

# CVSspam Documentation

For the latest version, visit <http://www.badgers-in-foil.co.uk/projects/cvssпам/>.

## 1. Installation

### 1.1. Configure CVS

To install CVSspam you'll need to alter the repository's configuration files.

Check out your repository's CVSROOT.

```
...set $CVSROOT to point at your repository...
$ cvs checkout CVSROOT
$ cd CVSROOT
$ ls
CVS          commitinfo  cvswrappers  loginfo  notify  taginfo
checkoutlist config          editinfo     modules  rcsinfo verifymsg
```

Alter `commitinfo` to call the CVSspam script that records the directories that have been committed:

```
# The "commitinfo" file is used to control pre-commit checks.
# The filter on the right is invoked with the repository and a list
# of files to check.  A non-zero exit of the filter program will
# cause the commit to be aborted.
#
# The first entry on a line is a regular expression which is tested
# against the directory that the change is being committed to, relative
# to the $CVSROOT.  For the first match that is found, then the remainder
# of the line is the name of the filter to run.
#
# If the repository name does not match any of the regular expressions in this
# file, the "DEFAULT" line is used, if it is specified.
#
# If the name "ALL" appears as a regular expression it is always used
# in addition to the first matching regex or "DEFAULT".

^myproject /path/to/record_lastdir.rb
```

Now you need to alter `loginfo` to record the log entry made by the user (and send off the email):

```
# The "loginfo" file controls where "cvs commit" log information
# is sent.  The first entry on a line is a regular expression which must match
# the directory that the change is being made to, relative to the
# $CVSROOT.  If a match is found, then the remainder of the line is a filter
```

```

# program that should expect log information on its standard input.
#
# If the repository name does not match any of the regular expressions in this
# file, the "DEFAULT" line is used, if it is specified.
#
# If the name ALL appears as a regular expression it is always used
# in addition to the first matching regex or DEFAULT.
#
# You may specify a format string as part of the
# filter. The string is composed of a '%' followed
# by a single format character, or followed by a set of format
# characters surrounded by '{' and '}' as separators. The format
# characters are:
#
#   s = file name
#   V = old version number (pre-checkin)
#   v = new version number (post-checkin)
#
# For example:
#DEFAULT (echo ""; id; echo %s; date; cat) >> $CVSROOT/CVSROOT/commitlog
# or
#DEFAULT (echo ""; id; echo %{sVv}; date; cat) >> $CVSROOT/CVSROOT/commitlog

^myproject /path/to/collect_diffs.rb --to me@somewhere.invalid %{sVv}

```

**Note:** The expression you use to select the project (the first thing on the line) must be the same in commitinfo and logininfo.

Commit your changes to these files. You should see a message from CVS like 'rebuilding administrative database'. You are now be ready to test the setup.

Checkout a copy of *myproject* and commit a change. An email should be sent to the address you specified.

## 1.2. Installing CVSspam files

The CVSspam scripts may be located anywhere in the CVS server's filesystem.

It's common to place these files inside the repository's CVSROOT, and this can be the only option if CVS is the only way you have to access a remote server. To do this, you need a checked-out copy of the CVSROOT, as described above. Place `record_last_dir.rb`, `collect_diffs.rb` and `cvssпам.rb` into this directory.

Add these three filenames into CVSROOT/checkoutlist

```
# The "checkoutlist" file is used to support additional version controlled
# administrative files in $CVSROOT/CVSROOT, such as template files.
#
# The first entry on a line is a filename which will be checked out from
# the corresponding RCS file in the $CVSROOT/CVSROOT directory.
# The remainder of the line is an error message to use if the file cannot
# be checked out.
#
# File format:
#
#      [<whitespace><filename><whitespace><error message><end-of-line>
#
# comment lines begin with '#'
record_lastdir.rb
collect_diffs.rb
cvssпам.rb
```

**cv**s **add** the three scripts to the repository, then **cv**s **commit** them, and the modified checkoutlist.

In `commitinfo` and `loginfo` you can now refer to the scripts with  
`$CVSROOT/CVSROOT/record_lastdir.rb` and `$CVSROOT/CVSROOT/collect_diffs.rb`

## 1.3. Configuration File

You can specify CVSspam options in a configuration file. By default, the file  
`$CVSROOT/CVSROOT/cvssпам.conf` is used, though you can specify another with the `--config`  
option to `collect_diffs.rb`.

