Generating a Controled Software Environment with Debian Snapshot Archive

Arnaud Legrand, Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP

Software Environment Reproducibility, May 2021

Context

Personal History

The early days (1997) tried Redhat, Slackware, Debian

- Regularly compiled my kernel to have the right modules
- Feared everytime I had to reconfigure X
- Updates through a 56kbps modem or a Zip drive

Happy Debian User since ≈ 1998

- Switched to Ubuntu for a few months but I quickly got back to Debian!
- Debian stable versions are often outdated. I live on the edge with Debian unstable most of the time
- I install stuff whenever I need and upgrade every 2-3 months

Software Management

Open source by default (except for Nvidia drivers and video codecs)

> *Source is provided to this software because we believe users have a right to know exactly what a program is going to do before they run it.*
>
> *– Nmap Reference Guide*

Install everything through `apt` (since ≈ 1999)

- Several friends were Debian developpers

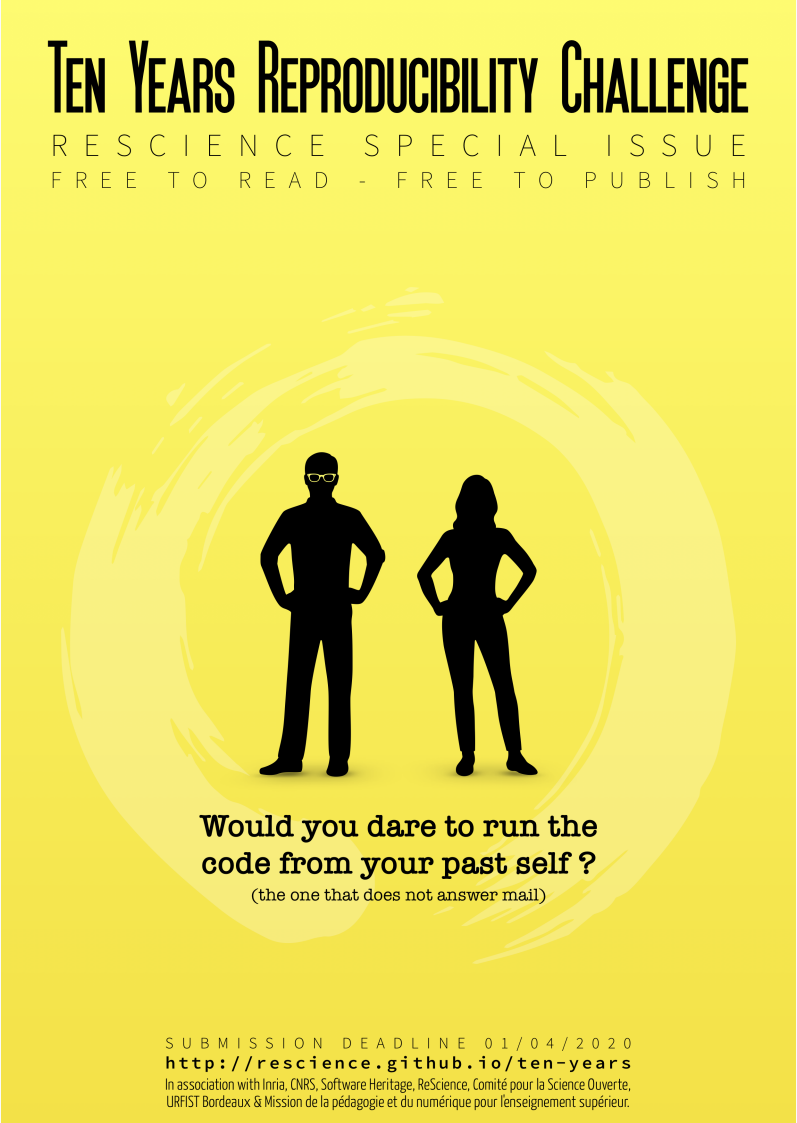
> *To what extent should one trust a statement that a program is free of Trojan horses? Perhaps it is more important to **trust** the people who wrote the software.*
>
> *– Ken Thompson, 1984*

- The only code I compile is mostly my own (in OCAML, C, perl)
- For a while, when using R, I had to install recent libraries with `install.packages`
  - Works well but this is bad practice
  - I generally "hate" python and the pip ecosystem

In my perspective software environment was never a real problem

- Just run `apt update; apt install`
- Enjoy all the most up-to-date software with bugfixes

