

(19) World Intellectual Property Organization  
International Bureau



(43) International Publication Date  
6 April 2006 (06.04.2006)

PCT

(10) International Publication Number  
**WO 2006/036277 A2**

(51) International Patent Classification: Not classified

(21) International Application Number:  
PCT/US2005/026354

(22) International Filing Date: 22 July 2005 (22.07.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:  
10/948,381 23 September 2004 (23.09.2004) US

(71) Applicant (for all designated States except US):  
**TRANSCONTINENTAL EVENTS, LLC** [US/US];  
4800 Beach Drive SW, Seattle, Washington 98116 (US).

(72) Inventor; and

(75) Inventor/Applicant (for US only): **ANSEL, Duane, A.**  
[US/US]; 4800 Beach Drive SW, Seattle, Washington  
98116 (US).

(74) Agent: **CHRISTENSEN, Kory, D.**; One Utah Center, 201  
So. Main Street, Suite 1100, Salt Lake City, Utah 84111  
(US).

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

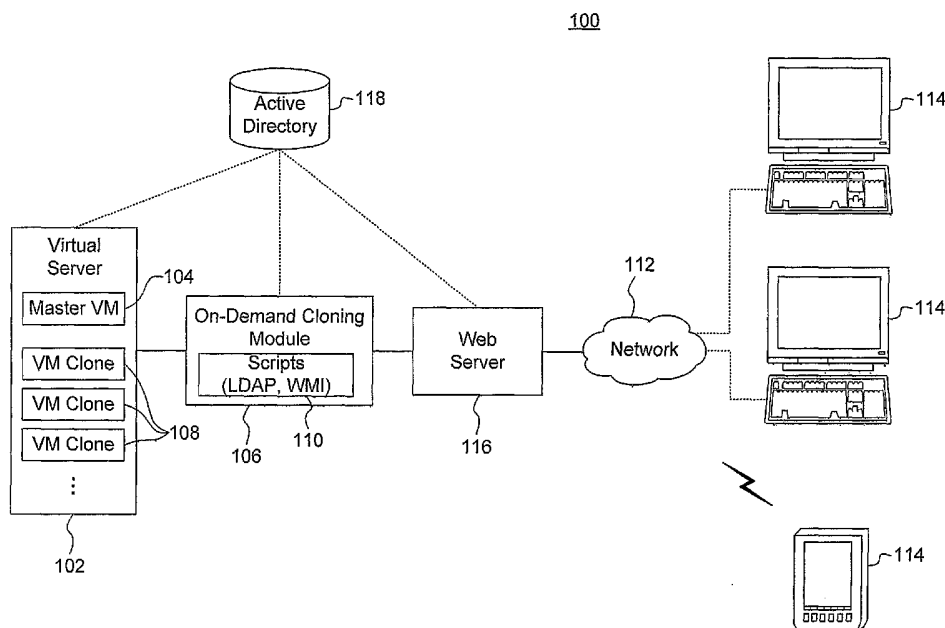
(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT,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

**Published:**

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

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

(54) Title: SYSTEM AND METHOD FOR ON-DEMAND CLONING OF VIRTUAL MACHINES



(57) Abstract: A system for on-demand cloning of virtual machines (VMs) includes a virtual server to host a number of VMs, the virtual server including at least one master VM. The system also includes a Web server to authenticate a user in response to a request for online access to a new VM on the virtual server. In addition, the system includes a cloning module, in communication with the Web server and the virtual server, to automatically clone the master VM to create a unique VM clone for the user on the virtual server responsive to the request.

WO 2006/036277 A2

## SYSTEM AND METHOD FOR ON-DEMAND CLONING OF VIRTUAL MACHINES

### Technical Field

**[0001]** The present invention relates to digital computing and in particular relates to improved delivery of computing resources to remote users so as to minimize the burdens and inconvenience imposed on such users.

### Background of the Invention

**[0002]** Personal computers "PCs" are now ubiquitous; at least in developed countries. PCs come in various form factors, including desktop models, towers, laptops, handheld devices, tablet PCs, etc. As such, the term PC or personal computer is used herein in a very broad sense, and is intended to include a wide variety of computing devices, including but not limited to the kinds just mentioned. By contrast, the term PC, as used herein, would probably not include so-called mainframe computers or other systems or clusters intended to service large numbers, *e.g.*, thousands, of simultaneous users or tasks.

**[0003]** PCs, of which there are now millions in use, can be found in homes and businesses large and small. Since the advent of the PC, somewhere around 1980, they have become dramatically smaller, faster, and cheaper. For most applications, software now provides graphical user interfaces (GUIs) that enable even unsophisticated users to accomplish substantial computing tasks. That being said, using, maintaining and upgrading a modern PC is challenging for most users. In the corporate environment, many companies are large enough to afford an "IT" (information technology) department that can assist users with such tasks. In many smaller businesses and in homes (*e.g.*, SOHO—small office/home office) situations, professional assistance is not readily available. In some cases, the user may employ on-line help or tutorials, but these resources are clumsy and few have the patience required to navigate them. In other cases, the user may pay hourly or per call charges for technical assistance.

**[0004]** To illustrate the burdens and challenges imposed on computer users, consider the following examples:

- Replacing a failed disk drive
- Restoring lost data resulting, *e.g.*, from a failed memory device

- Installing frequent updates to the operating system
- Installing frequent software updates or patches to fix security issues
- Adding more user memory (typically DRAM) or other hardware
- Installing or upgrading driver software, such as printer drivers
- Configuring the computer to use various network resources
- Installing new application programs

There are other examples; these are merely illustrative. Some of these tasks are merely tiresome and annoying, while others are require technical competence beyond that of most users. Until now, no current system has been developed to alleviate such burdens on computer users.

**[0005]** Because most of the above-described problems stem from each user having to configure and maintain a separate PC, one approach has been to move as much of the processing as possible to central servers and allow users to access applications hosted by these central servers via the Internet. The burdens placed on the individual PCs are much less, since they only need to have basic Web browser functionality.

**[0006]** For example, Application Service Providers (ASPs) provide applications to individuals or organizations across the Internet or other network connections. Hosted applications can vary, but examples include financial applications (like general ledger or purchasing software) and office applications, e.g., word processors, spreadsheet programs, etc.

**[0007]** Unfortunately, the applications hosted by ASPs are often slow and not very responsive due to system load, network bandwidth, and latency issues. As a result, ASPs cannot generally guarantee a particular quality of service (QoS) for a given application.

**[0008]** Thin-client systems, like Citrix™, as well as emulation programs, such as Attachmate™, sometimes provide better performance. Attachmate™, for example, Web-enables legacy host applications. Such applications run on thin terminals, often in character mode (black screen dumb terminal). However, not every application can be made to work with emulation programs like Attachmate™. Furthermore, setting up these applications is complicated, typically requiring an IT professional to configure and maintain the application.

**[0009]** Another conventional technique that attempts to address the problems described above is virtual machine (VM) technology. VM technology allows multiple operating systems to run concurrently on a single machine. Each VM emulates the hardware and software of a stand-alone computer, including its CPU, sound, video, and network adapters, etc., in a self-contained, isolated environment.

**[0010]** In particular, Virtual PC™ and Virtual Server™, developed by Connectix and Microsoft, allow one or more legacy operating systems to run on the same computer system as the current Windows operating system. Today, many older x86-based operating systems are supported by Virtual PC and Virtual Server. Virtual PC for Mac allows for one or more other operating systems to run the Macintosh operating system, allowing users to run a Windows operating system and Windows applications on a Mac.

**[0011]** VM technology serves a variety of purposes. It enables hardware consolidation, because multiple operating systems can run on one computer. Key applications for VM technology include cross-platform integration as well as the following:

- Legacy application migration. Legacy operating systems and applications can run on new hardware along with more recent operating systems and applications.
- Server consolidation. If several servers run applications that consume only a fraction of the available resources, VM technology can be used to enable them to run side by side on a single server, even if they require different versions of the operating system or middleware.
- Isolation for development, testing and technical support. Each VM acts as a separate environment, which reduces risk and enables developers to quickly recreate different operating system configurations or compare versions of applications designed for different operating systems. In addition, a developer can test an early version of an application in a VM without fear of destabilizing the system for other users.
- Software demonstrations. VM technology allows users to recreate a clean operating system environment or system configuration quickly.

- Courseware delivery and training. Organizations can configure and deploy a variety of training scenarios quickly.
- Software distribution. IT groups can deliver prevalidated configurations for complex software solutions for fast deployment.

Unfortunately, VM technology also has limitations. For example, an IT professional is generally required to create and set up each VM running on a Virtual Server. There is no automated method by which a novice user may remotely log into a Virtual Server and have a VM automatically created and initialized with all of the software and settings needed for the user to immediately begin working on a desired task.

#### Brief Description of the Drawings

- [0012]** FIG. 1 is a block diagram of a system for on-demand cloning of virtual machines;
- [0013]** FIG. 2 is a flowchart of a process for on-demand cloning of virtual machines;
- [0014]** FIG. 3 is a screen shot of a shortcut;
- [0015]** FIG. 4 is a screen shot of a sign-in page;
- [0016]** FIG. 5 is a screen shot of a subscriber profile form;
- [0017]** FIG. 6 is a screen shot of a user properties window;
- [0018]** FIG. 7 is a screen shot of a user properties window;
- [0019]** FIG. 8 is a screen shot of a Web page displayed while a VM clone is being created;
- [0020]** FIG. 9 is a screen shot of a user interface within a virtual server showing running VMs;
- [0021]** FIG. 10 is a screen shot of a boot process for an operating system;
- [0022]** FIG. 11 is a screen shot of a boot process for an operating system;
- [0023]** FIG. 12 is a screen shot of a login process for an operating system;
- [0024]** FIG. 13 is a screen shot of a shutdown process for an operating system;
- [0025]** FIG. 14 is a screen shot of a fully-configured and running VM clone;
- [0026]** FIG. 15 is a flowchart of a process for on-demand cloning of virtual machines in a corporate environment;
- [0027]** FIG. 16 is a screen shot of a sign-in page in a corporate environment;

[0028] FIG. 17 is a screen shot of a Web page displayed when a user is authenticated; and

[0029] FIG. 18 is a screen shot of a cloned virtual machine within a browser.

#### Detailed Description

[0030] Reference is now made to the figures in which like reference numerals refer to like elements. For clarity, the first digit of a reference numeral indicates the figure number in which the corresponding element is first used.

[0031] In the following description, numerous specific details of programming, software modules, user selections, network transactions, database queries, database structures, etc., are provided for a thorough understanding of the embodiments of the invention. However, those skilled in the art will recognize that the invention can be practiced without one or more of the specific details, or with other methods, components, materials, etc.

[0032] In some cases, well-known structures, materials, or operations are not shown or described in detail in order to avoid obscuring aspects of the invention. Furthermore, the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0033] FIG. 1 is a block diagram of a system 100 for on-demand cloning of virtual machines (VMs). As previously noted, a VM emulates the hardware and software of a stand-alone computer, including its CPU, sound, video, and network adapters, etc., in a self-contained, isolated environment. In one embodiment, the system 100 includes one or more virtual servers 102, such as Microsoft Virtual Server 2005™, for creating and managing one or more VMs.

[0034] As will be described in greater detail hereafter, the virtual server 102 may be pre-configured with one or more master VMs 104, which are "templates" of PCs including all of the necessary software and settings. In one embodiment, the system 100 also includes an on-demand cloning module 106 for automatically creating a VM clone 108, based on one of the master VMs 104, whenever a user needs to use a PC with a particular configuration. As explained below, the cloning of VMs may be accomplished, in one embodiment, using one or more scripts 110, such as LDAP (Lightweight Directory Access Protocol) and WMI (Windows Management Instrumentation) scripts.

