Writing FreeBSD Problem Reports

Dag-Erling Smørgrav
Mark Linimon

Revision: 45144

FreeBSD is a registered trademark of the FreeBSD Foundation.

IBM, AIX, OS/2, PowerPC, PS/2, S/390, and ThinkPad are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Intel, Celeron, Centrino, Core, EtherExpress, i386, i486, Itanium, Pentium, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

SPARC, SPARC64, and UltraSPARC are trademarks of SPARC International, Inc in the United States and other countries. SPARC International, Inc owns all of the SPARC trademarks and under licensing agreements allows the proper use of these trademarks by its members.

Sun, Sun Microsystems, Java, Java Virtual Machine, JDK, JRE, JSP, JVM, Netra, OpenJDK, Solaris, StarOffice, SunOS and VirtualBox are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the "™" or the "®" symbol.

2014-06-29 by eadler.

Abstract

This article describes how to best formulate and submit a problem report to the FreeBSD Project.

Table of Contents

1. Introduction 2
2. When to Submit a Problem Report 2
3. Preparations 4
4. Writing the Problem Report 5
5. Follow-up 14
6. If There Are Problems 15
7. Further Reading 15
Index 16

1. Introduction

One of the most frustrating experiences one can have as a software user is to submit a problem report only to have it summarily closed with a terse and unhelpful explanation like “not a bug” or “bogus PR”. Similarly, one of the most frustrating experiences as a software developer is to be flooded with problem reports that are not really problem reports but requests for support, or that contain little or no information about what the problem is and how to reproduce it.

This document attempts to describe how to write good problem reports. What, one asks, is a good problem report? Well, to go straight to the bottom line, a good problem report is one that can be analyzed and dealt with swiftly, to the mutual satisfaction of both user and developer.

Although the primary focus of this article is on FreeBSD problem reports, most of it should apply quite well to other software projects.

Note that this article is organized thematically, not chronologically. Read the entire document before submitting a problem report, rather than treating it as a step-by-step tutorial.

2. When to Submit a Problem Report

There are many types of problems, and not all of them should engender a problem report. Of course, nobody is perfect, and there will be times when what seems to be a bug in a program is, in fact, a misunderstanding of the syntax for a command or a typographical error in a configuration file (though that in itself may sometimes be indicative of poor documentation or poor error handling in the application). There are still many cases where submitting a problem report is clearly *not* the right course of action, and will only serve to frustrate both the submitter and the developers. Conversely, there are cases where it might be appropriate to submit a problem report about something else than a bug—an enhancement or a new feature, for instance.

So how does one determine what is a bug and what is not? As a simple rule of thumb, the problem is *not* a bug if it can be expressed as a question (usually of the form “How do I do

X?" or "Where can I find Y?"). It is not always quite so black and white, but the question rule covers a large majority of cases. When looking for an answer, consider posing the question to the FreeBSD general questions mailing list.

Some cases where it may be appropriate to submit a problem report about something that is not a bug are:

- Notification of updates to externally maintained software (such as ports or software in the contrib/ directory).

  For unmaintained ports (`MAINTAINER` contains `ports@FreeBSD.org`), such update notifications might get picked up by an interested committer, or you might be asked to provide a patch to update the port; providing it upfront will greatly improve your chances that the port will get updated in a timely manner.

  If the port is maintained, PRs announcing new upstream releases are usually not very useful since they generate supplementary work for the committers, and the maintainer likely knows already there is a new version, they have probably worked with the developers on it, they are probably testing to see there is no regression, etc.

  In either case, following the process described in Porter's Handbook will yield the best results. (You might also wish to read Contributing to the FreeBSD Ports Collection.)

A bug that cannot be reproduced can rarely be fixed. If the bug only occurred once and you can not reproduce it, and it does not seem to happen to anybody else, chances are none of the developers will be able to reproduce it or figure out what is wrong. That does not mean it did not happen, but it does mean that the chances of your problem report ever leading to a bug fix are very slim. To make matters worse, often these kinds of bugs are actually caused by failing hard drives or overheating processors — you should always try to rule out these causes, whenever possible, before submitting a PR.

Next, to decide to whom you should file your problem report, you need to understand that the software that makes up FreeBSD is composed of several different elements:

- Code in the base system that is written and maintained by FreeBSD contributors, such as the kernel, the C library, and the device drivers (categorized as `kern`); the binary utilities (`bin`); the manual pages and documentation (`docs`); and the web pages (`www`). All bugs in these areas should be reported to the FreeBSD developers.

