Make Linux developers fix your kernel bug

Thorsten Leemhuis
sometimes reports on kernel bugs will just fizzle out
in rare cases, developers will be unable to fix an issues
intro;

kernel contains code nobody is really responsible for
the Linux kernel is made by volunteers
you can't really force volunteers to do work they can't do or don't want to do
Linux kernel developers are obliged to fix some issues!
developers will gladly address most issues in their code
developers will gladly address most issues in their code, unless life gets in the way :-/
then bad bug reports are the first developers will let fall through the cracks!
developers will gladly address most issues in their code, unless life gets in the way :-/
developers will gladly address most issues in their code, if you write a decent report!
developers will gladly address most issues in their code, if you write a decent report!
that's how you make most developers fix your bug, if they are able to
intro;

you'll also learn when you can insist on a fix
and how to spot issues unlikely to be fixed
[ act 1 ]
1. create a decent report
Build your own 'Linus land'

Full instructions inside

Linux kernel community
Build your own 'Linus land'

Full instructions inside

Takes just a flick of your fingers
Build your own 'Linus land'

Full instructions inside
1. create a decent report
   a) ensure your kernel is vanilla
most kernels used in the wild are not vanilla
often heavily modified & enhanced
makes most distro kernel's unsuitable for reporting issues Linux kernel devs.
you might want to report the issue to your Linux distributor
decent report; vanilla;
or install a vanilla kernel yourself instead – for example a pre-built one
or compile a kernel yourself

hint: `make olddefconfig localmodconfig` makes things easier and relatively fast
decent report; vanilla;

check if issue happens with a vanilla kernel, too
focus on this kernel in your report, forget the distro's
mentioning the distro's even briefly often just complicates report unnecessarily
1. create a decent report

   a) ensure your kernel is vanilla
Build your own 'Linus land'

Full instructions inside
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
test with latest mainline (aka -RC) release
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td><a href="https://www.kernel.org/pub/">https://www.kernel.org/pub/</a></td>
</tr>
<tr>
<td>GIT</td>
<td><a href="https://git.kernel.org/">https://git.kernel.org/</a></td>
</tr>
<tr>
<td>RSYNC</td>
<td>rsync://rsync.kernel.org/pub/</td>
</tr>
</tbody>
</table>

**Latest Release**

<table>
<thead>
<tr>
<th>Release</th>
<th>Date</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainline:</td>
<td>5.17-rc7</td>
<td>[tarball] [patch] [inc. patch] [view diff] [browse]</td>
</tr>
<tr>
<td>stable:</td>
<td>5.16.14</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm:</td>
<td>5.15.28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm:</td>
<td>5.10.105</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>Protocol</td>
<td>Location</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------</td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td><a href="https://www.kernel.org/pub/">https://www.kernel.org/pub/</a></td>
<td></td>
</tr>
<tr>
<td>GIT</td>
<td><a href="https://git.kernel.org/">https://git.kernel.org/</a></td>
<td></td>
</tr>
<tr>
<td>RSYNC</td>
<td>rsync://rsync.kernel.org/pub/</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Version</th>
<th>Date</th>
<th>Formats</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainline</td>
<td>5.18</td>
<td>2022-05-22</td>
<td>tarball, pgp, patch</td>
<td>[view diff] [browse]</td>
</tr>
<tr>
<td>stable</td>
<td>5.18.1</td>
<td>2022-05-30</td>
<td>tarball, pgp, patch</td>
<td>[view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>stable</td>
<td>5.17.12</td>
<td>2022-05-30</td>
<td>tarball, pgp, patch, inc. patch</td>
<td>[view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>5.15.44</td>
<td>2022-05-30</td>
<td>tarball, pgp, patch, inc. patch</td>
<td>[view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>5.10.119</td>
<td>2022-05-30</td>
<td>tarball, pgp, patch, inc. patch</td>
<td>[view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>Protocol</td>
<td>Location</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td><a href="https://www.kernel.org/pub/">https://www.kernel.org/pub/</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GIT</td>
<td><a href="https://git.kernel.org/">https://git.kernel.org/</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSYNC</td>
<td>rsync://rsync.kernel.org/pub/</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainline:</td>
<td>5.17-rc7</td>
<td>2022-03-06 [tarball] [patch] [inc. patch] [view diff] [browse]</td>
</tr>
<tr>
<td>stable:</td>
<td>5.16.14</td>
<td>2022-03-11 [tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm:</td>
<td>5.15.28</td>
<td>2022-03-11 [tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm:</td>
<td>5.10.105</td>
<td>2022-03-11 [tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
</tbody>
</table>
focus your report on the freshest kernel you tested
mentioning older briefly somewhere can be okay, but often just make report hard to grasp
## The Linux Kernel Archives

**Protocol** | **Location** |
--- | --- |
HTTP | https://www.kernel.org/pub/ |
GIT | https://git.kernel.org/ |
RSYNC | rsync://rsync.kernel.org/pub/ |

### Latest Release

**5.16.14**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Location</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainline</td>
<td>5.17-rc7</td>
<td>2022-03-06</td>
</tr>
<tr>
<td>stable</td>
<td>5.16.14</td>
<td>2022-03-11</td>
</tr>
<tr>
<td>longterm</td>
<td>5.15.28</td>
<td>2022-03-11</td>
</tr>
<tr>
<td>longterm</td>
<td>5.10.105</td>
<td>2022-03-11</td>
</tr>
</tbody>
</table>
decent report; freshness;

some bugfixes are never backported to stable/longterm kernel series
decent report; freshness;

makes longterm (LTS) kernels quite unsuitable for reporting
decent report; freshness;

exception: regressions within a stable or longterm series

something breaks 5.15.10 -> 5.15.11
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
Build your own 'Linus land'

Full instructions inside

ORLY?

Linux kernel community
decent report;

1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
decent report; integrity;

[thl@t14s ~]$ cat /proc/sys/kernel/tainted
0

[thl@t14s ~]$
decent report; integrity;

[thl@t14s ~]$ cat /proc/sys/kernel/tainted
1

[thl@t14s ~]$
Nvidia's proprietary graphics driver
decent report; integrity;

all out-of-tree drivers are a problem
incl. Nvidia's new open kernel driver
decent report; integrity;
deinstall such drivers, reboot, check if issue still present and recheck the tainted flag!
decend report; integrity;

many other incidents
can taint kernel
an "Oops" for example
decent report; integrity;

tainted kernels most of the time unsuitable for reporting bugs
decent report; integrity;

big exception:
the first Oops, warning, etc.
Check ‘taint’ flag

Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.

The kernel marks itself with a ‘taint’ flag when something happens that might lead to follow-up errors that look totally unrelated. The issue you face might be such an error if your kernel is tainted. That’s why it’s in your interest to rule this out early before investing more time into this process. This is the only reason why this step is here, as this process later will tell you to install the latest mainline kernel; you will need to check the taint flag again then, as that’s when it matters because it’s the kernel the report will focus on.