**[0035]** Each VM clone 108 may be accessed and used over a network 112, such as a Local Area Network (LAN) or the Internet, by a different remote computer 114. In one embodiment, the ASPX framework may be used, although the system 100 is not limited to any particular framework or programming language. As illustrated, remote computers 114 may be embodied in a variety of form factors, including desktop and pocket computers. Moreover, the remote computers 114 may access the network using any both wired and wireless technology.

**[0036]** A standard Web server 116, such as Microsoft's Internet Information Server (IIS) or the like, may provide the remote computers 114 with online access to VMs on the virtual server 102. Such access may include, for example, replicating screen displays generated by the VMs on the remote computers 114, and forwarding user input from the remote computers 114 to the respective VMs.

**[0037]** In one embodiment, an active directory 118 provides authentication and directory services to the virtual server 102, on-demand cloning module 106, and Web server 116. An active directory 118 is a hierarchical collection of network resources that can contain users, computers, printers, and other Active Directories. Active Directory Services (ADS) allow administrators to handle and maintain all network resources from a single location.

**[0038]** FIG. 2 is a flowchart of an exemplary process for on-demand cloning of VMs. While the process is shown in the context of a subscriber establishing a new VM for use over the Internet, a similar process could provide VMs to employees within a corporation or the like, as will be described hereafter.

**[0039]** Referring to block 200 of FIG. 2A, a subscriber uses a Web browser to access the Web server 116 of FIG. 1. This may be accomplished, for example, by providing the Web browser with a URL (Uniform Resource Locator) of the Web server 116, e.g., [www.myvpc.net](http://www.myvpc.net).

**[0040]** Continuing to block 202, the subscriber clicks on a shortcut displayed in the browser in order to open a sign-in page. An example of a shortcut 302 is shown in FIG. 3. One possible sign-in page 402 is shown in FIG. 4. In this example, the sign-in page 402 is a .NET Passport sign-in page, although other authentication technologies may be used. In block 204, the subscriber authenticates with the system 100 by providing one or more login credentials and password (assuming a .NET Passport has already been established for the subscriber), after which the subscriber may click on a "sign-in" button 404 or the like.

**[0041]** In block 206, the system may search the active directory 118 of FIG. 1 for an existing subscriber based on the subscriber's PUID (.NET Passport Unique ID). In one embodiment, the PUID is returned by the sign-in page 402 when the subscriber is successfully authenticated. NET Passport is a Microsoft-operated service that provides Internet authentication for Web sites, no matter what kind of devices they use for access. It provides reliable Internet authentication and allows users to sign in once to access a variety of .NET Passport-enabled Web sites. To log in to a Passport-enabled site, users present credentials to Passport (via the SSI interface) and get an encrypted ticket cookie containing PUID that is decrypted by the site and used to authenticate them. A PUID is a 16-character string in the HexPUID property of the PassportIdentity class.

**[0042]** If an existing subscriber (PUID) is found in the active directory 118, control passes to block 208, in which the subscriber is taken to an ActiveX page that displays, in the subscriber's browser, his or her VM in its last known state. While ActiveX is used to display the VM in one embodiment, other technologies may be used within the scope of the invention.

**[0043]** If, however, an existing subscriber is not found, control passes to block 210 of FIG. 2B, in which the subscriber is asked to complete a profile form. As shown in FIG. 5, a subscriber profile form 502 may require the subscriber to provide, for example, his or her name, address, telephone number, billing information, and the like.

**[0044]** Once the form 502 has been completed, control passes to block 212, in which the on-demand cloning module uses a script 110 to create a new user object in the active directory 118 based, in one embodiment, on the subscriber's PUID. An example screen shot showing the creation of a new user is provided in FIG. 6. As illustrated, the new subscriber is identified by the 16-character PUID 602. The relevant information may be automatically completed in the "General" tab of the user properties window 604 based on the information provided in the subscriber profile form 502. An example script 110 for carrying out this process is as follows:

```
Imports System.DirectoryServices
```

```
Public Class User_Signup
```

```
    Inherits System.Web.UI.Page
```

```
    Protected WithEvents TABLE1 As System.Web.UI.WebControls.Table
```



```
        Protected WithEvents btnSignUp As
System.Web.UI.WebControls.Button
        Protected WithEvents txtUsername As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtPassword As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtPasswordConfirm As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtFirstName As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtLastName As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtStreet As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtCity As System.Web.UI.WebControls.TextBox
        Protected WithEvents txtState As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtPostalCode As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtCountry As
System.Web.UI.WebControls.TextBox
        Protected WithEvents txtEmail As
System.Web.UI.WebControls.TextBox
```

```
#Region " Web Form Designer Generated Code "
```

```
    'This call is required by the Web Form Designer.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub
InitializeComponent()
```

```
End Sub
```

```
Private Sub Page_Init(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Init
```

```
    'CODEGEN: This method call is required by the Web Form
Designer
```

```
    'Do not modify it using the code editor.
```

```
    InitializeComponent()
```

```
End Sub
```

```
#End Region
```

```

Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    'Put user code to initialize the page here
End Sub

Private Sub SignUp_Click(ByVal s As System.Object, ByVal e As
System.EventArgs) Handles btnSignUp.Click
    'Add User to Active Directory
    Dim strDisplayName = txtFirstName.Text & " " &
txtLastName.Text

    Try
        Dim strPath As String =
ConfigurationSettings.AppSettings("LDAPString")
        Dim myDirectoryEntry As New DirectoryEntry(strPath,
ConfigurationSettings.AppSettings("LDAPUser"), "Amsterdam05")
        Dim myNewUser As New DirectoryEntry()

        myDirectoryEntry.UsePropertyCache = True
        myNewUser = myDirectoryEntry.Children.Add("CN=" &
txtFirstName.Text & " " & txtLastName.Text, "user")
        myNewUser.Properties("sAMAccountName").Value =
txtUsername.Text
        myNewUser.Properties("givenname").Add(txtFirstName.Text)

        myNewUser.Properties("sn").Add(txtLastName.Text)
        myNewUser.Properties("displayname").Add(strDisplayName)

        'myNewUser.Properties("homeDirectory").Add(ConfigurationSettings.AppS
ettings("PhysicalPathtoVMs") & txtUsername.Text)
        myNewUser.CommitChanges()

        'myNewUser.Password = txtPassword.Text
        myNewUser.Invoke("SetPassword", "Amsterdam0333")
        Dim userflags As Object =
myNewUser.Properties("UserAccountControl")
        myNewUser.Properties("UserAccountControl").Value = &H220
        myNewUser.Properties("pwdlastset").Value = 0
        'myNewUser.Properties("memberOf")(0).Value = "CN=VM
User,OU=VM_Accounts,DC=TCEDMZ,DC=local"

```

```

myNewUser.CommitChanges ()

System.Diagnostics.Process.Start(ConfigurationSettings.AppSettings("P
hysicalPathToTools") & "InitialSignup\InitialSignup",
txtUsername.Text)

System.Diagnostics.Process.Start(ConfigurationSettings.AppSettings("P
hysicalPathToTools") & "RegisterVM\RegisterVM", "dmzvpc03 " &
txtUsername.Text & "_Original-XPPro.vmc " &
ConfigurationSettings.AppSettings("PhysicalPathToVMs") &
txtUsername.Text & "\")
    Catch ex As Exception
        Response.Write(ex.Message)
    End Try
End Sub
End Class

```

**[0045]** Once the new user has been created in the active directory 118, the script 110 continues at block 214 to add the new user to the security group of the active directory 118, as well as to set the primary group and give the user a full permission set. As shown in FIG. 7, the script 110 may do this automatically within the "Member of" tab of the user properties window 604.

**[0046]** Continuing to block 216, the on-demand cloning module 106 then creates a VM clone 108 from a pre-defined master VM 104 for the new user. In certain embodiments, the user may be able to select from a number of master VMs 104 that have been optimized for different tasks, *i.e.*, word processing, accounting, gaming, etc. The cloning is accomplished, in one implementation, using a batch file that copies a base image of the master VM 104, creating a new VM clone 108 uniquely renamed for the new subscriber based on the subscriber's PUID 602 (or other unique information, such as e-mail address, login credentials, or other identifying information provided by the user). An example script 110 for carrying out this process is as follows:

```

Imports System.IO
Module Module1
    Sub Main(ByVal CmdArgs() As String)
        ' 1. Create New User Directory under VirtualMachines Dir

```

```

' 2. Move 1st available Master XPPro VM

' Syntax: InitialSignup
'   CmdArgs:
'       0 - Username

Dim config As Configuration.ConfigurationSettings
Dim i As Integer
Dim objFileWriter As StreamWriter

For i = 1 To 5
    If File.Exists(config.AppSettings("MoveVM-
XPProMasterFrom") & i & ".vmc") Then
        ' Try
        If Not Directory.Exists(config.AppSettings("VMPath")
& CmdArgs(0)) Then

Directory.CreateDirectory(config.AppSettings("VMPath") & CmdArgs(0))
            End If
            System.IO.File.Move(config.AppSettings("MoveVM-
XPProMasterFrom") & i & ".vmc", config.AppSettings("VMPath") &
CmdArgs(0) & "\" & CmdArgs(0) & "_Original-XPPro.vmc")
            System.IO.File.Move(config.AppSettings("MoveVM-
XPProMasterFrom") & i & ".vhd", config.AppSettings("VMPath") &
CmdArgs(0) & "\" & CmdArgs(0) & "_Original-XPPro.vhd")

            FileReplace(config.AppSettings("VMPath") & CmdArgs(0)
& "\" & CmdArgs(0) & "_Original-XPPro.vmc", "111222333444555",
CmdArgs(0) & "\" & CmdArgs(0) & "_Original-XPPro")
            FileReplace(config.AppSettings("VMPath") & CmdArgs(0)
& "\" & CmdArgs(0) & "_Original-XPPro.vmc", "555444333222111", "UTF(-
16)")

            Exit Sub
        End If
    Next
End Sub

Sub FileReplace(ByVal filePath As String, ByVal oldString As
String, ByVal newString As String)
    Dim sr As StreamReader = New StreamReader(filePath)
    Dim file As String = sr.ReadToEnd()

```

```

        sr.Close()

        file = file.Replace(oldString, newString)

        Dim sw As StreamWriter = New StreamWriter(filePath)
        sw.Write(file)
        sw.Close()
    End Sub

End Module

```

**[0047]** While the VM clone 108 is being created, the user may be prompted to wait, as shown in the browser of FIG. 8, since the process of copying the base image may take several seconds or longer depending on its size.

**[0048]** In certain embodiments, multiple virtual servers 102 are provided for load balancing. Accordingly, in block 218, the script 110 may pull a list of potential virtual servers 102 to host the new VM clone 108. The system then attempts to determine which virtual server 102 has the resources (e.g., available memory, CPU resources, etc.) to best handle the new VM clone 108. In one implementation, this is done in block 220 using an XML (eXtensible Markup Language) file, which pings each of the virtual servers 102 to determine resource availability.

