Bioshadock

•••

O. Sallou - IRISA

Containers

Docker, LXC, Rkt and Co

Docker is the current "leader" in container ecosystem but not alone in ecosystem

Rkt compatible with Docker images

All those technologies are Linux based only, this is not virtualization. They make use of the Linux kernel running on host and Linux Kernel features such as cgroups, usernamespace etc...

Docker can run Linux containers on Windows and Mac, but not Windows containers on Linux!

Easy to build with a simple Makefile-like

Main features:

- Process isolation
- Limit resource usage (optional)
- Bridged network with port mapping or real IP address
- Image layering and sharing
  - several images use the same layers (read only) and user write in container on a new/personal layer
- Image contains an Operating System
  - Possibility to run multiple OS in containers on the same host (Fedora, Debian etc.) in parallel.
- User is root in container (but may be mapped to a different user on host)
- Volume sharing (mount a local directory in rw/ro ACL in container)
- Limited capabilities (by default) in container
  - Example: user is root but cannot mount a local host directory.

Containers needs root access on local host

Workaround:

- Start a virtual machine and install Docker, Rkt ...

How containers can help scientists and research ?

Reproducibility

How to reproduce research results?

Software can be difficult to be installed:

- specific dependencies (old or too recent gcc for example)
- lack of documentation
- Which compilation options to use (depending on OS, etc...) to be in the same condition?
- Sometimes very long compilation process (which may fail)
- Hard coded path/values in makefiles
- Uh? What is a Makefile? ;-)

So, how can I reproduce the result showed in this very interesting research article in my lab?

With containers, software is ready to run

No kidding ! Just execute on your computer or lab facility:

```
docker pull mylab/myextrasoft   <=  download the container image

docker run mylab/myextrasoft -i myfile -o myresult <= run the tool with arguments
```

- Starts in seconds
- Can run multiple (hundreds...) in parrallel
- Software compiled/preconfigured
- Same setup than the one used in the research article

Ohhh, so cool! Where do I get it?

On Docker Hub or other registries if you are lucky and if you can find it!

Maybe in a git repository with a Dockerfile so that you build it yourself.....

Technology is here BUT upstream authors MUST provide a cnotainer image with their software, research result.

It is easy to build, easy to share with registries and requires very few extra work (really!!).

Bioshadock central registry, or on premise….

Registry is open source, possibility to install a local Bioshadock instance for private usage OR use the central registry (http://docker-ui.france-bioinformatique.fr)

Only open source software on central registry. If usage is restricted (academic only), it is specified in tool description.

Licensed software CAN be containerized on a local instance if:

- Explicit software redistribution is allowed by upstream
- No license file must be included in container (end-user must provide his licence information himself).

Transparency

I can reproduce results but….

Containers can be black boxes

- Software is installed/configured but:
- Where is code source?
- How can it compiled?
- What if I want to redo the same?
- How can I extend the container?

Upstream must show how they made their containers

Containers can be build with kinda Makefiles, eg. Dockerfile with Docker.

Containers MUST be built automatically from those Makefiles with:

- Download source
- Compile
- Configure steps

User can use blindly the container or reproduce the steps to get the same result!

Scientific reproducibility NEEDS transparency

Why bioshadock?

Registries

Container images are stored in a registry ie. a marketplace to find/download them

DockerHub, Rkt registries, etc... provide public access to such marketplace

But lots of containers from bioinformatics to databases, monitoring, ...

Many tools with no description

Lots of blackbox tools (no transparency)

A registry for bioinformatics

Bioshadock is the official IFB (French Bioinformatics Institute) Docker registry.

IFB is the French Elixir node.

It provides a central place targeting bioinformatics tools and libraries.

For bioinformatics

Add business value

Along the registry (place to download/storage), Bioshadock provides a web interface to create and search for tools.

It also provides tags where one can add EDAM or other bioinformatics field information to ease search and give user additional information

Possibility to add a CWL description along the tool description

Curation and limited creation access

Because we do not want to see tons of containers for the same tool

Because we want bioinformatics only

Because we want Dockerfiles and not black boxes (unless really needed)

Anyone can download a container (though private repositories are available)

But only allowed users can create a repository (place to put a container)

Registry can be integrated to a local LDAP

Need an access? Simply ask to the support team

Automatic

Automatic builds

Once a repository is created, from a Dockerfile or a git repository, once can ask to build a container (manually or from a hook)

Background processes will build and tag the containers. Container will be available only if build process succeeds.

