely with a failure 200 of the mail client 'get' procedure. This in turn is notified 202 to the recipient user and to Amteus.

25 If however the decryption is successful 216, then the decrypted e-mail message is forwarded to the mail client for presentation to the user, namely the 'getby' the mail client is successful 218. The message received process 124 then checks 220 to see if there are any further messages to be delivered. If there are no further messages 222, the message received process 124 ends 192. If there are further messages to be delivered 224, then the process
30 loops back 226 to the stage where the client authorisation is successful 204 and continues as

has been described previously.

Referring now to Figure 6, the message waiting to send process 130 of Figure 3 is described. When there is an e-mail message in a queue waiting to be sent 230 then the status of the destination has to be determined 232. A check 234 is carried out to determine if the current status of the destination is in cache. If it is available in cache, then it is retrieved 236 from cache. If it is not available in cache 238, then it is retrieved 240 from the DS 28 and the cache is updated with the retrieved status.

10 The destination status is checked 242 to determine whether the destination is off-line. If it is off-line 244, then the message cannot be sent and the encrypted message is simply stored 246 in the message-waiting queue for a later attempt at transmission. If however, the destination is on-line, then the encrypted message is retrieved 248 from the queue (either in RAM or from the hard disk) and it is sent 250 directly to the intended recipient as a peer-to-peer communication. The details of how the accurate addressing of the e-mail message is carried out and how the message is sent are outlined in Figure 7 and are described later.

The transmission of the message is checked 252 and if for some reason the transmission failed 254, then a send error state 256 is entered where a procedure for determination of the type of error follows. This state 256 is also reached at the start of the message waiting to send process 130 if the mail server 52 is stopped 258 for some reason. In this send error state 256, Amteus are notified 260, in parallel with determination of the type of error. A check 262 is made to determine whether the error was due to the destination being off-line. This could occur if there has just been a change in status of the intended recipient. If the destination was off-line 244, then the encrypted message is stored 246 for later transmission. This ends this pass of the message waiting to send process 130, though not the process 130 in itself as that e-mail message will still be in the queue waiting to be sent.

If the error was not due to the destination being off-line, then another error is responsible 264 and a check 266 is carried out to determine whether it was a permanent or temporary failure. In the case of a permanent failure 268 (for example if the user no longer exists, or

the e-mail is too big), the mail server receives 270 a bounce back message saying that the message was undeliverable. If however, the error is due to a temporary failure 272 (due for example to a connection fault), then the encrypted e-mail message is stored 242 back in the queue and this pass of the message waiting to send process 130 ends.

5

Returning to the results of the transmission of the e-mail message, if it is confirmed that the message was delivered 274 then the local copy of the e-mail is deleted 276 for security purposes. A notification of the successful transmission is sent 260 to Amteus and a check 278 is made to determine if there are any more e-mail messages in the queue waiting to be sent. If there are, then the process 130 repeats returning to the checking 232 of the status of the destination (intended recipient) for that e-mail message and continuing as has been described above. Otherwise, there are no more messages to be processed 280 and this pass of the message waiting to send process 130 ends.

10

15 Referring now to Figure 7, the operation and interaction with the TSMs is described and in particular how they operate to establish the intended recipient addressing required for the peer-to-peer communications.

Firstly, it is to be appreciated that the TSMs are continually carrying out their address investigation process such that at any time when delivery of an e-mail message to an intended recipient is required, they are in possession of the most up-to-date current view of the address information for that intended recipient. Their investigations are carried out using a modified UDP (User Datagram Protocol) and comprise experiments, which help to establish the actual direct IP network address of that user. These experiments enable NAT traversal and are described in greater detail later with reference to Figures 9a to 9i.

20

25

The process 274 of determination and communication of the recipient's current IP address to the sender commences with the sender being sent 290 a list of TSMs from a master TSM 40, which holds the list. The mail server 52 then attempts 292 to contact the first of the TSMs 40 on the list and check 294 if it is successful. If it is not successful 296, then the next TSM 40 on the list is retrieved and another attempt 292 is made to connect. This

30

process continues until one connection is successful 298.

On connection, the selected TSM notifies 300 all the other TSMs 40 that he has the connection for this mail server 52. This connection may be made on the basis of which
5 TSM 40 is the most convenient to use with regard to load balancing. The sender mail server 52 then communicates 302 to the connected TSM 40 that it wishes to connect to the intended recipient mail server 52. The intended recipient is identified by way of user ID. It is the task of the TSM 40 to determine the IP address for that intended recipient mail server 52. The selected TSM 40 then checks 304 its addressing list (stored locally) to see if it has a
10 direct connection to the intended recipient mail server 52. If it does have a direct connection, namely it knows the unique address for the recipient mail server 52, then this is passed 306 onto the sender mail server 52. This address is determined using techniques described later with reference to Figures 9a to 9i. If it does not have the unique address itself, then it at least has the address of the TSM 40, which does have the address.
15 Accordingly in this case, the selected TSM 40 relays 308 the enquiry onto the correct TSM 40 such that it can look up the address and pass this back 306 to the sender mail server 52.

Once the sender mail server 52 has received the unique direct IP address of the intended recipient mail server 52, it connects 310 directly to the recipient mail server 52 and sends
20 312 the e-mail message using standard Internet protocol. Namely, the e-mail message is sent in the same way that conventional IP communications are sent on the Internet 16, by dividing the message into packets each with their own header and transmitting these packets to be routed by any appropriate route to the recipient where they are assembled together again for presentation to the intended recipient.

25

Figure 8 is a schematic diagram showing the different activities of a user 320, its mail client 50,322 and an administrator 324. The different activities are grouped into related functions as has been shown on the diagram. More specifically, the use 320 can carry out three different groups of related functions, namely set up activity 326, payment activities 328,
30 and file related activities 330. The mail client 50, 322 can carry out a single group of mail server-related activities 332. The administrator 324 can carry out a single group of user-

related activities 334. The functions that make up each of these groups are illustrated in Figure 8 and will be clear to the skilled addressee, and therefore no further explanation is provided herein.

5 Figures 9a to 9f show and explain the six stages of actions, which are carried out by the TSMs 40 in determining the unique IP address for the intended recipient. More specifically, an initial state is shown in Figure 9a, where Client A 350 has made a TCP/IP connection via NAT A 352 to Server S 354, and similarly, Client B 356 has also established a TCP/IP connection via NAT B 358 to Server S 354. At this stage, labelled Stage 1, Client A 350
10 desires to established a peer-to-peer connection with Client B 356.

In Stage 2A shown in Figure 9b, Client A 350 makes a peer-to-peer request to Server S 354 via Nat A 352. In response to receiving this request, the Server S 354 opens up a UDP port 360 on IP address S₁: s₁ 362. Subsequently, in Stage 2B, as shown in Figure 9c, Server
15 S 354 reports the address (S₁:s₁) 362 of the UDP port 360 to Client A 350 via the already established TCP/IP communications medium 364. Client A 350 then opens its own UDP port 366 at address A:a 368. This port address 368 is translated by NAT A 352 to A₁:a₁ 370. However, a UDP communications channel 372 is set up between Client A 350 and the Server S 354.

20

In Stage 3, shown in Figure 9d, Server S 354 opens up a second UDP port 376 on IP address S₂: s₂ 378 such that it can establish a UDP channel with Client B 356 in a similar manner to that described in relation to Figure 9c. Then Server S 354 reports the address (S₂:s₂) 378 of the UDP port 376 to Client B 356 via the already established TCP/IP
25 communications medium 364. Client B 350 then opens its own UDP port 380 at address B:b 382. This port address 382 is translated by NAT B 358 to B₁:b₁ 384. However, a UDP communications channel 386 is established between Client B 356 and the Server S 354 as can be seen in Figure 9e.

30 Referring more specifically to Figure 9e, where Stage 4 of the process for establishing the

unique IP address of the intended recipient is shown, the way in which UDP communications between Client A and Client B are carried out is now explained. Client B sends UDP packets from its UDP port 380 with address B:b 382 to the Server UDP port 376 with address S2:s₂ 378. However, Server S 354 sees that they are coming from the network translated UDP port address B₁:b₁ 384. In a similar manner, UDP packets can be received from Client A 350 at Server UDP port 360 with address S1:s₁ 362.

Server S 354 is able to forward packets received from network translated port address A₁:a₁ 370 to B₁:b₁ 384 and from network translated port address B₁:b₁ 384 to A₁:a₁ 370. This enables Client A 350 and Client B 356 believe that they have a direct UDP connection with each other.

Figure 9f shows Stage 5 in the process and establishes the general case for peer-to-peer communications. Here, Server S 354 firstly informs Client A 350 that Client B 356 is talking on network translated port address B₁:b₁ 384. Also Server S 354 informs Client B 356 that Client A 350 is talking on network translated port address A₁:a₁ 370. Therefore the ability for Client A 350 and Client B 356 to talk directly is established as they each have the other's network translated address 370, 384. However, depending on the application (not shown) running on Client A 350, communications may be over a first UDP channel 388 between UDP port A:a 366 and address B₁:b₁ 384 or both port addresses B₁:b₁ 384 and S1:s₁ 362. Similarly, depending on the application (not shown) running on Client B 356, communications may be over a second UDP channel 390 between UDP port B:b 380 and address A₁:a₁ 370 or both port addresses A₁:a₁ 370 and S2:s₂ 378.

Figure 9g shows how the general case of Figure 5 is applied for the case when both NATs are symmetric. More specifically, in Stage 6A if NAT A 352 is a symmetric NAT then Client B will receive UDP packets from network translated port address A₁:a₁ 370. Similarly if NAT B 358 is a symmetric NAT then Client A will receive UDP packets from network translated port address B₁:b₁ 384. When such UDP packet traffic is detected, Server S is informed and it breaks the UDP connections 372 between NAT A and its first

port 362 and between NAT B and its second port 376, which is the situation illustrated in Figure 9g. Also following this, TCP connections 364 between Clients A and B to Server S 354 could be dropped, but this is dependent on the applications running on Clients A and B.

- 5 Figure 9h shows how the solution of Figure 9g does not work when one of the NATs is asymmetric. More specifically, If NAT B 358 is asymmetric then UDP packet traffic from port address B:b 382 directed at network translated address A₁:a₁ 370 will appear to come from B₂:b₂ 392 but will still be received by Client A over a second UDP channel 390. However, UDP packet traffic from network translated address A₁:a₁ 370 to network
10 translated address B₁:b₁ 384 will be blocked by NAT B 358 as that port is no longer being used for communications to Client B 356.

- Figure 9i shows the solution provided by the present embodiment of how to deal with one of the NATs being an asymmetric NAT. More specifically, in Stage 6B Client A sees UDP
15 packet traffic coming from network translated address B₂:b₂ 392 rather than network translated address B₁:b₁ 384. Accordingly, NAT A 352 switches its UDP traffic output intended for network translated address B₁:b₁ 384 to network translated address B₂:b₂ 392. NAT A 352 can do this because NAT B 358 has used this address to write to network translated address A₁:a₁ 370 of NAT A 352 and so IP passes. In this way bi-directional
20 communications are established even when an asymmetric NAT is present.

If NAT A 352 is asymmetric and NAT B 358 is symmetric then the same process described above in relation to Figure 9i occurs in reverse.

- 25 If both NAT A 352 and NAT B 358 are asymmetric, then pure peer-to-peer communications are not possible. Traffic continues or is re-established via the Server S 354 acting as an intermediary. Note that any other address visible to both NAT A 352 and NAT B 358 can be substituted as the intermediary. This manner of dealing with this situation is described in detail later with reference to the second embodiment.

A second embodiment of the present invention is now described with reference to Figures 10 to 16. The second embodiment is similar in many respects to the first embodiment and so the following description will concentrate on the differences between the first and second embodiments. Some aspects of the first embodiment, which have only been described
5 briefly, will be expanded on in the description of the second embodiment for completeness.

The secure communications system 400 according to the second embodiment is distributed and can be complex due to its size. In one way, its overall architecture can be considered to be a hierarchical tree 402 as is set out in Figure 10. Starting at the very highest viewpoint,
10 the system 400 appears at three logical levels: the system administration level 404, the transport server level 406 and the end point level 408. The boxes shown in Figure 10 represent logical functions, which may actually be realised in a plurality of very different ways.

15 The top level 404 of the tree 402 is concerned with Administration functions of the system 400. This level 404 holds a database 410 (see Figure 12) of all computers permitted to use the system 400, and controls whether or not individual peer-to-peer communications are allowed to be set up.

20 Security is also controlled from this administration level 404. Individual communications connections at lower levels must be authorised from the Administration level 404; once so authorised, lower administration levels may negotiate encryption keys for example between themselves prior to creating additional connections.

25 The administration level 404 is also informed, to a customisable level of detail, of any additional connections that are created and may also record events that occur at lower levels. These events include at the very minimum, each attempted access to the network.

The administration level 404 is also capable of producing management reports concerning
30 the operation of the network, which is very useful for control and management of the

system 400.

At the lowest level 408 are the “end points” 412. These include “softphones” (software for handling VOIP calls) and localised e-mail servers. However, the end points 412 can also include other forms of data communications tools. The end points 412 are the key components that get installed on users’ computers to enable access to the system 400 and the secure method of communicating provided thereby. Each end point 412 is inherently secure as it is provided at the user’s location, therefore storage of communications at end points 412 does not compromise the security of the data communications in any way.

A so-called distributed “transport layer” 406, made up of a plurality of transport servers 414 is provided between the administration level 404 and the end points 412. Whilst this layer 406 effectively remains invisible to the users and administrator, it is an essential component of the system 400 in that it facilitates transport of two types of communications, namely address and status determination communications and actual peer-to-peer data communications. The end points 412 and administrative layer 404 only communicate via the transport server 414 that comprise the transport layer 406. This provides the advantages of simplicity and scalability.

Referring now to Figure 11, the administration level 404 is now described in greater detail. The administration level 404, in the example shown in Figure 11, has a hierarchical tree-like structure, which enables efficient management of control of the system 400. The distributed nature of the administration layer is evident from Figure 11 and it can be seen that at the top of the hierarchy is the Amteus Global administration node 416, which runs from a single central location. Then the client administration nodes 418 are provided, with only one shown in the Figure 11 example, which may be implemented on, for example, the computers of corporate or governmental bodies (not shown) which may wish to control communications between its registered users. The client at the client-specified location runs the single client administration node 418 shown in Figure 11.

The next layer in the hierarchy is the layer 420 of regional administration nodes 422 for a given client of which three are shown in Figure 11. These regional administration nodes 422 connect to the end points 412 (EP 1 to EP 3) via the distributed transport layer 406 (only represented as a dotted line). This lowest level of administration 420 is geographically based and is important for several issues including load balancing which is described in detail later. Similarly, in this example, a further end point EP4 412 has its administration functions implemented directly at the Amteus Global administration node 416 via the transport layer 406.

The way in which the administration level 404 works is now described and illustrated with an example. Each user, represented by an end point 412, is registered at one administration node 416, 418, 422, typically its regional administration node 422. This registration is communicated up the hierarchical administration tree and as a result the registration details are held at all administration nodes 418, 416 at higher levels. The registration information that is stored at these administration nodes 418, 416 includes all details about the user, as collected during a registration procedure, together with the current operating state of the end point 412. Any decisions as to whether to permit peer-to-peer communication between end points 412 are taken at the lowest level in the hierarchy at which both users registration details are available. Any resulting changes in operating state of these end points 412 are then passed to administration nodes at higher levels.

The EP (End Point) 412 is typically a telephone and/or e-mail user. In the example shown in Figure 11, EP1 412 is based in the Client's London Office. His details are stored in the London regional administration node 422, and at all administration nodes 416, 418 at higher levels. EP2 and EP3 412 are based in the Client's Athens Office. Their details are stored at the Athens regional administration node 422, and at all administration nodes 416, 418 at higher levels. EP4 412 is a private Amteus subscriber, known only to Amteus and as such he is registered only with the Amteus Global Admin node 416.

Assuming EP2 412 wants to communicate with EP3 412, the Athens regional

administration node 422 knows both end point users 412 and the request may therefore be handled locally in Athens by the Athens Admin node 422. Details may be sent to administration nodes 416, 418 at higher levels for information purposes, but these higher-level nodes 416, 518 are not involved in the setting up the desired communication channel.

5

If EP1 412 wishes to communicate with EP2 412, he will make his request to his local Administration node 422 in London. This London regional administration node 422 knows nothing about EP2 412, so it changes the status of EP1 as being in a Call Setup and passes the request up the hierarchical tree to the Client Administration node 418.

10

The client administration node 418 knows about both parties to the call, so it can handle the desired call setup. It knows whether EP2 416 is busy as it has the latest status information to hand, but there is a slight chance that its view on this is out-of-date and this is catered for as is described later. If the client administration node 418 believes EP2 416 to be busy, it so informs the London regional administration node 422, which in turn informs EP1 412. The request for set up thus ends in failure.

15

Otherwise, the client administration node 418 also marks the EP1 status as being in a 'Call Setup', and passes the request down to the Athens regional administration node 422. The Athens regional administration node 422 may know that EP2 416 is now busy (i.e., the status at the client administration node 422 was actually out of date). In this case, it returns a "busy", failure code to the client administration node 418, which then cancels the request for setting up the communications channel as in the previous case.

20

Otherwise, the Athens regional administration node 422 forwards the call request to EP2 412. EP2 412 decides whether or not to accept the call and passes the appropriate response back along the route to EP1 412 in reverse.

25

If EP1 412 wishes to communicate with EP4 412, a similar process occurs. In this case, the initial request is passed all the way up to the Amteus global administration node 416 via its

30

regional administration node 422 and the client node 418. The Amteus global administration node 416 then communicates directly with EP4 412 to set up the connection.

5 It is to be appreciated that the term 'locally' as used above is intended to mean a geographical locality to the server. However, in the strictest sense, this term simply means that there is a relatively direct connection between the server and the end point and that the end point is registered with that server, which makes that end point then local to that server.

10 Referring now to Figure 12, the transport level 406 of the hierarchy 402 of Figure 10 is made up of transport servers 414 connected to each other in a hierarchical tree structure. Each transport server 414 is responsible for facilitating the transportation of a communication from one location (end point) 412 to another without storing the message en-route. At the top of the hierarchical tree, a main transport server 424 is provided and this is connected to the Amteus global Administration (Database) Server 410, which tops the administration hierarchy. Also a hot standby transport server 426 is provided as a back up to the main transport server 424 in case of failure. Three regional transport servers 1, 2 and 3 428 are shown connected in the next level 430 of the hierarchy to the main transport server 424. These regional transport servers 428 in turn are connected with respective End Points 1.1, 1.2, 2.1 and 3.1 412.

20

The above-described transport level 406 represents a dynamic structure, which is not predefined. Rather, rules are defined and provided for connecting together the transport servers 414 dynamically to form the network. Furthermore, the administration level 404 which is functionally shown as a different layer in Figure 10, is actually incorporated into the hierarchical network of transport servers 406 and is utilised in the setting up of a peer-to-peer communication. Finally, the network has a built-in security overlay, which improves network security. The overlay is implemented in this embodiment as a functional rule which applies to all nodes of the network, namely: a node is not permitted to communicate to an end point 412 unless that end point 412 is known to that node. When the node does not know personally of the identity of the desired end point 412, it passes up the

25

30

request for communication further up the hierarchical tree.

Referring to Figure 13, each transport server 414 has a client link (uplink) 430 to a higher-level transport server 414 and/or a client link 432 to an Administration (database) server 410 (both have been shown for convenience in Figure 13). Note that when the transport server 414 is at the top of the hierarchical tree, the uplink 430 is to the Administration database server 410, whereas otherwise it is to a higher-level transport server 414. At the bottom of the transport server tree are end-points 412 as described above in relation to Figure 12. Each transport server 414 also has a downlink 434: a server connection, by which lower-level transport servers 414 and end-points 412 may initiate connections to the server 414. The uplink 430 on each transport server 414 is a TCP/IP client; and the downlink 434 is a TCP/IP server.

Each transport server 414 runs in an environment not subject to network address translation. This means that each transport server 414 has its own public IP address 436, which may be dynamically allocated as well as its own transport server ID 438. These are stored in its local database 440 together with list 442 of end point IDs that are connected to that particular transport server 414, the status 444 of the connections to each of those end points 412 and the permissions 446 associated with each of the end points 414. A database server 448 is provided to access and update this information as required. The lists of end point (user) information stored in the database 440 are arranged as described below.

Two lists are provided one in the database 440 which is a compilation of the IDs of locally registered users who are currently on-line 444 and the other which is a compilation 450 of the IDs of those locally registered end points 412 which have watches recorded against them. A watch is recorded when a particular end point 412 is connected but not available for communication, for example when it is busy. The watch monitors the status of the destination end point 412 and when it becomes available for communication, a trigger mechanism (not shown) in the watch triggers off a notification process. The notification process sends out messages to all of the end points 412 and transport servers 414 that have

recorded an interest in watching the change of status of the connection to the desired destination end point. Furthermore, the change of state is notified up to higher levels in the hierarchical network.

- 5 The database 440 also stores a set of public encryption keys 451 for the end points 412. These as will be described in greater detail later are used to decrypt encrypted messages sent from different end points 412.

10 The transport server 414 also comprises a location module 452 for determining the current geographical location of the transport server 414 and a heartbeat monitor 454 for checking its connections to different but adjacent transport servers 414 in the hierarchical network. A server control module 456, together with the database server 448, manages the operations of all of these links, modules and databases.

- 15 The transport servers 414 of the network must be authorised by the Administration System 404 (shown in Figure 11) exactly as must end points. As mentioned previously, each transport server 414 has its own entries in the database 440, with permissions to act as transport servers 414, and have User IDs.

- 20 Transport servers 414 are "location aware" by the functioning of the location module 452. This can be achieved in a plurality of different ways. For example, one simple way is for the location module 452 to be windows based, where the current time zone of the system on which the location module is running can be determined.

- 25 Using the heartbeat monitor 454, all transport servers 414 send out regular heartbeats to their parent transport servers 414 (if any). A heartbeat is a signal which contains update information regarding the transport server's registered connections (loading) and which confirms the existence of a communications link between the parent and child transport servers 414. The purpose of the heartbeat monitor 454 is to control load balancing of the
30 system 400. More specifically, the heartbeat information contains a count (usage count) of

the number of end points 412 currently registered directly at the originating (regional) transport server 414. On receiving a heartbeat, each (parent) transport server 414 responds with a message giving the list of the IP addresses and ports of the downlink, plus the usage count, of all the other transport servers at the same level. Thus, (in Figure 12) when the main transport server 424 receives a heartbeat from transport server 414, it replies with each of the regional transport server 1 and 3's 412 respective addresses and usage counts.

Referring now to Figure 14, a flow diagram is shown of the load balancing procedure 460 of this aspect of the present embodiment. As can be seen, when a new client (end point 412 or lower-level transport server 414) seeks to connect at to a transport server 414, a request is received at Step 464 by the transport server 414, the request containing the new client's heartbeat signal. The transport server 414 then retrieves at Step 466 its own stored usage count U_1 and the usage counts U_2 U_3 U_4 of other transport servers 414 at the same level and geographically at least as close (for example in the same time zone). The transport server then compares at Steps 468 and 470 its usage count U_1 against those U_2 U_3 U_4 of the other proximate transport servers 414 at the same hierarchical level. If its usage count U_1 is much higher than the usage counts U_2 U_3 U_4 any of its peers 414, a reply is sent back at Step 472 to the new client telling him to connect to the transport server 414, which currently has the lowest usage count, to effect the peer-to-peer communication. The connection with the current transport server 414 is then terminated at Step 474 and the load balancing procedure 460 ends at Step 476. Otherwise, the request for connection is accepted at Step 478, the client's user count is read at Step 480 and the local usage count of the transport server 414 is incremented at step 482 by the amount of the client's user count. Subsequently the load balancing procedure ends.

It is possible to extend this basic principle to further more refined levels of sophistication. For example, once the receiving transport server 414 has determined that at his hierarchical level and at his geographical location, it is able to accept the request, it may then examine whether there are any lower-level transport servers 414, which are geographically close to the new connection and are better capable of handling it. This is considered advantageous,

as it would aid overall system performance if users were connected at the extremities of the transport server tree where possible. This is because the majority of peer-to-peer communications are voice over IP traffic, which is heavy on bandwidth and also usually local. Therefore by keeping this main traffic at a local transport server level the whole hierarchical system 406 does not become burdened with this traffic, which would otherwise slow the system 400 down considerably.

