

VERDE™ Administrator Guide

The logo for Virtual Bridges features a green arc above the text. The word "Virtual" is in a light green color, and "Bridges" is in a dark blue color. A trademark symbol (TM) is positioned at the top right of the word "Bridges".

Virtual Bridges™

Copyright © 2009 Virtual Bridges, Inc.
All Rights Reserved.

Table Of Contents

Overview.....	3
Scope.....	3
References and Requirements.....	3
Supported Host Platforms.....	3
Supported Guest Virtual Desktop Platforms.....	3
VERDE Architecture.....	4
Capacity Planning.....	5
Guest Application Profile.....	5
VERDE Server Virtual Desktop Density.....	5
Network Bandwidth for Connected Users.....	6
Storage Planning.....	6
Installation.....	7
Preparing for Installation.....	7
Solaris Server Preparation.....	7
Linux Server Preparation.....	7
Installing and Licensing the VERDE Software Package.....	8
Installing a “Gold Image” Desktop Virtual Machine.....	9
Installing a Windows 2000/XP Desktop Virtual Machine Image.....	9
Installing a Linux Guest Virtual Machine.....	10
Starting the Virtual Desktop.....	12
Provisioning a “Gold Image” Virtual Machine.....	13
Publishing a “Gold Image” Virtual Machine.....	13
Deploying a “Gold Image” Virtual Machine.....	13
Virtual Desktop Administration.....	16
Adjusting Virtual Machine Settings.....	16
Administering the Guest Virtual Desktop.....	17
“Online” Updates for Small Configurations.....	17
“Replica” Updates for Large Configurations.....	17
Connecting Remote Users to VERDE.....	19
Session Management.....	20
Listing Running Sessions.....	20
Shutting Down Sessions.....	20
Login Scripting and Automation.....	22
Login “Hooks”.....	22
Example Uses.....	22
General Hook Rules.....	22
Login Hook Assignment.....	23
Login Hook Environment Variables.....	23
Dumping Virtual Bridges Client Parameters.....	24
Clustering.....	26
Legal.....	27

Overview

Scope

This document covers administration topics for the Virtual Bridges VERDE product on a per-node basis.

References and Requirements

The Virtual Bridges VERDE (Virtual Enterprise Remote Desktop Environment) product builds on the Win4VDI and Win4Lin/Win4Solaris platforms to deliver Linux or Windows virtual desktops as cloud computing applications. Many topics will refer to the *Win4Lin Pro* or *Win4Solaris Pro Users Guide* for additional information or specific reference points. It is strongly recommended that the reader be familiar with the structure of the above mentioned Users Guide in order to facilitate learning the VERDE administration process.

This document assumes either a Linux or Sun Solaris server platform, and unless necessary, will not distinguish between the two. Basic Linux and/or Sun Solaris system administration skills are assumed for anyone using this document.

Additionally, many administration tasks will require the use of the `root` user account on the server. This document assumes that the reader either has `root` (or `sudo`) access, or can easily acquire it in order to perform such tasks. Please note that the document will not distinguish between `sudo`, `su`, or `login` for `root` access – it will simply be referred to as `root` access. The reader should adjust commands accordingly for what is appropriate on his or her system.

Finally, the document assumes the reader either has access to the server's console, or can reach a shell on the server remotely using protocols such as `telnet` or `ssh`. Many tasks will require the use of an `X11` server on the display terminal, and it is up to the reader to properly configure local/remote `X11` access via the console, `ssh`, `telnet`, or other methods. Note that some methods may require the explicit setting and export of the `DISPLAY` environment variable. Also, unless otherwise noted, commands in this manual assume a Bourne or POSIX shell (e.g. `sh`, or `bash`).

Supported Host Platforms

- 32-bit or 64-bit x86 Intel or AMD processor – VT or AMD-V capabilities strongly recommended
- Most 32-bit or 64-bit x86 Linux distributions with 2.6 kernel – KVM-capable kernel version 2.6.20 or newer strongly recommended for best performance and compatibility
- 32 or 64-bit Sun Solaris 10 x86

Supported Guest Virtual Desktop Platforms

- 32-bit Microsoft Windows 2000 Professional or XP Professional, any service pack
- 32-bit (i386) Ubuntu 8.04 LTS Desktop Linux
- 32-bit (i386) Novell SuSE Linux Enterprise Desktop 10, any service pack
- 32-bit (i386) or 64-bit (x86_64) Red Hat Enterprise Linux 5 Workstation, any updates
- 32-bit (i386) or 64-bit (x86_64) CentOS 5, any updates

VERDE Architecture

VERDE is an all-in-one VDI solution that includes hypervisor, virtual desktop manager, and connection broker. The components are tightly-integrated and designed for virtual desktop use. Each VERDE server runs its own connection broker, which authenticates users and then uses the virtual desktop manager to either instantiate new virtual machines or connect users to existing virtual machines.

In the VERDE model, virtual machines are “stateless”. This means they do not need to be powered on or off, or created ahead of time. They are created on demand based on what “template” or “gold image” is provisioned. Each user in the system is assigned one or more dynamic desktops based on said “templates”, and can also optionally host his or her own self-managed virtual desktop if needed.

Authentication happens at the server level, not the virtual machine level. This means that each user must have a Linux/Solaris user ID and a home directory. VERDE uses PAM to authenticate users, so any existing directory/authentication method will work as long as the Linux/Solaris server hosting VERDE is configured to communicate with that service via PAM. Virtual desktops themselves usually run in an “auto login” configuration, in order to avoid redundant logins and preserve single sign-on capabilities. Since the virtual desktops are authenticated and authorized at the server level, security policies inside the virtual machines are not usually relevant. In fact, virtual Windows desktops usually run as the “Administrator” account, which gives maximum application compatibility. Since the user itself is already authorized with Linux/Solaris security and permissions, and the desktop is dynamic, such rights inside the virtual machine do not pose a problem. For Linux virtual desktops, auto-login with a common user ID from the GNOME/KDE Desktop Manager is highly recommended. Even though all users provisioned from a “template” or “gold image” seemingly log in as the same ID inside their virtual machines, they are still running discreetly and authorized with the host system's security, and their files are kept in their underlying host home directories with appropriate permissions.

