A Practical Guide to WinCVS and SSH

Patrick Reagan

Table of Contents

Introduction

The Concurrent Versions System, more commonly referred to as 'CVS', can become a crucial part of your software development environment. While CVS is traditionally a command line application, other developers saw the need for a GUI application to interface with the extended set of commands that CVS provides. One such application is WinCVS[1].

As it's name implies, WinCVS is basically a CVS client written for use on the Windows operating system. This application gives the user the option of connecting to a CVS server in a multitude of environments. There is support for connecting to a local repository, as well as a remote repository via pserver, SSH, or a number of other methods. We'll be examining how to access a remote repository using the SSH method.

I find CVS useful in my web development projects where I want the application hosted on a Linux/Apache machine, but want to develop on a Windows operating system. These instructions are written from that point of view. In order to get started with WinCVS, we must first make sure we have the prerequisites: a CVS server with a repository we can use, an SSH daemon on said server, and a means of connecting to the SSH server.

Installing and Configuring the CVS Software

Installing the CVS software is out of the scope of this document. For a detailed tutorial on getting the software up and running, check out the HOW-TO that is part of the Linux Documentation Project[2]. Follow the instructions to install the software, but don't bother with learning how to use the commands, we will cover the necessary ones later. If you are installing on a system that doesn't support RPM, you can download the source from the CVS home page[3].

Creating the Source Repository

Now that the software is installed, we need to create a source repository. From your home directory, issue the following commands:

```
$ mkdir myproj
$ cd myproj
$ touch my-project.txt
$ cvs import myproj Vendor-1 Release-1
```

The last action will bring up your default editor, which you can use to enter any comments about the project that you wish. Once that file is saved, the project files will be committed to the repository.

---

[1] The WinCVS Web site is located at: http://www.wincvs.org. The site provides links to download the client software as well as documentation to get you up and running with using CVS to manage your projects.

[2] The Linux Documentation Project provides some great resources. The CVS/RCS How-To will give you everything you need to set up a server installation of the CVS software. You can find the document at http://www.linuxdoc.org/HOWTO/CVS-RCS-HOWTO.html.

[3] The CVS Web site provides software downloads as well as all the necessary documentation to get you familiar with the software. The URL is: http://www.cvshome.org.

```
N myproj/my-project.txt
No conflicts created by this import
```

After the initial import, it is common practice to get rid of the original files since we will be retrieving them from CVS later. Delete the 'myproj' directory and all the files in it:

```
$ cd ..
$ rm -rf myproj/
```

That's all there is to it, the 'Vendor-1' and 'Release-1' tags are required to import the source code into the tree. They identify the vendor and release of the software. They can only contain numbers, letters, and other non-whitespace characters excluding the dot ('.') character.

Installing and Configuring SSH

Installing the SSH binaries from RPM is rather easy. Get the RPM files from the OpenSSH web site[4]. If you're using a Red Hat based system, follow the 'Linux RPM' links to download. Once downloaded, issue the command (as root):

```
# rpm -Uvh openssh-<version>.i386.rpm
```

This will install all the required binaries, configuration files, and startup scripts for the server. If you're installing from source, download the latest tarball and follow the included instructions. Keep in mind that the SSH server installation requires the OpenSSL libraries[5].

Logging into the SSH Server

Once you have verified that the server is running, you can test your connection by using the SSH client program from the same machine:

```
$ ssh <username>@localhost
```

All you need to do is enter the password, and you should see another prompt. You will see an informational message before entering your password that looks like:

```
The authenticity of host can't be established.
RSA1 key fingerprint is ...
Are you sure you want to continue connecting (yes/no)?
```

If you trust this host (which you should if you're just connecting to the local machine), type 'yes' after the prompt.

---

[4] If you need the OpenSSH server software, you can download it from: http://www.openssh.com/portable.html.

[5] The OpenSSL libraries that may be required for your OpenSSH server installation can be found at: http://www.openssl.org.