- Code in the base system that is written and maintained by others, and imported into FreeBSD and adapted. Examples include clang(1), and sendmail(8). Most bugs in these areas should be reported to the FreeBSD developers; but in some cases they may need to be reported to the original authors instead if the problems are not FreeBSD-specific.

- Individual applications that are not in the base system but are instead part of the FreeBSD Ports Collection (category `ports`). Most of these applications are not written by FreeBSD developers; what FreeBSD provides is merely a framework for installing the

application. Therefore, only report a problem to the FreeBSD developers when the problem is believed to be FreeBSD-specific; otherwise, report it to the authors of the software.

Then, ascertain whether the problem is timely. There are few things that will annoy a developer more than receiving a problem report about a bug she has already fixed.

If the problem is in the base system, first read the FAQ section on FreeBSD versions, if you are not already familiar with the topic. It is not possible for FreeBSD to fix problems in anything other than certain recent branches of the base system, so filing a bug report about an older version will probably only result in a developer advising you to upgrade to a supported version to see if the problem still recurs. The Security Officer team maintains the list of supported versions.

If the problem is in a port, note that you must first upgrade to the latest version of the Ports Collection and see if the problem still applies. Due to the rapid pace of changes in these applications, it is infeasible for FreeBSD to support anything other than the absolute latest versions, and problems with older version of applications simply cannot be fixed.

3. Preparations

A good rule to follow is to always do a background search before submitting a problem report. Maybe the problem has already been reported; maybe it is being discussed on the mailing lists, or recently was; it may even already be fixed in a newer version than what you are running. You should therefore check all the obvious places before submitting your problem report. For FreeBSD, this means:

- The FreeBSD Frequently Asked Questions (FAQ) list. The FAQ attempts to provide answers for a wide range of questions, such as those concerning hardware compatibility, user applications, and kernel configuration.
- The mailing lists—if you are not subscribed, use the searchable archives on the FreeBSD web site. If the problem has not been discussed on the lists, you might try posting a message about it and waiting a few days to see if someone can spot something that has been overlooked.
- Optionally, the entire web—use your favorite search engine to locate any references to the problem. You may even get hits from archived mailing lists or newsgroups you did not know of or had not thought to search through.
- Next, the searchable FreeBSD PR database (Bugzilla). Unless the problem is recent or obscure, there is a fair chance it has already been reported.
- Most importantly, attempt to see if existing documentation in the source base addresses your problem.

For the base FreeBSD code, you should carefully study the contents of `/usr/src/UPDATING` on your system or the latest version at `http://svnweb.freebsd.org/base/head/UPDATING?view=log` . (This is vital information if you are upgrading from one version to another—especially if you are upgrading to the FreeBSD-CURRENT branch).

However, if the problem is in something that was installed as a part of the FreeBSD Ports Collection, you should refer to `/usr/ports/UPDATING` (for individual ports) or `/usr/ports/CHANGES` (for changes that affect the entire Ports Collection). `http://svnweb.freebsd.org/ports/head/UPDATING?view=log` and `http://svnweb.freebsd.org/ports/head/CHANGES?view=log` are also available via svnweb.

4. Writing the Problem Report

Now that you have decided that your issue merits a problem report, and that it is a FreeBSD problem, it is time to write the actual problem report. Before we get into the mechanics of the program used to generate and submit PRs, here are some tips and tricks to help make sure that your PR will be most effective.

4.1. Tips and Tricks for Writing a Good Problem Report

- *Do not leave the "Synopsis" line empty.* The PRs go both onto a mailing list that goes all over the world (where the "Synopsis" is used for the `Subject:` line), but also into a database. Anyone who comes along later and browses the database by synopsis, and finds a PR with a blank subject line, tends just to skip over it. Remember that PRs stay in this database until they are closed by someone; an anonymous one will usually just disappear in the noise.

- *Avoid using a weak "Synopsis" line.* You should not assume that anyone reading your PR has any context for your submission, so the more you provide, the better. For instance, what part of the system does the problem apply to? Do you only see the problem while installing, or while running? To illustrate, instead of `Synopsis: portupgrade is broken`, see how much more informative this seems: `Synopsis: port ports-mgmt/portupgrade coredumps on -current.` (In the case of ports, it is especially helpful to have both the category and portname in the "Synopsis" line.)

