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

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2525249 C 2011/03/29

(11)(21) 2 525 249

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2004/04/15
(87) Date publication PCT/PCT Publication Date: 2004/12/02
(45) Date de délivrance/Issue Date: 2011/03/29
(85) Entrée phase nationale/National Entry: 2005/11/09
(86) N° demande PCT/PCT Application No.: GB 2004/001629
(87) N° publication PCT/PCT Publication No.: 2004/104902
(30) Priorité/Priority: 2003/05/22 (US10/443,675)

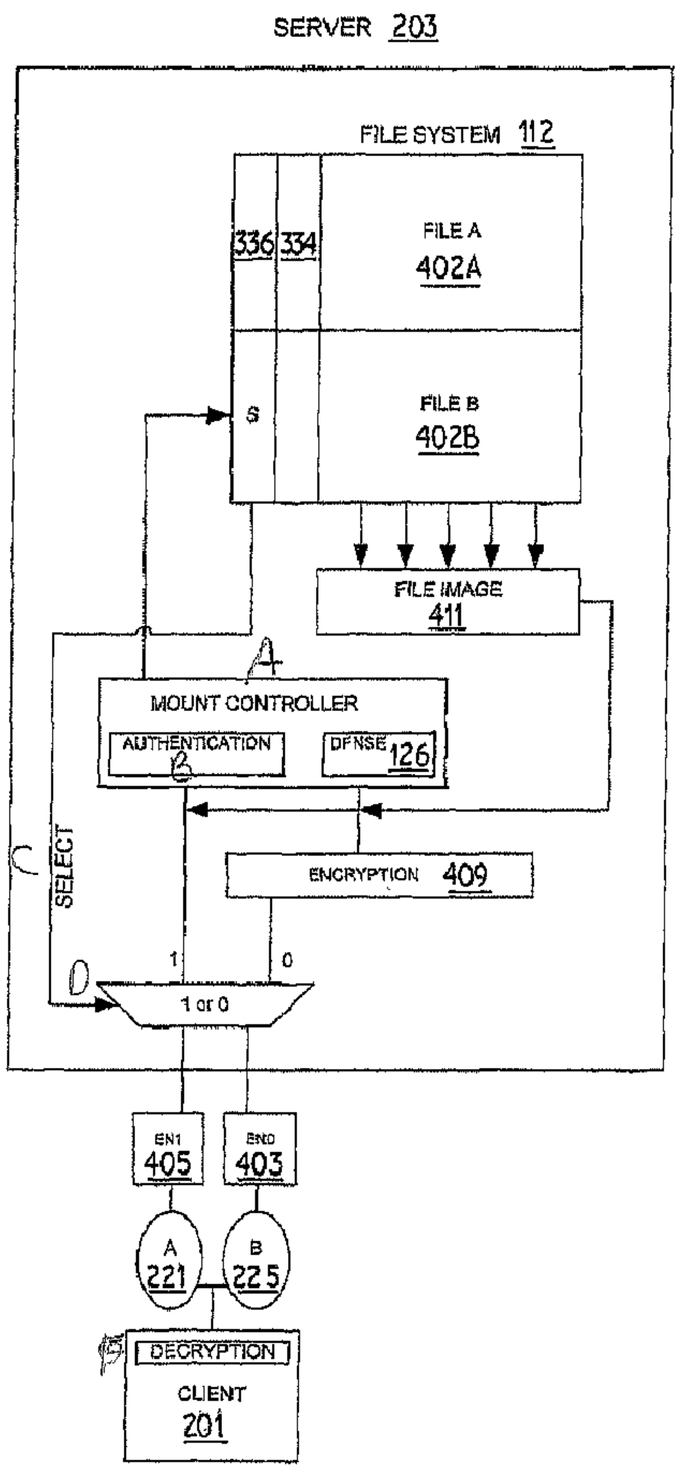
(51) Cl.Int./Int.Cl. *H04L 29/06* (2006.01), *G06F 1/00* (2006.01), *G06F 17/30* (2006.01)

(72) Inventeurs/Inventors:
KEOHANE, SUSANN MARIE, US;
MCBREARTY, GERALD FRANCIS, US;
MULLEN, SHAWN PATRICK, US;
MURILLO, JESSICA KELLEY, US;
SHIEH, JOHNNY MENG-HAN, US

(73) Propriétaire/Owner:
INTERNATIONAL BUSINESS MACHINES CORPORATION, US

(74) Agent: WANG, PETER

(54) Titre : EXTENSION DE LA SECURITE D'UN RESEAU DE FICHIERS REPARTI
(54) Title: DISTRIBUTED FILESYSTEM NETWORK SECURITY EXTENSION




(57) Abrégé/Abstract:
A security protocol that dynamically implements enhanced mount security of a filesystem when access to sensitive files on a networked filesystem is requested. When the user of a client system attempts to access a specially-tagged sensitive file, the server

CA 2525249 C 2011/03/29

(11)(21) 2 525 249

(13) C

(57) Abrégé(suite)/Abstract(continued):
hosting the filesystem executes a software code that terminates the current mount and reconfigures the server ports to accept a re-mount from the client via a more secure port. The server reconfigured server port is provided the IP address of the client and matches the IP address during the re-mount operation. The switch to a secure mount is completed in a seamless manner so that authorized users are allowed to access sensitive files without bogging down the server with costly encryption and other resource-intensive security features. No significant delay is experienced by the user, while the sensitive file is shielded from unauthorized capture during transmission to the client system.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 December 2004 (02.12.2004)

PCT

(10) International Publication Number
WO 2004/104902 A1

(51) International Patent Classification[7]: G06F 21/00, H04L 29/06, G06F 17/30, 1/00

(21) International Application Number: PCT/GB2004/001629

(22) International Filing Date: 15 April 2004 (15.04.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/443,675 22 May 2003 (22.05.2003) US

(71) Applicant *(for all designated States except US)*: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(71) Applicant *(for MG only)*: **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth Hampshire PO6 3AU (GB).

(72) Inventors; and

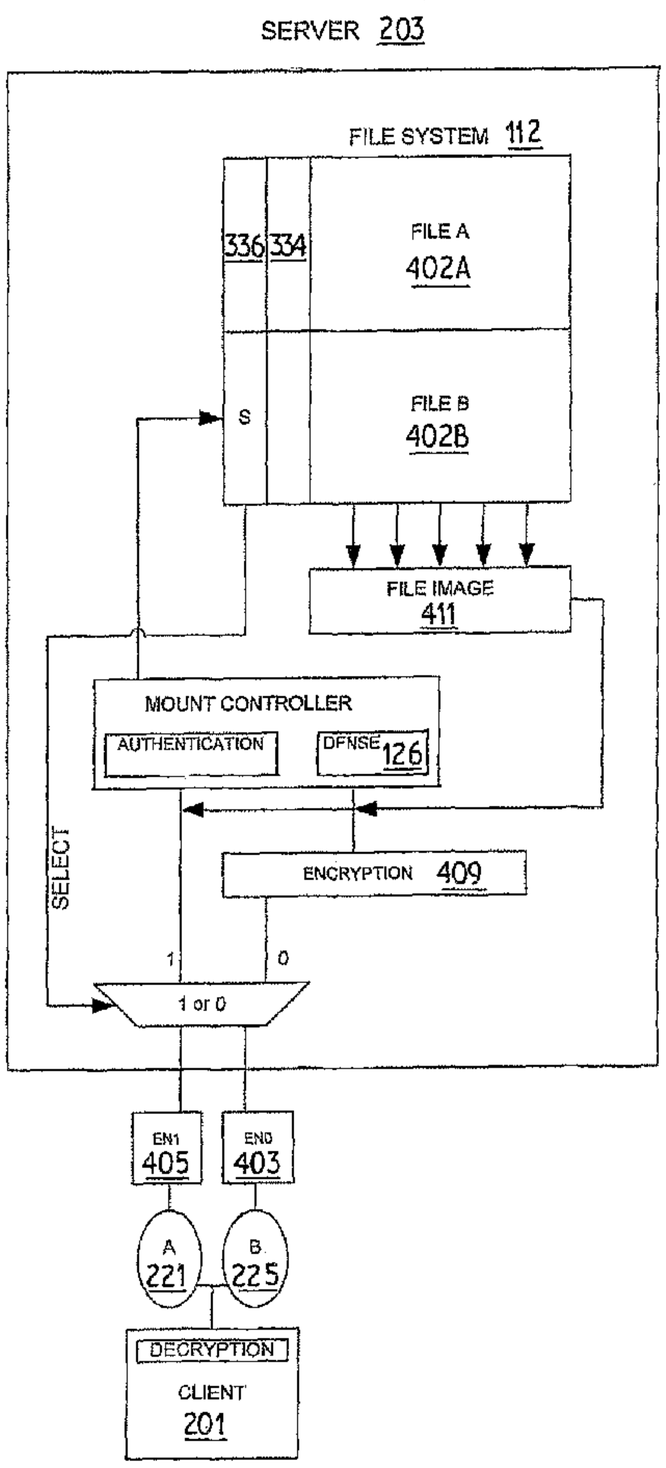
(75) Inventors/Applicants *(for US only)*: **KEOHANE, Susann, Marie** [US/US]; 1911 Brackenridge Street, Austin, Texas 78704 (US). **MCBREARTY, Gerald, Francis** [US/US]; 10709 Bayridge Cove, Austin, Texas 78759 (US). **MULLEN, Shawn, Patrick** [US/US]; 39 Country Oaks, Buda, Texas 78610 (US). **MURILLO, Jessica, Kelley** [US/US]; 980 Country Road 109, Hutto, Texas 78634 (US). **SHIEH, Johnny, Meng-Han** [US/US]; 5908 Upvalley Run, Austin, Texas 78731 (US).

