



# CI Made (Sort Of) Easy

How to build your projects without going insane

Ondřej Surý • [ondrej.sury@nic.cz](mailto:ondrej.sury@nic.cz) • 27. 10. 2016

# The premises

- Own your data
- Run your own infrastructure

# Contents

- Jenkins and Jenkins Job Builder
- Buildbot
- Gitlab CI

# Basic CI concepts

- Version Control Systems (Git ... and others)
  - Commits
  - Branches
  - Authors/Users
- Workers/Runners/Slaves
  - The machines
- Projects/Jobs/Builders
  - The job definition
- Schedulers/Triggers
  - When to run
- Publishers/Artifacts
  - The result of the build

# Basic CI Workflow

- Receive trigger (Commit, Time event)
- On each build node:
  - Checkout source code
  - Prepare the environment
  - Run the build script
  - Archive the artifacts
  - Cleanup the environment
- Trigger the next job

# Jenkins Job Builder

# Jenkins

- Complex system written in Java
- Can be configured via
  - web interface
  - XML, JSON, Python API
- Okay(ish) for small number of different jobs

# Jenkins Job Builder

- Build Jenkins jobs from Yaml/JSON description
- Adds job-templates, macros and projects
- Written in python (for OpenStack)



# Jenkins Job Builder Installation

```
$ virtualenv --no-site-packages jjb  
$ cd jjb/  
$ ./bin/pip install jenkins-job-builder  
$ ./bin/jenkins_jobs {update,test,delete,delete-all}
```

# Jenkins Job Builder Configuration

```
[jenkins]
```

```
user=<jenkins_user>
```

```
password=<jenkins_password>
```

```
url=https://howl.labs.nic.cz/jenkins/
```


# Job Configuration (Debian packaging)

```
- project: knot-resolver
  name: knot-resolver
  repo: knot-resolver
  distributions: !!python/tuple [jessie, stretch]
  architectures: !!python/tuple [amd64, i386, arm64, armhf]
  branches: !!python/tuple [master]
  sequential: false
  package:
    - knot-resolver:
      git-url: 'git://anonscm.debian.org/pkg-dns/knot-resolver.git'
  jobs:
    - '{repo}_{package}-source'
    - '{repo}_{package}-binaries'
    - '{repo}_{package}-piuparts'
```

# Job Configuration

```
- job-template:  
  name: '{repo}_{package}-source'  
  scm:  
    - git:  
      url: '{git-url}'  
      branches: '{obj:branches}'  
  builders:  
    - shell: |  
      distribution=${{distribution}} /usr/bin/generate-git-snapshot  
  Publishers:  
    - archive:  
      artifacts: '*.gz,*.xz,*.deb,*.dsc,*.git,*.changes'  
    - trigger:  
      project: '{repo}_{package}-binaries'
```

# Jenkins Interface (Example)

search ? Ondřej Surý | log out

Jenkins > Knot Resolver > ENABLE AUTO REFRESH


- New Item
- People
- Build History
- Edit View
- Delete View
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Credentials





































**Build Queue** -

No builds in the queue.

**Build Executor Status** -

- 1 Idle
- 2 Idle
- 3 Idle
- 4 Idle

 [add description](#)

All	Knot	<b>Knot Resolver</b>	PHP-binaries	PHP-piuparts	PHP-source	Sources	mesa	nginx	qbittorrent	+
S	W	Name ↓								
		<a href="#">knot-resolver_cmocka-binaries</a>								
		<a href="#">knot-resolver_cmocka-piuparts</a>								
		<a href="#">knot-resolver_cmocka-source</a>								
		<a href="#">knot-resolver_knot-binaries</a>								
		<a href="#">knot-resolver_knot-piuparts</a>								
		<a href="#">knot-resolver_knot-resolver-binaries</a>								
		<a href="#">knot-resolver_knot-resolver-piuparts</a>								
		<a href="#">knot-resolver_knot-resolver-source</a>								
		<a href="#">knot-resolver_knot-source</a>								
		<a href="#">knot-resolver_libuv1-binaries</a>								
		<a href="#">knot-resolver_libuv1-piuparts</a>								
		<a href="#">knot-resolver_libuv1-source</a>								

# Jenkins Job (Example)

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Multi-configuration project](#)

[Configure](#)

[Move](#)

## Project knot-resolver\_knot-resolver-binaries

Build Debian binary packages of knot-resolver.