For a build, all files needed for the build (source code, configuration files, ...) must be available (remote location, in git repository), ensuring transparency and reproducibility.

Security

Can container be safely run in my facility?

Container is based on an operating system.

All systems have vulnerabilities (ssh, certificates, ...) with CVE (Common Vulnerabilities and Exposures)

User MAY not care CVEs on local computer, but system administrators SHOULD care....

Bioshadock scans all containers against known CVEs and display a report in the web interface.

User can decide, or not, to use the container. Or ask upstream author to update his container...

Import from other sources

Packages…

Software may already be packaged in a Linux distribution (Debian deb, CentOS rpm, etc.)

Why should I care of containers? Why not a simple apt-get install ?

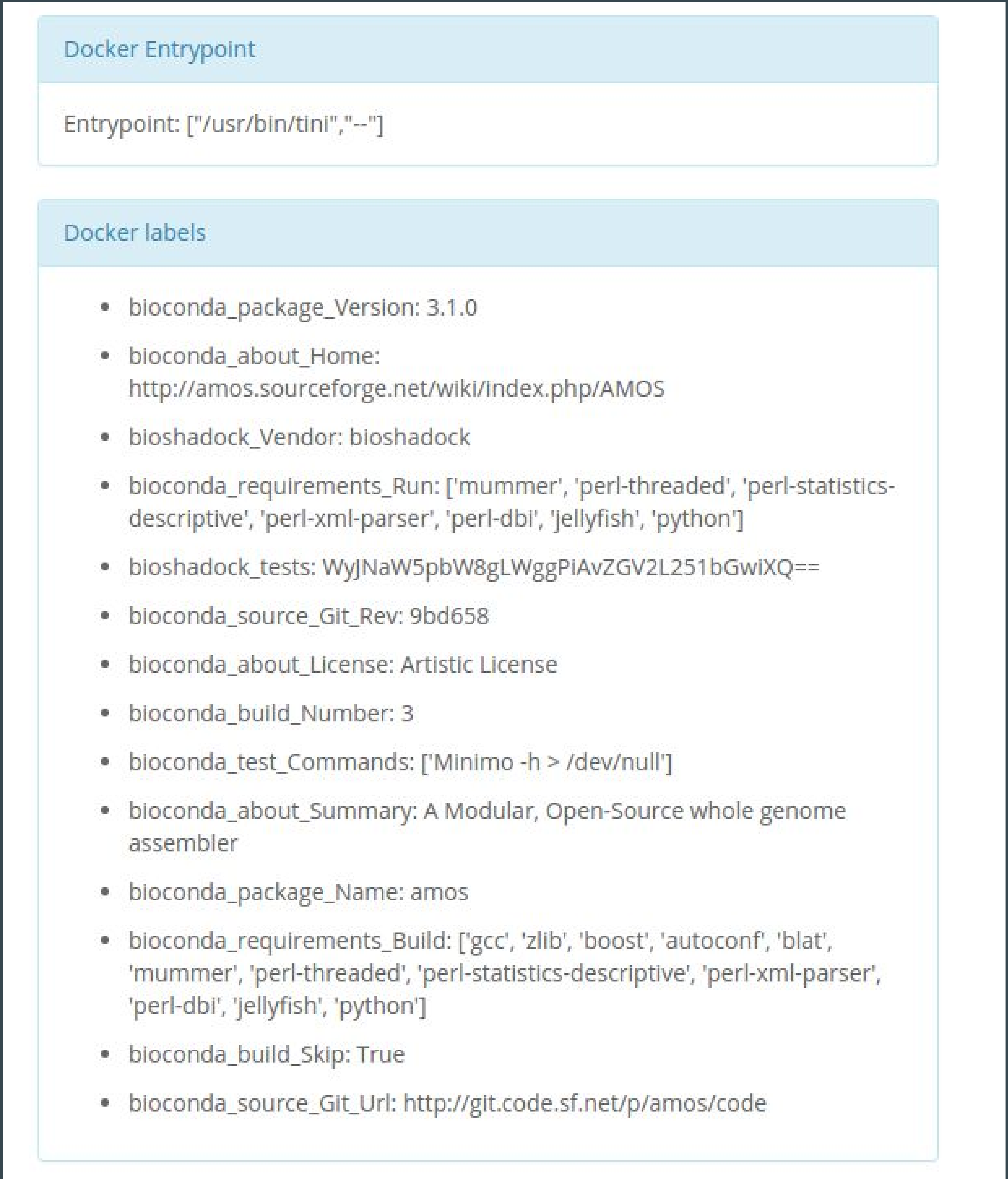
With containers, one can use multiple versions of a tool (Conda tries to fix this but needs to load environments, not easy to mix all configurations, needs frequent install/uninstall for cleanup)