Public Key Authentication

Public key authentication is crucial to the communication between the client and the SSH-enabled CVS server. Since every action on the CVS server will require authentication, public key authentication gives us the convenience of not having to enter a password while maintaining security in the authentication and data exchange process.

Generating and Installing Keys

While you are still logged-in to the server, you need to generate the key pair for public key authentication.

```
$ ssh-keygen -b 1024
```

The program will respond with some information and ask for a passphrase. Since we want CVS to operate seamlessly through SSH, press the 'Enter' key twice to use no passphrase:

```
Generating public/private rsa1 key pair.
Enter file in which to save the key (~/.ssh/identity):
(hit enter to store in the default)

Enter passphrase (empty for no passphrase):
Enter same passphrase again:

Your identification has been saved in ~/.ssh/identity.
Your public key has been saved in ~/.ssh/identity.pub.
The key fingerprint is: ....
```

The file 'identity' is the private key and the 'identity.pub' is the public key. The public key is stored on the server and the private key is used by the client for authentication. The next step is to install the keys on both the client and server.

Installing the public key on the server is easy:

```
$ cat ~/.ssh/identity.pub >> ~/.ssh/authorized_keys
$ chmod 400 ~/.ssh/authorized_keys
```

Now to install the private key, you must copy it to your client machine. The best way to do this is by using an SCP client program to connect to the server and transfer the file. I recommend using WinSCP, a free Windows GUI-based SCP client[6], to transfer the file somewhere you can easily locate it later.

Connecting With the Private Key

For our purposes we need a command line client for when we attempt to use CVS over a secure connection. You need to download the Windows OpenSSH port that is available from the NetworkSimplicity site[7].

---

[6] The excellent (and free!) WinSCP client is available from: http://winscp.vse.cz.

[7] To connect to your SSH server, you need the command line interface for SSH available from the NetworkSimplicity Web site: http://www.networksimplicity.com/openssh.

Once you have downloaded and installed the package, you need to set some environment variables. Open up the command line window (a.k.a. 'MS-DOS Prompt') and enter the following:

For Windows 2000:

```
set HOME=C:\Documents and Settings\<username>
```

For Windows NT 4.0

```
set HOME=C:\WINNT\Profiles\<username>
```

For Windows 9x

```
set HOME=C:\WINDOWS\Profiles\<username>
```

If the path "C:\Program Files\NetworkSimplicity\ssh" isn't set in the 'PATH' variable, add it. This will allow you to easily use the ssh program from the command prompt.

```
set PATH=%PATH%;C:\Program Files\NetworkSimplicity\ssh
```

Now, test your configuration by entering:

```
C:\>ssh <username>@<ssh-hostname>
```

where <username> is a valid account on the <ssh-hostname> machine. You will receive a prompt similar to this:

```
The authenticity of host '<ssh-hostname>' can't be established.
Are you sure you want to continue connecting (yes/no)?
```

Type 'yes' at the prompt, and the key will be added to the 'known_hosts' file on the client. Now enter your password at the prompt and you should be able to log into the machine.

Once you have the host key cached and password authentication working, It's time to get the public key authentication working. Since you have already created the public/private key pair, and have the private key stored on the client, move it to the %HOME%/.ssh/ directory. Now, issuing the command:

```
C:\>ssh <username>@<ssh-hostname>
```

or, if your private key file is not found:

```
C:\>ssh –i ~/.ssh/<private-key file> <username>@<ssh-hostname>
   (If you named your private key something other than 'identity')
```

This should log you right into the SSH server. If you are having problems, refer to the troubleshooting section at the end of this document.