On a running system is easy to check if the kernel tainted itself: if cat /proc/sys/kernel/tainted returns ‘0’ then the kernel is not tainted and everything is fine. Checking that file is impossible in some situations; that’s why the kernel also mentions the taint status when it reports an internal problem (a “kernel bug”), a recoverable error (a “kernel Oops”) or a
Table for decoding tainted state

<table>
<thead>
<tr>
<th>Bit</th>
<th>Log</th>
<th>Number</th>
<th>Reason that got the kernel tainted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>/P</td>
<td>1</td>
<td>proprietary module was loaded</td>
</tr>
<tr>
<td>1</td>
<td>/F</td>
<td>2</td>
<td>module was force loaded</td>
</tr>
<tr>
<td>2</td>
<td>/S</td>
<td>4</td>
<td>kernel running on an out of specification system</td>
</tr>
<tr>
<td>3</td>
<td>/R</td>
<td>8</td>
<td>module was force unloaded</td>
</tr>
<tr>
<td>4</td>
<td>/M</td>
<td>16</td>
<td>processor reported a Machine Check Exception (MCE)</td>
</tr>
<tr>
<td>5</td>
<td>/B</td>
<td>32</td>
<td>bad page referenced or some unexpected page flags</td>
</tr>
<tr>
<td>6</td>
<td>/U</td>
<td>64</td>
<td>taint requested by userspace application</td>
</tr>
<tr>
<td>7</td>
<td>/D</td>
<td>128</td>
<td>kernel died recently, i.e. there was an OOPS or BUG</td>
</tr>
<tr>
<td>8</td>
<td>/A</td>
<td>256</td>
<td>ACPI table overridden by user</td>
</tr>
<tr>
<td>9</td>
<td>/W</td>
<td>512</td>
<td>kernel issued warning</td>
</tr>
<tr>
<td>10</td>
<td>/C</td>
<td>1024</td>
<td>staging driver was loaded</td>
</tr>
<tr>
<td>11</td>
<td>/I</td>
<td>2048</td>
<td>workaround for bug in platform firmware applied</td>
</tr>
<tr>
<td>12</td>
<td>/O</td>
<td>4096</td>
<td>externally-built (&quot;out-of-tree&quot;) module was loaded</td>
</tr>
<tr>
<td>13</td>
<td>/E</td>
<td>8192</td>
<td>unsigned module was loaded</td>
</tr>
<tr>
<td>14</td>
<td>/L</td>
<td>16384</td>
<td>soft lockup occurred</td>
</tr>
<tr>
<td>15</td>
<td>/K</td>
<td>32768</td>
<td>kernel has been live patched</td>
</tr>
<tr>
<td>16</td>
<td>/X</td>
<td>65536</td>
<td>auxiliary taint, defined for and used by distros</td>
</tr>
<tr>
<td>17</td>
<td>/T</td>
<td>131072</td>
<td>kernel was built with the struct randomization plugin</td>
</tr>
</tbody>
</table>

1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
      [continued]
is your hardware working reliably and as specified?
memtest: great idea!
overclocking: stupid idea!
issue with file-system?
fsck the volume!
check `dmesg -H`
look out for anything red or bold
Allocating resources

Allocating resources

Allocating resources

Allocating resources

Allocating resources

done.

thermal thermal_zone0: failed to read out thermal zone (-61)

PM: suspend exit

Generic FE-GE Realtek PHY r8169-0-200:00: attached PHY driver (mii_bus:phy_addr=r8169-0-200:00, irq=MAC)

r8169 0000:02:00.0 enp2s0f0: Link is Down

psmouse serio1: Touchpad at isa0060/serio1/input0 lost sync at byte 6

psmouse serio1: Touchpad at isa0060/serio1/input0 lost sync at byte 6

psmouse serio1: Touchpad at isa0060/serio1/input0 lost sync at byte 6

psmouse serio1: Touchpad at isa0060/serio1/input0 lost sync at byte 6

psmouse serio1: Touchpad at isa0060/serio1/input0 lost sync at byte 6

psmouse serio1: issuing reconnect request

r8169 0000:02:00.0 enp2s0f0: Link is Down

PM: suspend entry (deep)

Filesystmes sync: 0.029 seconds

Bluetooth: hci0: Suspend notifier action (3) failed: 2

Freezing user space processes ... (elapsed 0.004 seconds) done.

OOM killer disabled.

Freezing remaining freezable tasks ... (elapsed 0.001 seconds) done.

printk: Suspending console(s) (use no_console_suspend to debug)

[drm] free PSP TMR buffer

PM: suspend devices took 0.341 seconds

ACPI: EC: interrupt blocked

amdgpu 0000:06:00.0: amdgpu: MODE2 reset

ACPI: PM: Preparing to enter system sleep state S3

ACPI: EC: event blocked

[drm] 55.00: No EDID
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
Build your own 'Linus land'

Full instructions inside

O RLY?

Linux kernel community
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
   d) submit your report to the right place
Welcome to Kernel.org Bugzilla

Please use your distribution’s bug tracking tools

This bugzilla is for reporting bugs against upstream Linux kernels.

If you did not compile your own kernel from scratch, you are probably in the wrong place. Please use the following links to report a bug to your distribution instead:

Ubuntu | Fedora | Arch | Mint | Debian | Red Hat | OpenSUSE | SUSE

To report an issue upstream, please consult this document before opening a new bug:

Reporting Issues

With questions about this site contact bugzilla admins. Please check the FAQ before you do so.
Welcome to Kernel.org Bugzilla

Please use your distribution's bug tracking tools

This bugzilla is for reporting bugs against upstream Linux kernels.

If you did not compile your own kernel from scratch, you are probably in the wrong place. Please use the following links to report a bug to your distribution instead:

Ubuntu | Fedora | Arch | Mint | Debian | Red Hat | OpenSUSE | SUSE

To report an issue upstream, please consult this document before opening a new bug:

Reporting Issues

With questions about this site contact bugzilla admins.
Please check the FAQ before you do so.
Check where you need to report your issue

Locate the driver or kernel subsystem that seems to be causing the issue. Find out how and where its developers expect reports. Note: most of the time this won’t be bugzilla.kernel.org, as issues typically need to be sent by mail to a maintainer and a public mailing list.

It’s crucial to send your report to the right people, as the Linux kernel is a big project and most of its developers are only familiar with a small subset of it. Quite a few programmers for example only care for just one driver, for example one for a WiFi chip; its developer likely will only have small or no knowledge about the internals of remote or unrelated “subsystems”, like the TCP stack, the PCIe/PCI subsystem, memory management or file systems.

Problem is: the Linux kernel lacks a central bug tracker where you can simply file your issue and make it reach the developers that need to know about it. That’s why you have to find the right place and way to report issues yourself. You can do that with the help of a script (see below), but it mainly targets kernel developers and experts. For everybody else the MAINTAINERS file is the better place.

How to read the MAINTAINERS file

To illustrate how to use the MAINTAINERS file, lets assume the WiFi in your Laptop suddenly misbehaves after updating the kernel. In that case it’s likely an issue in the WiFi driver. Obviously it could also be some code it builds upon.
Check where you need to report your issue

Locate the driver or kernel subsystem that seems to be causing the issue. Find out how and where its developers expect reports. Note: most of the time this won’t be bugzilla.kernel.org, as issues typically need to be sent by mail to a maintainer and a public mailing list.

It’s crucial to send your report to the right people, as the Linux kernel is a big project and most of its developers are only familiar with a small subset of it. Quite a few programmers for example only care for just one driver, for example one for a WiFi chip; its developer likely will only have small or no knowledge about the internals of remote or unrelated “subsystems”, like the TCP stack, the PCIe/PCI subsystem, memory management or file systems.

Problem is: the Linux kernel lacks a central bug tracker where you can simply file your issue and make it reach the developers that need to know about it. That’s why you have to find the right place and way to report issues yourself. You can do that with the help of a script (see below), but it mainly targets kernel developers and experts. For everybody else the MAINTAINERS file is the better place.

How to read the MAINTAINERS file

To illustrate how to use the MAINTAINERS file, let’s assume the WiFi in your laptop suddenly misbehaves after updating the kernel. In that case it’s likely an issue in the WiFi driver. Obviously it could also be some code it builds on top of. Now look for the WiFi driver in the MAINTAINERS file and see who is in charge: There is a section for WiFi, which lists a person in charge, an email address, and a mailing address. Note: the email address is not necessarily the developer of the WiFi driver, but it’s the right person to contact.

