METHODS AND APPARATUS FOR REMOTE AUTHENTICATION IN A SERVER-BASED COMPUTING SYSTEM

A method and apparatus for remotely authenticating a user to a server in a server-based computing environment. A client computing device receives user credentials and generates user authentication data based upon the received credentials. The client computing device transmits the generated user authentication data to a server computing device. The server computing device authenticates the user responsive to the transmitted user authentication data.
METHODS AND APPARATUS FOR REMOTE AUTHENTICATION IN A SERVER-BASED COMPUTING SYSTEM

Field of the Invention

The present invention relates to methods and apparatus for providing distributed application processing in a server-based computing system and, in particular, to remote authentication techniques in such a system.

Background of the Invention

Technologies for providing remote access to networked resources include a variety of client/server software combinations. One of these technologies is often referred to as a "thin-client" or "server-based computing" system. In these systems, an application program is executed by a server computing device on behalf of one or more client computing devices. Only client input to the application and application output are transmitted between a client computing device and a server computing device. These systems generally require users of client computing devices to authenticate themselves before applications may be executed on their behalf by the server computing devices.

The client computing device may require the user to log on locally before using the device. Logging on locally to the client computing device usually requires a username and password, but many client operating systems allow the logon mechanism to be replaced, for example, to require the user to log on with a token-based scheme, a smartcard or a biometric such as a fingerprint.

Despite authenticating to the client computing device, the user will often also need to authenticate to the server computing device. However, if a replacement logon mechanism such as those described above is used, the user may not be able to authenticate to the server computing device because the server computing device will often accept only a username and password; if the user has authenticated using a technique such as a biometric or smart card, the user may not know a valid username-password combination useful to authenticate to the server computing device. Some technologies allow
interception by the client computing device of a user-supplied username-password combination. However, these technologies will not work if the standard authentication mechanism is replaced.

Accordingly, improved methods of remotely authenticating users of a thin-client system are desired.

Summary of the Invention

In one aspect the present invention uses the industry standard Generic Security Services Application Program Interface (GSSAPI) in conjunction with a thin-client protocol such as the ICA protocol, manufactured by Citrix Systems, Inc. of Ft. Lauderdale, Florida, or the RDP protocol, manufactured by Microsoft Corporation of Redmond, Washington, to remotely authenticate users of client computing devices.

In another aspect, the invention enhances security by providing an alternative to the network provider method of pass-through authentication. Instead of the client computing device sending user authentication credentials, e.g., a password, to the server computing device, an authentication virtual channel driver sends user authentication data over a virtual channel within a thin-client protocol. User authentication data can be used to verify the identity of a user but does not reveal the user's underlying authentication credentials. The transmitted user authentication data is used to authenticate the user to the server computing device. The client, therefore, never accesses the user authentication credentials; it is not required to install a network provider which requires administrator privileges and makes the authentication credentials available to any local processes on the client computing device; the user's authentication credentials are not sent over the network in any form; and remote authentication is performed by the server computing device's underlying operating system.

In one aspect of the present invention, a client computing device receives user credentials and generates user authentication data based on the received credentials. The client computing device transmits the generated user authentication data to a server computing device. The server computing device
authenticates the user responsive to the user authentication data. The server generates new user authentication data based on the received user authentication data. The server transmits the new user authentication data to a second server. The second server authenticates the user, responsive to the received user authentication data. In one embodiment, the user accesses available resources over a connection between the first server and the second server.

In another aspect the present invention uses a virtual channel within the ICA protocol or the RDP protocol to exchange authentication data for remote authentication of users. A virtual channel is any logical association between two or more endpoints for the purpose of transmitting data between the endpoints.

In another aspect the present invention uses the GSSAPI for authentication, and therefore works with any authentication method supported by a GSSAPI implementation, such as the Kerberos authentication method, and can be used on any client computing platform or device that supports GSSAPI.

In another aspect of the invention, user authentication credentials (e.g. a password) are not explicitly intercepted or handled by either the client or the server and user authentication credentials are not transmitted between a client computing device and a server computing device.

In another aspect of the invention the client computing device authenticates the server computing device and the server computing device authenticates the user of the client computing device.

In another aspect of the invention the delegation security policy in Microsoft Windows is upheld. For example, if the user account setting “Account is sensitive and must not be delegated” is enabled, that user will not be able to use this remote authentication technique to logon to another server.

**Brief Description of the Drawings**

These and other aspects of this invention will be readily apparent from the detailed description below and the appended drawings, which are meant to illustrate and not to limit the invention, and in which:
FIG. 1 is a block diagram of an environment suitable for practicing the illustrative embodiment of the present invention;

FIG. 2A and 2B are block diagrams depicting embodiments of computers useful in connection with the present invention;

FIG. 3A is a block diagram depicting an embodiment of the network 40 in which the invention may be performed;

Fig. 3B is a block diagram depicting an embodiment of the process by which a client node initiates execution of an available application and a server presents the results of the application to the client node;

Fig. 3C is a block diagram depicting an embodiment of the process by which a client node initiates execution of an available application via the World Wide Web;

Fig. 3D is a block diagram depicting an embodiment of the process of communication among a client node and two server nodes; and

FIG. 4 is a block diagram of an embodiment of a system for remotely authenticating a user of a client node to a server computing device.