Package may not provide the version I want.

But packages provide clean install, metadata, license checks...

Bioshadock automatically creates and imports Debian and Conda packages for bioinformatics, providing containers for each new package version and an already large set of containers (~585 debian, ~1400 bioconda).

Extra metadata (homepage, license, ...) are extracted from packages to provide them via the web interface (and indexed).



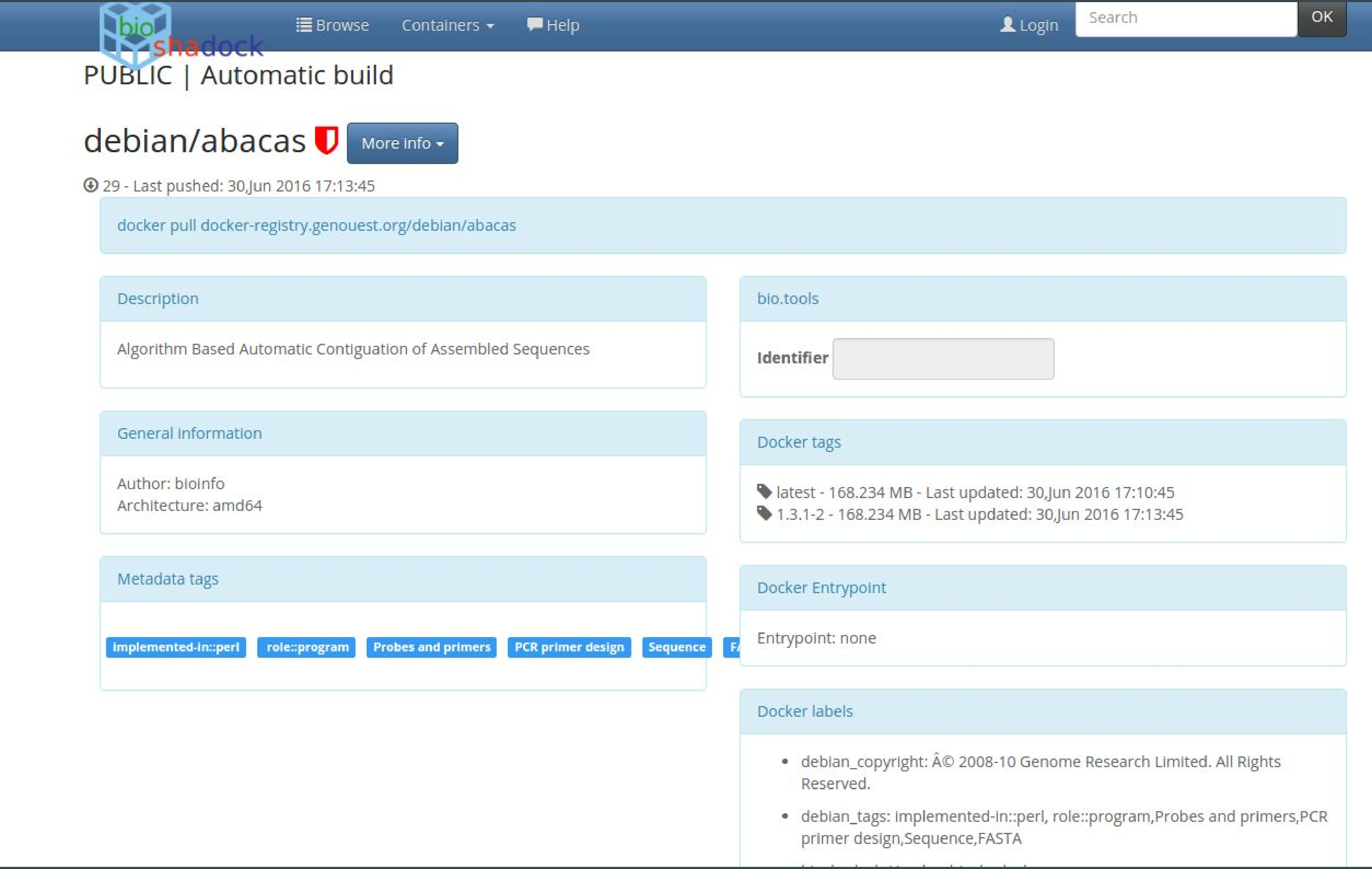
Docker Entrypoint

Entrypoint: ["/usr/bin/tini","--"]

Docker labels

- bioconda_package_Version: 3.1.0
- bioconda_about_Home: http://amos.sourceforge.net/wiki/index.php/AMOS
- bioshadock_Vendor: bioshadock
- bioconda_requirements_Run: ['mummer', 'perl-threaded', 'perl-statistics-descriptive', 'perl-xml-parser', 'perl-dbi', 'jellyfish', 'python']
- bioshadock_tests: WyJNaW5pbW8gLWggPiAvZGV2L251bGwiXQ==
- bioconda_source_Git_Rev: 9bd658
- bioconda_about_License: Artistic License
- bioconda_build_Number: 3
- bioconda_test_Commands: ['Minimo -h > /dev/null']
- bioconda_about_Summary: A Modular, Open-Source whole genome assembler
- bioconda_package_Name: amos
- bioconda_requirements_Build: ['gcc', 'zlib', 'boost', 'autoconf', 'blat', 'mummer', 'perl-threaded', 'perl-statistics-descriptive', 'perl-xml-parser', 'perl-dbi', 'jellyfish', 'python']
- bioconda_build_Skip: True
- bioconda_source_Git_Url: http://git.code.sf.net/p/amos/code

Overview



PUBLIC | Automatic build

debian/abacas

29 - Last pushed: 30,Jun 2016 17:13:45

docker pull docker-registry.genouest.org/debian/abacas

Description

Algorithm Based Automatic Contiguation of Assembled Sequences

General information

Author: bioinfo
Architecture: amd64

Metadata tags

implemented-in::perl | role::program | Probes and primers | PCR primer design | Sequence | F

bio.tools

Identifier

Docker tags

latest - 168.234 MB - Last updated: 30,Jun 2016 17:10:45
1.3.1-2 - 168.234 MB - Last updated: 30,Jun 2016 17:13:45