Let’s look at this section in detail:

```
# WiFi
# a floating chunk of people
# associated with a particular "device" (here: WiFi)

# WiFi
# the name of the "device" this section is about

# Count: 1
# This number counts how many times the device (here: WiFi) is mentioned in this file.

# Name: wifi
# The name of the "device" (here: WiFi) and the people responsible for it.

# - albert@redhat.com
# - albert@redhat.com

# Email: albert@redhat.com
# The email address of the people responsible for the "device".

# Mailing: albert@redhat.com
# The mailing address of the people responsible for the "device".
```
### BT8XX GPIO DRIVER

<table>
<thead>
<tr>
<th>Mail</th>
<th>Michael Buesch <a href="mailto:m@bues.ch">m@bues.ch</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Maintained</td>
</tr>
<tr>
<td>Files</td>
<td>drivers/gpio/gpio-bt8xx.c</td>
</tr>
</tbody>
</table>

### BTRFS FILE SYSTEM

<table>
<thead>
<tr>
<th>Mail</th>
<th>Chris Mason <a href="mailto:clm@fb.com">clm@fb.com</a>, Josef Bacik <a href="mailto:josef@toxicpanda.com">josef@toxicpanda.com</a>, David Sterba <a href="mailto:dsterba@suse.com">dsterba@suse.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mailing list</td>
<td><a href="mailto:linux-btrfs@vger.kernel.org">linux-btrfs@vger.kernel.org</a></td>
</tr>
<tr>
<td>Status</td>
<td>Maintained</td>
</tr>
<tr>
<td>Patchwork</td>
<td><a href="http://patchwork.kernel.org/project/linux-btrfs/list/">http://patchwork.kernel.org/project/linux-btrfs/list/</a></td>
</tr>
<tr>
<td>Chat</td>
<td>irc://irc.libera.chat/btrfs</td>
</tr>
<tr>
<td>SCM</td>
<td>git <a href="https://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux.git">https://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux.git</a></td>
</tr>
<tr>
<td>Files</td>
<td>filesystems/btrfs fs/btrfs/include/linux/btrfs* include/uapi/linux/btrfs*</td>
</tr>
</tbody>
</table>

BT8XX_GPIO DRIVER

Mail: Michael Buesch <m@bues.ch>
Status: Maintained
Web-page: http://bu3sch.de/btgpio.php
Files: drivers/gpio/gpio-bt8xx.c

BTRFS_FILE_SYSTEM

Mail: Chris Mason <clm@fb.com>, Josef Bacik <josef@toxicpanda.com>
Mailing list: linux-btrfs@vger.kernel.org
Status: Maintained
ACPI

Mail: “Rafael J. Wysocki” <rafael@kernel.org>
Reviewer: Len Brown <lenb@kernel.org>
Mailing list: linux-acpi@vger.kernel.org
Status: Supported
Web-page: https://01.org/linux-acpi
Patchwork: https://patchwork.kernel.org/project/linux-acpi/list/
**bugs:** https://bugzilla.kernel.org

SCM: git git://git.kernel.org/pub/scm/linux/kernel/git/rafael/linux-pm
Files: Documentation/ABI/testing/configfs-acpi Documentation/ABI/testing/sysfs-bus-acpi
       Documentation/firmware-guide/acpi/ drivers/acpi/ drivers/pci/ */acpi* drivers/pci/*acpi*
       drivers/pnp/pnpacpi/ include/acpi/ include/linux/acpi.h include/linux/fwnode.h tools/power/acpi/

ACPI APEI

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/MAINTAINERS
RADEON and AMDGPU DRM DRIVERS

Mail: Alex Deucher <alexander.deucher@amd.com>, Christian König <christian.koenig@amd.com>, Pan, Xinhui <Xinhui.Pan@amd.com>
Mailing list: amd-gfx@lists.freedesktop.org
Status: Supported
SCM: git https://gitlab.freedesktop.org/aod5f/linux.git
bugs: https://gitlab.freedesktop.org/drm/amd/-/issues
chat: irc://irc.oftc.net/radeon
Files: drivers/gpu/drm/amd/drivers/gpu/drm/radeon/include/uapi/drm/amdGPU_drm.h
      include/uapi/drm/radeon_drm.h

RADEON FRAMEBUFFER DISPLAY DRIVER

Mail: Benjamin Herrenschmidt <benh@kernel.crashing.org>
Mailing list: linux-fbdev@vger.kernel.org
Status: Maintained

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/MAINTAINERS
decent report; right place;
sadly MAINTAINERS contains more than 2000 entries:-/
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
   d) submit your report to the right place
1. create a decent report
   a) ensure your kernel is vanilla
   b) ensure your kernel is fresh
   c) ensure your kernel's and system's integrity
   d) submit your report to the right place
   e) depict the problem adequately
how to write a good report" worth its own, quite long talk
decent report; depiction;

a balancing act
think of it as asking for a favor
a favor from someone that doesn't have to help you
decent report; depiction;
a favor from a someone that might be stressed or really short on time
decent report; depiction;

hence, make your depiction easy to grasp for recipients
describe the problem neither to brief nor as a novella
mention version, vanilla, and taint status
upload & link clearly relevant logs or attach them but *don't* overload the report!
decent report; depiction;

often relevant: output from `dmesg` & `lspci -nn`;
maybe kernel's '.config', too
decent report; depiction;

add two or three sentences summarizing the situation on top of your depiction
use a even more condensed and crystal-clear depiction as subject
in general: don't over-think or overdo your report!
short report will often do

getting the basics right (vanilla, fresh version, no taint, easy to grasp, ...) is important
check for existing reports about the problem to join
check <insert favorite search engine>, lore.kernel.lorg/all/, and bugzilla.kernel.org
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order. That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem: those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- Make sure it’s not the kernel’s surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of.
it tells you to check what kind of issue you deal with
[ act 2 ]
2. the kind of issue at hand

a) issues someone is obliged to address

I. security vulnerabilities

II. devastating bugs

III. regressions
2. the kind of issue at hand

a) issues someone is obliged to address
2. the kind of issue at hand
   a) issues someone is obliged to address
      I. security vulnerabilities
Security bugs

Linux kernel developers take security very seriously. As such, we’d like to know when a security bug is found so that it can be fixed and disclosed as quickly as possible. Please report security bugs to the Linux kernel security team.

Contact

The Linux kernel security team can be contacted by email at <security@kernel.org>. This is a private list of security officers who will help verify the bug report and
2. the kind of issue at hand
   a) issues someone is obliged to address
      I. security vulnerabilities
      II. devastating bugs
kind of issue; must fix; devastating;

something really really bad

data is lost or damaged,
hardware is bricked, ...
kind of issue; must fix; devastating;

make impact & urgency obvious in your report
2. The kind of issue at hand
   a) Issues someone is obliged to address
      I. Security vulnerabilities
      II. Devastating bugs
      III. Regressions
something breaks when updating the kernel
say from 5.15 -> 5.16 or from 5.17.3 -> 5.17.4
first rule of Linux kernel development: "we don't cause regressions"
Linux kernel regression status

[current] [mainline] [stable/longterm] [dormant] [resolved] | [new] | [all]

current cycle (v5.18.. aka v5.18-post), culprit identified
none known by regzbot

current cycle (v5.18.. aka v5.18-post), unkown culprit
none known by regzbot

previous cycle (v5.17..v5.18), culprit identified, with activity in the past three months

- **ff04f4a9b05** (v5.18-rc1)
  - mm: chiq_test runs 7 minutes instead of ~ 1 second. by Stefan Wahren
    Earliest & latest **activity: 7 & 3 days ago.** Noteworthy: [patch].

