ENABLING PROXY SERVICES USING REFERRAL MECHANISMS

A NAS (Network Attached Storage) switch authenticates a client on multiple file servers for proxy services. The NAS switch enables proxy services by successively authenticating the client on the file servers using referrals. The NAS switch further comprises a connection manager to establish connections to the client and the file servers, a referral manager to redirect the client for successive authentication's, and a transaction manager to perform data transfers with the file servers on behalf of the client. The system components support DFS (Distributed File System), and communicate using a protocol dialect that supports referral mechanisms such as NFSv4 (Network File Server version 4) or CIFS (Common Internet File System). The transaction manager also performs a protocol dialect translation service when the connection manager negotiates one protocol dialect with the client, and a different protocol dialect with the file server.
The text is a document on enabling proxy services using referral mechanisms. It includes a cross-reference to related applications, background information, and a description of related art. The content is about storage networks and file servers, specifically referring to NAS (Network Attached Storage) file servers. The text describes how these servers provide an inexpensive and easily configurable solution for storage networks and how they allow interoperability with clients running different operating systems through open protocols like NFS and CIFS. The document is an excerpt from a patent application and discusses the benefits of using referral mechanisms in proxy services.
By contrast, more expensive and powerful SAN (Storage Area Network) file servers use resources connected by Fibre Channel on a back-end, or a dedicated network. A SAN file system is part of the operating system or an application running on the client. But heterogeneous client operating systems may require additional copies of each file to be stored on the storage network to ensure compatibility on the SAN file server. Additionally, communication between clients and file servers on a SAN use proprietary protocols and thus are typically provided by a common vendor. As a result, NAS file servers are preferred when price and ease of use are major considerations. However, the benefits of NAS storage networks over SAN storage networks also have drawbacks.

One drawback with NAS file servers is that there is no centralized control. Accordingly, each client must maintain communication channels between each of the NFS file servers separately. When NAS file servers are either added or removed from the storage network, each client must mount or unmount directories for the associated storage resources as appropriate. This is particularly inefficient when there are changes in hardware, but not in the particular files available on the network, such as when a failing NAS file server is swapped out for an identically configured back-up NAS file server.

However, when a proxy server is introduced between clients and NAS file servers, a number of issues arise. The proxy must form connections with multiple file servers, which requires authentication with each of them on behalf of the user on the client. This is problematic because the proxy must have a password, or hashing thereof, to respond to server challenges. The proxy could connect to one of the NAS file servers, and pass through an authentication challenge from the file server, and pass back a
response from the client, but this technique only allows the client to connect to that one file server. In order to connect with additional file servers, the proxy would be required to respond to the file server challenges itself. A proxy serving numerous users would thus be required to store all of those passwords, thereby introducing an additional security risk. Furthermore, the proxy server would have to maintain password updates.

[0007] Additionally, the proxy server introduces protocol dialect issues. For each connection with a file server, the proxy server negotiates a single protocol dialect. The proxy server separately negotiates a single protocol dialect with each client, which can be different from that of the file server connection. As a result, these clients would not be able to communicate with the file server.

[0008] Therefore, what is needed is a robust network device to authorize a client with several file servers to provide secure transparency of decentralized file servers such as NAS file servers. Furthermore, the network device should enable communications between a client and a file server using different protocol dialects.
**BRIEF SUMMARY OF THE INVENTION**

[0009] The present invention meets these needs by authenticating a client on multiple file servers for proxy services. In one embodiment, a NAS (Network Attached Storage) switch in the data path of a client and file servers on the storage network, enables proxy services by successively authenticating the client on the file servers using referrals. Advantageously, the client benefits from centralized services on a decentralized network while maintaining security standards.

[0010] An embodiment of a system configured according to the present invention comprises the NAS switch in communication with the client on a front-end of a storage network, and the file servers on a back-end. The NAS switch further comprises a connection manager to establish connections to the client and the file servers, a referral manager to redirect the client for successive authentications, and a transaction manager to perform data transfers with the file servers on behalf of the client. In one embodiment, the system components support DFS (Distributed File System). In another embodiment, the system components communicate using a protocol dialect that supports referral mechanisms such as NFSv4 (Network File Server version 4) or CIFS (Common Internet File System).