Any transport server 414 accepting a new connection informs its new child user of its own parent transport server's address. It also informs its parent transport server 414 of the new user's connection. The parent transport server 414 thereafter considers the new user to be connected to it.

Once connected, a new user 412 is able to initiate send messages to its transport server 414 in the following categories:

- Admin Request

The transport server 414 passes such requests to its parent transport server 414, after tagging them with its own transport address/port 438.

- Message destined for the connected transport server 414. These are sent in the conventional manner over TCP/IP, and require no special handling.

- Message destined for another end-point 412 or transport server 414 other than its parent. The message contains the UserID for the source and destination end points. The transport server 414 looks up the UserID in the list of currently connected users (end points). This list is created by looking at list 442 of users registered with that transport server 414 and looking at their respective statuses 444. If it finds the user, then the message is forwarded to that user, possibly via one or more lower-level transport servers 414. If the user is not found, the message is passed to the transport server's parent transport server 414. If the top of the transport server tree 424 is reached and the destination is not found, failure is reported back to the source. This process is effectively sending a reply message back from the top-of-tree user ID to the original

message source. The process of checking the data stored at each transport server's parent 414 relies on the fact that each registration of a user is passed back up the tree to the parent of each node such that the top of the tree has a picture of all the connections. Accordingly, if the desired user is not found at the top of the tree 424, they cannot have registered with the network.

It is to be appreciated that when an end point 412 loses a connection to its direct parent transport server 414, it attempts to connect to its grandparent transport server 414. The grandparent transport server 414 may then find an alternative connection at the original level.

In the end point level 408 of the hierarchy 402 of Figure 10, each end point 412 has a single client TCP/IP connection to their transport server parent 414. This connection is used for access to both higher-level transport servers 414 and to administration servers 410, 404.

Each end point 412 provides a communication interface (not shown) with the user to facilitate VoIP communications, for example. In this embodiment, each end point 412 also uses a so-called "Multimedia" engine (not shown). This multimedia engine comprises microphones, speakers or headphones, WaveFile players and recorders and video players and recorders. The multimedia engine also incorporates a digital signal-processing module, allowing audio to be translated between different wave formats, attenuated, amplified and mixed. The multimedia engine is not described further as the functions of the multimedia engine are commonplace within the art and the skilled addressee will need no further explanation to construct such an engine.

Referring now to Figures 15 and 16, a method 500 of establishing a peer-to-peer connection using the above-described system 400 is described below. The method 500 is essentially a two-stage process where in the first stage (described in Figure 15), it is established whether it is possible to set up the desired peer-to-peer connection between the specified end points 412, and in the second stage (described in Figure 16) a peer-to-peer connection is set up

between the source 412 and the desired destination 412. Once a peer-to-peer connection has been established, direct two-way communications between the source 412 and destination 412 are possible in a secure manner, using standard Internet communications protocols; in this embodiment over a dedicated UDP channel 372. The following description relates to

5 the first and second stages described above.

Peer-to-peer (P2P) connections may be set up between end points 412. These connections are then, in the current embodiment, used to transmit audio and email data but other forms of communications can also be used.

10

The need for a new P2P connection first arises at an end point 412 – for example, when an end user decides to make a VoIP telephone call to another end user. At this stage, the originator of the connection (source) knows only the network address of the desired destination (the target user's ID); he does not know the status of the target user, namely

15 whether the user is online, busy etc. (He may know that the target user was recently online etc, through his address book, but will never be absolutely certain of the current state).

Referring now to Figure 15, the method 500 of implementing the first stage commences with the source user 412 sending at Step 502 a request to its transport server 414, of the

20 form "I (User S) wish to communicate with User D". The transport server 414 looks in its list 442, 444 of connected users 412 and checks at Step 506 to see if User D is found. If it finds User D 412 connected, it checks at Step 508 whether the user D 412 is available. If not, it returns at Step 510 a "Failed – Busy" response and preferably sets up at Step 512 a watch against User D; finally User S waits pre-determined time period at Step 514 and then

25 retries to connect again and/or simply awaits notification of the watch being triggered at the current transport server in response to a change in User D's availability status. However, if User D is available as determined at Step 508, then the transport server 414 changes at Step 516 the status of both User S and User D to be in a "Call Setup in progress" state. The transport server 414 then forwards at Step 518 the request to User D possibly via other

30 transport servers 414 at lower levels than the current transport server 414. At this point User

D is considered at Step 520 to have been found and connected to and the process 500 continues as is described below in relation to Figure 16.

If User D subsequently has a change of status in that he becomes available for communication, this is notified to User D's local transport server 414 and the watch set up there for User S is triggered. User S is then notified at Step 514 of the availability of User D and the first stage can commence once again as has been described above. Alternatively, or in combination, User S can wait at Step 514 a predetermined amount of time before restarting the process 500 from scratch as has been indicated in Figure 15.

If User D is not currently connected to the Transport Server (TS) as determined from the check carried out at Step 506, a check at Step 522 is carried out to determine if a parent Transport server 414 exists. If a parent does exist, the request is forwarded at Step 524 to the parent TS 414. Thereafter, the parent transport server 414 is considered at Step 526 to be the current transport server and the process continues with the current transport server 414 checking at Step 504 its list of registered end users 412. If there is no parent TS 414 as determined by the check carried out at Step 522, the transport server sends at Step 528 a reply back to the end point indicating a failure to connect and the process ends at Step 530.

A call request arriving at a transport server 414 from a higher-level transport server 414 is handled in exactly the same way except that, in the event that the user is not connected, a failure is returned rather than forwarding the request back from whence it came.

Referring now to Figure 16, the second stage of the communication process 500 is now described. Once User D has been connected to, it has to convey its current IP address and port number back to User S such that it can be connected to directly for facilitating a peer-to-peer communication. Accordingly, if User D is still online and available when it receives the call from User S, it too changes at Step 532 its state to "Call Setup". User D then sends at Step 534 a RINGING1 message to its parent, which will always be a transport server 414.

The RINGING1 message also contains the User ID of the call originator (User S), so that

intermediate transport servers 414 can route it back using the same mechanism as in the original call request.

On receiving at Step 536 a RINGING 1 message, a transport server 414 first constructs also
5 at Step 536 a UDP (User Datagram Protocol) channel 372. It then constructs at Step 540 a RINGING2 message, containing the external IP address and port number of the UDP channel, and sends also at Step 540 this message both to the end point User D and to the call originator end point User S. The RINGING2 message also contains the User ID of the transport server 414, which received the RINGING1 message, since this will be used to set
10 up the peer-to-peer connection.

It is to be appreciated that a transport server 414 never sends a RINGING1 message. Hence a RINGING1 message arriving at a transport server is always considered to have come directly from an end point 412.

15

When an end point 412 receives at Step 542 a RINGING2 message, it checks at Step 544 that it is in Call Setup state. If it is not, then it resets at Step 546 all the named parties to the call and the process 500 ends at Step 530. Otherwise, each end point 412 sets up at Step 548 a UDP channel and starts to send at Step 550 messages periodically to the port specified in
20 the RINGING2 message. This will be the transport server 414 closest to the Callee. Unless there is some general network failure, most of these messages should arrive correctly.

When the transport server 414 closest to User D receives the first UDP message from each end point 412, it also extracts at Step 552 the (possibly translated – described later) network
25 address and port number of the end point's UDP channel 372. The transport server 414 is then able to send (not shown) replies down each end point's UDP channel 372, which includes the UDP network address and port number of the other end point's UDP channel, and these should arrive even at each end point if their firewalls are present. (Normal firewall action at the end points should allow this since the end points initially wrote to the
30 transport server.)

Once the transport server 414 has received packets from both end points, it knows how their UDP ports are addressed externally. The transport server 414 then sends at Step 554 a TALKDIRECT message to each end point 412 telling it of the other end point's address.

5 Once these UDP addresses have been received at Step 556, the end points 412 are able to communicate with each other with the source end point 412 setting up at Step 556 a direct UDP communications channel 372 to the destination end point 412, and the process 500 then ends at Step 530.

10 In a significant proportion of cases, the above procedure enables such direct communications to be established without difficulty. However, where network address translation (NAT) in combination with some firewall action at the end points exists, problems can arise. These problems can be overcome in many situations as has been described previously with reference to the examples shown in Figures 9a to 9i of the first
15 embodiment.

The approach, which is used for both the first and second embodiments, is described in further detail below and a mechanism for handling all the different combinations of NATs 352, 358 is also described.

20

Network Address Translators (NATs) 352, 358 are broadly categorised into two classes of translation function: Symmetric and Asymmetric. Symmetric NATs use the same Internet-side address and port combination regardless of the destination of a packet; Asymmetric NATs use different Internet-side address/port combinations for each destination. Whilst it is
25 not exactly clear what the reasoning for this is, it is considered that it is based on a belief that it is wise to keep out data packets from unknown sources.

A bit of thought regarding the necessity for these NATs indicates that:

1. The recipient can see from whom each packet comes, and can easily reject those from
30 an unknown source if required without the requirement for a NAT.

2. The IP address and port in an Internet packet are easy to fake. Such draconian action by NATs is unlikely to deter a reasonably competent hacker for very many minutes.

Firewalls can also cause a problem when seeking to establish a peer-to-peer communication. The normal firewall action is to impose a “don’t speak until you’re spoken to” rule. If A and B want to talk to each other, either A or B has to speak first. The other can then reply. But, if A speaks first, and B is behind a firewall, B will not see the packet until it has written to A.

- 10 Moderately sensible firewall behaviour would dictate that the rule would be implemented as stated above. Many firewalls actually interpret the rule as “He who speaks before he is spoken to will never be listened to again”. Such behaviour can cause severe problems in trying to establish a peer-to-peer communications channel.
- 15 There are several procedures carried out in the above-described embodiments, which ameliorate the situation. Firstly, it has been appreciated that an asymmetric NAT will not normally open an *arbitrary* port to write to a new destination; frequently it will be the original port + 1 (or 2, 3, etc. according to how many times it has occurred in between the first and second writes). In the present embodiment, when an asymmetric NAT is
- 20 encountered, a few consecutive ports above the port address stated in the RINGING2 message are written to, to cover this possibility.

Some manufacturers indicate that Firewalls/NATs commonly implement a uPnP (universal Plug ‘n’ Play) interface by which their actions may be interrogated and, subject to security

25 constraints, amended. The present embodiment includes a uPnP interface (not shown) which has been implemented in the transport servers 414.

In a proportion of cases, it is simply not possible to implement a direct peer-to-peer connection either because of severe firewall implementations or because both of the NATs

- 30 352, 358 are asymmetric. In these cases, the present embodiment is arranged to ‘bounce’

out messages off a local transport server 414 that acts as a trusted intermediary. This mitigates issues of unknown sender as the local transport server 414 will always be known to the local asymmetric NAT 352, 358 such that even the most severe rules at a firewall, for example, will not prevent such indirect pseudo peer-to-peer communications. Furthermore, the possible reduction in security of such an indirect communications channel is minimised by ensuring that the bounced out packets are forwarded by the transport server 414 acting as the trusted intermediary with minimal analysis, and certainly no storage.

In the case where the idiotic (very severe) interpretation of firewall action is encountered, the present system is arranged to implement a true peer-to-peer communication in one direction and to bounce off a local transport server 414 in the other direction. This variation is worth doing because it minimises the reduction in security. For example, in the case of e-mail transmission, it is advantageous to arrange that the e-mail is sent directly, and that the acknowledgements are bounced back.

The actual way in which the system 10 automatically discovers that asymmetric NATs 352, 358 must be present or a severe firewall is in place is described below. From the high-level viewpoint, when the local transport server 414 knows the addresses of the UDP ports of both end points 412, as has been described above in relation to Figure 16, it sends the TALKDIRECT message to each, giving them the other end point's address. The end points 412 attempt to communicate directly, and also set up a timer (not shown). If packets are received directly, the transport server 414 is told the connection is talking, and the timer is stopped and used no further. However, if the timer fires (namely a predetermined period of time lapses) with no packets having been received, a TALKTHROUGH message is sent to the transport server 414 and thence to the other end point 412, indicating that further communications are to be sent directly to the local server 414 for the intended destination end point 412. Messages are thereafter "bounced" off the transport server 414 that is local to the desired end point 412.

The system 10 commences setting up the peer-to-peer connection before the callee accepts

the call. If the call is not accepted, then the peer-to-peer connection will be closed immediately. The present system 10 starts the process only when the call is accepted; where the timeout / TALKTHROUGH method is implemented, this can lead to an awkward delay in establishing the voice link, for example. It is also the case that the majority of calls / e-mails, where a user is logged in, will be accepted.

A third embodiment of the present invention is now described with reference to Figures 17 to 20. The third embodiment is very similar to the second embodiment except for the issues relating to security. More specifically, the third embodiment substitutes the permissions 446 stored at each node with a more robust security system as described below.

Prior art insecure communications systems are unusable – they acquire unwanted users who spoil it for the registered users. Known insecure products have a natural ceiling in terms of the number of users, by virtue of the fact that as more people use the system, the profile is elevated thus pulling in the attackers who undermine the “popular” system itself. This effect works a little like negative feedback in control systems.

The third embodiment attenuates this cycle by building in security from the beginning. Attackers can never be eliminated from the system 10 entirely, but there is enough effective security built in to prevent their presence from undermining the availability of the system 10 to registered users. It is to be appreciated that designing a system that completely eliminates attackers would actually result in bankruptcy for the designers.

The security requirements of the third embodiment are listed below:

Requirement 1: NODEs must be authenticated to the system 10.

Requirement 2: Communication over NODE to TS links must be confidential.

Requirement 3: CLIENTs must be authenticated to each other (mutual authentication)

Requirement 4: Communication over CLIENT to CLIENT connections must be confidential.

Meeting Security Requirements 1 & 2

Much of the terminology that follows is taken from the RFC 1510, the Kerberos authentication system. Kerberos is an authentication mechanism that is used to verify user or host identity, it is also the preferred authentication method for services in Microsoft Windows Server 2003. The Kerberos authentication protocol provides a mechanism for mutual authentication between a client and a server, or between one server and another, before a network connection is opened between them. The protocol assumes that initial transactions between clients and servers take place on an insecure communications network. Such an insecure environment could well be exemplified by the Internet, where an attacker can easily pose as either a client or a server, and can readily eavesdrop on or tamper with communications between legitimate clients and servers.

The Kerberos technique relies heavily on an authentication technique involving shared secrets. The basic concept is quite simple: If a secret is known by only two people, then either person can verify the identity of the other by confirming that the other person knows the secret. The password is kept secret by using secret key cryptography. Rather than sharing a password, communication partners share a cryptographic key, and they use knowledge of this key to verify one another's identity. For the technique to work, the shared key must be symmetric i.e. a single key must be capable of both encryption and decryption. One party proves knowledge of the key by encrypting a piece of information, the other by decrypting it.

The basic principle is implemented by authenticators, which can work as follows. A simple protocol that uses secret key authentication begins when someone is outside a communications door and wants entry. To gain access, this person presents an authenticator in the form of a piece of information encrypted in the secret key. The information in the authenticator must be different each time the protocol is executed; otherwise an old authenticator could be reused by anyone who happens to overhear the communication.

On receiving an authenticator, the person guarding the door decrypts it and knows from what is inside whether decryption was successful. If it was successful, the doorkeeper knows that the person presenting the authenticator has the correct key. Only two people have the correct key; the doorkeeper is one of them, so the person who presented the authenticator must be the other.

If the person outside the door wants mutual authentication, the same protocol can be executed in reverse, with a slight difference. The doorkeeper can extract part of the information from the original authenticator, encrypt it in a new authenticator, and then give the new authenticator to the person waiting outside the door. The person outside the door can then decrypt the doorkeeper's authenticator and compare the result with the original. If there is a match, the person outside the door will know that the doorkeeper was able to decrypt the original, so he must have the correct key.

The basic principle of how the Kerberos technique is used in the present embodiment is illustrated in Figure 17. Here a Server 602, a Client 604 and a Key Distribution Centre 606 are provided. The Client 604 wishes to connect to the Server 602 and first applies to the Key Distribution Centre (KDC) 606 to obtain a ticket for establishing a valid connection to the Server 602. The ticket is used effectively to authenticate the Client 604 to the Server 602, who is already authenticated to the system 600.

The acronyms listed below are used in the following detailed description of the security method of the present third embodiment which is now described with reference to Figures 18 and 19:

TS = Transport Server 414.

NODE = Combined CLIENT and TS. Sometimes referred to as simply a CLIENT, or simply a TS, in which case the other function is available but diminished in terms of the particular role being described for the NODE 608.

AS = Authentication Server 610.

KEY = Value based on the password assigned to a NODE 608.
 TGS = Ticket Granting Server 612
 REALM 614 = Community of NODEs 608 sharing an AS 610.

- 5 Firstly, it is assumed that an Authentication Server 610 associated with a particular REALM 614 containing a number of NODEs 608 is available. The AS 610 stores the equivalent of a {USERNAME, PASSWORD} doublet for each registered NODE 608 in its local database 616. Secondly, it is assumed (as shown in Figure 19) that the client at NODE_A 608 wishing to connect to the network, has used an Internet browser 618 to connect to a web
 10 server 620 of the KDC 610 to register and thereby obtained a username "UserA" 622 and a Password "pwdA" 624.

The basic mechanics of the procedure are described with reference to the following acronyms:

- 15 NODE_A = NODE 608 with username A.
 TGT_A = Ticket Granting Ticket 626 for NODE_A.
 S_A = Session Key 628 between AS 610 and NODE_A 608.
 K_AB = Shared Key 630 between NODE_A and NODE_B.
 TICKET_AB = Ticket 632 to allow NODE_A to gain access to NODE_B.
 20 {X}K_Y = Value X encrypted by the key, K_Y.

Referring now to Figures 18 and 19, the authentication process commences with NODE_A 608 running up and being authenticated to the system 600 as follows:

- 25 1. NODE_A 608 connects to a TS 414 and sends an AS_REQ 634 message to the AS 610 for the REALM 614 that NODE_A 608 belongs to (typically the message passes up through the hierarchical network until a TS 414 is found with a connection to the AS 610).
 2. The AS 610 generates a session key S_A 628 and replies with an AS_RESP
 30 message 636 containing TGT_A 626 and S_A 628 both encrypted with the pwdA

624.

It is also possible for the AS 610 to reply with an encrypted TIMESTAMP (not shown) for additional authentication security. This assumes that the AS 610 receives a reliable source of time through an Network Time Protocol (NTP) or Simple NTP connection, and that
5 NODEs 608 are synchronised to the AS's time when they authenticate to the system 600. NODEs 608 have to associate the AS-supplied TIMESTAMP with their current Win32 tick count, and calculate their new current time whenever they need to generate their own TIMESTAMP value.

10

Once NODE_A 608 has been authenticated, it is assumed that it is informed by the network of the name of the TS 414 that it must now "link" to in order to join the network (i.e. be able to make and receive VoIP calls), and that the target TS 414 is called NODE_B 608. Where timestamping has been used, the sequence is as follows:

15

1. Send a TGS_REQ message 638 to AS 610 containing TGT_A 626 and the name of NODE_B 608.
2. AS 610 replies with a TGS_RESP message 640 containing K_AB 630 (encrypted under S_A 628) and a TICKET_AB 632.
- 20 3. NODE_A 608 extracts the value of K_AB 630, calculates a current TIMESTAMP value and generates its authenticator to NODE_B 608 as {TIMESTAMP}K_AB. NODE_A 608 sends an AP_REQ message 642 to NODE_B 608 containing {TIMESTAMP}K_AB and TICKET_AB 632 to NODE_B 608.
4. For NODE_B 608 to allow NODE_A 608 to link to it, NODE_B must decrypt
25 TICKET_AB 632 to extract K_AB 630 and then decrypt {TIMESTAMP}K_AB to be able to verify the TIMESTAMP. Assuming this is successful then NODE_B 608 replies with an AP_RESP message 644 containing {TIMESTAMP + 1}K_AB which allows NODE_A 608 to authenticate NODE_B 608.

30 It will be readily apparent to the skilled addressee how the above can operate if

timestamping is not to be used and therefore an explicit description of how the above sequence of events is carried out without timestamping is not described further.

At this point, NODE_A 608 is now "linked" into the network via NODE_B 608 (in reality the CLIENT 604 in NODE_A is being served by the TS 414 in NODE_B 608).

The above implementation is based on an adapted sub-set of the Kerberos authentication system, which re-uses the basic protocols, messages and data structures but differs in functionality which makes the new authentication technique of the present embodiment effective for use in the context of peer-to-peer communications.

Meeting Security Requirements 3 & 4:

The following acronyms are used to describe this stage of the authentication process:

15	DH_PK_A	= Diffie-Hellman Public Key 646 for CLIENT_A.
	DH_SK_A	= Diffie-Hellman Private Key 648 for CLIENT_A.
	DH_PK_B	= Diffie-Hellman Public Key 650 for CLIENT_B.
	DH_SK_B	= Diffie-Hellman Private Key 652 for CLIENT_B.
	K_AC	= Shared Key 654 between NODE_A and NODE_C.

20

Referring now to Figure 19, when CLIENT_A 604 in NODE_A 608, which is now connected to the network, wishes to call CLIENT_C 604 in NODE_C 608, then the present embodiment requires that the CLIENTs 604 authenticate each other and subsequent communications over any direct connection established between them (such as a peer-to-peer connection) is encrypted by a shared key K_AC 654.

25

Two types of possible CLIENT to CLIENT authentication are considered. The first of these is described as "Authentication-by-induction" and effectively piggy-backs on the chain of authenticated links that must exist between NODE_A 608 and NODE_C 608 before any connection between CLIENT_A 604 and CLIENT_C 604 can be established. The

30

procedure is as follows:

1. CLIENT_A 604 generates a Diffie-Hellman (DH) key pair and sends a message to CLIENT_C 604 that includes the value of DH_PK_A 646. The message is sent via the chain of authenticated links 656 that connects them (as part of a communications channel setup protocol).
2. On receipt of the message, CLIENT_C 604 assumes that it must have been sent by an authenticated source (although CLIENT_C 604 is not in a position to formally authenticate that source, hence the implied weaker label for this form of authentication, "authentication-by-induction").
3. CLIENT_C 604 generates a Diffie-Hellman (DH) key pair and sends a message to CLIENT_A 604 that includes the value of DH_PK_B 650. The message is sent back to CLIENT_A 604 via the same (although it could be via a possibly different) chain of authenticated links 656.
4. Both CLIENT_A 604 and CLIENT_C 604 are now in a position to independently calculate the shared key K_AC 654. Any party intercepting DH_PK_A 646 or DH_PK_B 650 will not be in a position to calculate K_AC 654 and impersonate CLIENT_A 604 or CLIENT_C 604.
5. CLIENT_A 604 and CLIENT_C 604 use K_AC 654 to generate bulk encryption keys for the Advanced Encryption System (AES), or equivalent, allowing confidential communications.

Overall, "authentication-by-induction" is not true authentication, but rather it equates to saying that:

"I have established a confidential call to an authenticated party, X, who say that their name is X, although I have no means of verifying this" (although for VoIP calls this is clearly less of a worry than for e-mails for example!).

It is also possible to use an AS 610 to generate a TICKET 632 that allows CLIENT_A 604

and CLIENT_C 604 to authenticate each other, but the temporal overhead in carrying out this operation as part of communication channel setup is excessive. Therefore, “authentication-by-induction” is used in preference to this alternative.

- 5 The second possible type of authentication is provided externally to the Amteus system 600, but is one with which the Amteus system 600 must interoperate. It is assumed that an externally provided Public Key Infrastructure (PKI) is available and that CLIENTs are Public Key Enabled (PKE). In these circumstances, the above protocol would be extended to an RSA cryptographic algorithm which would digitally sign the messages sent between
10 CLIENT_A 604 and CLIENT_C 604, such that strong authentication would be available (further discussion of PKI is not provided here as it is well understood in the art).

The present embodiment uses the less secure but more practical “authentication-by-induction” approach.