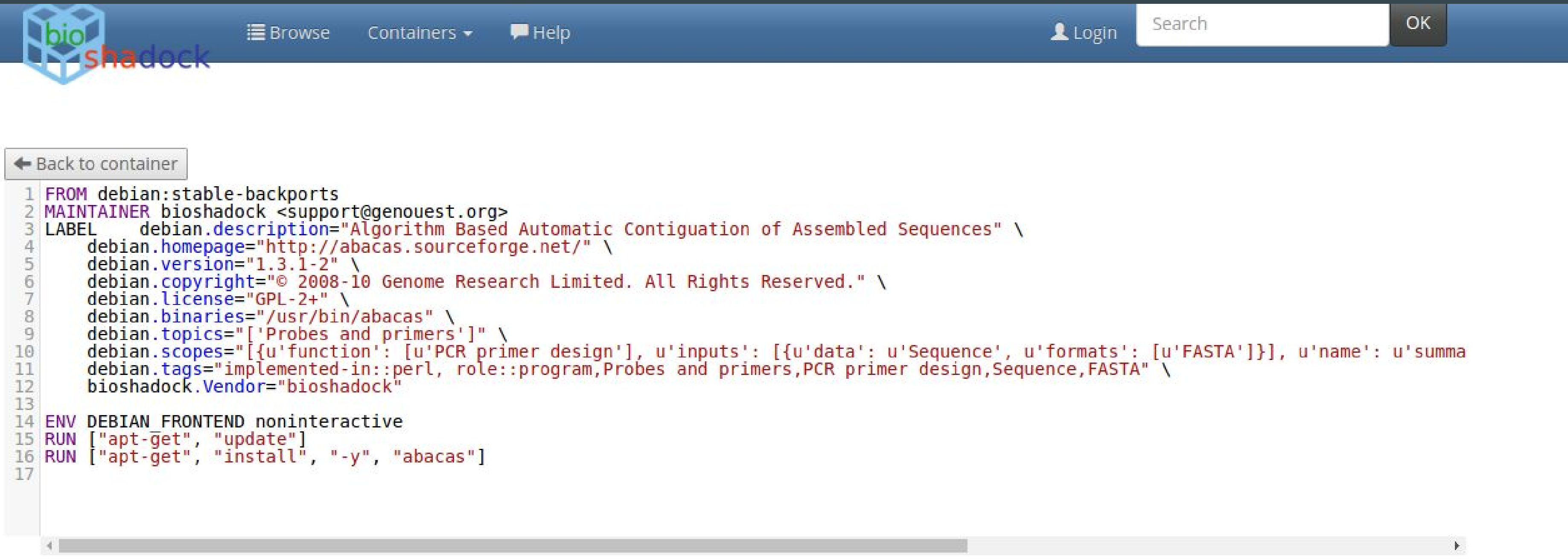
Docker Entrypoint

Entrypoint: none

Docker labels

- debian_copyright: Â© 2008-10 Genome Research Limited. All Rights Reserved.
- debian_tags: implemented-in::perl, role::program,Probes and primers,PCR primer design,Sequence,FASTA