**[0049]** Once a virtual server 102 has been selected, control passes to block 222, in which the new VM clone 108 is added to the virtual server 102. This may be accomplished, in one embodiment, using a RegisterVM.exe program, which has the usage "RegisterVM ServerName MachineName MachinePath." Other programs or techniques may also be used for this purpose. A supporting script 110 for this process is as follows:

```

Imports System.Reflection
Imports System.Runtime.InteropServices
Imports Microsoft.VirtualServer.Interop

Public Class InitVS

    <DllImport("ole32.dll", _
        PreserveSig:=False, _
        ExactSpelling:=True, _
        EntryPoint:="CoInitializeSecurity", _

```

```

        CallingConvention:=CallingConvention.StdCall, _
        SetLastError:=False)> _
Private Shared Sub CoInitializeSecurity( _
    ByVal pSD As IntPtr, _
    ByVal cAuthSvc As UInt32, _
    ByVal asAuthSvc As IntPtr, _
    ByVal pReserved1 As IntPtr, _
    ByVal dwAuthnlevel As UInt32, _
    ByVal dwImpLevel As UInt32, _
    ByVal pAuthInfo As IntPtr, _
    ByVal dwCapabilities As UInt32, _
    ByVal pvReserved2 As IntPtr)
End Sub

Private Const RPC_C_AUTHN_LEVEL_NONE As Long = 1
Private Const RPC_C_IMP_LEVEL_IMPERSONATE As Long = 3
Private Const EOAC_NONE As Long = 0

Public Sub New()
    CoInitializeSecurity(System.IntPtr.Zero, _
        Convert.ToUInt32(0), _
        System.IntPtr.Zero, _
        System.IntPtr.Zero, _
        Convert.ToUInt32(RPC_C_AUTHN_LEVEL_NONE), _
        Convert.ToUInt32(RPC_C_IMP_LEVEL_IMPERSONATE), _
        System.IntPtr.Zero, _
        Convert.ToUInt32(EOAC_NONE), _
        System.IntPtr.Zero)
End Sub

Public Function GetVMVirtualServerClass(ByVal server As String) _
    As VMVirtualServerClass

    Dim typeVSClass As Type
    Dim typeDCOM As Type
    Dim objDCOM As Object

    typeVSClass = GetType(VMVirtualServerClass)
    typeDCOM = Type.GetTypeFromCLSID(typeVSClass.GUID, server,
True)
    objDCOM = Activator.CreateInstance(typeDCOM)

```

```

        GetVMVirtualServerClass = CType( _
            Marshal.CreateWrapperOfType(objDCOM, typeVSClass), _
            VMVirtualServerClass)
    End Function

    'Get VMVirtualServerClass instance from local server using COM
    Public Function GetVMVirtualServerClass() As VMVirtualServerClass
        GetVMVirtualServerClass = New VMVirtualServerClass
    End Function
End Class

Module RegisterVM
    <MTAThread()> _
    Sub Main(ByVal CmdArgs() As String)          ' Initialize COM with
        Impersonate first
        Dim myApp As InitVS
        myApp = New InitVS
        Dim myApp1 As InitVS

        ' Check command line arguments help flag
        If CmdArgs.Length > 0 Then
            If Left$(CmdArgs(0), 1) = "-" Or Left$(CmdArgs(0), 2) =
"/?" Then
                Dim sHelp As String
                sHelp = "USAGE: RegisterVM {servername} {vmcfile}
{vmcpath}
                MsgBox(sHelp, MsgBoxStyle.OKOnly, "ShowVSVersion
Help")

                Exit Sub
            End If
        End If

        ' Connect locally or remotely
        Dim myVS As VMVirtualServer
        Dim myVM As VMVirtualMachine

        Dim myVSErrMsg As String
        Try
            If CmdArgs.Length > 0 Then

```

```

        myVSErrMsg = " at " & CmdArgs(0)
        myVS = myApp.GetVMVirtualServerClass(CmdArgs(0))
        myVM = myApp1.GetVMVirtualServerClass(CmdArgs(0))
    Else
        myVSErrMsg = " on local machine"
        myVS = myApp.GetVMVirtualServerClass()
        myVM = myApp1.GetVMVirtualServerClass()
    End If

    Catch ex As Exception
        MsgBox("Cannot connect to Virtual Server" & myVSErrMsg,
        MsgBoxStyle.OKOnly, "ShowVSVersion")
        Exit Sub
    End Try

    myVS.CreateVirtualMachine(CmdArgs(1), CmdArgs(2))
    myVM.AddHardDiskConnection(CmdArgs(2) & CmdArgs(1),
    VMDriveBusType.vmDriveBusType_IDE, 0, 1)
    End Sub
End Module

```

**[0050]** Once the new VM clone 108 has been added to the virtual server 102, it will appear in a listing of VMs on the virtual server 102, as shown in FIG. 9. In the present example, the new VM clone 108 is at the top of the list and shows a running time of 6 seconds. Note that each of the VM clones 108 have a "virtual machine name" based on a different PUID 602 for a respective subscriber.

**[0051]** Continuing at block 224, the script 110 starts and tests the new VM clone 108. FIG. 10 illustrates a POST (Power On Self Test) screen generated by the VM clone during the booting process, which is displayed in the subscriber's browser. Thereafter, as shown in FIG. 11, the Windows XP™ (or another OS) startup screen is displayed as the VM clone 108 continues by loading the installed OS.

**[0052]** At block 226, the script 110 may initiate an auto-login procedure, as shown in FIG. 12, causing the VM clone 108 to automatically log into the "Administrator" account, or another account with suitable privileges to permit system-level changes to various OS settings, as described below. A modification to the system registry for accomplishing the auto-login procedure is as follows:

```
Windows Registry Editor Version 5.00
```



```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon]
"DefaultUserName"="administrator"
"DefaultUserPassword"=""
"AutoAdminLogon "="1"
```

**[0053]** For example, at block 228, the system may run a WMI script against the new VM clone 108, changing the computer name based, in one embodiment, on information from the subscriber profile form 502. An example of a suitable WMI script is as follows:

```
strComputer = "."
Set objWMIService =
GetObject("winmgmts:{impersonationLevel=impersonate}!\\" &
strComputer & "\root\cimv2")
Set colComputers = objWMIService.ExecQuery ("Select * from
Win32_ComputerSystem")
For Each objComputer in colComputers
    err = ObjComputer.Rename(WScript.arguments.item(0))
    Wscript.Echo err
Next
```

**[0054]** Similarly, at block 230, the system may run a WMI script adding the new VM clone 108 to a particular domain or workgroup, also based on information from the subscriber profile form 502. For example, the user may specify that he or she wants to be a part of a particular workgroup or domain of a "virtual office" provided by the present system 100.

**[0055]** At block 232, the script 110 will reboot the new VM clone 108 to make all of the necessary changes to the system registry (or other similar data structure in a non-Windows 2000/XP environment). The process of shutting down Windows and rebooting is illustrated in FIG. 13, which may be performed automatically without the need for intervention by the subscriber. One possible script for carrying out this procedure may include the command, "shutdown /r /t 1 /d p:2:3 /c 'Shutting Down VM.'"

**[0056]** Once the VM clone 108 has been rebooted, the subscriber is taken, at block 234 to an ActiveX webpage displaying the new VM clone 108, as shown in

FIG. 14. The subscriber can then, at block 236, access VM from any PC connected to the Internet using their Microsoft Passport.

**[0057]** Because the VM clones 108 are centrally hosted on one or more virtual servers 102, they may be easily updated or repaired by service personnel without having to make an in-home visit. For example, updates to virus scanners or the operating system, itself, may be automatically made by administrators of the virtual servers 102. Moreover, if the user desires a new software application, the user may simply purchase it online, after which the application may be automatically installed on the user's VM clone 108. Automatic backups of the VM clone 108 may be made daily or even more frequently, permitting service personnel to easily troubleshoot serious problems with a minimum of downtime.

**[0058]** Furthermore, the user may easily "upgrade" the VM clone 108 with more power "hardware." Since the VM clone's "CPU" is merely a share of the processing resources of a physical computer hosting the virtual server 102, that share may be increased or decreased based on the user's needs. In fact, processing, graphics, sound, networking, and other resources can be allocated on an application-by-application basis. For instance, the VM clone 108 may not normally provide computer gaming capability. However, should the user desire to run one of the latest games, his or her "machine" may be temporarily "upgraded" with the requisite virtual hardware, the cost of which would be billed to the user, since the upgraded machine would be using a greater share of the physical host system.

**[0059]** As noted above, the foregoing embodiment is specific to a subscriber creating a new VM clone 108 to be accessed over the Internet. In an alternative embodiment, a corporation may provide its employees with VM clones 108 hosted on a corporate server. Accordingly, rather than providing each employee with a separate, full-featured computer, the corporation may provide each employee with a basic terminal with a Web browser, as well as a hosted VM clone 108 with all of the necessary software and settings for the employee's job description.

**[0060]** The process flow described in FIG. 2 would only be changed in minor ways in a corporate environment. For example, FIG. 15 partially illustrates the corporate process, which begins at block 1500 with the employee accessing any corporate terminal. Moving to block 1502, the employee clicks on a shortcut, which opens an internal MyvPC.local sign-in page 1502, as illustrated in FIG. 16. In this case, the employee would not log in using .NET Passport (as in FIG. 4), but rather,

would log in with the employee's corporate login credentials or a pre-assigned guest/demo/training account.

**[0061]** For greater security, the employee may log in with an external e-mail address. In response, the system 100 may send an e-mail message to the specified address containing a special link to the Web server 116 which facilitates online creation of, or access to, a VM clone 108. Accordingly, the user cannot use a VM clone 108 unless he or she provides a valid external e-mail address during the login procedure.

**[0062]** At block 1504, the user may be authenticated, for example, using standard LDAP authentication. Once the user is authenticated, a screen display similar to the one in FIG. 17 may be shown, which indicates that the user "cp" has been authenticated using LDAP authentication.

**[0063]** Continuing to block 1506, the system 100 may search the MyvPC.local active directory 118 for an existing user based on the employee's login credentials, *i.e.*, username and password. If the user exists in the active directory 118, control passes to block 1508, in which the employee is taken to an ActiveX page, which displays the employee's VM in its last known state. This process is illustrated in FIG. 18.

**[0064]** If, however, the user does not exist in the active directory 118 (*e.g.*, a new employee), control passes to block 1510 in which the employee completes an employee profile form. The employee profile form may be similar to the new subscriber profile form 502 of FIG. 5. Alternatively, the employee profile form may have been completed previously, in which case information from the form may be accessed from a stored location.

**[0065]** Continuing to block 1512, the script 110 will create a new user object in the active directory 118 based on the employee's login credentials. Thereafter, in block 1514, the script 110 will add the new user object to a security group of the active directory 118, as well as to set a primary group for the new user object.

**[0066]** In block 1516, the script 110 will create a clone of a pre-defined (or user-selected) master VM 104. In one embodiment, this is accomplished using a batch file that copies a base image of the master VM 104, creating a new VM clone 108 uniquely renamed for the new employee based, for example, on the employee's login credentials.

**[0067]** The remainder of the process flow may occur essentially as outlined in the remainder of FIG. 2. For example, the system 100 may selectively assign the VM clone 108 to the virtual server 102 best able to accommodate it. Additionally, the system 100 may boot the VM clone 108 within the selected virtual server 102, automatically logging in the VM clone 108 into an “administrator” account or the like in order to enable changing the computer name and/or adding the VM clone 108 to a particular corporate domain or workgroup.

**[0068]** The above-described process of hosting VM clones 108 results in many benefits to a corporation, not the least of which is eliminating the cost of providing a full-featured computer to each employee. In addition, using the principles discussed herein, the corporation may install updates, enforce security policies, and troubleshoot systems much more easily than the conventional model, in which the corporation purchases individual computers for employees.

**[0069]** While specific embodiments and applications of the present invention have been illustrated and described, it is to be understood that the invention is not limited to the precise configuration and components disclosed herein. Various modifications, changes, and variations apparent to those of skill in the art may be made in the arrangement, operation, and details of the methods and systems of the present invention disclosed herein without departing from the spirit and scope of the present invention.

**[0070]** Embodiments of the invention may include various steps, which may be embodied in machine-executable instructions to be executed by a general-purpose or special-purpose computer (or other electronic device). Alternatively, the steps may be performed by hardware components that contain specific logic for performing the steps, or by any combination of hardware, software, and/or firmware.