- **f26b3fa04611** (v5.18-rc1)
  - mm: [mm/page_alloc] f26b3fa046: netperf.Throughput_Mbps -18.0% regression by kernel test robot
    Earliest & latest **activity: 39 & 7 days ago.** Noteworthy: [patch].

older cycles (..v5.17), culprit identified, with activity in the past three months

- **44c57f205876** (v5.15-rc1)
  - qla2xxx: tape drive not removed after unplug FC cable by Tony Battersby
    Earliest & latest **activity: 3 & 1 days ago.**

- **b2af264ad3af** (v5.16-rc1)
  - bluetooth: HSP/HFP mSBC profile broken with QCA6174 by bugzilla-daemon@kernel.org
    Earliest & latest **activity: 110 & 1 days ago.** Noteworthy: [1], [2], [3], [4], [fix incoming].

- **bdd8b6c98239** (v5.17-rc1)
  - Xorg SEGV in Xen PV dom0 after updating from 5.16.18 to 5.17.5 by Marek Marczykowski-Górecki
    Earliest & latest **activity: 25 & 1 days ago.** Noteworthy: [1], [patch].

- **453e41085183** (v5.12-rc1)
  - tboot suspend broken on Lenovo T460p by Derek Dolney
    Earliest & latest **activity: 23 & 0 days ago.** Noteworthy: [1].
This website is part of a project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 871528.
Linux kernel regression status

[current_cycle (v5.18.. aka v5.18-post), culprit identified]
none known by regzbot

[current_cycle (v5.18.. aka v5.18-post), unkown culprit]
none known by regzbot

[previous_cycle (v5.17..v5.18), culprit identified, with activity in the past three months]
- **ff04f4a9b05** (v5.18-rc1)
  - **mm: chiq_test runs 7 minutes instead of ~ 1 second.** by Stefan Wahren
    Earliest & latest activity: 7 & 3 days ago. Noteworthy: [patch].
- **f26b3fa04611** (v5.18-rc1)
  - **mm: [mm/page_alloc] f26b3fa046: netperf.Throughput_Mbps -18.0% regression** by kernel test robot

[older cycles (...v5.17), culprit identified, with activity in the past three months]
- **44c57f205876** (v5.15-rc1)
  - **qla2xxx: tape drive not removed after unplug FC cable** by Tony Batterby
    Earliest & latest activity: 3 & 1 days ago.
- **b2af264ad3af** (v5.16-rc1)
  - **bluetooth: HSP/HFP mSBC profile broken with QCA6174** by bugzilla-daemon@kernel.org
    Earliest & latest activity: 110 & 1 days ago. Noteworthy: [1], [2], [3], [4], [fix incoming].
- **bdd8b6c98239** (v5.17-rc1)
  - **Xorg SEGV in Xen PV dom0 after updating from 5.16.18 to 5.17.5** by Marek Marczykowski-Górecki
    Earliest & latest activity: 25 & 1 days ago. Noteworthy: [1], [patch].
- **453e41085183** (v5.18-rc1)
  - **tboot suspend broken on Lenovo T460p** by Derek Dolney
Reporting regressions

“We don’t cause regressions” is the first rule of Linux kernel development; Linux founder and lead developer Linus Torvalds established it himself and ensures it’s obeyed.

This document describes what the rule means for users and how the Linux kernel’s development model ensures to address all reported regressions; aspects relevant for kernel developers are left to [Handling regressions](https://www.kernel.org/doc/html/latest/admin-guide/reporting-regressions.html).

The important bits (aka “TL;DR”)

1. It’s a regression if something running fine with one Linux kernel works worse or not at all with a newer version. Note, the newer kernel has to be compiled using a similar...
kind of issue; mustfix; regressions;

make it obvious your report is about a regression
kind of issue; mustfix; regressions;

CC for forward the report to regressions@lists.linux.dev
kind of issue; mustfix; regressions;

fine print(1):
only userland interfaces matter
[it's thus not a regression if your out-of-tree kernel module breaks]
kind of issue; mustfix; regressions;

fine print(2):
the build config of the newer kernel version must be similar to the older one
kind of issue; mustfix; regressions;

fine print(3):

you often will be asked to find the culprit yourself
kind of issue; mustfix; regressions;

if you find the culprit, a fix is pretty much guaranteed
2. the kind of issue at hand
   a) issues someone is obliged to address
      I. security vulnerabilities
      II. devastating bugs
      III. regressions
2. the kind of issue at hand
   a) issues someone is obliged to address
   b) issues most likely to be ignored
2. the kind of issue at hand
   b) issues most likely to be ignored
      I. known deficits
Linux contains many incomplete drivers
kind of issue; unlikely; deficits;

might lack a volunteer with enough time and/or motivation to improve it
kind of issue; unlikely; deficits;
or some real-world issue prevents improvements
kind of issue; unlikely; deficits;

check internet and docs for known deficits
2. the kind of issue at hand
   b) issues most likely to be ignored
      I. known deficits
2. the kind of issue at hand
   b) issues most likely to be ignored
      I. known deficits
      II. code without an active maintainer
kind of issue; unlikely; w/o maintainer;

code often remains, as it useful for people
EARTH_PT1 MEDIA DRIVER

Mail: Akihiro Tsukada <tskd08@gmail.com>
Mailing list: linux-media@vger.kernel.org
Status: Odd Fixes
Files: drivers/media/pci/pt1/

EARTH_PT3 MEDIA DRIVER

Mail: Akihiro Tsukada <tskd08@gmail.com>
Mailing list: linux-media@vger.kernel.org
Status: Odd Fixes
Files: drivers/media/pci/pt3/
kind of issue; unlikely; w/o maintainer;

sending at least a quick brief report definitely a good idea
<table>
<thead>
<tr>
<th><strong>CAFE CMOS INTEGRATED CAMERA CONTROLLER DRIVER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mailing list:</strong></td>
</tr>
<tr>
<td><strong>Status:</strong></td>
</tr>
<tr>
<td><strong>SCM:</strong></td>
</tr>
<tr>
<td><strong>Files:</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>CAIF NETWORK LAYER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mailing list:</strong></td>
</tr>
<tr>
<td><strong>Status:</strong></td>
</tr>
<tr>
<td><strong>Files:</strong></td>
</tr>
</tbody>
</table>
kind of issue; unlikely; w/o maintainer;
sending at least a quick brief report likely worth it
2. the kind of issue at hand
   a) issues someone is obliged to address
   b) issues most likely to be ignored
2. the kind of issue at hand

a) issues someone is obliged to address
b) issues most likely to be ignored
c) all the other issues
kind of issue; unlikely; the rest;

cal quality of your report!
[ grand finale ]
take this with you
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order. That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- **Are you facing an issue with a Linux kernel a hardware or software vendor provided?** Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- **Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the [Linux Kernel Mailing List (LKML)](https://mail.linux-kernel.org).** If you find matching reports, join the discussion instead of sending a new one.
- **See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem:** those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- **Make sure it’s not the kernel’s surroundings that are causing the issue you face.**
- **Create a fresh backup and put system repair and restore tools at hand.**
- **Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.**
- **Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.**
- **Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of...**
takeaways;

almost all kernel developer are volunteers
they should act on every bug report, but can and will ignore bad reports
takeaways;

act accordingly and sent a decent report, then you'll be heard
takeaways;

(1) check what kind of issue you deal with, as it...
takeaways;

(a) might save you from wasting time on reporting known deficits
takeaways;

(b) tells you what to expect from developers
takeaways;

(2) do your homework
takeaways;

(a) test and report with a *vanilla* kernel
takeaways;

(b) test with a fresh mainline kernel
(c) rule out local interferences
(d) check MAINTAINERS to submit the report to the right place
takeaways;