[0011] In another embodiment, the referral manager sends an error message to the client in response to a client transaction request. The error message triggers a client supporting DFS to respond with a referral request. The referral manager uses the referral request to redirect the client to one or more file servers which its user has yet to authenticate. To authenticate, the referral manager can pass through a challenge and response algorithm, a password, and other authentication information.
[0012] In one embodiment, the connection manager uses a proxy server to for authentication through short-term connections. This is because some clients will only authenticate once to a single IP address, so a different IP address must be presented to the client for further authentication. Once authentication is complete, the referral mechanism can associate the file server with the virtual server, and refer the client to the virtual server. The virtual server also serves as a single point of communication for both the client and the storage network. In another embodiment, the referral manager uses a cookie having a sharename to uniquely identify the client when redirecting from the proxy server to the virtual server.

[0013] In yet another embodiment, the transaction manager can limit communication protocol dialects to those commonly available between clients and file servers. For example, if a client or a file server supports protocol dialect A, but not B, then the transaction manager limits set-ups to A. The transaction manager can intercept negotiation messages and alter the choices according to those commonly available. In another embodiment, the transaction manager performs a protocol dialect translation service. Translations are necessary, for example, when the connection manager negotiates one protocol dialect with the client, and a different protocol dialect with the file server.
BRIEF DESCRIPTION OF THE DRAWINGS

[0014] FIG. 1 is a high-level block diagram illustrating a storage network system according to one embodiment of the present invention.

[0015] FIG. 2 is a block diagram illustrating an authentication module according to one embodiment of the present invention.

[0016] FIG. 3 is a high-level flow chart illustrating a method of authenticating clients with file servers for proxy services according to one embodiment of the present invention.

[0017] FIG. 4 is a flow chart illustrating a method of establishing a connection with a client according to one embodiment of the present invention.

[0018] FIG. 5 is a flow chart illustrating a method of authenticating a client with a file server using a referral according to one embodiment of the present invention.

[0019] FIG. 6 is an exemplary flow chart illustrating a method of performing transactions with file servers on behalf of clients according to one embodiment of the present invention.

[0020] FIGS. 7A-F is a sequence diagram illustrating a method of authenticating a client with file servers for proxy services according to one embodiment of the present invention.
**Detailed Descriptions of the Invention**

[0021] The present invention provides authentication of a client on multiple file servers for proxy services. A NAS (Network Attached Storage) switch in the data path of a client and NAS file servers on the storage network, successively authenticates a client on file servers using referrals. Some embodiments of a system are described with respect to FIGS. 1-2, and some embodiments of a method operating therein are described with respect to FIGS. 3-7.

[0022] The processes, features, or functions of the present invention can be implemented by program instructions that execute in an appropriate computing device. Example computing devices include enterprise servers, application servers, workstations, personal computers, network computers, network appliances, personal digital assistants, game consoles, televisions, set-top boxes, premises automation equipment, point-of-sale terminals, automobiles, and personal communications devices. The program instructions can be distributed on a computer readable medium, storage volume, or the Internet. Program instructions can be in any appropriate form, such as source code, object code, or scripts.

[0023] FIG. 1 is a high-level block diagram illustrating a storage network system 100 according to one embodiment of the present invention. The system 100 comprises a NAS switch 110 coupled to a client 140 and a storage network 175 through a network 195. The storage network 175 further comprises file servers 130a-c, each also coupled in communication with the NAS switch 110 through the sub-network 196. Note that there can be various configurations of the system 100, such as embodiments including additional clients 140, additional file servers 130, and additional NAS switches 110.
[0024] The system 100 components are implemented in, for example, a personal computer with an x86-type processor executing an operating system and/or an application program, a workstation, a specialized NAS device with an optimized operating system and/or application program, a modified server blade, and the like. In one embodiment, the storage network 175 comprises a NAS storage network, using protocols such as NFSv4 (Network File System version 4) and CIFS (Common Internet File System) that preferably support referral mechanisms. In another embodiment, the storage network 175 comprises a combination of NAS, SAN (Storage Area Network), and other types of storage networks. In yet another embodiment the storage network 175 comprises a decentralized or proprietary storage network other than NAS.