A typical VERDE server configuration will have the following qualities:

- 1 or more “template” or “gold image” virtual desktop installations stored under Linux/Solaris user account(s)
- 1 or more dynamic desktops provisioned from that “template” or “gold image” to Linux/Solaris user account(s)
- user documents and personal settings for provisioned dynamic desktop instances stored under the respective Linux/Solaris user's home directory
- Each virtual desktop user will have a unique Linux/Solaris user ID to log into the server with

VERDE desktop virtual machines run as Linux/Solaris processes, authorized as the user who logged into the connection broker. To the host system, they appear as regular applications, and obey all process limits and restrictions as set by the system administrator. This includes `ulimit`, `nice`, and `quota` settings.

Capacity Planning

VERDE server capacity must be planned for peak concurrent virtual desktop usage. It is important that the server have enough resources to accommodate this, otherwise performance and virtual desktop usability will deteriorate.

Please note that concurrent usage is not the same as connected usage. Even if users are not connected to the server, they may still have a virtual desktop environment consuming resources, and capacity planning must account for this. Any virtual desktop environment running on the server, whether a user is connected to it or not, counts toward concurrent usage.

Guest Application Profile

The actual applications (and use case for them) running inside guest virtual machines will play a major factor in determining the virtual desktop density of a given VERDE server. For example, normal office/business applications scale much better than high-fidelity multimedia programs. For the purpose of illustration, the following figures assume a normal office/business application workload.

VERDE Server Virtual Desktop Density

When determining the virtual desktop density possible on particular hardware for a VERDE server, the following information is needed:

1. Number of CPU sockets (C)
2. Number of CPU cores per socket (c)
3. Total system RAM (M)
4. Guest virtual machine RAM assignment (m)

The number of concurrent sessions that can fit in memory on a particular VERDE server (T_1), either connected or disconnected, can be calculated as follows:

$$T_1 = M(.75) / m$$

For example, on a system with 16GB of total RAM, where each guest session would require a 512MB RAM assignment, the number of concurrent sessions that can fit in memory without degrading server performance is:

$$T_1 = 16384(.75) / 512$$

(24)

While more sessions may fit, anything above this number would reduce overall performance on the VERDE server since memory must either be over-committed (backed by swap space), or host operating system buffer cache must be reduced.

Additionally, a common *guideline* metric for calculating the number of concurrent sessions that can be executed on a given CPU core is 5. Note that depending on application profile, this number may be as low as 2 and as high as 10 (or more). For the purpose of planning moderate application load (e.g. office applications), it is safe to use the 5 concurrent sessions per core metric. To calculate the maximum number of concurrent sessions that can be executed on a given VERDE Server without degrading session performance is:

$$T_2 = 5(C(c))$$

For example, on a system with 2 sockets and 4 cores per socket:

$$T_2 = 5(2(4))$$

(40)

The actual maximum number of concurrent sessions that will both fit in memory and execute with expected performance on a given VERDE server (T) is the lesser of the values T_1 and T_2 . In the examples above, this number would be $T = 24$. In order to support the $T = 40$ concurrent users that the CPU cores are capable of, the server would need at least $M = 27GB$ of RAM.

Network Bandwidth for Connected Users

The per-session remote display and device performance will depend heavily on the amount of total network bandwidth available. Generally speaking, the higher the switched bandwidth, the faster and more responsive the end-user sessions will be.

VERDE sessions require a minimum of 512Kbps bandwidth in order to produce a good desktop user experience.

In cases where not all users will be connected at the same time, the actual total network bandwidth may be lower without sacrificing session responsiveness since only a portion of users will be transmitting at any given time.

Storage Planning

VERDE sessions use a “copy-on-write” mechanism to minimize the actual persistent storage needed per user of a given template configuration. For example, if a template guest installation consumes 32 gigabytes of storage, each deployed user running an instance of it may only need less than 1 gigabyte of persistent storage space.

The copy-on-write information itself does however require transient storage. Transient storage requirements vary greatly depending on applications, use, and even runtime length of sessions. However, a conservative estimate is to use 20% of the template image size for each deployed instance. For example, if a template guest installation consumes 32 gigabytes of storage, the transient storage size for each server should be 6.4 gigabytes per user. For 50 concurrent users, this level, assuming the above example, would be 320 gigabytes.

Installation

Installing VERDE is a multi-step process:

1. Preparing for installation
2. Installing and licensing the VERDE software package
3. Installing a “gold image” desktop virtual machine
4. Provisioning the “gold image” as dynamic instances for user(s) or group(s)

Preparing for Installation

Solaris Server Preparation

On Solaris, it is important to determine the kernel architecture before installing the respective VERDE (Win4SolPro) package. The VERDE package architecture must match exactly the Solaris kernel architecture, either `i386` or `amd64`. To determine this, run the following command:

```
isainfo -k
```

Next, you must ensure that the system has at least twice as much swap space as physical memory. For example, if the system has 4GB of physical RAM, it should have 8GB of swap space. Note that failure to observe this rule may result in processing errors in your virtual machines.

Finally, if you plan on using a physical CDROM or DVD disc to install the “gold image” guest virtual machine, make sure that the device node corresponding to the drive has read access for the user who will host this virtual machine. On Solaris 10, the default CDROM device is `/dev/sr0`.

Linux Server Preparation

On Linux, you must install the “kernel build toolchain”, which typically includes the kernel headers as well as the C compiler (`gcc`) used to compile the kernel. Many Linux distributions do not install these packages by default, so you must do this manually before attempting to install the VERDE package (Win4LinPro). Please refer to the section *Preparing for Installation* under *Installing Win4Lin Pro* in the *Win4Lin Pro/Win4Solaris Pro Users Guide* for additional details. Various examples are provided for popular Linux server distributions. If you need additional help, please contact your Linux distribution vendor in order to complete this step (e.g. Red Hat or Novell).

If you are using a VT or SVM-capable hardware, you should ensure the kernel drivers for KVM are loaded. To load these drivers, execute one of the following commands, as root, depending on whether you have an AMD or Intel processor:

```
/sbin/modprobe kvm_intel
```

```
/sbin/modprobe kvm_amd
```

Note that some systems disable hardware virtualization in the BIOS by default. This will cause the above modules to fail to load, and the error message found in the kernel's `dmesg` output will indicate that support is disabled in the BIOS. If this is the case, reboot the system, enter the BIOS configuration screen, and enable Hardware Virtualization support. Please refer to the documentation provided by your hardware vendor for more details on how to do this. If the `dmesg` command reports a different error, then your system is likely not equipped with VT or SVM hardware virtualization support.