(74) Agent: **LING, Christopher, John**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM,

*[Continued on next page]*

(54) Title: DISTRIBUTED FILESYSTEM NETWORK SECURITY EXTENSION




(57) Abstract: A security protocol that dynamically implements enhanced mount security of a filesystem when access to sensitive files on a networked filesystem is requested. When the user of a client system attempts to access a specially-tagged sensitive file, the server hosting the filesystem executes a software code that terminates the current mount and reconfigures the server ports to accept a re-mount from the client via a more secure port. The server reconfigured server port is provided the IP address of the client and matches the IP address during the re-mount operation. The switch to a secure mount is completed in a seamless manner so that authorized users are allowed to access sensitive files without bogging down the server with costly encryption and other resource-intensive security features. No significant delay is experienced by the user, while the sensitive file is shielded from unauthorized capture during transmission to the client system.

WO 2004/104902 A1

WO 2004/104902 A1

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

**(84) Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, 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, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

**Published:**
— *with international search report*

*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.*

# DISTRIBUTED FILESYSTEM NETWORK SECURITY EXTENSION

## Field of the Invention

The present invention relates in general to network systems and in particular to distributed filesystems. Still more particularly, the present invention relates to security features for access to distributed filesystems.

## Background of the Invention

In general purpose computing systems, such as those supporting versions of the Unix operating system (OS), applications may access data stored on disk drives by means of a set of operating system services including a filesystem (Unix is a registered trademark, exclusively licensed through the Open Group). A filesystem may be employed by a computer system to organise a large collection of files into individual files and directories of files and to map those files to storage devices such as disks. Filesystems comprise two primary components, the programs that control the physical representation of the files and the files themselves that are stored on the disk.

In a distributed computing environment, a number of computing systems can be interconnected by way of a communication network or other coupling facility and can share files by way of a distributed filesystem. A filesystem exporter is typically executed on the server node (the computing system that controls access to the disk containing the filesystem data), while a filesystem importer is typically executed on the client nodes (other computing systems utilised to access the files on the disk). Accesses to shared files by users on the client nodes are referred to as "remote" accesses. Accesses to shared files made by users on the server node are referred to as "local" accesses.

The network filesystem is stored on a server or node of a network, and the server or node is accessible from client terminals (i.e., user computers) that are typically remotely linked to the network. The actual link may be a wired link, as in a standard Ethernet-based local area network (LAN) or a wireless link, such as a Bluetooth Virtual Private Network (VPN). The process of accessing the filesystem via the client terminals is referred to as "mounting a filesystem." When a filesystem is mounted, the filesystem's control program reads certain information from the disk concerning the layout of filesystem objects. From this

information, the filesystem constructs data structures known as "virtual filesystems" or Vfs's. Each time a file is opened, or made accessible, the filesystem creates a data structure, referred to as a "vnode", which is chained to the vfs.

Each vnode contains information about a given file and contains references to physical file system data structures. The physical file system data structures contain information such as the owner of the file, the size of the file, the date and time of the file's creation and the location of the blocks of the file on the disk. Filesystems include internal data, called meta-data, to manage files. Meta-data may include data that indicates: where each data block of a file is stored; where memory-modified versions of a file are stored; and the permissions and owners of a file.

With more and more companies using remote/network-accessible distributed filesystems to electronically store and later retrieve files/documents, including some with sensitive information, security of distributed filesystems is becoming increasingly important. The IP Security (IPSec) suite of standards was introduced and provides two primary security features: authentication and encryption. In other words, IPSec ensures that sending and receiving machines really are what they claim to be, and IPSec enables data to be scrambled in flight so the data will be incomprehensible if intercepted.

Most systems thus require an authentication of the user during the initial mount, which typically includes verifying user-passwords, etc. However, password-protection and similar security measures are notorious for being open to cracking and can easily be compromised, and the industry has recognised that password-protected systems offer very little protection to sensitive files once general access to the filesystem is obtained.

More advanced hackers also gain access to the files stored on the filesystem by tapping into a transmission during an authorised mount and simply copying the data as it is being transmitted from filesystem to client system. This occurs because, with most password-protected distributed filesystems, once the several levels of security log-in (password verification, etc. are completed, the actual transmission of the files from the filesystem occurs in clear text. Thus, when the transmission includes very sensitive data, additional security measures

are required to ensure that the clear text data is not available by simply copying the file during transmission.

The ease at which the security of the sensitive information may be compromised via this latter method depends to some extent on the medium being utilised by authorised users to mount/access the filesystem. For example, wireless access/transmission is typically more prone to eavesdropping and cracking that wire-full (wired) network media. However, even the standard Ethernet can easily be breached without detection, and thus the standard Ethernet is also an unsafe option for routing sensitive data.

As mentioned above, the industry has responded to the growing need for security on the transmission medium by imposing heavy encryption on all transmitted data during a mount of the filesystem. Currently, there are several encryption algorithms and standards (e.g., wireless transport layer security) designed to provide security for the transmissions between client system/node and the server hosting the filesystem. Utilization of heavy encryption requires placing a heavy processing burden on the client system and the server for all traffic. The overall performance of the system is degraded, and significant costs are incurred by companies that wish to implement system-wide encryption for access to their filesystem. Encryption is built into the communication mechanisms and applied to all traffic between client system and server although the majority of traffic may not require that level of security (e.g., non-sensitive information/files).

The utilization of wireless systems to access filesystems is increasing as companies provide remote access to users who may be mobile and wish to connect to the network remotely. Wireless connections are, however, more susceptible to cracking than wired connections. Some wireless users use WTLS, but this security feature is known to be a relatively weak level of security. One solution requires a Virtual Private Network (VPN) data encapsulation/encryption to access sensitive data, even when the majority of clients are accessing the filesystem via token ring. This VPN data encapsulation would further negatively impact the speed of the servers as they encrypt and decrypt all data.

It is also possible to configure VPNs or servers on a VPN to recognise IP addresses or subnets and only require encryption on certain subnets. One problem with this solution is that the administrator of the distributed filesystem server must have knowledge of every wireless node that is not within the network. If a wireless network is set up by an

organisation within their department, the server administrator would need to be made aware of the wireless network so that the subnet could be added to the VPN list of IP addresses.

In light of the foregoing, the present invention recognises that it would be desirable to have a method, system and data processing system that dynamically implements enhanced mount security when access to sensitive files on a distributed filesystem is requested. A method and system that would automatically provide a secure mount whenever sensitive file/data are about to be accessed during an ongoing session would be a welcomed improvement. It would be further desirable if the secure mount was completed in a seamless manner so that the authorised user receives access to the sensitive file without experiencing a disconnect and re-mount authentication process, while the sensitive file is shielded from unauthorised capture by routing the sensitive file via the more secure mount.