15

Authentication Server Replication

In the prior art Kerberos system, the management of Security Databases for authentication is kept as simple as possible, and the same principles are adhered to in the present embodiment. The characteristics of the present embodiment are:

20

- Attempt to maintain static data only – avoid storage of any session or call-related data (i.e. dynamic data)
- Maintain a master read/write copy for creating accounts, modifying account details and deleting accounts.
- 25 - Periodically replicate the master copy to a number of read-only copies.

Cross-realm Authentication

The present embodiment there can be a plurality of REALMS 614 and these are organised in a hierarchical fashion for the administration system 404. In this regard, authentication
30 between NODEs 608 in different REALMS 614 (so called ‘cross-realm authentication’) is

carried out.

Cross-realm authentication effectively piggy-backs onto the existing TGS (Ticket Granting Service) system 600 described above. If NODE_A 608 in REALM_A 614 wishes to link to
5 NODE_B 608 in REALM_B 614, then it is necessary for TGS_B 638, 340 to be registered with the AS 610 for REALM_A 608. In this way, NODE_A 608 simply requests at Step 634 a TGT 626 from TGS_A, which is sent to TGS_B in order to obtain a TICKET 632 to send to NODE_B 608 to effect the new link.

- 10 Even though the realms 614 are organised hierarchically (see Figure 21), typically it would still be necessary for two communicating realms 614 to register with each other, rather than registration being in one direction only.

Cross-realm authentication and authorisation is described in more detail below. However,
15 prior to this a brief overview of the NODE security operations used in the embodiment is set out as described with reference to Figures 20a and 20b.

In general terms, a NODE 608 is a collection of sub-systems, the relevant sub-set of which is illustrated in Figures 20a and 20b. Connections or links are made to/from a NODE 608.

- 20 The distinction is made here between a simple connection, which may be ephemeral and does not require NODE-NODE encryption, and a link which is a more permanent arrangement and has an associated encryption key 630, 654.

Figures 20a and 20b show two NODEs 608 both of which contain identical sub-systems
25 (i.e. they are different instances of the same executable). The parent NODE 608 (Figure 20a) is a primary NODE in that a Data Client (DC) 656 is connected to a Data Server (DS) 612 and the KDC 610 is active. The child NODE 608 (Figure 20b) does not have a DS connection, so the DC 656 and KDC 610 are present but inactive. The primary NODE 608 handles by use of its security services 660, all security requests received from its children.

The CLIENT 604 and TS 414 communicate with the SS 660 using handles (not shown). A handle is associated with each connection or link (i.e. each active server stream), whether a CLIENT connection/link 662 to a parent NODE 608 or a TS connection/link 664 from a child NODE 608.

5

When a NODE 608 starts-up the DC 656 will attempt to connect and login to a DS 658, if successful the KDC 610 will also be initialised. Note that a primary NODE 608 does not need to be authenticated to its own KDC 610, as the CLIENT 604 cannot link to another NODE 608 in the same REALM 614 – see later.

10

In almost all cases, a CLIENT sub-system will attempt to establish a link 662 to a parent NODE 608 (this is also true for a primary NODE 608, in which case the link is cross-realm – see later). The SS 660 is instructed to build security requests for sending to the parent NODE 608 as well as processing security replies received from the parent NODE 608.

15

Once a link is established to a parent NODE 608, the CLIENT 604 can make calls into the SS 660 in order to encrypt or decrypt link messages. If the CLIENT 604 receives a security message than an appropriate call is made into the SS 660; if this NODE 608 is not the target of the message then the CLIENT 604 routes the message to the TS 414 for passing further down the hierarchy until chosen destination NODE 608 is reached and the message

20

consumed.

Child NODEs 608 communicate messages to the TS 414. If the TS 414 identifies a message as a security message, then an appropriate call is made into the SS 660; if the call is unsuccessful or the target is further up the hierarchy, then the TS 414 routes the security message to the CLIENT 604 for passing up to the parent NODE 608.

25

Where an active (connected to a DS 658) KDC 610 is available at a NODE 608, because it is the primary NODE 608, security requests received from children at the TS 414 must be satisfied by this NODE 608 and not propagate higher up.

30

NOTE: The simplest implementation is for CLIENT 604 to assign some states for managing link setup. The SS 660 will NOT make callbacks into the CLIENT 604 or TS 414. Tight integration of sub-systems is assumed, such that CLIENT 604 and TS 414 must maintain states for management of security requests and replies.

5

A REALM 614 as has been described above is a set of registered users (similar to a Windows Domain), where each user is registered for that REALM 614 (typically via a web server 620) and has an account (not shown) maintained on the DS 612 for that REALM 614.

10

Referring now to Figure 21, a DS (Data Server) 612 holds all the authentication data for a particular REALM 614 and this data does not propagate outside the REALM 614 in which it is created. The primary NODE 666 is effectively the security root for a REALM 614 and security requests are not allowed to propagate further up the hierarchy.

15

To allow for the propagation of dynamic data up and down the hierarchy, some means is required for joining a child NODE 668 in one REALM 614 to a suitable parent NODE 670 in a "higher" REALM 614 - the nature of the tree topology is such that only one such link 672 is required. Figures 20 and 21 show the solution provided by the present embodiment, the primary NODE 666 for REALM A 614 is allowed to link to a parent NODE 670 in the "higher" REALM B 614. This link 672 is possible because an account is set-up in DS B 612 to allow the primary 666 of REALM A 614 to be authenticated to REALM B 614 and the link 672 created.

20

Because of the cross-realm link 672, it is now possible for call data to propagate from REALM A 614 to REALM B 614 and down to REALM C 614, allowing NODE X 608 to make a cross-REALM VoIP call 674 to NODE Y 608, for example.

25

Detailed Scenarios illustrating how cross-REALM linking is achieved are now described.

30

Primary NODE: Startup

It is assumed that the Data Server (DS) 612 is already running. A primary NODE 666 is run-up local to the DS 612, typically supplied with command arguments to allow it to log into the DS 612 and generate a key 628 for the KDC (a secondary NODE would be run-up with the same arguments). Any {username, password} supplied via the GUI is for logging into an account in a higher REALM 614.

1. Bootstrap sequence instructs the DC (Data Client) 656 to log into a local DS 612.
2. Bootstrap sequence calls procedure SecurityService::Initialise(). The KDC 610 generates its key and establishes a handshake with the DC 656.
3. User enters {username, password} to allow the CLIENT 604 to link to a parent NODE 670 (in another REALM 614).
4. The CLIENT 604 gets a handle for the new connection to the parent node 656 via procedure SecurityService::CreateSecurityContext().
5. The CLIENT 604 calls procedure SecurityService::BuildAsReq() to build an AS_REQ message 634 and then sends the AS_REQ message 634 to the parent NODE 670.
6. The CLIENT 604 receives an AS_REP message 636 and passes it to procedure SecurityService::ProcessAsRep().
7. The CLIENT 604 calls procedure SecurityService::BuildTgsReq() with the ID of the NODE it wishes to connect to, so the SS (Security Service) 660 can build an appropriate TGS_REQ message 638. The TGS_REQ message 638 is sent to the parent NODE 670.
8. The CLIENT 604 receives a TGS_REP message 640 and passes it to procedure SecurityService::ProcessTgsRep().
9. At this point the CLIENT 604 will either be promoting the existing connection to link status or reconnecting to a different parent NODE 670 (but in either case, the target NODE must be the NODE that was named in the procedure BuildTgsReq() call). CLIENT 604 calls procedure SecurityService::BuildApReq() to construct an AP_REQ message 642, which the CLIENT 604 then sends to the parent NODE 670.
10. CLIENT 604 receives an AP_REP message 644 and passes it to procedure

SecurityService::ProcessApRes(). On successful completion of this call, the SS 660 should have the necessary security context to support encrypt/decrypt operations on the link 672 to the parent NODE 670.

5 Non-Primary NODE: Startup

1. NODE 608 is not supplied with DS login arguments so the bootstrap sequence does not instruct DC 656 to log into a local DS 612.
2. The bootstrap sequence calls procedure SecurityService::Initialise(). KDC 610 is not supplied with parameters for generating a key and DC 656 indicates that it is not
10 connected to a DS 612.
3. The user enters {username, password} to allow the CLIENT 604 to link to a parent NODE in the same REALM 614 (the link may or may not be to a primary NODE 666).
4. The CLIENT 604 gets a handle for the new connection to the parent node via procedure SecurityService::CreateSecurityContext().
- 15 5. The CLIENT 604 calls procedure SecurityService::BuildAsReq() to build an AS_REQ message 634 and sends the AS_REQ message 634 to the parent NODE.
6. The CLIENT 604 receives an AS_REP message 636 and passes it to procedure SecurityService::ProcessAsRep().
7. The CLIENT 604 calls procedure SecurityService::BuildTgsReq() with the ID of the
20 NODE 608 it wishes to connect to, so the SS 660 can build an appropriate TGS_REQ message 638. The TGS_REQ message 638 is sent to the parent NODE.
8. The CLIENT 604 receives an TGS_REP message 640 and passes it to procedure SecurityService::ProcessTgsRep().
9. At this point, the CLIENT 604 will either be promoting the existing connection to link
25 status or reconnecting to a different parent NODE (but in either case the target NODE must be the NODE 608 that was named in the procedure BuildTgsReq() call). CLIENT calls procedure SecurityService::BuildApReq() to construct an AP_REQ message 642, which the CLIENT 604 sends to the parent NODE.
10. The CLIENT 604 receives an AP_REP message 644 and passes it to procedure
30 SecurityService::ProcessApRes(). On successful completion of this call, the SS 660

should have the necessary security context to support encrypt/decrypt operations on the link to the parent NODE.

Primary NODE: Receiving AS_REQ or TGS_REQ requests from child NODEs

5 Assume that the primary NODE 666 is initialised and has a KDC 610 that can accept security requests.

1. A TS 414 receives an AS_REQ message 634 from a child NODE 668. If the connection/link has a security context then procedure SecurityService:
10 :DecryptLinkData() is called to decrypt the message.
2. The TS 414 passes the AS_REQ message 634 to procedure SecurityService:
:ProcessAsReq(). To fulfil the request, the KDC 610 extracts the Username from the AS_REQ message 634 and uses it to request the (hashed) password held in the DS 612 (via the DC 656).
- 15 3. If successful, procedure SecurityService::ProcessAsReq() returns an AS_REP message 636. (NOTE: A security context is not needed when the SS 660 is processing requests – this is different to when the SS 660 is processing replies).
4. The TS 414 sends the AS_REP message 636 to the child NODE 668 that sent the AS_REQ message 634.

20

If a primary NODE 666 receives a TGS_REQ message 638 then the same procedure is followed (except that now the (hashed) password for the NODE 608 to be linked to is requested from the DS 612).

25 Non-Primary NODE: Receiving AS_REQ or TGS_REQ from child NODEs

1. The TS 414 receives the AS_REQ message 634 from a child NODE 668. If the connection/link has a security context then procedure SecurityService:
:DecryptLinkData() is called to decrypt the message 634.
2. The TS 414 passes the AS_REQ message 634 to procedure SecurityService:
30 :ProcessAsReq(), which returns a suitable error code to indicate that an active KDC

610 is not available.

3. The TS 414 routes the AS_REP message 636 to the CLIENT 604 for passing to the parent NODE 670.

5 Primary or Non-Primary NODE: Receiving AP_REQ

Any NODE 608 receiving an AP_REQ message 642 must successfully process it locally or return an error message, it cannot pass it on.

- 10 1. The TS 414 receives AP_REQ message 642 from a child NODE 668. An existing security context, and therefore link, should not be available on this connection, so it should not be necessary to decrypt the message.
- 15 2. The TS 414 passes the AP_REQ message 642 to procedure SecurityService::ProcessApReq(), which uses its local NODE password (and not the KDC 610) to process the message 642 and generate a suitable AP_REP message 644 for returning. If the procedure SecurityService::ProcessAsReq() call succeeds, then a full security context is now established and the connection to the child NODE 668 is promote to link status.
3. The TS 414 returns the AP_REP message 644 to the child NODE 668.

20 Primary or Non-Primary NODE: Receiving AS_REP or TGS_REP from a parent NODE

If the CLIENT 604 in a primary NODE 666 receives an AS_REP message 636 or a TGS_REP message 640, then the primary NODE 666 must be the source of the original AS_REQ message 634 or the TGS_REQ message 638, this has already been discussed above for start-up cases (there should not be an existing security context).

25

- If a NODE 608 receiving an AS_REP message 636 or the TGS_REP message 640 is not a primary NODE 666, then the requests are routed to the TS 414 for passing down to the appropriate child NODE 668. For this case, there must be a security context for the link 672 to the parent NODE 670 and procedure SecurityService::DecryptLinkData() would be called to decrypt the received message. The link 672 down to the child NODE 668 may or
- 30

may not have a security context (the child NODE 668 may be sending the AS_REQ and TGS_REQ messages 634, 638 up to a higher KDC 610 in order to link to this NODE 608).

Primary or Non-Primary NODE: Receiving AP_REP from a parent NODE

5 This is already covered for the start-up cases.

It is to be appreciated that whilst the above description references several procedures such as SecurityService::DecryptLinkData(), specific details of these procedures has not been provided as their functions are relatively trivial to implement given the context and
10 functionality set out above and therefore this would be part of the skilled addressees skill and knowledge.

The above embodiments have been described using a push mechanism to send out the latest status of the registered users on line. However, in alternative embodiments this can be by a
15 pull mechanism where the e-mail server identifies the person to whom the e-mail is to be sent and then requests his status information from the DS. The advantage of the pull mechanism is that it minimises the transmission overhead for the data server. However, the advantage of the push mechanism is the ease of use for the e-mail server.

20 It is to be appreciated that the present invention is not limited to the specific embodiments described above. For example the e-mail message to be sent can be split into multiple parts and each part separately encrypted. Then each part can be sent separately to a respective transport server mechanism. Each part will be routed independently to the destination and can be decrypted and reassembled at the intended recipient. The advantage of this is that
25 interception and successful decryption of one e-mail message does not disclose the message. This is a very robust way of improving security of messaging. Typically, the text of an e-mail can be sampled to obtain one part of the message to be sent, e.g. say five TSMs were to be used then every fifth letter of the e-mail text could be sampled to create one of the five e-mail messages. The other e-mail messages would also be sampled at every fifth
30 letter but would each have a different offset from the first e-mail message.

It is also to be appreciated that rather than using wired networks, the present invention is applicable to wireless telecommunications networks (such as networks employing WIFI). Also rather than computers, PDAs (Personal Digital Assistants) and mobile telephones
5 could also be used as the source and destination communication devices in this peer-to-peer communications system.

Finally, the skilled addressee will appreciate that other encryption techniques can be used in preference to PKI.

Claims:

1. A method of carrying out a secure peer-to-peer data communication, such as an e-mail communication, between a first remote party computer and a second remote party computer over a data communications medium, the method comprising:

receiving the address details and current status of a connection to the data communications medium of each remote party computer;

creating the data communication at the first remote party computer;

checking the current connection status of the second remote party computer; and

sending the data communication from the first remote party computer directly to the second remote party computer without any storage of the data communication en route, only when the connection status of the second remote party computer indicates that it is currently connected to the data communication medium.

2. A method according to Claim 1, wherein the receiving step is repeated each time there is a change in the connection status of any of the remote party computers.

3. A method according to Claim 1, further comprising:

requesting the connection status of the second remote party computer from a data server which stores the current status; and

wherein the receiving step is carried out in response to the requesting step.

4. A method according to any preceding claim, further comprising providing, to a data server, address details and current status of the connection to the data communications medium of the first remote party computer.

5. A method according to any preceding claim further comprising encrypting the data communication with information relating to the second remote party computer, such that only the second party remote computer can decrypt the communication on receipt.

6. A method according to Claim 5, wherein receiving step comprises receiving the information relating to the second remote party computer for the purposes of carrying out the encrypting step.

7. A method according to any preceding claim, wherein the first remote party computer comprises a plurality of locally interconnected secure computers at the first remote party location, and the creating step is carried out on a different computer to the receiving, checking and sending steps.

8. A method according to any preceding claim, further comprising:
storing a set of permissions relating to the ability of the remote party computers to communicate with each other;
checking the permission of the first remote party computer to communicate with the second remote party computer; and
allowing the sending step to occur only if the checking step indicates that the first remote party computer has the permission to communicate with the second remote party computer.

9. A method according to any preceding claim further comprising verifying the identity of an intended recipient specified in the data communication as being the second remote party computer.

10. A method according to any preceding claim, further comprising:
temporarily storing the data communication locally and waiting a predetermined time interval after the checking step, if the connection status of the second remote party computer indicates that it is currently not available for receiving a message via the data communication medium; and
thereafter repeating the checking step and sending steps.

11. A method according to any preceding claim, further comprising:

assessing the transmission of the data communication from the first remote party computer to the second remote party computer and if the transmission failed:

determining the cause of the transmission failure and depending on the determined cause carrying out one of the following:

5 generating a transmission failed notice at the first remote party computer; or
 waiting a predetermined time period and repeating the sending step.

12. A method according to any preceding claim, further comprising:
 dividing the data communication into a plurality of separate data messages; and
10 wherein the sending step further comprises sending each one of the data messages separately to the second remote party computer, thereby improving the security of the data communication procedure.

13. A method according to any preceding claim, wherein data communication
15 comprises a voice over IP communication.

14. A communications server arranged to carry out a secure peer-to-peer data communication, such as an e-mail communication, between a first remote party computer including the server and a second remote party computer over a data communications
20 network, the server comprising:

 receiving means for receiving the address details and current status of a connection to the data communications network of each remote party computer;

 a message generation module for creating the data communication at the first remote party computer;

25 a checking module for checking the current connection status of the second remote party computer; and

 a transport module arranged to send the data communication from the first remote party computer directly to the second remote party computer without any storage of the data communication en route, only when the connection status of the second remote party
30 computer indicates that it is currently connected to the data communication network.

15. A communications system comprising:

a plurality of communication servers according to Claim 14; and

a data server connectable to the plurality of communications servers by the data

communications network,

wherein the data server is arranged to receive, collate, and store the current status of the connection of each of the communication servers to the communications network together with the current network address of each of the communications servers and to forward at least part of this information to the plurality of communications servers to enable them to effect peer-to-peer communications between ones of the plurality of communications servers.

16. A communications system according to Claim 15, wherein the central data server is arranged to carry out investigations into the current addresses of the plurality of communications servers using the User Datagram Protocol (UDP).

17. A communications server arranged to assist in establishing a secure peer-to-peer data communication, such as an e-mail communication, between first and second user computers over a data communications network, the server being provided at a given hierarchical level within a server network comprised of a plurality of the communications servers, the server comprising:

connection means enabling the communications server to be able to connect operably to other communications servers at other hierarchical levels within the server network;

registration means for registering a plurality of local user computers with the communications server; and

a data store for storing registration details of each registered local user computer, the registration details including address information and current status of a connection to the data communications network of each local user computer;

wherein the connection means is arranged to forward the stored registration details

to an adjacent communications server in the next higher hierarchical level of the server network and to receive and store registration details of any local user computers for a connected communications server at a lower level in the hierarchical network.

- 5 18. A communications server according to Claim 17, wherein the stored registration details comprise the server network identity of the communications server, the server network identity of at least one registered local user computer and the current status of the connection of the local user computer with the communications server.
- 10 19. A communications server according to Claim 17 or 18, wherein the stored registration details comprise the server network identity of a communications server from a lower level in the hierarchy and the server network identity of the lower level communications server's registered local user computers.
- 15 20. A communications server according to Claim 18 or 19, wherein the stored registration details comprise the global network address of each locally registered user computer.
- 20 21. A communications server according to any of Claims 17 to 20, wherein the stored registration details comprise the public encryption key of each locally registered user computer.
- 25 22. A communications server according to any of Claims 17 to 21, wherein the connection means comprises registration checking means arranged, on receipt of a request to establish a connection to the second user computer, to check the data store for local registration of the second user computer and to accept the request if the identity of the second user computer is stored in the local data store.
- 30 23. A communications server according to Claim 22, wherein the registration checking means is arranged to instruct the connection means to forward the request to the adjacent

communications server if the second user computer is not found to be registered locally.

24. A communications server according to Claim 22 or 23, wherein the connection means further comprises status checking means arranged to check data store for the status of a registered user computer and the connection means is arranged to seek establishing a communication with the second user computer if the second user computer is registered and has a status indicating that it can be connected to.

25. A communications server according to any of Claims 22 to 24, wherein the connection means is arranged to update the status of the second user computer in response to accepting a request to connect to the second user computer.

26. A communications server according to any of Claims 22 to 25, wherein the connection means is arranged to set a data communication status of the first and second user computers in response to accepting a request to connect to the second user computer.

27. A communications server according to any of Claims 17 to 26, further comprising responding means arranged to construct responses to the request for a connection to the second user computer; the responding means being arranged to send the current network address of the second user computer to the first user computer and to send the current network address of the first user computer to the second user computer.

28. A communications server according to Claim 27, wherein the responding means is arranged to determine a global network address of the second user computer.

29. A communications server according to Claim 28, wherein the global network address comprises a UDP channel address.

30. A communications server according to any of Claims 17 to 29, wherein the data store further comprises a set of permissions relating to the ability of the remote party

computers to communicate with each other; and the server further comprises means for using the stored set of permissions to determine whether the first user computer is permitted to communicate with the second user computer.

5 31. A communications system for carrying out a secure peer-to-peer data communication, such as an e-mail communication, between selected ones of a plurality of user computers over a data communications network, the system comprising:

10 a plurality of communications servers according to Claim 17, arranged in a hierarchically connected server network, each communications server representing a node of the server network, and the server network being connectable to the plurality of user computers to support establishment of communications between the selected ones of the plurality.

15 32. A method of assisting in establishing a secure peer-to-peer data communication, such as an e-mail communication, between first and second user computers over a data communications network, the method being implemented on a communications server at a given hierarchical level within a server network comprised of a plurality of the communications servers, the method comprising:

20 establishing operable network connections to communications servers at other hierarchical levels within the server network;

registering a plurality of local user computers with the communications server; and storing registration details of each registered local user computer, the registration details including address information and current status of a connection to the data communications network of each local user computer;

25 wherein the establishing step comprises forwarding the stored registration details to an adjacent communications server in the next higher hierarchical level of the server network and receiving and storing registration details of any local user computers for a connected communications server at a lower level in the hierarchical network.

30 33. A method of searching for an intended recipient computer of a peer-to-peer

communications message to assist in establishing a secure peer-to-peer data communication, such as an e-mail communication, between a sender computer and the intended recipient computer over a data communications network, the method being implemented within a hierarchical network of communications servers and comprising:

5 sending to a local server a request for data communication, the request including the intended recipient computer's identity and the sender computer's identity;

 determining whether the intended recipient is known to the local server;

 retrieving stored details regarding the intended recipient if the same is known to the local server, and transmitting these details back to the sender computer;

10 forwarding the request to an adjacent communications server in a next higher hierarchical level of the server network if the intended recipient is not known locally, the adjacent communications server then becoming the local server; and

 repeating the determining, retrieving and forwarding steps until the intended recipient is found or the server at the summit of the hierarchical network has been checked.

15

34. A method according to Claim 33, further comprising sending a notification back to the sender computer of a failure to connect if the intended recipient computer has not been identified and the summit of the hierarchical network has been reached.

20 35. A method according to Claim 33 or 34, further comprising checking the status of a connection to the intended recipient computer once the same has been identified.

36. A method according to Claim 35, further comprising sending a notification back to the sender computer of a failure to connect if the connection to the intended recipient
25 computer is unavailable for use.

37. A method according to Claim 36, further comprising waiting a predetermined time period to allow for a possible change in status and thereafter re-executing the sending, determining, retrieving, forwarding and repeating steps.

30

38. A method according to any of Claims 33 to 37, wherein the retrieving step comprises transmitting the request to a communications server located at a lower level in the hierarchy at which the intended recipient is registered, if the intended recipient is known to the communications server but is not locally registered there.

5

39. A method according to Claim 38, wherein the retrieving step comprises retrieving the network location of the intended recipient from its locally registered server as part of the stored details of the intended recipient.

10 40. A method according to any of Claims 33 to 39, further comprising:

storing and executing a watching function at the server for future use in finding the desired intended recipient computer, the watching function comprising comparing the identity of any new computer added to the local server to the identity of the desired intended recipient computer and notifying the sender computer on the occurrence of a
15 match.