Do not edit this job through the web, it is generated via jenkins-job-builder!

### Usage instructions how to remotely access and use the repository (as root user):

```
apt-get install lsb-release apt-transport-https
wget -O /etc/apt/trusted.gpg.d/knot-resolver.gpg https://packages.sury.org/knot-resolver_apt.gpg
cat << EOF > /etc/apt/sources.list.d/knot-resolver.list
deb https://packages.sury.org/knot-resolver $(lsb_release -cs) main
#deb-src https://packages.sury.org/knot-resolver $(lsb_release -cs) main
EOF
apt-get update
```

**Build History** [trend](#)

- #33** Oct 26, 2016 2:24 AM
- #32** Aug 24, 2016 2:27 PM
- #31** Aug 24, 2016 8:48 AM

[RSS for all](#) [RSS for failures](#)

Configuration Matrix	jessie	stretch
amd64		
i386		
arm64		
armhf		

[edit description](#)

[Disable Project](#)



[Latest Test Result](#) (no failures)

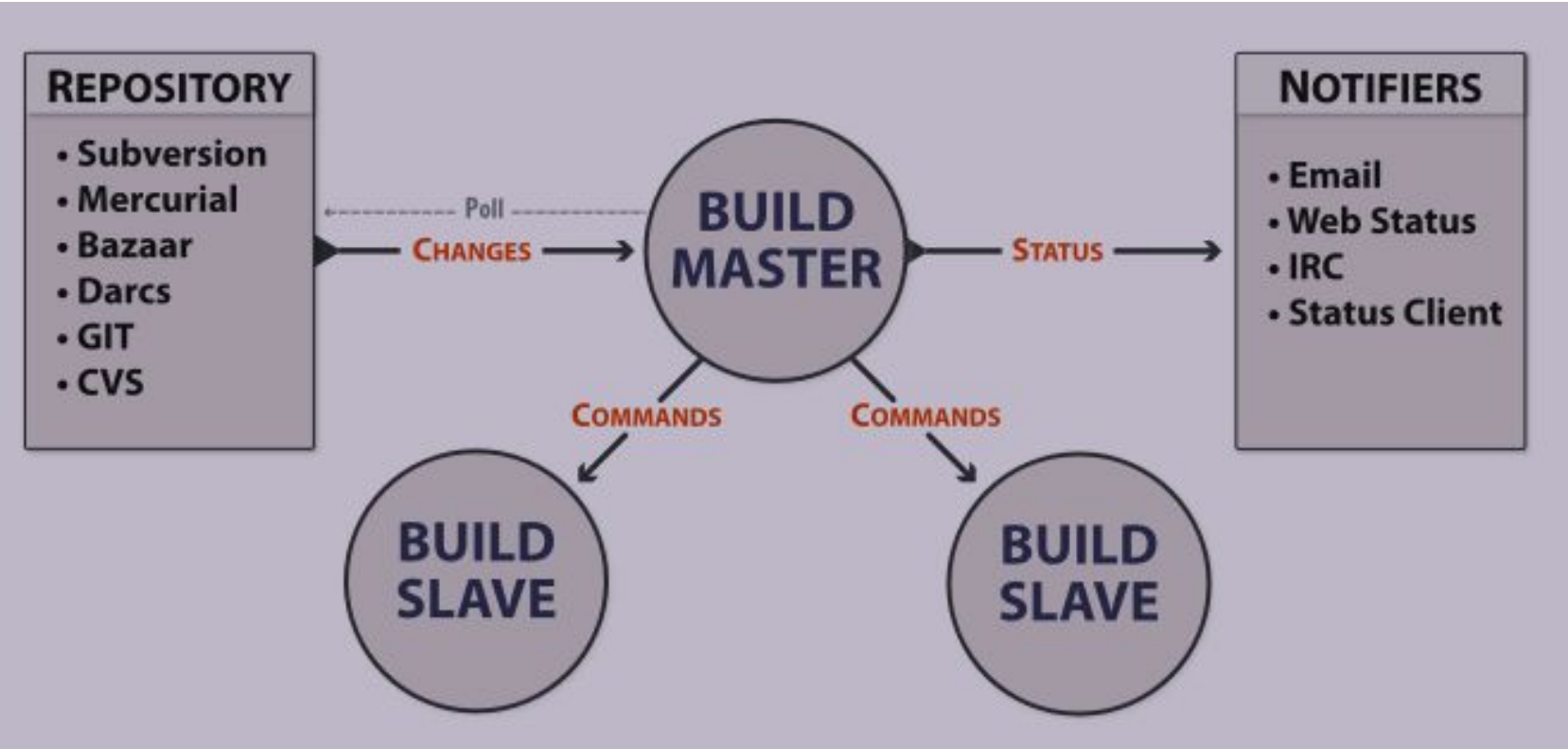
Buildbot

# Buildbot

- Job Scheduling System
- Written in Python
- The configuration is a Python Script



# Buildbot



# Buildbot Master Installation

```
$ virtualenv --no-site-packages bb-master
$ cd bb-master
$ ./bin/pip install buildbot[bundle]
$ ./bin/buildbot create-master master
$ mv master/master.cfg.sample master/master.cfg
$ $EDITOR master/master.cfg