## DISCLOSURE OF THE INVENTION

Disclosed is a method, system and computer program product that dynamically implements enhanced mount security of a filesystem when access to sensitive files on a networked filesystem is requested. The client system initiates a standard mount and authentication process for access to files of the filesystem. When the user of the client system attempts to access a specially tagged sensitive file, the server executes a software code that terminates the current mount. The server is re-configured to route to a secure port any attempts to re-mount the server from the IP address associated with the client. When a session is terminated by the server, the client system is programmed to automatically attempt to re-mount the server. The server recognizes the IP address of the client during the remount operation and routes the client to the secure port.

A secure mount is thus automatically provided whenever sensitive files/data are about to be accessed during an ongoing session that was initiated on a standard mount. Then routing via a secure mount is completed in a seamless manner so that the authorized user receives access to the sensitive file without experiencing significant delay or a visible disconnect that requires user-initiated remount and authentication processes. Meanwhile the sensitive file is shielded from unauthorized capture by routing the sensitive file via the more secure mount established.

## BRIEF DESCRIPTION OF THE DRAWINGS

The invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

**Figure 1A** is a block diagram of a data processing system within which the features of the invention may be implemented;

**Figure 1B** is a block diagram representation of the files within the filesystem of **Figure 1A** having a security tag indicating a required level of security according to one embodiment of the invention;

**Figure 2** is a block diagram of a distributed network within which features of the invention may be implemented according to one embodiment of the invention;

**Figure 3A** is a flow chart of the process by which a client is provided access to sensitive files during access via a standard mount according to one embodiment of the invention;

**Figures 3B** and **3C** are flow charts of the processes by which a server monitors and controls client requests for access to sensitive files to ensure that access to those files is routed via a secure channel according to one embodiment of the invention; and

**Figure 4** is a block diagram illustrating logic components for seamlessly completing a switch of a client session on a filesystem from a standard, non-secured channel to a secured channel during a single continuing session in accordance with one embodiment of the invention.

## DETAILED DESCRIPTION OF THE INVENTION

With reference now to the figures and in particular with reference to **Figure 1A**, there is illustrated a block diagram of a computer system, which may be utilised as either a server hosting the distributed filesystem or a client system utilised to mount the server at which the distributed filesystem is hosted. Computer system **100** comprises processor **102** and memory **104** connected via system bus/interconnect **110**. Computer system **100** also comprises Input/Output (I/O) Channel Controller (CC) **109**, which is coupled to interconnect **110**. I/O CC **109** provides connection to I/O devices **106**, including Redundant Array of Disk (RAID) **114**. RAID **114**

stores instructions and data that are loaded to memory as needed by the applications being executed by the processor. According to the illustrative embodiment, RAID **114** provides the storage media for the plurality of files that constitute filesystem **112**.

Computer system **100** further comprises network connection devices **108**, which may include wireline modem, wireless modem, and Ethernet card, among others. Access to and from I/O devices **106** and network connection devices **108** are routed through I/O channel controller (I/OCC) **109**, which includes logic for completing the automatic re-establishment of a mount to/from computer system **100** via a "secure" path/channel/port when required, as further described below.

Computer system **100** includes operating system (OS) **122**, filesystem software application **124**, mount code **125** and Distributed Filesystem Network Security Extension (DFNSE) **126**. Filesystem software application **124** provides the basic accessing, maintaining, and updating of filesystem **112**, when computer system **100** is being utilised to host a filesystem **112**.

When within a client system, filesystem software application **124** includes client version of mount code **125** for completing a mount and automatic re-mount of the server hosting the filesystem. In the illustrative embodiment, the automatic remount process is implemented by the client system whenever an established mount with the server is disrupted/lost without the client having completed an unmount of the server. In the described embodiment, the server may issue a FYN command to terminate a current mount and thus force the client to initiate a re-mount of the server. The FYN command is issued in response to access to particular files that require special security protections, as will be explained in greater detail below.

Returning to **Figure 1A**, and the description of filesystem software application **124**, when executed within a server, filesystem software application **124** includes code for receiving, maintaining, and verifying credential information of various users and client systems, code for maintaining filesystem **112** and code for selectively initiating a security software and associated response, called Distributed Filesystem Network Security Extension (DFNSE) **126**. DFNSE **126** provides the backbone of the inventive features herein and the execution of DFNSE **126** on a server is described below with reference to **Figures 3A-3C** and **4A**. At a basic level, DFNSE **126** determines what level of security access is permitted/authorised

for specific files of the filesystem **112** and when to initiate the enhanced security measures of the invention.

With reference now to **Figure 2**, there is illustrated an example network comprising multiple interconnected computer systems, which may be similarly configured to computer systems **100** of **Figure 1** that are advantageously utilised to provide either the server or client functionality for respectively hosting or accessing a distributed filesystem. Network **200** comprises a distributed filesystem **202** hosted on three (or more) interconnected servers **203**. Network **200** also comprises a plurality of client systems **201** connected to distributed filesystem **202** via a network backbone **210**. Network backbone **210** comprises one or more network-connectivity systems (or sub-networks) that may be configured according to network protocols such as Ethernet or TokenRing. These sub-networks may be, for example, wire-line or wireless local area networks (LAN) or a wide area network (WAN), such as the Internet. Additionally, sub-networks may include fibre optic network as well.

Distributed filesystem **202** is directly coupled to network backbone **210** via one or more ports (not shown) on at least one of the servers **203**. Client systems **201** may be either directly coupled to the network backbone (wireline) or communicatively connected via a wireless medium illustrated by wireless antenna **207**. Client systems **201** access the distributed filesystem **202** via one of the various available media utilising one of the various network configurations, each one having a different level of susceptibility to cracking. Thus, client system **201** may access and mount filesystem **202** via a non-secure wireless network **227**, or client system **201** may mount filesystem **202** utilising a secure fibre-optic network **225**. For simplicity of describing the invention, the wireless network **227** will be assume to be a standard, non-secure network without encryption, while the fibre-optic network **225** is assumed to be a special, secured connection with encryption. Each connection is routed via a different one of the ports available to the server **203** at which the mount of filesystem **202** is supported.

**Figure 1B** illustrates a block diagram representation of filesystem **202** with a more detailed delineation of the files that comprise filesystem **202**. As illustrated, filesystem **202** comprises a control block **131** and a plurality of files **132a-n**, each of which includes a metadata tag **112** with header/identifier field **334** and security field **336**. Header/ID field **334** contains information about the file ID and the users who have access to

the file. Security field is a single bit field, which indicates the level of security attributable to that file and consequently the type of user-access permitted. According to the illustrative embodiment, certain files that require highest levels of security and which are restricted to being accessed solely on a secured mount (e.g., files 1 and 3) are tagged with "1" in the security field **336** of their respective metadata. Other files not so tagged (i.e., tagged with a 0) are normal (e.g., file 2) and may be access by any authorised user without a special secured mount.

As mentioned above, the invention introduces an enhanced security mechanism, which, in the illustrative embodiment, is referred to as DFNSE (Distributed Filesystem Network Security Extension). With DFNSE, the filesystem server is able to infer from the file permissions associated with a file or directory the level of network security that is required when providing access to the file by particular users. DFNSE is a server-level filesystem security enforcement application and/or procedure. Accordingly, with DFNSE, only the server is required to have knowledge of the networks connections or adapters being utilised by the server.

Specific hardware, logic and software components are provided within each server capable of providing a mount to the filesystem to implement DFNSE. **Figure 4** is a block diagram illustrating some of these components. As illustrated in **Figure 4**, server **203** may be provided with two Ethernet network adapters (or ports): en0 **403**, which is secure, and en1 **405**, which is not secure. In one embodiment, the network topology behind these adapters is consistent. That is, the sub-network selected itself provides the security and the server is able to dynamically select between sub-networks based on the level of security required. In another embodiment, additional encryption or other security features are provided with the secure adapter, en0 **403**.