If you want to put your config into the repository, follow the instructions above for installing files into  
CVSROOT

To see the available options, see the example `cvssпам.conf` provided.

## 1.4. Debugging installation problems

1. No email coming from CVS commits, and no error messages on the command line

Did you specify the right email address?

Does the regular expression you specified in `commitinfo` and `loginfo` really match the project? Try  
changing the entry to something like

```
^myproject echo "Hello world"
```

When you commit a change to *myproject*, 'Hello world' should appear in your terminal. If it doesn't, check that expression on the left is correct.

Check that the CVS server corectly handles email. By default CVSspam invokes sendmail. Try running sendmail by hand on the CVS server machine

```
$ echo test | /usr/sbin/sendmail me@somewhere.invalid
```

2. Why do I see the message `No such file or directory cvs` on the console after committing?

The cvs executable is probably not in the default executable search path available to the CVSspam scripts. Tell them explicitly where to find cvs by setting the `$cvs_prog` option in the configuration file.

## 2. Configuration Options

### 2.1. Integration with Bug Tracking Software

CVSspam can generate simple links to web based bug tracking systems, currently supporting Bugzilla (<http://www.mozilla.org/projects/bugzilla/>) and JIRA (<http://www.atlassian.com/software/jira/>).

For bugzilla, when a CVS log comment contains text like **Fix for bug 123...**, the text "bug *nnn*" will become a hyperlink to that Bugzilla page in the generated email.

To enable, give your Bugzilla's URL in CVSspam's configuration file

```
$bugzillaURL = "http://bugzilla.mozilla.org/show_bug.cgi?id=%s"
```

The marker `%s` tells CVSspam where in the URL to put the bugId from the log message.

For JIRA, include the issueId in the log comment. JIRA issue Ids have a project name and issue number, seperated by a dash. For example JRA-1545.

To enable, give your JIRA installation's URL in CVSspam's configuration file

```
$jiraURL = "http://jira.atlassian.com/secure/ViewIssue.jspa?key=%s"
```

The marker `%s` tells CVSspam where in the URL to put the issue Id from the log message.

For systems that like to talk about *tickets*, CVSspam will make links from text in the log comment that looks like “bug *nnn*” (where *nnn* is a number). For instance with RT (<http://fsck.com/projects/rt/>), supply the location of `Display.html`

```
$ticketURL = "http://localhost/rt2/Ticket/Display.html?id=%s"
```

## 2.2. Integration with ViewCVS/CVSweb/Chora

If you have ViewCVS (<http://viewcvs.sourceforge.net/>), CVSweb (<http://www.freebsd.org/projects/cvsweb.html>) or Chora (<http://www.horde.org/chora/>) web-access to your repository, CVSspam can generate links to it in the emails. Links the file before and after the commit are very useful for images, as only changes to binary text files are mailed. You’ll get a link to the side-by-side view of the changes as well.

To enable ViewCVS support, specify the URL of the top-level ViewCVS directory in `cvssпам.conf`.

```
$viewcvsURL = "http://localhost/cgi-bin/viewcvs.cgi/"
```

For CVSweb, specify the URL of `cvsweb.cgi`,

```
$cvswebURL = "http://localhost/cgi/cvsweb.cgi/"
```

For Chora, specify the URL of the directory containing `cvs.php`,

```
$choraURL = "http://localhost/hord/chora/"
```

## 2.3. Global Email Addresses

Recipient email addresses can be put in the configuration file as well as in each `loginfo` entry. For example,

```
addRecipient "code-review@somewhere.invalid"
addRecipient "project-owner@somewhere.invalid"
```