I’m building an automated reporting system which generates PDF reports. My approach is to use R Markdown to write the report and render to PDF using the excellent `{pagedown}` package.

Ultimately the system needs to be packaged in Docker and deployed in the cloud.

Setup

To illustrate what I’m doing, we’ll use a simple dummy document, `test.Rmd`.

```
---
title: "Test Document"
output: html_document
---

This is a test document.
```

To convert this into PDF run:

```
pagedown::chrome_print("test.Rmd")
```

I got this all running in my local environment quite easily. However, I ran into a snag when trying to package the code with Docker.

The Chrome Problem

I created a `Dockerfile` based on `rocker/r-ver`, adding Chrome and `{pagedown}`, then copying across `test.Rmd`.

```
FROM rocker/r-ver:4.1.0

RUN apt-get update -qq && \
    apt-get install -y -qq --no-install-recommends \
        libz-dev \
        libpoppler-cpp-dev \
        pandoc \
        curl

RUN curl -L http://bit.ly/google-chrome-stable -o chrome.deb && \
    apt-get -y install ./chrome.deb && \
    rm chrome.deb

RUN install2.r --error --deps TRUE pagedown

COPY test.Rmd .
```

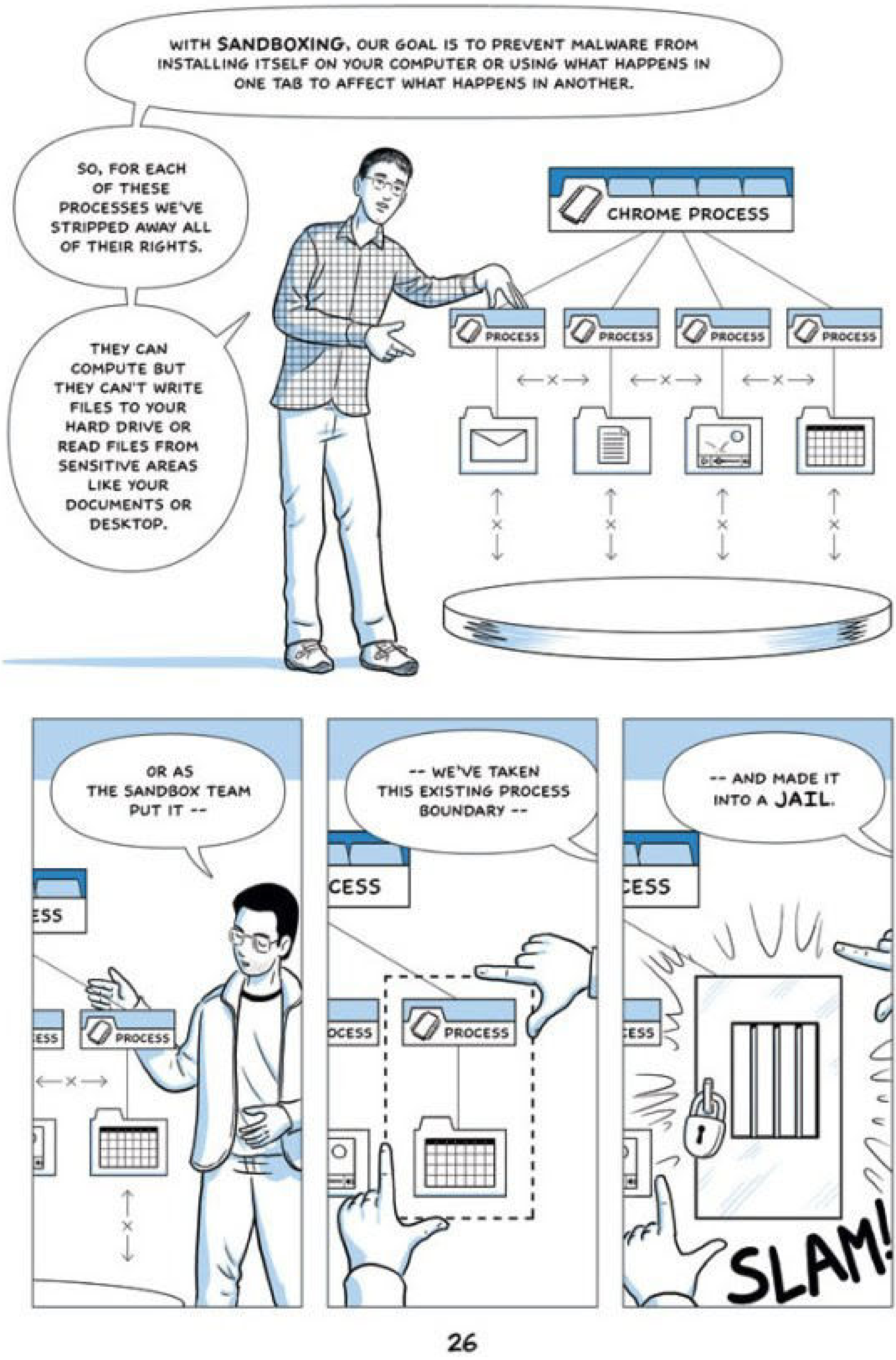
Running `pagedown::chrome_print()` from a container produces an error.

```
Error in is_remote_protocol_ok(debug_port, verbose = verbose) :
  Cannot find headless Chrome after 20 attempts
```

Bummer!