Detailed Description of the Invention

The illustrative embodiment of the present invention is applicable to a distributed networking environment where a remote user requests access to content. Prior to discussing the specifics of the present invention, it may be helpful to discuss some of the network environments in which the illustrative embodiment of the present invention may be employed.

Referring now to FIG. 1, in brief overview, one embodiment of a client-server system in which the present invention may be used is depicted. A first computing device (client computing device) 100 communicates with a second computing device (server computing device) 140 over a communications network 180. In some embodiments the second computing device is also a client device 100. The topology of the network 180 over which the client devices 100 communicate with the server device 140 may be a bus, star, or ring
topology. The network 180 can be a local area network (LAN), a metropolitan area network (MAN), or a wide area network (WAN) such as the Internet.

The client and server devices 100, 140 can connect to the network 180 through a variety of connections including standard telephone lines, LAN or WAN links (e.g., T1, T3, 56 kb, X.25), broadband connections (ISDN, Frame Relay, ATM), and wireless connections. Connections can be established using a variety of communication protocols (e.g., TCP/IP, IPX, SPX, NetBIOS, NetBEUI, SMB, Ethernet, ARCNET, Fiber Distributed Data Interface (FDDI), RS232, IEEE 802.11, IEEE 802.11a, IEEE 802.11b, IEEE 802.11g and direct asynchronous connections). Other client devices and server devices (not shown) may also be connected to the network 180.

The client device 100 can be any device capable of receiving and displaying output from applications executed on its behalf by one or more server computing devices 140 and capable of operating in accordance with a protocol as disclosed herein. The client computing device 100 may be a personal computer, windows-based terminal, network computer, information appliance, X-device, workstation, mini computer, personal digital assistant or cell phone.

Similarly, the server computing device 140 can be any computing device capable of: receiving from a client computing device 100 user input for an executing application, executing an application program on behalf of a client computing device 100, and interacting with the client computing device using a protocol as disclosed herein. The server computing device 140 can be provided as a group of server devices logically acting as a single server system, referred to herein as a server farm. In one embodiment, the server computing device 140 is a multi-user server system supporting multiple concurrently active client connections.

FIGs. 2A and 2B depict block diagrams of a typical computer 200 useful as client computing devices 100 and server computing devices 140. As shown in FIGs. 2A and 2B, each computer 200 includes a central processing unit 202, and a main memory unit 204. Each computer 200 may also include other optional elements, such as one or more input/output devices 230a-230-b.
(generally referred to using reference numeral 230), and a cache memory 240 in communication with the central processing unit 202.

The central processing unit 202 is any logic circuitry that responds to and processes instructions fetched from the main memory unit 204. In many embodiments, the central processing unit is provided by a microprocessor unit, such as: the 8088, the 80286, the 80386, the 80486, the Pentium, Pentium Pro, the Pentium II, the Celeron, or the Xeon processor, all of which are manufactured by Intel Corporation of Mountain View, California; the 68000, the 68010, the 68020, the 68030, the 68040, the PowerPC 601, the PowerPC604, the PowerPC604e, the MPC603e, the MPC603ei, the MPC603ev, the MPC603r, the MPC603p, the MPC740, the MPC745, the MPC750, the MPC755, the MPC7400, the MPC7410, the MPC7441, the MPC7445, the MPC7447, the MPC7450, the MPC7451, the MPC7455, the MPC7457 processor, all of which are manufactured by Motorola Corporation of Schaumburg, Illinois; the Crusoe TM5800, the Crusoe TM5600, the Crusoe TM5500, the Crusoe TM5400, the Efficeon TM8600, the Efficeon TM8300, or the Efficeon TM8620 processor, manufactured by Transmeta Corporation of Santa Clara, California; the RS/6000 processor, the RS64, the RS 64 II, the P2SC, the POWER3, the RS64 III, the POWER3-II, the RS 64 IV, the POWER4, the POWER4+, the POWER5, or the POWER6 processor, all of which are manufactured by International Business Machines of White Plains, New York; or the AMD Opteron, the AMD Athlon 64 FX, the AMD Athlon, or the AMD Duron processor, manufactured by Advanced Micro Devices of Sunnyvale, California.

Main memory unit 204 may be one or more memory chips capable of storing data and allowing any storage location to be directly accessed by the microprocessor 202, such as Static random access memory (SRAM), Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM),
SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), or Ferroelectric RAM (FRAM). In the embodiment shown in FIG. 2A, the processor 202 communicates with main memory 204 via a system bus 220 (described in more detail below). FIG. 2B depicts an embodiment of a computer system 200 in which the processor communicates directly with main memory 204 via a memory port. For example, in FIG. 2B the main memory 204 may be DRDRAM.