En0 **403** connects to a fibre network **225** that is utilised to route all sensitive data, while en1 **405** connects to a standard Ethernet-based wired network **221** and is utilised for routing all other (non-sensitive) data communication. En1 **405** is the default port for mounting the filesystem's server. The exemplary embodiment assumes that the ease at which the standard Ethernet can be breached without detection makes the Ethernet an unsafe option for routing sensitive data. During a mount of the filesystem, the server, which has detail knowledge of the file permissions **334** and security level **336** for each file tracks user access and determines when to force the client to switch over to the secure network based on the file permissions in place for the files being

accessed. According to the illustrative embodiment, the network topology is consistent for both the secure and non-secure routes so that no additional hardware and/or routing protocol upgrades are required to account for different topologies during the switch from non-secured to secured sub-network.

Server **203** of **Figure 4** also includes a mount controller **407**, which performs conventional mount support and unmount operations for filesystem **202** as well as re-mount configuration in accordance with the features of the present invention. Mount controller **407** includes DFNSE **126** and is preferably embodied as software code executing on server **203**. DFNSE **126** operates to trigger mount controller **407** to route a request for a mount via either standard port, En1 **405**, or secured port, En0 **403**. Server **203** also includes encryption module **409** utilised in conjunction with DFNSE **126** and En0 **403**, when encryption is implemented on secured port.

During filesystem access, when a request for access to sensitive operation is received at server **203**, mount controller **407** marks the client's IP address as one needing access to sensitive data. Server **203** then breaks the current connection, (i.e., the server sends a FYN to the client). The client automatically attempts to reconnect, and mount controller **407** recognizes the client IP address during the re-mount. The client's session is then directed to a secure SSL port. Thus, while primary access is provided via the standard port, access is dynamically switched to the SSL secure port when access to sensitive data/files is required.

Turning now to **Figure 3B**, there is illustrated a flow chart of the process by which the software-implemented DFNSE security features are implemented within the above hardware/logic configuration of the server hosting the filesystem. The process begins with a standard mount request received at the standard port of the server from a client as shown at block **321**. The user is prompted for authentication data (password, etc.) and the client system's IP address is retrieved from the data packet as indicated at block **323**. A session is opened on the standard port and both the IP address and the user's authentication data are stored within a parameter file linked to the particular session as shown at block **325**. Once the session is established, i.e., server logic associated with access permissions, etc., monitors the user's interaction as shown at block **327** and a determination made at block **329** whether the user is requesting access to a sensitive file.

The server that is satisfying a clients request for a file is programmed with the authorization/credentials of the user on the remote client and the permissions of the file being accessed. If the file being accessed is not sensitive, regular access is provided to the user on the standard port as shown at block **331**. However, when the file being requested by the client is a sensitive file that requires a more secure channel before access can be granted, a next determination is made at block **333** whether the user has proper access permission to access the file. If the user does not have proper access permission the request is denied as shown at block **335**. If, however, the user's credentials indicates the user has permission to access the particular file, the DFNSE security protocol is activated as shown at block **337**. Activation of DFNSE causes the server to force an unmount of the client by issuing a FYN to the client and concurrently configuring a more secure port to accept the re-mount from the client having the IP address saved with the session parameters. The server then provides secured access to the file via the secured port as shown at block **339**.

**Figure 3A** is a flow chart of the processes involved in the implementation of the invention primarily from the perspective of the user/client system. The process begins when a user (via the client system) first mounts a server hosting the filesystem as illustrated at block **302** and requests access to the filesystem as shown at block **304**. When the client initially mounts the NFS filesystem, the mount is completed over a standard TCP connection by default. For example, the connection may be to the well known NFS port of 2048. The server has a listening socket bound to this port and operates according to the standard (non-secure) protocol. Notably, in the illustrative embodiment, the standard protocol is enhanced by the DFNSE protocol, which is implemented when access to sensitive files are requested.

When a connection is requested from the client, the listening socket of the server basically duplicates itself and bounds the connection to the remote client. The listening socket then remains open to handle other connection requests. Authentication of the client is initiated as shown at block **306** and a determination made at block **308** whether the client's authentication was successful. If the client/user authentication process is unsuccessful, access to the filesystem is denied and the mount is disconnected as shown at block **310**. Then the process ends as indicated at block **311**. Otherwise, a session is opened and the user is provided access to the filesystem as shown at block **309**.

The client system monitors the connection for disconnects as shown at block **312** and determines as indicated at block **314** whether the connection becomes unresponsive or is prematurely broken at the server side (i.e., ideally when a server issued FYN is received). When the connection becomes unresponsive or broken, the client initiates a remount that is routed to the port indicated by the server as shown at block **316**.

Notably, reconnection in response to a server-initiated dismount is directed on a secure port at the server, although the actual port may be unknown to the client system. Utilizing the security protocols of DFNSE and based on the knowledge of which port is secure and whether the session requires a secure port, the server is able to request that the client re-mount over a secure port. For example, the client may be made to re-mount utilizing a port running Secure Socket Layer. Notably, no user-action is required to complete the re-mount and port-switching procedures. The monitoring for server-side unmount and subsequent re-mount all occur as background processes at the client system, and the user (client) is not made aware of the switch to a more secure port.

A more detailed account of the internal processing required for rerouting via a more secure port at the server side is illustrated by the flow chart of **Figure 3C**. The process begins when access to a sensitive file is identified by DFNSE as shown at block **351**. The server checks the current port security as indicated at block **352**. A determination is made at block **354** whether the current port security is sufficient for accessing the requested file (depending on how sensitive the file is, which is deduced by reading the security bit of the file in the illustrative embodiment). If the current port security is sufficient for accessing the requested file, the access is provided as shown at block **356**.

In one alternate embodiment, the remount function may be selectively automated and the process would require a next determination whether the feature for automatic remount is enabled. With this alternate embodiment, if the automatic remount capability is not enabled, the user will actually be prompted to remount via a secure mount.

Returning to the illustrated embodiment of **Figure 3C**, when the port security is insufficient, the server responds by selecting a more secure port (e.g., En0) for the session as shown at block **358**. The server takes a snapshot of the authentication and mount parameters of the session, including the client's IP address, and transfers these parameters to the control logic of the more secure port as depicted at block **360**. The transfer occurs with very little latency and the more secure port is thus

automatically configured to receive a re-mount from that client and continue supporting the session in progress. After the secure port has been configured, the corresponding port number is given to the mount controller along with the IP address of the client. The server terminates the mount on the first standard port and re-establishes the session via the more secure port when a re-mount is received from the client as shown at block **362**.

Notably, in response to the server terminating the initial mount, the client initiates the re-mount which is directed by the user to the second, secured link. This re-establishes the initial session of the client but via the second port. Re-establishing the connection involves checking the clients IP address and matching it to the port that is set up to receive the connection from that IP address. The entire process occurs in the background and thus a seamless switching of ports is completed from the user's perspective.

In one alternate embodiment, the level of security attributable to a particular file is determined by the users (or selected client systems) that are provided access to the particular file. Thus, if file access permissions are restricted to filesystem administrators only, then the security level is high, while file access permission given to regular employees indicates a relatively low level of security required. Determination of the security level for a file is completed when the user initially creates the file and assigns the access permission to that file. Once the file is placed within the filesystem, the file automatically inherits the network security protection that is in place. With this implementation, existing file permissions on files within the filesystem (e.g., UNIX -rwx,rwx,rwx for user, group, other) fold into the security model provided herein without requiring extensive system administration and configuration. Thus, the present invention eliminates the need for reconfiguring existing filesystems on a per file basis. With the invention, there is also no requirement to move sensitive files to a secure server.

# CLAIMS

1. A method for providing security for transmission of at least a first file, the method being for use in a data processing system comprising (1) a storage medium on which is stored said at least a first file having a preset access permission, (2) at least a first standard port and second secure port for connecting said data processing system to external client systems, (3) a logic component for selectively routing transmission of said at least one file via said first port and said second port, and (4) a reconfiguration logic component for configuring said first standard port and said second secured port for supporting a mount by said client system, said method comprising:

responsive to a request for access to said first file by said external client system, checking said preset access permission of said first file; and

when said preset access permission of said first file indicates secured access is required for said first file, dynamically routing a transmission of said first file to external client system via said second port, said dynamic routing step comprising:

first configuring said second secure port to support a remount operation received from said client system;

terminating a current mount on said first standard port with said client system; and

storing session parameters of said current mount to enable seamless continuation of said session on said second secure port.