41. A method according to any of Claims 33 to 40, further comprising using a stored set of permissions to determine whether the sender computer is permitted to communicate with the desired intended recipient computer, and only allowing the transmitting step to be
20 carried out if the stored permissions allow such communication.

42. A method of establishing a secure peer-to-peer data communication, such as an e-mail communication, between a sender computer and an intended recipient computer over a data communications network, the establishing method being implemented within
25 hierarchical server network comprised of a plurality of communications servers, the establishing method comprising:

a method of searching for an intended recipient computer according to any of Claims 33 to 41;

communicating the current global communications address of the intended recipient
30 computer to the sender computer;

communicating the current global communications address of the sender computer to the intended recipient computer; and

using the global communications addresses to set up a peer-to-peer communications channel between the sender and the intended recipient computers.

5

43. A method according to Claim 42, further comprising:

storing a set of permissions relating to the ability of the remote party computers to communicate with each other;

checking a stored set of permissions relating to the ability of the sender computer to communicate with the intended recipient computer; and

allowing the first and second communicating steps and the sending step to occur only if the checking step indicates that the sender computer has the permission to communicate with the intended recipient computer.

15 44. A method according to Claim 42 or 43, further comprising setting up a first direct channel between the sender computer and the local server.

45. A method according to Claim 44, wherein the second communication step is carried out over the first direct channel.

20

46. A method according to any of Claims 42 to 45, further comprising setting up a second direct channel between the intended recipient computer and the local server.

47. A method according to Claim 46, wherein the first communication step is carried out over the second direct channel.

25

48. A method according to any of Claims 44 to 47, wherein the setting up of a first or second direct channel comprises setting up a first or second UDP (User Datagram Protocol) direct channel.

30

49. A transport server for use in establishing a peer-to-peer data communication, such as an e-mail communication, between a first and second user computers, the transport server comprising:

receiving means for receiving from the first user computer a request for connection to the second user computer which is registered with the transport server;

verifying means for verifying the current connection status of a connection to the second user computer;

a data store for storing details of the request as a watch if the current connection status of the second user computer indicates that a peer-to-peer communication cannot at present be established with the second user computer; and

response means responsive to the verifying means to send a message to the first user computer indicating the on-line status of the second user computer if the status of the same changes to indicate that a peer-to-peer communication can now be established with the second user computer;

wherein the verifying means is arranged to periodically check and update the current connection status to the second user computer, and when an update indicates that the second user status has changed to on-line, to check for the existence of a corresponding watch and if found to activate the response means to send the message.

50. A transport server according to Claim 49, wherein the data store is arranged to store the identity of the second user computer and the identity of the first user computer.

51. A transport server according to Claim 49 or 50, wherein the receiving means is arranged to receive a plurality of requests for connection to the second user computer, the data store is arranged to store details of each request in the watch against the second user computer and the response means is arranged to send out a plurality of messages, one to each of the user computers which have requested a connection to the second computer, once the connection status of the second user computer changes to on-line.

52. A transport server according to any of Claims 49 to 51, wherein the transport server

is arranged to be connected to a hierarchy of transport servers

53. A transport server according to Claim 52, wherein the transport server is arranged to send details of the connection status of each of its registered user computers to transport servers located at higher positions within the hierarchy and the response means is arranged to send a message indicating the on-line status of the second user computer to at least one transport server located at a higher position within the hierarchy

54. A transport server according to any of Claims 49 to 53, wherein the data store is arranged to store a plurality of different watches against different ones of the user computers registered with the transport server; and

the verifying means is arranged to periodically check and update the current connection status to all of its registered user computers, and when an update indicates that any user status has changed to on-line, to check for the existence of a corresponding watch amongst the plurality of stored watches and if found, to activate the response means to send the message corresponding the found watch.

55. A transport server according to any of Claims 49 to 54, wherein the data store is arranged to store a list of registered user computers currently on line; and the transport server further comprises on-line checking means for checking the currently on line list of user computers prior to storing a watch in the data store.

56. A transport server according to Claim 55, further comprising transfer means for transferring the watched user computer identity to the currently on line list, when the status of the registered user computer indicates it has changed to be on-line.

57. A method of assisting the establishment of a peer-to-peer data communication, such as an e-mail communication, between a first and second user computers, the method comprising:

receiving from the first user computer a request for connection to the second user

computer which is locally registered;

verifying the current connection status of a connection to the second user computer;

storing details of the request as a watch if the current connection status of the second user computer indicates that a peer-to-peer communication cannot at present be established
5 with the second user computer; and

sending a message to the first user computer indicating the on-line status of the second user computer if in response to the verifying step the status of the second user computer changes to indicate that a peer-to-peer communication can now be established with the second user computer;

10 wherein the verifying step comprises periodically checking and updating the current connection status to the second user computer, and when an update indicates that the second user status has changed to on-line, checking for the existence of a corresponding watch and if found activating the response means to send the message.

15 58. A method of establishing a secure peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where at least the first user computer has its communications handled by a first Network Address Translator (NAT), the method comprising:

requesting a direct connection over an established Transmission Control
20 Protocol/Internet Protocol (TCP/IP) communications link between the first user computer and a transport server through the first NAT;

establishing a first User Datagram Protocol (UDP) port at the transport server;

reporting the address of the first UDP port to the first user computer via the TCP/IP communications link;

25 opening a second UDP port at the first user computer;

transmitting data packets from the second UDP port via the first NAT to the first UDP port such that the transport server is able to determine the first NAT address of the second UDP port;

obtaining at the transport server a third UDP port address of the second user
30 computer; and

notifying each of the first and second user computers of the other user computer's UDP port address such that they can establish a secure peer-to-peer communication therebetween.

5 59. A method according to Claim 58, further comprising creating the TCP/IP communications link from the first user computer to a transport server through the first NAT.

10 60. A method according to Claim 59, wherein the creating step comprises searching for and identifying a local transport server on a transport server network at which the second user computer is registered, and connecting to this local transport server.

15 61. A method according to any of Claims 58 to 60, wherein the second user computer is registered with the transport server and has a TCP/IP communications link established therebetween via a second NAT, and the obtaining step further comprises:

establishing a fourth UDP port at the transport server;

reporting the address of the fourth UDP port to the second user computer via the TCP/IP communications link;

opening the third UDP port at the second user computer; and

20 transmitting data packets from the third UDP port via the second NAT to the fourth UDP port such that the transport server is able to determine the second NAT address of the third UDP port of the second user computer.

25 62. A method according to any of Claims 58 to 61, further comprising the first user computer switching the destination of the second UDP port from the first UDP port to the third UDP port.

30 63. A method according to Claim 62, wherein the first user computer is arranged to send peer-to-peer messages directly from the second UDP port to the third UDP port of the second user computer.

64. A method according to any of Claims 58 to 63, further comprising the second user computer switching the destination of the third UDP port from the fourth UDP port to the second UDP port.

5

65. A method according to Claim 64, wherein the second user computer is arranged to send peer-to-peer messages directly from the third UDP port to the second UDP port of the second user computer.

10 66. A method according to any of Claims 58 to 65, further comprising:
detecting inactivity of the first UDP port when a peer-to-peer communication has been established between the first and second user computers; and
breaking the connection with the first user computer by closing the first UDP port.

15 67. A method according to Claim 66, further comprising the transport server closing the TCP/IP connection with the first user computer.

68. A method according to any of Claims 58 to 67, further comprising:
detecting inactivity of the fourth UDP port when a peer-to-peer communication has
20 been established between the first and second user computers; and
breaking the connection with the second user computer by closing the fourth UDP port.

69. A method according to Claim 68, further comprising the transport server closing the
25 TCP/IP connection with the second user computer.

70. A method according to any of Claims 58 to 69, wherein the either the first or the second NAT comprises an asymmetrical NAT such that the UDP port for each different destination is different, and the method further comprises:
30 noting the new address of the direct connection from the asymmetrical NAT; and

readdressing the UDP port output to the new address of the asymmetrical NAT.

71. A transport server for assisting in establishing a secure peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where at least the first user computer has its communications handled by a first Network Address Translator (NAT), the method comprising:

request receiving means for receiving a request for a direct connection over an established Transmission Control Protocol/Internet Protocol (TCP/IP) communications link between the first user computer and the transport server through the first NAT;

establishing means for establishing a first User Datagram Protocol (UDP) port at the transport server;

reporting means for reporting the address of the first UDP port to the first user computer via the TCP/IP communications link;

data packet receiving means for receiving data packets transmitted from a second UDP port set up at the first user computer via the first NAT to the first UDP port, the transport server being arranged to determine the first NAT address of the second UDP port;

obtaining means for obtaining at the transport server a third UDP port address of the second user computer; and

notifying means for notifying each of the first and second user computers of the other user computer's UDP port address such that they can establish a secure peer-to-peer communication therebetween.

72. A method of establishing a secure pseudo peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where both the user computers have their communications handled by respective first and second asymmetric Network Address Translators (NATs), the method comprising:

creating a Transmission Control Protocol/Internet Protocol (TCP/IP) communications link from each of the user computers to a transport server through the

respective first and second asymmetric NATs;

establishing a first and a second User Datagram Protocol (UDP) port at the transport server in response to receiving a request for a direct connection, the request being received over the TCP/IP communications link between the first user computer and a transport server
5 through the first NAT;

reporting the addresses of the first and second UDP ports to the first and second user computers respectively via their respective TCP/IP communications links;

opening a third UDP port at the first user computer and a fourth UDP port at the second user computer;

10 transmitting data packets from the third UDP port via the first NAT to the first UDP port and from the fourth UDP port via the second NAT to the second UDP port such that the transport server is able to determine the first NAT address of the third UDP port and the second NAT address of the fourth UDP port; and

retransmitting data packets received at the first UDP port to the NAT address of the fourth UDP port via the second UDP port and data packets received at the second UDP port
15 to the NAT address of the third UDP port via the first UDP port, such that pseudo peer-to-peer communications between the first and second user computers is established by effectively bouncing data packets off of the transport server.

20 73. A method according to Claim 72, wherein the retransmitting step comprises momentary transient storage of the received data packets prior to retransmission.

74. A method according to Claim 72 or 73, wherein the creating step comprises searching for and identifying a local transport server on a transport server network at which
25 the second user computer is registered, and connecting to this local transport server.

75. A method according to any of Claims 72 to 74, further comprising the transport server closing the TCP/IP connections with the first and second user computers once the pseudo peer-to-peer communications have been established.

76. A transport server for assisting in establishing a secure pseudo peer-to-peer data communication channel, such as an e-mail communication channel, between a first and second user computers over a data communications network where both the user computers have their communications handled by respective first and second asymmetric Network Address Translators (NATs), the transport server comprising:

creating means for creating a Transmission Control Protocol/Internet Protocol (TCP/IP) communications link from each of the user computers to a transport server through the respective first and second asymmetric NATs;

establishing means for establishing a first and a second User Datagram Protocol (UDP) port at the transport server in response to receiving a request for a direct connection, the request being received over the TCP/IP communications link between the first user computer and a transport server through the first NAT;

reporting means for reporting the addresses of the first and second UDP ports to the first and second user computers respectively via their respective TCP/IP communications links;

receiving means for receiving data packets transmitted from a third UDP port set up at the first computer via the first NAT to the first UDP port and from the fourth UDP port set up at the second user computer via the second NAT to the second UDP port such that the transport server is able to determine the first NAT address of the third UDP port and the second NAT address of the fourth UDP port; and

retransmitting means for retransmitting data packets received at the first UDP port to the NAT address of the fourth UDP port via the second UDP port and data packets received at the second UDP port to the NAT address of the third UDP port via the first UDP port, such that pseudo peer-to-peer communications between the first and second user computers is established by effectively bouncing data packets off of the transport server.

77. A method of connecting a user computer to a hierarchical connection network of transport servers for assisting in establishing a peer-to-peer communication between a sender computer and an intended recipient computer over a data communications network, the method comprising:

receiving information describing the current loading of each of a plurality of peer transport servers at the same hierarchical level in the connection network as a local transport server;

receiving a request to connect a local user computer to the local transport server;

5 comparing the current loading of the local server with each of the peer servers;

sending a response to the local user computer indicating that it should connect to the peer server having the lowest loading if the loading of the local transport server is significantly greater than the loading of any one of the peer servers; and

10 accepting the request to connect if the loading of the local transport server is not significantly greater than the loading of any one of the peer servers, and updating the current loading of the local transport server.

78. A method according to Claim 77, wherein the current loading of a transport server is determined by the number of user computers registered with that transport server.

15

79. A method according to Claim 77 or 78, further comprising forwarding the current loading of the local transport server to a transport server at higher level in the hierarchical network.

20 80. A method according to Claim 79, wherein the forwarding step comprises sending out the current IP address and port of the local transport server.

81. A method according to Claim 79 or 80, wherein the forwarding step comprises sending out a heartbeat signal at regular intervals, each heartbeat signal providing updated
25 current information about the loading of the local transport server.

82. A method according to any of Claims 77 to 81, wherein the receiving step comprises receiving information describing the current IP address and port of each of the plurality of peer transport servers.

30

83. A method according to any of Claims 77 to 82, wherein the receiving step comprises regularly receiving a heartbeat signal about each of a plurality of peer transport servers, the heartbeat signal including the identity of the peer transport server and information describing the current loading of the peer transport server.

5

84. A method according to any of Claims 77 to 83, further comprising determining the geographical location of the local transport server.

85. A method according to Claim 84, wherein the receiving step comprises receiving
10 information regarding the geographical location of each of the plurality of peer transport servers.

86. A method according to Claim 84 or 85, wherein the geographical information comprises the time zone information regarding the local or peer transport server.

15

87. A method according to Claim 85 or Claim 86 as dependent on Claim 85, wherein the comparing step comprises comparing the current loading of the local server with each of the peer servers in the same geographical location as the local user; and

the sending step comprises sending the response to the local user computer
20 indicating that it should connect to the peer transport server in the same geographical location as the local user computer which has the lowest loading, if the loading of the local transport server is significantly greater than the loading of any one of the peer servers in the same geographical location.

25 88. A method according to any of Claims 77 to 87, wherein the sending step comprises sending the communications address of a parent transport server which resides in an adjacent and higher hierarchical level of the connection network to the local transport server.

30 89. A method according to any of Claims 77 to 88, further comprising:

acquiring and considering the loading of all child transport servers connected to the local transport server and residing at a lower level in the hierarchical connection network
evaluating the current loading of the local server with each of the child servers;
transmitting a response to the local user computer indicating that it should connect
5 to the child server having the lowest loading if the loading of the local transport server is significantly greater than the loading of any one of the child servers.

90. A method according to Claim 89, further comprising determining the geographical location of the local transport server and wherein the acquiring step comprises receiving
10 information regarding the geographical location of each of the plurality of child transport servers.

91. A method according to Claim 90, wherein the geographical information comprises the time zone information regarding the local or child transport server.

15

92. A method according to Claim 90 or 91, wherein the evaluating step comprises comparing the current loading of the local server with each of the child servers in the same geographical location as the local user; and

the transmitting step comprises transmitting the response to the local user computer
20 indicating that it should connect to the child transport server in the same geographical location as the local user computer which has the lowest loading, if the loading of the local transport server is significantly greater than the loading of any one of the child servers in the same geographical location.

25 93. A method of joining a node to a network of authenticated communications servers set up to be used in establishing a peer-to-peer data communications between user computers registered with the communications servers, the method comprising:

using a user identity and password received from an authentication server to authenticate the node to the authentication server;

30 being notified of the identity of a selected communications server to which the node

needs to connect to join the network;

requesting selected communications server and node specific data from the authentication server for a connection to the selected communications server;

5 receiving selected communications server and node specific data and a shared encryption key shared by the node and the selected communications server;

10 transmitting the selected communications server and node specific data and global data encrypted by the shared encryption key to the selected communications server, such that the selected communications server has the tools by which to authenticate the node to the network and thereby join to the same without the need to seek verification from the authentication server.

15 94. A method according to Claim 93, further comprising receiving confirmation data encrypted by the shared encryption key from the selected communications server such that the node has the information to verify the authenticity of the selected communications server.

95. A method according to Claim 93 or 94, further comprising obtaining a user ID and password by registering the node with the authentication server of the network;

20 96. A method according to any of Claims 93 to 95, wherein the being notified step comprises accepting a ticket generation ticket for use in the requesting step.

25 97. A method according to Claim 96, wherein the being notified step comprises getting a session key for the node for use in the receiving step.

98. A method according to any of Claims 93 to 97, wherein the ticket generation ticket and the session key are encrypted with the password and the method further comprises decrypting the ticket generation ticket and the session key.

30 99. A method according to any of Claims 96 to 98, wherein the requesting step

comprises conveying the ticket generating ticket and the identity of the selected communications server to the authentication server.

100. A method according to any of Claims 93 to 99, wherein the receiving step comprises
5 obtaining a ticket specific to the selected communications server and node.

101. A method according to Claim 100, wherein the ticket contains a copy of the shared encryption key.

102. A method according to Claim 101, further comprising decrypting the shared key that
10 has been encrypted with the session key.

103. A method according to any of Claims 93 to 102, further comprising calculating the
15 value of a timestamp as the global data.

104. A method according to any of Claims 93 to 103, further comprising the selected
communications server decrypting the received data using the shared key.

105. A method according to any of Claims 93 to 104, further comprising the selected
20 communications server verifying the validity of the received data.

106. A method according to Claim 104 or 105, further comprising the selected
communications server sending a confirmation that the decrypted received data is
25 acceptable.

107. A method according to any of Claims 93 to 106, further comprising receiving the
authentication data modified in a predetermined manner.

108. A method according to Claim 107, wherein the modified authentication information
30 is encrypted and the method further comprises decrypting the modified authentication

information.

109. A method of a first user computer securely communicating with a second user computer which is a part of a network of authenticated communications servers set up to be used in establishing a peer-to-peer data communications between user computers registered with the communications servers, the method comprising:

Joining the first computer to the network as described in any of Claims 1 to 16;

Creating public and private keys at the first and second user computers;

Exchanging public keys between the first and second user computers via the network;

Creating shared encryption/decryption keys from the non-shared private keys and the shared public keys at each of the first and second user computers; and

Using the shared encryption keys to encrypt data messages sent directly between the first and second user computers over an established peer-to-peer connection between the first and second user computers.

110. A method of connecting a first hierarchical realm of interconnected transport server nodes to a second hierarchical realm of interconnected transport server nodes, the method comprising:

providing a local authenticating server within each realm, the authenticating server being connected to a primary node at the highest level of the respective hierarchical realm and being arranged to control the authentication issues related to all servers within the realm;

registering the primary node of the first hierarchical realm to the authenticating server of the second hierarchical realm;

authenticating the primary node of the first hierarchical realm to the authenticating server of the second hierarchical realm;

being notified of the identity of a lowermost node server to which the primary node of the first hierarchical realm needs to connect to join the hierarchical realms together;

receiving shared data and shared encryption keys for both of the lowermost node

server of the second hierarchical realm and the primary node of the first hierarchical realm;

using the receiving shared data and shared encryption keys to authenticate the primary node of the first hierarchical realm to the lowermost node server of the second hierarchical realm and thereby to join the first and second realms together without the need to seek verification from the authenticating server of the second realm.

5

111. A method according to Claim 110, further comprising requesting data specific to the primary node of the first realm and the lowermost node server of the second realm from the authentication server for a connection to the selected lowermost node server of the second realm.

10

112. A method according to Claim 110 or 111, further comprising transmitting the shared data and global data encrypted by the shared encryption key to the lowermost node server, such that the lowermost server has the tools by which to authenticate the primary node to the second hierarchical realm.

15

113. A method according to any of Claims 110 to 112, wherein the authentication step is carried out via the lowermost node server of the second hierarchical realm.

20

114. A method according to any of Claims 110 to 113, further comprising:
registering a primary node of the second hierarchical realm an authenticating server of the first hierarchical realm; and
authenticating the primary node of the second hierarchical realm to the authenticating server of the first hierarchical realm.

25

115. A method of a first user computer connected to a first realm of transport servers securely communicating with a second user computer connected to a second realm of transport servers set up to be used in establishing a peer-to-peer data communications between user computers registered with the communications servers, the method comprising:

30

Connecting the first user computer of the first realm to the second realm of transport servers as described in any of Claims 110 to 114; and

Determining the network locations within the interconnected realms for the first and second user computers;

5 Using the established network locations to establish a peer-to-peer communications channel between the first and second user computers; and

 Sending encrypted data communications across the established peer-to-peer communications channel.

10 116. A method according to Claim 115, wherein the using step comprises:

 Creating public and private keys at the first and second user computers;

 Exchanging public keys between the first and second user computers via the network;

15 Creating shared encryption/decryption keys from the non-shared private keys and the shared public keys at each of the first and second user computers; and

 Using the shared encryption keys to encrypt data messages sent directly between the first and second user computers over an established peer-to-peer connection between the first and second user computers.

