## Copyright Assignment

By opening a pull request to https://github.com/presidentbeef/brakeman,

you agree to assign all rights to the code to Synopsys, Inc. under the [Brakeman Public Use License](LICENSE.md).

## Submitting a Pull Request

Pull requests are welcome!

Please follow the typical GitHub flow:

\* Fork Brakeman

\* Clone locally `git clone your\_new\_fork`

\* Create a new branch `git checkout -b fix\_some\_broken\_stuff`

\* Add new tests

\* Make fixes, follow coding conventions of project

\* Run tests with `ruby test/test.rb` or just `rake`

\* Push your changes `git push origin fix\_some\_broken\_stuff`

\* Go to \*your\* fork, click "Submit pull request"

\* Provide a description of the bug and fix

\* Submit!

### Code Conventions

These are some code conventions to follow so your code fits into the rest of Brakeman.

\* Must use typical Ruby 2 space indentation

\* Must work with Ruby 2.3.0

\* Prefer to wrap lines near 80 characters but it's not a hard rule

### Preparing Tests

Tests are very important to ensure fixes actually work, and to make it obvious what your changes are supposed to fix. They also protect against breaking features in the future.

#### Run Tests

To run Brakeman tests:

ruby test/test.rb

or

rake

To check test coverage, install `simplecov` before running tests. Then open `coverage/index.html` in a browser. For a correct report, run the tests from the root directory.

#### Add a Test Case

Brakeman has several Rails applications in the `test/apps` directory. Choose the one that best matches your situation and modify it to reproduce the issue. It is preferable to modify the application in such a way that the fewest existing tests are broken. In particular, the tests for "expected number of reported warnings" will probably change, but no other tests should. Unless the tests or expected behavior are broken.

In the `test/tests` directory, each application has its own set of tests. Most of these consist of `assert\_warning` or `assert\_no\_warning`, which test for warnings generated by Brakeman.

When adding a test for a false positive, use `assert\_no\_warning` so the expected behavior is clear.

#### Generating Tests

Writing the `assert\_warning` tests can be tedious, especially in bulk. There is a tool which will convert Brakeman reports to tests in `tests/to\_test.rb`. This file takes exactly the same options as Brakeman. This makes it easy to generate a smaller set of tests (as opposed to tests for every Brakeman warning, which probably already have tests).

Example:

```

ruby to\_test.rb apps/rails2 -t Execute

```

will generate some boilerplate and then a set of methods:

```ruby

#...

def test\_command\_injection\_1

assert\_warning :type => :warning,

:warning\_type => "Command Injection",

:line => 34,

:message => /^Possible\ command\ injection/,

:confidence => 0,

:file => /home\_controller\.rb/

end

def test\_command\_injection\_2

assert\_warning :type => :warning,

:warning\_type => "Command Injection",

:line => 36,

:message => /^Possible\ command\ injection/,

:confidence => 0,

:file => /home\_controller\.rb/

end

#...

```

The boilerplate is unnecessary unless you are adding a whole new test application.

When adding a single test or set of tests, copy the tests from here, change the names to something descriptive, and you are done!

Note that when adding an `assert\_no\_warning` test for false positives, you can still generate the test with the false positive, then change the assertion.