Packaging and Configuration

Chapel Team, Cray Inc.

Chapel version 1.14

October 6, 2016

Safe Harbor Statement

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

Outline

- Chapel Configuration File
- Anonymous ‘printchplenv’
- Chapel Docker Image
- Chapel Debian Package

Chapel Configuration File

ChplConfig: Background

- **CHPL variable priority in 1.13:**
  - Explicit compiler flags: `chpl --env=value`
  - Environment variables: `CHPL_ENV=value`
  - Inferred environment variables: `printchplenv`
- **Setting new default values requires:**
  - Modifying environment through a shell script or module
  - Modifying the printchplenv / chplenv source code
- **Problematic for system-wide installs and packaging**

ChplConfig: This Effort

- **CHPL variable priority in 1.14:**
  - Explicit compiler flags: `chpl --env=value`
  - Environment variables: `CHPL_ENV=value`
  - **Chapel configuration file: `~/.chplconfig`**
  - Inferred environment variables: `printchplenv`
- **Example Chapel Configuration File:**

```
# Default to multi-locale
CHPL_COMM=gasnet

CHPL_TASKS=qthreads # Use Qthreads

# System GMP is available on these machines
CHPL_GMP=system
```

ChplConfig: This Effort

- Filename priority
  - `chplconfig`
  - `.chplconfig`
- File search path priority
  - `$CHPL_CONFIG`
  - `$HOME`
  - `$CHPL_HOME`
- Write current overrides to a Chapel configuration file
  - `printchplenv --overrides > ~/.chplconfig`
- Write all settings to a Chapel configuration file
  - `printchplenv --simple > ~/.chplconfig`

ChplConfig: Status and Next Steps

Status:

- Users can now override default CHPL vars with a file
- Documented in online docs

Next Steps:

- Utilize the configuration file for reproducibility
  - Generate a config file for each configuration built under lib/
  - Use config files in testing infrastructure for different test environments

printchplenv --anonymize

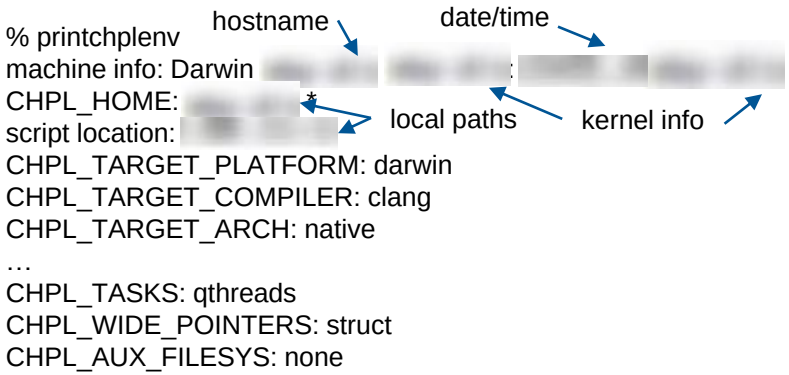
printchplenv --anonymize

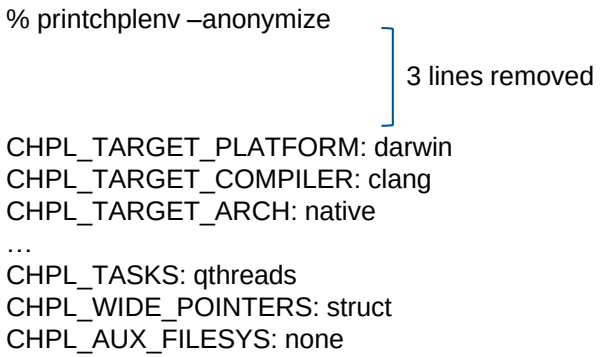
Background: We ask for printchplenv output in bug reports

- prints information about the machine to help reproduce the bug
  - lots of useful information
  - but may include details people don't wish to share (paths, hostname, etc.)

This Effort: Add an --anonymize flag to printchplenv

- Strips potentially sensitive details from the output

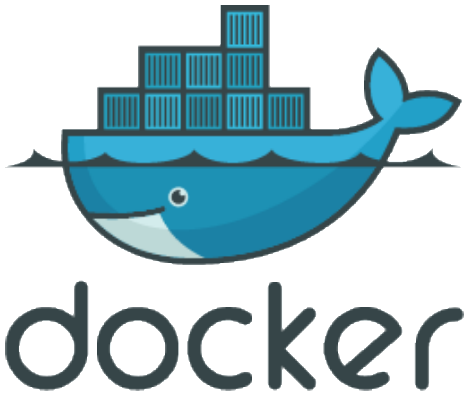




Impact: Bug reporters no longer need to edit this by hand

Chapel Docker Image

Chapel Docker Image



Background:

- Docker improves ease of installation and use
  - Improves accessibility in cloud services

This Effort:

- Official Docker image hosted on:
  - https://hub.docker.com/r/chapel/chapel/
- Easily hosted as a Docker container on Linux distros and OS X
- Contributed by Nick Park

Status:

- Docker images for 1.13.1 and 1.14.0 available on DockerHub

Next Steps:

- Get Chapel's docker image into https://github.com/docker-library
- Offer broader Chapel configurations as docker containers
- Integrate docker image testing into nightly build/test infrastructure

Chapel Debian Package

Chapel Debian Package: Background

- **Created initial Debian package for 1.13.0**
  - Submitted *request for package* (RFP) and *intent to package* (ITP)

- **Packaging Chapel has faced many challenges**
  - Debian has a strict packaging policy
    - Packages expected to build and install in a specific way
    - Third-party dependencies must be package-dependencies
  - Chapel has a non-traditional build system
    - Requires a *$CHPL_HOME*
    - Third-party dependencies are bundled into repository
    - No *configure* or *make install* build steps

Chapel Debian Package: This Effort & Impact

This Effort: Simplify our package

- Reduced Debian package to *chapel-minimal*
  - Stripped away all third-party packages (except utf-8 decoder)
  - Stripped away docs, tests, and test infrastructure
  - Similar approach done with other packages, e.g. *python-minimal*
- Received several iterations of feedback online and on ITP
- Chapel configuration file, *chplconfig*, takes place of patching overrides

Impact: Progress made towards acceptance

- Nearly all feedback has been addressed
- The simplified package has eliminated a lot of the initial challenges

Chapel Debian Package: Next Steps

- **Get *chapel-minimal 1.14.0-1* accepted into Debian**
  - Requires packaging UTF-8 decoder
  - Requires following filesystem hierarchy standard
- **Pursue full-featured Chapel package**
  - Including third-parties
- **Expedite propagation of package elsewhere**
  - *unstable* → *stable*
  - downstream distributions, such as Ubuntu
- **Extend to other linux distributions**
  - Fedora, SUSE, Arch, …
- **Encourage community members to build packages**
  - Packaging relies more on knowledge of the distribution than Chapel

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

CRAY

THE SUPERCOMPUTER COMPANY