10 YEARS REPRODUCIBILITY CHALLENGE



http://rescience.github.io/ten-years/

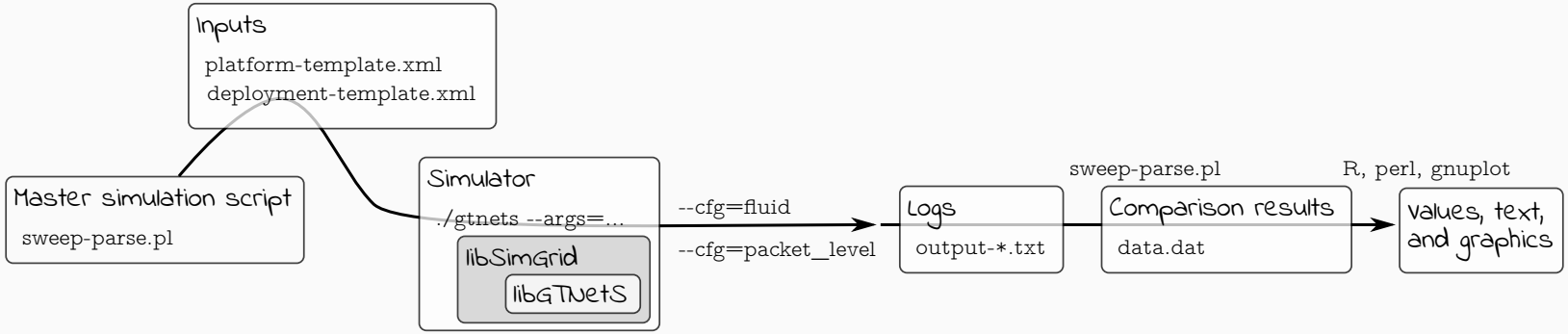
A 2009 Article

"Accuracy Study and Improvement of Network Simulation in the SimGrid Framework", Velho and Legrand, SimuTools 2009

Simulation comparison coarse-grain (SimGrid) vs. fine-grain (GTNetS)

- Compare a fluid bandwidth sharing model (`SimGrid`) with a packet-level model (`GTNetS`) of TCP
- Mostly simulations: 3 kind of network topologies, parameterized and randomized workload

Simulation workflow



Already a reproduction effort *"Speed and Accuracy of Network Simulation in the SimGrid Framework"*, Fujiwara and Casanova, ValueTools 2007

- We could not obtain the exact same results so we took notes and tried to make stuff available

Getting my Hands Dirty

Step 1: Getting the pieces together

Finding out data and instructions

- Do not trust your memory, ask the PhD student and his webpage!
- Provides `GTNetS`, the patches to apply and detailed instructions for SimGrid integration
- Experiments run in late 2008; previous SimGrid release $\approx$ 2007

Figuring out software versions

- GTNets: discontinued development since October 2008
- SimGrid dates from 2000 and Martin Quinson cares a lot:
  - Was hosted on `gforge.inria.fr`, then `github.com` (2010) and `gitlab.inria.fr`, and now `framagit`
  - SimGrid v3.3? Stable releases (`autotools`) are only on `gforge.inria.fr`

Step 2: Building an environment and compiling code

Shallow (but useful) description (README)

```
Author        : Pedro Velho
last modified : 03/11/2008

1. Disclaimer [...]
2. Short History [...]
3. Directory Structure [...]
4. Global System Requirements
   - GTNets patched simgrid version, we kindly provide GTNets with
     patches in the simgrid contrib svn repository [FIXME]
   - SimGrid, configured and compiled with GTNets support [FIXME]
   For plotting graphs and explore the data:
   - R - the gnu version of S [FIXME]
   - Gnuplot [FIXME]
```

Dependencies

- perl, R, gnuplot: easy!! ☺
- simgrid: easy (autotools, C, a bit of C++ for GTNets) ☺
- gtnets: Qt3!!! ☹

I really need a 2009 software environment!!!

Let's grab a docker image from a 10 years old distro and consider Pedro used Debian stable (codename Lenny back then)

```
docker search debian-lenny
```

```
NAME                        DESCRIPTION
pblaszczyk/debian-lenny     5.0.10 amd64
lpenz/debian-lenny-amd64    Debian 5.0.10 Released 10 March 2012 for amd…
lpenz/debian-lenny-i386     Debian 5.0.10 Released 10 March 2012 for i386
...
```

Then write the Dockerfile

Docker file (1/2)

```
FROM lpenz/debian-lenny-i386

# Software dependencies
RUN apt-get update \
        && apt-get install -y --force-yes gcc g++ make wget \
                   unzip subversion patch less libqt3-mt \
                   libqt3-headers libqt3-mt-dev qt3-dev-tools

# Downloading GTNetS
RUN cd /root; svn checkout \
                  svn://scm.gforge.inria.fr/svn/simgrid/contrib/trunk/GTNetS/
# Downloading SimGrid
RUN cd /root; wget \
    https://gforge.inria.fr/frs/download.php/file/21430/simgrid-3.3.tar.gz
```

Docker file (2/2)

```
# Building GTNetS
RUN cd /root/GTNetS/; unzip gtnets-current.zip ; tar zxvf gtnets-current-patch.tgz
RUN cd /root/GTNetS/gtnets-current; cat ../00*.patch | patch -p1
RUN cd /root/GTNetS/gtnets-current; ln -sf Makefile.linux Makefile && \
    make depend && make opt

# Installing GTNetS
RUN cd /root/GTNetS/gtnets-current/ && \
    mkdir -p /root/usr/lib/ &&         \
    ln -sf `pwd`/libgtsim-opt.so /root/usr/lib/libgtnets.so && \
    ln -sf `pwd`/libgtsim-opt.so /usr/lib/libgtnets.so &&      \
    mkdir -p /root/usr/include/ &&    \
    cp -fr SRC/*.h /root/usr/include/

# Building SimGrid
RUN cd /root/ && tar zxf simgrid-3.3.tar.gz
RUN cd /root/simgrid-3.3/ && \
    ./configure --with-gtnets=/root/usr/ && \
    export LD_LIBRARY_PATH=/root/usr/lib/libgtnets.so && \
    ldconfig && \
    make

RUN apt-get clean
```

Debian snapshot-archive...