**[0071]** Embodiments of the present invention may also be provided as a computer program product including a machine-readable medium having stored thereon instructions that may be used to program a computer (or other electronic device) to perform processes described herein. The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, CD-ROMs, DVD-ROMs, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, propagation media or other type of media/machine-readable medium suitable for storing electronic instructions. For example, instructions for performing described processes may be transferred from a remote computer (e.g., a server) to a requesting computer (e.g., a

client) by way of data signals embodied in a carrier wave or other propagation medium via a communication link (e.g., network connection).

What is claimed is:

Claims

1. A system, comprising:
  - a virtual server to host a number of virtual machines (VMs), the virtual server including at least one master VM;
  - a Web server to authenticate a user in response to a request for online access to a new VM on the virtual server; and
  - a cloning module, in communication with the Web server and the virtual server, to automatically clone a master VM to create a unique VM clone for the user on the virtual server responsive to the request.
2. The system of claim 1, wherein the virtual server comprises a base image for the master VM, and wherein the cloning module is to create the unique VM clone by copying the base image and renaming the copied image according to unique information pertaining to the user.
3. The system of claim 2, wherein the unique information comprises a PUID.
4. The system of claim 2, wherein the unique information comprises identification information provided by the user.
5. The system of claim 2, wherein the unique information comprises one or more login credentials of the user.
6. The system of claim 1, wherein the Web server is to authenticate the user with a .NET Passport authentication system that returns a PUID in response to the user being successfully authenticated.
7. The system of claim 1, wherein the cloning module is further to create a new user object in an active directory according to unique information pertaining to the user.
8. The system of claim 7, wherein the unique information comprises a PUID.

9. The system of claim 7, wherein the unique information comprises identification information provided by the user.
10. The system of claim 7, wherein the unique information comprises one or more login credentials of the user.
11. The system of claim 7, wherein the cloning module is further to add the new user object to a security group of the active directory as well as to set a primary group for the new user object.
12. The system of claim 1, wherein the cloning module is further to check resources of a plurality of virtual servers to select a virtual server with adequate resources to host the VM clone and assign the VM clone to the selected virtual server.
13. The system of claim 12, wherein the cloning module is further to boot the VM clone using an auto-login sequence into an administrator account.
14. The system of claim 13, wherein the cloning module is further to execute a script while the VM clone is logged into the administrator account to change a computer name associated with the VM clone.
15. The system of claim 14, wherein the cloning module is further to execute a script while the VM clone is logged into the administrator account to add the VM clone to a domain.
16. The system of claim 14, wherein the cloning module is further to execute a script while the VM clone is logged into the administrator account to add the VM clone to a workgroup.
17. The system of claim 1, wherein the cloning module includes an LDAP script, and wherein the cloning module uses the LDAP script to clone the master VM.

18. The system of claim 1, wherein the cloning module, in response to a user being authenticated, determines whether the user has an existing VM clone and, in response to the user having an existing VM clone, provides the user with online access to the VM clone.

19. The system of claim 18, wherein the cloning module is to determine whether the user has an existing VM clone by checking for a VM clone uniquely named according to a PUID of the user.

20. The system of claim 1, wherein the cloning module is further to prompt the user to select a master VM for cloning from a plurality of master VMs, each master VM being optimized for a different set of tasks.

21. A method, comprising:

providing a virtual server to host a number of virtual machines (VMs), the virtual server comprising at least one master VM;

authenticating a user in response to a request for online access to a new VM on the virtual server; and

automatically cloning a master VM to create a unique VM clone for the user on the virtual server responsive to the request.

22. The method of claim 21, wherein the virtual server comprises a base image for the master VM, and wherein creating the unique VM clone comprises:

copying the base image; and

renaming the copied image according to unique information pertaining to the user.

23. The method of claim 22, wherein the unique information comprises a PUID.

24. The method of claim 22, wherein the unique information comprises identification information provided by the user.

25. The method of claim 22, wherein the unique information comprises one or more login credentials of the user.



26. The method of claim 21, wherein authenticating comprises authenticating the user with a .NET Passport authentication system that returns a PUID in response to the user being successfully authenticated.

27. The method of claim 21, further comprising creating a new user object in an active directory according to unique information pertaining to the user.

28. The method of claim 27, wherein the unique information comprises a PUID.

29. The method of claim 27, wherein the unique information comprises identification information provided by the user.

30. The method of claim 27, wherein the unique information comprises one or more login credentials of the user.

31. The method of claim 27, further comprising adding the new user object to a security group of the active directory as well as to set a primary group for the new user object.

32. The method of claim 21, further comprising:  
checking resources of a plurality of virtual servers to select a virtual server with adequate resources to host the VM clone; and  
assigning the VM clone to the selected virtual server.

33. The method of claim 32, further comprising booting the VM clone using an auto-login sequence into an administrator account.

34. The method of claim 33, further comprising executing a script while the VM clone is logged into the administrator account to change a computer name associated with the VM clone.

35. The method of claim 34, further comprising executing a script while the VM clone is logged into the administrator account to add the VM clone to a domain.

36. The method of claim 34, further comprising executing a script while the VM clone is logged into the administrator account to add the VM clone to a workgroup.

37. The method of claim 21, further comprising using an LDAP script to clone the master VM.

38. The method of claim 21, further comprising:

determining, in response to a user being authenticated, whether the user has an existing VM clone; and

in response to the user having an existing VM clone, providing the user with online access to the VM clone.

39. The method of claim 38, wherein determining whether the user has an existing VM clone comprises checking for a VM clone uniquely named according to a PUID of the user.

40. The method of claim 21, further comprising prompting the user to select a master VM for cloning from a plurality of master VMs, each master VM being optimized for a different set of tasks.

41. An article of manufacture, comprising:

a computer-readable medium comprising computer-readable code including:

computer-readable code to authenticate a user in response to a request for online access to a new virtual machine on a virtual server, the virtual server including at least one master VM; and

computer-readable code to automatically clone a master VM to create a unique VM clone for the user on the virtual server responsive to the request.

42. The article of manufacture of claim 41, wherein the virtual server comprises a base image for the master VM, and wherein the computer-readable code to automatically clone the master VM is to create the unique VM clone by copying the base image and renaming the copied image according to unique information pertaining to the user.

43. The article of manufacture of claim 42, wherein the unique information comprises a PUID.

44. The article of manufacture of claim 42, wherein the unique information comprises identification information provided by the user.

45. The article of manufacture of claim 42, wherein the unique information comprises one or more login credentials of the user.

46. The article of manufacture of claim 41, wherein the computer-readable code to authenticate a user employs a .NET Passport authentication system that returns a PUID in response to the user being successfully authenticated.

47. The article of manufacture of claim 41, wherein the computer-readable code to automatically clone the master VM is further to create a new user object in an active directory according to unique information pertaining to the user.

48. The article of manufacture of claim 47, wherein the unique information comprises a PUID.

49. The article of manufacture of claim 47, wherein the unique information comprises identification information provided by the user.

50. The article of manufacture of claim 47, wherein the unique information comprises one or more login credentials of the user.

51. The article of manufacture of claim 47, wherein the computer-readable code to automatically clone the master VM is further to add the new user object to a security group of the active directory as well as to set a primary group for the new user object.

52. The article of manufacture of claim 41, wherein the computer-readable code to automatically clone the master VM is further to check resources of a plurality of

virtual servers to select a virtual server with adequate resources to host the VM clone and assign the VM clone to the selected virtual server.

53. The article of manufacture of claim 52, wherein the computer-readable code to automatically clone the master VM is further to boot the VM clone using an auto-login sequence into an administrator account.

54. The article of manufacture of claim 53, wherein the computer-readable code to automatically clone the master VM is further to execute a script while the VM clone is logged into the administrator account to change a computer name associated with the VM clone.

55. The article of manufacture of claim 54, wherein the computer-readable code to automatically clone the master VM is further to execute a script while the VM clone is logged into the administrator account to add the VM clone to a domain.

56. The article of manufacture of claim 54, wherein the computer-readable code to automatically clone the master VM is further to execute a script while the VM clone is logged into the administrator account to add the VM clone to a workgroup.

57. The article of manufacture of claim 41, wherein the computer-readable code to automatically clone the master VM uses an LDAP script for cloning the master VM.

58. The article of manufacture of claim 41, wherein the computer-readable code to automatically clone the master VM, in response to a user being authenticated, determines whether the user has an existing VM clone and, in response to the user having an existing VM clone, provides the user with online access to the VM clone.

59. The article of manufacture of claim 58, wherein the computer-readable code to automatically clone the master VM is to determine whether the user has an existing VM clone by checking for a VM clone uniquely named according to a PUID of the user.

60. The article of manufacture of claim 41, wherein the computer-readable code to automatically clone the master VM is further to prompt the user to select a master VM for cloning from a plurality of master VMs, each master VM being optimized for a different set of tasks.

61. An apparatus for providing on-demand access to virtual machines (VMs), comprising:

means for authenticating a user in response to a request for online access to a new VM on a virtual server, the virtual server comprising one or more master VMs representing personal computers optimized for particular sets of tasks; and

means for automatically cloning one of the master VMs to create a unique VM clone for the user on the virtual server responsive to the request.

