

SNiFF+™

Version 3.2 for Unix and Windows

Integrating PVCS



TakeFive Software, Inc.

Cupertino, CA

E-mail: info@takefive.com

TakeFive Software GmbH

5020 Salzburg, Austria

E-mail: info@takefive.co.at

Copyright

Copyright © 1992–1999 TakeFive Software Inc.

All rights reserved. TakeFive products contain trade secrets and confidential and proprietary information of TakeFive Software Inc. Use of this copyright notice is precautionary and does not imply publication or disclosure.

Parts of SNIFF+:

Copyright 1991, 1992, 1993, 1994 by Stichting Mathematisch Centrum,
Amsterdam, The Netherlands.

Portions copyright 1991-1997 Compuware Corporation.

Trademarks

SNIFF+ is a trademark of TakeFive Software Inc.

Other brand or product names are trademarks or registered trademarks of their respective holders.

Credits

The first version of Sniff was developed at the Informatics Laboratory of the Union Bank of Switzerland. Its development was considerably facilitated by the public domain application framework ET++.

Authors of the first version:

Walter Bischofberger (Sniff)

Erich Gamma (Sniffgdb)

Erich Gamma and André Weinand (ET++)

Table of Contents

Introduction	5
Version and Compatibilty information	5
Feedback.	5
PVCS Version Manager	7
PVCS adapter documentation	8
Using the PVCS preferences dialog	14
How to Install an Earlier SNIFF+ 2.x PVCS Adapter in SNIFF+ 3.x19	

Integrating PVCS

Introduction

This paper describes how to integrate INTERSOLV's PVCS Version Manager with SNIFF+. PVCS Version Manager is a version control system for Windows, Unix, and OS/2. The PVCS adapter provided with this package can be used for both Unix and Windows platforms. It ensures a complete mapping of SNIFF+'s version control interface to PVCS Version Manager, and it is also the missing link between the cross-platform abilities of SNIFF+ as well as PVCS. Hence, it would be possible, for instance, to create a repository on platform A, check out a file on platform B, switch back to platform A, and check the file in there. On both platforms, you would issue the corresponding commands from SNIFF+, with PVCS as the underlying version control tool.

The PVCS Adapter is included in the SNIFF+ 3.x installation.

Assumptions made in this paper

It is assumed that the reader is familiar with the basic concepts of both tools used in this integration.

Version and Compatibility information

- Adapter version: **1.8 b10**
- Tested PVCS version: **5.3.x, 6.0.x, 6.5**
- SNIFF+ version: **3.0.2 or higher**
- Operating System: **Windows NT 4.0**

Feedback

Your feedback is always welcome. If you have any questions or comments, concerning SNIFF+'s integration with PVCS Version Manager please send an e-mail to the addresses listed below.

When asking for support, please describe your project settings, the PVCS settings and customization of the python adaptor. Please provide the used PVCS configuration files as well.

TakeFive Support

Europe:

sniff-support@takefive.co.at

USA:

sniff-support@takefive.com

Other Useful links

SNiFF+ web pages:

- SNiFF+ Users Mailing List

<http://www.takefive.com/support/sniff-list.html>

- SNiFF+ Users Mailing List Archive

<http://www.takefive.com/sniff-list>

- Frequently Asked Questions

<http://www.takefive.com/support/faq.html>

- Customer Newsletter

http://www.takefive.com/news/customer_newsletter.htm

PVCS Version Manager

The basic features of this version control system are:

Member storage

PVCS simplifies access to project components: code, data files, documentation, object modules, and other binary files. Developers and administrators need not remember where each module is stored. PVCS locates files or projects even across system boundaries. High-level support for module sharing makes PVCS essential on heterogeneous networks.

Access control

PVCS offers precise control of file locking on the LAN to prevent conflicting updates of shared files. Users can be assigned to groups that have specific file access capabilities.

Parallel development

PVCS lets developers create multiple logical views of the same physical modules, so they can create, maintain, and test special versions of an application. Unlimited branching offers flexible, yet safe, parallel development and makes alternative versions easy to maintain. A merge facility resolves conflicts as you merge multiple development efforts to produce the application.

Furthermore, PVCS Version Manager comes with a graphical user interface on all platforms, however all commands can also be called from the command line.