2. The method of Claim 1, further comprising:

routing said transmission of said first file via said first standard port when said preset access permission indicates a regular access is sufficient.

3. The method of Claim 1, further comprising:

enabling a first mount of said data processing system via said first standard port; and

enabling a second mount of said data processing system via said second secure port only when said first file requires secured access.

4. The method of Claim 1, wherein said data processing system further comprises an encryption module associated with said second secured port, said dynamic routing step comprising:

first encrypting said first file utilizing said encryption module.

5. The method of Claim 4, wherein said configuring and storing step includes:

retrieving an IP address of said client system;

placing said IP address in a configuration of said second secure port, wherein said second secure port automatically recognizes a remount operation from said client system and re-establishes the session with said client system.

6. The method of Claim 1, wherein said preset access permission is a bit within metadata linked to said first file and said method further comprises reading a value of said bit to evaluate whether said first file requires secure access.

7. The method of Claim 1, wherein said preset access permission includes an identification of which specific users are permitted to access said first file via a secured access, said method further comprising:

comparing a user of said client system with said specific users with permission to access said file; and

when said user is one of said specific users, automatically initiating a re-routing of a transmission of said first file via said second secure port.

8. The method of Claim 1, wherein said first standard port connects to said client system via a first unsecured network and said second secure port connects to said client system via a second secured network.

9. The method of Claim 1, wherein:

said data processing system is a server within a network having a first subnet connecting said first standard port to said client system and a second subnet connecting said second secure port to said client system;

said first file is stored within a filesystem;

said checking step includes accessing said filesystem and locating said first file; and

said routing step includes transmitting said file via said second subnet when said file requires secure access and transmitting said first file via said first subnet when said first file does not require secure access.

10. A system for providing security for transmission of at least a first file, for use in a data processing system comprising (1) a storage medium on which is stored said at least a first file having a preset access permission, (2) at least a first standard port and a second secure port for connecting said data processing system to external client systems, and (3) means for selectively routing transmission of said at least one file via said first port and said second port, said system comprising:

means, responsive to a request for access to said first file by said external client system, for checking said preset access permission of said first file;

reconfiguration means for configuring said first standard port and said second secured port for supporting a mount by said client system; and

when said preset access permission of said first file indicates secured access is required for said first file, means for dynamically routing a transmission of said first file to external client system via said second port, said means for dynamically routing comprising:

means for first configuring said second secure port to support a remount operation received from said client system;

means for terminating a current mount on said first standard port with said client system; and

means for storing session parameters of said current mount to enable seamless continuation of said session on said second secure port.

11. The system of Claim 10, further comprising:

means for routing said transmission of said first file via said first standard port when said preset access permission indicates a regular access is sufficient.

12. The system of Claim 10, further comprising:

means for enabling a first mount of said data processing system via said first standard port; and

means for enabling a second mount of said data processing system via said second secure port only when said first file requires secured access.

13. The system of Claim 10, wherein said data processing system further comprises an encryption module associated with said second secured port, said logic for dynamically routing comprising:

means for first encrypting said first file utilizing said encryption module.

14. The system of Claim 10, wherein said configuring and storing step includes:

means for retrieving an IP address of said client system;

means for placing said IP address in a configuration of said second secure port,

wherein said second port automatically recognizes a remount operation from said client system and re-establishes the session with said client system.

15. The system of Claim 10, wherein said preset access permission is a bit within metadata linked to said first file and said system further comprises reading a value of said bit to evaluate whether said first file requires secure access.

16. The system of Claim 10, wherein said preset access permission includes an identification of which specific users are permitted to access said first file via a secured access, said system further comprising:

means for comparing a user of said client system with said specific users with permission to access said file; and

when said user is one of said specific users,

means for automatically initiating a re-routing of a transmission of said first file via said second secure port.

17. The system of Claim 10, wherein said first standard port connects to said client system via a first unsecured network and said second secure port connects to said client system via a second secured network.

18. The system of Claim 10, wherein:

said data processing system is a server within a network having a first subnet connecting said first standard port to said client system and a second subnet connecting said second secure port to said client system;

said first file is stored within a filesystem;

said means for checking includes means for accessing said filesystem and locating said first file; and

said means for routing includes means for transmitting said file via said second subnet when said file requires secure access and transmitting said first file via said first subnet when said first file does not require secure access.

