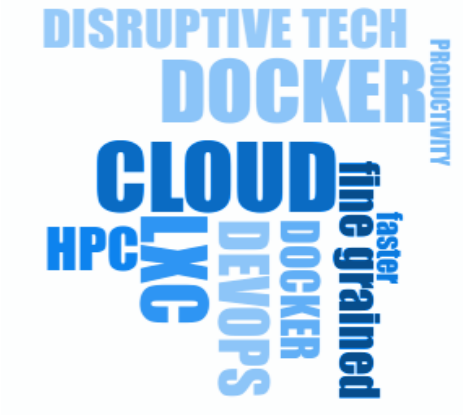
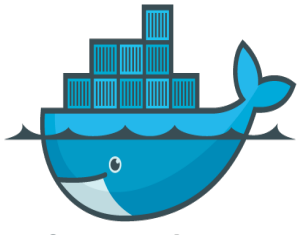
Docker : devops, shared registries, HPC and emerging use cases

François Moreews & Olivier Sallou

UMR IRISA



Presentation



“**Docker** is an open-source engine to easily create **lightweight, portable**, **self-sufficient containers** from any application. The same container that a developer **builds** and **test** on a laptop can run at **scale**, in **production**, on Vms,[...], public **clouds** and more.”

Presentation

chroot

- Each process/command on Linux has current working directory called root directory of a process/command. Chroot changes the root directory of a command, which ends up changing the root directory for both current running process and its children.

- A process/command that is run in such a modified environment **cannot access files outside the root directory**. This modified environment is known as "**jailed directory**" .

Presentation

What are Docker “containers” ?

Technically: ~chroot on steroids

- a container is a set of processes (running on top of common kernel)
- isolated* from the rest of the machine (cannot see/affect/harm host or other containers)
- using namespaces to have private view of the system (network interfaces, PID tree, mountpoints...)
- and cgroups to have metered/limited/reserved resources (to mitigate “bad neighbor” effect)

Presentation

What are Docker “containers” ?

From a distance: looks like a VM

- I can SSH into my container
- I can have root access in it
- I can install packages in it
- I have my own eth0 interface
- I can tweak routing table, iptables rules
- I can mount filesystems

...

Presentation

What are Docker “containers” ?

- boot in milliseconds
- just a few MB of intrinsic disk/memory usage
- bare metal performance is possible

Lightweight, fast, disposable...virtual environments :

An efficient new way to build, ship, deploy & run your apps !

Why it works–separation of concerns

- the **Developer** worries about what's "**inside**" the container
  - His code
  - His Libraries
  - His Package Manager
  - His Apps
  - His Data
- All Linux servers look the same

- the **Ops** (admin) worries about what's "**outside**" the container
  - Logging
  - Remote access
  - Monitoring
  - Network config
- All containers start, stop, copy, attach, migrate, etc. the same way

Presentation

- LinuX Containers (LXC)
- Control Groups & Namespaces
- AUFS
- Client – Server with an HTTP API

Build : Dependencies & Dockerfile

```
more api/Dockerfile


FROM giltarchitecture/
                        ubuntu-openjdk-7-jre-headless:12.0.4

ADD . /apidoc

ENTRYPOINT ["/apidoc/bin/apidoc-api"]
```