FIGs. 2A and 2B depict embodiments in which the main processor 202 communicates directly with cache memory 240 via a secondary bus, sometimes referred to as a "backside" bus. In other embodiments, the main processor 202 communicates with cache memory 240 using the system bus 220. Cache memory 240 typically has a faster response time than main memory 204 and is typically provided by SRAM, BSRAM, or EDRAM.

In the embodiment shown in FIG. 2A, the processor 202 communicates with various I/O devices 230 via a local system bus 220. Various busses may be used to connect the central processing unit 202 to the I/O devices 230, including a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, or a NuBus. For embodiments in which the I/O device is an video display, the processor 202 may use an Advanced Graphics Port (AGP) to communicate with the display. FIG. 2B depicts an embodiment of a computer system 200 in which the main processor 202 communicates directly with I/O device 230b via HyperTransport, Rapid I/O, or InfiniBand. FIG. 2B also depicts an embodiment in which local busses and direct communication are mixed: the processor 202 communicates with I/O device 230a using a local interconnect bus while communicating with I/O device 230b directly.

A wide variety of I/O devices 230 may be present in the computer system 200. Input devices include keyboards, mice, trackpads, trackballs, microphones, and drawing tablets. Output devices include video displays, speakers, inkjet printers, laser printers, and dye-sublimation printers. An I/O device may also provide mass storage for the computer system 200 such as a hard disk drive, a floppy disk drive for receiving floppy disks such as 3.5-inch, 5.25-inch disks or
ZIP disks, a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive, tape drives of various formats, and USB storage devices such as the USB Flash Drive line of devices manufactured by Twintech Industry, Inc. of Los Alamitos, California.

In further embodiments, an I/O device 230 may be a bridge between the system bus 220 and an external communication bus, such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, or a Serial Attached small computer system interface bus.

General-purpose desktop computers of the sort depicted in FIGs. 2A and 2B typically operate under the control of operating systems, which control scheduling of tasks and access to system resources. Typical operating systems include: MICROSOFT WINDOWS, manufactured by Microsoft Corp. of Redmond, Washington; MacOS, manufactured by Apple Computer of Cupertino, California; OS/2, manufactured by International Business Machines of Armonk, New York; and Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah, among others.

In other embodiments, the client computing device 100 may have different processors, operating systems, and input devices consistent with the device. For example, in one embodiment the client computing device 100 is a Zire 71 personal digital assistant manufactured by Palm, Inc. In this embodiment, the Zire 71 uses an OMAP 310 processor manufactured by Texas Instruments, of Dallas, Texas, operates under the control of the PalmOS operating system and includes a liquid-crystal display screen, a stylus input device, and a five-way navigator device.

Referring now to FIG. 3A, a block diagram depicts an embodiment of the network 40 in which the invention may be performed. The servers 30, 32, and 34 can belong to the same domain 38. In the network 40, a domain is a sub-network comprising a group of application servers and client nodes under control of one security database. A domain can include one or more "server
farms." (A server farm is a group of servers that are linked together to act as a single server system to provide centralized administration.) Conversely, a server farm can include one or more domains. For servers of two different domains to belong to the same server farm, a trust relationship may need to exist between the domains. A trust relationship is an association between the different domains that allows a user to access the resources associated with each domain with just one log-on authentication.

In one embodiment, the application server 36 is in a different domain than the domain 38. In another embodiment, the application server 36 is in the same domain as servers 30, 32, and 34. For either embodiment, application servers 30, 32, and 34 can belong to one server farm, while the server 36 belongs to another server farm, or all of the application servers 30, 32, 34, and 36 can belong to the same server farm. When a new server is connected to the network 40, the new server either joins an existing server farm or starts a new server farm.

The client nodes 10, 20 may be in a domain, or may be unconnected with any domain. In one embodiment, the client node 10 is in the domain 38. In another embodiment, the client node 10 is in another domain that does not include any of the application servers 30, 32, 34 or 36. In another embodiment, the client node 10 is not in any domain.

In one embodiment the client node 10 is in the domain 38 and the user of the client node provides user credentials to log onto the client node 10. User credentials typically include the name of the user of the client node, the password of the user, and the name of the domain in which the user is recognized. The user credentials can be obtained from smart cards, time-based tokens, social security numbers, user passwords, personal identification (PIN) numbers, digital certificates based on symmetric key or elliptic curve cryptography, biometric characteristics of the user, or any other means by which the identification of the user of the client node can be obtained and submitted for authentication.
From the user-provided credentials, the client node 10 generates user authentication data. The client node 10 transmits this user authentication data to the server 30. In this embodiment, the client credentials are not transmitted over a network, only the resulting user authentication data is transmitted by the client node.

