SSH slaves and Cygwin

Background

Cygwin is a Unix emulation library for Windows. It allows source code written for POSIX environment to be compiled with Cygwin stub files into Windows executables. When these Cygwin-enabled libraries run, a DLL gets loaded into these processes that implement all the POSIX APIs on top of Win32 APIs.

Because of the way it works, cygwin's presence does not change the way Java virtual machines work. JVMs that run on Cygwin-enabled Windows continue to behave exactly the same way as JVMs that run on cygwin-less Windows. It will continue to use '\' as the separator, not '/', `java.io.File` will not suddenly start understanding Unix path, etc. The path translation from the UNIX style to the Windows style happens within Cygwin DLL. Programs that are not compiled against Cygwin will not load Cygwin DLL, and as such they will not go through the path translation.

So when you use Cygwin and mix native Windows programs and Cygwin-compiled programs, you need to be mindful when and where the path conversions happen. For example, when you execute Cygwin-compiled processes (say `bash.exe`), these programs expect Unix-style paths in their command line. The same applies when we talk to Cygwin-compiled processes over various protocols, such as SFTP.

To make this further confusing, native Windows APIs recognizes '\' as the directory separator, in addition to '/'. Simiarly, Cygwin-emulated POSIX APIs accept Windows paths, in addition to the Unix paths. This helpful "smart" behaviour sometimes makes it difficult for users to understand where the path translation is really happening.

SSH Slaves plugin and Cygwin SSHD.

Cygwin comes with OpenSSH server, which works well with SSH Slaves plugin. This is one of the recommended way of controlling Windows slaves from Jenkins, if you don't mind the added effort of installing Cygwin and sshd :

1. Download cygwin with the following packages: (Admin) **cygrunsrv**, and (Net) **openssh**
2. Open a cygwin shell window and run the SSH configure: ssh-host-config -y
3. Run ssh daemon : cygrunsrv -S cygsshd
4. Check that your firewall allow TCP port 22
5. Java must be available from your ssh client: for example, add a symbolic link : cd /usr/local/bin && ln -s /cygdrive/c/Program\ Files\(x86\)/Java /jre1.8.0_211/bin/java.exe java

When you use SSH launcher to launch a slave on Cygwin-enabled Windows, you should still specify Windows style path as the remote FS root (such as `c:\jenkins`). This is because the slave JVM that eventually gets launched doesn't receive the Cygwin path translation. If you specify Unix style path (such as `/cygdrive/c/jenkins`), then Jenkins will end up trying to create both `c:\jenkins` (when it copies over `slave.jar` via SFTP) and `c:\cygdrive\c\jenkins` (when slave JVM actually starts and copy more files.)

If you run Jenkins on behalf of other users, you'll discover that some of your users will not understand when and where the path translation happens, and will inevitably write build scripts that break. You can explain to them what's going on, or you can surrender and use mklink to create a symlink or junction point that maps `c:\cygdrive\c\jenkins` to `c:\jenkins`. This will make those broken scripts work happily.

Treating Cygwin slaves like Unix slaves

Sometimes you want to treat Cygwin slaves like real Unix slaves, such as running shell scripts. When you do it, you often see error messages like this:

```
java.io.IOException: Cannot run program "/bin/bash" (in directory "c:\test\workspace\foo"):
  CreateProcess error=3, The system cannot find the path specified
```

This is because Jenkins is trying to call Windows API and execute "/bin/bash" without going through Cygwin path translation. Windows interprets `/bin/bash` as `c:\bin\bash.exe`, and unless that path exists, it will fail.

In this case, what's needed is to have Jenkins perform the Cygwin path translation without relying on Cygwin DLL. This is what Cygpath Plugin does; it checks if a Windows slave have Cygwin, and if you try to run executable that looks like Unix path name, it'll use Cygwin to translate that into its Windows path before calling Windows API.

Further Reading

Jenkins slaves running on Cygwin-enabled Windows is still susceptible to all the other problems Windows slaves face. See My software builds on my computer but not on Jenkins for the discussion of those, including desktop and network drive access.