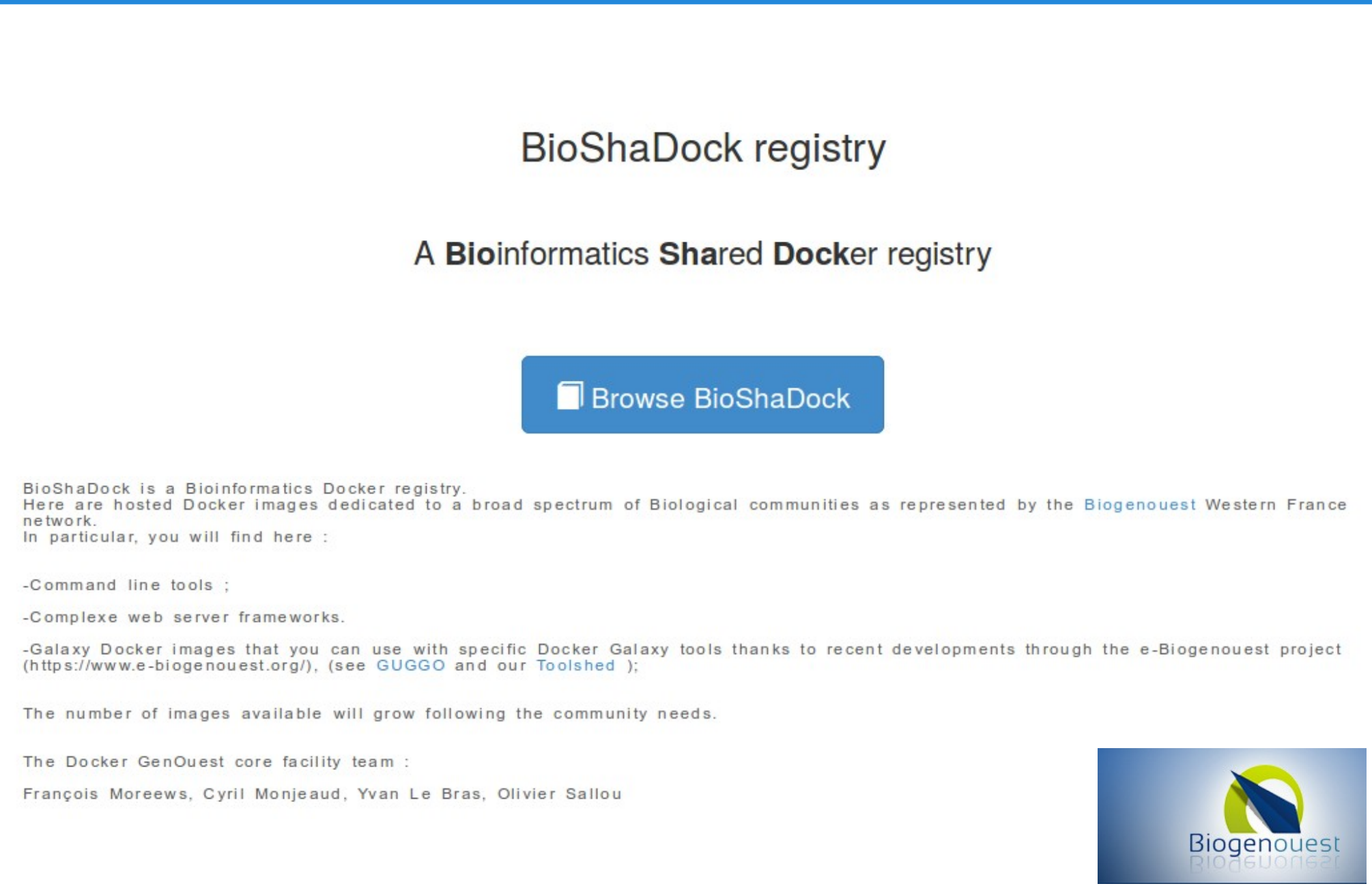
Run Docker

```
docker -run
   --expose 80
   -p 9000:80
   mydockercontainer-1-2-3
   -Denv.port=90
   -Denv.conffile=file.conf
```