1 / 29

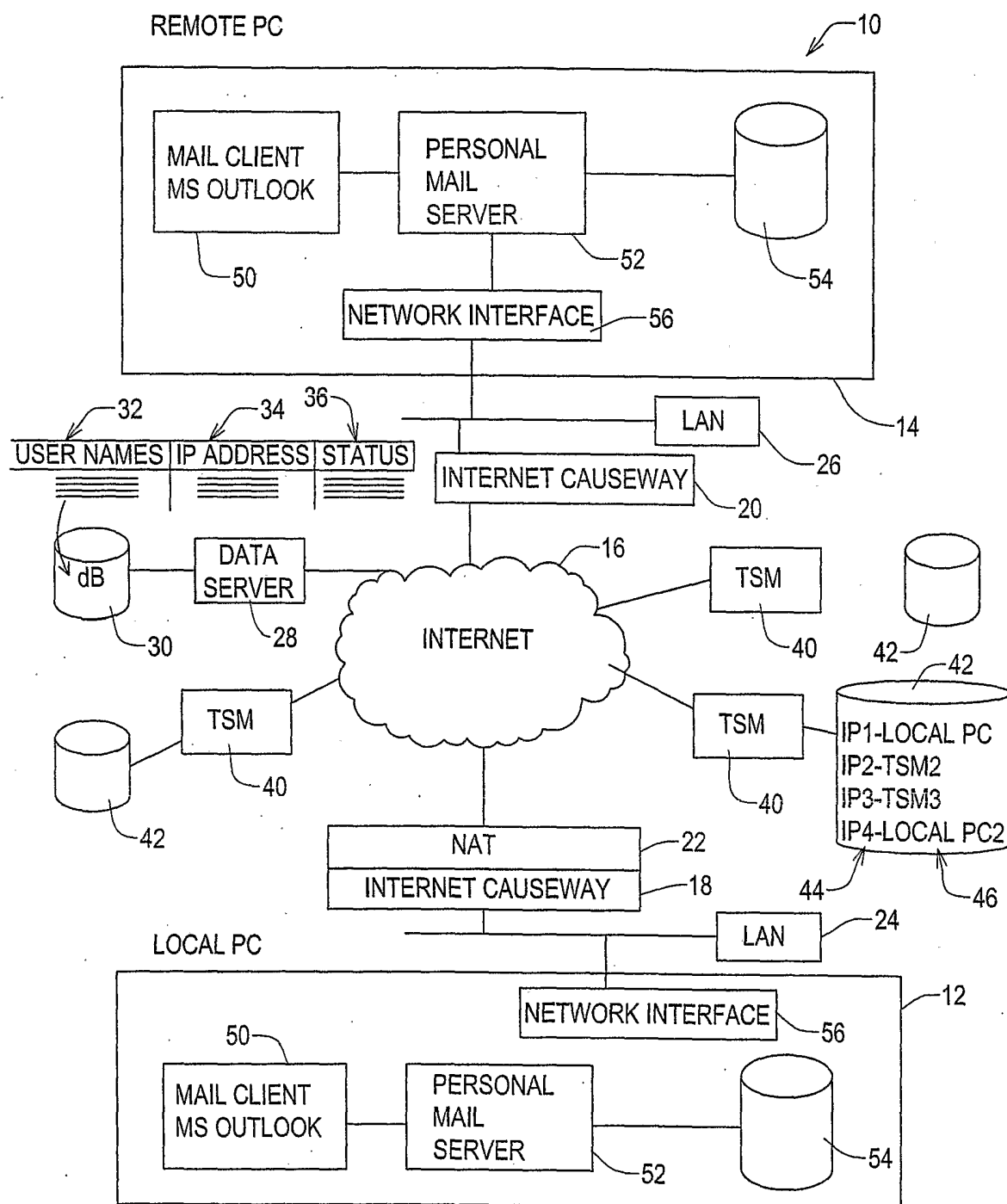
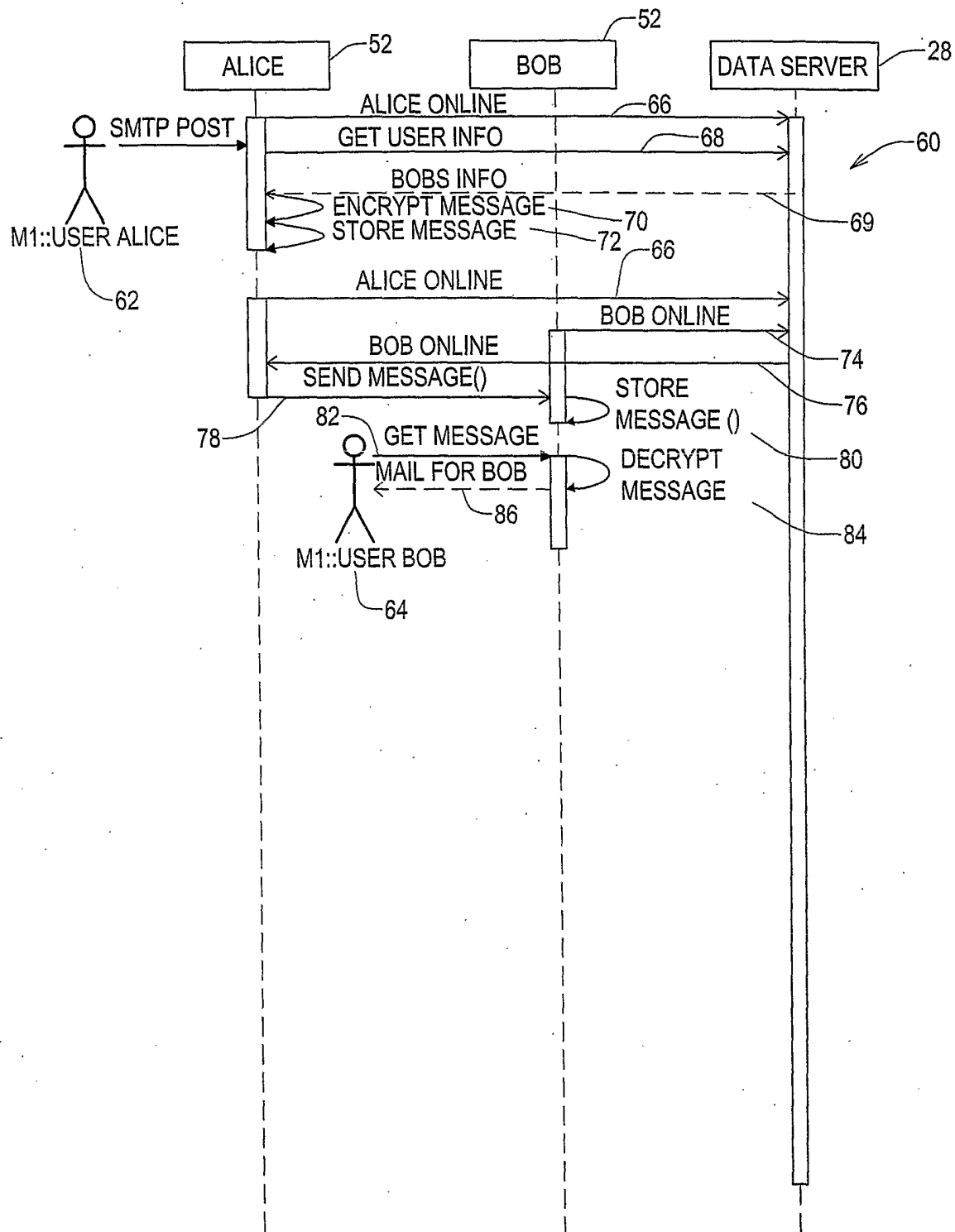
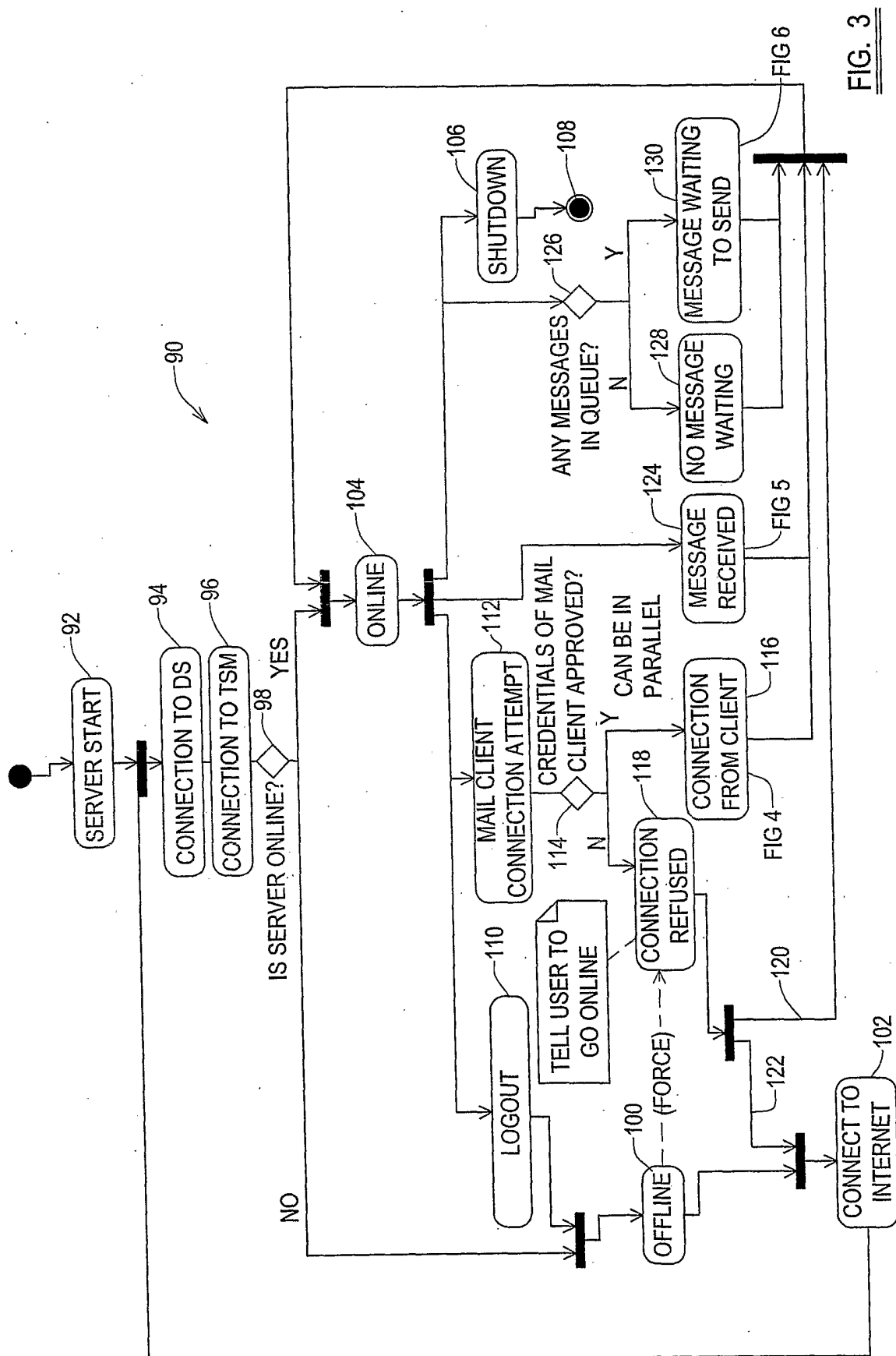


FIG. 1

2 / 29

FIG. 2



4 / 29

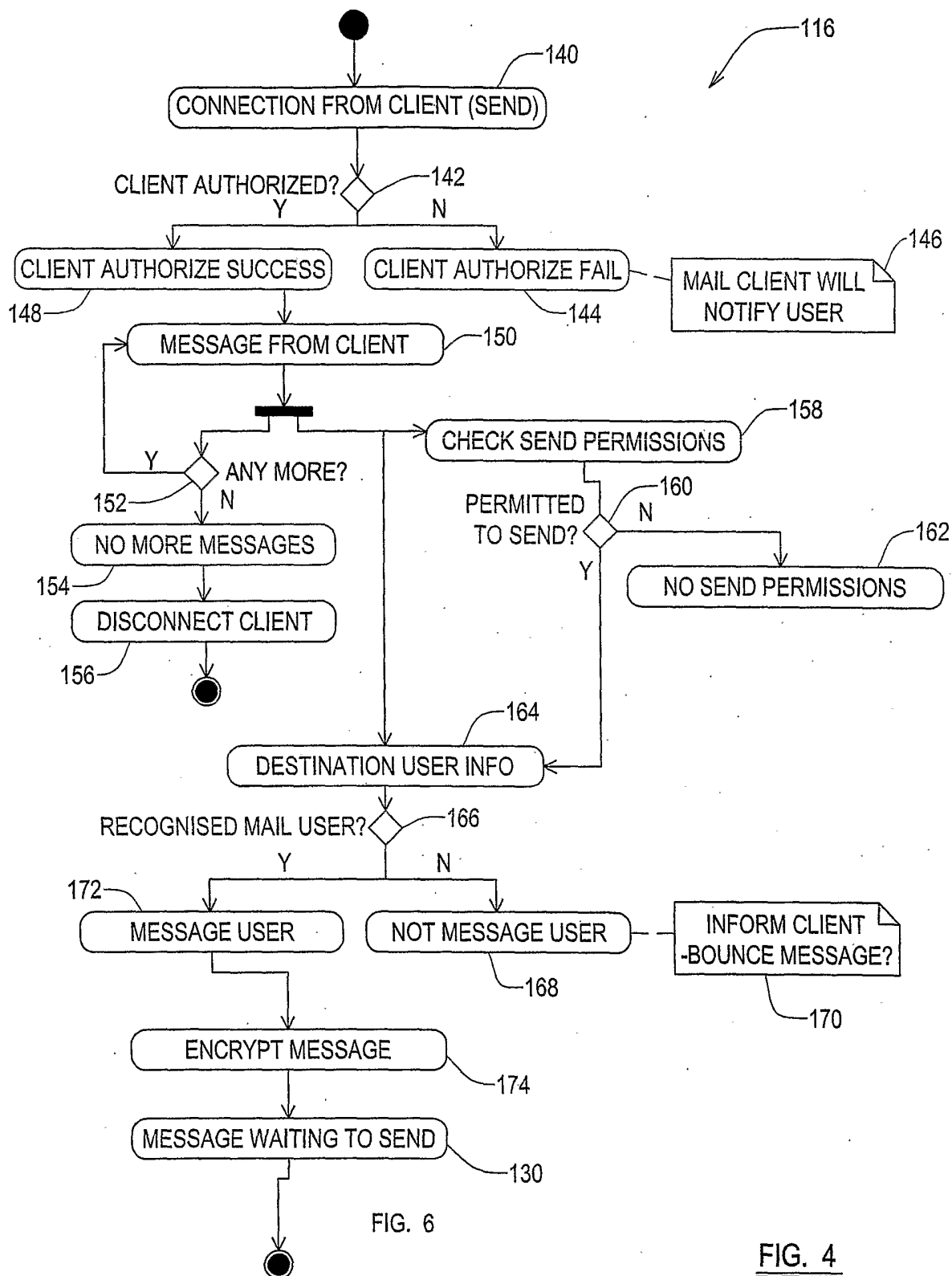


FIG. 6

FIG. 4

5 / 29

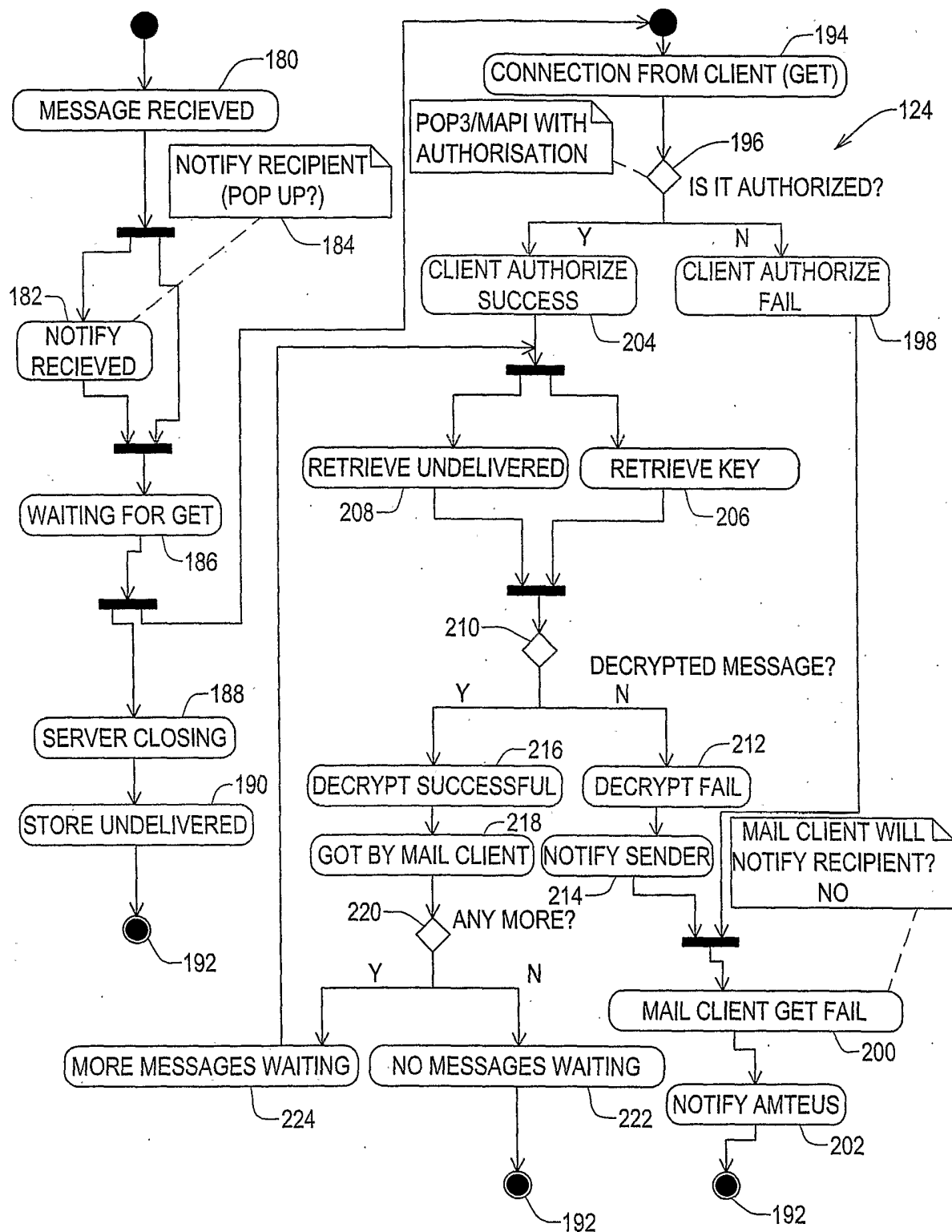
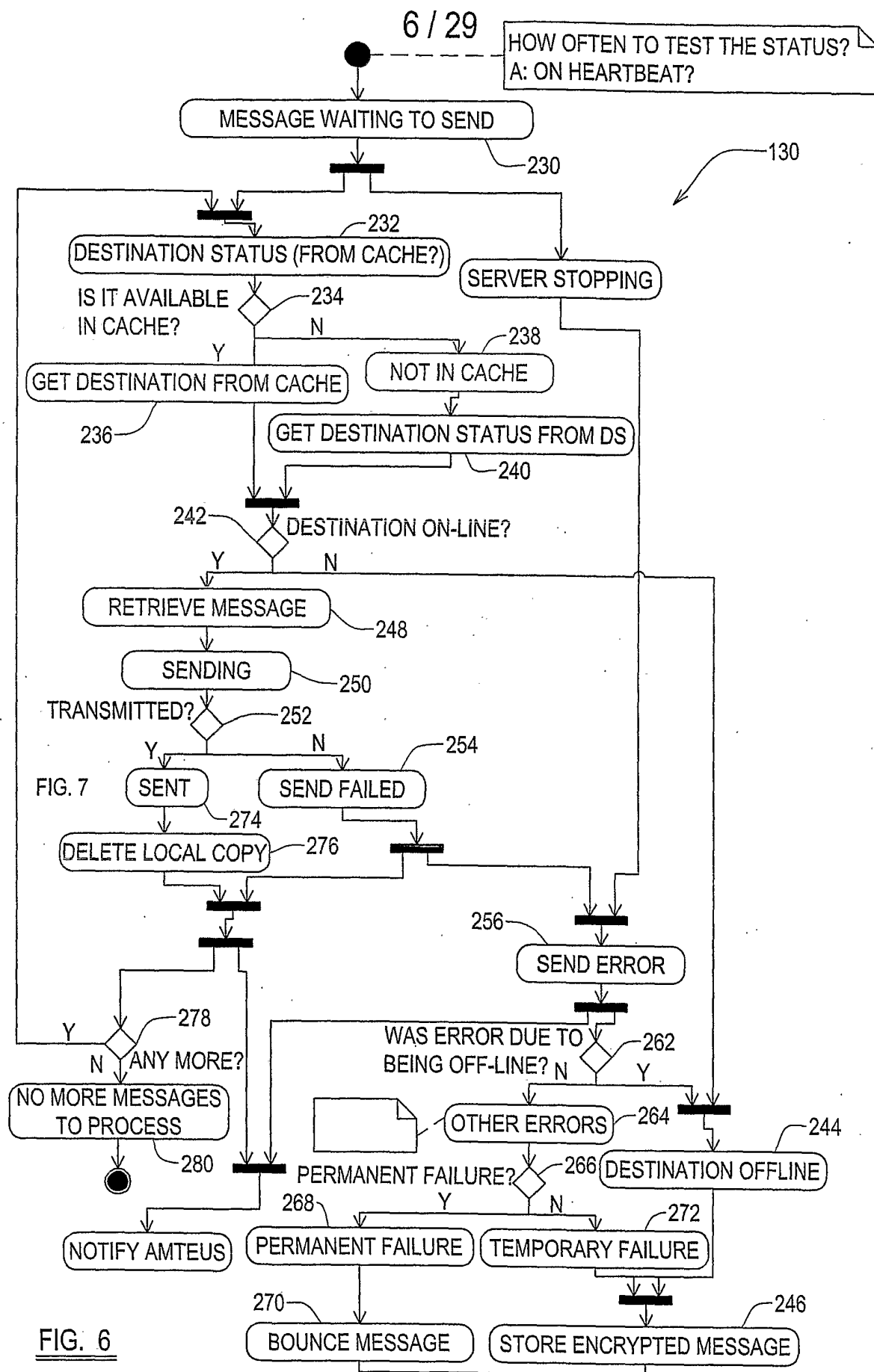
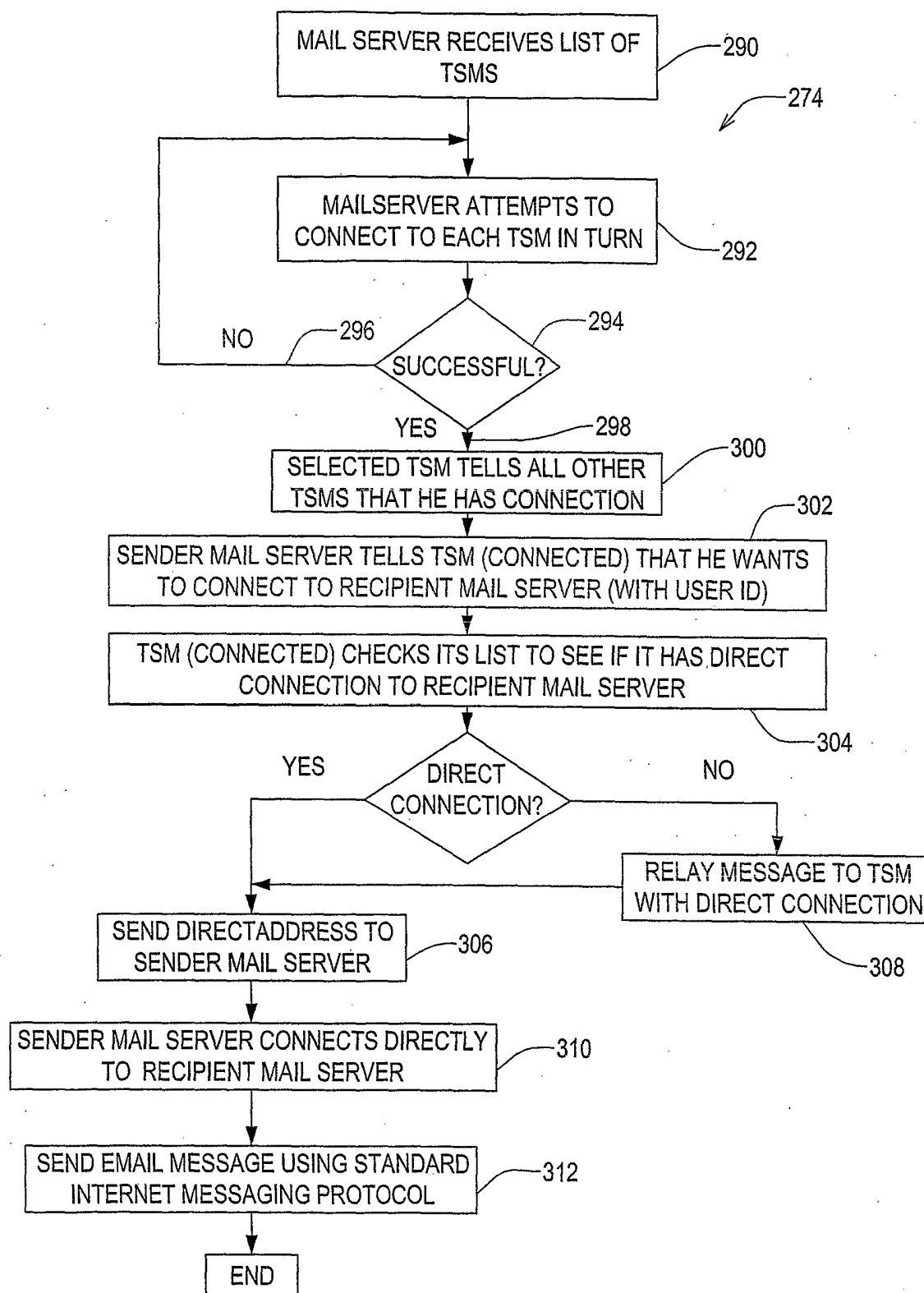
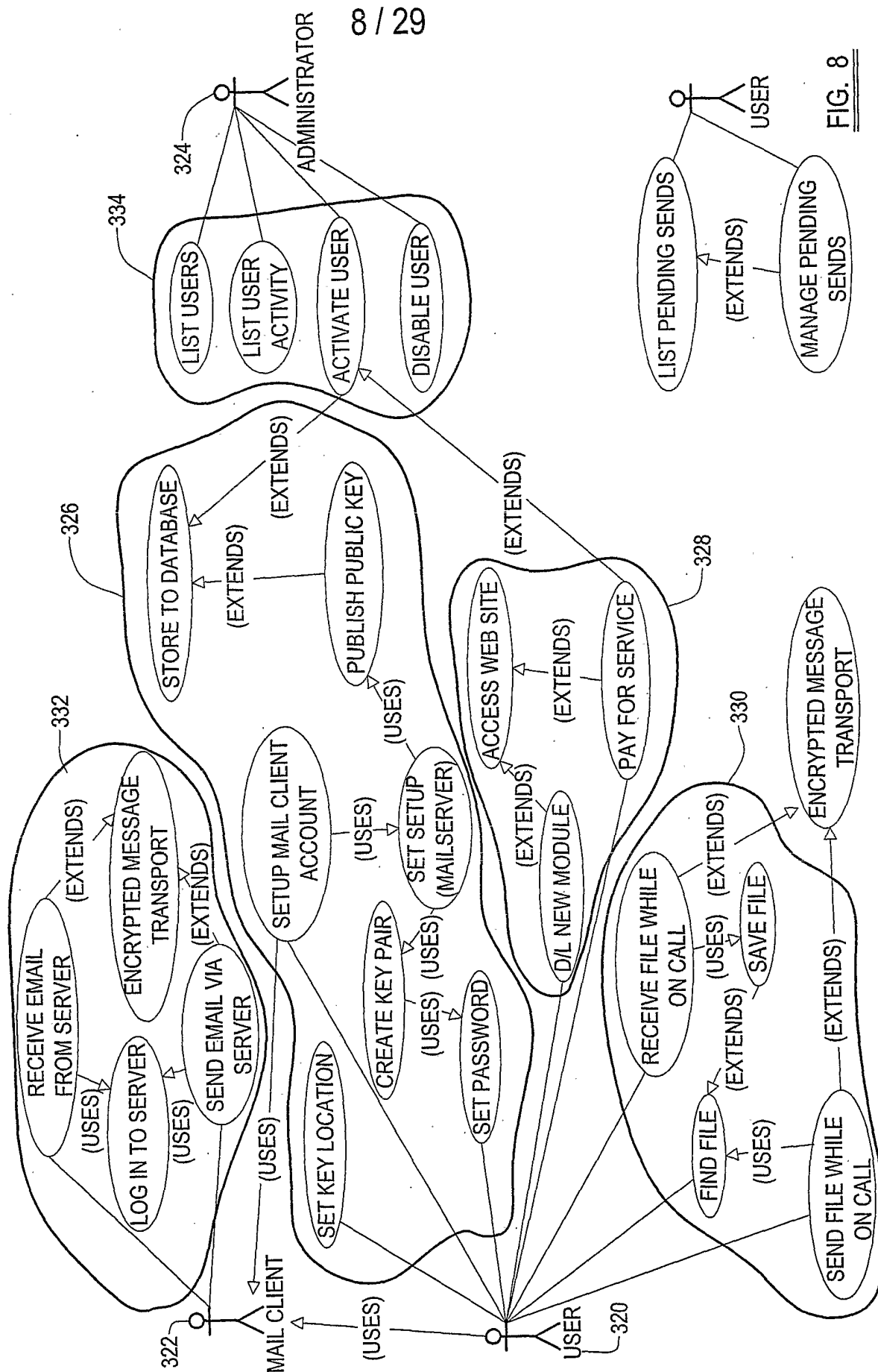