- *If you have a patch, say so.* A PR with a patch included is much more likely to be looked at than one without. If you are including one, put the string `[patch]` (including the brackets) at the beginning of the "Synopsis". (Although it is not mandatory to use that exact string, by convention, that is the one that is used.)

- *If you are a maintainer, say so.* If you are maintaining a part of the source code (for instance, a port), you might consider adding the string `[maintainer update]` (including the brackets) at the beginning of your synopsis line, and you definitely should set the

"Class" of your PR to `maintainer-update`. This way any committer that handles your PR will not have to check.

- *Be specific.* The more information you supply about what problem you are having, the better your chance of getting a response.

  - Include the version of FreeBSD you are running (there is a place to put that, see below) and on which architecture. You should include whether you are running from a release (e.g., from a CD-ROM or download), or from a system maintained by Subversion (and, if so, what revision number you are at). If you are tracking the FreeBSD-CURRENT branch, that is the very first thing someone will ask, because fixes (especially for high-profile problems) tend to get committed very quickly, and FreeBSD-CURRENT users are expected to keep up.

  - Include which global options you have specified in your `make.conf`. Note: specifying `-O2` and above to gcc(1) is known to be buggy in many situations. While the FreeBSD developers will accept patches, they are generally unwilling to investigate such issues due to simple lack of time and volunteers, and may instead respond that this just is not supported.

  - If the problem can be reproduced easily, include information that will help a developer to reproduce it themselves. If a problem can be demonstrated with specific input then include an example of that input if possible, and include both the actual and the expected output. If this data is large or cannot be made public, then do try to create a minimal file that exhibits the same issue and that can be included within the PR.

  - If this is a kernel problem, then be prepared to supply the following information. (You do not have to include these by default, which only tends to fill up the database, but you should include excerpts that you think might be relevant):

    - your kernel configuration (including which hardware devices you have installed)

    - whether or not you have debugging options enabled (such as `WITNESS`), and if so, whether the problem persists when you change the sense of that option

    - the full text of any backtrace, panic or other console output, or entries in `/var/log/messages`, if any were generated

    - the output of `pciconf -l` and relevant parts of your `dmesg` output if your problem relates to a specific piece of hardware

    - the fact that you have read `src/UPDATING` and that your problem is not listed there (someone is guaranteed to ask)

    - whether or not you can run any other kernel as a fallback (this is to rule out hardware-related issues such as failing disks and overheating CPUs, which can masquerade as kernel problems)

  - If this is a ports problem, then be prepared to supply the following information. (You do not have to include these by default, which only tends to fill up the database, but you should include excerpts that you think might be relevant):

    - which ports you have installed

    - any environment variables that override the defaults in `bsd.port.mk` , such as `PORTSDIR`

    - the fact that you have read `ports/UPDATING` and that your problem is not listed there (someone is guaranteed to ask)

- *Avoid vague requests for features.* PRs of the form "someone should really implement something that does so-and-so" are less likely to get results than very specific requests. Remember, the source is available to everyone, so if you want a feature, the best way to ensure it being included is to get to work! Also consider the fact that many things like this would make a better topic for discussion on `freebsd-questions` than an entry in the PR database, as discussed above.

- *Make sure no one else has already submitted a similar PR.* Although this has already been mentioned above, it bears repeating here. It only take a minute or two to use the web-based search engine at `https://bugs.freebsd.org/bugzilla/query.cgi` . (Of course, everyone is guilty of forgetting to do this now and then.)

- *Report only one issue per Problem Report.* Avoid including two or more problems within the same report unless they are related. When submitting patches, avoid adding multiple features or fixing multiple bugs in the same PR unless they are closely related—such PRs often take longer to resolve.

- *Avoid controversial requests.* If your PR addresses an area that has been controversial in the past, you should probably be prepared to not only offer patches, but also justification for why the patches are "The Right Thing To Do". As noted above, a careful search of the mailing lists using the archives at `http://www.FreeBSD.org/search/search.html#mailinglists` is always good preparation.

- *Be polite.* Almost anyone who would potentially work on your PR is a volunteer. No one likes to be told that they have to do something when they are already doing it for some motivation other than monetary gain. This is a good thing to keep in mind at all times on Open Source projects.

4.2. Before Beginning

Similar considerations apply to use of the web-based PR submission form. Be careful of cut-and-paste operations that might change whitespace or other text formatting.

Finally, if the submission is lengthy, prepare the work offline so that nothing will be lost if there is a problem submitting it.