[0025] The NAS switch 110 comprises an authentication module 112 to obtain permission for the NAS switch 110 to perform transactions on behalf of the client 140. In one embodiment, the authentication module 112 does so by successively referring the client 140 to present authentication credentials to different file servers 130; effectively tricking the client 140 to log-in with each file server. The authentication module 112 establishes connections with the client 140 and one of the file servers 130. When the client 140 sends a transaction request to a file server, the authentication module 112 returns an error message indicating the object is not available. The authentication module 112 then uses a subsequent referral request by the client 140 seeking to locate the object to redirect the client 140 to a different file server 130. The client 140 provides authentication information to the other file server in order to again request the object. This process repeats until the NAS switch 110 has logged in to all necessary file servers 130.
In one embodiment, the NAS switch 110 provides proxy services such as continuous transparency to the client 140 with respect to object management. Specifically, the NAS switch 110 can off-load tasks related to physical configurations, object management, object migration, object replication, efficient storage and/or other services on the storage network 175. Preferably, the NAS switch 110 emulates file server processes to the client 140 and emulates client processes to the file servers 130. Accordingly, the client 140 is unaware of the NAS switch 110 since the NAS switch 110 is able to redirect NAS requests intended for the file servers 130 to appropriate locations on the storage network 175. Thus, the client 140 submits object requests, such as file writes and directory reads, directly to the NAS switch 110. Likewise, the file servers 130 are unaware of the NAS switch 110 since the NAS switch 110 is able to resubmit requests, associated with NAS file handles, as if they originated from the client 140. To do so, the NAS switch 110 can use mapping, translating, bridging, packet forwarding, other network interface functionality, and other control processes to perform file handle switching, thereby relieving the client 140 of the need to track changes in a file's physical location.

To provide transparency, the NAS switch 110 can present a unified namespace to the client module 112. The NAS switch 110 creates the unified namespace from exported file system directories sent by the file servers 130. The exported file directories contain NAS file handles indicating object locations on the file server. To create compatibility with the client 140, the NAS switch 110 maps the file system directories to an internal switch file system containing switch file handles which it sends to the client 140. To request an object, the client 140 traverses an exported switch file
system and selects a switch file handle which it sends to the NAS switch 110 along with a requested operation.

[0028] In one embodiment, the NAS switch 110 also tracks reconfigurations resulting from migration, replication and other object relocation processes (e.g. adding or removing file server capacity) with a nested system of tables, or information otherwise linked to the switch file systems. The switch file handles are static as they are persistent through the relocation processes, but the associated NAS file handles can be dynamic as they are selected depending upon an object’s current location. To track various copies of an object, the file server module 114 maintains a directory server table, a file location cache, file handle migration table, and/or a file handle replication table corresponding to each file system that maps NAS file handles of migrated and replicated objects to locations on the storage network 175.

[0029] In general, NAS file handles uniquely identify objects, such as a directory file server, on the file servers 130 as long as that object exists. NAS file handles are file server specific, and are valid only to the file servers 130 that issued the file handles. The process of obtaining a file handle from a file name is called a look-up. The NAS file handle may be formatted according to protocols such as NFS or CIFS. By contrast, a switch file handle identifies a directory or file object independent of location, making it persistent through file replications, migrations, and other data transfers. The switch file can be a NAS file handle modified to refer to the NAS switch file system. An original NAS file handle refers to an initial object location.

[0030] Object transaction requests handled by the NAS switch 110 include, for
example, directory and/or file reads, writes, creation, deletion, moving, and copying. As used herein, various terms are used synonymously to refer to a location of an object prior to replication (e.g., "primary"; "source"; "original"; and "first") and various terms are used to refer to a location of the same object after migration (e.g., "replica"; "destination"; "substitute"; and "second"). Further embodiments of the NAS switch 110 and methods operating therein are described below.