FIG. 5

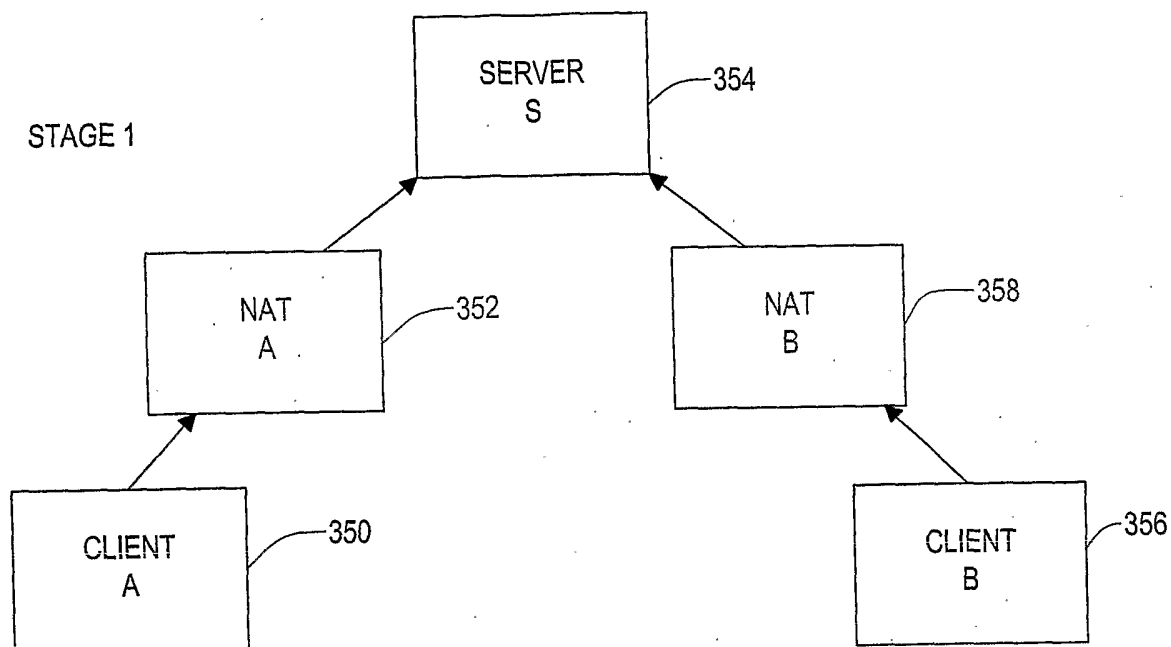


7 / 29

FIG. 7



9 / 29

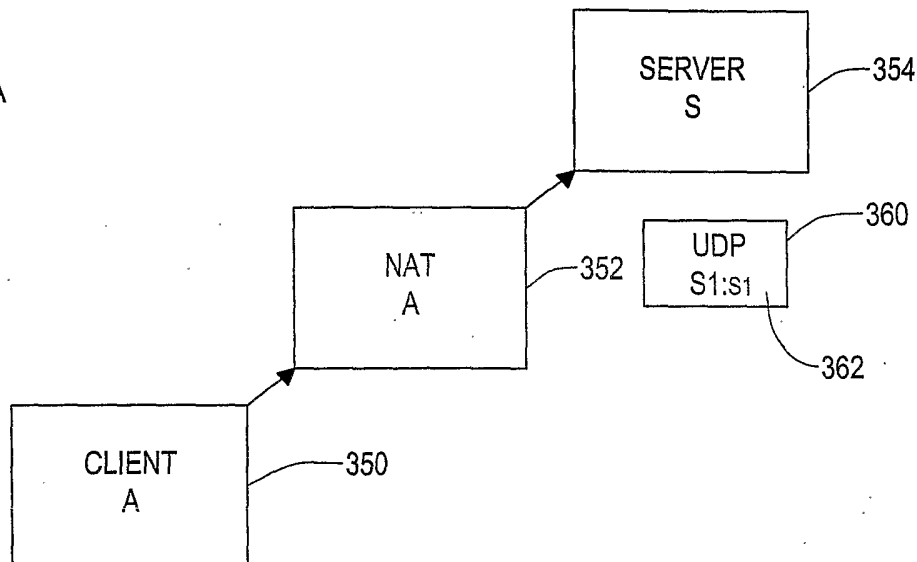


- INITIAL STATE: A & B HAVE TCP/IP CONNECTIONS THROUGH NATS TO SERVER S
- CLIENT A DESIRES A P2P CONNECTION TO CLIENT B

FIG. 9a

10 / 29

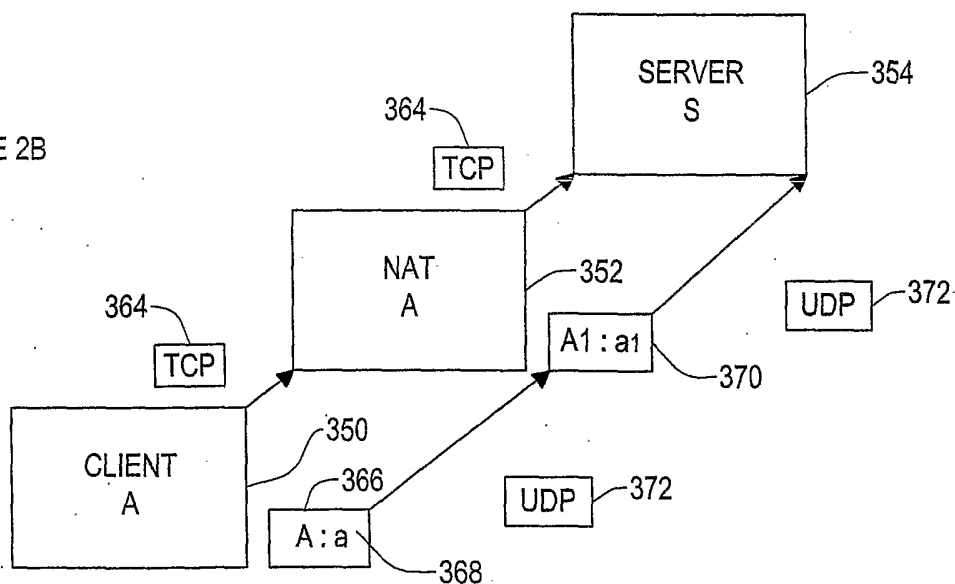
STAGE 2A



- CLIENT A HAS MADE P2P REQUEST TO SERVER S
- SERVER S OPENS UDP PORT ON IP ADDRESS S PORT s

FIG. 9b

STAGE 2B

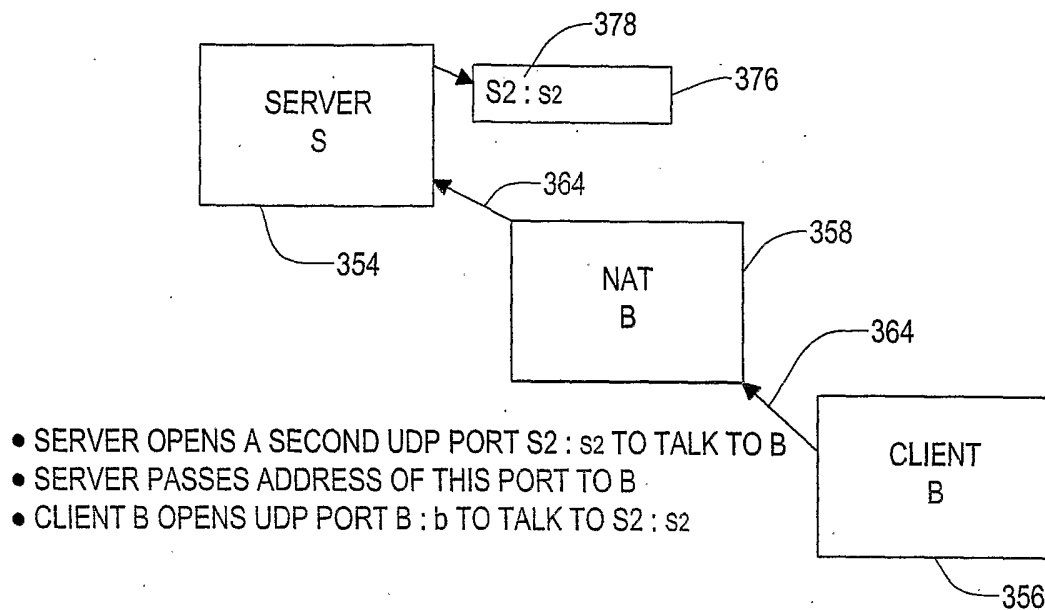


- S REPORTS ADDRESS S1 : s1 (VIA TCP) TO CLIENT A. A OPENS UDP A : a TO TALK TO IT. AFTER GOING THROUGH NAT A THIS APPEARS AS A1 : a1

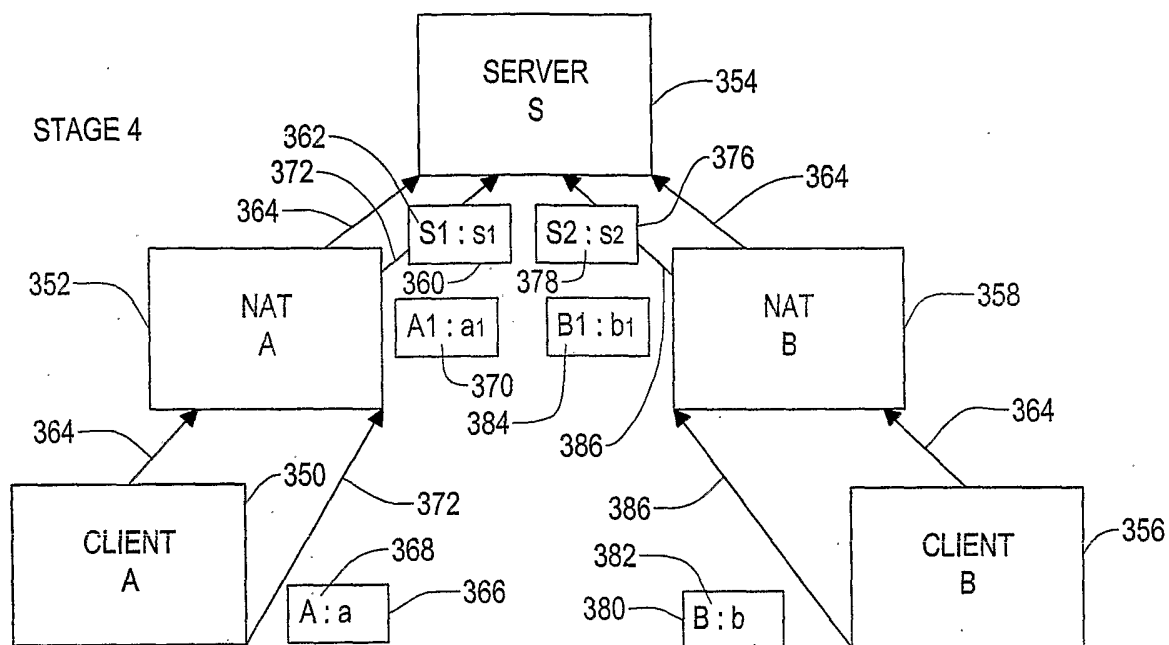
FIG. 9c

11 / 29

STAGE 3

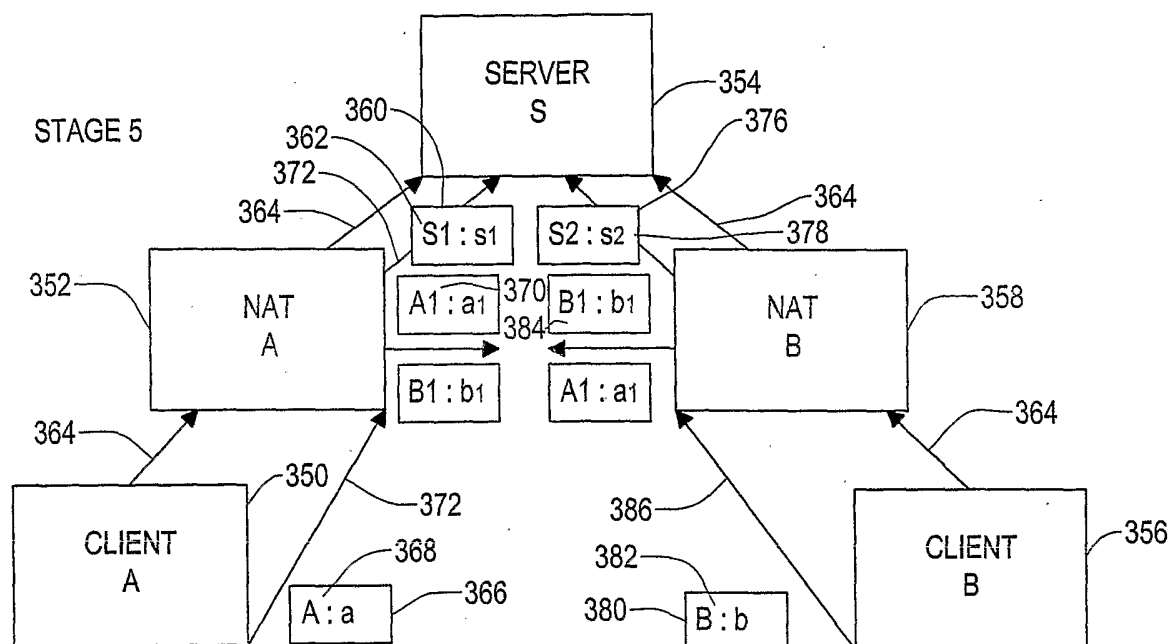
FIG. 9d

12 / 29

FIG. 9e

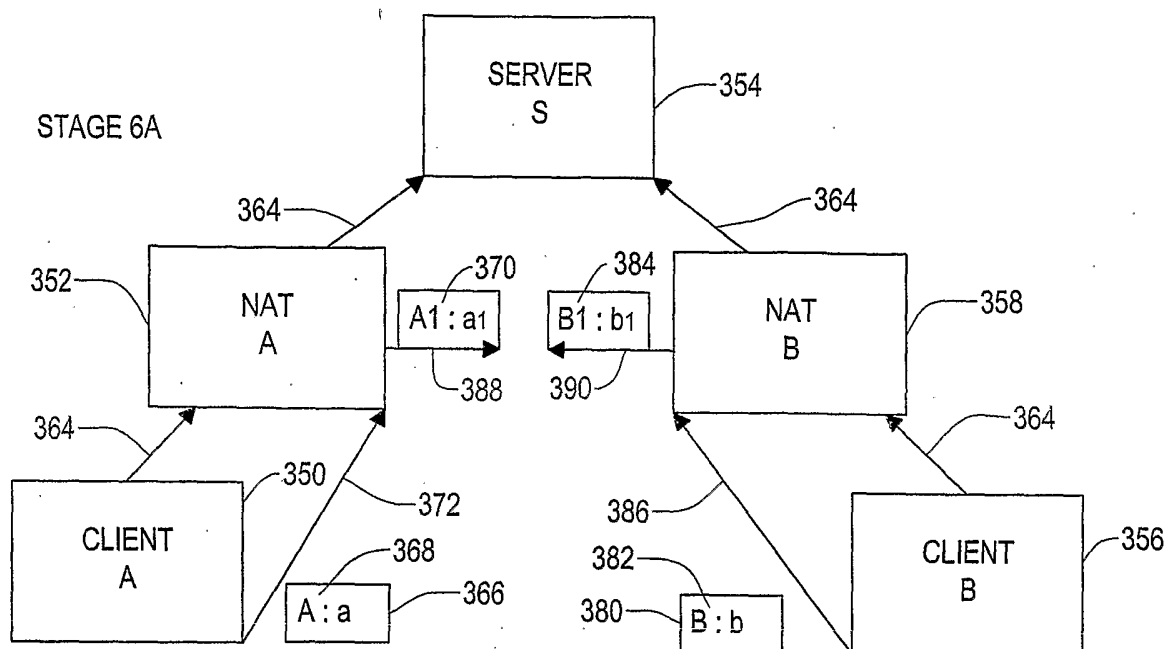
- CLIENT B SENDS UDP PACKETS S2 : s2. SERVER SEES THEM AS COMING FROM B1 : b1
- SERVER S CAN FORWARD PACKETS RECEIVED
- FROM A1 : a1 TO B1 : b1 AND
FROM B1 : b1 TO A1 : a1 WHICH WILL MAKE A AND B THINK THEY HAVE A DIRECT CONNECTION

13 / 29

FIG. 9f

- SERVER S INFORMS A THAT B IS TALKING ON B1 : b1
AND B THAT A IS TALKING ON A1 : a1
- DEPENDING ON THE APPLICATION A TALKS TO B1 : b1 OR BOTH B1 : b1 AND S1 : s1
B TALKS TO A1 : a1 OR BOTH A1 : a1 AND S2 : s2

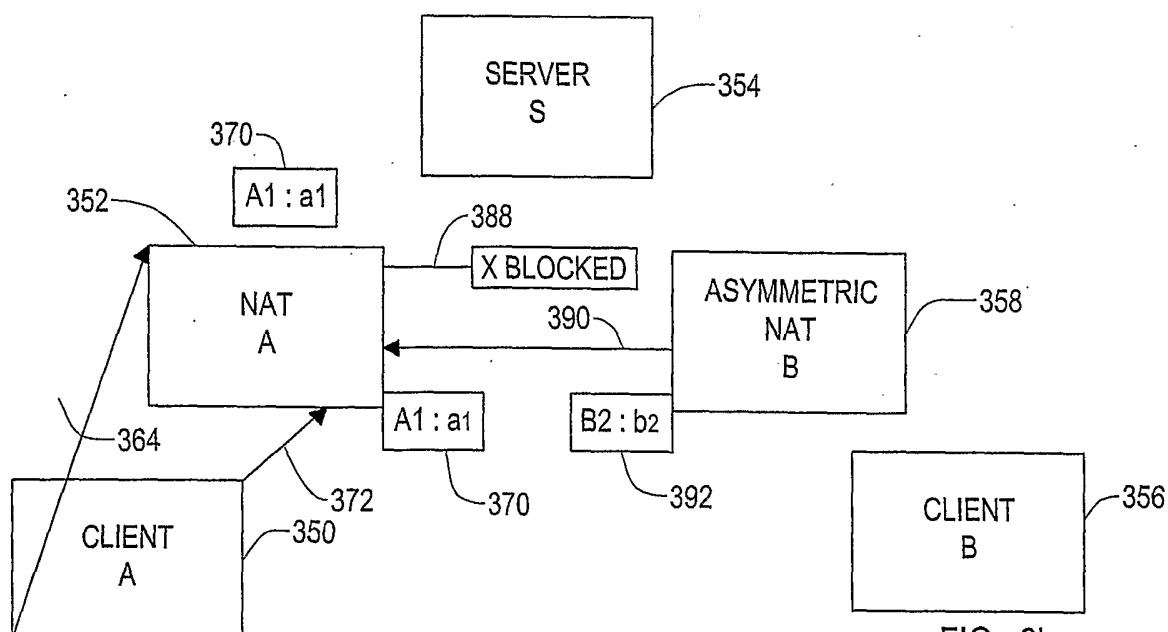
14 / 29



- IF NAT A IS SYMMETRIC, B WILL RECIEVE TRAFFIC FROM A1 : a1
- IF NAT B IS SYMMETRIC, A WILL RECIEVE TRAFFIC FROM B1 : b1
- WHEN SUCH TRAFFIC IS DETECTED , SERVER S IS INFORMED AND BREAKS CONNECTION. SO IF NAT A AND NAT B ARE SYMMETRIC WE ARE LEFT WITH THE ABOVE.
- THEN TCP CONNECTIONS TO S COULD BE DROPPED DEPENDING ON THE APPLICATION

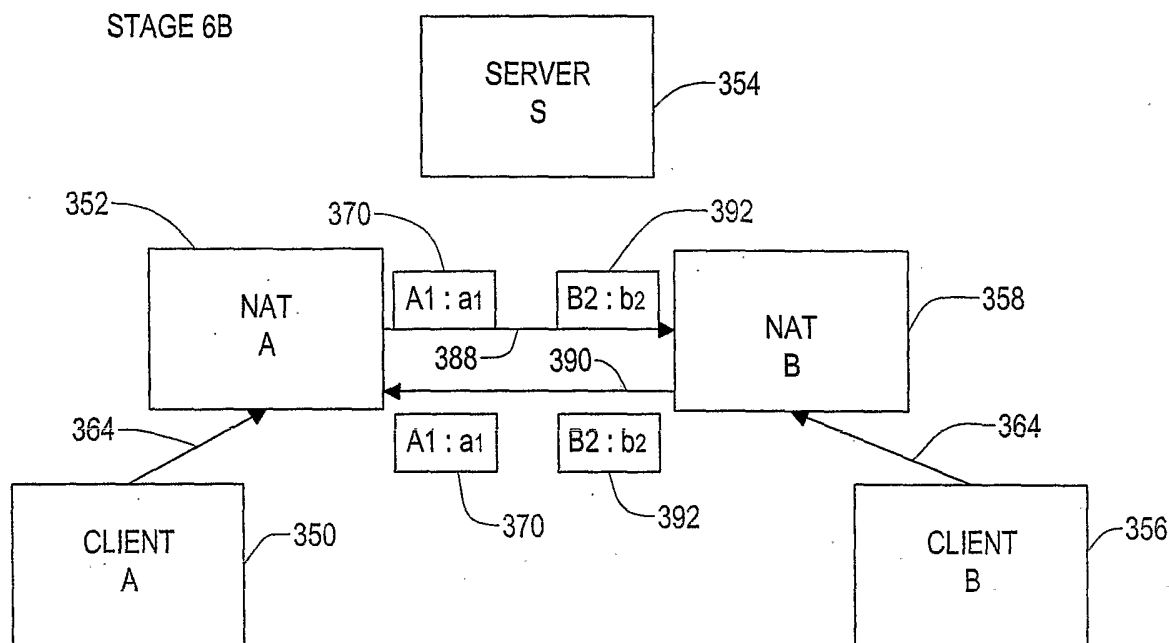
FIG. 9g

- IF NAT B IS ASYMMETRIC , THEN OUTPUT FROM B : b DIRECTED AT A1 : a1 WILL APPEAR TO COME FROM B2 : b2 TRAFFIC FROM A1 : a1 TO B1 : b1 WILL BE BLOCKED BY NAT B.