4.3. Attaching Patches or Files

When attaching a patch, be sure to use `-u` with diff(1) to create or unified diff and make sure to specify the exact SVN revision numbers of the files you modified so the developers who read your report will be able to apply them easily. For problems with the kernel or the base utilities, a patch against FreeBSD-CURRENT (the HEAD Subversion branch) is preferred since all new code should be applied and tested there first. After appropriate or substantial testing has been done, the code will be merged/migrated to the FreeBSD-STABLE branch.

If you attach a patch inline, instead of as an attachment, note that the most common problem by far is the tendency of some email programs to render tabs as spaces, which will completely ruin anything intended to be part of a Makefile.

Do not send patches as attachments using `Content-Transfer-Encoding: quoted-printable`. These will perform character escaping and the entire patch will be useless.

Also note that while including small patches in a PR is generally all right—particularly when they fix the problem described in the PR—large patches and especially new code which may require substantial review before committing should be placed on a web or ftp server, and the URL should be included in the PR instead of the patch. Patches in email tend to get mangled, and the larger the patch, the harder it will be for interested parties to unmangle it. Also, posting a patch on the web allows you to modify it without having to resubmit the entire patch in a followup to the original PR. Finally, large patches simply increase the size of the database, since closed PRs are not actually deleted but instead kept and simply marked as complete.

You should also take note that unless you explicitly specify otherwise in your PR or in the patch itself, any patches you submit will be assumed to be licensed under the same terms as the original file you modified.

4.4. Filling out the Template

In the email template only, you will find the following single-line fields:

- *Submitter-Id:* Do not change this. The default value of `current-users` is correct, even if you run FreeBSD-STABLE.
- *Confidential:* This is prefilled to `no`. Changing it makes no sense as there is no such thing as a confidential FreeBSD problem report—the PR database is distributed worldwide.

- *Severity:* One of `non-critical`, `serious` or `critical`. Do not overreact; refrain from labeling your problem `critical` unless it really is (e.g., data corruption issues, serious regression from previous functionality in -CURRENT) or `serious` unless it is something that will affect many users (kernel panics or freezes; problems with particular device drivers or system utilities). FreeBSD developers will not necessarily work on your problem faster if you inflate its importance since there are so many other people who have done exactly that — in fact, some developers pay little attention to this field because of this.

- *Priority:* This field indicates how widespread the effects of this bug is likely to be.

The next section describes fields that are common to both the email interface and the web interface:

- *Originator:* Please specify your real name, optionally followed by your email address in angle brackets. In the email interface, this is normally prefilled with the `gecos` field of the currently logged-in user.

Note

The email address you use will become public information and may become available to spammers. You should either have spam handling procedures in place, or use a temporary email account. However, please note that if you do not use a valid email account at all, we will not be able to ask you questions about your PR.

- *Organization:* Whatever you feel like. This field is not used for anything significant.

- *Synopsis:* Fill this out with a short and accurate description of the problem. The synopsis is used as the subject of the problem report email, and is used in problem report listings and summaries; problem reports with obscure synopses tend to get ignored.

  As noted above, if your problem report includes a patch, please have the synopsis start with `[patch]` (including the brackets); if this is a ports PR and you are the maintainer, you may consider adding `[maintainer update]` (including the brackets) and set the "Class" of your PR to `maintainer-update`.

- *Category:* Choose an appropriate category.

  The first thing you need to do is to decide what part of the system your problem lies in. Remember, FreeBSD is a complete operating system, which installs both a kernel, the standard libraries, many peripheral drivers, and a large number of utilities (the "base system"). However, there are thousands of additional applications in the Ports

Collection. You'll first need to decide if the problem is in the base system or something installed via the Ports Collection.

Here is a description of the major categories:

- If a problem is with the kernel, the libraries (such as standard C library `libc`), or a peripheral driver in the base system, in general you will use the `kern` category. (There are a few exceptions; see below). In general these are things that are described in section 2, 3, or 4 of the manual pages.

- If a problem is with a binary program such as sh(1) or mount(8), you will first need to determine whether these programs are in the base system or were added via the Ports Collection. If you are unsure, you can do `whereis` *`programname`*. FreeBSD's convention for the Ports Collection is to install everything underneath `/usr/local`, although this can be overridden by a system administrator. For these, you will use the `ports` category (yes, even if the port's category is `www`; see below). If the location is `/bin`, `/usr/bin`, `/sbin`, or `/usr/sbin`, it is part of the base system, and you should use the `bin` category. (A few programs, such as gcc(1), actually use the `gnu` category, but do not worry about that for now.) These are all things that are described in section 1 or 8 of the manual pages.

