Logs and Backtraces

How to provide meaningful problem reports

Guido Günther

2023-08-06

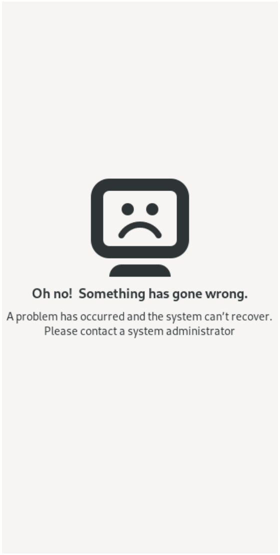
Outline

Topic

About me

- Debian Developer
- phosh / GNOME contributor
- Freelancing Free Software Developer
- Working with Purism on the Librem 5 Phone

This can happen


Oh no! Something has gone wrong.

A problem has occurred and the system can't recover.
Please contact a system administrator

Providing meaningful information, why?

- You want the issue fixed as it impacts you badly
- You want to be close to the fix to apply it quickly

Providing meaningful information, why?

- You want to know more on how the system works
- You want to move out of the consumer seat

Providing meaningful information, why?

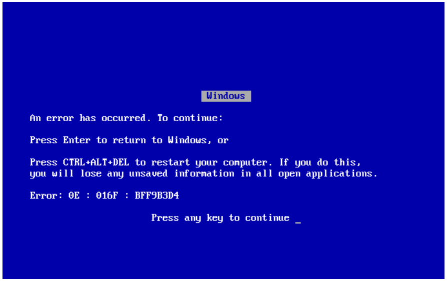
- Developer attention can be limited
  (focus on fixable bugs, assume good faith)

You might know more than you think

- Not a proprietary device

You might know more than you think

- Not a proprietary device

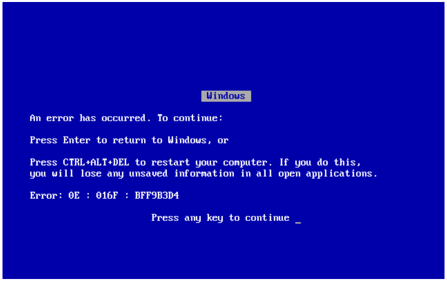


```
                        Windows

An error has occurred. To continue:

Press Enter to return to Windows, or

Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.

Error: 0E : 016F : BFF9B3D4

                    Press any key to continue _
```

You can (and are invited to) look at everything. Problem is where to look.

You might know more than you think

- Not a proprietary device



```
                         Windows

An error has occurred. To continue:

Press Enter to return to Windows, or

Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.

Error: 0E : 016F : BFF9B3D4

                      Press any key to continue _
```

You can (and are invited to) look at everything. Problem is where to look.

- Developers want you to find information

You might know more than you think

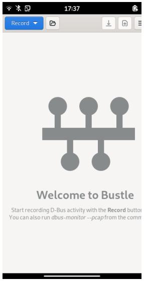
- Phone → Desktop → Server
  If you know how to debug Linux desktops or server you know a lot of tools already

You might know more than you think

- command line:
  htop, netstat, powertop, strace, ls{cpu,mem,...},
  dbus-monitor, gsettings, ...
  Recommended: Michael Prokop's Debugging for Sysadmin's talk

You might know more than you think

- GUI
  dconf-editor, d-feet, bustle, sysprof, . . .



Start recording D-Bus activity with the **Record** butto
You can also run *dbus-monitor --pcap* from the comm

But how?

1. What is the next bit of information that I can extract that helps me to pinpoint the cause?
2. Write that down

But how?

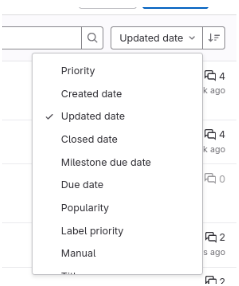
1. What is the next bit of information that I can extract that helps me to pinpoint the cause?
2. Write that down
   Then go back to one

But how?

- Check the bugtracker
  Once you have an idea what the component is.

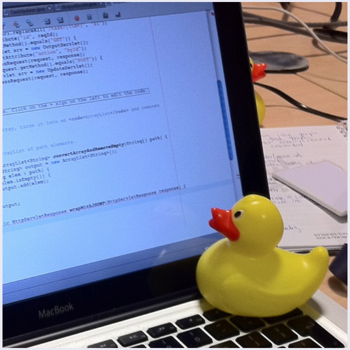
But how?

- Check the bugtracker
  Once you have an idea what the component is.
  Gitlab can sort by recent activity!



