Researcher Guidelines

The Linux kernel community welcomes transparent research on the Linux kernel, the activities involved in producing it, and any other byproducts of its development. Linux benefits greatly from this kind of research, and most aspects of Linux are driven by research in one form or another.

The community greatly appreciates if researchers can share preliminary findings before making their results public, especially if such research involves security. Getting involved early helps both improve the quality of research and ability for Linux to improve from it. In any case, sharing open access copies of the published research with the community is recommended.

This document seeks to clarify what the Linux kernel community considers acceptable and non-acceptable practices when conducting such research. At the very least, such research and related activities should follow standard research ethics rules. For more background on research ethics generally, ethics in technology, and research of developer communities in particular, see:

- History of Research Ethics
- IEEE Ethics
- Developer and Researcher Views on the Ethics of Experiments on Open-Source Projects

The Linux kernel community expects that everyone interacting with the project is participating in good faith to make Linux better. Research on any publicly-available artifact (including, but not limited to source code) produced by the Linux kernel community is welcome, though research on developers must be distinctly opt-in.

Passive research that is based entirely on publicly available sources, including posts to public mailing lists and commits to public repositories, is clearly permissible. Though, as with any research, standard ethics must still be followed.

Active research on developer behavior, however, must be done with the explicit agreement of, and full disclosure to, the individual developers involved. Developers cannot be interacted with/experimented on without consent; this, too, is standard research ethics.

To help clarify: sending patches to developers *is* interacting with them, but they have already consented to receiving *good faith contributions*. Sending intentionally flawed/vulnerable patches or contributing misleading information to discussions is not consented to. Such communication can be damaging to the developer (e.g. draining time, effort, and morale) and damaging to the project by eroding the entire developer community's trust in the contributor (and the contributor's organization as a whole), undermining efforts to provide constructive feedback to contributors, and putting end users at risk of software flaws.

Participation in the development of Linux itself by researchers, as with anyone, is welcomed and encouraged. Research into Linux code is a common practice, especially when it comes to developing or running analysis tools that produce actionable results.

When engaging with the developer community, sending a patch has traditionally been the best way to make an impact. Linux already has plenty of known bugs -- what's much more helpful is having vetted fixes. Before contributing, carefully read the appropriate documentation:

- Documentation/process/development-process.rst
- Documentation/process/submitting-patches.rst
- Documentation/admin-guide/reporting-issues.rst
- Documentation/admin-guide/security-bugs.rst

Then send a patch (including a commit log with all the details listed below) and follow up on any feedback from other developers.

When sending patches produced from research, the commit logs should contain at least the following details, so that developers have appropriate context for understanding the contribution. Answer:

- What is the specific problem that has been found?
- How could the problem be reached on a running system?
- What effect would encountering the problem have on the system?
- How was the problem found? Specifically include details about any testing, static or dynamic analysis programs, and any other tools or methods used to perform the work.
- Which version of Linux was the problem found on? Using the most recent release or a recent linux-next branch is strongly preferred (see Documentation/process/howto.rst).
- What was changed to fix the problem, and why it is believed to be correct?
- How was the change build tested and run-time tested?
- What prior commit does this change fix? This should go in a "Fixes:" tag as the documentation describes.
- Who else has reviewed this patch? This should go in appropriate "Reviewed-by:" tags; see below.

For example:

```
From: Author <author@email>
Subject: [PATCH] drivers/foo_bar: Add missing kfree()

The error path in foo_bar driver does not correctly free the allocated
struct foo_bar_info. This can happen if the attached foo_bar device
rejects the initialization packets sent during foo_bar_probe(). This
would result in a 64 byte slab memory leak once per device attach,
wasting memory resources over time.
```

```
This flaw was found using an experimental static analysis tool we are
developing, LeakMagic[1], which reported the following warning when
analyzing the v5.15 kernel release:

 path/to/foo_bar.c:187: missing kfree() call?

Add the missing kfree() to the error path. No other references to
this memory exist outside the probe function, so this is the only
place it can be freed.

x86_64 and arm64 defconfig builds with CONFIG_FOO_BAR=y using GCC
11.2 show no new warnings, and LeakMagic no longer warns about this
code path. As we don't have a FooBar device to test with, no runtime
testing was able to be performed.

[1] https://url/to/leakmagic/details

Reported-by: Researcher <researcher@email>
Fixes: aaaabbbbccccdddd ("Introduce support for FooBar")
Signed-off-by: Author <author@email>
Reviewed-by: Reviewer <reviewer@email>
```

If you are a first time contributor it is recommended that the patch itself be vetted by others privately before being posted to public lists. (This is required if you have been explicitly told your patches need more careful internal review.) These people are expected to have their "Reviewed-by" tag included in the resulting patch. Finding another developer familiar with Linux contribution, especially within your own organization, and having them help with reviews before sending them to the public mailing lists tends to significantly improve the quality of the resulting patches, and there by reduces the burden on other developers.

If no one can be found to internally review patches and you need help finding such a person, or if you have any other questions related to this document and the developer community's expectations, please reach out to the private Technical Advisory Board mailing list: <tech-board@lists.linux-foundation.org>.