(e) write a friendly and decent report easy to gasp for others
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order. That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem: those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- Make sure it’s not the kernel’s surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of...
takeaways;

chances then are pretty good someone will help you
takeaways;

and nearly perfect, if you report a bisected regression
takeaways;

that's how you make the Linux developers fix kernel bugs they are able to fix
questions?
Thorsten Leemhuis

mail: linux@leemhuis.info
GPG Key: 0x72B6E6EF4C583D2D
social media: @kernellogger, @knurd42, @knurd42rhfc, @thleemhuis and @thleemhuisfoss on #twitter & #friendica

#EOF
Make Linux developers fix your kernel bug

Thorsten Leemhuis

* let me start by being fully honest
* the title promises a little more than the reality can fulfill

[simple reason]
sometimes reports on kernel bugs will just fizzle out

* this will always happen

[not even worsed news yet]
in rare cases, developers will be unable to fix an issue

[one more bad thing]
kernel contains code nobody is really responsible for

[there is a simple reason for these three aspects]
the Linux kernel is made by volunteers

* note: volunteers != hobbyists
* some of them hobbyists
* most of them employees
* but employees from companies contributing voluntarily

[thing with them is]
intro;

you can't really force volunteers to do work they can't do or don't want to do

* motivate them
* which Linus Torvalds actually does
* only works up to a point
* risk alienating them
* might make them stop contributing
* companies might decide to team up and fork

[analogy helps understanding this situation]
* Linux is a bit like an playground built and maintained completely by a volunteers
* some of those volunteers are hobbyist that wanted to build something for their kids, to learn new stuff, or enjoy helping
* many of those volunteers are actually employees from local or international companies that see some benefit in helping – for example if they have a coffee or gift shop nearby
[but the thing is: sooner or later all all hobbyists and companies move to something new, as their interest and priorities change over time]
* say because kids became adults or companies closed
* some volunteer then vanish
* others still help when at least kindly asked
* often some other volunteer will step in
* but you can force them
* luckily their often is no need to
* unless some play structure breaks or is found to be dangerous

[and that's the same with Linux and the reason]
Linux kernel developers are obliged to fix some issues!

* if they don't they will be looked at like this
* luckily things seldom breaks or become dangerous, as Software doesn't decay like play structures on a playground, ;-) 

[more good news: developer should fix other issues as well; and most]
developers will gladly address most issues in their code

* as most feel proud of what they build and want to ensure it works well
* thing is:
intro;

developers will gladly address most issues in their code, unless life gets in the way :-/

* the particular developer might be short on time
* stressed, sick, overwhelmed with reports or the boss forces the developer to focus on other things
* thing is: that happens frequently
* I guess you all known these from your life
[and then…]
then bad bug reports are the first developers will let fall through the cracks!

* happens quite often
* most kernel devs have a lot on their plate already
* it's in your hand to prevent this fate

[so let's reframe...]
intro;

developers will gladly address most issues in their code, unless life gets in the way :-/

[as it's more like this]
developers will gladly address most issues in their code, if you write a decent report!
developers will gladly address most issues in their code, if you write a decent report!

[that's why I'll tell you how to write one]
that's how you make most developers fix your bug, if they are able to

[in addition …]
you'll also learn when you can insist on a fix

* in case such a report isn't acted upon
intro;

and how to spot issues unlikely to be fixed

* safe yourself trouble

[which concludes the intro and brings us…]
[ act 1 ]
create a decent report

a) ensure your kernel is vanilla
b) ensure your kernel is fresh
c) ensure your kernel’s and system’s integrity
d) submit your report to the right place
e) depict the problem adequately
* let me please stretch the playground analogy a bit further
* for two reasons

[first: the example is to small; think of something bigger]
* these days Linux more like a really big amusement park
* even bigger than this
* let's call it "Linus land"
* no entry free
* doesn't need staff
* built, maintained, and constantly improved by volunteers
* that's more accurate

[the second: the kernel is immaterial ]
Linux is this more like an freely available ebook with instructions how to build your own "Linus land"
* maintained by volunteers
* and everyone everywhere can put the book into a gigantic 3d printer to build their own "Linus land"
  within a few minutes
* or update theirs in a few minutes

* sorry, a bit fat fetched, but good real life analogies for Software are hard to come by

[got that? okay]
* you visit some park build from the book
* your kid is injured on an water coaster a really good friend from school days designed
* you tell your friend, who's living 2000 km away and just got a kid
* friend checks the instructions for hours
* can't find anything and reluctantly flies over,
* notices that a few things look slightly different
* turns out: the one that built that park modified the book before building the park
* bigger water pipes and higher water pressure were used to "improve the performance", which lead to the accident
* your friend travels home really annoyed, he wasted money and hours and was blamed for an accident that's someone else fault
* you don't want to do that to a friend, don't you?
* that's way you don't want to do that to the volunteers that make Linux

[and that's why want to...]
create a decent report
ensure your kernel is vanilla

* vanilla == built from sources as distributed by kernel.org

[thing is: most kernels...]
Most kernels used in the wild are not vanilla, often heavily modified & enhanced. Especially those from RHEL, SLE, and Ubuntu kernels.

[ Such modification make... ]
makes most distro kernel's unsuitable for reporting issues Linux kernel devs.

* most kernel devs don't care at all about bugs with them
* small mods can have a big impact
* that's why some devs even reject bugs from distros that use a lightly patched kernel, like Fedora
you might want to report the issue to your Linux distributor

* warning: but most of the time it will be a dead end, as they don't have the resources to deal with all the reports they got

[that's why you might…]
or install a vanilla kernel yourself instead – for example a pre-built one

* pretty easy
* available for all the big distros
* and a few actually use them directly

[there is another option]
decent report; vanilla;

or compile a kernel yourself
hint: `make olddefconfig localmodconfig`
makes things easier and relatively fast

* lots of howtos on the net
* use those with the mentioned commands

[after installing vanilla...]
decent report; vanilla;

check if issue happens with a vanilla kernel, too
focus on this kernel in your report, forget the distro's 
mentioning the distro's even briefly 
often just complicates report unnecessarily
create a decent report
ensure your kernel is vanilla

* concludes this point
* next one is related
* you build a park and complain to your friend about a problem with an attraction designed by the friend
* checks unsuccessfully and flies over

* turns out: you used a two year old book that had a bug eliminated 18 months ago
* friend was not aware of the bug as it was caused by the infrastructure used by friend's attraction
* or accidentally fixed it with a big redesign or improvement
* or fixed it and forgot about it

[you don't want to do this to a friend, don't you?]
create a decent report

- ensure your kernel is vanilla
- ensure your kernel is fresh

* kernel changes a lot all the time
* bug might be fixed already

[what qualifies as fresh?]
test with latest mainline (aka -RC) release

* every bugfix has to land here first
* no, it's testing RCs is dangerous, they are pretty stable
* and you have backups, don't you?

