# Contributing to Metasploit

Thank you for your interest in making Metasploit -- and therefore, the

world -- a better place! Before you get started, please review our [Code of Conduct](https://github.com/rapid7/metasploit-framework/wiki/Code-Of-Conduct). This helps us ensure our community is positive and supportive for everyone involved.

## Code Free Contributions

Before we get into the details of contributing code, you should know there are multiple ways you can add to Metasploit without any coding experience:

- You can [submit bugs and feature requests](https://github.com/rapid7/metasploit-framework/issues/new) with detailed information about your issue or idea:

- If you'd like to propose a feature, describe what you'd like to see. Mock ups of console views would be great.

- If you're reporting a bug, please be sure to include the expected behaviour, the observed behaviour, and steps to reproduce the problem. Resource scripts, console copy-pastes, and any background on the environment you encountered the bug in would be appreciated. More information can be found [below](#bug-reports).

- [Help fellow users with open issues]. This can require technical knowledge, but you can also get involved in conversations about bug reports and feature requests. This is a great way to get involved without getting too overwhelmed!

- [Help fellow committers test recently submitted pull requests](https://github.com/rapid7/metasploit-framework/pulls). Again this can require some technical skill, but by pulling down a pull request and testing it, you can help ensure our new code contributions for stability and quality.

- [Report a security vulnerability in Metasploit itself] to Rapid7. If you see something you think makes Metasploit vulnerable to an attack, let us know!

- [Add module documentation](https://github.com/rapid7/metasploit-framework/wiki/Generating-Module-Documentation). New documentation is always needed and cleaning up existing documents is just as important! If you're a non-native english speaker, you can help by replacing any ambiguous idioms, metaphors, or unclear language that might make our documentation hard to understand.

## Code Contributions

For those of you who are looking to add code to Metasploit, your first step is to set up a [development environment]. Once that's done, we recommend beginners start by adding a [proof-of-concept exploit from ExploitDB,](https://www.exploit-db.com/search?verified=true&hasapp=true&nomsf=true) as a new module to the Metasploit framework. These exploits have been verified as recreatable and their ExploitDB page includes a copy of the exploitable software. This makes testing your module locally much simpler, and most importantly the exploits don't have an existing Metasploit implementation. ExploitDB can be slow to update however, so please double check that there isn't an existing module before beginning development! If you're certain the exploit you've chosen isn't already in Metasploit, read our [writing an exploit guide](https://github.com/rapid7/metasploit-framework/wiki/How-to-get-started-with-writing-an-exploit). It will help you to get started and avoid some common mistakes.

Once you have finished your new module and tested it locally to ensure it's working as expected, check out our [guide for accepting modules](https://github.com/rapid7/metasploit-framework/wiki/Guidelines-for-Accepting-Modules-and-Enhancements#module-additions). This will give you a good idea of how to clean up your code so that it's likely to get accepted.

Finally, follow our short list of do's and don'ts below to make sure your valuable contributions actually make it into Metasploit's master branch! We try to consider all our pull requests fairly and in detail, but if you do not follow these rules, your contribution

will be closed. We need to ensure the code we're adding to master is written to a high standard.

### Code Contribution Do's & Don'ts:

--

#### <u>Pull Requests</u>

\*\*Pull request [PR#9966] is a good example to follow.\*\*

\* \*\*Do\*\* create a [topic branch] to work on instead of working directly on `master`. This helps to:

\* Protect the process.

\* Ensures users are aware of commits on the branch being considered for merge.

\* Allows for a location for more commits to be offered without mingling with other contributor changes.

\* Allows contributors to make progress while a PR is still being reviewed.

\* \*\*Do\*\* follow the [50/72 rule] for Git commit messages.

\* \*\*Do\*\* write "WIP" on your PR and/or open a [draft PR] if submitting \*\*working\*\* yet unfinished code.

\* \*\*Do\*\* target your pull request to the \*\*master branch\*\*.

\* \*\*Do\*\* specify a descriptive title to make searching for your pull request easier.

\* \*\*Do\*\* include [console output], especially for effects that can be witnessed in the `msfconsole`.

\* \*\*Do\*\* list [verification steps] so your code is testable.

\* \*\*Do\*\* [reference associated issues] in your pull request description.

\* \*\*Don't\*\* leave your pull request description blank.

\* \*\*Don't\*\* abandon your pull request. Being responsive helps us land your code faster.

\* \*\*Don't\*\* post questions in older closed PRs.