PVCS adapter documentation

Feature overview

This version of the PVCS adaptor provides the following key features:

- Full access to all basic version control functions
 - Version of files, History, Configurations, Branches
- Full support of PVCS configuration files (see [PVCS and adapter configuration files — page 12](#) for details)
- Support for PVCS keyword expansion
- Full support for the PVCS archive suffix rules
- Support for setting a version label to a single file or a group of files (including SNiFF+ change sets and floating labels)
- Possibility to move a label from one revision to another revision for one file
- Automatic enabling/disabling of the PVCS custom menu in the SNiFF+ Project Editor (see [Note about automatic PVCS custom menu — page 10](#) for details)
- Automatic reread of the configuration files, if there is any change during runtime
- Possibility to run the PVCS command remotely on a UNIX box from a Windows NT machine.

Limitations and known issues of the current PVCS adapter

Comments and descriptions passed on to an archive during Checkin must not be seeded with quotes (single, double, backward,...)

The previous PVCS adapter used RCS type extensions for repository files, e.g. the file name had a `*.cpp,v` type of extension. The current adapter uses a PVCS configuration file, where you can customize the used archive suffixes. The PVCS adaptor uses this configuration file as well to calculate the archive file extension. The archive suffix rules can be customized using the PVCS preferences dialog (see [Using the PVCS preferences dialog — page 14](#) for details).

Multiple PVCS archive suffix metarules (include a '?' inside the rule) are currently not supported, because PVCS seems to have a problem itself with multiple metarules. TakeFive will clear this issue as soon as possible.

The adaptor hasn't been currently tested on a Windows / Unix environment under cross platform development circumstances.

PVCS access lists, access database and the promotion model is currently not supported by this version of the adaptor.

If the adaptor is used on a legacy repository, it can be, that the repository contains labels with a space character inside. This labels are not supported by SNIFF+. The adaptor replace the space character by a underscore ('_') during reading the symbol list. Using the converted label name from inside SNIFF+ can lead to errors. Instead of using the symbol name directly, use the corresponding revision number to access the version of the label.

Note

Windows 95/98: If you want to use the PVCS integration on the operating systems Windows 95 or Windows 98 you **must** set the environment variable `USERNAME`. This variable must contain your login username, otherwise, there is no guarantee that the integration is working correctly.

The default behavior is that the adapter uses the default PVCS archive filename extensions (3rd position is replaced with a 'v' (?.? = ??v____)). If you want to customize the extensions to the one you used directly with PVCS, resetup all archive suffix rules from PVCS in the PVCS Preferences Dialog under the tab sheet "Suffix Rules" (see [Using the PVCS preferences dialog — page 14](#) for details). The changes are immediatly available.

Note

PVCS allows to specify suffix rules like `'.cpp -> .cpp_v'`. PVCS doesn't report any error or warning using this suffix rule, but the resulted archive is miscalculated (PVCS creates a archive extension `'.cpp'` instead the expected extension `'.cpp_v'`). To workaround this problem, use the suffix rule `'.cpp -> +_v'` instead.

Note about the remote calling possiblities inside the PVCS adapter:

The PVCS adapter uses a generic calling interface to the PVCS command line commands. Which real commands are used are defined in the two files `'PvcsLocal.py'` and `'PvcsRemote.py'`. Which interface is used will be controlled by the environment variable `'SNIFF_CMVC_USE_CALL_INTERFACE'`. If this variable is set to the value `'0'`, then the PVCS adapter uses the locally installed PVCS commands. If this variable has the value `'1'`, then the PVCS adapter calls the PVCS commands remotely on another machine. The name of this machine is given by the environment variable `'PVCS_REMOTE_HOST'`. To run the commands correctly on the remote machine, the adapter needs to know which Windows path correspond to which Unix path. This mapping has to be set in the Python dictionary `'PathMap'` inside the file `'$SNIFF_DIR/lib/python/Sniff/Pvcs/PvcsRemote.py'`.

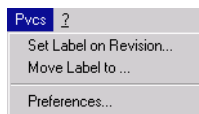
For example:

```
PathMap = {  
    "G:":"/Project/Repository",  
    "H:":"/Project/WorkArea"  
}
```

At least the drive letter including the colon has to be mapped to the correct corresponding Unix path. If the user name on the remote machine differs from the user name on your local machine, then set the correct remote user name using the environment variable 'SNIFF_CMVC_REMOTE_USER'.