[0031] The client 140 authenticates on the file servers 130 through the NAS switch 110 in order to access objects on the storage network 175. The client 140 can authenticate using a challenge and response algorithm. In one embodiment, the client 140 supports Kerberos authentication for further security against man-in-the-middle attacks. The client 140 can track down moved objects by requesting a referral from the file server 130 whenever the client receives an error message in response to requesting an object that has been moved elsewhere on the storage network 175. In one embodiment, the client 140 supports Distributed File System (DFS) which allows it to view the unified namespace provided by the NAS switch 110. The client 140 refers to objects in the unified namespace with switch file handles submitted to the NAS switch 110 with transaction requests. In one embodiment, the client 140 provides access to a storage network 175 other than a NAS storage network.

[0032] The network 195 facilitates data transfers between connected hosts (e.g., 110, 120, 130, 140). The connections to the network 195 may be wired and/or wireless, packet and/or circuit switched, and use network protocols such as TCP/IP (Transmission Control Protocol/Internet Protocol), IEEE (Institute of Electrical and Electronics Engineers) 802.11, IEEE 802.3 (i.e., Ethernet), ATM (Asynchronous Transfer Mode), or
the like. The network 195 comprises, for example, a LAN (Local Area Network), WAN (Wide Area Network), the Internet, and the like. In one embodiment, the NAS switch 110 acts as a gateway between the client 140, connected to the Internet, and the file servers 130, connected to a LAN.

[0033] FIG. 2 is a block diagram illustrating the authentication module 112 according to one embodiment of the present invention. The authentication module 112 comprises a connection manager 210, a referral manager 220, and a transaction manager 230. Generally, the connection manager 210 sets up communications channels between the client 140 and the file servers 130 for the transaction manager 230 by using referrals from the referral manager 220. Methods operating within the authentication module 112 are discussed below in greater detail with respect to FIG. 3. Note that rather than being strict structural separations, “modules” and “managers” are merely exemplary groupings of functionality corresponding to one or many structures.

[0034] More specifically, the connection manager 210 of the present embodiment comprises a virtual server 212 and proxy servers 214. The virtual server 212 manages long-term communication channels enabled by short-term communications channels of the proxy servers 214. In one embodiment, the proxy servers are uniquely identifiable by the client 140 such as by having separate IP addresses, port numbers, or any other identifiers. Accordingly, each proxy server 214 can correspond to a file server 130 so that the client 140 interacts with the proxy server 214 in the same manner as it would directly interact with the file server 130. Once a connection is established the proxy server 214 associates the connection with the virtual server 212 so that the virtual server 212 becomes a single point of communications. In one embodiment, the proxy server
214 communication channels are torn down shortly thereafter. In another embodiment, the connection manager comprises physical communication accessories such as an Ethernet input/output port, communications software for managing logical ports, and the like.

[0035] The referral manager 220 redirects the client 140 from one proxy server 214 to another for authentication. When the referral manager 220 detects a client without authentication on one or more file servers 130, it can generate error messages in response to a transaction request. Furthermore, in response to referral requests by the client 140, the referral manager 220 can redirect the client 140 to the one or more file servers 130.

[0036] The transaction manager 230 performs transactions with the file servers 130 on behalf of the client 140. To do so, the transaction manager 230 performs file switching between client file handles to the unified namespace and NAS file handles used by the file servers 130. In one embodiment the transaction manager 230 translates between protocol dialects when the client connections and server connections have negotiated different protocol dialects. In another embodiment, the transaction manager 230 provides services that are not initiated by the client 140 such as file migration, file replication, and directory services.

[0037] FIG. 3 is a high-level flow chart illustrating a method 300 of authenticating clients 140 with file servers 130 for proxy services according to one embodiment of the present invention. Note that the embodiments of FIGS. 3-7 are merely exemplary and can vary between protocol different protocols. An example of the method 300 is set forth in a sequence diagram of FIG. 7.
[0038] In one embodiment, the connection manager 210 first establishes 310 a connection with a client 140 as discussed in greater detail with respect to FIG. 4. Either the client 140 or the NAS switch 110 can initiate the connection. The referral module 220 authenticates 320 the client with a file server 130 using a referral as discussed in greater detail with respect to FIG. 5. Note that the three file servers 130 of the storage network 175 are merely exemplary as there may be only one or many more than three file servers 130. The referral module 220 continues authenticating 330 until the client 140 has logged in to all file servers 130. In one embodiment, the authentication module 112 authenticates multiple clients 140 and/or users 340.