[find it on kernel.org]
* ignore the big yellow field
* look at the top of table
* pick that version

[unless it looks like]
* then test the latest stable release

[but when a rc-kernel exists...]*
* you better avoid the latest stable
* developer will wonder if the bug was fixed already by someone
* already increases the chances your report might be ignored
* while it's not ideal to use such kernel, but not totally bad
* okay as fallback
* definitely better reporting with this than not at all
focus your report on the freshest kernel you tested
mentioning older briefly somewhere can be okay, but often just make report hard to grasp

[one more thing: don't use a longterm kernel]
* not even when a new version was released today

[because]
some bugfixes are never backported to stable/longterm kernel series

* sometimes that's simply too risky
* quite a few known bugs there
makes longterm (LTS) kernels quite unsuitable for reporting

* still better than not reporting at all
* but there is a high risk that your report will not lead to anything
* depends on the developer

[no rule without...]
decent report; freshness;

exception: regressions within a stable or longterm series something breaks 5.15.10 -> 5.15.11

* then it's okay to test the latest version from that series
create a decent report

ensure your kernel is vanilla
ensure your kernel is fresh

[another important aspect follows]
* accidents regularly in your own up2date Linus land
* say water and roller coasters stop somewhere along the track often
* friend can't explain things and flies over
* spot a mobile attraction in a corner of your park you allowed to come by every day and use the park's infra
* friend notices the workers of the mobile attraction even modified some water pipes in the park
* friend suspects the power grid is able to handle the extra load
* but is not allowed to look closer at the mobile attraction, as owners consider it their "trade secret"

[you don't want to annoy friends like that]
create a decent report

- ensure your kernel is vanilla
- ensure your kernel is fresh
- ensure your kernel's and system's integrity

* IOW: make sure kernel remains vanilla when used and is healthy
* kernel can detect some integrity problems itself

[check...]
decent report; integrity;

* this is how it should look like

[and not like...]
* everything other than a 0:
* kernel likely unsuitable for reporting

[one popular thing that can cause a "1" here]
Nvidia's proprietary graphics driver

* uses the kernel in unexpected ways and even changes it
* that's why most kernel developers don't care about reports with kernels using this drivers

[but the thing is]
decent report; integrity;

all out-of-tree drivers are a problem
incl. Nvidia's new open kernel driver

* kernel not vanilla anymore
* taint number for FOSS drivers just different

[that's why you...]
deinstall such drivers, reboot, check if issue still present and recheck the tainted flag!

* then you are free to proceed with reporting

[but note, there are]
decent report; integrity;

many other incidents can taint kernel
an "Oops" for example

* Oops = a critical error that could was detected, intercepted, and contained
* kernel can continue, but is an undefined state
* which can lead to subsequent faults
* and thus considered unreliable
* taint flag indicates that
[BTW: the oops shows if kernel is tainted]
[in the end: tainted kernels in the end...]
tainted kernels most of the time unsuitable for reporting bugs

[exception]
decent report; integrity;

big exception: the first Oops, warning, etc.

[the kernel's docs explain this in more detail]
* reporting issues
* section
* links to a page dedicated to tainted kernels
* which has a script and a table to decode the taint number/flags
### Table for decoding tainted state

<table>
<thead>
<tr>
<th>Bit</th>
<th>Log</th>
<th>Number</th>
<th>Reason that got the kernel tainted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>GP</td>
<td>1</td>
<td>proprietary module was loaded</td>
</tr>
<tr>
<td>1</td>
<td>/F</td>
<td>2</td>
<td>module was force loaded</td>
</tr>
<tr>
<td>2</td>
<td>/S</td>
<td>4</td>
<td>kernel running on an out of specification system</td>
</tr>
<tr>
<td>3</td>
<td>/R</td>
<td>8</td>
<td>module was force unloaded</td>
</tr>
<tr>
<td>4</td>
<td>/M</td>
<td>16</td>
<td>processor reported a Machine Check Exception (MCE)</td>
</tr>
<tr>
<td>5</td>
<td>/B</td>
<td>32</td>
<td>bad page referenced or some unexpected page flags</td>
</tr>
<tr>
<td>6</td>
<td>/U</td>
<td>64</td>
<td>taint requested by userspace application</td>
</tr>
<tr>
<td>7</td>
<td>/D</td>
<td>128</td>
<td>kernel died recently, i.e. there was an OOPS or BUG</td>
</tr>
<tr>
<td>8</td>
<td>/A</td>
<td>256</td>
<td>ACPI table overridden by user</td>
</tr>
<tr>
<td>9</td>
<td>/W</td>
<td>512</td>
<td>kernel issued warning</td>
</tr>
<tr>
<td>10</td>
<td>/C</td>
<td>1024</td>
<td>staging driver was loaded</td>
</tr>
<tr>
<td>11</td>
<td>/I</td>
<td>2048</td>
<td>workaround for bug in platform firmware applied</td>
</tr>
<tr>
<td>12</td>
<td>/O</td>
<td>4996</td>
<td>externally-built (&quot;out-of-tree&quot;) module was loaded</td>
</tr>
<tr>
<td>13</td>
<td>/E</td>
<td>8192</td>
<td>unsigned module was loaded</td>
</tr>
<tr>
<td>14</td>
<td>/L</td>
<td>16384</td>
<td>soft lockup occurred</td>
</tr>
<tr>
<td>15</td>
<td>/K</td>
<td>32768</td>
<td>kernel has been live patched</td>
</tr>
<tr>
<td>16</td>
<td>/X</td>
<td>65536</td>
<td>auxiliary taint, defined for and used by distros</td>
</tr>
<tr>
<td>17</td>
<td>/T</td>
<td>131072</td>
<td>kernel was built with the struct randomization plugin</td>
</tr>
</tbody>
</table>

[there is more about integrity]
create a decent report

- ensure your kernel is vanilla
- ensure your kernel is fresh
- ensure your kernel's and system's integrity

[continued]

* there are things the kernel can't detect
* that's why you better want to think about a few other things as well
is your hardware working reliably and as specified?

memtest: great idea!
overclocking: stupid idea!
issue with file-system?
fsck the volume!
decent report; integrity;

check `dmesg -H`
look out for anything red or bold

[looks like this]
* it might tell you what's wrong
* might give you an error msg to google
* and save everyone a lot of time
create a decent report

ensure your kernel is vanilla
ensure your kernel is fresh
ensure your kernel's and system's integrity
* you have a valid problem but only mention it on a school reunion where the friend later got pretty drunk and headed off with love interest from the school days
* or you reported it via a chat, message board, or forum on a website you know your friend used to visit when you both were young
* even after months or years, your friend didn't do anything to fix the problem
* that's your fault, as you friend might not visit the website anymore
* and maybe someone else is responsible anyway these days
create a decent report

- ensure your kernel is vanilla
- ensure your kernel is fresh
- ensure your kernel's and system's integrity
- submit your report to the right place

* web-forums definitely won't work
* distro's bug tracker often a dead end as well

[sadly. most of the time bugzilla.kernel.org is the wrong place, too]
* might looks like the central bug tracker
* but it's not, which you lean when you follow that link
bugzilla situation: it's complicated
* set up by some people that thought it was a good idea
* some devs liked it and started using it
* but many (most?) devs never liked the idea
* didn't really fit into the email based work-flow
* the idea was to have volunteers as go-between for such subsystems/maintainers
* that never really worked out
* that's why even today a lot of reports never reach the responsible developers (and are thus ignored)

* that's why bugzilla.kernel.org often is a bad idea [instead do, what...]
Check where you need to report your issue

Locate the driver or kernel subsystem that seems to be causing the issue. Find out how and where its developers expect reports. Note: most of the time this won’t be bugzilla.kernel.org, as issues typically need to be sent by mail to a maintainer and a public mailing list.

It’s crucial to send your report to the right people, as the Linux kernel is a big project and most of its developers are only familiar with a small subset of it. Quite a few programmers for example only care for just one driver, for example one for a WiFi chip; its developer likely will only have small or no knowledge about the internals of remote or unrelated “subsystems”, like the TCP stack, the PCIe/PCI subsystem, memory management or file systems.

Problem is: the Linux kernel lacks a central bug tracker where you can simply file your issue and make it reach the developers that need to know about it. That’s why you have to find the right place and way to report issues yourself. You can do that with the help of a script (see below), but it mainly targets kernel developers and experts. For everybody else the MAINTAINERS file is the better place.