- If you believe that the error is in the startup `(rc)` scripts, or in some kind of other non-executable configuration file, then the right category is `conf` (configuration). These are things that are described in section 5 of the manual pages.

- If you have found a problem in the documentation set (articles, books, man pages), the correct choice is `docs`.

- If you are having a problem with the FreeBSD web pages, the proper choice is `www`.

Note

if you are having a problem with something from a port named `www/`*`someportname`*, this nevertheless goes in the `ports` category.

There are a few more specialized categories.

- If the problem would otherwise be filed in `kern` but has to do with the USB subsystem, the correct choice is `usb`.

- If the problem would otherwise be filed in `kern` but has to do with the threading libraries, the correct choice is `threads`.

- If the problem would otherwise be in the base system, but has to do with our adherence to standards such as POSIX®, the correct choice is `standards`.

- If the problem has to do with errors internal to a Java Virtual Machine™ (JVM™), even though Java™ was installed from the Ports Collection, you should select the `java` category. More general problems with Java™ ports still go under `ports`.

This leaves everything else.

- If you are convinced that the problem will only occur under the processor architecture you are using, select one of the architecture-specific categories: commonly `i386` for Intel-compatible machines in 32-bit mode; `amd64` for AMD machines running in 64-bit mode (this also includes Intel-compatible machines running in EMT64 mode); and less commonly `arm`, `ia64`, `powerpc`, and `sparc64`.

Note

These categories are quite often misused for "I do not know" problems. Rather than guessing, please just use `misc`.

Example 1. Correct Use of Arch-Specific Category

You have a common PC-based machine, and think you have encountered a problem specific to a particular chipset or a particular motherboard: `i386` is the right category.

Example 2. Incorrect Use of Arch-Specific Category

You are having a problem with an add-in peripheral card on a commonly seen bus, or a problem with a particular type of hard disk drive: in this case, it probably applies to more than one architecture, and `kern` is the right category.

- If you really do not know where the problem lies (or the explanation does not seem to fit into the ones above), use the `misc` category. Before you do so, you may wish to

ask for help on the FreeBSD general questions mailing list first. You may be advised that one of the existing categories really is a better choice.

Here is the current list of categories (taken from `http://svnweb.freebsd.org/base/head/gnu/usr.bin/send-pr/categories`):

- `advocacy`: problems relating to FreeBSD's public image. Obsolete.
- `amd64`: problems specific to the AMD64 platform.
- `arm`: problems specific to the ARM platform.
- `bin`: problems with userland programs in the base system.
- `conf`: problems with configuration files, default values, and so forth.
- `docs`: problems with manual pages or on-line documentation.
- `gnu`: problems with imported GNU software such as gcc(1) or grep(1).
- `i386`: problems specific to the i386™ platform.
- `ia64`: problems specific to the ia64 platform.
- `java`: problems related to the Java™ Virtual Machine.
- `kern`: problems with the kernel, (non-platform-specific) device drivers, or the base libraries.
- `misc`: anything that does not fit in any of the other categories. (Note that there is almost nothing that truly belongs in this category, except for problems with the release and build infrastructure. Temporary build failures on `HEAD` do not belong here. Also note that it is easy for things to get lost in this category).
- `ports`: problems relating to the Ports Collection.
- `powerpc`: problems specific to the PowerPC® platform.
- `sparc64`: problems specific to the SPARC64® platform.
- `standards`: Standards conformance issues.
- `threads`: problems related to the FreeBSD threads implementation (especially on FreeBSD-CURRENT).
- `usb`: problems related to the FreeBSD USB implementation.
- `www`: Changes or enhancements to the FreeBSD website.

- *Class:* Choose one of the following:
  - `sw-bug:` software bugs.
  - `doc-bug:` errors in documentation.
  - `change-request:` requests for additional features or changes in existing features.
  - `update:` updates to ports or other contributed software.
  - `maintainer-update:` updates to ports for which you are the maintainer.
- *Release:* The version of FreeBSD that you are running. This is filled out automatically if you are using send-pr(1) and need only be changed if you are sending a problem report from a different system than the one that exhibits the problem.