[0039] Once the communication channels have been properly established, the transaction module 230 performs 350 transactions with file servers 130 on behalf of clients 140 as discussed in greater detail with respect to FIG. 6. Additional embodiments of performing 350 transactions are described in US Patent Application No. 10/831,376 and U.S. Patent Application No. 10/831,701.

[0040] FIG. 4 is a flow chart illustrating the method 310 of establishing a connection with the client 140 according to one embodiment of the present invention. A first proxy server 214 receives 410 a connection request from the client 140. To enable communications, the first proxy server 214 negotiates 420 a protocol dialect with the client 140. In addition, the first proxy server 214 sends 430 a connection request to the first file server 130. The first proxy server 214 also negotiates 440 a protocol dialect with the first file server 130. In one embodiment, the protocol dialects differ. For example, the protocol dialects can differ due to lack of capability by either the client 140 or the first file server 130. In another example, the first file server 130 is already be connected
to the NAS switch 110 using a different protocol dialect.

[0041] The proxy server 214 authenticates 450 the user on the first file server 130. In one embodiment, the file server 130 presents a challenge to the proxy server 214, which the proxy server 214 forwards to the client 140. The client 140 a response, such as a hashed bit string, back to the file server 130 through the proxy server 214. In another embodiment, the client 140 is required to send a password or other authentication information. As a result, the proxy server 214 successfully establishes a direct communication channel with the first file server 130 without storing the user’s authentication information.

[0042] Since proxy server 214 connections may only be temporary, the proxy server 214 associates the first file server 214 with a long-term connection to the virtual server 212. In one embodiment, the proxy server 214 tears down the client connection after this transfer.

[0043] FIG. 5 is a flow chart illustrating the method 320 of authenticating the client 140 with a next file server 130 using a referral according to one embodiment of the present invention. The first proxy server 214 receives 510 a client 140 transaction request. For example, the client 140 can send an OPEN ("\some\file.txt") message. In response, the referral manager 220 sends 520 an error message to the client 140 such as a PATH_NOT_COVERED message. The referral manager 220 receives 530 a referral request from the client 140 such as a GET_DFS_REFERRAL ("\some\file.txt") message. In response, the referral manager 220 sends 540 a referral for a next file server 130 or a list of file servers 130. The referral can be a file name from the unified namespace,
specific to a file server 130, include an IP address, or any other identifying information.
In one embodiment, the referral manager 220 sends a cookie having a sharename to
uniquely identify the user and/or client 140.

[0044] The next proxy server 214 receives 550 a transaction request from the
client 140 based on the referral. The next proxy server 214 and the client 140 negotiate a
protocol dialect since the client 140 treats connection as separate from the virtual server
212 connection. Additionally, the user needs to authenticate on the next file server 130,
so the next proxy server 214 sends 560 a connection request. If not already connected,
the next proxy server 214 negotiates 570 a protocol dialect with the next file server 570.
In one embodiment, the protocol dialect differs from either the client 140 protocol
dialect, or the previous file server 130 protocol dialect. The next proxy server 214
authenticates 580 the user on the next file server 130. Again, the next proxy server 214
associates the next file server 214 with the virtual server 212 for a long-term connection.

[0045] FIG. 6 is a flow chart illustrating the method 350 of performing
transactions with file servers 130 on behalf of clients 140 according to one embodiment
of the present invention. The transaction manager 230 receives 610 a transaction request
from the client 140 using, for example, a switch file handle from the unified namespace.
The transaction manager 230 determines 620 which file server 130 is associated with the
transaction request. To do so, the transaction manager 230 matches the switch file handle
to a NAS file handle exported by the file servers 130. In one embodiment, the transaction
manager 230 keeps abreast of changes on the storage network 175 such as file migrations
and file replications, and makes corresponding updates to NAS file handles. For
example, for a migrated file, the transaction manager 230 will use a NAS file handle
pointing to the file’s new location.