FIG. 9h

15 / 29

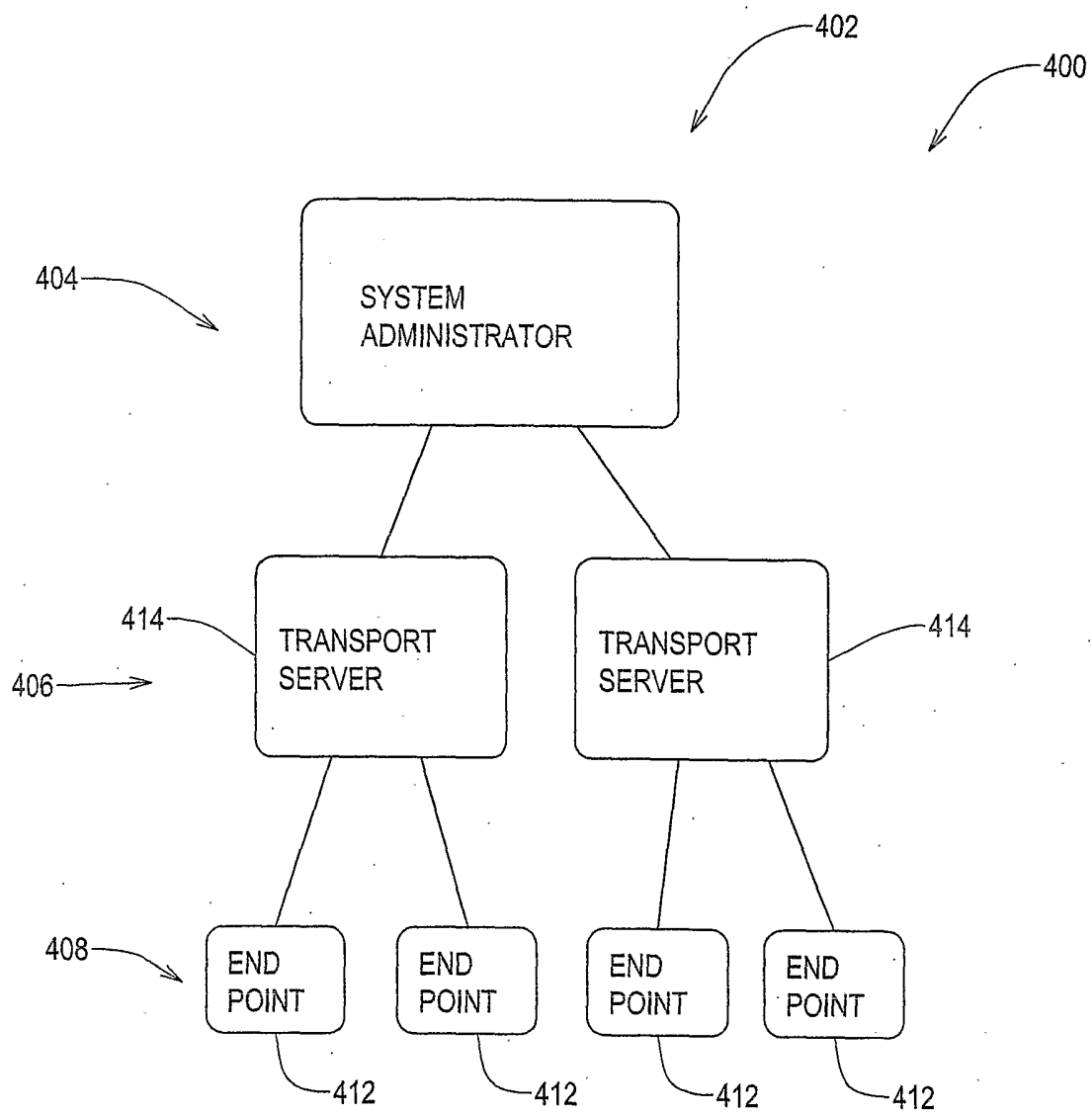
- CLIENT A SEES TRAFFIC COMING FROM B2 : b2. IT THEREFORE SWITCHES ITS OUTPUT TO B2 : b2 NAT B HAS USED THIS ADDRESS TO WRITE TO A1 : a1, SO IP PASSES



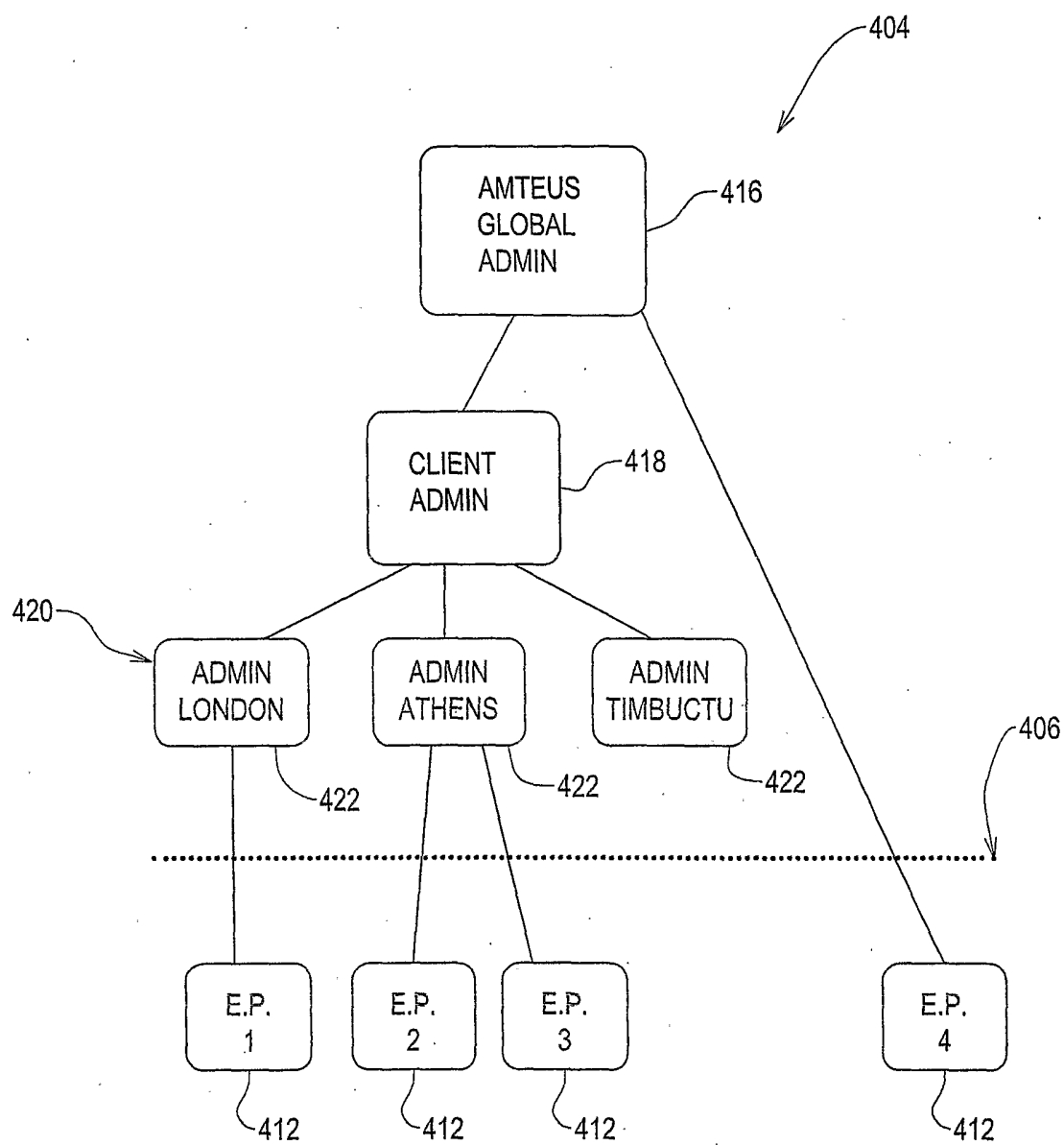
- SO BI-DIRECTIONAL COMMUNICATION IS ESTABLISHED
- IF NAT A IS ASYMMETRIC AND NAT B SYMMETRIC THE SAME PROCESS OCCURS IN REVERSE
- IF BOTH NAT A AND NAT B ARE ASYMMETRIC, THEN PEER TO PEER COMMUNICATION IS NOT POSSIBLE. TRAFFIC CONTINUES, OR IS RE-ESTABLISHED VIA THE SERVER AS AN INTERMEDIARY. NOTE THAT ANY OTHER ADDRESS VISIBLE TO BOTH NAT A AND NAT B CAN BE SUBSTITUTED AS INTERMEDIARY.

