Automatic Dependency Management for Scientific Applications on Clusters

Ben Tovar*, Nicholas Hazekamp, Nathaniel Kremer–Herman, Douglas Thain
Scientist says: “This demo task runs on my laptop, but the real application has thousands of tasks. I wonder if it can be run on this cloud/grid/opportunistic resources I have heard so much about?”
Sure! What do you need?
- Linux.
- Which version?
  - ...
  - Let’s say RHEL7 for starters. I guess you need python?
  - Yes!
- Which version?
  - ...
  - Let’s say python 2.7. Anything else?
  - No. I don’t think so.
Hey, it doesn’t work. The error says that blastn is missing.

Could you install it in all the sites and machines I may end up using?
Hey, it doesn’t work. The error says that `blastn` is missing.

Could you install it in all the sites and machines I may end up using?
Three problems

1. Finding dependencies is an interactive discovery adventure for the user

We may point the user in some direction, but it is not possible to know all the dependencies for all the software a scientist/postdoc/grad-student may end up using.
Three problems

2. The sysadmins don’t have the time to install all of the software

Even if they had the time, they couldn’t anyway in opportunistic resources.

And even if they could, scientific software has many ad-hoc custom installations.

And even if it didn’t, it changes very rapidly.
Three problems

3. Not all dependencies are software packages.

Operating systems, filesystem mounts, network access, among others.
I know what you are thinking
I know what you are thinking

Let’s use containers!

(or vms if you don’t think yourself as a whippersnapper)
Containers start like this:

We need to fill the container by solving the aforementioned problems.

Containers can be used later to deliver the solutions we may find.
Our solution

The **vc3-builder**, a command-line tool for deploying software environments on clusters.

```
vc3-builder
   --require-os redhat:7
   --mount /scratch=/data
   --require python:2.7 -- myapp ...my args...
```
Use case

```
vC3-builder --require python:2.7.5 --require ncbi-blast myapp
```
Use case

vc3-builder --require python:2.7.5 --require ncbi-blast myapp

compute node

with:
  python 2.7.12

without:
  blast
vc3-builder --require python:2.7.5 --require ncbi-blast myapp

compute node

with:
  python 2.7.12
without:
  blast
Use case

vc3-builder --require python:2.7.5 --require ncbi-blast myapp

compute node

with:
  python 2.7.12
without:
  blast

import

task sandbox

install blast and its dependencies

vc3-builder
Help users with dependency discovery

1. As users locally discover dependencies using a clean sandbox, they write shell (sh) recipes in a JSON document.
Rely less on sysadmins for installation

2. All vc3-builder actions are done only with the privileges the user already has (e.g., no sudos).

- Everything is installed to a directory the user has write access (e.g., $HOME/myOwnDir)
- As much as possible, dependencies already available in the host system are not re-installed.
More than software dependencies

3. Allow for the dependencies on particular operating systems, filesystems, or mount points to be described.
Technology independence

```
./vc3-builder --interactive --require-os redhat:7

OS trying:    redhat:v77     os-native
OS fail prereq: redhat:v7    os-native
OS trying:    redhat:v7.4    singularity
OS fail prereq: redhat:v7.4  singularity
OS trying:    redhat:v7.4    docker
sh-4.2$ cat /etc/redhat-release
Red Hat Enterprise Linux Server release 7.4 (Maipo)
```

These were not popular a few years ago, and new technologies will be a few years hence. The vc3-builder does not tie down the user to a single technology.
An important consideration

**Lightweight bootstrap**

vc3-builder is a completely self-contained program, including recipes. Meant to be easy to deploy.

It can be compiled to a truly statically linked binary.

```bash
wget github.com/vc3-project/vc3-builder/releases/download/release/v0.1.0/vc3-builder
chmod 755 vc3builder
./vc3-builder --require ...
```
Other considerations

**Batch context**
Interaction with the user is not desirable.
Installation as a preliminary step to job execution.

**Dynamic composition**
Use what’s already there.
Each package to its own sandbox.
The builder can be used inside another builder instance.
### Differences with established players