What's going on here? Chrome is clearly installed, so why is R failing to find it? Well, I think what's happening is R is actually *finding* Chrome it but it's failing to *run* it.

And the problem appears to relate to Chrome's sandbox. This is a safety feature built into Chrome. However, in this instance we need to circumvent it to get things working.


WITH SANDBOXING, OUR GOAL IS TO PREVENT MALWARE FROM INSTALLING ITSELF ON YOUR COMPUTER OR USING WHAT HAPPENS IN ONE TAB TO AFFECT WHAT HAPPENS IN ANOTHER.
SO, FOR EACH OF THESE PROCESSES WE'VE STRIPPED AWAY ALL OF THEIR RIGHTS.
THEY CAN COMPUTE BUT THEY CAN'T WRITE FILES TO YOUR HARD DRIVE OR READ FILES FROM SENSITIVE AREAS LIKE YOUR DOCUMENTS OR DESKTOP.
CHROME PROCESS
PROCESS
PROCESS
PROCESS
PROCESS
OR AS THE SANDBOX TEAM PUT IT --
-- WE'VE TAKEN THIS EXISTING PROCESS BOUNDARY --
-- AND MADE IT INTO A JAIL.
SLAM!
26

Cartoon from the Google Chrome Comic.

Print to PDF using Docker

The `{pagedown} documentation` suggests two approaches to solving the problem.

Use `--no-sandbox` Argument

One solution is to send `--no-sandbox` to Chrome via the `extra_args` argument.

```
pagedown::chrome_print("test.Rmd", extra_args = c("--no-sandbox"))
```

This is perfectly reasonable. And it works! So, from a pragmatic perspective, it's perfect.

However, I'm going to be making a bunch of calls to `pagedown::chrome_print()` and, in the interests of simplicity, I'd prefer not to have to provide the extra argument every time.

Specify Security Options

An alternative is to use `docker run` with `--security-opt` to specify some custom security options. Again, this works, but it's just added complexity! Also, I prefer a solution that's actually baked into the Docker image.

A Chrome Solution

A Chrome Shim

I created a BASH script shim, `google-chrome`, with the following contents:

```
#!/bin/bash

/usr/bin/google-chrome --no-sandbox $*
```

It basically executes Chrome, passing along all command line arguments plus `--no-sandbox`.

I made the script executable.

```
chmod u+x google-chrome
```

An Environment File

I also added the root folder, `/`, to the `PATH` environment variable in a file called `Renviron`.

```
PATH="/:${PATH}"
```

Tweaking the Dockerfile

The `Dockerfile` requires two small tweaks:

- copy the `google-chrome` script across to `/usr/local/bin/`; and
- copy `Renviron` as `.Renviron`.

The revised `Dockerfile` looks like this:

```
FROM rocker/r-ver:4.1.0

RUN apt-get update -qq && \
    apt-get install -y -qq --no-install-recommends \
        libz-dev \
        libpoppler-cpp-dev \
        pandoc \
        curl
```

```
RUN curl -L http://bit.ly/google-chrome-stable -o chrome.deb && \
    apt-get -y install ./chrome.deb && \
    rm chrome.deb

RUN install2.r --error --deps TRUE pagedown

COPY test.Rmd .

COPY Renviron /.Renviron
COPY google-chrome /usr/local/bin/
```

The actual Chrome executable is located at `/usr/bin/google-chrome`. But `/usr/local/bin/` comes before `/usr/bin/` in `PATH`, so when R looks for Chrome it finds the shim script first. This in turn adds in the `--no-sandbox` argument and my PDFs are then happily built by `pagedown::chrome_print()`.

A Chromium Solution

How about using Chromium instead of Chrome? We need to make some changes to the `Dockerfile` to get Chromium installed.

```
FROM rocker/r-ver:4.1.0

# Install Chromium with apt not snap!
COPY bionic-updates.list /etc/apt/sources.list.d/
COPY chromium-deb-bionic-updates /etc/apt/preferences.d/

RUN apt-get update -qq && \
    apt-get install -y -qq --no-install-recommends \
        libz-dev \
        libpoppler-cpp-dev \
        pandoc \
        chromium-browser

RUN install2.r --error --deps TRUE pagedown

COPY test.Rmd .

COPY Renviron /.Renviron
COPY chromium-browser /usr/local/bin/
```

Adding in a shim script which supplies the `--no-sandbox` option to Chromium and we're sorted! 🚀

Admittedly this is a relatively deep rabbit hole for such a simple (and probably inconsequential) issue. But it was fun and instructive.