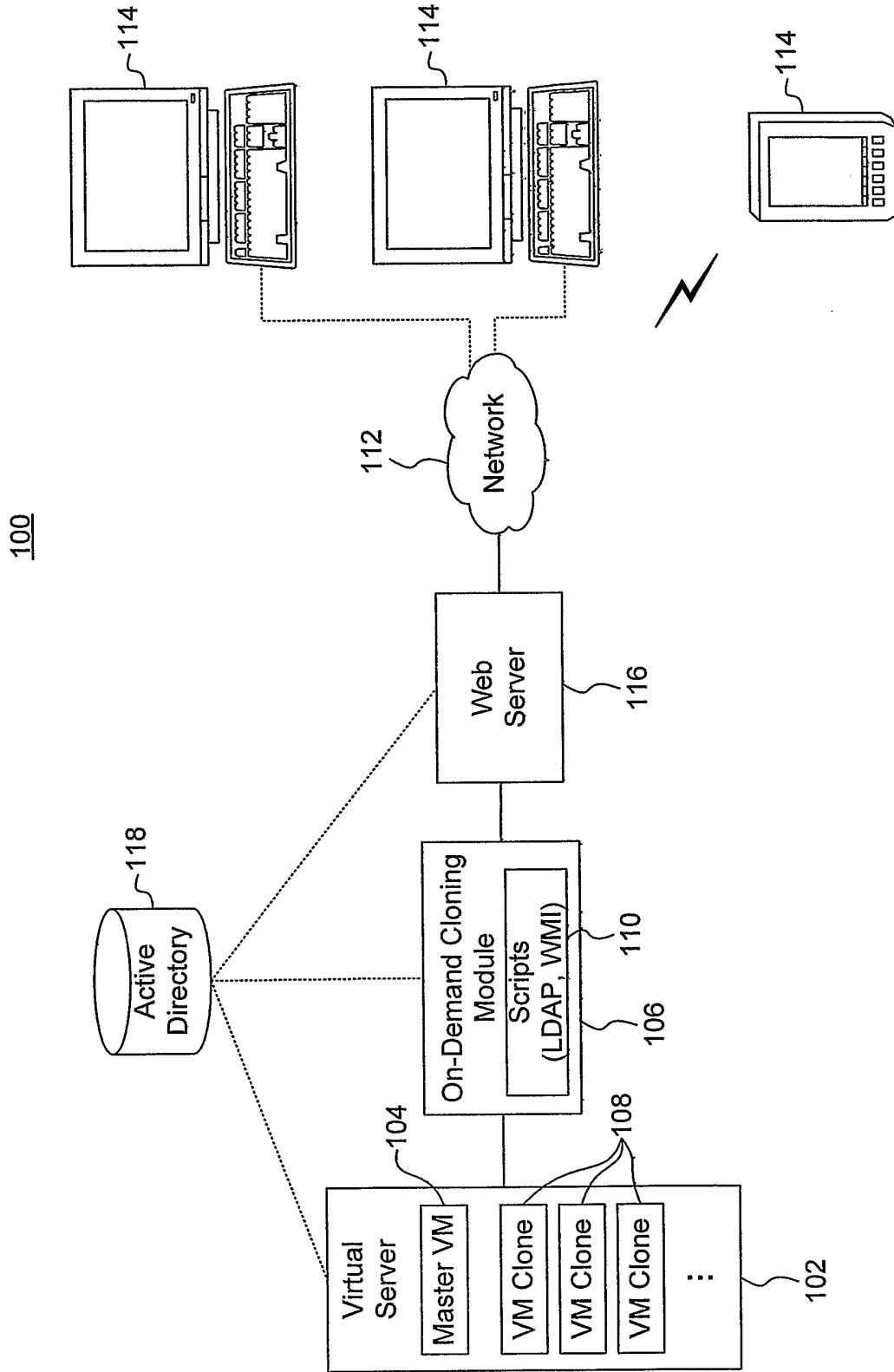


FIG. 1

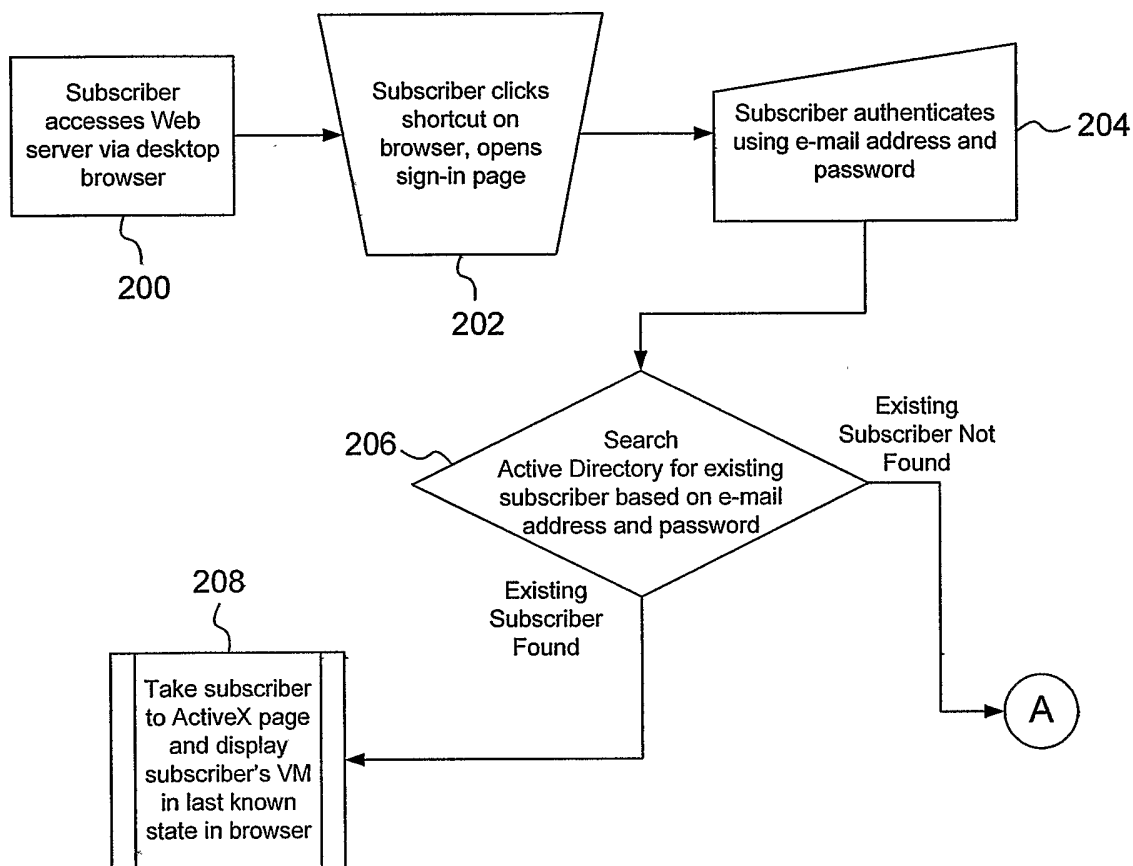


FIG. 2A

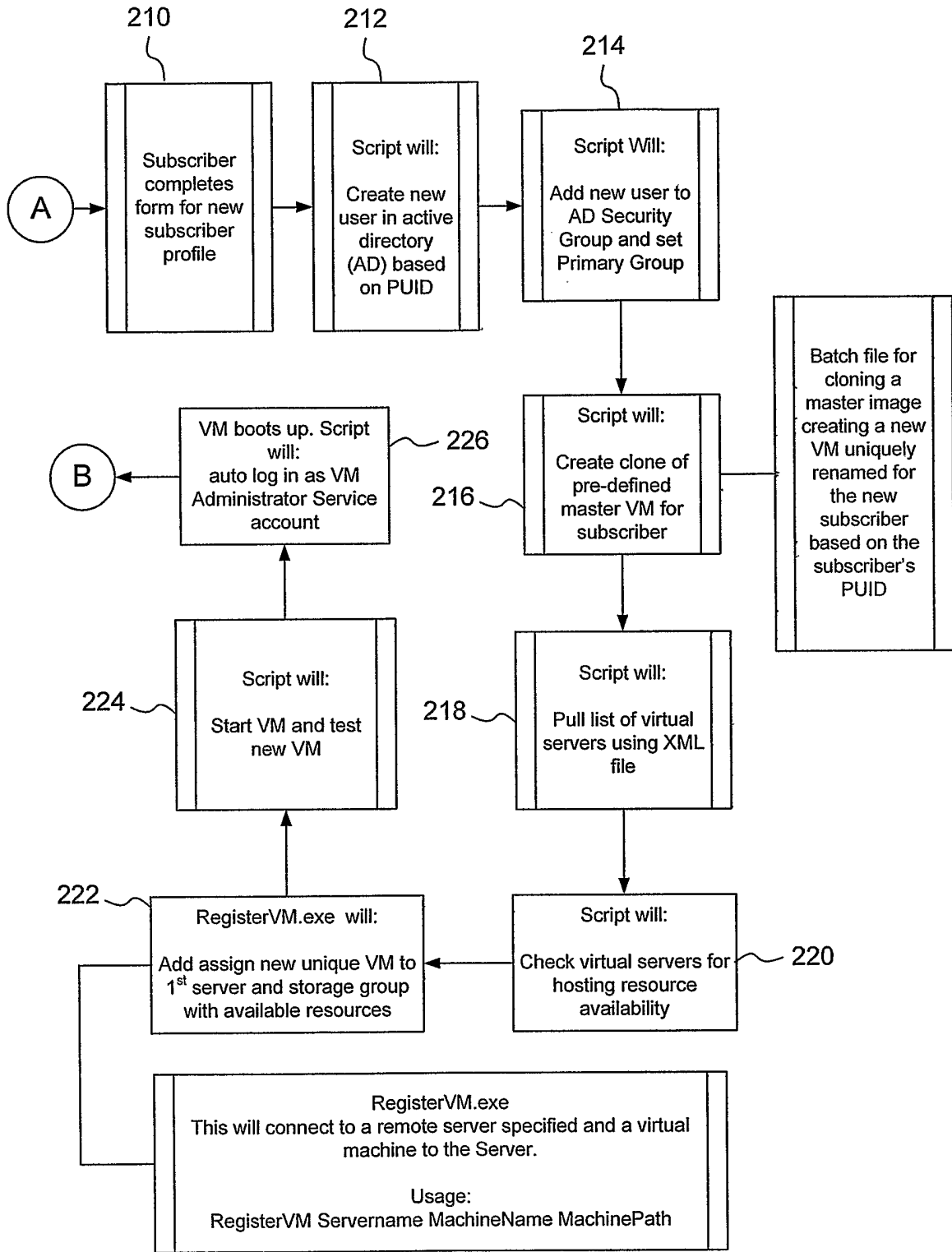


FIG. 2B



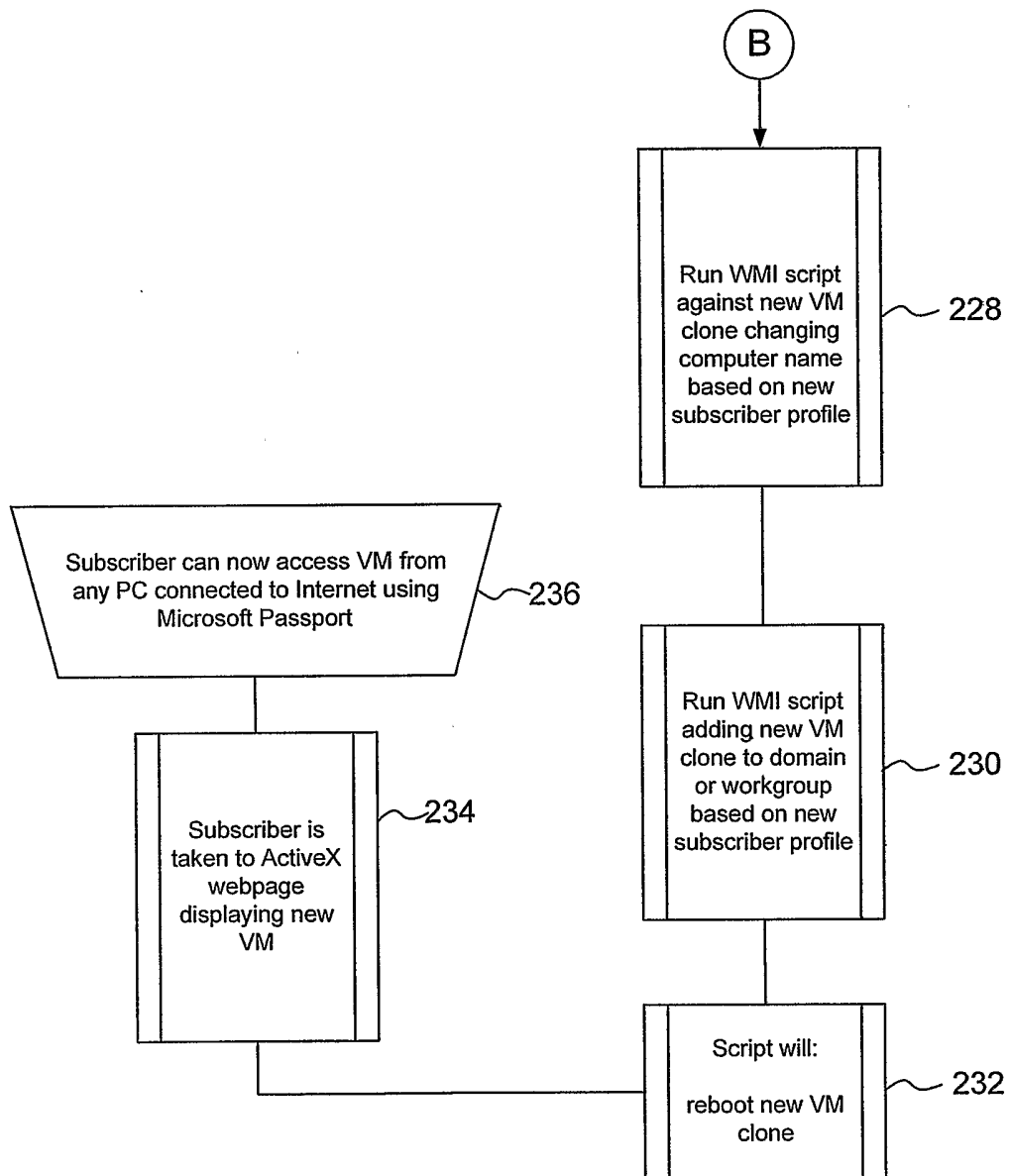


FIG. 2C

5/13

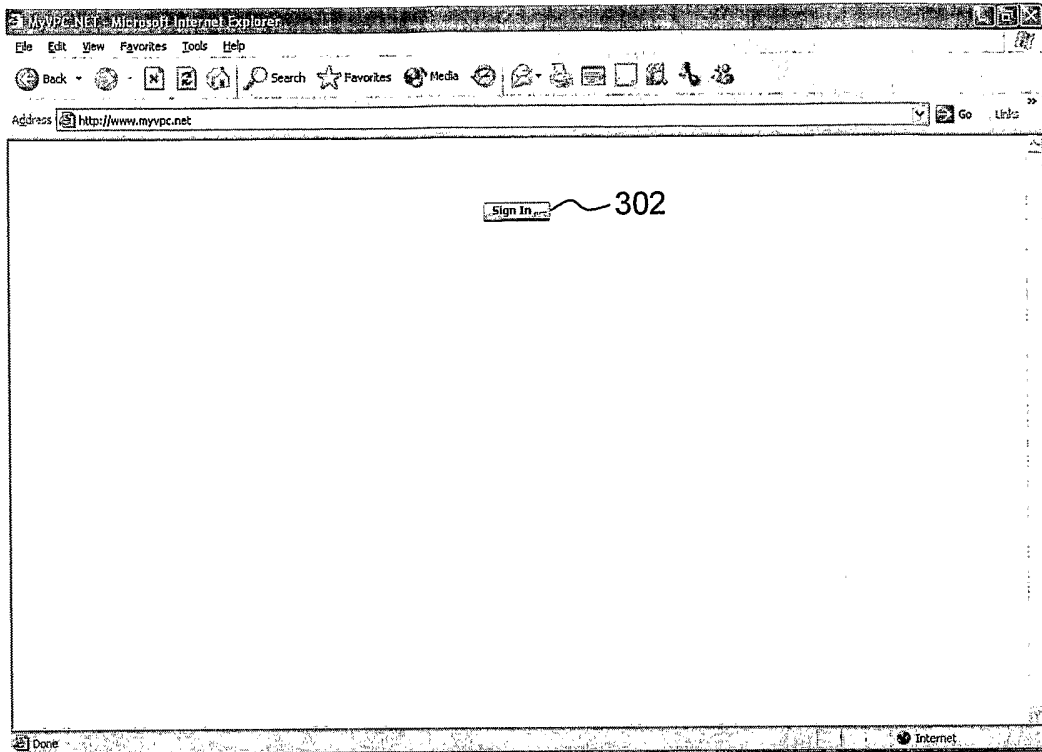


FIG. 3

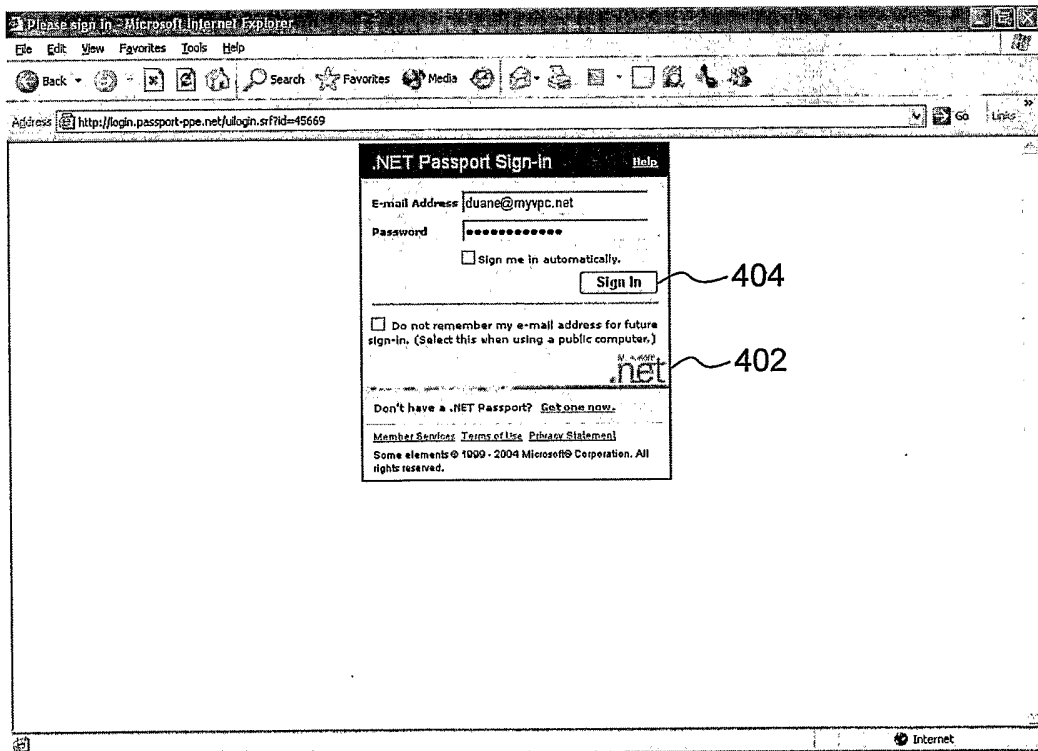


FIG. 4

The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying [http://www.myvpc.net/User\\_Signup.asp](http://www.myvpc.net/User_Signup.asp). The main content area is titled "New Virtual Machine Account" and contains the following form fields:

- Profile Information:** First Name, Last Name, Title, Department, Web Page.
- Address:** Street, Office/Bldg, City, State/Province (dropdown), Postal Code, Country/Region (dropdown).
- Telephone Numbers:** Office, Mobile.