#### <u>New Modules</u>

\* \*\*Do\*\* license your code as BSD 3-clause, BSD 2-clause, or MIT.

\* \*\*Do\*\* stick to the [Ruby style guide] and use [Rubocop] to find common style issues.

\* \*\*Do\*\* set up `msftidy` to fix any errors or warnings that come up as a [pre-commit hook].

\* \*\*Do\*\* use the many module mixin [API]s.

\* \*\*Do\*\* include instructions on how to setup the vulnerable environment or software.

\* \*\*Do\*\* include [Module Documentation] showing sample run-throughs.

\* \*\*Don't\*\* include more than one module per pull request.

\* \*\*Don't\*\* submit new [scripts]. Scripts are shipped as examples for automating local tasks, and anything "serious" can be done with post modules and local exploits.

#### <u>Library Code</u>

\* \*\*Do\*\* write [RSpec] tests - even the smallest change in a library can break existing code.

\* \*\*Do\*\* follow [Better Specs] - it's like the style guide for specs.

\* \*\*Do\*\* write [YARD] documentation - this makes it easier for people to use your code.

\* \*\*Don't\*\* fix a lot of things in one pull request. Small fixes are easier to validate.

#### <u>Bug Fixes</u>

\* \*\*Do\*\* include reproduction steps in the form of verification steps.

\* \*\*Do\*\* link to any corresponding [Issues] in the format of `See #1234` in your commit description.

## Bug Reports

Please report vulnerabilities in Rapid7 software directly to security@rapid7.com. For more on our disclosure policy and Rapid7's approach to coordinated disclosure, [head over here](https://www.rapid7.com/security).

When reporting Metasploit issues:

\* \*\*Do\*\* write a detailed description of your bug and use a descriptive title.

\* \*\*Do\*\* include reproduction steps, stack traces, and anything that might help us fix your bug.

\* \*\*Don't\*\* file duplicate reports; search for your bug before filing a new report.

\* \*\*Don't\*\* attempt to report issues on a closed PR.

If you need some more guidance, talk to the main body of open source contributors over on our

[Metasploit Slack] or [#metasploit on Freenode IRC].

Finally, \*\*thank you\*\* for taking the few moments to read this far! You're already way ahead of the

curve, so keep it up!

[Code of Conduct]:https://github.com/rapid7/metasploit-framework/wiki/CODE\_OF\_CONDUCT.md

[Submit bugs and feature requests]:http://r-7.co/MSF-BUGv1

[Help fellow users with open issues]:https://github.com/rapid7/metasploit-framework/issues

[help fellow committers test recently submitted pull requests]:https://github.com/rapid7/metasploit-framework/pulls

[Report a security vulnerability in Metasploit itself]:https://www.rapid7.com/disclosure.jsp

[development environment]:http://r-7.co/MSF-DEV

[proof-of-concept exploits]:https://www.exploit-db.com/search?verified=true&hasapp=true&nomsf=true

[Ruby style guide]:https://github.com/bbatsov/ruby-style-guide

[Rubocop]:https://rubygems.org/search?query=rubocop

[50/72 rule]:http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html

[topic branch]:http://git-scm.com/book/en/Git-Branching-Branching-Workflows#Topic-Branches

[draft PR]:https://help.github.com/en/articles/about-pull-requests#draft-pull-requests

[console output]:https://help.github.com/articles/github-flavored-markdown#fenced-code-blocks

[verification steps]:https://help.github.com/articles/writing-on-github#task-lists

[reference associated issues]:https://github.com/blog/1506-closing-issues-via-pull-requests

[PR#9966]:https://github.com/rapid7/metasploit-framework/pull/9966

[pre-commit hook]:https://github.com/rapid7/metasploit-framework/blob/master/tools/dev/pre-commit-hook.rb

[API]:https://rapid7.github.io/metasploit-framework/api

[Module Documentation]:https://github.com/rapid7/metasploit-framework/wiki/Generating-Module-Documentation

[scripts]:https://github.com/rapid7/metasploit-framework/tree/master/scripts

[RSpec]:http://rspec.info

[Better Specs]:http://www.betterspecs.org/

[YARD]:http://yardoc.org

[Issues]:https://github.com/rapid7/metasploit-framework/issues

[Metasploit Slack]:https://www.metasploit.com/slack

[#metasploit on Freenode IRC]:http://webchat.freenode.net/?channels=%23metasploit&uio=d4