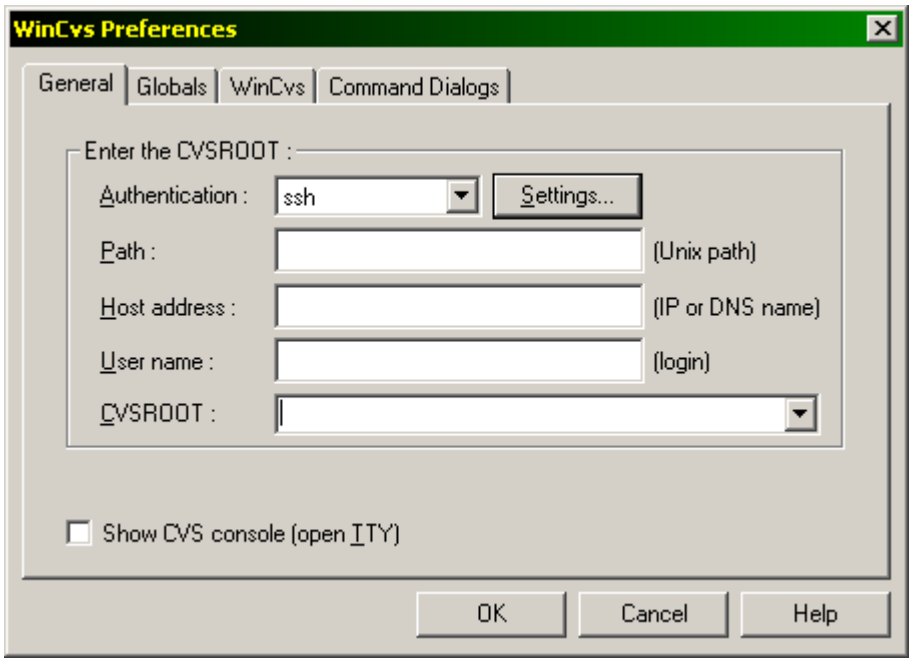
Once the SSH client configuration is complete, we can connect to our CVS repository over a secure connection.

Installing and Configuring WinCVS

The WinCVS application is a useful piece of software for accessing and managing CVS repositories on a multitude of hosts using different configurations. To get the latest version of the software, point your browser to the WinCVS home page and follow the link to the download page. Once the software is installed, you can begin to configure it for use with the CVS server we just set up. At the time of writing, the newest version was 1.3 Beta 4. If you have a newer version, the following screens and options may differ slightly from what you see.

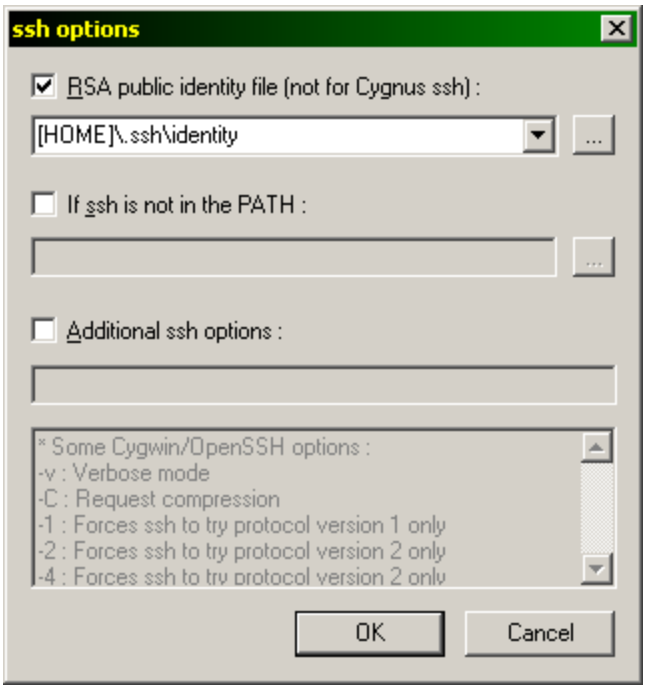
Setting Up WinCVS to use SSH

After installation, the next step is to configure the application to connect to the remote repository. To start this process, click on the 'Admin' menu item and select 'Preferences…'. You will be presented with the following dialog box:

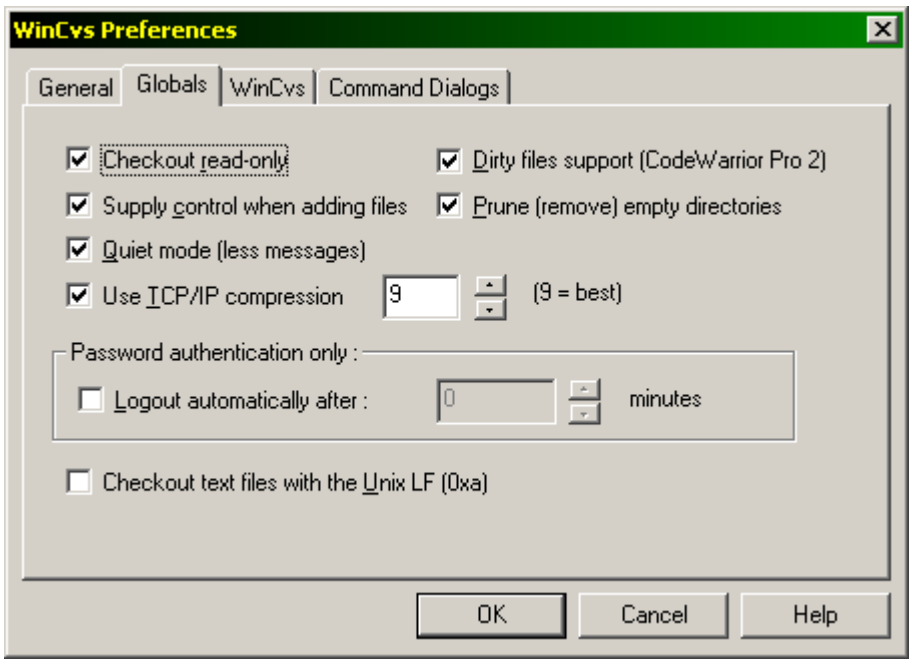


It's obvious that we will be using SSH authentication, but the others may not be that obvious. In the 'Path' field, enter the path to the repository root (from the CVS configuration section). Typically this will be '/home/cvsroot', but you or your administrator may have set this differently. The 'Host address' is the IP address or fully qualified domain name of the server where the CVS repository is located. The 'User name' is your username that you use to log onto the machine through SSH.

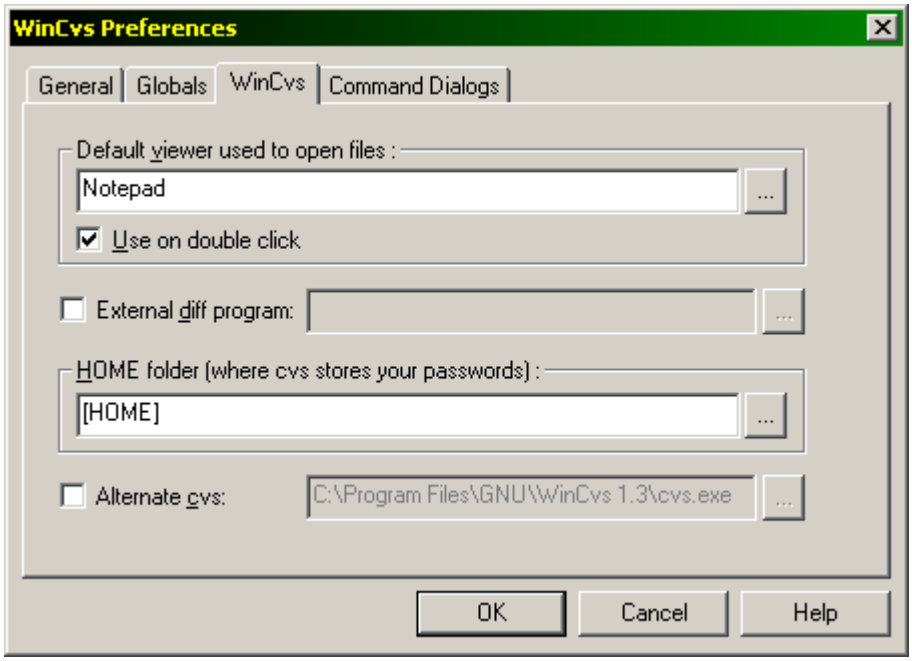
There are some other options that need to be configured for the application to work properly with our CVS server. Click on the 'Settings…' button to bring up the SSH options dialog:



At the very least, we need to select the RSA public identity file to authenticate ourselves with the server (this was created in the SSH public key configuration section). The use of [HOME] in the above example refers to the path you set as HOME in the earlier SSH configuration (i.e. 'C:\WINNT\Documents and Settings\<username>'). The SSH application should be in the PATH from the SSH configuration section, so you typically won't need to specify this option. You can also specify other options for SSH if you need to connect on another port or would like to see additional debugging information. Click OK' once you have set all the options and then click on the 'Globals' tab in the main dialog:



Many of the options on this page will be selected by default. You should check the 'Checkout read-only' option and the 'Use TCP/IP compression' option as well. Selecting '9' from the list will give you the best gzip compression during the data transfer with the CVS server. Once these options are set, click on the 'WinCVS' tab to enter some final options:



Here you need to supply the path to the 'HOME' directory, what you initially set as the home for storing all your SSH options. Once this is completed, click the 'OK' button. You will see the following message displayed in the output frame at the bottom:

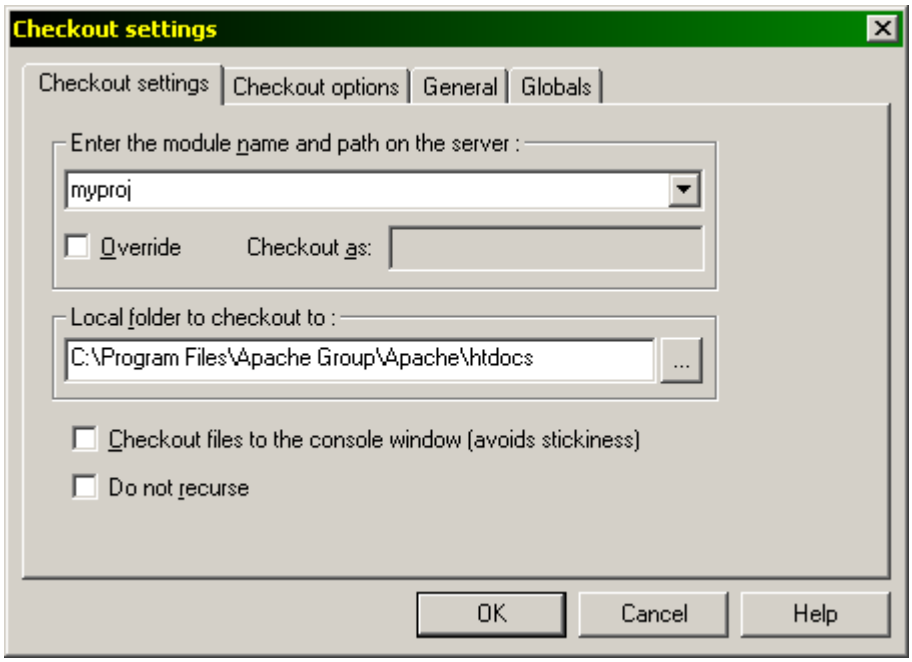
```
NEW CVSROOT: <user>@<cvs-server>:/home/cvsroot (ssh authentication)
```

Using the WinCVS GUI

Retrieving the Source

Now it's time to get the source repository that we recently created. If you have used CVS before using the 'pserver' authentication mechanism, you know that you have to log in to the repository before you can perform actions against any file stored within. This is not the case with SSH authentication because every CVS command executed against the main repository is authenticated and filtered through SSH first.