If your system is VT capable but your Linux distribution does not ship a KVM driver, you can download and install this yourself from:

<http://sourceforge.net/projects/kvm/>

Please note that Virtual Bridges, Inc. does not produce nor maintain the KVM drivers. This is an Open Source project that is an extension of the Linux kernel.

Once you load the KVM drivers, you must ensure that the `/dev/kvm` device node has read/write access for the users and/or groups you wish to deploy virtual machines for. For the most flexible operation, it is recommended that you set the permissions on the device to `0666`, since in general terms every user who logs into the system will require KVM support. For example, as root:

```
chmod 666 /dev/kvm
```

You must then determine the kernel architecture and make sure the VERDE package (Win4LinPro) matches. You can use the following command to determine this:

```
uname -m
```

If the command returns `i386`, `i486`, `i586`, `i686`, or `athlon`, you should choose the `i386` VERDE package to install. If the command returns `x86_64` or `amd64`, you should choose the `x86_64` or `amd64` VERDE package to install. Note that in packaging terms, `amd64` is compatible with both AMD and Intel 64-bit x86 processors.

Installing and Licensing the VERDE Software Package

Once you have determined the appropriate package to install, use your host operating system's default package manager to install it into the system. On Solaris you will need to use the `pkgadd` command, while on Linux you will use either the `rpm` or `dpkg` command depending on whether your system is RPM-based or Debian package-based. For further installation instructions, including package removal and package update, please see *Installing Win4Lin Pro/Win4Solaris Pro* in the *Win4Lin Pro/Win4Solaris Pro Users Guide*.

To license the VERDE software, you must create a license file that is owned by root. On Solaris, this file is called `/var/lib/VBICw4s/license.lic`, while on Linux, the file is called `/var/lib/win4linpro/license.lic`. The format of the file is as follows:

```
LICENSE_CODE=XXXXXX
CUSTOMER_NAME="CCCCCC"
```

Replace `XXXXXX` with the license code you received at the time of purchase or as part of an evaluation package from Virtual Bridges, Inc. Be sure to enter it (or copy and paste it) exactly as it appears in the official correspondence. Replace `CCCCCC` with your name, or your organization's name. This text will appear on the splash screen of the virtual machine loader and will be visible to all users. Note that you should enclose this name in quotes, especially if there are blank spaces in it. You can verify that the license is applied correctly by running the `win4-licinfo` command:

Solaris:

```
/opt/VBICw4s/bin/win4-licinfo
```

Linux:

```
/usr/lib/win4linpro/bin/win4-licinfo
```

The command should report the license status as "Product is licensed" if you created the license file correctly. It is also recommended that you give this file permissions of `0600`.

VERDE server software is licensed with a base concurrent user count of either 10 or 25. If you received one or

more “bump” license codes from Virtual Bridges, Inc., you must also add them to the license file. These “bump” codes increase the concurrent session limit on your VERDE server from the 10 or 25 baseline.

For information on applying bump licenses, please see the section *Applying Additional “Bump” Licenses* under the *Installing Win4Solaris Pro/Win4Lin Pro VDI* in the *Win4Lin Pro/Win4Solaris Pro Users Guide*.

Installing a “Gold Image” Desktop Virtual Machine

Once you have selected the Linux/Solaris user account which will host the “gold image” desktop virtual machine, log in as that user. Make sure you have the ability to run X11 applications under that login (e.g. set the `DISPLAY` variable accordingly, or use the account on the server console itself.) Next, use either the `installwin4` command or the `win4-install-linux` command to install a Windows 2000/XP or Linux virtual machine, respectively.

IMPORTANT NOTE: Never attempt to install or start a virtual desktop as the `root` user on your system. Virtual desktops may only be installed for and used by non-`root` users!

Installing a Windows 2000/XP Desktop Virtual Machine Image

Usage: `installwin4` [*<options...>*] [*<config-name>*]

<config-name> is the configuration name to install (default: `win4`); a subfolder of the user's home directory will be created with this name

Option	Description
<code>-h</code>	Display help page
<code>-r</code>	Overwrite existing installation if present
<code>-y</code>	Do not prompt to overwrite or install
<code>-i</code>	Install desktop icon for newly installed session
<code>-m <size></code>	Amount of RAM for the guest, in megabytes (default: 128)
<code>-d <size></code>	Maximum guest disk size, in gigabytes (default: 8)
<code>-c <path></code>	Name of CD/DVD device or .iso image file (default: CD device is guessed)
<code>-u <path></code>	Name of CD/DVD device or .iso image file containing prior version of Windows media, as required for Windows upgrade CDs; by default this is not used, but you must set it if you are installing from a Windows Upgrade CD
<code>-k <key></code>	Windows Product Key to pass to Windows installer (default: installer prompts for the key)
<code>-t <title></code>	Virtual Machine Window title (default: <i><config-name></i>)

Example 1: install a Windows 2000/XP virtual desktop, with the bootable Windows CDROM in the default CD/DVD device on the system, under the default configuration name `win4`:

```
installwin4
```

Example 2: install a Windows 2000/XP virtual desktop, from an ISO 9660 image of a bootable Windows CDROM in your home directory, under the default configuration name `win4`:

```
installwin4 -c $HOME/winxppro.iso
```

Example 3: install a Windows 2000/XP virtual desktop, from an ISO 9660 image of a bootable Windows

CDROM in your home directory, with 16GB virtual “C:” disk size, under the configuration name `winxp`:

```
installwin4 -c $HOME/winxp.iso -d 16 winxp
```

In every case, the installation is completely automatic. If your Windows media requires it, the Windows installer will prompt you for the Windows Product Key – be sure to enter it exactly as it appears on your Microsoft documentation when prompted. This may also be a volume license key.

The virtual machine console will appear on your regular X11 desktop as a standard window, and the virtual machine session will end automatically when the Windows installer finishes. This may take up to 1 hour depending on the speed and load of your system.