- Account Information:** Email, Account Password.

At the bottom of the form are "Sign Up" and "Cancel" buttons. A callout line labeled "502" points to the right side of the form area.

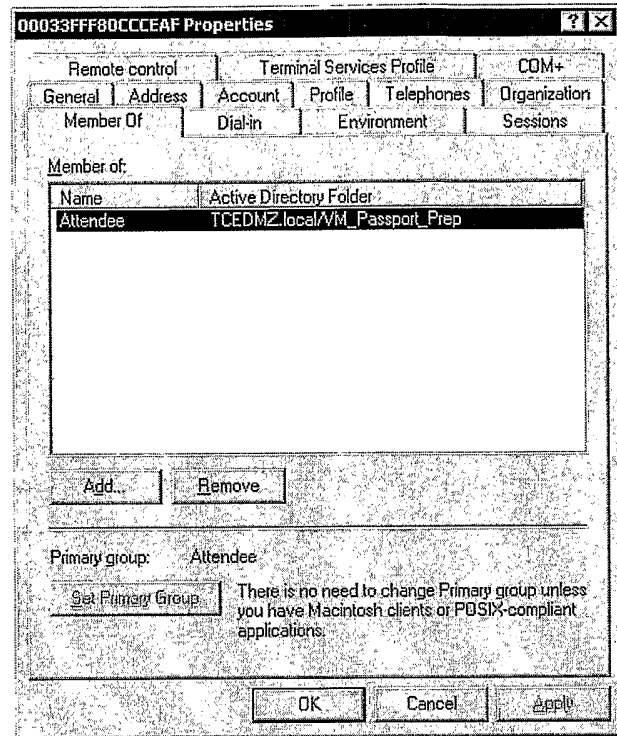
FIG. 5

The screenshot shows a "00033FFF80CCCEAF Properties" dialog box. The "General" tab is selected, and the user information is as follows:

- Member Of:** 00033FFF80CCCEAF
- First name:** duane
- Last name:** ansel
- Description:** duane ansel
- Office:** demo
- Telephone number:** demo
- Web page:** MyvPC.NET

At the bottom are "OK", "Cancel", and "Apply" buttons. A callout line labeled "602" points to the top-left corner of the dialog box, and another callout line labeled "604" points to the right side of the dialog box.