Note about automatic PVCS custom menu

The current PVCS adapter version generates the necessary custom menu to access the dialogs for setting a version label and the PVCS preferences dialog automatically. By default, SNIFF+ looks for all enabled custom menus in the file `$SNIFF_DIR/config/SiteMenus.sniff` and merge these menus to the file `$SNIFF_DIR/Profiles/$USERNAME/UserMenus.sniff` (on Windows) or `$HOME/.sniffrc/UserMenus.sniff` (on Unix) or if the environment variable `$SNIFF_RESOURCE_DIR` is set, to `$SNIFF_RESOURCE_DIR/UserMenus.sniff`. Already existing custom menus on a user level will not be deleted and not overwritten. As a last step, SNIFF+ copies the contents of the file `$SNIFF_DIR/config/PvcsMenu.sniff` to the end of the UserMenus file. The resulting menu looks like the following:



If you want to customize this menu, you should make the changes in the file `$SNIFF_DIR/config/PvcsMenu.sniff`. Before restarting SNIFF+ to apply the changes, you **MUST** delete the old menu manually from the UserMenus file. In case you used custom menus before applying this version of the PVCS adapter, you should create a backup of your custom menus and you should check the correctness of your custom menus after the first start of SNIFF+ with this version of the adapter.

If you don't want to have the automatic merge at all or you want to merge it to the SiteMenus file, you can control the behavior of the automatic menu merge with setting two variables inside the responsible python file `$SNIFF_DIR/config/editor/InitPVCSCustomMenu.py`. The variable `forceUserLevel` (line 12 disable User- and enable SiteLevel; line 13 enable User- and disable SiteLevel) controls the destination of the custom menu. If `forceUserLevel` is set to "0", the custom menu will be written to the SiteMenus file. If `forceUserLevel` is set to "1", the custom menu will be written to UserMenus file (this is the default). If you want to switch off the automatic merge of the custom menu, then set the variable `enableAutoMenu` to "0" (line 16). Only if the variable `enableAutoMenu` is set to "1" (line 17, this is the default) the python script will try to merge the menus.

The generated custom menu contains the following actions:

- **Set Label on Revision...**
 - Calls the set label dialog and provides the possibility to set a fixed or a floating label and a fixed or a floating SNIFF+ changeset. See [Using the Set Label on Revision dialog — page 16](#) for further details
- **Move Label to...**
 - Moves a selected label in the history pane of the project editor to a new revision. If there is more than one label assigned to the corresponding revision, select the correct label in the upcoming dialog. Floating labels or changesets become fixed after a move operation
- **Preferences...**
 - Opens the PVCS integration preferences dialog. See [Using the PVCS preferences dialog — page 14](#) for further details.

Setting up projects with SNIFF+ and PVCS

If you want to create a new project, choose **Project > New Project... > With defaults...** from SNIFF+'s Launch Pad. Switch to the **Version Control System** node of the Project Attributes Dialog and select **PVCS** as **VCS Tool**.

If you want to create a PVCS repository for an already existing SNIFF+ project, or if you want to set up a SNIFF+ project for an already existing PVCS repository, from SNIFF+'s Project Editor, choose **Project > Attributes of Project *projectname***. In the Project Attributes dialog, switch to the **Version Control System** node and select **PVCS** as **VCS Tool**. Now you have full access control of a PVCS repository from within SNIFF+. For instance, if you want to check in and check out files, or if you want to create a new branch version, just choose the right menus from SNIFF+.

The PVCS adapter has all version control features for which SNIFF+ provides an interface, e.g. all Configuration Manager menus (see [Feature overview — page 8](#) for more details). Furthermore, the adapter can also deal with all repository actions related to SNIFF+'s working environment concept. For further details about working environments, please refer to the SNIFF+ User's Guide.

Sometimes, however, you might want to use the graphical user frontend of PVCS together with SNIFF+, e.g. because you slowly want to migrate to the fully integrated development approach pursued by SNIFF+. In this case, you can extend the custom menu (file `$SNIFF_DIR/config/PvcsMenu.sniff`) and call PVCS commands or the PVCS graphical interface from SNIFF+. For further details on custom menus, please refer to the

SNiFF+ Reference Guide. We recommend, that you use the PVCS graphical user frontend only to check the correctness of the integration with SNiFF+ and for customizing the used PVCS master and/or project configuration file.

Note

Changes to the archive suffix rules should be done by using the PVCS preferences dialog which is supplied together with this version of the adapter. Otherwise, we can't make any guarantees for the correct handling of the archive suffix rules in all cases.