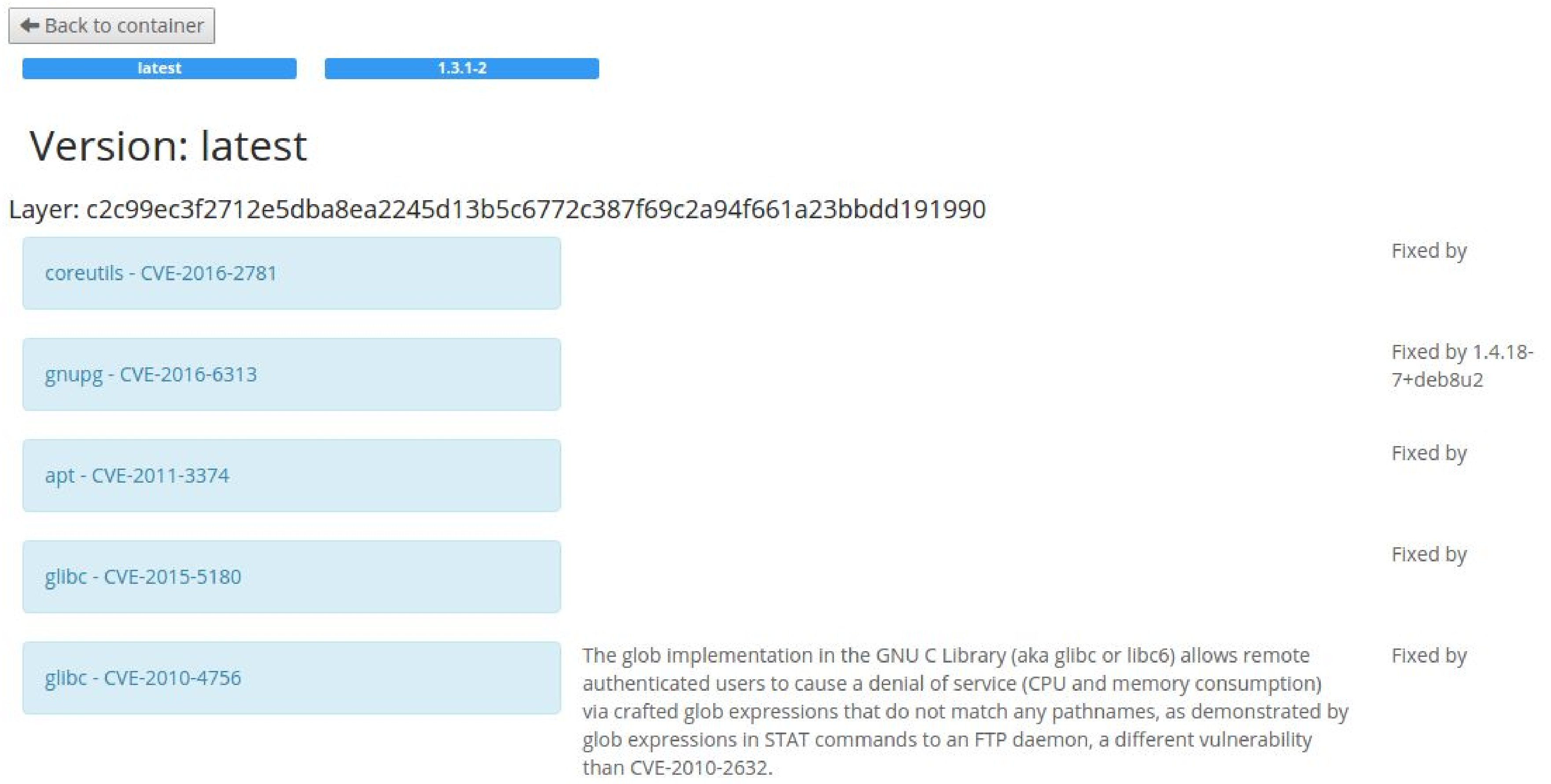
Dockerfile



```
FROM debian:stable-backports
MAINTAINER bioshadock <support@genouest.org>
LABEL    debian.description="Algorithm Based Automatic Contiguation of Assembled Sequences" \
    debian.homepage="http://abacas.sourceforge.net/" \
    debian.version="1.3.1-2" \
    debian.copyright="© 2008-10 Genome Research Limited. All Rights Reserved." \
    debian.license="GPL-2+" \
    debian.binaries="/usr/bin/abacas" \
    debian.topics="['Probes and primers']" \
    debian.scopes="[{u'function': [u'PCR primer design'], u'inputs': [{u'data': u'Sequence', u'formats': [u'FASTA']}], u'name': u'summa
    debian.tags="implemented-in::perl, role::program,Probes and primers,PCR primer design,Sequence,FASTA" \
    bioshadock.Vendor="bioshadock"

ENV DEBIAN_FRONTEND noninteractive
RUN ["apt-get", "update"]
RUN ["apt-get", "install", "-y", "abacas"]
```

Security report



Version: latest

Layer: c2c99ec3f2712e5dba8ea2245d13b5c6772c387f69c2a94f661a23bbdd191990

coreutils - CVE-2016-2781		Fixed by
gnupg - CVE-2016-6313		Fixed by 1.4.18-7+deb8u2
apt - CVE-2011-3374		Fixed by
glibc - CVE-2015-5180		Fixed by
glibc - CVE-2010-4756	The glob implementation in the GNU C Library (aka glibc or libc6) allows remote authenticated users to cause a denial of service (CPU and memory consumption) via crafted glob expressions that do not match any pathnames, as demonstrated by glob expressions in STAT commands to an FTP daemon, a different vulnerability than CVE-2010-2632.	Fixed by

Integration

Containers used by tools ecosystem

CWL workflow tools: possibility to execute a workflow using containers, Bioshadock can store tool CWL descriptor.

GoDocker: batch scheduling with Docker (SGE-like)

Galaxy

D4 workflow portal

GA4GH API to query Docker registries for bioinformatics: beta API integration available

Bio.tools: API integration in progress to link Bioshadock containers to bio.tools tool registry

Registry integration

Bioshadock provides a REST API to query/add container repositories.

The best solution?

NO, but...

Ideally, softwares have native packages for the different distributions and architectures, managing dependencies etc...

But upstream authors seldom do the job

Installing all required packages makes a growable list of software that remains on servers.

Container images can be frequently deleted when not used, and redownloaded on demand.

Thank you!

https://docker-ui.genouest.org

https://bitbucket.org/osallou/bioshadock

Authors: O. Sallou, F. Moreews