From the user authentication data and the application-related information, the server 30 can also determine which application programs hosted by the application server farm containing server 30 are available for use by the user of the client node. The server 30 transmits information representing the available application programs to the client node 10. This process eliminates the need for a user of the client node to set up application connections. Also, an administrator of the server farm can control access to applications among the various client node users.

The user authentication performed by the server 30 can suffice to authorize the use of each hosted application program presented to the client node 10, although such applications may reside at another server. Accordingly, in this embodiment, when the client node launches (i.e., initiates execution of) one of the hosted applications, additional input of user credentials by the user will be unnecessary to authenticate use of that application. Thus, a single entry of the user credentials can serve to determine the available applications and to authorize the launching of such applications without an additional, manual log-on authentication process by the client user.

Fig. 3B shows another exemplary process by which the client node 10 initiates execution of an available application and a server presents the results of the application to the client node 10. The user of client node 10 requests the launch of the application 41 (e.g., by clicking on an icon displayed on the client node 10 representing the application). The request 42 for the application is directed to the first server node, in this example server 30. The first server node 30 can execute the application, if the application is on the first server node 30, and return the results to the client node 10. Alternatively, the first server node 30 can indicate (arrow 43) to the client node 10 that the
application 41 is available on another server, in this example server 32. The client node 10 and server 32 establish a connection (arrows 45 and 46) by which the client node 10 requests execution of the application 41. The server 32 can execute the application 41 and transfer the results (i.e., the graphical user interface) to the client node 10.

Fig. 3C shows another exemplary process by which a client node 20 initiates execution of an available application, in this example via the World Wide Web. A client node 20 executes a web browser application 80, such as MICROSOFT INTERNET EXPLORER, manufactured by Microsoft Corporation of Redmond, Washington. The client node 20, via the web browser 80, transmits a request 82 to access a Uniform Resource Locator (URL) address corresponding to an HTML page generated dynamically by server 30. In some embodiments the first response returned 84 to the client node 20 by the server 30 is an authentication request that seeks to identify the client node 20.

A user provides user credentials to the client node 20. The client node 20 generates user authentication data, based upon the user credentials provided to it. The authentication request allows the client node 20 to transmit user authentication data, via the web browser 80, to the server 30 for authentication. Transmitted user authentication data is verified by the server 30.

From the user authentication data and the application-related information, the server 30 can also determine which application programs hosted by the application servers are available for use by the user of the client node 20. The server 30 generates an HTML page containing information representing the available application programs and transmits this to the client node 20, via the web browser 80. The information includes a distinct launch URL address corresponding to each available application.

In this embodiment, the available applications are displayed on the client node 20 via the web browser 80. The client node display has a window 58 in which appears a graphical icon 57 representing an available application program. A user of the client node 20 can launch the application program by clicking the icon 57 with the mouse. The client node 20, via the web browser 80,
transmits a request 86 to access the URL address corresponding to the application launch service residing on server 30. The server node 30 transmits launch information 88 to the client node 20, via the web browser 80, which indicates how a connection can be established to cause execution of the application and transfer the results to the client node 20.

Fig. 3D shows an exemplary process of communication among the client node 10, the first server node, in this example server 30, and the server 32. The client node 10 has an active connection 72 with the server 32. The client node 10 and server 32 can use the active connection 72 to exchange information regarding the execution of a first application program. The user authentication data generated by the client node 10 from received user credentials are stored at the client node. Such storage of the user authentication data is in cache memory.

In this embodiment, the available applications are displayed on the client node 10. The client node display has a window 58 in which appears a graphical icon 57 representing a second application program. A user of the client node 10 can launch the second application program by double-clicking the icon 57 with the mouse. The request passes to the first server node 30 via a connection 59. The first server node 30 indicates to the client node 10 via the connection 59 that the sought-after application is available on server 32. The client node 10 signals the server 32 to establish a second connection 70. The server 32 requests user authentication data from the client node 10 to authenticate access to the second application program. The client node 10 generates user authentication data based upon the stored user authentication data. The client node 10 then transmits the user authentication data to the server 32. Upon a successful authentication, the client node 10 and server 32 establish the second connection 70 and exchange information regarding the execution of the second application program. Accordingly, the client node 10 and the server 32 communicate with each other over multiple connections.

FIG. 4 depicts in more detail a system for remotely authenticating a user of a client node 100 to a server computing device 140. As shown in FIG. 4, the
client computing device 100 includes an authentication module 310 in communication with a thin-client program 320. The authentication module 310 receives user authentication credentials provided for the purposes of authenticating a user to the client computing device 100, the server computing device 140, or both. Received authentication credentials can include username-password combinations, graphical password data, data derived from time-based tokens such as the SecuriID line of tokens manufactured by RSA Security Inc. of Bedford, Massachusetts, challenge-response data, information from smart cards, and biometric information such as fingerprints, voiceprints, or facial features. The authentication module 310 may use the provided authentication credentials to authenticate the user to the client computing device 100. For example, in WINDOWS-based environments, the authentication module 310 may be provided by the MSGINA dynamically-linked library. In other embodiments, for example, in Unix-based environments, the authentication module 310 may be provided by the Unix Pluggable Authentication Manager, using the pam_krb module. In still other embodiments, the authentication module 310 may be provided by the Unix kinit command program.