PVCS and adapter configuration files

PVCS uses configuration files to store all settings used to control the behavior of the PVCS commands. Multiple levels of configuration files are possible with PVCS up to the depth of 20. If you are using the graphical user interface of PVCS, you can set at least a master configuration file and a project configuration file. By default, PVCS only uses the master configuration file. If you have changed the configuration file setting for one PVCS project, this is your project configuration file. The PVCS adapter of SNiFF+ can handle multiple levels of configuration files as well and you can set an additional master configuration file and an additional project configuration file using the PVCS preferences dialog supplied with this version of the adapter. The PVCS adapter uses a fixed file to store all configuration setting necessary for SNiFF+:

```
$SNIFF_DIR/lib/python/Sniff/Pvcs/Pvcs4Sniff.cfg
```

This is the master configuration file of the PVCS adapter. These file contain all settings necessary for SNiFF+ and at least the default archive suffix rule ".? = ??v__".

Note

All archive suffix rules all archive suffix rules will be stored in this file if you are using the PVCS preferences dialog. That means, this file must be writeable for all users which should have the permission to change the archive suffix rules.

The configuration files will be read in the following order:

- the default PVCS master configuration file (**only PVCS 6.0**)
- all included configuration files
- the SNiFF+ master configuration file

The settings in these configuration files will be merged in an incremental process. That means, that a setting which is only present in the master configuration file will be applied as well as a setting in the SNiFF+ master configuration file. For all settings which are in more than one configuration file, only the setting in the last configuration file will be used. In this

case, the SNIFF+ master configuration file is the last instance for configuration settings. The settings in the following chapter shouldn't be changed or overwritten. The correct behavior of the adapter is only guaranteed with at least these settings.

Non Default Settings in the Configuration File

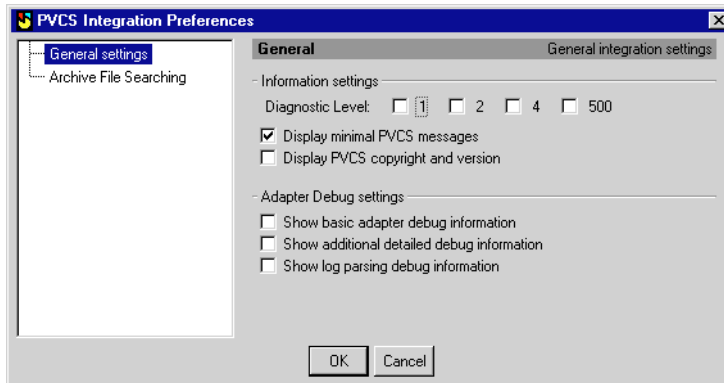
The following settings overwrite the default settings of PVCS:

- QUIET	... display only minimal messages
- NOSIGNON	... no copyright message of PVCS
- NOMULTILOCK	... NOMULTILOCK REVISION or USER is the default PVCS setting, Checkin with Concurrent Lock will set 'MULTILOCK REVISION
- IGNOREPATH	... use VCSDIR to specify repository directory
- NODELETEWORK WRITEPROTECT	... check in doesn't remove the workfiles
- EXPANDKEYWORDS TOUCH	... keyword expansion change the timestamp of the workfile
- LOGIN HOST, VCSID, UNKNOWN	... use the login sources HOST, VCSID and 'No Source' (UNKNOWN)
- NOSHARE	... set exclusive use of archives for all operations
- DATEFORMAT	... set DATEFORMAT SNIFF+ Configuration Manager compatible (yyyy/mm/dd hh:mm:ss)

All other configuration settings can be overwritten or deleted from the SNIFF+ configuration file if you are using the additional master and/or project configuration files. Otherwise deletion of configuration settings can disable some functionality.

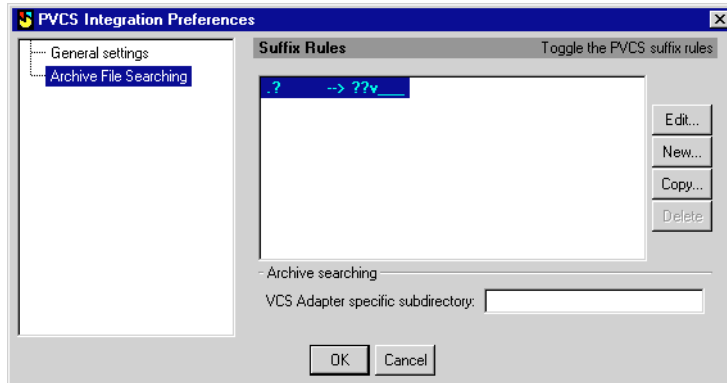
Using the PVCS preferences dialog

The PVCS preferences dialog is the place, where you can set the additional configuration files and where you can customize the used archive suffix rules. The following dialog will be shown if you select **PVCS > Preferences...**:



The **General settings** view shows you some settings for the information level of the PVCS commands. Please refer the PVCS documentation to the directives "DIAGNOSTIC", "QUIET/VERBOSE" and "SIGNON/NOSIGNON" for information about these settings. The second section of this page contains the settings for switching on or off the debug possibilities of the PVCS adapter itself. The basic adapter debug information includes information about general process control and the called PVCS commands. The detailed debug information contains additional information about adapter internal calls and all the processed data. This level can print out a lot of information. Please stay patient in this case until the log window scrolling is stopped. The log parser debug information shows additional information about the floating label handling for the SNIFF+ Project Editor History Pane.

The **Archive File Searching** view gives you the possibility to setup the used archive suffix rules to calculate the responsible archive file for a work file.



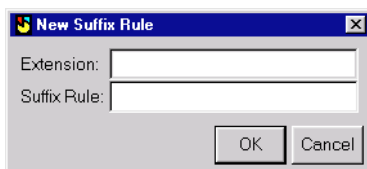
At least one rule **MUST** be in this list, the rule for all file extensions ".?" (per default ".? -> ??v____"). You can edit this rule, but you cannot delete this rule (all rules which you cannot delete are show in a red font). You can create new rules (button **New...**), you can modify existing rules (button **Edit...**) and you can copy an existing rule to create a new rule (button **Copy...**).

At least one rule **MUST** be in this list, the rule for all file extensions ".?" (per default ".? -> ??v____"). You can edit this rule, but you cannot delete this rule (all rules which you cannot delete are show in a red font). You can create new rules (button **New...**), you can modify existing rules (button **Edit...**) and you can copy an existing rule to create a new rule (button **Copy...**).

Note

Multiple PVCS archive suffix metarules (include a '?' inside the rule) are currently not supported, because PVCS seems to have a problem itself with multiple metarules. TakeFive will clear this issue as soon as possible.

The New Suffix Rule dialog:



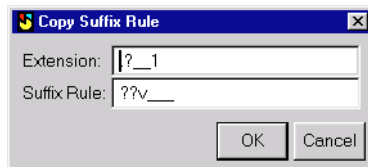
You have to fill out the field for the extension (the extension of the work file) and the field for the suffix rule (describe how we should calculate the archive extension).

The Edit Suffix Rule dialog:



The field extension and the field suffix rule are by default filled with the values of the selected suffix rule.

The Copy Suffix Rule dialog:

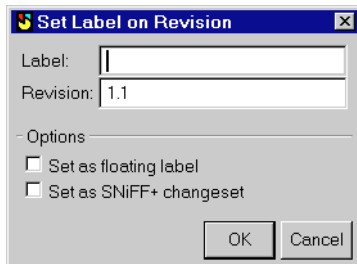


The Copy Suffix Rule dialog sets by default the fields for extension and suffix rule with the values of the selected suffix rule in the list and appends to the extension a unique number which makes the extension itself unique to avoid double extensions. You must change the extension to the one you want to add.

The field **VCS Adapter specific subdirectory** specifies an additional subdirectory which will be appended to the repository directory. The field contains by default the directory **Pvcs**. If you use a repository which was created by PVCS itself, then clear this field to find the existing archives.

Using the Set Label on Revision dialog

With the Set Label on Revision dialog it is possible to set a version label (PVCS fixed and floating labels) and/or a SNIFF+ change set to a specific version of a single file or to the tip revision (HEAD) of multiple selected files. Use the custom menu **PVCS > Set Label...** to show the following Set Label on Revision dialog.

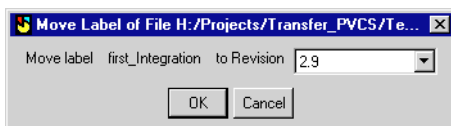


In the field **Label** you can specify the label name you want to set. The dialog excepts all label names which are valid for PVCS itself. The field **Revision** contains the revision number for which you want to set the label. The field is pre - filled with the number of the tip revision

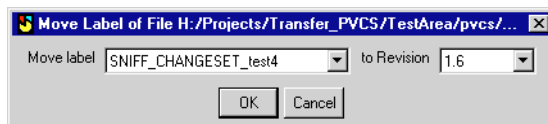
(HEAD) if you don't select a specific revision in the 'History Panel' of the SNIFF+ Project Editor, otherwise the selected version is there. If you select more than one file, everytime the string 'TIP' is presented. In this case, the field is not editable and the label will be set on the newest revisions of all selected files. By default, all labels are set as fixed labels. If you want to set a label as floating label or as SNIFF+ changeset, checkmark the corresponding checkbox.