While the guest virtual machine RAM assignment may be changed after installation, the virtual disk image size may not. It is important that you assign an appropriate amount of disk space with the `-d` option if you believe 8GB will not be enough. In most cases 8GB is enough to hold the Windows system files as well as numerous Windows applications. User Documents and Settings are stored on a separate virtual disk, and users may also store documents (by default) on their underlying Linux/Solaris home directory.

Note that the virtual disk image size is a maximum setting that the image can grow to. The disk space is not allocated in advance of the guest system requesting it. A typical Windows installation without applications installed will consume about 1.5GB of disk space.

Installing a Linux Guest Virtual Machine

Usage: `win4-install-linux` [*<options...>*] [*<config-name>*]

<config-name> is the configuration name to install (default: `win4`); a subfolder of the user's home directory will be created with this name

Option	Description
<code>-h</code>	Display help page
<code>-s</code>	Use Software Virtualization only (disable KVM use if available)
<code>-S <num></code>	Enable SMP (requires KVM)... <i><num></i> is number of virtual CPUs, 2-8
<code>-64</code>	Use 64-bit guest CPU (requires KVM)
<code>-u</code>	Use user-mode (safe) virtualization (default) – requires <code>-s</code> or non-KVM system
<code>-k</code>	Use kernel-mode virtualization – requires <code>-s</code> or non-KVM system
<code>-l</code>	Use legacy graphics mode (required for older Linux, such as SLED/OpenSuSE 10)
<code>-r</code>	Overwrite existing installation if present
<code>-y</code>	Do not prompt to overwrite or install
<code>-i</code>	Install desktop icon for newly installed session
<code>-m</code>	Amount of RAM for guest, in megabytes (default: 256)
<code>-d</code>	Maximum guest system disk size, in gigabytes (default: 12)
<code>-H</code>	Home virtual disk size, in megabytes; valid values: 1024, 2048, 4096, 8192, 16384 only (default: 2048)
<code>-c <path></code>	Name of CD/DVD device or .iso image file (default: CD device is guessed)
<code>-t <title></code>	Window title (default: <i><config-name></i>)

Example 1: install a Linux virtual desktop, with the bootable Linux CDROM/DVD in the default CD/DVD device on the system, under the default configuration name `win4`:

```
win4-install-linux -u
```

Example 2: install a Linux virtual desktop, from an ISO 9660 image of a bootable Linux CDROM/DVD in your home directory, under the default configuration name `win4`:

```
win4-install-linux -u -c $HOME/linux.iso
```

Example 3: install a Linux virtual desktop, from an ISO 9660 image of a bootable Linux CDROM/DVD in your home directory, with 16GB virtual system disk size, under the configuration name `linux`:

```
win4-install-linux -u -c $HOME/linux.iso -d 16 linux
```

Note that in all cases, the `-u` flag is specified. Most Linux distributions will fail to install and/or run properly without this flag. If your system is KVM-capable, and you did not specify `-s`, the flag is ignored. However, if your system is not KVM-capable, or you specified `-s`, `-u` is generally required. Generally speaking, you should never explicitly specify `-s`.

64-bit guest support is limited only to certain Linux distributions. See the section *Supported Guest Virtual Desktop Platforms* under *Overview* for information on which 64-bit distributions are supported.

While the guest virtual machine RAM assignment may be changed after installation, the virtual disk image size may not. It is important that you assign an appropriate amount of disk space with the `-d` option if you believe 12GB will not be enough. In most cases 12GB is enough to hold the Linux system files as well as numerous Linux applications. User files are stored on a separate virtual disk (the home partition), and users may also store files (by default) on their underlying host Linux/Solaris home directory.

The virtual home partition size can also not be adjusted after installation, so if you believe 2GB is not enough, specify a larger size with the `-H` option. 2GB is generally sufficient for most virtual desktop uses, especially if users are allowed to store files in their underlying host Linux/Solaris home directory.

Note that the virtual disk image size is a maximum setting that the image can grow to. The disk space is not allocated in advance of the guest system requesting it. A typical Linux guest installation without applications installed will consume about 3GB of disk space.

Linux guest installation is not completely automatic and requires you to interact with the installer of the particular distribution of your choice. Here are some guidelines for installing Linux in the guest virtual machine:

1. The virtual machine has 2 virtual disks – either `/dev/hda` and `/dev/hdb`, or `/dev/sda` and `/dev/sdb`, depending on your Linux distribution. You should not install anything on nor initialize `/dev/hdb` or `/dev/sdb` – if prompted to format it or initialize its partition table, always decline to do this. The only disk you should write on is `/dev/hda` or `/dev/sda`.
Please note: initializing or writing to `/dev/hdb` or `/dev/sdb` with the Linux installer will result in a broken virtual machine installation.
2. It is recommended that you always accept the default partition layout for the first virtual disk. The only time you should change this is if the installer defaults to putting partitions on the second virtual disk (see #1 above).
3. Always accept the default values for all virtual hardware configuration, including display, networking, and audio. VERDE will control the virtual hardware automatically after installation and changing any settings during installation will interfere with this.
4. Networking in the virtual machine must be configured to use DHCP. This is the default for all supported guest Linux types, and should not be changed.

5. If prompted for a non-root user account to configure in the installer, you should choose something very generic. Remember that authentication in the virtual machine is not important in the VERDE model, since you are already authenticated at the server level. You should also choose to use local authentication if prompted, unless there is an organizational requirement to authenticate separately inside the virtual machine, resulting in redundant logins. In most cases it is recommended that you choose local authentication and name the non-root user “verde”. You should later set up automatic login in the virtual desktop’s login screen configuration (GDM) in order to achieve true single sign-on.
6. If you are installing an older Linux distribution, such as SuSE Linux Enterprise Desktop 10 or OpenSuSE 10, you must specify the `-l` flag to `win4-install-linux`. This will select “legacy” graphics support for the virtual machine, and will limit the virtual desktop resolutions to 800x600 and 1024x768 regardless of what the actual physical desktop resolution of the client is. Note that failing to specify `-l` for these older distributions will result in failure to install and/or crashes!

Once the Linux guest installation completes, you will either have the choice of rebooting the virtual machine, or simply logging into the virtual desktop, depending on what distribution you are using. If prompted to reboot (e.g. Ubuntu), do it. When you are presented with the login screen, login as root (if using Red Hat/CentOS or SuSE/Novell), or as the user you configured during installation (if using Ubuntu).