In the embodiment shown in FIG. 4, the client computing device also includes a security service 312. In other embodiments, the authentication module 310 and the security service 312 are provided as the same dynamically-linked library. The security service 312 provides security services to modules and applications on the client computing device, including the authentication module 310 and the thin-client application 320, such as authentication to the client computing device and authentication to remote hosts or network services. For example, the security service 312, which may be the GSSAPI specified by the Internet Engineering Task Force (IETF) or the SSPI manufactured by Microsoft Corporation of Redmond, Washington, may obtain a Kerberos ticket in response to receipt of the user authentication credentials and use this ticket to obtain additional Kerberos tickets to authenticate the user to remote hosts or network services, at the request of modules or applications on the client computing device. The security service 312 may then generate user authentication data using these Kerberos tickets if needed for remote authentication. In one embodiment, the security service 312 may generate the
user authentication data using an external authentication service, such as a Key Distribution Center in a Kerberos environment or Active Directory in a Windows-based environment.

The security service 312 provides the generated user authentication data, e.g., Kerberos ticket and associated Kerberos authenticator, to the thin-client application 320. The thin-client application 320 transmits the user authentication data to a server computing device 140 for remote authentication of the user. Thus, unlike existing single sign-on mechanisms for server-based computing, user-provided authentication credentials are not transmitted over the network 180 to a server computing device 140. The user authentication data generated by the security service 312 is independent of the method used by the user to authenticate to the client computing device 100. Thus, for example, a Kerberos ticket for the user of client computing device 100 is obtained whether the user uses a username-password combination or a biometric to authenticate to the client computing device 100.

In the embodiment shown in FIG. 4, the thin-client application 320 communicates with the server computing device 140 via a thin-client protocol having one or more virtual channels 335. In these embodiments, the thin-client application 320 loads a virtual channel driver and uses it to send and receive messages on the authentication virtual channel. In some embodiments, the virtual channel driver exposes functions for opening the virtual channel and sending data over it.

The thin-client application 320 passes a data structure to the server computing device 140 for the virtual channel 335 when the thin-client protocol connection is established, indicating to the server-side thin-client application 350 that the authentication virtual channel is available. In one embodiment, the virtual channel data structure for the authentication virtual channel contains the virtual channel information and a representation of the size of the largest data packet the client computing device 100 can accept from or send to the server computing device 140 over the virtual channel 335. The data packet size is constrained by the maximum thin-client size and any specific memory
restrictions imposed by the client computing device 100. In one particular embodiment, the data structure for the authentication virtual channel is defined as:

```c
typedef struct _C2H
{
    VD_C2H   Header;
    U1NT16   cbMaxDataSize;
} C2H, *PC2H;
```

The server-side thin-client application 350 indicates to the thin-client application 320 its intention to perform authentication using the authentication virtual channel 335 by opening the virtual channel and sending a bind request message onto the channel. Once the virtual channel has been opened, the virtual channel driver in the thin-client application 320, in one embodiment, reads a message requesting a binding from the virtual channel, sends a message onto the virtual channel responding to the bind request; and reads a "commit" message from the channel. In one embodiment, the message requesting a binding includes data specifying the protocol version that is supported. In other embodiments, the protocol version can be negotiated between the thin-client application 320 and the server-side thin-client application 350 using the bind request and bind response messages.

The bind request, bind response, and bind commit initialization messages allow the server-side thin-client application 350 and the thin-client application 320 to conduct a 3-way handshake initiated by the server-side thin-client application 350, and negotiate capabilities. A 2-way handshake may be initiated by the server-side thin-client application 350 when the current set of virtual channel capabilities can be negotiated using a 2-way handshake only, but a 3-way handshake is supported to allow more flexibility that might be required by new capabilities or future enhancements to current capabilities. For example, in a 3-way handshake, after receiving a "menu" of capabilities from the server-side thin-client application 350, the thin-client application 320 can exhibit a specific preference or could instead acknowledge a whole set of options pertaining to a specific capability thus letting the server-side thin-client application 350 decide
on a specific option. In a 2-way handshake to be initiated by the thin-client application 320, the thin-client application 320 could not exhibit a specific preference because it might not be supported by the host.

Following channel setup, the virtual channel driver of both the thin-client application 320 and the server-side thin-client application 350 does the following in a loop until a “stop” message or an “error” message is received: retrieve authentication data from the security service 312, 312', providing as input any authentication data sent by the other party via the virtual channel; and send the retrieved authentication data (if any) onto the virtual channel in a data message. If the retrieval of data from the security service 312, 312' returned a “STOP” message, then signal stop and close the authentication virtual channel. In some embodiments the virtual channel driver may reset itself on a “stop” signal. If the retrieval of data from the security service 312, 312' returned a “CONTINUE” message, then continue. If the retrieval of authentication data from the security service 312, 312' returned an “ERROR”, then signal that an error has occurred and close the authentication virtual channel.