How to read the MAINTAINERS file

To illustrate how to use the MAINTAINERS file, let's assume the WiFi in your laptop suddenly misbehaves after updating the kernel. In that case it's likely an issue in the WiFi driver. Obviously it could also be some code it builds...
BT8XX GPIO DRIVER

Mail: Michael Buesch <m@bues.ch>
Status: Maintained
Web-page: http://bu3sch.de/btgpio.php
Files: drivers/gpio/gpio-bt8xx.c

BTRFS FILE SYSTEM

Mail: Chris Mason <clm@fb.com>, Josef Back <josef@toxicpanda.com>, David Sterba <dsterba@suse.com>
Mailing list: linux-btrfs@vger.kernel.org
Status: Maintained
Web-page: http://btrfs.wiki.kernel.org/
Patchwork: http://patchwork.kernel.org/project/linux-btrfs/list/
chat: irc://irc.libera.chat/btrfs
SCM: git git://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux.git
Files: filesystems/btrfs/fs/btrfs/include/linux/btrfs*include/uapi/linux/btrfs*

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/MAINTAINERS

[most of the entries]
* mention the maintainer and the list
* just sent your report by mail there
* always CC the lists!
* most prefer this way and it should always work
* some subsystems uses a bug-tracker
* MAINTAINERS file mentions those few that do
* mainly ACPI, PCI, and PM
* about 20 out of more than 2000 entries

[there are also a few that use other bug-trackers]
RADEON and AMDGPU DRM DRIVERS

Mail: Alex Deucher <alexander.deucher@amd.com>, Christian König <christian.koenig@amd.com>, Pan, Xinhui <Xinhui.Pan@amd.com>
Mailing list: amd-gfx@lists.freedesktop.org
Status: Supported
SCM: git https://gitlab.freedesktop.org/aqd5f/linux.git
bugs: https://gitlab.freedesktop.org/drm/amd/-/issues
chat: irc://irc.oftc.net/radeon
Files: drivers/gpu/drm/amd/drivers/gpu/drm/radeon/ include/uapi/drm/amdgpu_drm.h
       include/uapi/drm/radeon_drm.h

RADEON FRAMEBUFFER DISPLAY DRIVER

Mail: Benjamin Herrenschmidt <benh@kernel.crashing.org>
Mailing list: linux-fbdev@vger.kernel.org
Status: Maintained

* graphics drivers for AMD, Intel, etc.
decent report; right place;

sadly MAINTAINERS contains more than 2000 entries:-/

* why are things so complicated and bugzilla.kernel.org?
* not design, just happened over time
* and no volunteer in sight to bring order into this
create a decent report

ensure your kernel is vanilla
ensure your kernel is fresh
ensure your kernel's and system's integrity
submit your report to the right place

[brings us to the last point]
* imagine your friend showing you a bug report from someone you both both went to school with, but never really liked

* you read subject and first para of a report and don't get the slightest idea what this is all about
* and the whole text is confusing and full of unnecessary or distracting details
* and has five attachments and ten links
* and is written in a unfriendly, demanding, and bearish way
* reminder, friend can ignore this without consequences
* would you suggest to do that?
create a decent report

ensure your kernel is vanilla
ensure your kernel is fresh
ensure your kernel's and system's integrity
submit your report to the right place
depict the problem adequately
"how to write a good report" worth its own, quite long talk
decent report; depiction;

a balancing act
think of it as asking for a favor
a favor from someone that doesn't have to help you
a favor from a someone that might be stressed or really short on time
hence, make your depiction easy to grasp for recipients
describe the problem neither to brief nor as a novella
mention version, vanilla, and taint status

* avoids doubts
* mention environment (distro, hw if relevant)
upload & link clearly relevant logs or attach them
but *don't* overload the report!

* if something missing, developer will ask for it
A decent report; depiction;

often relevant: output from `dmesg` & `lspci -nn`;
maybe kernel's `.config`, too

* depends on the issue

[then]
decent report; depiction;

add two or three sentences summarizing the situation on top of your depiction

* really important to get right
* developers get a lot of reports
* many will stop reading after first para

[some don't even get that far]
use a even more condensed and crystal-clear depiction as subject

* also really important to get right
* most people will only read that
in general: don't over-think or overdo your report!
short report will often do
getting the basics right (vanilla, fresh
version, no taint, easy to grasp, ...)
is important

[ohh, and remember]
check for existing reports about the problem to join
check <insert favorite search engine>, lore.kernel.lorg/all/, and bugzilla.kernel.org

* You might wonder:
* shouldn't I have done this earlier
* correct, that's why you should do things in the order described by this talk
[instead]
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order: That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem; those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- Make sure it’s not the kernel’s surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down consciously how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of

* tries to catch local problems early
* reference section providing details when you need them

[document also tells you]
it tells you to check what kind of issue you deal with

* a some require a few additional steps
* there is another reason why you want to do that
* it determines what you can expect
* which is kinda important, too

[which brings us to]
[ act 2 ]
the kind of issue at hand

[one kind are those]
the kind of issue at hand

issues someone is obliged to address

* and there are three of those
[the first]
I. security vulnerabilities
II. devastating bugs
III. regressions

the kind of issue at hand

issues someone is obliged to address

security vulnerabilities

* will only happen to a few of you
* but if follow the reporting issue

[which will point you]
Security bugs

Linux kernel developers take security very seriously. As such, we’d like to know when a security bug is found so that it can be fixed and disclosed as quickly as possible. Please report security bugs to the Linux kernel security team.

Contact

The Linux kernel security team can be contacted by email at <security@kernel.org>. This is a private list of security officers who will help verify the bug report and
kind of issue; must fix;

the kind of issue at hand
issues someone is obliged to address
security vulnerabilities
devastating bugs

* not "the paint is off" somewhere

[something…]
kind of issue; mustfix; devastating;
something really really bad
data is lost or damaged,
hardware is bricked, ...
[luckily even more rare]
kind of issue; must fix; devastating;

make impact & urgency obvious in your report

[and in case it's not quickly acted upon, get Linus in the loop]
[which brings us to the third, more common type]
kind of issue; mustfix;

the kind of issue at hand
issues someone is obliged to address
security vulnerabilities
devastating bugs
regressions

[a regression is...]
something breaks when updating the kernel
say from 5.15 -> 5.16 or from 5.17.3 -> 5.17.4

[not allowed in Linux]
first rule of Linux kernel development: "we don't cause regressions"

* coined and enforced by Linus
* wants to take the fear out of updating
* they nevertheless happen frequently :-/
* sadly some of the reports even fall through the cracks :-/
* that why I volunteered as the kernel's regression tracker

[and built a bot]
Linux kernel regression status

[next] [mainline] [stable/longterm] [dormant] [resolved] [new] [all]

current cycle (v5.18.. aka v5.18-post), culprit identified
none known by regzbot

current cycle (v5.18.. aka v5.18-post), unknown culprit
none known by regzbot

previous cycle (v5.17..v5.18), culprit identified, with activity in the past three months
- ff04f4e0b05
  (v5.18-rc1)
  mmm: chiq_test runs 7 minutes instead of ~1 second. by Stefan Wahren
  Earliest & latest activity: 7 & 3 days ago. Noteworthy: [patch].
- f26b3f3a04611
  (v5.18-rc1)
  mmm: [mmu/page_alloc] f26b3f3a04611: netperf: Throughput_Mbps - 18.0% regression by kernel test robot
  Earliest & latest activity: 19 & 7 days ago. Noteworthy: [patch].

older cycles (v5.17), culprit identified, with activity in the past three months
- 44c577205876
  (v5.15-rc1)
  qia2xxx: tape drive not removed after unplug FC cable by Tony Batterby
  Earliest & latest activity: 3 & 1 days ago.
- b2af264ad3af
  (v5.16-rc1)
  bluetooth: HSP/HFP mSBC profile broken with QCA6174 by bugzilla-daemon@kernel.org
  Earliest & latest activity: 310 & 1 days ago. Noteworthy: [1], [2], [3], [4], [inprogress].
- bdd86c98239
  (v5.17-rc1)
  Xorg SEGV in Xen PV dom0 after updating from 5.16.18 to 5.17.5 by Marek Marczykowski-Górecki
  Earliest & latest activity: 25 & 1 days ago. Noteworthy: [1], [patch].
- 453e41085183
  (v5.17-rc1)
  thumb suspend broken on Lenovo T460p by Derek Dolney
  Earliest & latest activity: 48 & 1 days ago. Noteworthy: [1], [patch].