Once logged into the virtual desktop, do not adjust any desktop settings. Immediately double click on the desktop CDROM icon that reads Win4Lin Pro or Win4Solaris Pro, and then double click on the appropriate “Finish” icon for your distribution. On Ubuntu you will be prompted to enter the password again. The “Finish” script will run in a terminal window, and automatically shut down the virtual machine when complete. Once the virtual machine window disappears, the guest image is set up correctly.

For convenience, there are several Linux guest-specific “Quick-start” guides included in the VERDE documentation suite. It is recommended that these guides be used at least for the first Linux guest installation you do, so that you can properly follow all the steps required.

Starting the Virtual Desktop

To start the virtual machine you just installed, use the `win4` command as the non-root user:

```
win4 [<config-name>]
```

If you used the default configuration name of `win4`, you do not need to specify it on the command line.

Example 1: start the default `win4` configuration for this non-root user:

```
win4
```

Example 2: start the `linux` configuration for this non-root user:

```
win4 linux
```

The first time you start the virtual machine, you should perform some manual configuration to ensure that it is as effective a “gold image” as possible:

1. (Windows) activate the XP installation if required
2. (Windows) disable System Restore and Automatic Updates
3. (Linux) configure GDM to automatically log in the non-root user you created during installation. If you did not create a non-root user, do it first, and then configure GDM to log it in automatically in order to achieve “single sign-on”. *Typical GNOME desktops use the option Login Window under the Administration category of the System menu to control GDM.*

Once you complete this configuration, shut down the guest session (using the `Start->Shutdown` option

for Windows, or the appropriate equivalent for Linux).

Provisioning a “Gold Image” Virtual Machine

After installing, starting, and shutting down the “gold image” virtual desktop as described in the previous section, you may now provision dynamic instances of it to one or more users or groups.

The VERDE terms for this mechanism are “publish” and “deploy”. You publish a virtual desktop, and then you deploy it to the users or groups that should be able to start dynamic instances of it. These dynamic instances present a transient “copy-on-write” system image (C: drive for Windows guests, or / partition for Linux guests), but with persistent user settings and documents. Any changes made in the published “gold image” automatically propagate to dynamic instances the next time those users start their virtual desktops. Once a “gold image” is published, there is no need to publish it again if you make changes to it in the future (e.g. install software or make system settings adjustments). The mechanism is completely automatic and transparent to users.

Even though the `root` user must not host virtual desktops, you must log in as `root` (or use `sudo`) to publish and deploy virtual desktops. The commands explained below assume you are running them with `root` privileges.

Publishing a “Gold Image” Virtual Machine

```
win4-publish-session [-U] <username|uid> [<config>]
```

Option	Description
<code>-U</code>	indicates that the virtual desktop is being unpublished, rather than published; this reverses the effect of publishing a “gold image”, which essentially disables the ability to deploy it to other users as dynamic instances
<code>username uid</code>	Linux/Solaris user name or numeric user ID of the user for whom you installed the “gold image” virtual machine
<code>config</code>	optional configuration name (folder you installed virtual machine in) to publish; if you do not specify it, the default is <code>win4</code>

Example 1: publish the default “gold image” virtual desktop (`win4`) for the user `gold`:

```
win4-publish-session gold
```

Example 2: publish the “gold image” virtual desktop named `maindesktop` for the user `gold`:

```
win4-publish-session gold maindesktop
```

Example 3 : unpublish the default “gold image” virtual desktop (`win4`) for the user `gold`:

```
win4-publish-session -U gold
```

Deploying a “Gold Image” Virtual Machine

To deploy a published “gold image” virtual desktop to one or more users or groups of users, use the `win4-deploy-published` command. Note that this command has several forms:

```
win4-deploy-published <published-user> [<config>] -u <users...>
win4-deploy-published <published-user> [<config>] -U <user> [<config>]
```

```
win4-deploy-published <published-user> [<config>] -g <groups...>
win4-deploy-published -x <user> [<config>]
```

Option	Description
<i>published-user</i>	Linux/Solaris user name or numeric user ID of the user for whom the published virtual desktop is installed
<i>config</i>	optional configuration name; if not specified for the published virtual desktop, the default is <code>win4</code> ; if not specified for the deployed virtual desktop (when using <code>-U</code>), the default is either <code>win4</code> if it is not installed for that target user, or <code>win4-n</code> , where <i>n</i> is the first number (starting at 1) of a configuration name that does not exist for that user; if not specified when using <code>-x</code> , the default is <code>win4</code>
<code>-u</code>	specifies that the following <i>users</i> lists 1 or more Linux/Solaris user name(s) or numeric user ID(s), separated by white space, for whom to deploy the published virtual desktop to; each user will receive the dynamic instance of the published virtual desktop in the configuration <code>win4</code> if not already installed for that user, or <code>win4-n</code> , where <i>n</i> is the first number (starting at 1) of a configuration name that does not exist for each respective user
<code>-U</code>	specifies that the published virtual desktop is to be deployed to a single <i>user</i> (Linux/Solaris user name or numeric user ID), as an optional <i>config</i> name; if <i>config</i> is not specified, the default is <code>win4</code> if not already installed for that user, or <code>win4-n</code> , where <i>n</i> is the first number (starting at 1) of a configuration name that does not exist for that respective user
<code>-g</code>	specifies that the following <i>groups</i> lists 1 or more Linux/Solaris group name(s) or numeric group ID(s), separated by white space, for whom to deploy the published virtual desktop to; each user in the group(s) (except the user for whom the published virtual desktop is installed, if in one of the groups) will receive the dynamic instance of the published virtual desktop in the configuration <code>win4</code> if not already installed for that user, or <code>win4-n</code> , where <i>n</i> is the first number (starting at 1) of a configuration name that does not exist for each respective user
<code>-x</code>	specifies that the following <i>user</i> (Linux/Solaris user name or numeric user ID) should have its dynamic desktop removed; if <i>config</i> is not specified, the default is <code>win4</code>

Example 1: deploy published virtual desktop `win4` from user `gold` as a dynamic desktop for user `user1` into the default folder (`win4`, or `win4-n`):

```
win4-deploy-published gold -u user1
```