Finally, there is a series of multi-line fields:

- *Environment:* This should describe, as accurately as possible, the environment in which the problem has been observed. This includes the operating system version, the version of the specific program or file that contains the problem, and any other relevant items such as system configuration, other installed software that influences the problem, etc. —quite simply everything a developer needs to know to reconstruct the environment in which the problem occurs.
- *Description:* A complete and accurate description of the problem you are experiencing. Try to avoid speculating about the causes of the problem unless you are certain that you are on the right track, as it may mislead a developer into making incorrect assumptions about the problem.
- *How-To-Repeat:* A summary of the actions you need to take to reproduce the problem.
- *Fix:* Preferably a patch, or at least a workaround (which not only helps other people with the same problem work around it, but may also help a developer understand the cause for the problem), but if you do not have any firm ideas for either, it is better to leave this field blank than to speculate.

4.5. Sending the Problem Report

If you are using send-pr(1):

Once you are done filling out the template, have saved it, and exit your editor, send-pr(1) will prompt you with `s)end, e)dit or a)bort?` . You can then hit **`s`** to go ahead and submit the problem report, **`e`** to restart the editor and make further modifications, or **`a`** to abort. If you choose the latter, your problem report will remain on disk (send-pr(1) will tell you the filename before it terminates), so you can edit it at your leisure, or maybe

transfer it to a system with better net connectivity, before sending it with the `-f` to send-pr(1):

```
% send-pr -f ~/my-problem-report
```

This will read the specified file, validate the contents, strip comments and send it off.

If you are using the web form:

Before you hit `submit`, you will need to fill in a field containing text that is represented in image form on the page. This unfortunate measure has had to be adopted due to misuse by automated systems and a few misguided individuals. It is a necessary evil that no one likes; please do not ask us to remove it.

Note that you are `strongly advised` to save your work somewhere before hitting `submit`. A common problem for users is to have their web browser displaying a stale image from its cache. If this happens to you, your submission will be rejected and you may lose your work.

If you are unable to view images for any reason, and are also unable to use send-pr(1), please accept our apologies for the inconvenience and email your problem report to the bugbuster team at <freebsd-bugbusters@FreeBSD.org >.

5. Follow-up

Once the problem report has been filed, you will receive a confirmation by email which will include the tracking number that was assigned to your problem report and a URL you can use to check its status. With a little luck, someone will take an interest in your problem and try to address it, or, as the case may be, explain why it is not a problem. You will be automatically notified of any change of status, and you will receive copies of any comments or patches someone may attach to your problem report's audit trail.

If someone requests additional information from you, or you remember or discover something you did not mention in the initial report, please submit a follow up. The number one reason for a bug not getting fixed is lack of communication with the originator.

- The easiest way is to use the comment option on the individual PR's web page, which you can reach from the PR search page.

If the problem report remains open after the problem has gone away, just add a comment saying that the problem report can be closed, and, if possible, explaining how or when the problem was fixed.

Sometimes there is a delay of a week or two where the problem report remains untouched, not assigned or commented on by anyone. This can happen when there is an increased

problem report backlog or during a holiday season. When a problem report has not received attention after several weeks, it is worth finding a committer particularly interested in working on it.

There are a few ways to do so, ideally in the following order, with a few days between attempting each communication channel:

- Find the relevant FreeBSD mailing list for the problem report from the list in the Handbook and send a message to that list asking about assistance or comments on the problem report.
- Join the relevant IRC channels. A partial listing is here: https://wiki.freebsd.org/IrcChannels. Inform the people in that channel about the problem report and ask for assistance. Be patient and stay in the channel after posting, so that the people from different time zones around the world have a chance to catch up.
- Find committers interested in the problem that was reported. If the problem was in a particular tool, binary, port, document, or source file, check the SVN Repository. Locate the last few committers who made substantive changes to the file, and try to reach them via IRC or email. A list of committers and their emails can be found in the Contributors to FreeBSD article.

Remember that these people are volunteers, just like maintainers and users, so they might not be immediately available to assist with the problem report. Patience and consistency in the follow-ups is highly advised and appreciated. With enough care and effort dedicated to that follow-up process, finding a committer to take care of the problem report is just a matter of time.

6. If There Are Problems

If you found an issue with the bug system, file a bug! There is a category for exactly this purpose. If you are unable to do so, contact the bug wranglers at <bugmeister@FreeBSD.org >.

7. Further Reading

This is a list of resources relevant to the proper writing and processing of problem reports. It is by no means complete.

- How to Report Bugs Effectively—an excellent essay by Simon G. Tatham on composing useful (non-FreeBSD-specific) problem reports.
- Problem Report Handling Guidelines—valuable insight into how problem reports are handled by the FreeBSD developers.

Index

P

problem reports, 2