$ ./bin/buildbot start master
Following twisted.log until startup finished..
The buildmaster appears to have (re)started correctly.
```

# Buildbot Worker Installation

```
$ virtualenv --no-site-packages bb-worker
$ cd bb-worker
$ ./bin/pip install buildbot-worker
$ ./bin/buildbot-worker create-worker worker <hostname> <user> <pass>
$ ./bin/buildbot-worker start worker # ← just a directory
Following twisted.log until startup finished..
The buildbot-worker appears to have (re)started correctly.
```

# Buildbot Configuration

```
from buildbot.plugins import *
c = BuildmasterConfig = {}
knot_git = 'https://gitlab.labs.nic.cz/labs/knot.git'
c['workers'] = [worker.Worker("<user>", "<pass>")]
c['change_source'].append(changes.GitPoller(knot_git,
      workdir='knot-wd', branch='master', project='knot', pollinterval=300))
f_knot = util.BuildFactory()
f_knot.addStep(steps.Git(repourl=knot_git,mode='incremental'))
f_knot.addStep(steps.ShellCommand(name="configure", command=["./configure"]))
f_knot.addStep(steps.ShellCommand(name="make all", command=["make", "all"]))
f_knot.addStep(steps.ShellCommand(name="make check", command=["make", "check"]))
c['builders'] = [util.BuilderConfig(name="knot-build", workernames=["komorebi"],
factory=f_knot)]
[...]
```