Example 2: deploy published virtual desktop `mainimage` from user `gold` as a dynamic desktop for `user1` into the default folder (`win4`, or `win4-n`):

```
win4-deploy-published gold mainimage -u user1
```

Example 3: deploy published virtual desktop `mainimage` from user `gold` as a dynamic desktop for users `user1` and `user2` into their default folders (`win4`, or `win4-n`):

```
win4-deploy-published gold mainimage -u user1 user2
```

Example 4: deploy published virtual desktop `mainimage` from user `gold` as a dynamic desktop for all

users in the groups `users` and `testers`:

```
win4-deploy-published gold mainimage -g users testers
```

Example 5: deploy published virtual desktop `mainimage` from user `gold` as a dynamic desktop named `maininst` for the user `user1`:

```
win4-deploy-published gold mainimage -U user1 maininst
```

Example 6: remove a deployed virtual desktop image named `win4` from the user `user1`:

```
win4-deploy-published -x user1
```

Example 7: remove a deployed virtual desktop image named `maininst` from the user `user1`:

```
win4-deploy-published -x user1 maininst
```

Once you deploy published virtual desktops to users as dynamic desktops, each virtual desktop will inherit the published desktop's system image (C: drive if Windows, or / partition if Linux), which contains the guest operating system configuration and applications. The user image (D: drive if Windows, or /home partition if Linux), which contains user settings and personal files, will be unique and persistent to each dynamic virtual desktop's user. Any changes made to the published session, in terms of guest operating system configuration, system settings, or applications, will be available to dynamic instances the next time they are started, without harming the individual user settings. This allows for seamless centralized updates and application deployment without having to manage the dynamic desktops individually.

For more information on provisioning virtual machines for user accounts, please see the section *Provisioning User Accounts* in *Part 2* of the *Win4Lin Pro/Win4Solaris Pro Users Guide*.

Virtual Desktop Administration

Virtual Desktop Administration is a two part process. The first part focuses on virtual machine parameters, such as RAM and shared folder assignments. The second part focuses on installing, updating, and configuring software inside the virtual desktop environment itself.

Adjusting Virtual Machine Settings

Using the *Win4Lin Pro/Win4Solaris Pro Console*, you can adjust many virtual machine parameters. Examples include RAM assignment, shared folder configuration, and networking parameters. After you install a virtual desktop, a default parameter set is created using the “best-practice” values for the particular virtual machine type (hosting either a Windows or Linux desktop). However, in cases where parameters need to be adjusted, you should log in as the user who hosts the virtual machine, and run the following command:

```
win4console
```

From this environment you may adjust properties, create backups, and even install new virtual machines graphically. Please note however that the installer is geared specifically toward Windows guests, and should not be used to install Linux virtual desktops.

While most settings in the *Properties* dialog box apply specifically to Windows guests, many translate well to Linux guests as well. Important settings you may consider for either type of virtual desktop include:

- Virtual desktop RAM assignment and keyboard locale under the *General* tab
- Network parameters under the *Networking* tab (e.g. if more advanced networking is required, such as Bridged); please note that in most dynamic desktop deployments, Bridged networking is not only unnecessary but may also reduce the scalability of both the server and the surrounding network for large deployments
- Shared folder configuration under the *Shared Folders* tab – for Linux guests, these folders may be accessed graphically from the desktop as SMB shares, or even mounted as `cifs` or `smbfs` file systems automatically using the guest's `/etc/fstab` file

For desktops deployed dynamically or accessed remotely, the following types of settings should never be modified from their default values:

- *Display*
- *Audio*
- *Protection*
- *RDP*

For best results, do not modify any settings in the *Advanced* tab unless you are absolutely certain of the changes. Doing so may render the virtual desktop unbootable.

Finally, please note that virtual machine *Properties* may only be adjusted for “gold image” and/or static/stand-alone configurations. Dynamic virtual desktops inherit their properties from the “gold image” they are provisioned from.

For more information on using the *Win4Lin Pro/Win4Solaris Pro Console* to configure virtual desktops, please see the sections *Using the Win4Lin Pro/Win4Solaris Pro Console* and *Maximizing the Win4Lin Pro/Win4Solaris Pro User Experience* in *Part 1* of the *Win4Lin Pro/Win4Solaris Pro Users Guide*.

Administering the Guest Virtual Desktop

To install, update, or configure patches and applications within the “gold image” virtual desktop itself, you must actually start the virtual machine and perform these changes as you would on a native desktop.

“Online” Updates for Small Configurations

On single-server systems, you may choose to perform “online” updates to the virtual desktop image. This requires that you first shut down or abort all running dynamic instances of the “gold image”. Note that you cannot start the “gold image” itself if there are any dynamic instances of it running, as this would corrupt the dynamic instances. VERDE will not permit this, and generates an error if you attempt it.

Once all dynamic instances of the “gold image” are shut down or aborted, you simply start the “gold image” virtual desktop as explained in *Starting the Virtual Desktop* under the section *Installing a “Gold Image” Desktop Virtual Machine of Installation*.

After you perform the necessary updates, which may include installing new software, applying guest operating system security patches, and/or adjusting guest system configuration settings, simply shut down the guest using its OS-specific shut down feature (e.g. select *Shutdown* from the *Start Menu* if using a Windows guest). At this point users may once again start dynamic instances of this “gold image”, at which point they will automatically inherit all the changes you made during the update process.

For information on shutting down dynamic instances from the system administrator role of the server, please see the section *Session Management* later in this guide.

“Replica” Updates for Large Configurations

If your server is part of a cluster or you just wish to perform “gold image” maintenance without having to first shut down dynamic instances, you can use the “replica” method instead of the “online” method described above”. This involves copying the entire “gold image” virtual desktop folder, updating the copy, and then renaming the copy back to the original “gold image” folder.

This method has a two advantages over the “online” method. First, it is strongly recommended if you use a management workstation that works on shared storage and authentication rather than performing the maintenance directly on the VERDE server. This is especially relevant if you are administering virtual desktop images on a cluster.

Second, you do not need to schedule a specific maintenance window, since you do not need to shut down dynamic instances first. This means users that depend on this “gold image” can keep working as you update it. All they would need to do is restart their virtual desktop session (or log shut it down and start it again) in order to inherit the updates.