In the present embodiment, if the client 140 and file server 130 communicate with the virtual server 212 using different protocol dialects 630, the transaction manager 230 can perform 640 a translation service. Once the transaction request has been processed, the virtual server 212 sends 650 the transaction request to the file server 130. For data transfers such as a read operation, the transaction manager 230 performs 660 a data transfer by forwarding data received from the file server 130 to the requesting client 140.

FIGS. 7A-F are exemplary sequence diagrams illustrating a method 700 of authenticating the client 140 with file servers 130 for proxy services according to one embodiment of the present invention. In FIG. 7A, the client 140 connects to the proxy server 214a, which initiates a connection with the file server 130a. The proxy server 214a can negotiate separate protocol dialects for the client 140 connection and the file server 130a connection. The proxy server 214a can also pass a challenge and response algorithm, log-in credentials, or other authentication information. In FIG. 7B, once authentication has been successfully completed, the connection manager 210 refers the client to the virtual server 212, and associates the file server 130a with the virtual server 212. The proxy server 214a can tear down the client 140 connection.

In FIG. 7C, the connection manager 210 next refers the client to the proxy server 214b in order to authenticate with file server 130b. In FIG. 7D, the connection manager 210 associates the authenticated file server 130b with the virtual server 212 and tear down the client 140 connection. This process is repeated in FIGS. 7E-F in
authenticating the client 140 with the file server 130c.

[0049] The above description is included to illustrate the operation of the preferred embodiments and is not meant to limit the scope of the invention. The scope of the invention is instead be limited only by the following claims.
WE CLAIM:

1. A method for authenticating a client supporting referrals with a plurality of file servers to provide proxy services for the client, comprising the steps of:
   establishing a connection with the client;
   authenticating the client with the plurality of file servers by successively
   redirecting the client to provide authentication information to each of the
   plurality of file servers with a plurality of referrals; and
   performing transactions with the plurality of file servers on behalf of the client.

2. The method of claim 1, wherein establishing the connection with the client comprises:
   negotiating a first protocol dialect with the client.

3. The method of claim 2, further comprising:
   negotiating a second protocol dialect with at least one file server from the
   plurality of file servers, the second protocol dialect differing from the first
   protocol dialect,
   wherein the performing transactions comprises translating a transaction request
   from the first protocol dialect to the second protocol dialect.

4. The method of claim 2, further comprising:
   negotiating a third protocol dialect with at least one file server from the plurality
   of file servers, the third protocol dialect differing from the second protocol
   dialect.

5. The method of claim 2, wherein the first protocol dialect comprises one
   from the group consisting of: NFS (Network File System) and CIFS (Common Internet
   File System)

6. The method of claim 1, wherein the authenticating the client with the
   plurality of file servers comprises:
   receiving a transaction request from the client; and
sending an error message to the client responsive to the transaction request.

7. The method of claim 1, wherein the authenticating the client with the plurality of file servers comprises:
   receiving a referral request from the client; and
   referring the client to a file server from the plurality of file servers.

8. The method of claim 1, wherein the authenticating the client with the plurality of file servers comprises:
   receiving authentication information for at least one file server; and
   establishing a connection to the at least one file server using the authentication information.

9. The method of claim 8, wherein the authenticating information comprises Kerberos authentication information.

10. The method of claim 1, wherein the authenticating the client with the plurality of file servers comprises:
    negotiating a third protocol dialect with a second file server.

11. The method of claim 1, wherein the authenticating the client with the plurality of file servers comprises:
    associating the plurality of file servers with the virtual server; and
    referring the client with to virtual server using a sharename that uniquely identifies the client.

12. The method of claim 1, wherein the authenticating the client with the plurality of file servers comprises:
    authenticating the client with the plurality of file servers using a plurality of proxy servers, each at a separate network address.