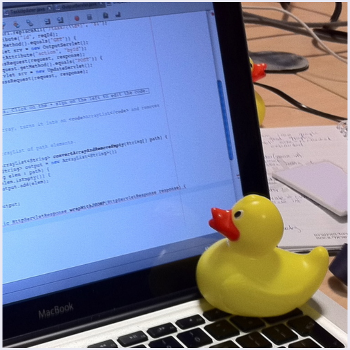
But how?

Try rubberducking



But how?

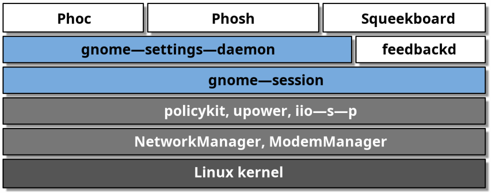
Try rubberducking



A bug report can be a rubber duck

Topic

High level overview



High level overview

- There's a user session and system services
- DBus is prevalent for inter process communication
- There's two: user and system
- Get an overview with d-feet or

```
busctl --system --list
busctl --user --list
```

High level overview

Hence there's two systemd daemons:

- `/sbin/init`
- `systemd --user`

Get an overview with

```
systemctl --user status
systemctl --system status

systemctl --user --failed
systemctl --system  --failed
```

High level overview

When you see a failure: system or user session?

Log output

- Getting Log Output
  No standard way on Linux. But

Log output

- Getting Log Output
  No standard way on Linux. But

```
journalctl --user
journalctl --system
```

Log output

- Getting Log Output

  No standard way on Linux. But

  ```
  journalctl --user
  journalctl --system
  ```

  See a pattern?

Log output

- Get more output: Startup Time
  - Environment variables

```
G_MESSAGES_DEBUG=all /usr/bin/gnome-calculator
```

  - Command line optons

```
chatty -vvvv
```

Log output

- Get more output: Runtime

```
kill -SIGUSR1 $(pidof /usr/libexec/phosh)
```

What works should be in the application's ‘README‘.

Log output

- Don't cite but copy information: Details are important

  "It says something like. . . "

- Provide the complete output

  It might not make sense to you or me but maybe to others.

Log output

- But there's more details

```
journalctl -o json |
  jq  --unbuffered
    'select(.GLIB_DOMAIN == "phosh-lockscreen")'
```

```
{
  "_AUDIT_LOGINUID": "1000",
  "__REALTIME_TIMESTAMP": "1691147633303845",
  "CODE_FUNC": "load_background",
  "MESSAGE": "Ignoring XML background
    'file:///usr/share/phosh/backgrounds/logo.xml'",
  "CODE_FILE": "../src/lockscreen.c",
  "_COMM": "phosh",
  "GLIB_DOMAIN": "phosh-lockscreen",
}
```

Log output

- The above gives you a good idea where to look at
- Can be educating as not (yet) programmer
- Will likely give a clue if you're on the right track
- For usable log domains:

```
  gbp import-dsc apt:phosh
  git grep G_LOG_DOMAIN
src/call-notification.c:#define G_LOG_DOMAIN "phosh-call
src/call.c:#define G_LOG_DOMAIN "phosh-call"
src/calls-manager.c:#define G_LOG_DOMAIN "phosh-calls-ma
src/connectivity-info.c:#define G_LOG_DOMAIN "phosh-conn
src/docked-info.c:#define G_LOG_DOMAIN "phosh-docked-inf
src/docked-manager.c:#define G_LOG_DOMAIN "phosh-docked-
```

Crashes

What is a crash

- Operating system ends the process: killed by a signal (usually SIGSEGV/11)
- Program hits an internal assertion (SIGABRT/6)
- Process exits (no crash dump)

Crashes

What is a crash or core dump

- Memory image and stack of the process at crash time
- Debugger gets a call stack
- Needs to be enabled
- Devices like GPUs can create their own core dumps

Crashes

- Investigate crash dumps
  again: first check the journal

```
Aug 06 11:57:45 foo systemd-coredump[6148]:
    [.] Process 6092 (gnome-calculato) of user
        1000 dumped core.
```

Crashes

- Investigate crash dumps
  Now we're onto something

  ```
  coredumpctl list
  ```

  ```
  TIME                                PID   UID   GID SIG
      COREFILE EXE
  Mon 2023-07-31 15:18:30 CEST  355561 1000 1000 SIGABRT
      present  /usr/bin/phoc
  Mon 2023-07-31 15:18:30 CEST  355573 1000 1000 SIGABRT
      present  /usr/bin/phoc
  Mon 2023-07-31 15:18:30 CEST  355576 1000 1000 SIGABRT
      present  /usr/bin/phoc
  ```