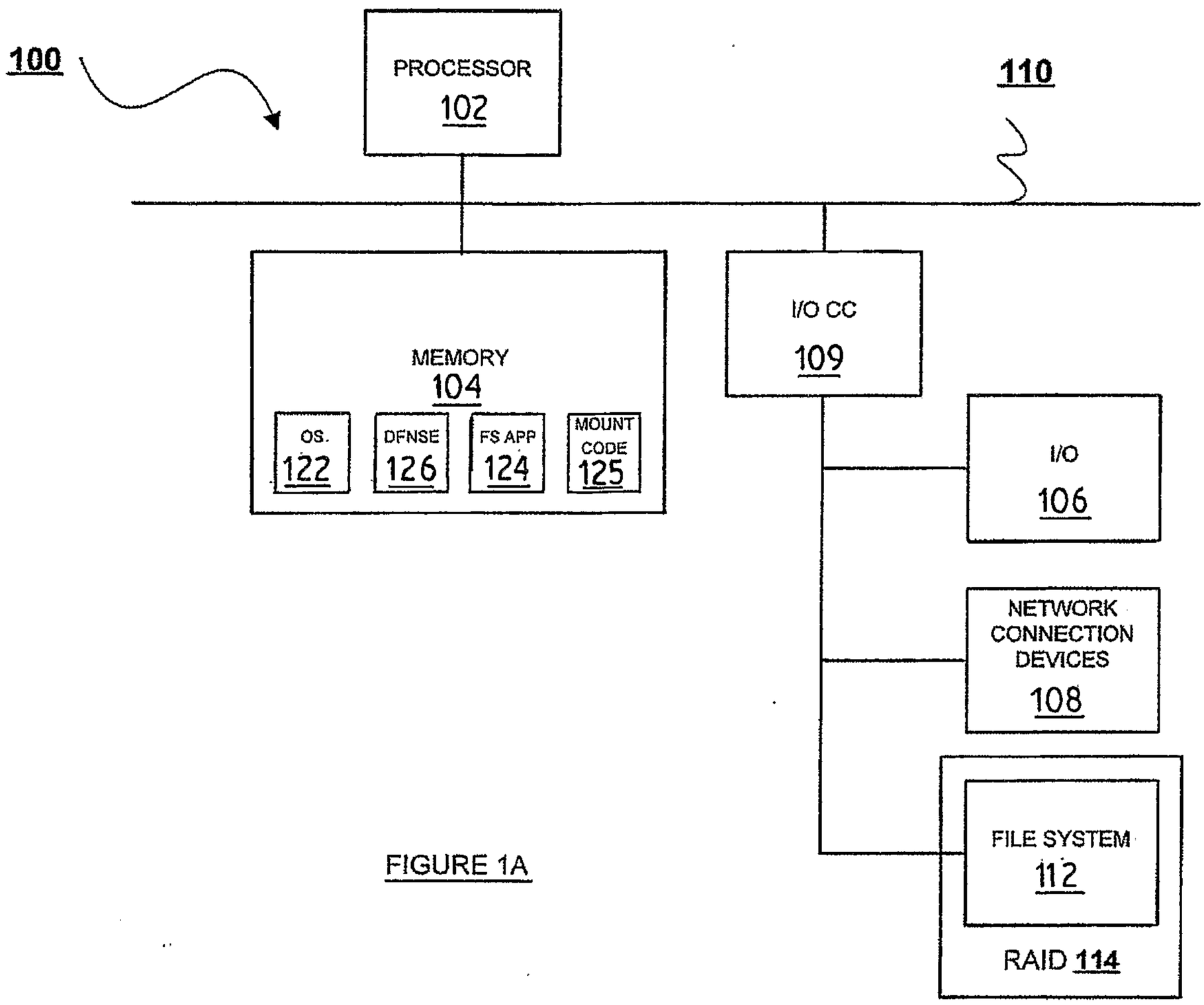



FIGURE 1A

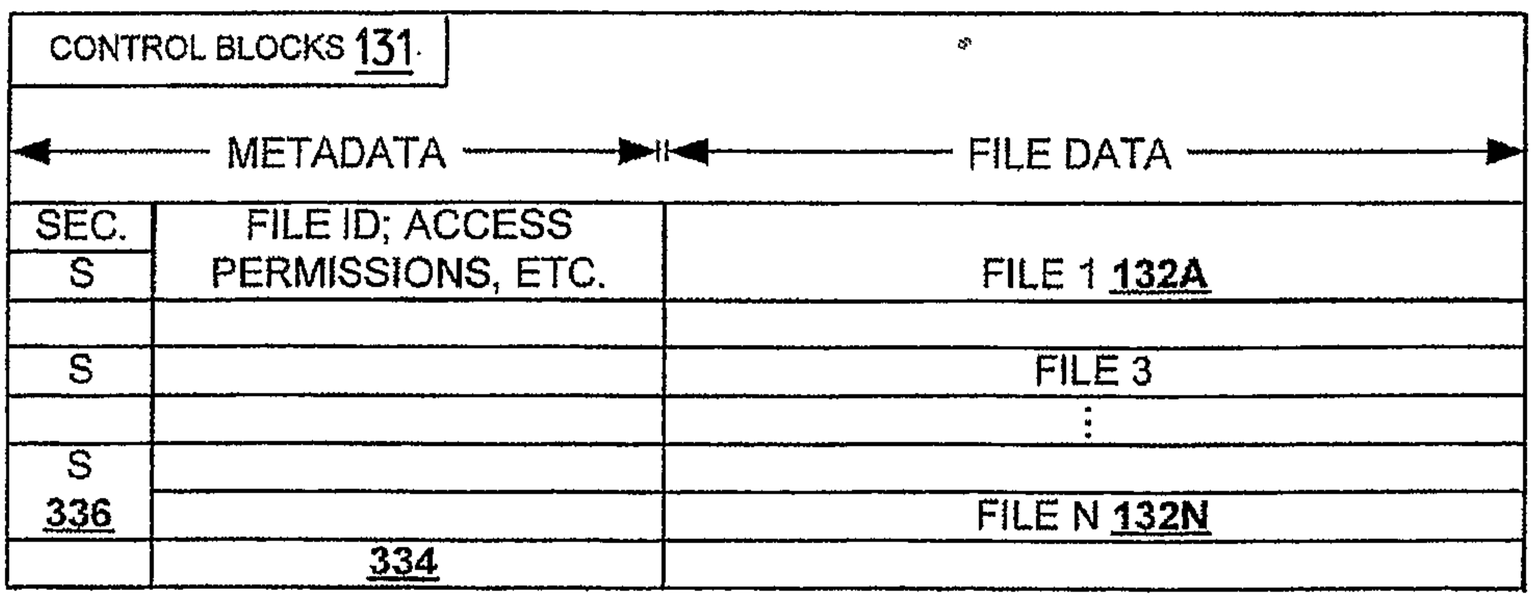



FIGURE 1B

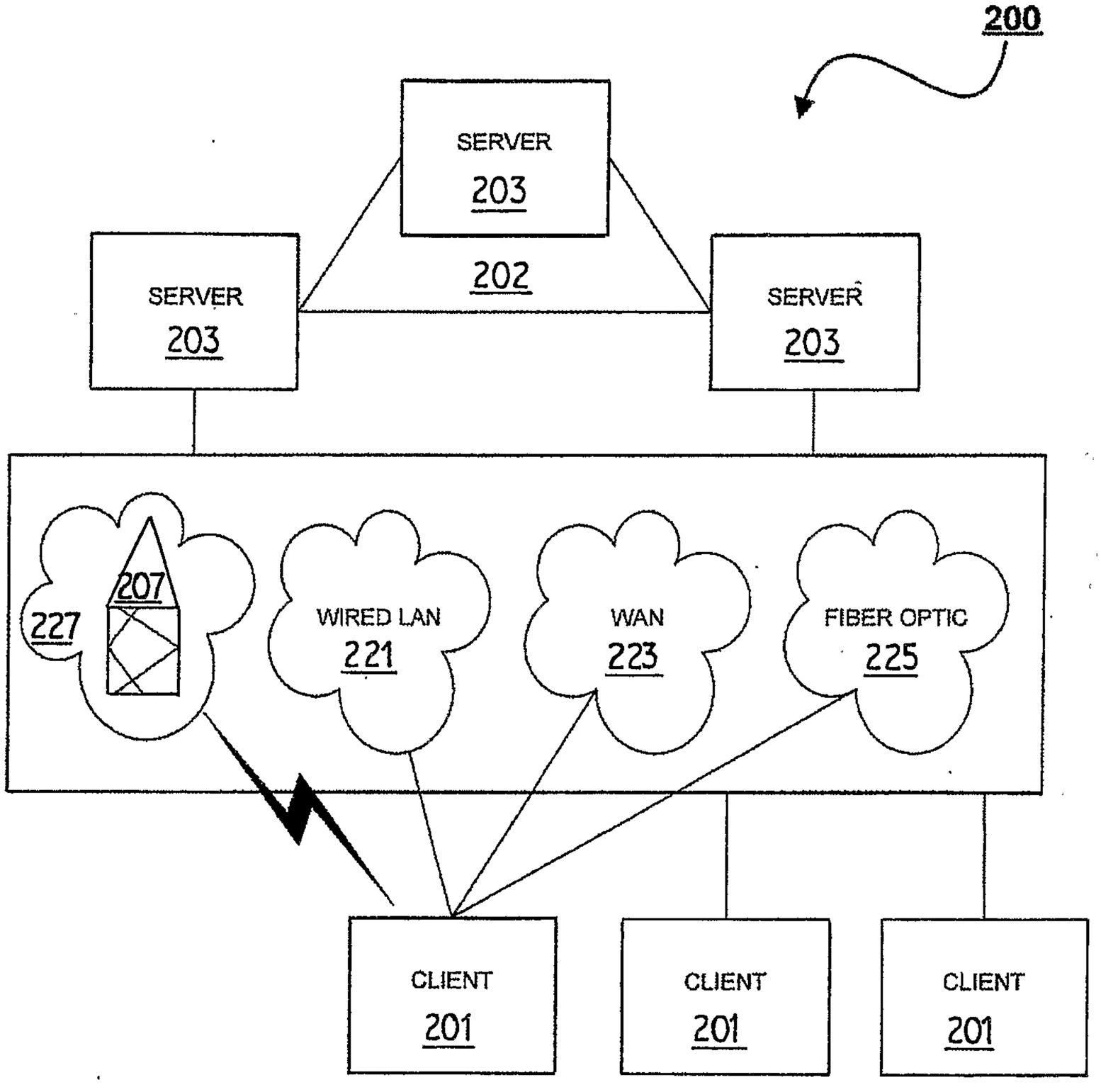

200
SERVER
203
202
SERVER
203
SERVER
203
207
227
WIRED LAN
221
WAN
223
FIBER OPTIC
225
CLIENT
201
CLIENT
201
CLIENT
201

FIGURE 2

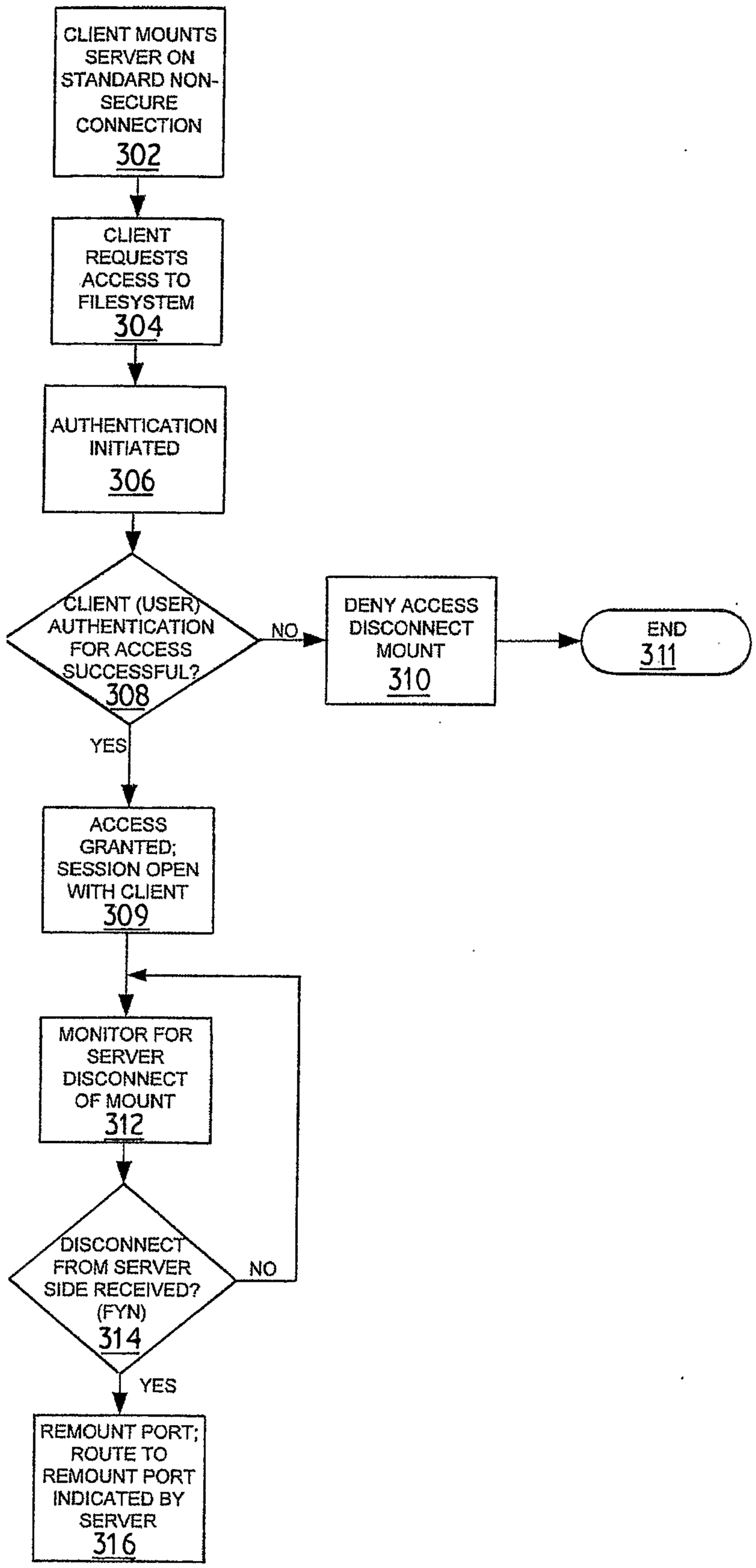

CLIENT MOUNTS SERVER ON STANDARD NON-SECURE CONNECTION
302
CLIENT REQUESTS ACCESS TO FILESYSTEM
304
AUTHENTICATION INITIATED
306
CLIENT (USER) AUTHENTICATION FOR ACCESS SUCCESSFUL?
308
NO
DENY ACCESS DISCONNECT MOUNT
310
END
311
YES
ACCESS GRANTED; SESSION OPEN WITH CLIENT
309
MONITOR FOR SERVER DISCONNECT OF MOUNT
312