# Buildbot Master Interface

Knot DNS Family



Knot DNS Family

Waterfall View

NAVIGATION

Home



Waterfall View



Console View



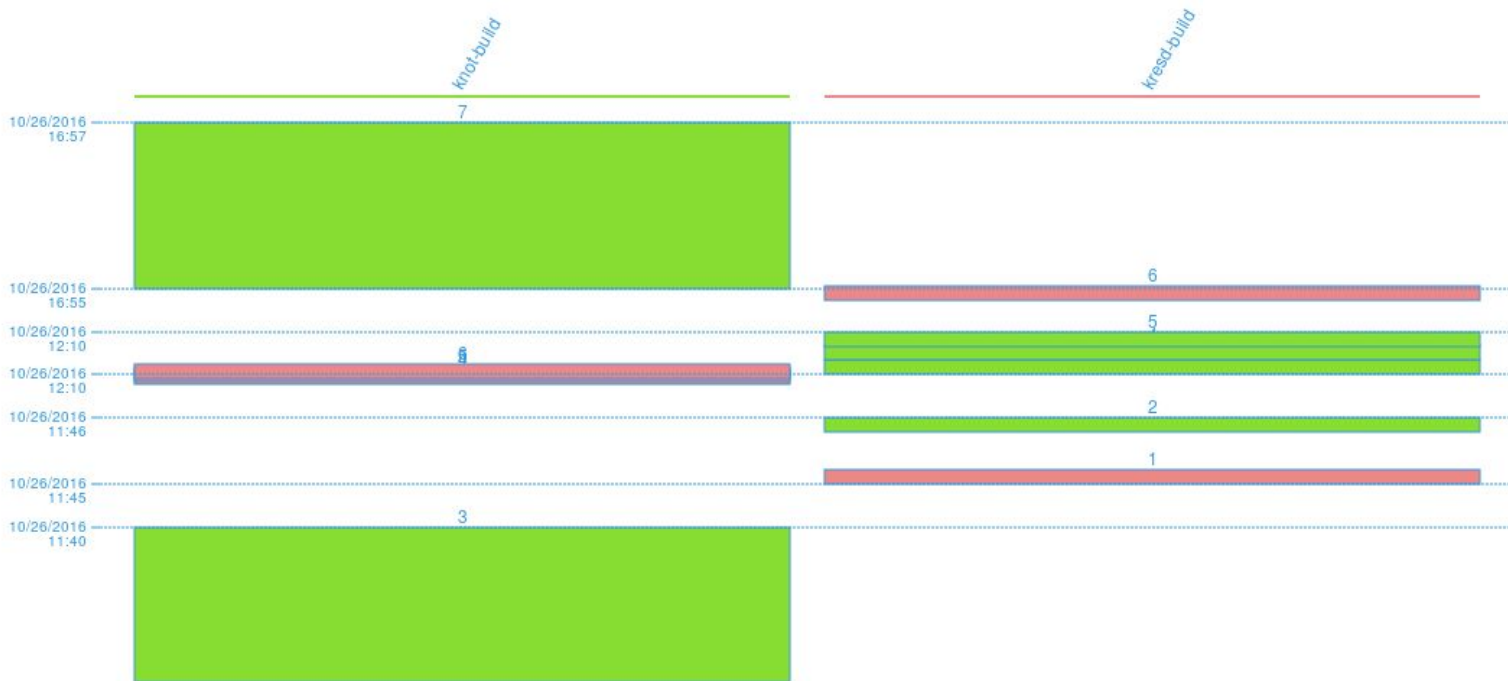
> Builds



About



Settings



# Buildbot Build (Example)

Build steps

Build Properties

Worker: komorebi

Responsible Users

Changes

Debug

☑ All knot-build/7

1:55 finished **SUCCESS**

- 0 ☑ make clean 1 s 'make clean'
  - 0 [stdio](#) (316 lines)
- 1 ☑ git 1 s update
  - 0 [stdio](#) (439 lines)
- 2 ☑ autoreconf -fi 6 s 'autoreconf -fi'
  - 0 [stdio](#) (83 lines)
- 3 ☑ configure 5 s './configure'
  - 0 [stdio](#) (278 lines)
- 4 ☑ make all 1:09 'make'
  - 0 [stdio](#) (401 lines)
- 5 ☑ make check 32 s 'make check'
  - 0 [stdio](#) (450 lines)

Gitlab CI

# Gitlab CI

- Part of Gitlab
- Written in Ruby (on Rails)
  - With GitLab Runner written in Go (apparently RoR is not cool enough now :))
- Uses `.gitlab-ci.yml`



# Gitlab CI Runner Installation

- Run couple of random `curl https:// | sudo bash` scripts from all over the web
  - Now we are COOL!
  - Because adding one apt gpg key and extra repository is too hard
- Or carefully analyze those scripts and add the repository by hand
  - And be the old, grumpy, bearded guy...

```
# curl https://packages.gitlab.com/gpg.key | gpg --import
# gpg --export E15E78F4 > /etc/apt/trusted.gpg.d/gitlab-ci.gpg
# curl https://packages.gitlab.com/.. > /etc/apt/sources.list.d/gitlab-ci.list
# apt-get install gitlab-ci-multi-runner package
# gitlab-ci-multi-runner register
    ## Enter your gitlab-uri, registration token, tags and run method (shell,
ssh, docker, ...)
# gitlab-ci-multi-runner start
```

# Gitlab CI Configuration (.gitlab-ci.yml)

before\_script:

- apt-get update -qq && apt-get build-dep -y knot-resolver

job\_build:

script:

- make -j2 all V=1

job\_check:

script:

- make check V=1



Questions?  
Comments?  
Tomato Throwing?

Ondřej Surý • [ondrej.sury@nic.cz](mailto:ondrej.sury@nic.cz) • 27. 10. 2016