This method is generally preferred over the “online” method for large organizations or for applying emergency patches or updates that cannot interrupt users.

Note that this method involves duplication of storage for the “gold image” virtual disk images, and can only be performed on the same partition, since it involves renaming adjacent folders. You must have sufficient free storage in the home directory of the “gold image” user before attempting it.

The following example assumes that the you are logged in as the non-`root` user who hosts the “gold image” virtual desktop. Notice that first the image is copied to a replica, then the replica is started, then the replica is renamed back to the original image. Comments are provided for convenience and clarity:

```
# make sure current directory is home directory
cd
```

```
# create a replica of the win4 virtual machine, deleting any previous
# replicas
rm -rf win4.replica
cp -Rp win4 win4.replica

# start the replica virtual machine after performing the
# necessary updates inside, the guest OS should be shut down cleanly
win4 win4.replica

# move the existing win4 virtual machine out of the way,
# then rename the replica to win4; first, delete any
# left-over copies of the old win4 virtual machine to avoid
# move problems and save storage
rm -rf win4.old
mv win4 win4.old; mv win4.replica win4
```

It is recommended that the `win4` configuration be backed up first before attempting this, in case the procedure fails or you need to revert the changes. Note that after the last command is executed above, users logging into the server or cluster who start dynamic instances of this virtual desktop will automatically inherit the latest changes.

Connecting Remote Users to VERDE

While VERDE servers accept many inbound connect methods for remote sessions, using a native Virtual Bridges Client program, available for various client platforms, leads to the best possible user experience. Also, if connecting to a VERDE cluster, the Virtual Bridges Client is the only supported method.

The Virtual Bridges Client programs for all supported client platforms are available from a link on the VERDE/Win4VDI download page.

For more information about connecting remote users, including fire wall considerations, please see the section *Connecting Users to Win4Lin Pro/Win4Solaris Pro VDI* in *Part 2* of the *Win4Lin Pro/Win4Solaris Pro Users Guide*.

Session Management

VERDE servers provide tools for listing, shutting down, and aborting user sessions. Since each session runs as a Linux/Solaris process group, it has a unique PID number.

Listing Running Sessions

To list the running sessions on the server, run the `win4-sessions` command:

```
win4-sessions [-n]
```

The -n parameter is used to optionally list the user for each session as a numeric UID rather than a name. This may be desirable if you are wrapping this command in a custom script.

Example: list the running sessions on the server:

```
win4-sessions
```

Note that there is no need to run this command as root, as it simply lists information.

Each session actually manages a group of processes. The PID you see in the list is the “runtime” process ID, which is the parent of all related processes for that VM. The actually core virtual machine process running in that process group is either called `kvm` or `qemu`, depending on the capabilities of the host operating system and hardware. You should never attempt to `kill` individual processes in the process group unless killing the runtime PID, described by the `win4-sessions` command, is not effective. Also, never attempt to use `kill -9` until you have attempted `kill -15` and waited a few seconds with no result. Note that using `kill -9` on any process in this process group may leave system resources such as shared memory and semaphores uncleared, and should only be used as a last resort.

Shutting Down Sessions

The `win4-shutdown` command, which must be run as `root`, is used to gracefully shut down or abort virtual desktop sessions:

```
win4-shutdown [-a] [-s] [-t <timeout>] <pid>
```

Option	Description
-a	Abort session immediately; please note that this option should be used only if the session is unresponsive, and in some cases may lead to minor data loss or corruption. It is the equivalent of pressing the power button on a native PC
-s	Attempt graceful shut down; this is the equivalent of using the Shut down option in the guest operating system (from the <i>Start</i> menu on Windows, or with the <code>init 0</code> command on Linux)
-t <timeout>	If combined with -s parameter, specifies a timeout, in seconds, to wait for the session to shut down gracefully before aborting it
<pid>	Process ID of session as reported by <code>win4-sessions</code> command

Example 1: abort session with PID of 12543:

```
win4-shutdown -a 12543
```

Example 2: shut down session with PID of 12543:

```
win4-shutdown -s 12543
```

Example 3: shutdown session with PID of 12543, waiting up to 60 seconds for it to shut down cleanly before aborting it:

```
win4-shutdown -s -t 60 12543
```

Example 4: Bourne shell script to shut down all running sessions on this server gracefully:

```
#!/bin/sh

# walk through list of all sessions, skipping the header, getting just
# the PID
for i in `win4-sessions |grep -v ^PID |cut -d ' ' -f 1`; do
    win4-shutdown -s $i
done
```

Login Scripting and Automation

Login “Hooks”

The VERDE connection broker, which presents users with a server login screen and instantiates or resumes virtual machines based on authentication and authorization, provides several integration “hooks” where system administrators can add custom scripts. These hooks include:

1. Pre-show login window (run as `root`, as soon as user connects but before being challenged for authentication)
2. Post-show login window (run as `root`, immediately after user login dialog box is presented)
3. Post-login success (run as `root`, immediately after user is successfully authenticated)
4. Post-login failure (run as `root`, immediately after user authentication fails)
5. Pre-session launch (as authorized non-`root` user, can be used to perform user-level tasks after privileges are dropped but before user virtual machine starts)
6. User session launch command (as authorized non-`root` user, can be used to wrap virtual machine startup/resume via the default `win4` command)
7. User desktop launch command (as authorized non-`root` user, can be used to start a non-virtual machine application or desktop environment if the user has no provisioned virtual desktop)

Example Uses

All hooks except for the user session launch command and user desktop launch command are unassigned by default, meaning that they are not executed. You may assign a script, executable program, or shell command to any or all hooks. Examples of why hooks may be used include:

- To present users with important, organizational-specific information or set a custom login screen background upon connection to the server
- To create home directories for users who may not have home directories yet (e.g. users created on a directory/authentication server but who have never logged in to the VERDE server)
- To deploy dynamic virtual machines to users who may not have such virtual machines provisioned yet (e.g. new users logging into the server for the first time belonging to a certain group or role)
- To collect, analyze, and/or save client-specific parameters from Virtual Bridges Client connections
- To delete remnants of persistent user data before starting virtual desktops in the case where a completely transient user experience is desired
- Many other, organization-specific tasks

General Hook Rules

The following rules apply to hook commands, scripts, and executable programs:

- Each hook command must be executable by the `/bin/sh` shell on the system
- Each hook command must return a 0 value, or the user will be presented with an error message

indicating that the hook command failed (this may be desirable in situations where failures should be reported)

- Hook commands should execute as quickly as possible to avoid “hanging” the user for prolonged periods of time – note that hook commands execute serially

Login Hook Assignment

Assigning hook commands to the login process on the server can be performed by running the `win4console` command as `root` on a graphical session. The *Remote Login “Hooks”* tab provides a user interface for assigning these commands.