As long as “stop” or “error” are not signaled, the virtual channel driver of the thin-client application 320 and the server-side thin-client application 350 are free to exchange data messages until the security service 312, 312' stops producing data buffers to be sent. In some embodiments, the number of messages exchanged may be limited by the virtual channel driver, the server-side thin-client application 350, or the virtual channel 335. In other embodiments, the virtual channel driver of the thin-client application 320 and the server-side thin-client application 350 exchange messages sequentially, that is, two messages are not sent in one direction without a reply to the first being sent in the other. In either embodiment, message exchange can stop after a message has been sent in either direction.

In some particular embodiments, the data messages are sent over the virtual channel Least Significant Double Word (LSDW), Least Significant Word (LSW), Least Significant Byte (LSB) first. In other particular embodiments, the data messages are aligned at a byte boundary and fully packed in memory. In
these embodiments, data fields will be aligned in memory as written to or read from the virtual channel.

Some messages transmitted on the authentication virtual channel span multiple virtual channel packets. To support this, every message must be preceded by a message specifying the length of the next transmitted command. An example of a message that may be used to specify the length of the next command is:

```c
typedef struct_PKT_CMDLEN
{
    UINT32   Length;
    UINT8    Command;
    UINT8    FlagsBitMask;
}_PKT_CMDLEN, *PPKT_CMDLEN;
```

In some of these embodiments, _PKT_CMDLEN_ also contains a command number to indicate what type of message is to follow:

```c
#define CMD_BIND_REQUEST  0x00
#define CMD_BIND_RESPONSE  0x01
#define CMD_BIND_COMMIT    0x02
#define CMD_SSPI_DATA      0x03
```

A _PKT_CMDLEN_ packet containing _Length=0_ indicates that no more data will follow (i.e. a logical channel close).

The server-side thin-client application 350 passes the authentication data it receives over the authentication virtual channel to its security service 312'. If the server-side security service 312' is able to verify the data, it generates an access token representing a logon session for the user, allowing the user to authenticate to the server computing device 140 without resubmitting authentication credentials. An access token is a data object that includes, among other things, a locally unique identifier (LUID) for the logon session. If the server-side security service 312' is not able to verify the data, the user is prompted to resubmit authentication credentials.

In some embodiments, until the server-side security service 312' authenticates the user, the only virtual channel over which the user may
communicate with the server computing device 140 is the authentication virtual channel. In some of these embodiments, after authentication, new virtual channels are initiated for communication. In other embodiments, only one virtual channel exists and it may only be used for authentication-related communications until the user is authenticated, and it may be used for other communications after the user is authenticated.

For embodiments in which the server-side computing device 140 operates under control of a MICROSOFT WINDOWS operating system, the access token generated by the server-side security service 312' is an impersonation token that has only network logon rights. That is, the generated access token is not suitable to use for starting applications to run interactively, as is required in the WINDOWS server-based computing environment. To allow applications to run interactively, a primary access token is needed that has interactive logon rights. In one embodiment, the generated access token is modified to provide the appropriate rights. In another embodiment, a new token is generated for the user.

For embodiments in which the server-side computing device 140 operates under control of a Unix-based operating system, if the server-side security service 312' verifies the authentication data it receives over the authentication virtual channel from the server-side thin-client application 350, the server-side thin-client application 350 will grant the user access to the resources. In these embodiments, the server-side security service 312' does not generate an access token.

In some embodiments, after the server has authenticated the user, the server presents an enumeration of resources available to the user. In these embodiments, the server may create a page describing a display of resources, hosted by a plurality of servers, available to the client computing device. The server may then transmit the created page to the client computing device for display and receive from the client computing device, a request to access one of the hosted resources.
In some of these embodiments, the selected one of the available resources hosted by one of the plurality of servers is then executed without requiring further receipt of user authentication data from the client computing device. In some of these embodiments, the server initiates, in response to successful authentication by the user, a connection from the server to a second server which is hosting a resource available to the user. In these embodiments, the available resource is executed over the connection. In some embodiments, the connection is a virtual channel.

In other embodiments, the first server is hosting the selected one of the available resources. In some of these embodiments, the server makes the resource available to the user over the existing connection. In others of these embodiments, the server makes the resource available to the user over a new connection. In some of those embodiments, the new connection comprises a virtual channel.

The present invention may be provided as one or more computer-readable programs embodied on or in one or more articles of manufacture. The article of manufacture may be a floppy disk, a hard disk, a CD ROM, a flash memory card, a PROM, a RAM, a ROM, or a magnetic tape. In general, the computer-readable programs may be implemented in any programming language. Some examples of languages that can be used include C, C++, or JAVA. The software programs may be stored on or in one or more articles of manufacture as object code.