13. The method of claim 12, further comprising:
    associating the plurality of file servers with a virtual server at a network address differing from the proxy server network addresses.
14. The method of claim 1, wherein the client supports DFS (Distributed File System).

15. The method of claim 1, wherein the performing the transactions comprises:
   performing transactions from the group consisting of: create file, delete file, move file, copy file, read file, and write file.

16. A method for authenticating a NAS client supporting referrals with a plurality of NAS file servers to provide proxy services for the NAS client, comprising the steps of:
   establishing a connection with the NAS client that supports DFS (Distributed File System);
   receiving a transaction request from the NAS client; and
   sending an error message to the NAS client responsive to the transaction request
   receiving a referral request from the NAS client; and
   referring the NAS client to a NAS file server from the plurality of file servers
   receiving authentication information for the NAS file server; and
   establishing a connection to the NAS file server using the authentication information; and
   performing transactions with the plurality of NAS file servers on behalf of the NAS client.

17. A NAS (Network Attached Storage) switch to authenticate a client supporting referrals with a plurality of file servers to provide proxy services for the client, comprising:
   a connection manager to establish a connection with the client;
   a referral manager to authenticate the client with the plurality of file servers by successively redirecting the client to provide authentication information to each of the plurality of file servers with a plurality of referrals; and
   a transaction manager to perform transactions with the plurality of file servers on behalf of the client.
18. The NAS switch of claim 17, wherein the connection manager negotiates a first protocol dialect with the client.

19. The NAS switch of claim 18, wherein the connection manager negotiates a second protocol dialect with at least one file server from the plurality of file servers, the second protocol dialect differing from the first protocol dialect, and the transaction manager translates a transaction request from the first protocol dialect to the second protocol dialect.

20. The NAS switch of claim 18, wherein the connection manager negotiates a third protocol dialect with at least one file server from the plurality of file servers, the third protocol dialect differing from the second protocol dialect.

21. The NAS switch of claim 18, wherein the first protocol dialect comprises one from the group consisting of: NFS (Network File System) and CIFS (Common Internet File System).

22. The NAS switch of claim 17, wherein the referral manager receives a transaction request from the client and, in response, sends an error message to the client.

23. The NAS switch of claim 17, wherein the referral manager receives a referral request from the client and, in response refers the client to a file server from the plurality of file servers.

24. The NAS switch of claim 17, wherein the referral manager receives authentication information for at least one file server, and establishes a connection to the at least one file server using the authentication information.

25. The NAS switch of claim 24, wherein the authenticating information comprises Kerberos authentication information.

26. The NAS switch of claim 17, wherein the referral manager negotiates a third protocol dialect with a second file server.
27. The NAS switch of claim 17, wherein the referral manager associates the plurality of file servers with the virtual server, and refers the client with the virtual server using a sharename that uniquely identifies the client.

28. The NAS switch of claim 17, wherein the authenticating the client with the plurality of file servers comprises:
   authenticating the client with the plurality of file servers using a plurality of proxy servers, each at a separate network address.

29. The NAS switch of claim 28, further comprising:
   associating the plurality of file servers with a virtual server at a network address differing from the proxy server network addresses.

30. The NAS switch of claim 17, wherein the client supports DFS (Distributed File System).

31. The NAS switch of claim 17, wherein the performing the transactions comprises:
   performing transactions from the group consisting of: create file, delete file, move file, copy file, read file, and write file.

32. A computer program product, comprising a computer-readable medium having computer program instructions for a method for authenticating a client supporting referrals with a plurality of file servers to provide proxy services for the client, the method comprising the steps of:
   establishing a connection with the client;
   authenticating the client with the plurality of file servers by successively redirecting the client to provide authentication information to each of the plurality of file servers with a plurality of referrals; and performing transactions with the plurality of file servers on behalf of the client.

33. The computer program product of claim 32, wherein establishing the connection with the client comprises:
negotiating a first protocol dialect with the client.

