Guix – Write a Bourne-shell compiler front-end for Guile

James Wood, lamudri@gmail.com

March 24, 2016

Summary

Guix relies on a set of bootstrap binaries (GCC, Binutils, libc, Bash, Coreutils, etc.) from which everything is built. These binaries must be trusted, so we'd rather have as few bootstrap binaries as possible.

The goal of this project is to build on Bournish, a simplistic Bourne-shell-like front-end to Guile's compiler, and extend it to support all the features needed to run a *configure* script (including pipes, functions, and globbing), including features normally provided by sed, grep, find, and Coreutils. When that is done, a whole bunch of binaries can be removed from the bootstrap binaries.

Benefits

The main benefit to come from this project is the removal of trusted binary files from Guix's base. This would allow users to have greater trust in the system.

Deliverables

Throughout the project, work will be adding to the existing Bournish tool. This will produce a program capable of interpreting some subset of Bash language scripts. By the end of the project, bootstrapping with the new Bournish and reduced list of binaries should be demonstrated.

Plan

Work will start in mid-June. I will likely require a week to set up a development environment and become familiar with some of the idioms of Guile and Guix. The next two weeks should be focused on creating a parser, probably making use of libraries such as SILex, and then making a dummy interpreter for the produced AST. The final five weeks will then be spent implementing the required Bash machinery and commands. By the midpoint of the project, the front-end should be able to run some simple Bash script, showing the completion of the parser and some successful runtime support. Hopefully, I will be able to run the interpreter on the bootstrapping script, and so will use this as a test case towards the end of the project. If I don't finish the project by the end of the eight weeks, I am happy to continue working on it right up until the end of September, if necessary.

Communication

I expect to work on a branch of the Guix git repository, and will commit and push frequently. As for other types of communication, I am quite comfortable with IRC, and will probably use the #guix Freenode channel extensively when I need advice (nick: mudri). I check emails frequently, but don't currently have any preference in real-time private messaging.

Qualification

I am already somewhat involved in the NixOS project, particularly valuing the isolation between packages provided by Nix and Guix. I see Guix's benefit in allowing configuration in a general-purpose language, allowing use of libraries and tools made for that language. However, I have stayed away from Guix due to perceiving lack of maturity and support. I would love to help make this a non-issue, and switch to using GuixSD.