While the invention has been shown and described with reference to specific preferred embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the following claims.
CLAIMS

What is claimed is:

1. A method for remotely authenticating a user to a server in a server-based computing environment, the method comprising the steps of:

   (a) receiving user credentials at a client computing device;
   (b) generating, at the client computing device, user authentication data based on the received credentials;
   (c) transmitting the generated user authentication data to a server computing device; and
   (d) authenticating, at the server computing device, the user responsive to the transmitted user authentication data.

2. The method of claim 1 wherein step (a) comprises receiving biometric user credentials at a client computing device.

3. The method of claim 1 wherein step (a) comprises receiving a time-based passcode at a client computing device.

4. The method of claim 1 wherein step (a) comprises receiving, at a client computing device, user credentials from a smart card.

5. The method of claim 1 wherein step (b) comprises generating, at the client device, an encrypted bit string representative of the received user credentials.

6. The method of claim 1 wherein step (b) comprises generating, by a security provider subsystem at the client device, user authentication data based on the received credentials.

7. The method of claim 6 wherein the user authentication data is generated using an external authentication service.

8. The method of claim 6 wherein the user authentication data is generated by the Microsoft Windows Security Service Provider Interface (SSPI).
9. The method of claim 6 wherein the user authentication data is generated by the Generic Security Service Application Program Interface (GSSAPI).

10. The method of claim 1 wherein step (c) comprises transmitting at least one virtual channel message to a server computing device, the at least one virtual channel message including at least a portion of the generated user authentication data.

11. The method of claim 1 wherein step (d) comprises:

   (d-a) receiving, at the server computing device, the transmitted user authentication data;

   (d-b) providing the received user authentication data to a locally-executing security subsystem; and

   (d-c) authenticating, by the locally-executing security subsystem, the user in response to the user authentication data.

12. The method of claim 1 further comprising the steps of:

   (e) creating, by the server, a page describing a display of resources hosted by a plurality of servers and available to the client computing device;

   (f) transmitting, by the server, the created page to the client computing device for display;

   (g) receiving, from the client computing device, a request to access one of the hosted resources; and

   (h) executing a selected one of the available resources hosted by one of the plurality of servers without requiring further receipt of user authentication data from the client computing device.

13. The method of claim 1 further comprising the step of initiating, in response to successful authentication by the user, a connection from the client computing device to a server hosting a resource available to the user.
14. The method of claim 13 wherein the connection comprises a virtual channel.

15. The method of claim 1 further comprising the steps of:

(a) generating, by the first server, user authentication data based on the received user authentication data;
(b) transmitting, by the first server, to a second server the user authentication data;
(c) authenticating, by the second server, the user, responsive to the received user authentication data; and
(d) maintaining, by the second server, the connection with the first server.

16. The method of claim 15 further comprising the step of allowing the user to access resources available to the user over the connection between the first server and the second server.

17. A system for remotely authenticating a user to a server in a server-based computing environment, the system comprising:

a client computing device including a client-side authentication module, a client-side security provider subsystem, and a client-side virtual channel driver, said client-side authentication module receiving user authentication credentials and transmitting the credentials to said client-side security provider subsystem, said client-side security provider subsystem generating user authentication data based on the user authentication credentials and transmitting the user authentication data to said client-side virtual channel driver for transmission; and

a server computing device including a server-side virtual channel driver, a server-side security provider subsystem, and a server-side authentication module, the server-side virtual channel driver receiving the transmitted user authentication data and providing the received authentication data to said server-side authentication module, said server-side authentication module
transmitting the user authentication data to said server-side security provider subsystem, said security provider subsystem responding to the user authentication data with an authentication indication.

18. The system of claim 17 wherein said client computing device comprises a personal computer.

19. The system of claim 17 wherein said client computing device comprises a personal digital assistant.

20. The system of claim 17 wherein said client computing device comprises a cell phone.

21. The system of claim 17 wherein said client-side authentication module comprises the Microsoft Windows Graphical Identification and Authentication module.

22. The system of claim 17 wherein said client-side authentication module comprises the Unix Pluggable Authentication Manager using the pam_krb module.

23. The system of claim 17 wherein said client-side authentication module comprises the Unix kinit command program.

24. The system of claim 17 wherein said client-side security provider subsystem comprises the Microsoft Windows Security Service Provider Interface (SSPI).

25. The system of claim 17 wherein said client-side security provider subsystem comprises the Generic Security Service Application Program Interface (GSSAPI).

26. The system of claim 17 wherein said client-side virtual channel driver transmits at least one virtual channel message including at least a portion of the user authentication data.
27. The system of claim 17 wherein said server-side authentication module comprises the Microsoft Windows Graphical Identification and Authentication module.

28. The system of claim 17 wherein said server-side security provider subsystem comprises the Microsoft Windows Security Service Provider Interface (SSPI).