34. The computer program product of claim 33, further comprising:
  negotiating a second protocol dialect with at least one file server from the
  plurality of file servers, the second protocol dialect differing from the first
  protocol dialect,
  wherein the performing transactions comprises translating a transaction request
  from the first protocol dialect to the second protocol dialect.

35. The computer program product of claim 33, further comprising:
  negotiating a third protocol dialect with at least one file server from the plurality
  of file servers, the third protocol dialect differing from the second protocol
  dialect.

36. The computer program product of claim 33, wherein the first protocol
  dialect comprises one from the group consisting of: NFS (Network File System) and
  CIFS (Common Internet File System)

37. The computer program product of claim 32, wherein the authenticating the
  client with the plurality of file servers comprises:
  receiving a transaction request from the client; and
  sending an error message to the client responsive to the transaction request.

38. The computer program product of claim 32, wherein the authenticating the
  client with the plurality of file servers comprises:
  receiving a referral request from the client; and
  referring the client to a file server from the plurality of file servers.

39. The computer program product of claim 32, wherein the authenticating the
  client with the plurality of file servers comprises:
  receiving authentication information for at least one file server; and
  establishing a connection to the at least one file server using the authentication
  information.
40. The computer program product of claim 39, wherein the authenticating information comprises Kerberos authentication information.

41. The computer program product of claim 32, wherein the authenticating the client with the plurality of file servers comprises:
   negotiating a third protocol dialect with a second file server.

42. The computer program product of claim 32, wherein the authenticating the client with the plurality of file servers comprises:
   associating the plurality of file servers with the virtual server; and
   referring the client with to virtual server using a sharename that uniquely identifies the client.

43. The computer program product of claim 32, wherein the authenticating the client with the plurality of file servers comprises:
   authenticating the client with the plurality of file servers using a plurality of proxy servers, each at a separate network address.

44. The computer program product of claim 43, further comprising:
   associating the plurality of file servers with a virtual server at a network address differing from the proxy server network addresses.

45. The computer program product of claim 32, wherein the client supports DFS (Distributed File System).

46. The computer program product of claim 32, wherein the performing the transactions comprises:
   performing transactions from the group consisting of: create file, delete file, move file, copy file, read file, and write file.

47. A NAS (Network Attached Storage) switch to authenticate a client supporting referrals with a plurality of file servers to provide proxy services for the client, comprising:
   a means for establishing a connection with the client;
a means for authenticating the client with the plurality of file servers by
successively redirecting the client to provide authentication information to
each of the plurality of file servers with a plurality of referrals; and
a means for performing transactions with the plurality of file servers on behalf of
the client.
FIG. 2

AUTHENTICATION MODULE
112

Connection Manager
210

Virtual Server
212
Proxy Servers
214

Referral Manager
220

Transaction Manager
230
START

Establish Connection With Client

Authenticate Client With A Next File Server Using A Referral

More File Servers?

More Clients?

Perform Transactions With File Servers On Behalf Of Clients

END

FIG. 3
ESTABLISH CONNECTION WITH CLIENT 310

Receive Client Connection Request At First Proxy Server 410

Negotiate Protocol Dialect With Client 420

Send Proxy Request To First File Server 430

Negotiate Protocol Dialect With First File Server 440

Authenticate User On First File Server 450

Associate First File Server With Virtual Server 460

RETURN 495

FIG. 4
AUTHENTICATE CLIENT WITH A NEXT FILE
SERVER USING A REFERRAL
320

Receive Client Transaction Request
510

Send Error Message To Client
520

Receive Client Referral Request
530

Send Referral For Next File Server With Share Name
540

Receive Client Transaction Request At Next File Server
550

Send Proxy Connection Request To Next File Server
560

Negotiate Protocol Dialect With Next File Server
570

Authenticate User On Next File Server
580

Associate Next File Server With Virtual Server
590

RETURN
595

FIG. 5
PERFORM TRANSACTIONS WITH FILE SERVERS ON BEHALF OF CLIENTS

Receive Client Transaction Request

Determine File Server Associated With Transaction Request

Same Protocol Dialect?

Perform Translation Between Protocol Dialects

Send Client Transaction Request To Associated File Server

Perform Data Transfer

RETURN