<table>
<thead>
<tr>
<th></th>
<th>spack</th>
<th>*nix</th>
<th>vc3-builder</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lightweight bootstrap</strong></td>
<td>tarball + env script, 104 MB.</td>
<td>tarball + env script, 67 MB</td>
<td>single executable, 600 KB</td>
</tr>
<tr>
<td><strong>Batch context</strong></td>
<td>meant to run in headnode</td>
<td>meant to run in headnode</td>
<td>part of job execution</td>
</tr>
<tr>
<td><strong>OS and filesystem dependencies</strong></td>
<td>no</td>
<td>no</td>
<td>wrapping singularity, docker and the parrot virtual file system</td>
</tr>
<tr>
<td><strong>Focus</strong></td>
<td>High Performance Computing (HPC)</td>
<td>reproducibility</td>
<td>High Throughput Computing (HTC)</td>
</tr>
<tr>
<td><strong>Bit-to-bit guarantees</strong></td>
<td>system packages discouraged</td>
<td>yes, by compiling everything from source (even the compiler), binaries if write access to /</td>
<td>system packages encouraged</td>
</tr>
<tr>
<td><strong>Recipes</strong></td>
<td>python abstractions</td>
<td>nix expression language</td>
<td>sh commands as JSON strings</td>
</tr>
<tr>
<td><strong>Number of recipes</strong></td>
<td>thousands</td>
<td>thousands</td>
<td>dozens</td>
</tr>
<tr>
<td><strong>Dependency resolution</strong></td>
<td>powerful</td>
<td>powerful</td>
<td>primitive</td>
</tr>
</tbody>
</table>
What do we gain?

- lightweight bootstrap
- batch context
- harness batch/cloud resources
What do we gain?

The vc3-builder was written in the context of the Virtual Clusters for Community Computation (VC3) project.

Users go to a website and create short lived clusters across heterogeneous resources.
What do we gain?

The vc3-builder was written in the context of the Virtual Clusters for Community Computation (VC3) project.

Users go to a website and create short lived clusters across heterogeneous resources.
What do we gain?

The vc3-builder is deployed as part of job submission to provide consistent software environments.
What do we gain?

The vc3-builder is deployed as part of job submission to provide consistent software environments.

The software environment may be constructed using other tools (e.g. spack)
Another gain: batch system mode

```
./vc3-builder --parallel-mode slurm ...
```

Several vc3-builders running in parallel for the same installation, as dependencies allow.

vc3-builders are submitted as regular batch jobs.

SLURM, HTCondor, WorkQueue, SGE, PBS, and torque are supported.

Only if a shared filesystem is available.
Case study: MAKER

MAKER is a genome annotation pipeline used in bioinformatics.

In total, an installation of MAKER consists of 39 dependencies.

The size of the sources is 785M.
The installation size is 4.2G.

(see paper for Octave and cvmfs case studies)
MAKER without the vc3-builder

Someone sends you an email with a link with instructions how to install MAKER.

Dependencies of MAKER (themselves have other dependencies)

MAKER proper is over here...
The table below shows the MAKER timings at different sites and for different modes:

<table>
<thead>
<tr>
<th>site</th>
<th>Notre Dame</th>
<th>Comet</th>
<th>Bridges</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>sequential</td>
<td>parallel</td>
<td>distributed</td>
</tr>
<tr>
<td></td>
<td>distributed</td>
<td>parallel</td>
<td>distributed</td>
</tr>
<tr>
<td>time</td>
<td>00h56m</td>
<td>00h29m</td>
<td>00h23m</td>
</tr>
<tr>
<td>concurrency</td>
<td>1</td>
<td>15</td>
<td>17</td>
</tr>
</tbody>
</table>

- At headnode, one package at a time.
- At headnode, several packages at a time.
- Using the batch system, several packages at a time.

Mostly waiting in HTCondor queue at Comet and mostly waiting in SLURM queue at Bridges.
Conclusions

In a laptop users can easily install software dependencies (e.g. rpms).

Making those dependencies explicit is hard, but needed in batch/cloud contexts.

Containers by themselves don’t help us in making dependencies explicit.

Configure in a laptop, run anywhere in a batch/cloud without sudo, as part of job execution.
Thanks!

https://github.com/vc3-project/vc3-builder
https://www.virtualclusters.org

funding: Department of Energy NGNS program via grant DE-SC0015711

thanks to the VC3 team, specially to Kenyi Hurtado for testing the vc3-builder