Alternatively, you may edit the `/var/lib/win4linpro/settings.global` (Linux server) or `/var/lib/VBICw4s/settings.global` (Solaris file) file manually. The following settings correspond to each login hook command:

Setting	Description
<code>WIN4_HOOKCMD_PRESHOW</code>	Pre-show login window (as <code>root</code>)
<code>WIN4_HOOKCMD_POSTSHOW</code>	Post-show login window (as <code>root</code>)
<code>WIN4_HOOKCMD_LOGINOK</code>	Post-login success (as <code>root</code>)
<code>WIN4_HOOKCMD_LOGINFAIL</code>	Post-login failure (as <code>root</code>)
<code>WIN4_HOOKCMD_SESSEXEC</code>	User session launch command (as authorized non- <code>root</code> user)
<code>WIN4_HOOKCMD_DESKTOPEXEC</code>	User desktop launch command (as authorized non- <code>root</code> user)
<code>WIN4_HOOKCMD_PRELAUNCH</code>	User session pre-launch command (as authorized non- <code>root</code> user)

You should enclose the values of these settings in quotes if they contain spaces in them. Some example values may include:

```
WIN4_HOOKCMD_PRESHOW="/usr/local/bin/myscript"
```

```
WIN4_HOOKCMD_LOGINOK="/usr/lib/win4linpro/bin/win4-cda-paramset >p.log"
```

```
WIN4_HOOKCMD_SESSEXEC="/usr/local/bin/win4-wrapper.sh"
```

Note that the above examples are for illustration purposes only. Also note that you should specify absolute paths to commands for best security/reliability practice.

Login Hook Environment Variables

Variable	Set for Hook
<code>PATH</code>	All hooks; set to <code>"/sbin:/usr/sbin:/bin:/usr/bin"</code> for hooks that run as <code>root</code> , and <code>"/usr/bin:/bin:/usr/bin/X11:/usr/openwin/bin"</code> for hooks that run as non- <code>root</code> users
<code>DISPLAY</code>	All hooks
<code>HOME</code>	All hooks that run as authorized user; also current working directory for these hooks is set to home directory

Variable	Set for Hook
PWD	All hooks; set to / for hooks that run as <code>root</code> , and the respective home directory for hooks that run as <code>non-root</code> users
WIN4_USERNAME	All hooks except Pre-show and Post-show – indicates the user name specified or authenticated
WIN4_CONFIGNAME	Pre-launch and User session launch command – indicates the virtual machine configuration that is to start (default is <code>win4</code>)

Dumping Virtual Bridges Client Parameters

The `/usr/lib/win4linpro/bin/win4-cda-paramset` (Linux server) or `/opt/VBICw4s/bin/win4-cda-paramset` (Solaris server) program can be used from login hook commands to dump an indexed list of client-specific parameters that Virtual Bridges Client programs record:

```
/usr/lib/win4linpro/bin/win4-cda-paramset [-h] [-g] [-q] [-m] <param>
/opt/VBICw4s/bin/win4-cda-paramset [-h] [-g] [-q] [-m] <param>
```

Option	Description
-h	Show the help screen
-g	Show graphical progress bar while fetching parameters
-q	Exit quietly if client does not provide parameters
-m	Use Microsoft-format (CRLF) for dump
<param>	Parameter number (0-255), or all to dump all parameters, one on each line

All parameter values are dumped to the standard output and may of course be redirected to a file. In some cases parameters may be analyzed by hook scripts, or even dumped to a file in a well known location (relative to a user's home directory) so that it is then available via shared-folders from inside the virtual machine session.

Parameter values include:

Parameter #	Description
128	User name specified in Virtual Bridges Client connection GUI
129	Local (client) user domain (Windows)
131	Local (client) authenticated user ID that is running the Virtual Bridges Client program
140	Client operating system type (e.g. Windows, Linux, Solaris, etc.)
141	Client operating system version
150	Client operating system's root path/drive letter
151	Client operating system's "system" path/drive letter
152	Client operating system's user home directory/drive letter
153	Client operating system's temporary file directory/drive letter
154	Client operating system's user desktop directory/drive letter
155	Client operating system's user document directory/drive letter

Parameter #	Description
156	Client operating system's local default printer name
160	Client computer's local default interface IPv4 address
161	Client computer's local default interface IPv6 address
162	Client computer's local default interface Ethernet MAC address
163	Client computer's local host name
164	Client computer's local default domain name
170	Client computer's local time zone "bias" - offset in minutes from UTC time
171	Client computer's local time Daylight Savings Time setting (either <i>yes</i> or <i>no</i>)
172	Client computer's local connection time (HH:MM:SS)
173	Client computer's local connection date (MM:DD:YYYY)
180	Client computer's local native screen resolution (WxH)

Please note that not all parameters may be defined, and the actual values are general specific to the client operating system type and version.

Clustering

VERDE Clustering is an extension of VERDE server. A VERDE cluster consists of multiple VERDE servers to provide distributed virtual desktops to large numbers of users. Most of the concepts relating to virtual desktop installation, provisioning, and automation apply identically in a clustered environment.

Virtual Bridges provides the following ancillary documentation for clustering:

VERDE Cluster Overview – provides high-level and architectural information on the clustering technology, including design, protocols, resource planning, and capacity metrics.

VERDE Cluster Quick-start Guide – provides step-by-step instructions for configuring and deploying a basic VERDE cluster.

Please refer to the above mentioned documents for information on clustering VERDE systems.

Legal

VERDE, Virtual Bridges, the Virtual Bridges logo, and Win4VDI are trademarks of Virtual Bridges, Inc. Other company, product, or service names may be trademarks or service marks of others.

Copyright © 2009 Virtual Bridges, Inc. All Rights Reserved.