FIG. 6



604

FIG. 7

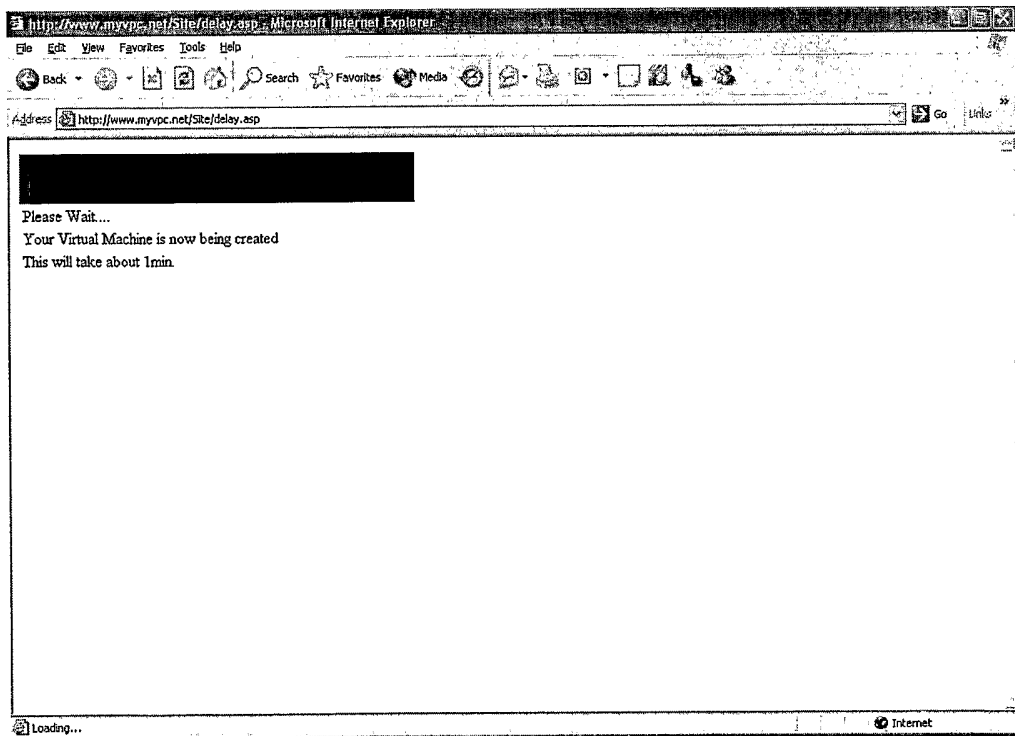


FIG. 8

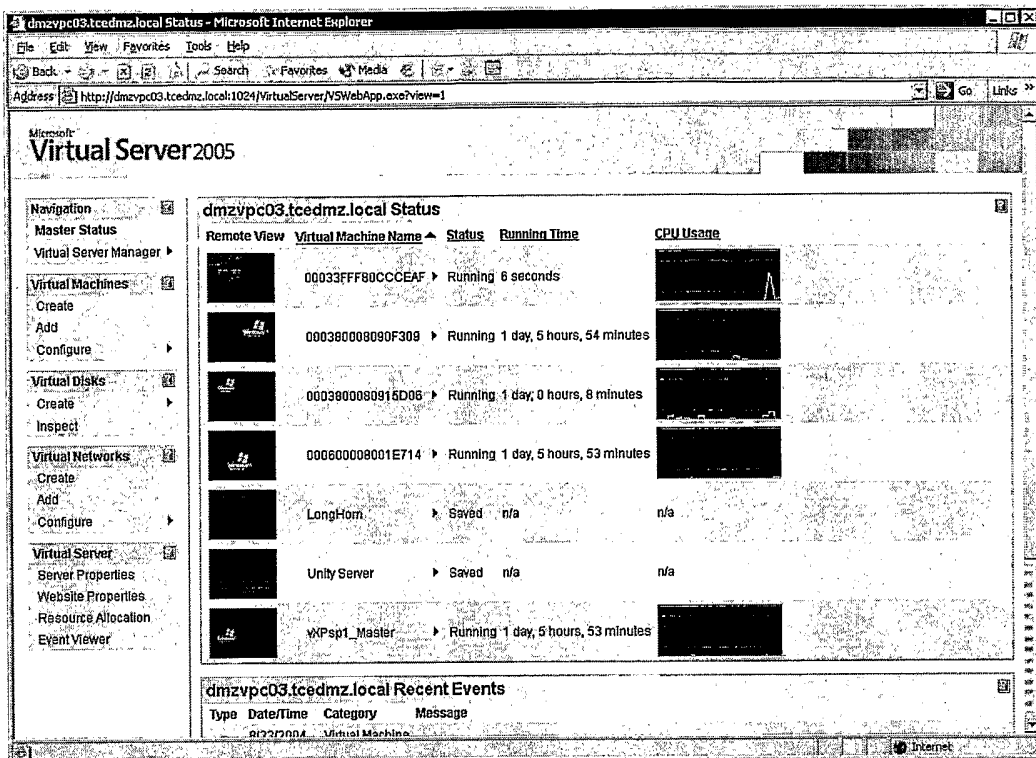


FIG. 9

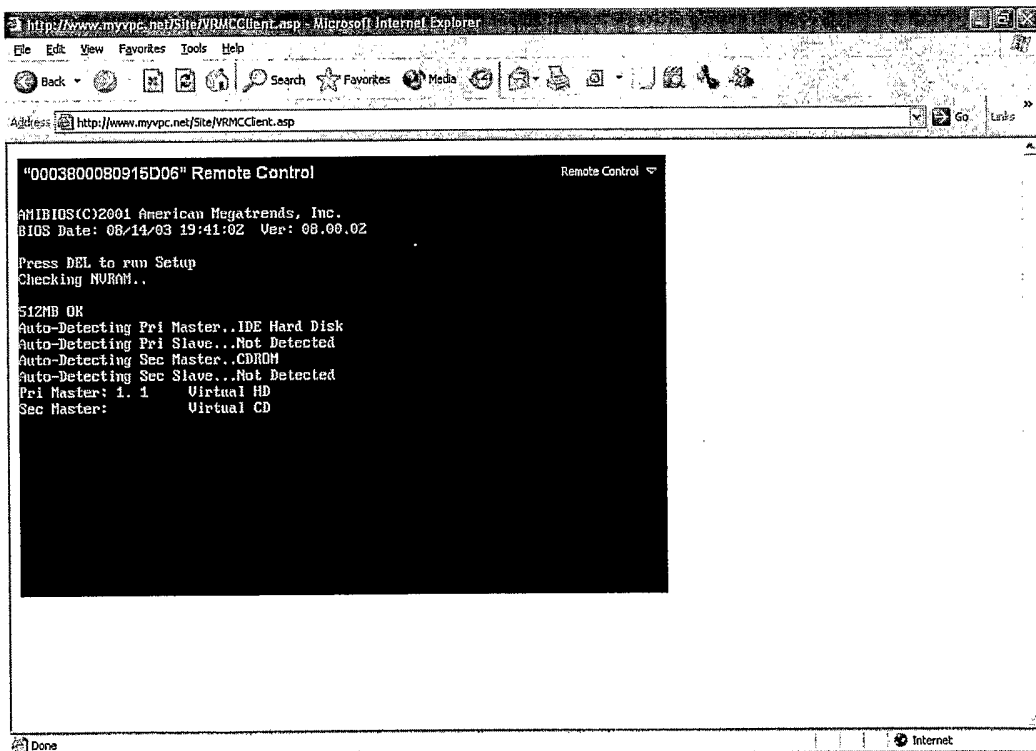


FIG. 10

9/13

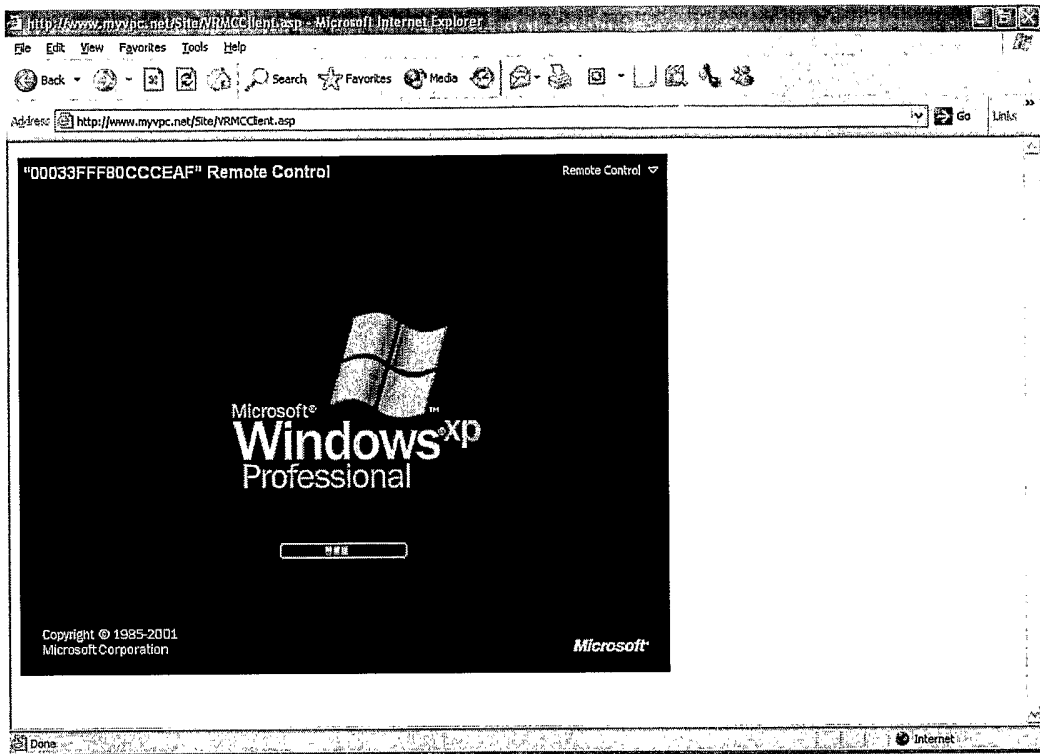


FIG. 11

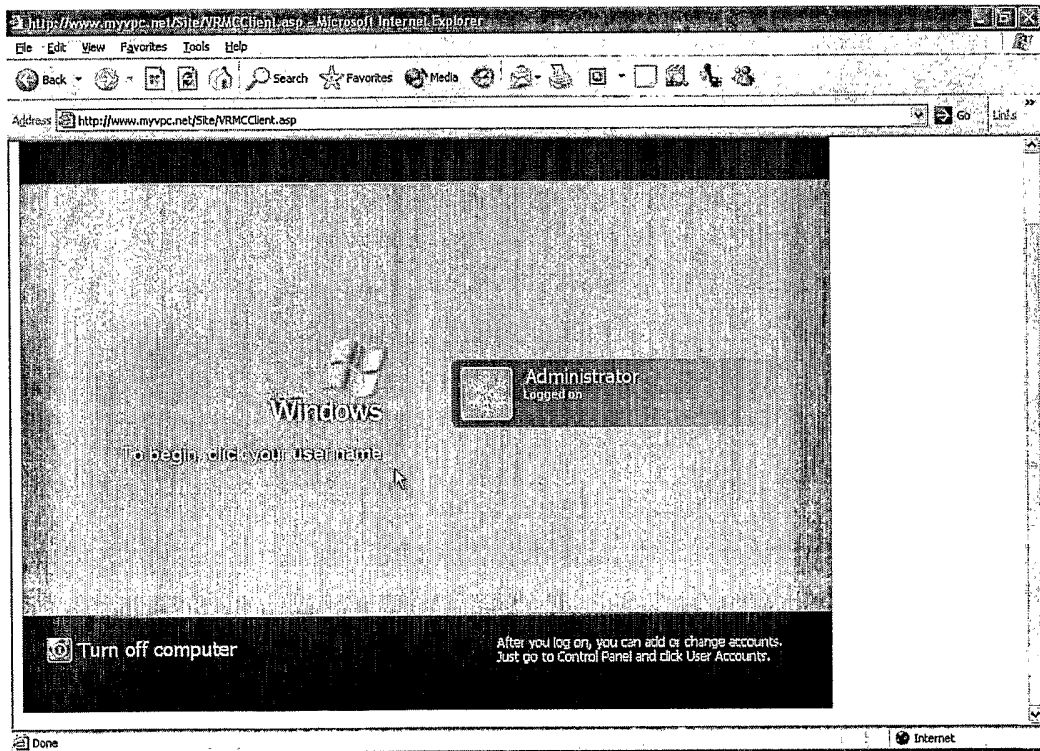


FIG. 12