It all went super smooth thanks to the instructions in the README ☺

What's broken in this Dockerfile?

- `gforge.inria.fr` will die in a few months
  - Use Software Heritage instead (zip files in an svn... shame ☹)
  - TODO: Save all simgrid stable archives (zenodo ?)
- `FROM lpenz/debian-lenny-i386`
  - `/etc/apt/sources.list` indicates:

    ```
    deb http://archive.debian.org/debian lenny main
    ```

    Last version (5.0.10) dates from March 2012
  - Debian Snapshot

    ```
    deb https://snapshot.debian.org/archive/debian/20091004T111800Z/
    lenny main
    ```

Discover the `debuerreotype`

```
debuerreotype-init rootfs testing 2009-05-01-T03:27:08Z
```

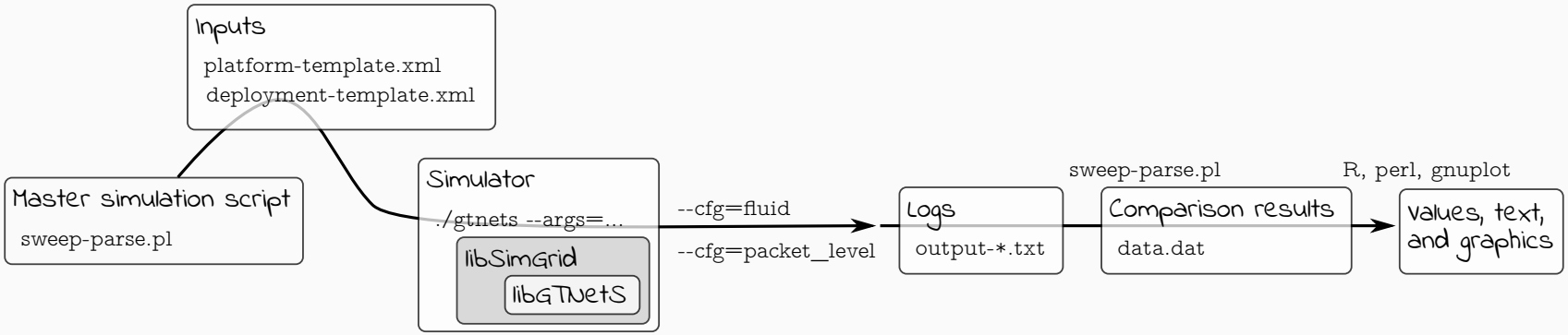
A few important things to know:

- Requires `root` privileges (see issue 66)
- `--keyring=/usr/share/keyrings/debian-archive-removed-keys.gpg`
- `vsyscall=emulate` (see issue 80)

This is the right way to proceed

but I was in the plane and fought against this `vsyscall=emulate` thing when I tried

Step 3: run stuff



- A home-made perl script with hard-coded paths (painful but worked)
- Obtained the same intermediate results (thanks to the `bin/ log/ dat/` organization)
- Long computation ⇝ stopped it before the end
  - But I could run the analysis and obtained similar output (linear regression, 3D plot)

- Only ran the simulation and the analysis
  - no workload generation (no information was given, but it would have required `Java` and no information on the seed was kept)

Conclusion

Conclusion on the challenge

I greatly underestimated:

1. Link rot (simgrid, gforge closure, webpage)
2. Lack of automation (org-mode or snakemake would have made everything much simpler)
3. Environment reconstruction (limited information)
   - Yet, I could easily rebuild a working environment (Debian)
   - Was it the exact same code ?
     - ¯_(°_o)_/¯ gave the same results for the parts I ran 😉

Conclusion on Docker

Docker is easy to use but does not provide with any help/warranty!

Docker can be quite helpful for the average scientist to build reproducible environments provided a few precautions are taken:

1. Regularly work in a container with minimal dependencies
   - Also separate the code from the data (, which can be painful)
2. Use high quality and trusted software packages (Debian)

   To what extent should one trust a statement that a program is free of Trojan horses? Perhaps it is more important to trust the people who wrote the software. – *Ken Thompson, 1984*
3. Freeze the sources (debian snapshot-archive)
4. Document the creation (the DockerHub is not an archive!)

What if I had to redo this today (e.g., with snakemake)

- ship snakemake in my docker image ? Nope!
- snakemake in docker running my code in singularity ?