29. The system of claim 17 wherein said server-side security provider subsystem comprises the Generic Security Service Application Program Interface (GSSAPI).

30. The system of claim 17 wherein said server-side security provider subsystem generates a logon session and corresponding access token in response to the user authentication data.

31. An article of manufacture having embodied thereon computer-readable program means for remotely authenticating a user to a server in a server-based computing environment, the article of manufacture comprising:

   computer-readable program means for receiving user credentials;
   computer-readable program means for generating user authentication data based on the received credentials;
   computer-readable program means for transmitting the generated user authentication data to a server computing device; and
   computer-readable program means for authenticating the user responsive to the transmitted user authentication data.

32. The article of manufacture of claim 31 wherein said computer-readable program means for receiving comprises computer-readable program means for receiving biometric user credentials.

33. The article of manufacture of claim 31 wherein said computer-readable program means for receiving comprises computer-readable program means for receiving user credentials from a smart card.
34. The article of manufacture of claim 31 wherein said computer-readable program means for generating comprises computer-readable program means for generating an encrypted bit string representative of the received user credentials.

35. The article of manufacture of claim 31 wherein said computer-readable program means for generating comprises computer-readable program means for generating user authentication data based on the received user credentials.

36. The article of manufacture of claim 31 wherein said computer-readable program means for transmitting comprises computer-readable program means for transmitting at least one virtual channel message to a server computing device, the at least one virtual channel message including at least a portion of the generated user authentication data.

37. The article of manufacture of claim 31 further comprising computer-readable program means for initiating, in response to successful authentication by the user, a connection from the client computing device to a server hosting a resource available to the user.
Fig. 1
Fig. 2A
Fig. 2B
Fig. 3B
Fig. 3D
## INTERNATIONAL SEARCH REPORT

### A. CLASSIFICATION OF SUBJECT MATTER

**IPC 7** G06F1/00

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

### B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

**IPC 7** G06F

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

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

EPO-Internal, WPI Data

### C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>WO 01/63567 A (IDENTIX INCORPORATED; KHIDEKEL, YURI; BALASHOV, ALEX; BASHMAKOV, VLADI) 30 August 2001 (2001-08-30) abstract; figures 1,2,4,5 page 1, line 9 - page 2, line 14 page 3, line 13 - page 4, line 13 page 5, line 24 - page 6, line 29 page 7, line 9 - page 8, line 8 page 9, line 8 - line 17</td>
<td>1-37</td>
</tr>
<tr>
<td>A</td>
<td>US 6 490 679 B1 (TUMBLIN HENRY R ET AL) 3 December 2002 (2002-12-03) column 3, line 16 - line 19 column 5, line 14 - line 57</td>
<td>9,17-30</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of box C

Patent family members are listed in annex.

**X** Further documents are listed in the continuation of box C

**X** Patent family members are listed in annex.

* Special categories of cited documents:

- **A** document defining the general state of the art which is not considered to be of particular relevance
- **E** earlier document but published on or after the international filing date
- **L** document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- **O** document referring to an oral disclosure, use, exhibition or other means
- **P** document published prior to the international filing date but later than the priority date claimed

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

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

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

**S** document of the same patent family

Date of the actual completion of the international search

17 May 2005

Date of mailing of the international search report

27/05/2005

Name and mailing address of the ISA

European Patent Office, P B 5818 Patentlaan 2 NL – 2280 HV Ridgewijk
Tel: (+31-70) 343-2040, Tx 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Powell, D
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>US 2003/187925 A1 (INALA SUMAN KUMAR ET AL) 2 October 2003 (2003-10-02) figure 9 paragraph '0160!' - paragraph '0162!'</td>
<td>12</td>
</tr>
<tr>
<td>A</td>
<td>WO 02/095589 A (IDENTIX INCORPORATED) 28 November 2002 (2002-11-28)</td>
<td></td>
</tr>
<tr>
<td>Patent document cited in search report</td>
<td>Publication date</td>
<td>Patent family member(s)</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>WO 0163567 A</td>
<td>30-08-2001</td>
<td>AU 5170101 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 1269425 A2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 0163567 A2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001027527 A1</td>
</tr>
<tr>
<td>US 6490679 B1</td>
<td>03-12-2002</td>
<td>AU 2967300 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 1147637 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 0042730 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 6199077 B1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002032782 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AU 4574401 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 0180067 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AU 4577401 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 0171563 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AU 758865 B2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AU 4359300 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BR 0011015 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CN 1353838 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 1192558 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2003501725 T</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 0073921 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003120774 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002023104 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 6477565 B1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002078079 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003191832 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 6278993 B1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001000537 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001016034 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001023414 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001032182 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2001051907 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002007330 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002095651 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002019810 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002015480 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2004254881 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2005034055 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AU 1739600 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002184534 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2004107269 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 6842782 B1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2002059369 A1</td>
</tr>
</tbody>
</table>