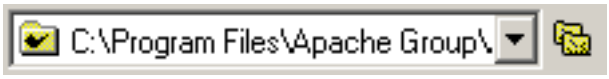
With this in mind, the first step in working with the repository is to checkout the newly created project. Click on the 'Create' menu item and select 'Checkout module…' from the menu. The following screen will appear:



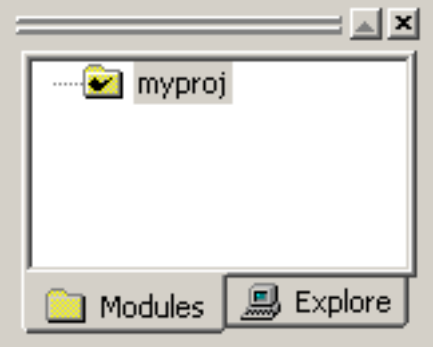
The two options you need to configure here are the module name (project) and the local folder. The module name is the name of the project we created earlier, 'myproj'. I am using Apache for Windows when I develop most of my applications, so I just use the web root to check the files out to. Feel free to change this to suit your needs. Clicking 'OK' will retrieve the current source tree from the repository; the output window will show the results of your checkout operation:

```
cvs -z9 -q checkout -P myproj
   (in directory C:\Program Files\Apache Group\Apache\htdocs)

U myproj/my-project.txt

*****CVS exited normally with code 0*****
```

Once you have checked out the source tree from the repository, it is helpful to change the location of the 'modules' window on the left hand side. To do this, click on the 'Change location' icon (looks like two folders) to the right of the path drop-down menu:



Select the 'myproj' directory under the checkout folder you selected in the previous step. The display in the modules section should look like this:



This will make it easier for managing your project. If you are working on several different projects, you may want to select the main repository as the module root. That way, you can view multiple projects at the same time. The 'Change location' option stores a history of locations in the drop-down, so going back to previous locations is simple.

Editing Files

Highlighting the 'myproj' in the modules section will display the listing of files in the root of this source repository:

Name	Rev.	Status	Date
my-project.txt	1.1.1.1	File	Thu Nov 15 15:48:48 2001

In addition to the list of CVS-managed files, this will also display any folders in the project as well as files that are in the local directory but are not stored in the main repository.

Now it's time to start modifying the source code in the repository. Since all the files in the repository are marked read-only, we need to flag them for editing. You can cheat and remove the read-only flag in Windows, but we need to inform the CVS server that we are going to edit the file. Highlight the 'my-project.txt' file and select 'Edit' from the 'Trace' menu. You will see some informational messages in the output pane at the bottom indicating the status of the 'edit' command. When this is successful and the file is still highlighted, select 'Trace->Editors' from the main menu. This will display the list of editors that are currently editing this particular file. You should see your username and other information in the output pane. If others are editing this file, you will see their usernames as well.

Now that you are an editor of the file, fire up your favorite text editor and get to work. Type a brief message into the file and save it. Once you save the file, looking at the file listing in the main WinCVS window will show that the file has changed:

Name	Rev.	Status	Date
my-project.txt	1.1.1.1	Mod. File	Thu Nov 15 15:48:48 2001

The 'pencil' icon to the left of the file indicates that this file is editable, while the red color of the 'file' icon indicates that this file differs from the version stored in the main repository. When you're finished making changes to your version of the file, highlight the file in the listing and select 'Modify->Commit…' from the main menu. Enter any notes you want to store about this file and click the 'OK' button. You'll notice that the icons in the file listing look the same as before, and the number under the 'Rev.' header changes to '1.2'.