* ugly, isn't it? :-/

[funding from EU]
This website is part of a project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 871528.

https://linux-regtracking.leemhuis.info/about/

* many thx for that
* project ended
* could talk about regressions and tracking them for hours
* no time for it

[once again there is a document]
* new in 5.18
* mentioning everything important
[that among others includes]
kind of issue; must fix; regressions;

make it obvious your report is about a regression
kind of issue; mustfix; regressions;

CC for forward the report to regressions@lists.linux.dev

[note, there is some...]
fine print(1):

only userland interfaces matter

[it's thus not a regression if your
out-of-tree kernel module breaks]

* these modules use kernel-internal interfaces
kind of issue; mustfix; regressions;

fine print(2):
the build config of the newer kernel version must be similar to the older one

* otherwise optional new features might interfere
* say a new security technique blocking something a few very rare apps need
* the doc I mentioned explains you how to realize
you often will be asked to find the culprit yourself

* many bug only happen in a certain environment
* that why the change that causes often needs to be found by the reporter
* the aforementioned doc explains you how to do that with a bisection using "git bisec"
* sounds hard, but might only take an hour or two
* initial report without this is okay, as problem might be known already

[and the good thing is]
if you find the culprit, a fix is pretty much guaranteed

* and the responsible volunteer and subsystem will be known
* might be possible to revert it
* tell me if that doesn't work out
* that's why you really want to do that in case you face a regression!
the kind of issue at hand

issues someone is obliged to address
security vulnerabilities
devastating bugs
regressions

* enough about regressions and issues that have to be fixed
* just one more thing you might be wondering about
* who will take care of fixing such bugs
* for regressions it's the author of the culprit
* if MIA and for everything else it's the [maintainer]
[and sometimes this person]
the kind of issue at hand
issues someone is obliged to address
issues most likely to be ignored
the kind of issue at hand
issues most likely to be ignored
known deficits
Linux contains many incomplete drivers

* a basic, incomplete driver is way better than none at all

[sometimes these drivers are never improved, if...]
kind of issue; unlikely; deficits;

might lack a volunteer with enough time and/or motivation to improve it

[second reason for known deficits]
or some real-world issue prevents improvements

* example: Nouveau
* docs scarce
* firmware prevents using the full capabilities of the hardware

* what do these known deficits mean for your report? [if it looks like a missing feature]
check internet and docs for known deficits

* prevents wasting your time on preparing a report
* if in a doubt, send a quick "is this known" before writing a proper and lengthy report
the kind of issue at hand
issues most likely to be ignored
known deficits

[another reason why some bugs are ignored]
the kind of issue at hand

issues most likely to be ignored

known deficits

code without an active maintainer

* Linux contains quite a bit of such code
[and it remains]
code often remains, as it useful for people

* removing it would cause a regression, too
* "no regression rule" should ensure it nothings break
* if people like you and me tests and reports problems

[two different kinds of unmaintained code]
* nearly orphaned, but not fully
kind of issue; unlikely; w/o maintainer;

sending at least a quick brief report definitely a good idea

* the "odd fixer" or someone else might take care of it

[there is also fully orphaned code]
CAFE CMOS INTEGRATED CAMERA CONTROLLER DRIVER

Mailing list: linux-media@vger.kernel.org
Status: Orphan
SCM: git git://linuxtv.org/media_tree.git
Files: Documentation/admin-guide/media/cafe_ccic*
drivers/media/platform/marvell-ccic/

CAIF NETWORK LAYER

Mailing list: netdev@vger.kernel.org
Status: Orphan
Files: Documentation/networking/caif/drivers/net/caif/
kind of issue; unlikely; w/o maintainer;

sending at least a quick brief report likely worth it

* maybe you find others affected and can team up with them
the kind of issue at hand

issues someone is obliged to address
issues most likely to be ignored

* this concludes this section

[leaves the big and wibbly-wobbly area in between those two extremes]
the kind of issue at hand

issues someone is obliged to address
issues most likely to be ignored
all the other issues

[what matters here is quickly explained, as we discussed this in act 1 already]
kind of issue; unlikely; the rest;

the quality of your report!

[which brings us...]

grand finale

* summary
take this with you
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else, there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order. That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- Are you facing an issue with a Linux kernel, a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem; those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- Make sure it’s not the kernel’s surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down carefully how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of the

* looks a bit scary, but it is not
* tries to catch local problems early
* it's in your own interest to follow the steps

[to understand why things are as they are, always keep in mind]
takeaways;

almost all kernel developer are volunteers
takeaways;

they should act on every bug report, but can and will ignore bad reports
act accordingly and sent a decent report, then you'll be heard

[to do that, you]
takeaways;

(1) check what kind of issue you deal with, as it...
takeaways;

(a) might save you from wasting time on reporting known deficits
(b) tells you what to expect from developers

[in addition to that]
takeaways;

(2) do your homework
takeaways;

(a) test and report with a *vanilla* kernel
takeaways;

(b) test with a fresh mainline kernel
takeaways;

(c) rule out local interferences
(d) check MAINTAINERS to submit the report to the right place
(e) write a friendly and decent report easy to gasp for others
Step-by-step guide how to report issues to the kernel maintainers

The above TL;DR outlines roughly how to report issues to the Linux kernel developers. It might be all that’s needed for people already familiar with reporting issues to Free/Libre & Open Source Software (FLOSS) projects. For everyone else there is this section. It is more detailed and uses a step-by-step approach. It still tries to be brief for readability and leaves out a lot of details; those are described below the step-by-step guide in a reference section, which explains each of the steps in more detail.

Note: this section covers a few more aspects than the TL;DR and does things in a slightly different order: That’s in your interest, to make sure you notice early if an issue that looks like a Linux kernel problem is actually caused by something else. These steps thus help to ensure the time you invest in this process won’t feel wasted in the end:

- Are you facing an issue with a Linux kernel a hardware or software vendor provided? Then in almost all cases you are better off to stop reading this document and reporting the issue to your vendor instead, unless you are willing to install the latest Linux version yourself. Be aware the latter will often be needed anyway to hunt down and fix issues.
- Perform a rough search for existing reports with your favorite internet search engine; additionally, check the archives of the Linux Kernel Mailing List (LKML). If you find matching reports, join the discussion instead of sending a new one.
- See if the issue you are dealing with qualifies as regression, security issue, or a really severe problem; those are ‘issues of high priority’ that need special handling in some steps that are about to follow.
- Make sure it’s not the kernel’s surroundings that are causing the issue you face.
- Create a fresh backup and put system repair and restore tools at hand.
- Ensure your system does not enhance its kernels by building additional kernel modules on-the-fly, which solutions like DKMS might be doing locally without your knowledge.
- Check if your kernel was ‘tainted’ when the issue occurred, as the event that made the kernel set this flag might be causing the issue you face.
- Write down coarsely how to reproduce the issue. If you deal with multiple issues at once, create separate notes for each of:

* reporting issues takes care of this

takeaways;

chances then are pretty good someone will help you
takeaways;

and nearly perfect, if you report a bisected regression

[and in the end that is]
takeaways;

that's how you make the Linux developers fix kernel bugs they are able to fix

[which is in everybody's interest and makes everyone happy]
questions?
Thorsten Leemhuis

mail: linux@leemhuis.info
GPG Key: 0x72B6E6EF4C583D2D
social media: @kernellogger, @knurd42, @knurd42rhfc, @thleemhuis and @thleemhuisfoss on #twitter & #friendica

#EOF