FIG. 9i

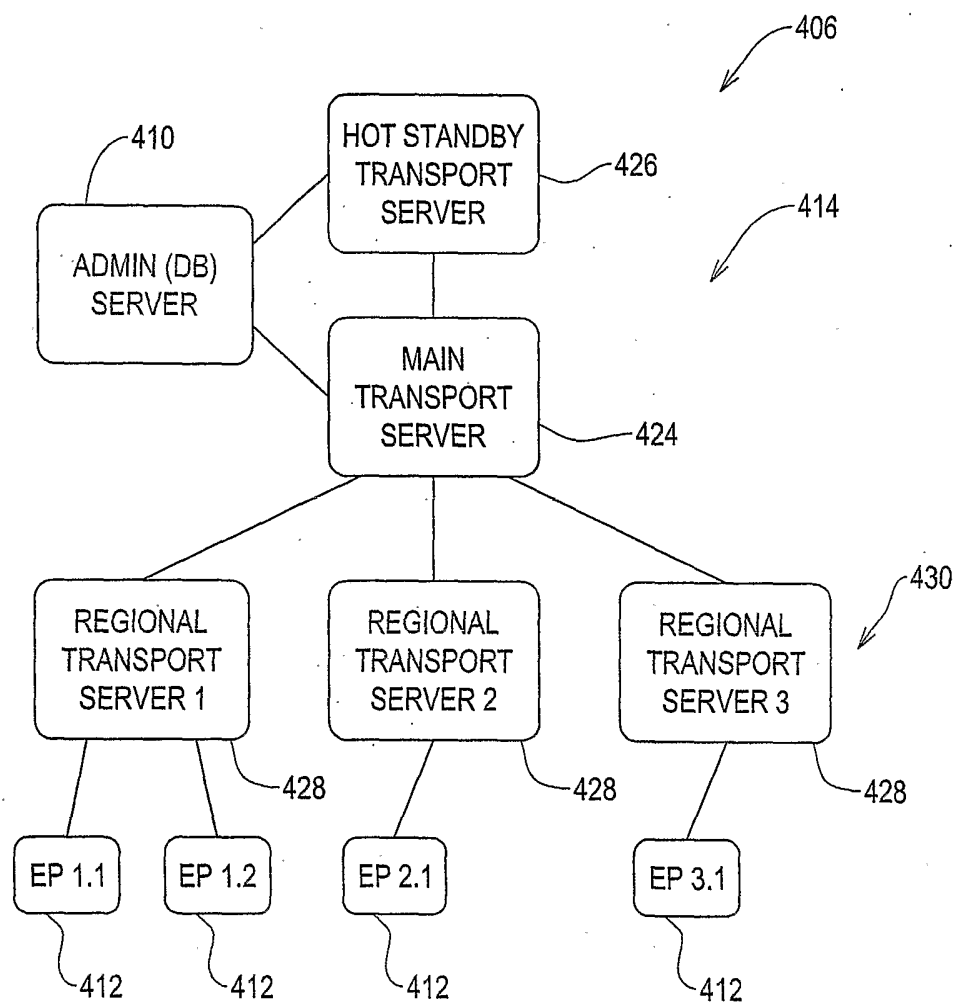
16 / 29

FIG. 10

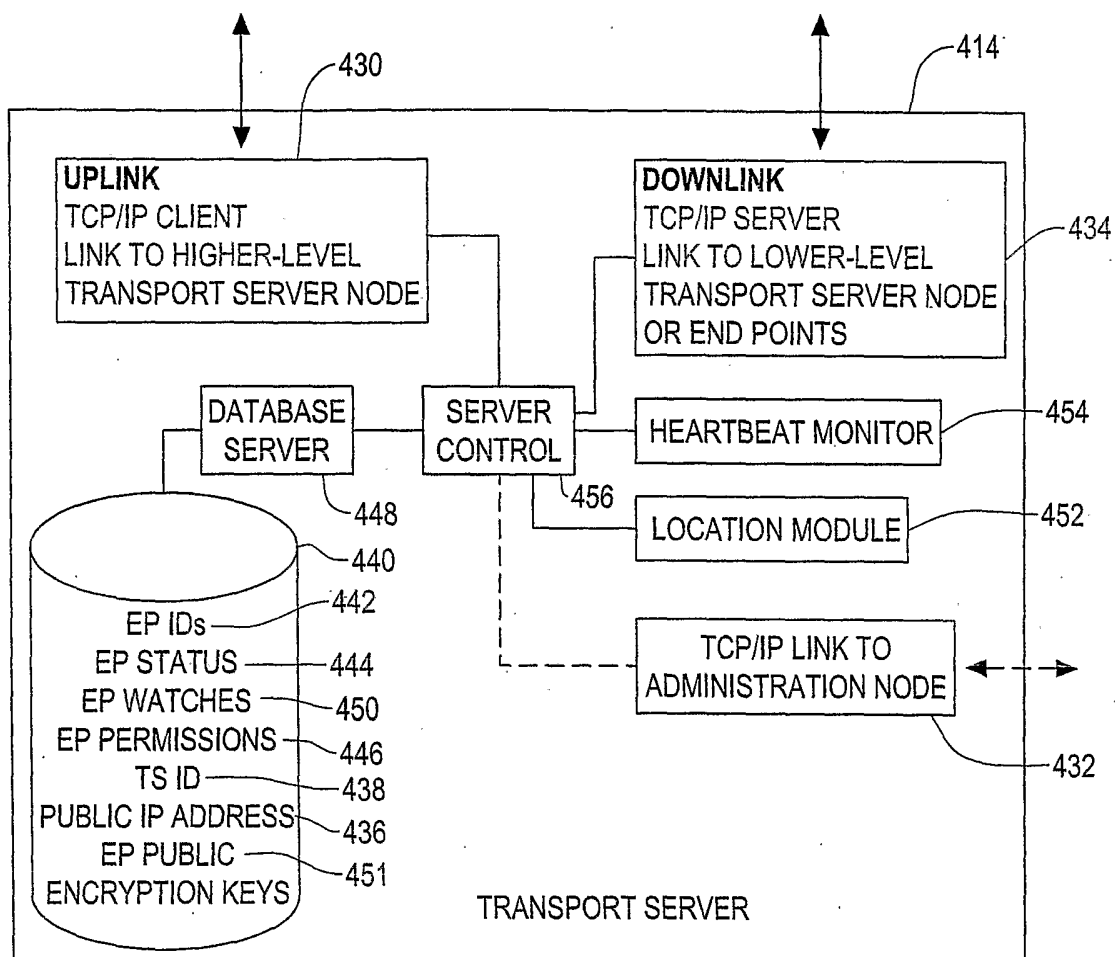
17 / 29

FIG. 11

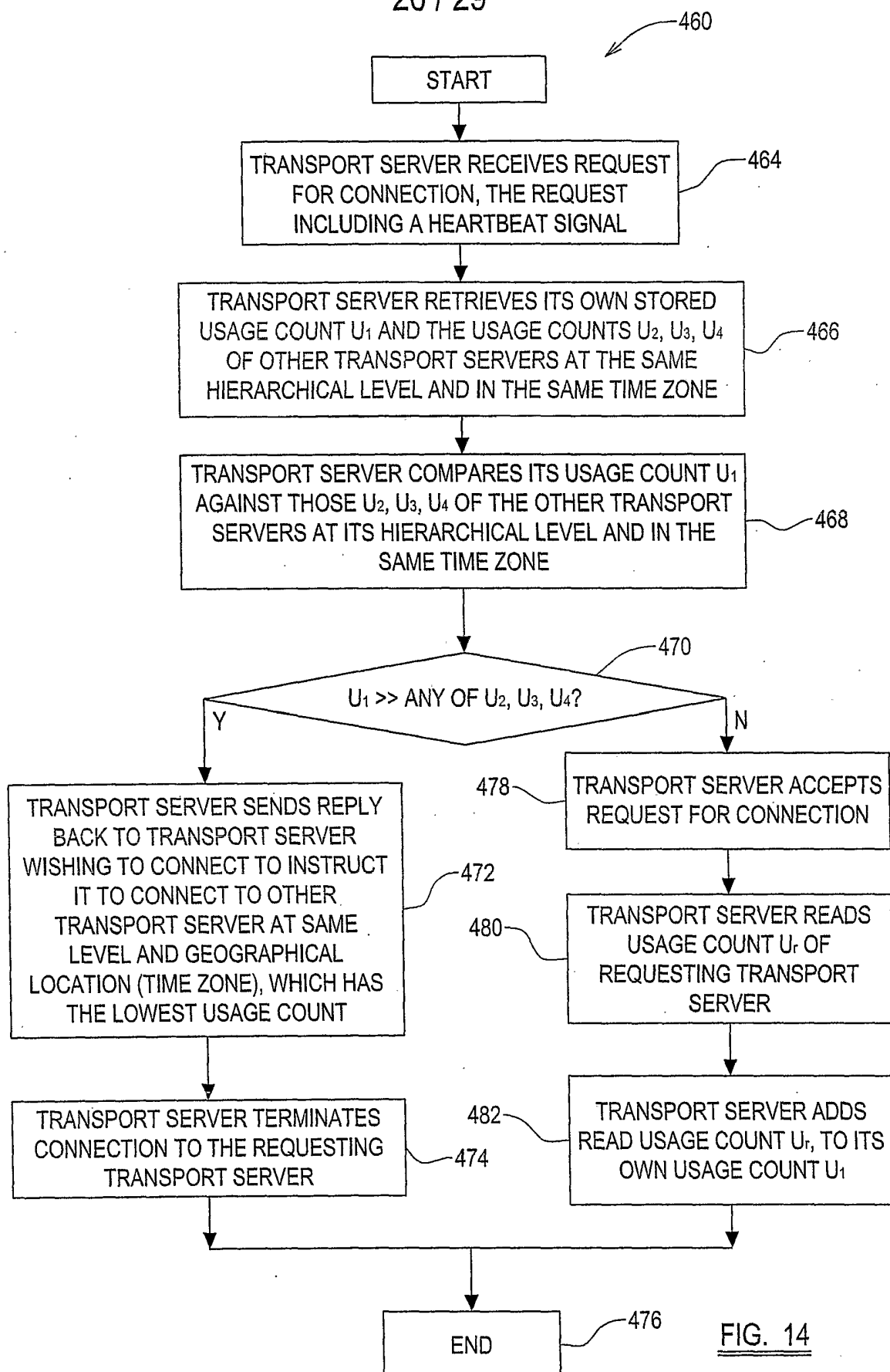
18 / 29

FIG. 12

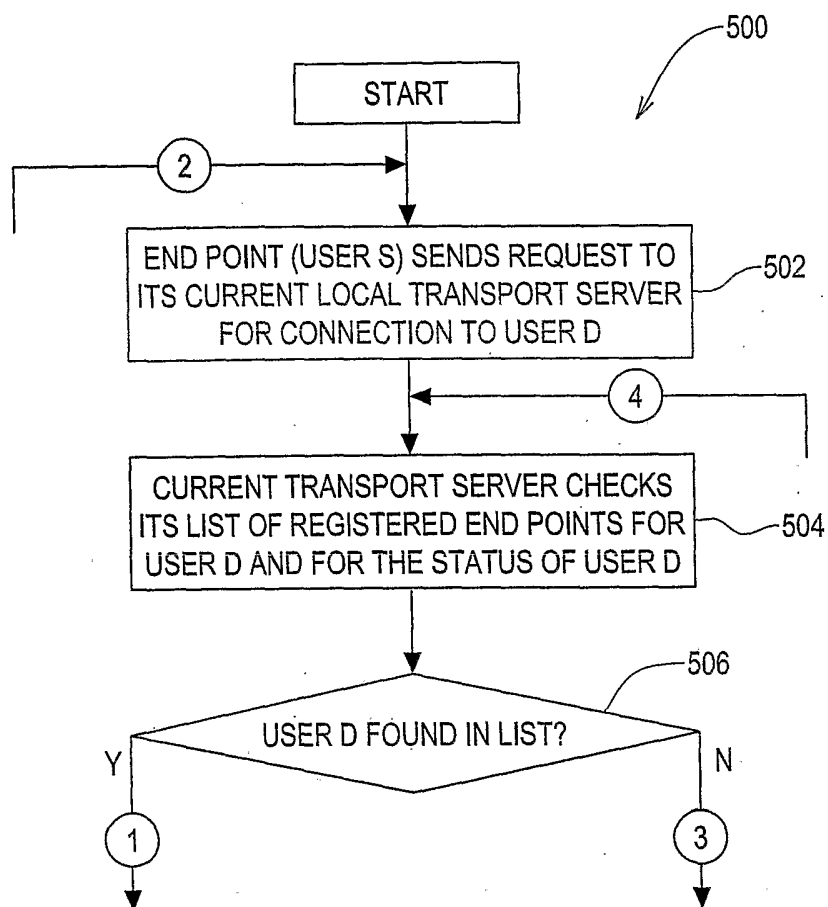
19 / 29

FIG. 13

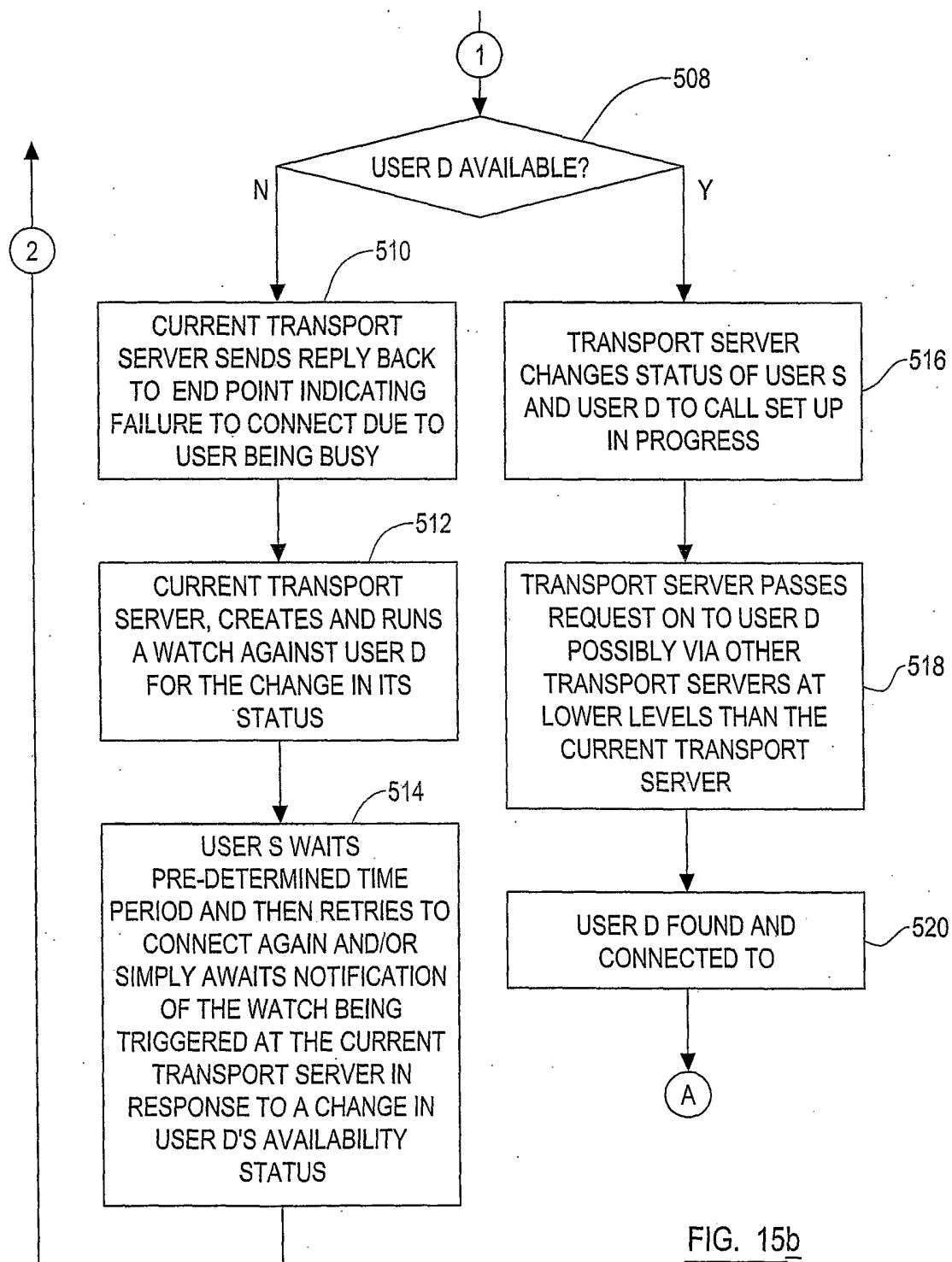
20 / 29



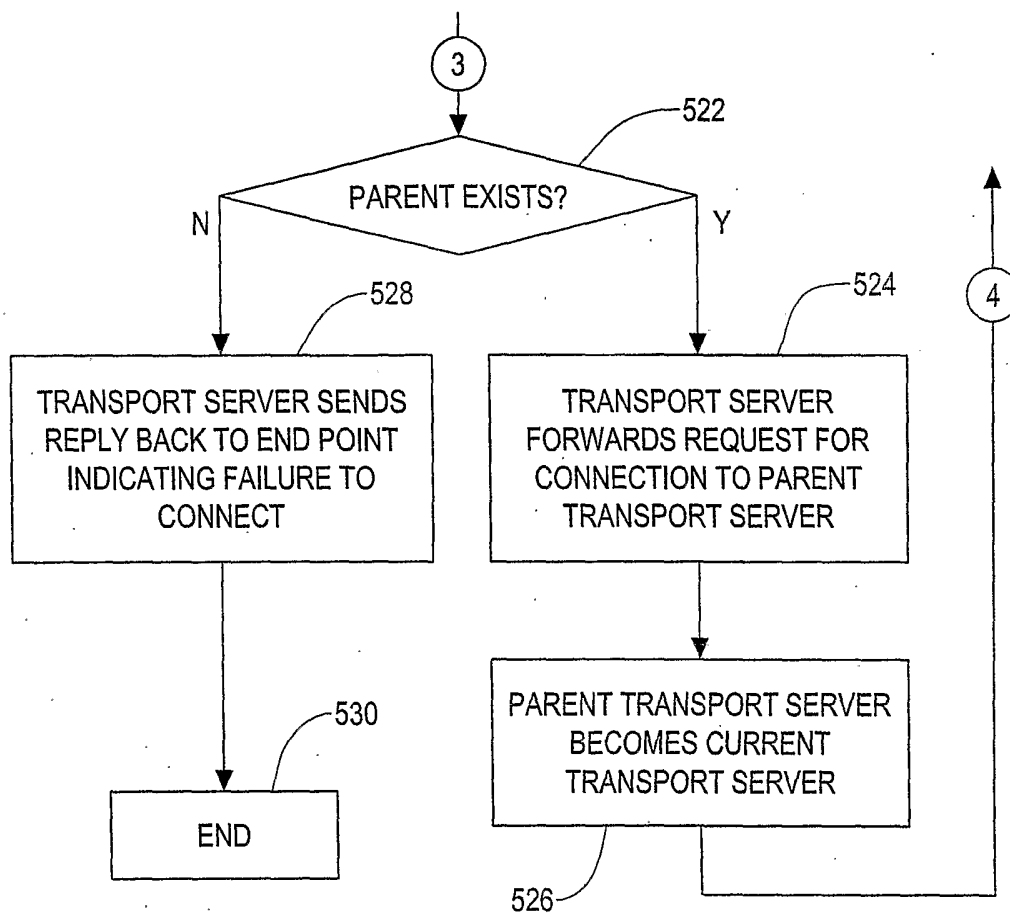
21 / 29

FIG. 15a

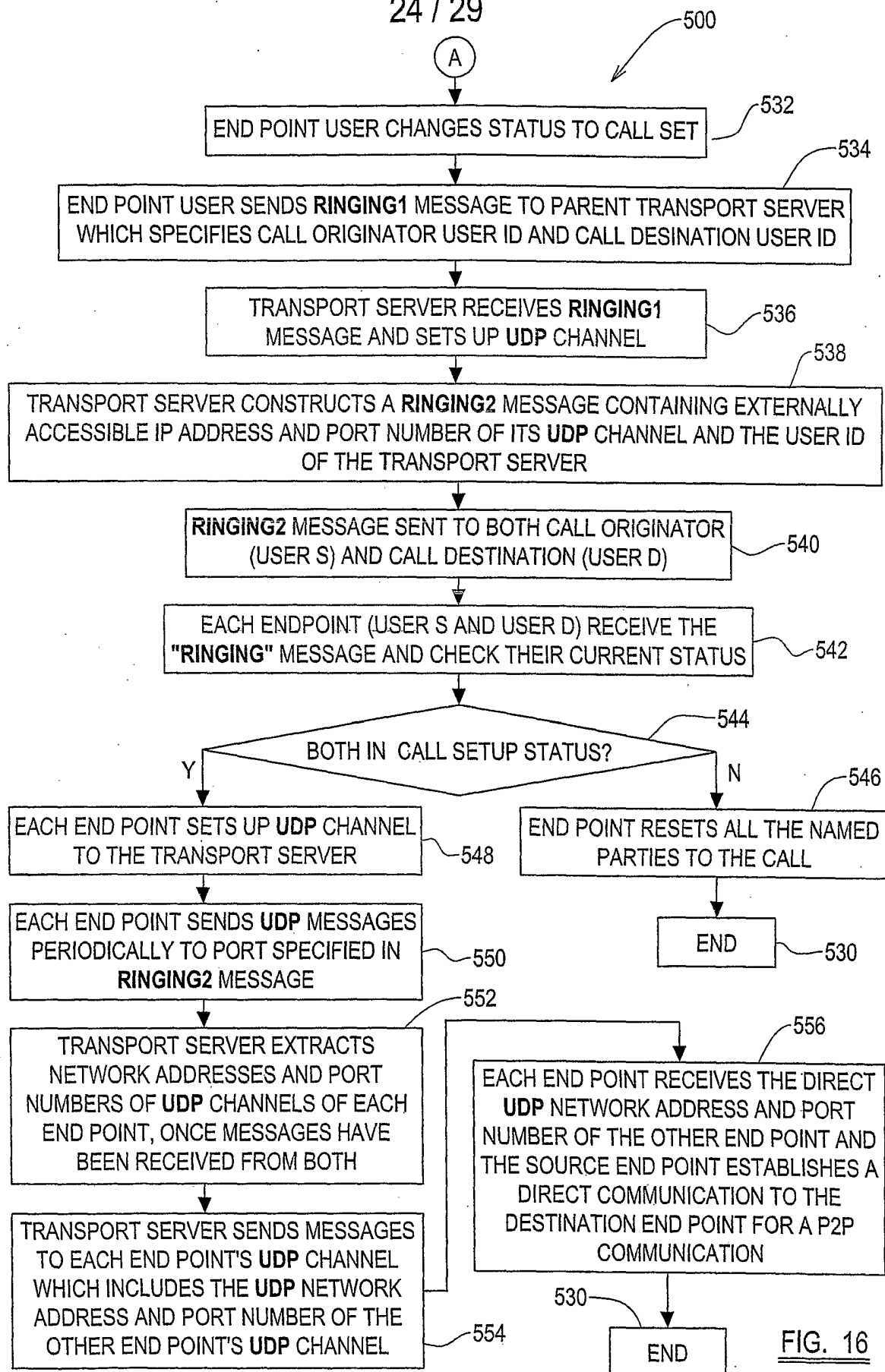
22 / 29



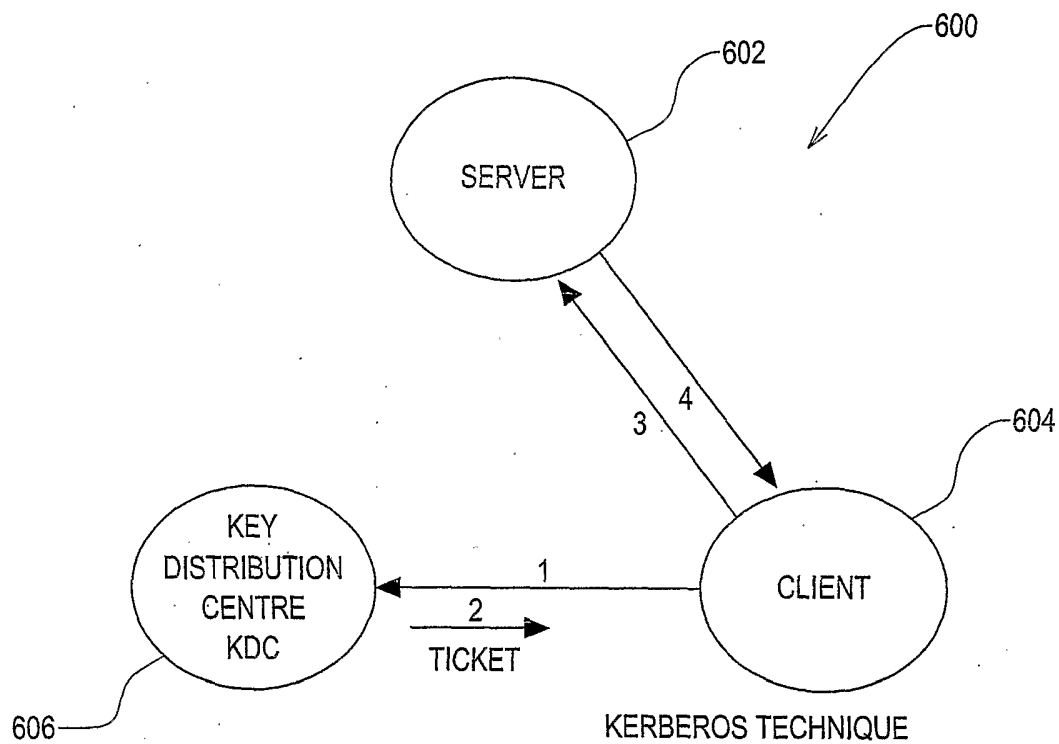
23 / 29

FIG. 15c

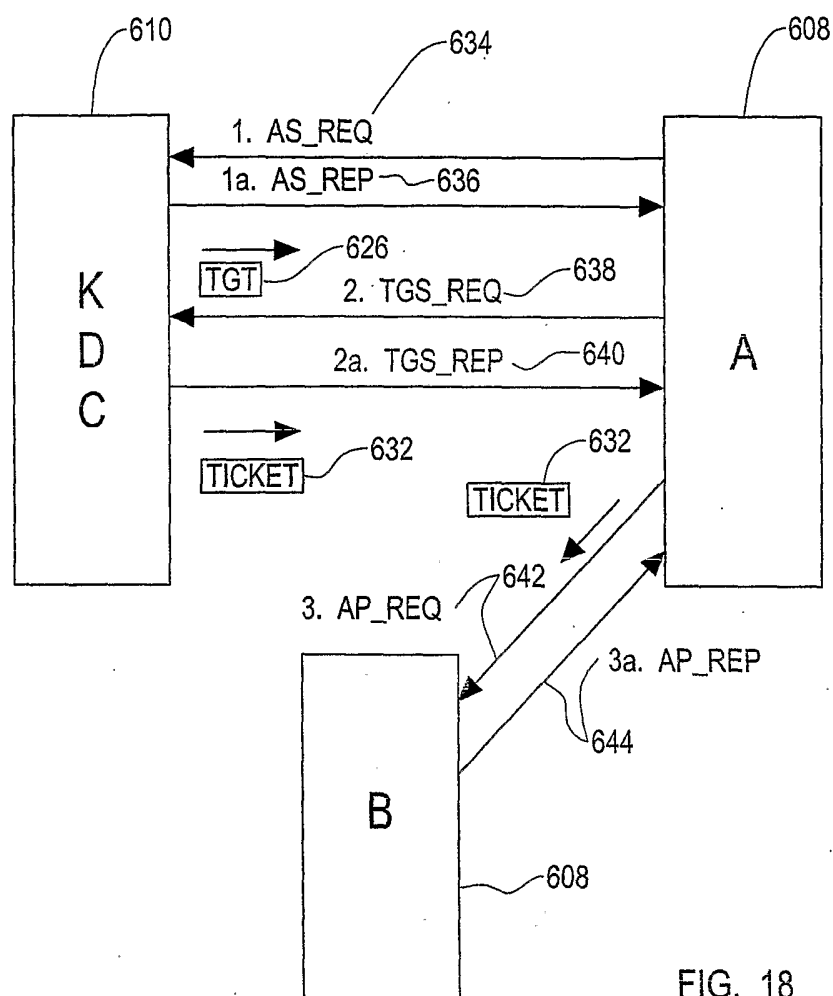
24 / 29



25 / 29

FIG. 17

26 / 29

FIG. 18

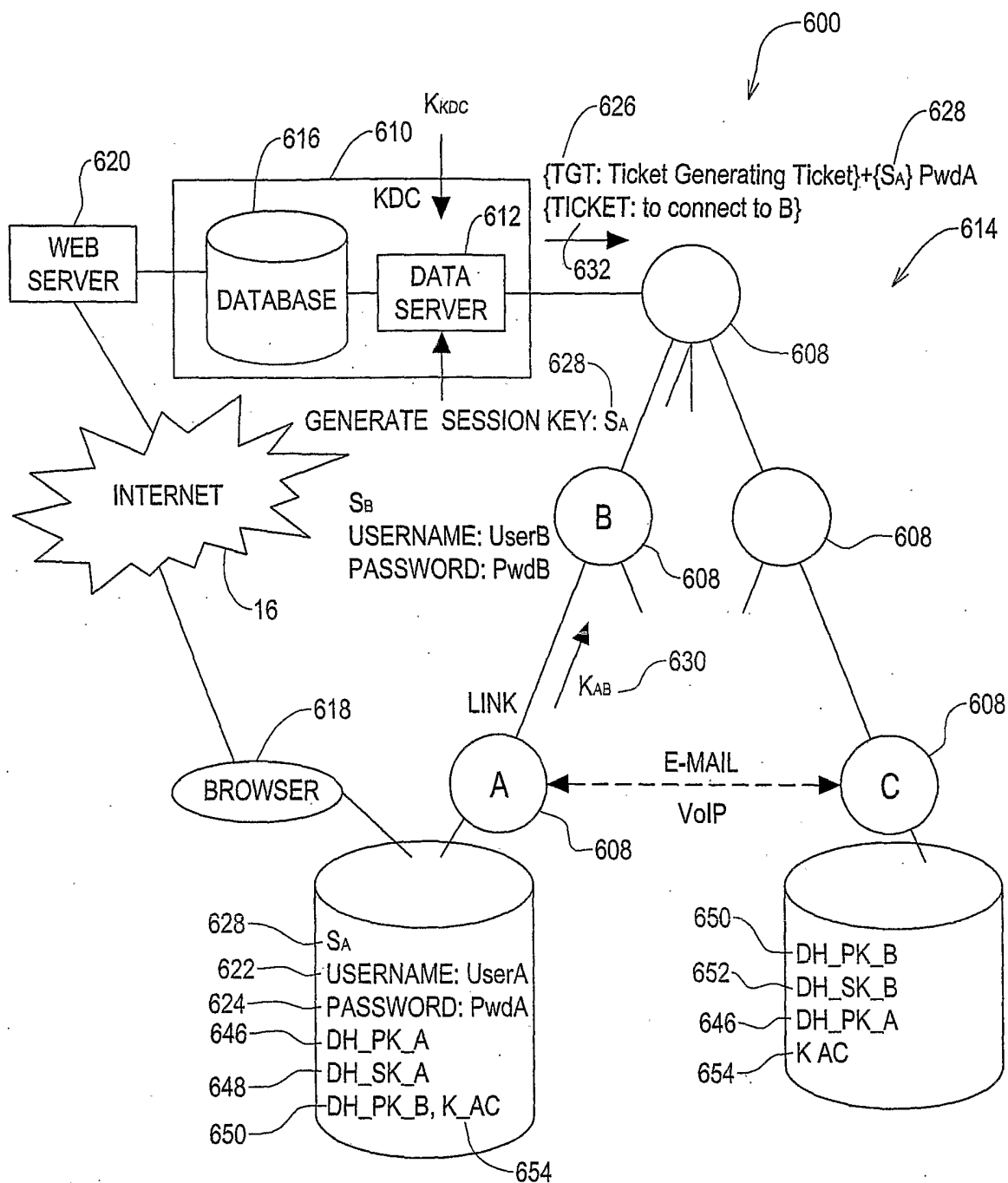
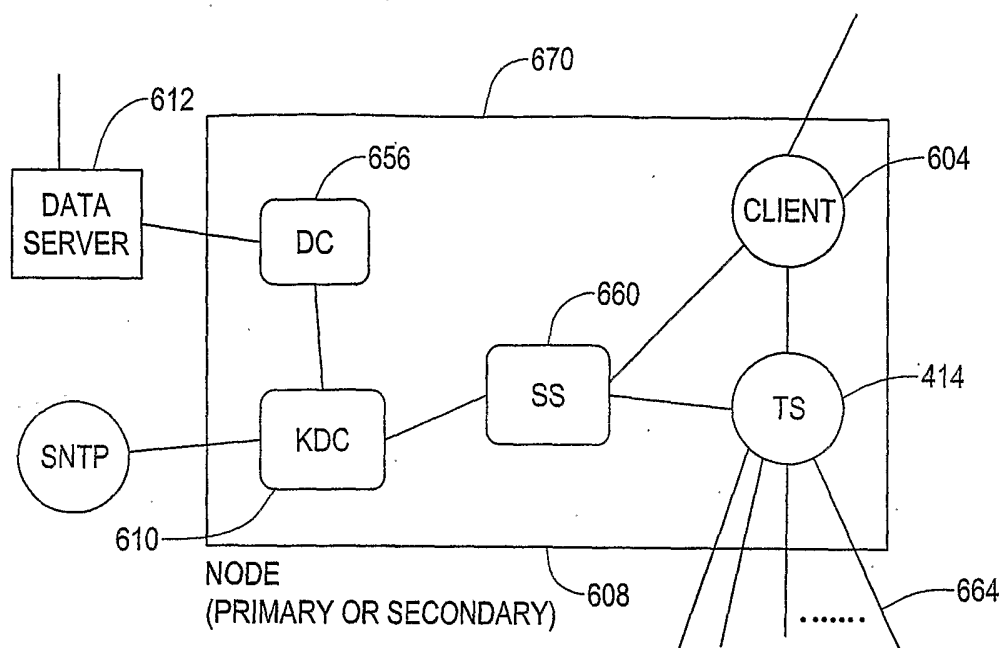
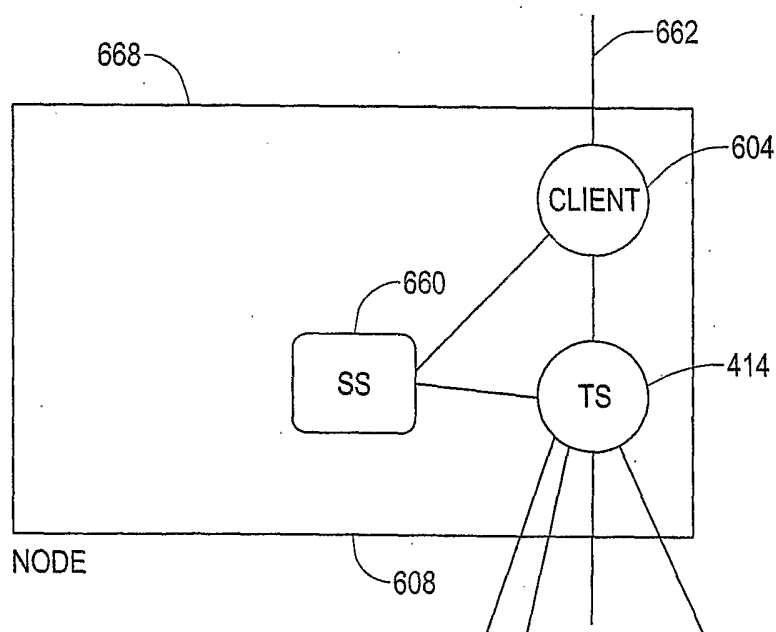


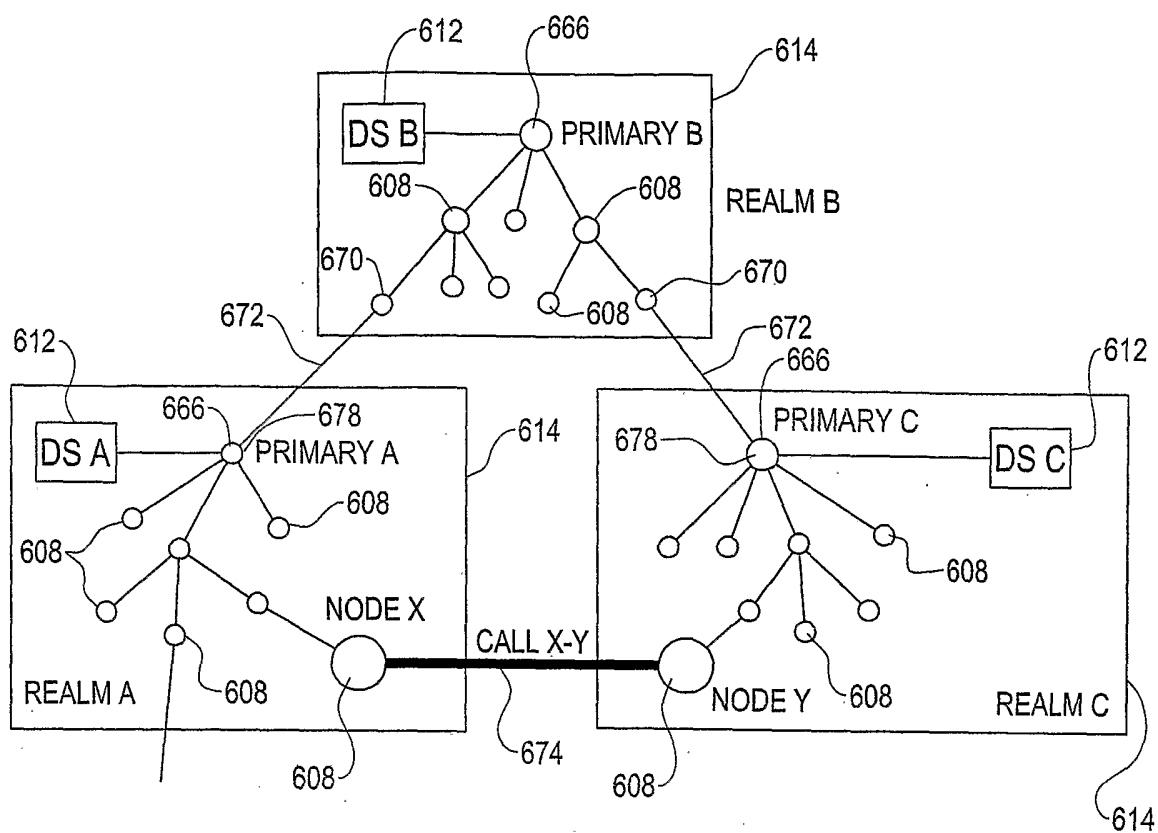
FIG. 19

28 / 29

FIG. 20aFIG. 20b

KDC - Key Distribution Centre SNTP = Simple Network Time Protocol
 SS = Security Services TS = Transport Server
 DC = Data Client

29 / 29

FIG. 21