10/13

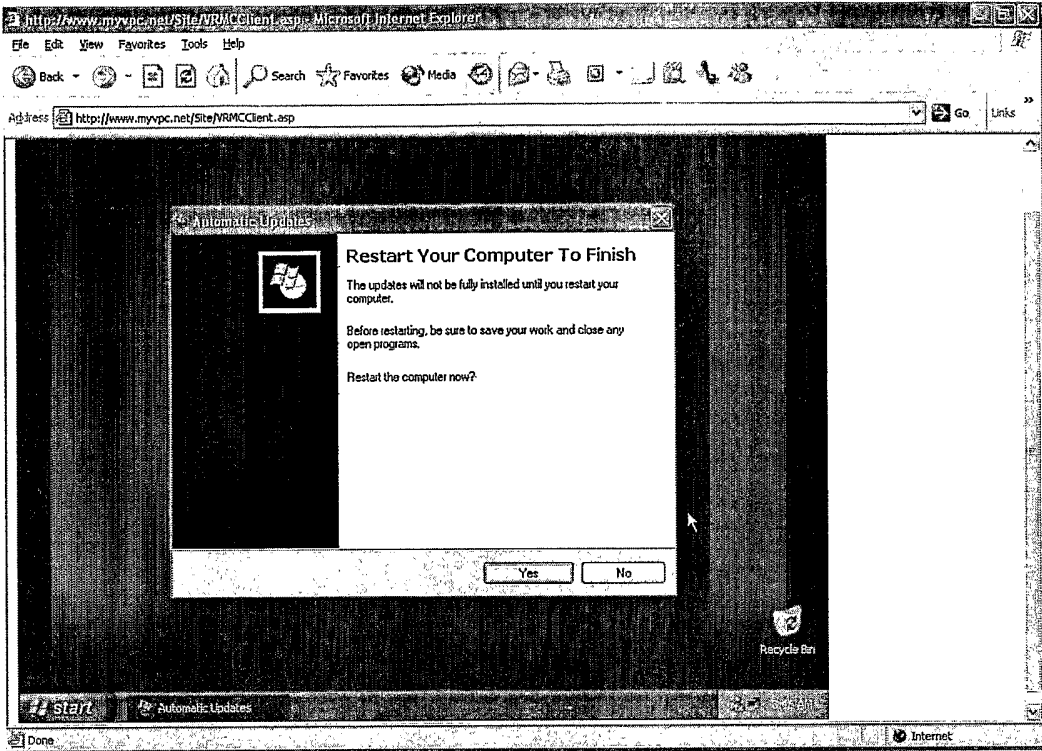


FIG. 13



FIG. 14

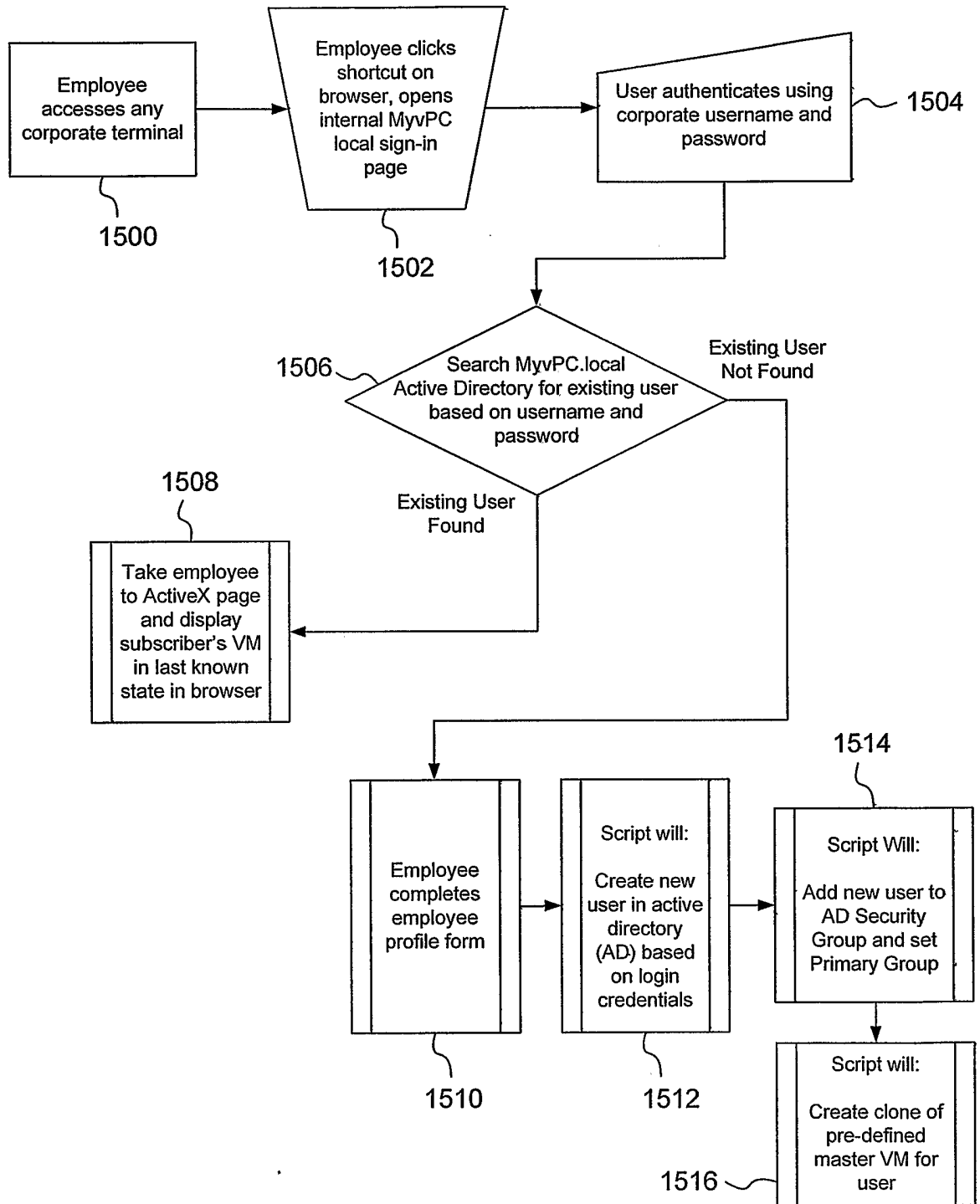


FIG. 15



12/13

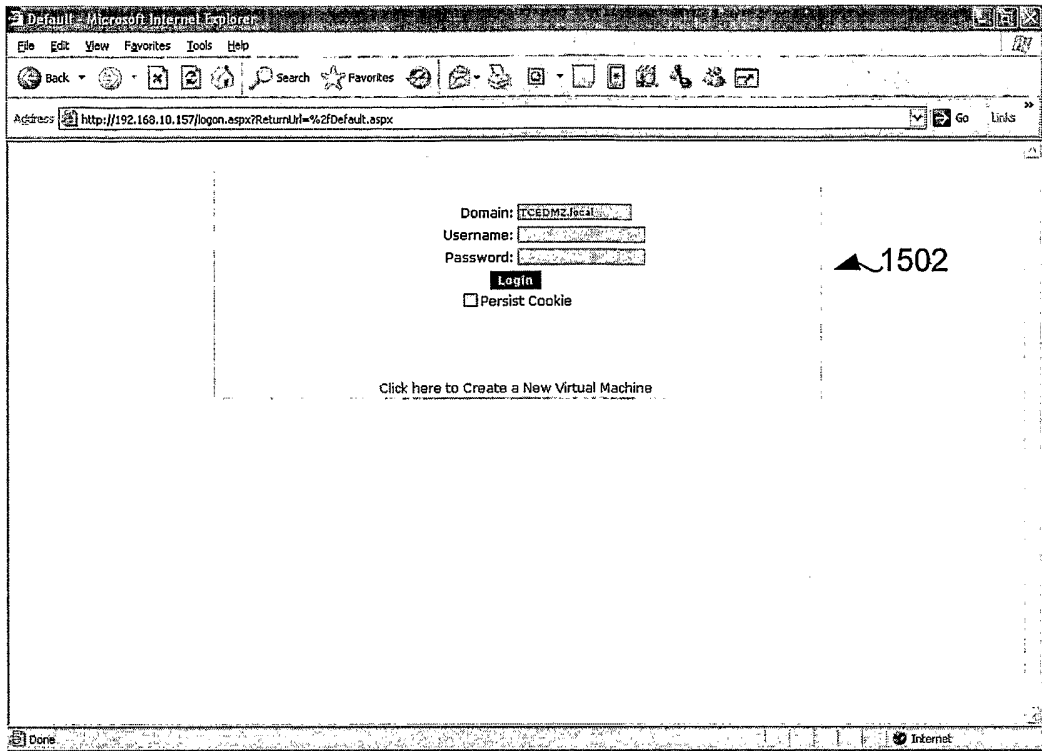


FIG. 16

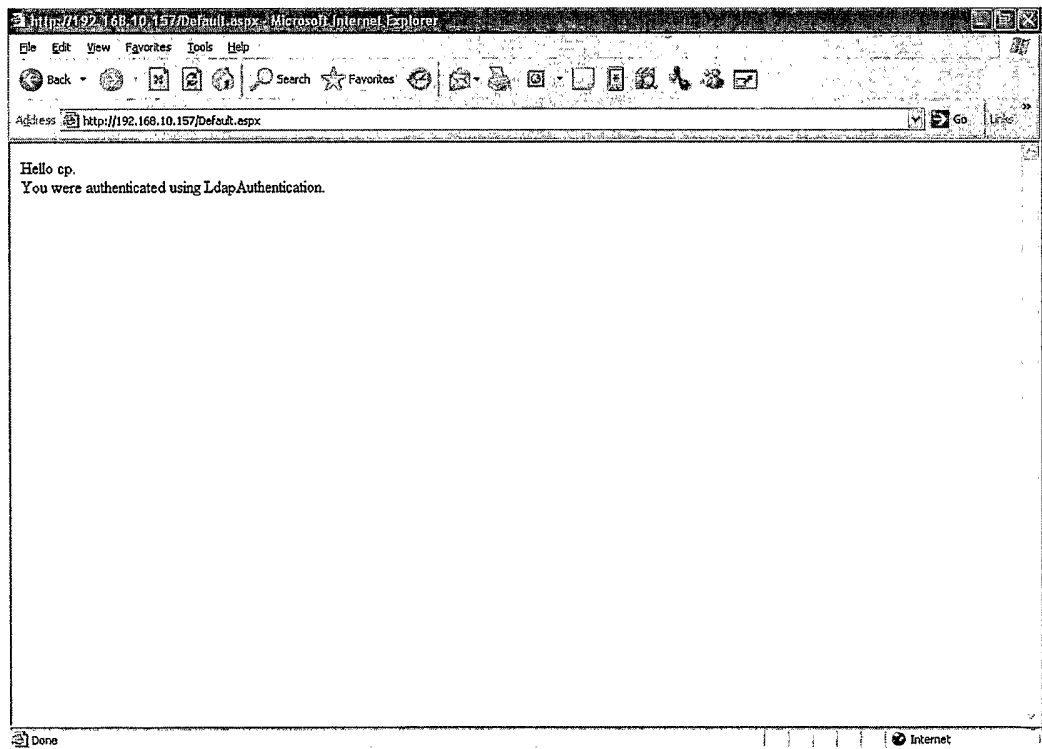


FIG. 17

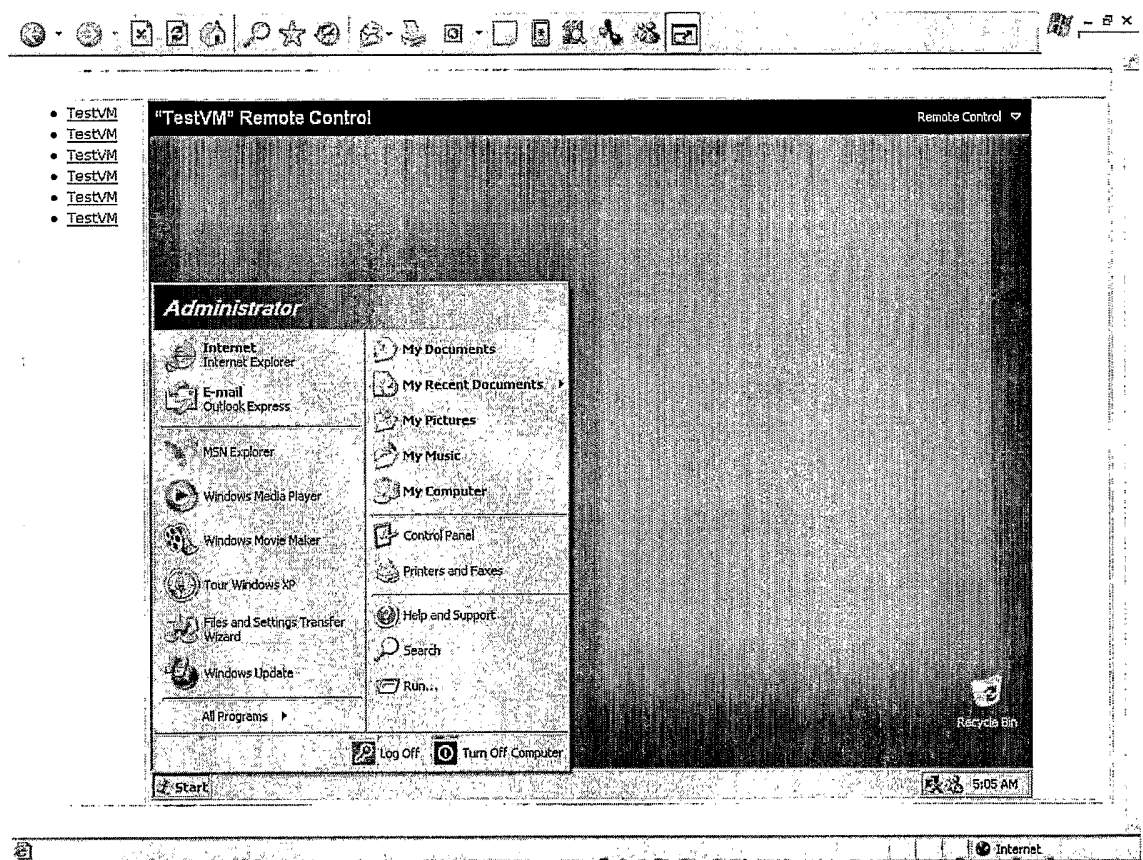


FIG. 18