Adding Files to the Repository

When you create new files in the project folder on your local machine, these files will appear in the WinCVS file-listing window with a status of 'NonCVS file'. This means that these files exist on the local machine, but the CVS server knows nothing about them. To inform the CVS server of these files, you must add them to the repository. Highlight the file or files you wish to add, and select 'Modify->Add selection' from the top menu. The CVS server will respond:

```
cvs server: use 'cvs commit' to add this file permanently
```

Now, all you need to do is commit the file to the source repository. One item of note regarding adding files is the concept of text versus binary files. When using CVS for web applications, you will have a collection of source code and graphics. In this case, you will need to distinguish the source files as text (regular CVS files) and the graphics as binary. The crucial point in doing this is during the addition stage. Instead of selecting 'Modify->Add selection' like we did in the last step, you would select 'Modify->Add binary' from the menu. This will flag the file in the repository as a binary file, and keep it from being corrupted after future commits.

Removing Files from the Repository

There are two sides to removing files from the source repository: removing the file from the local working directory and removing the file from the repository on the server. To remove a file, highlight it in the file listing and select 'Modify->Delete' from the main menu. WinCVS will automate this two-step process by first moving the file to the Windows Recycle Bin and then scheduling the file for removal from the repository. Once this action is complete you just need to commit the file and it will be removed from the current source tree.

Recovering Removed Files

If, during this process, you decide that removing the file was a mistake, you can easily recover it. First, open the Recycle Bin and find the file that was deleted. Right-click the file and select 'restore' from the menu. This will move it back into the directory you removed it from. It will now appear in the WinCVS file listing as a 'NonCVS file'. Now, just follow the steps to add it to the repository and it will be created again. In this case, the message from the server is a bit different:

```
cvs server: re-adding file my-project.txt (in place of dead revision
1.3)
cvs server: use 'cvs commit' to add this file permanently
```

The deleted file will retain the original history of the deleted file as well as a new revision. If you accidentally delete the file on the local machine outside of WinCVS, you can simply update the source tree to retrieve the last saved version. We'll learn more about the update process in the next section.

Working with other Developers

Updating your local sources

When working on a project with other developers, changes will be made to some of the files in the source tree as you're working on others. From time to time, you may need these updated sources to continue with your work. The process of retrieving the current sources is known in the CVS world as 'updating'.

To get the latest version, highlight the root folder of your project in the module window on the left and select 'Modify->Update selection...' from the main menu. If any files have changed, you will see a list of these files in the CVS output window at the bottom.

Dealing With Conflicts

With multiple developers working on a single project, sometimes conflicts occur due to the design of the CVS software. Since CVS is a concurrent versioning system, multiple developers have the ability to modify the same file simultaneously. Sometimes when performing a 'commit' operation, your changes will conflict with another developer's changes. If this happens, you'll see a message like:

```
cvs server: Up-to-date check failed for `conflict-file.txt'
cvs [server aborted]: correct above errors first!
```

This means that someone has committed a new version of the file since the last time you updated your sources. Highlight the offending file and select 'Modify->Update Selection...' from the main menu. Click 'OK' for any dialog boxes that may appear. If you're lucky, the update will succeed and you can then commit your changes. In the event of a conflict you'll see something similar to:

```
warning: conflicts during merge
cvs server: conflicts found in <cvs-filename>
C <cvs-filename>
```

Where <cvs-filename> is the name of the file you are committing. Since there was a conflict with the commit operation, your changes were not added to the source in the repository. Instead, the original file contains both your changes and the changes the other developers have made. Here is an example:

```
<<<<<<< conflict-file.txt
user entered this line..
=======
this will conflict
>>>>>>> 1.6
```

The angle brackets delimit the conflicting bits of code. The top portion (i.e. everything before the '=' characters) is what you attempted to add to the file, while the bottom is what existed in the repository before you tried to commit your changes. To fix this

problem, decide which code is correct and remove all the delimiters. Once the file looks correct, you can then commit it to the repository. Remember that you should be communicating with the other developers both to avoid potential conflicts as well as resolve them when they occur.

Troubleshooting

Connecting via SSH

When testing the SSH connection via the command line and the public key authentication method fails, use the '-v' option to give some verbose output. These messages will assist you in diagnosing the problem.