DISCONNECT FROM SERVER SIDE RECEIVED? (FYN)
314
NO
YES
REMOUNT PORT; ROUTE TO REMOUNT PORT INDICATED BY SERVER
316

FIGURE 3A

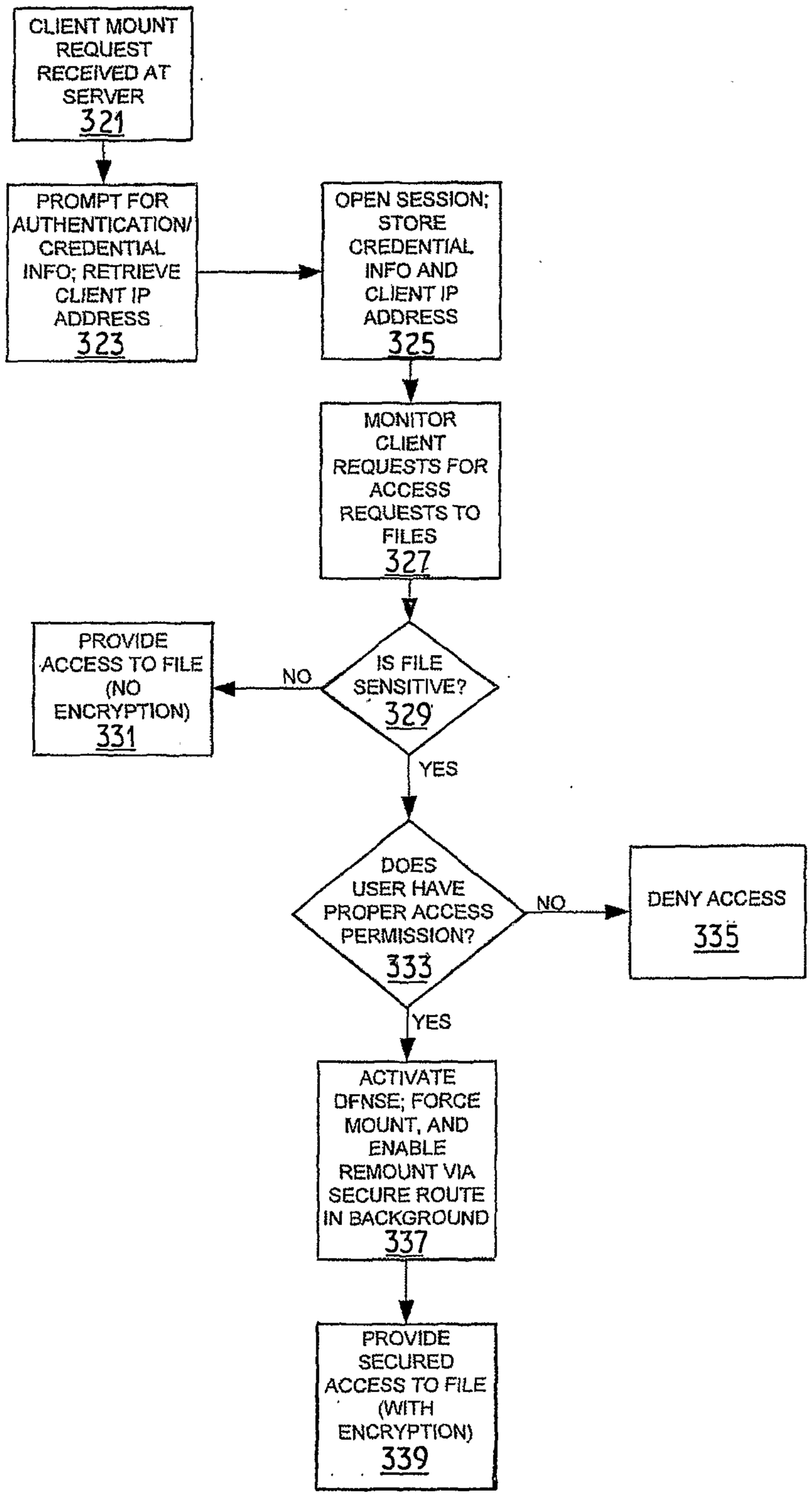

CLIENT MOUNT REQUEST RECEIVED AT SERVER
321
PROMPT FOR AUTHENTICATION/ CREDENTIAL INFO; RETRIEVE CLIENT IP ADDRESS
323
OPEN SESSION; STORE CREDENTIAL INFO AND CLIENT IP ADDRESS
325
MONITOR CLIENT REQUESTS FOR ACCESS REQUESTS TO FILES
327
IS FILE SENSITIVE?
329
NO
PROVIDE ACCESS TO FILE (NO ENCRYPTION)
331
YES
DOES USER HAVE PROPER ACCESS PERMISSION?
333
NO
DENY ACCESS
335
YES
ACTIVATE DFNSE; FORCE MOUNT, AND ENABLE REMOUNT VIA SECURE ROUTE IN BACKGROUND
337
PROVIDE SECURED ACCESS TO FILE (WITH ENCRYPTION)
339