Using the Move Label dialog

The 'Move Label' Dialog provides functionality to reassign a version label on a revision of one file to another revision on the same file. Multiple file operations are not allowed. To use the 'Move Label' Dialog, select the label you want to reassign in the history pane of the SNIFF+ Project Editor. Choose the menu item 'Move Label to ...' and you get the dialog



or if there are more than one label (not changesets !) assigned to the revision



In both cases select the revision to witch you want to reassign the selected label. If there are more than one label assigned to a revision, all label are listed as show in the second dialog in the most left combobox. Select there the right label to move again.

Note

SNIFF+ can handle only one changeset per revision. This is limitation of SNIFF+ and not of the underlying Version Control Tool. The 'Set Label'- and the 'Move Label' – Dialog shows a warning if someone tries to set more than one changeset per revision.

Note About Branching

When locking a non - tip (non HEAD) revision in PVCS, a check in is only allowed in a new branch. To identify the tip revision in the branch, the adapter implements a mechanism to calculate an automatic name for the branch. The user will be asked, if he/she accepts the name or not. If the user declines the automatic generated name, the only way to check in a locked non - tip revision is to check in with a new branch (checkmark **New Branch** in the Check in dialog).

Customizing the PVCS Adapter

The PVCS adapter is written in the Python scripting language. Customization of the adapter needs to be done in Python. Alternatively you could use a PVCS adapter from an earlier SNIFF+ version, see next section. For customizing the archive suffixes, use the earlier described PVCS Preferences dialog

Customization details:

- Refer to the Python home page
<http://www.python.org/>
for more details about the Python language.
- If necessary, switch on the different adapter debug levels inside the PVCS Preferences Dialog.
- Reloading the VCS adapter after every change restart SNIFF+

How to Install an Earlier SNIFF+ 2.x PVCS Adapter in SNIFF+ 3.x

Please use this PVCS Adapter written in Python if you didn't customize an earlier PVCS adapter in SNIFF+ 2.3.x or 2.4.

In case you heavily customized a previous PVCS adapter written in TCL, we recommend that you use your existing adapter. In this case you should follow these steps to install it in SNIFF+ 3.x under Windows:

- SNIFF+ 2.x and SNIFF+ 3.x must be installed on the same system.
 - SNIFF+ specific environment variables and PATH should be set for SNIFF+ 3.x.
- Run SNIFF+ 2.x and SNIFF+ 3.x simultaneously. To run SNIFF+ 2.x:
 - Open a DOS shell
 - Change to the SNIFF+ 2.x directory.
 - Run the `%SNIFF_DIR%\SetEnvVars.bat` command to set the environment variables.
 - Set the PATH environment variable from the DOS prompt to reflect the SNIFF+ 2.x bin and mks path, e.g.:
 - `SET PATH=C:\sniff24\bin;C:\sniff24\mks\mks-5.2\mksnt;%PATH%`
 - Start SNIFF+ from the DOS prompt.
- Open **Preferences...** from any SNIFF+ tool, choose the **Site Level** tab, choose the **Version Control System** node and select **PVCS** in SNIFF+ 2.x and also in SNIFF+ 3.x.
- Do a manual copy and paste operation of all edit fields. Example:
 - Change the Lockers field from

```
python -c #A
to
%d\\bin\\tclsh %d\\vcs\\pvcs.tcl pvcs_lockers \"%A\"
```

Note

No line breaks are allowed, also no line feed after the first line! Copy everything to one line. If there is more than one line, e.g. a shell script, you have to create a shell script file and call the script as a one line command.

- In SNIFF+ 3.x, create a directory `%SNIFF_DIR%\vcs`.

-
- Copy file `%SNIFF_DIR%\vcs\pvcs.tcl` from SNIFF+ 2.x to the `%SNIFF_DIR%\vcs` directory of SNIFF+ 3.x.
 - In case you get the error message can't read "gPathVar" in the log window when using the PVCS adapter, you should change the case of environment variable `Path=` from mixed case to all upper case, e.g. `PATH=`.

A similar procedure also works for a PVCS Adapter under Unix.