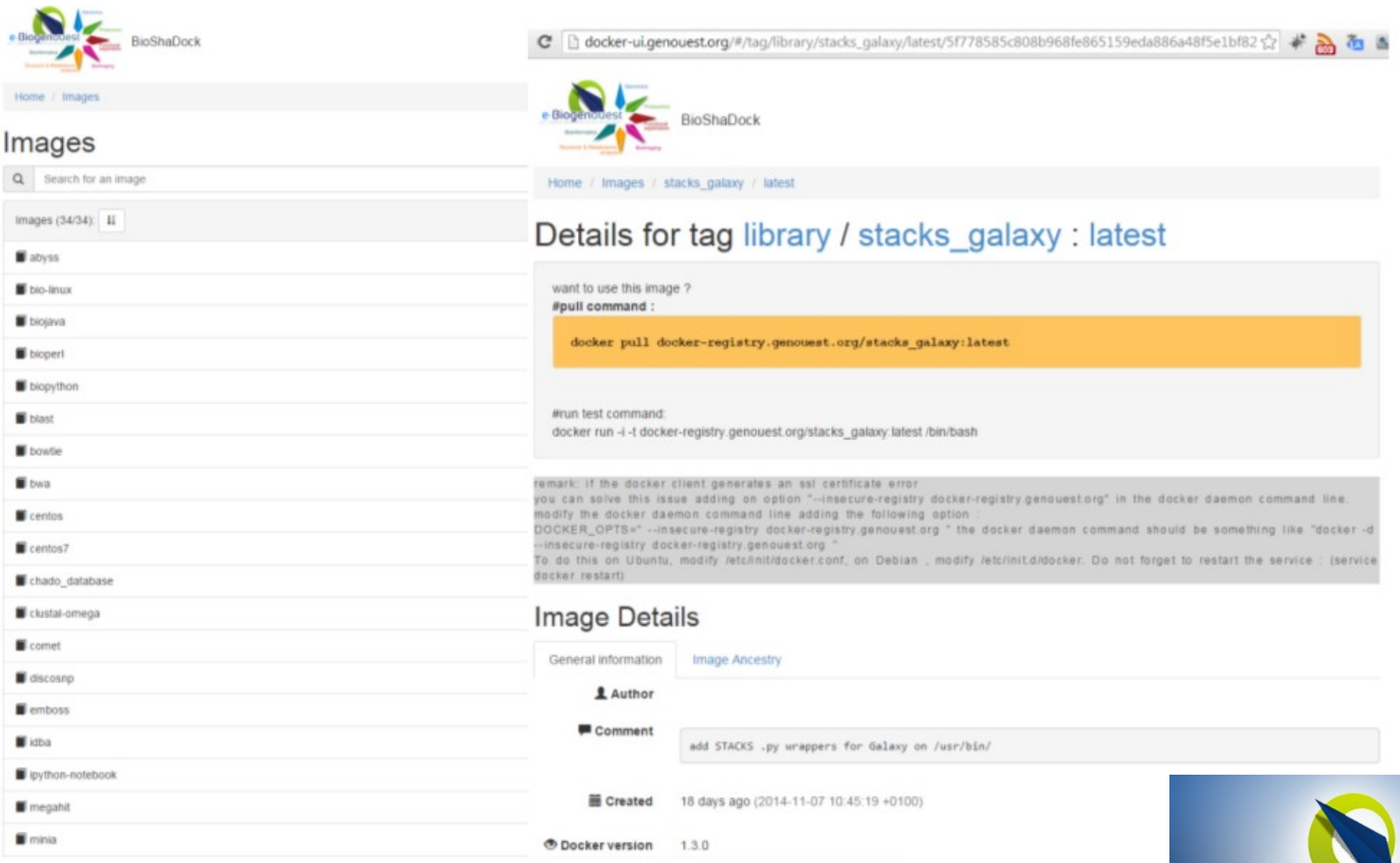
deamon mode (-d) or interactif mode (-i)

Docker Registries

Shared registries : yours



Shared registries : yours



Biogenouest

Google Container Engine

A Container based cloud architecture

Google Container Engine

- Alpha

- the Google Container Engine is inspired by Google's experience with building and running container-based distributed systems, Container Engine re-imagines some of Google's most powerful internal systems, so that you can develop and manage containers the way Google's engineers do.

- With container-based computing, application developers can focus on their application code, instead of on deployments and integration into hosting environments. At the same time, applications can be built with few constraints. Operations can provide a robust platform that quickly provisions compute resources and easily manages applications. The tools need to support the right controls for such application and resource management.

- The focus with Container Engine is on building these tools and controls for operations. At the same time, google want to allow for workload mobility, where containerized applications can run multi-cloud. They have, therefore, designed Container Engine to support Kubernetes, the open source technology, so that customers can run on multiple clouds.

Google Kubernetes

A Container based cloud architecture

Google Kubernetes

- Kubernetes is an open source container cluster manager. It schedules any number of container replicas across a group of node instances. A master instance exposes the Kubernetes API, through which tasks are defined. Kubernetes spawns containers on nodes to handle the defined tasks.

- The number and type of containers can be dynamically modified according to need. An agent (a kubelet) on each node instance monitors containers and restarts them if necessary.

- Kubernetes is optimized for Google Cloud Platform, but can run on any physical or virtual machine.

DOCKER : CLOUD & HPC

GO Docker

Batch Scheduler with Docker

Job/Interactive context

If user request root access:

- mount job directory in container
- mount /softs, /db as read-only
- execute command
- chown job directory with user id after command completion

If user does not request root access:

- mount ~user in container
- mount job directory in container
- mount /home/user
- mount additional user requested directories if user has read access (uid/gid)
- mount /softs, /db as read-only
- execute command as user (same uid/gid)