- Investigate crash dump

  ```
  echo bt | coredumpctl debug 355576
  ```

Crashes

- Debug information makes the backtrace more meaningful
- Debug info packages or debuginfod
- Compile options `-Dbuilttype=debug`

Crashes

- Demo

Crashes

```
dpkg -S libglib-2.0.so.0

echo "deb http://deb.debian.org/debian-debug/ sid-debug mai
    > /etc/apt/sources.list.d/debuginfo.list
apt update
apt install libglib2.0-0-dbgsym
```

Crashes

- Again: Use the source

Crashes

- In case of Segmentation faults (SIGSEGV) Might just be a symptom: `valgrind`

Practical contains

- Replicate on desktop
- Run phosh nested on desktop (Phosh Nested blog post)

Practical concerns

- Get yourself non-osk access

  ```
  apt install ssh
  systemctl disable ssh
  ```

  When needed

  ```
  systemctl start ssh
  systemctl stop ssh
  ```

Practical concerns

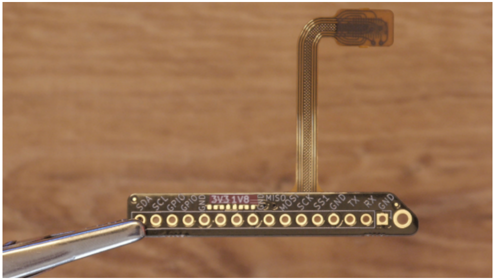
- Use usb-gadget
  Connecting via USB to laptop should give you a shell:

```
Bus 001 Device 010: ID 1d6b:0104 Linux Foundation Mult
```

```
screen /dev/ttyACM0 115200
```

Practical concerns

- Capturing early boot output: Serial console



What else

- Tracing (e.g. systemtap)
- /sys/kernel/debug/tracing

```
echo '1' > /sys/kernel/debug/tracing/\
                  events/tps6598x/enable
echo '1' > /sys/kernel/debug/tracing/tracing_on
cat /sys/kernel/debug/tracing/trace_pipe
```

Topic

Use the bug tracker

Keeps more consistent record than chat. Chat might be good to get an idea where to file the bug.

Avoid

- Dismissive wording (annoying, totally broken, . . .) (Developers are humans too, assume good faith)
- This is wrong because xyz makes it differently. It has to be done because foo does the same

Avoid

- plain: I'm seeing this too

Avoid

- plain: I'm seeing this too
  Add device information, what varies from what the original reporter experienced. Add the details you figured out.

Examples

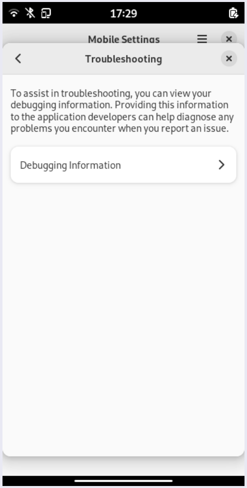
After updating gnome-calls from version to version on my foo phone it crashes when I do x. This is the backtrace.

What baseline information can I provide?

- Device
- Operating System and Version
- Software versions of the relevant components

What baseline information can I provide?

phosh-mobile-settings



To assist in troubleshooting, you can view your debugging information. Providing this information to the application developers can help diagnose any problems you encounter when you report an issue.

How to validate the fix?

- Build yourself. (Might be easier than you think thanks to meson)
- Development builds
- Ask your distro maintainers for help (they might already provide nightly builds somewhere)

Summary

- Bug chasing can be exciting
- Use the logs
- Try to identify affected component
- Look at crash/core dumps
- There's usually always more information
- Provide baseline information

Topic

Contact

- Mail: <agx@sigxcpu.org>
- Fediverse: agx@librem.one
- Matrix: @agx:librem.one
- IRC OFTC: agx

Image/Video references

- cc-by-sa-3.0:
  - Screenshots: myself using `https://gitlab.gnome.org/World/Phosh/phosh`
  - Rubberduck: Tom Morris `https://commons.wikimedia.org/wiki/File:Rubber_duck_assisting_with_debugging.jpg`
- Public Domain
  - Window 9X BSOD: Akhristov

Links

- Michael Prokop's: Debugging for Sysadmin's talk: `https://media.ccc.de/v/glt23-334-debugging-fr-sysadmins`
- Phosh Nested blog post: `https://phosh.mobi/posts/phosh-dev-part-0/`