FIGURE 3B

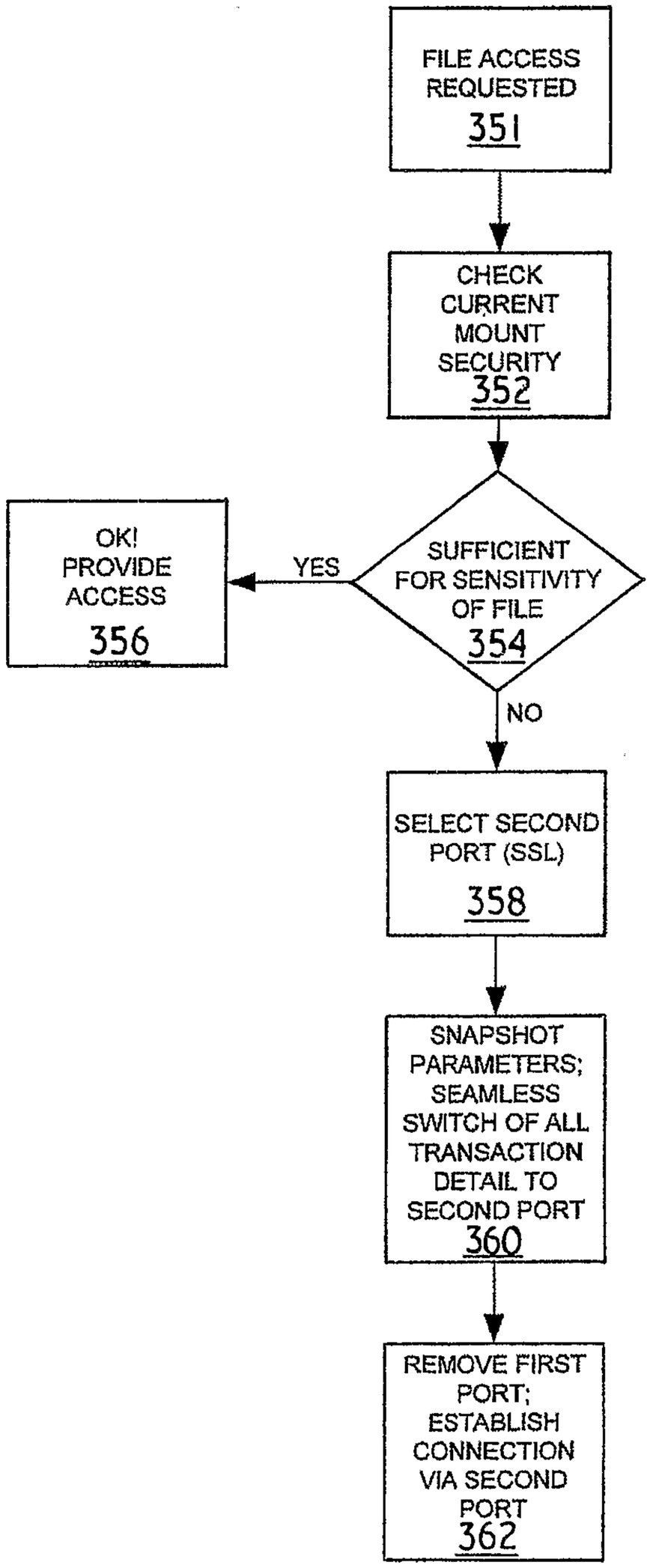

FILE ACCESS REQUESTED
351
CHECK CURRENT MOUNT SECURITY
352
SUFFICIENT FOR SENSITIVITY OF FILE
354
YES
OK! PROVIDE ACCESS
356
NO
SELECT SECOND PORT (SSL)
358
SNAPSHOT PARAMETERS; SEAMLESS SWITCH OF ALL TRANSACTION DETAIL TO SECOND PORT
360
REMOVE FIRST PORT; ESTABLISH CONNECTION VIA SECOND PORT
362

FIGURE 3C

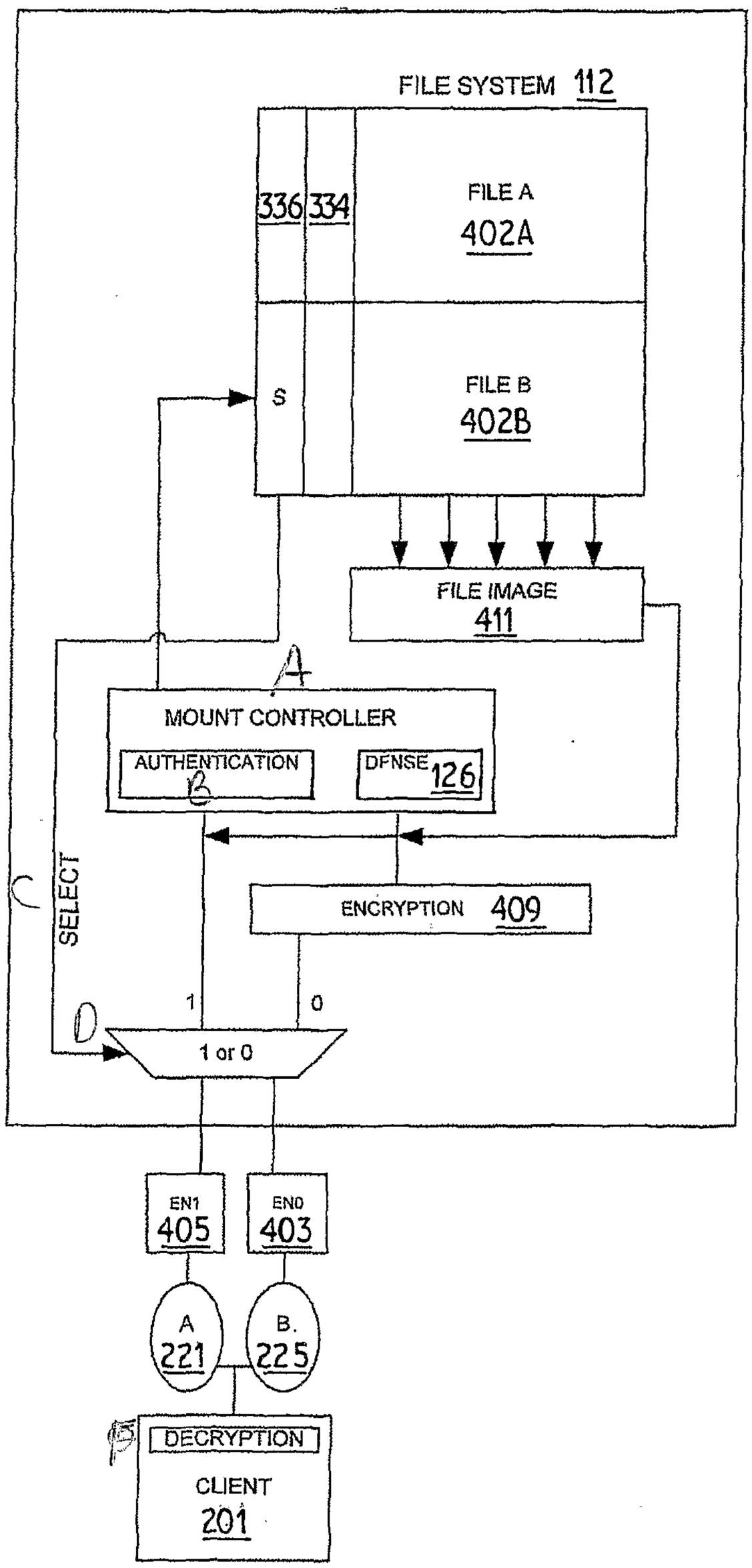

SERVER 203
FILE SYSTEM 112
336
334
FILE A
402A
S
FILE B
402B
FILE IMAGE
411
A
MOUNT CONTROLLER
AUTHENTICATION
B
DFNSE 126
SELECT
C
ENCRYPTION 409
1
0
D
1 or 0
EN1
405
EN0
403
A
221
B.
225
DECRYPTION
CLIENT
201

FIGURE 4

SERVER 203

FILE SYSTEM 112

336 334

FILE A
402A

S

FILE B
402B

FILE IMAGE
411

A

MOUNT CONTROLLER

AUTHENTICATION

B

DFNSE 126

SELECT

C

ENCRYPTION 409

1

0

D

1 or 0

EN1
405

END
403

A
221

B
225

DECRYPTION

E

CLIENT
201