How to Submit Patches to the libseccomp Project

https://github.com/seccomp/libseccomp-golang

This document is intended to act as a guide to help you contribute to the libseccomp project. It is not perfect, and there will always be exceptions to the rules described here, but by following the instructions below you should have a much easier time getting your work merged with the upstream project.

Test Your Code Using Existing Tests

There are two possible tests you can run to verify your code. The first test is used to check the formatting and coding style of your changes, you can run the test with the following command:

```
# make check-syntax
```

... if there are any problems with your changes a diff/patch will be shown which indicates the problems and how to fix them.

The second possible test is used to ensure the sanity of your code changes and to test these changes against the included tests. You can run the test with the following command:

```
# make check
```

... if there are any faults or errors they will be displayed.

Add New Tests for New Functionality

Any submissions which add functionality, or significantly change the existing code, should include additional tests to verify the proper operation of the proposed changes.

Explain Your Work

At the top of every patch you should include a description of the problem you are trying to solve, how you solved it, and why you chose the solution you implemented. If you are submitting a bug fix, it is also incredibly helpful if you can describe/include a reproducer for the problem in the description as well as instructions on how to test for the bug and verify that it has been fixed.

Sign Your Work

The sign-off is a simple line at the end of the patch description, which certifies that you wrote it or otherwise have the right to pass it on as an open-source patch. The "Developer's Certificate of Origin" pledge is taken from the Linux Kernel and the rules are pretty simple:

```
Developer's Certificate of Origin 1.1

By making a contribution to this project, I certify that:

(a) The contribution was created in whole or in part by me and I
    have the right to submit it under the open source license
    indicated in the file; or

(b) The contribution is based upon previous work that, to the best
    of my knowledge, is covered under an appropriate open source
    license and I have the right under that license to submit that
    work with modifications, whether created in whole or in part
    by me, under the same open source license (unless I am
    permitted to submit under a different license), as indicated
    in the file; or

(c) The contribution was provided directly to me by some other
    person who certified (a), (b) or (c) and I have not modified
    it.

(d) I understand and agree that this project and the contribution
    are public and that a record of the contribution (including all
    personal information I submit with it, including my sign-off) is
    maintained indefinitely and may be redistributed consistent with
    this project or the open source license(s) involved.
```

... then you just add a line to the bottom of your patch description, with your real name, saying:

```
Signed-off-by: Random J Developer <random@developer.example.org>
```

You can add this to your commit description in `git` with `git commit -s`

Post Your Patches Upstream

The libseccomp project accepts both GitHub pull requests and patches sent via the mailing list. GitHub pull requests are preferred. This sections below explain how to contribute via either method. Please read each step and perform all steps that apply to your chosen contribution method.

Submitting via Email

Depending on how you decided to work with the libseccomp code base and what tools you are using there are different ways to generate your patch(es). However, regardless of what tools you use, you should always generate your patches using the "unified" diff/patch format and the patches should always apply to the libseccomp source tree using the following command from the top directory of the libseccomp sources:

```
# patch -p1 < changes.patch
```

If you are not using git, stacked git (stgit), or some other tool which can generate patch files for you automatically, you may find the following command helpful in generating patches, where "libseccomp.orig/" is the unmodified source code directory and "libseccomp/" is the source code directory with your changes:

```
# diff -purN libseccomp.orig/ libseccomp/
```

When in doubt please generate your patch and try applying it to an unmodified copy of the libseccomp sources; if it fails for you, it will fail for the rest of us.

Finally, you will need to email your patches to the mailing list so they can be reviewed and potentially merged into the main libseccomp repository. When sending patches to the mailing list it is important to send your email in text form, no HTML mail please, and ensure that your email client does not mangle your patches. It should be possible to save your raw email to disk and apply it directly to the libseccomp source code; if that fails then you likely have a problem with your email client. When in doubt try a test first by sending yourself an email with your patch and attempting to apply the emailed patch to the libseccomp repository; if it fails for you, it will fail for the rest of us trying to test your patch and include it in the main libseccomp repository.

Submitting via GitHub

See this guide if you've never done this before.