One such problem is that public key authentication may not work with SSH protocol version 2 in some instances. To force the connection with version 1, just add '-1' (dash one) to the command:

```
C:\>ssh -1 <username>@<ssh-hostname>
```

Another common problem when connecting is the message:

```
Remote: Authentication refused: bad ownership or modes for directory
/home/<username>
```

This simply means that the permissions on your home directory are not set correctly for SSH public-key authentication to work, however, password authentication will continue to work. To get around this problem you must set the permissions of your home directory to at least 750, but it is advisable to set them to something more restrictive like 700.

Conclusion

Both the WinCVS application and the SSH protocol provide great enhancements to the CVS application software. While it is a bit more difficult to set up the public-key authentication with SSH, the benefit is the increased end-to-end security that SSH adds to the CVS software. Despite these features, this method also has its drawbacks. In order to make this work, each CVS user must have a shell account on the server. While this poses no problem for internal projects, this may not be an effective method for other public CVS managed projects.

Credits and Resources

I wrote this guide after figuring out how to use CVS over an SSH connection. The documentation on the WinCVS web site proved useful, but some of the instructions were outdated and points I thought needed more coverage were glossed over. The 'SSH with WinCVS' tutorial[8] on the WinCVS site was very helpful as was the detailed 'CVS Manual'[9]. For more information on these tools, check out the appropriate sites:

CVS	http://www.cvshome.org
WinCvs	http://www.wincvs.org

[8] This tutorial can be found at http://www.wincvs.org/ssh.html
[9] The main CVS manual can be found at http://www.cvshome.org/docs/manual/cvs.html

Please send any errors or corrections to me at reaganpr@hotmail.com.

Appendix A: Status Information

The WinCVS application has a couple ways of informing the user of the current status of any file stored in the local directory. Some are strictly limited to the WinCVS application, while others are inherent to the CVS software.

WinCVS Status Icons

These icons appear to the left of files in the main file display window. Here is what each icon represents:

 = Non CVS file
 = File added to repository, not committed
 = Binary file added to repository, not committed
 = CVS file that has been modified locally
 = CVS binary file that has been modified
 = CVS file scheduled for removal from the repository
 = CVS file with conflicts
 = Unchanged CVS file
 = Unchanged CVS binary file

CVS Status Codes

You may have noticed some messages scrolling by in the CVS output window regarding files in your source tree. Each message is color-coded and has a code to the left of the file name. The 'U' in the following display represents an internal CVS status code:

```
U myproj/conflict-file.txt
```

Here is what each code represents:

- **U** – The file was updated in your local source directory.
- **P** – Similar to 'U', but your local copy was patched instead of being replaced.
- **A** – The file has been added to the local directory and will be added to the main repository on the next commit.
- **R** – The file has been removed from your local directory and will be removed from the main repository on the next commit.
- **M** – The file has been modified in the local directory. This means that you have altered the file yourself, or another user modified the file and the merge was successful.
- **C** – A conflict occurred when trying to commit the file, you must correct the conflict and try again (See the 'Dealing with Conflicts' section).
- **?** – The file exists in your local directory, but the CVS server has no knowledge of it. This will alert you to files that may need to be added to the repository.

Appendix B: GUI Configuration

While the WinCVS application is useful in its own right, I like to make a few configuration changes to make managing my sources a bit easier. Instead of having to use the menu items or toolbar icons, I like to have some of the more frequently used commands available when I right-click on an item. To customize this menu, simply right-click on an item in the main file-listing pane and select 'Customize this menu...'. From here, you can add and remove items using the 'Customize menu' dialog box.

Some of the actions I like to have in the context menu are:

```
Add Selection
Add Selection Binary
Update Selection
Commit Selection
Edit Selection
Unedit Selection
Editors of Selection
```

While configuring the context menu is not necessary, I find that it helps me perform the actions on one or more source files quickly and efficiently.