GOOGLE SUMMER OF CODE 2016

GENTOO FOUNDATION

CONTINUOUS STABILIZATION OF GENTOO PACKAGES

Motivation for Project / Goal

Gentoo is an operating system with extreme focus on configurability and performance. To provide fully customizable experience, without interfering with the stability of the system, Gentoo has a concept of masked packages. These masked packages (or versions) are either untested, or are known to be unstable and are installed only if the user explicitly unmasks them. While this concept is a boon to the stability of the operating system, the current implementation requires the packages to be tested manually by a team of developers. This significantly increases the time in which new packages are made safely available to the users. The goal of this project is to provide a mechanism to test and stabilize the packages automatically with little or no human intervention.

Expected Results:

Deliverables for mid term evaluation:

- Capability to run stabilization job on pre existing packages using a minimal set of packages required for a stable Gentoo installation.
- Ability to monitor newly added packages for stabilization, as well as bugzilla to find packages that haven't had a reported bug for a sufficiently long period of time.
- Docker images ready that would be used as the Gentoo container for stabilization scripts for both CI and BOINC clients.
- At least some work done on the post install test framework.
- Documentation for the work done till this point.

Deliverables for final evaluation:

- BOINC server set up and hosted on cloud, responsible for distributing stabilization jobs to volunteers' computers.
- BOINC client side scripts ready to receive and execute jobs from the server.
- Docker based package caching system to prevent multiple downloads of same packages by the BOINC client user.
- Post install test framework to allow package maintainers to include custom scripts for CI testing after the package has been built and installed.

After the project is completed, given the docker images and stabilization scripts, it should be possible to keep up with all newly added packages and updates in near real time and without human intervention. The goal of this project is to reach a point where MOST of the packages have their latest versions stabilized and unmasked.

Implementation Details:

1. **Container:**
   There are a couple of common problems in Arch testing. Firstly, the environment in which we build affects how the package will be built a lot. Because of this, we require the environment to be as clean as possible. Secondly, since the jobs may need to be run on multiple locations - CI, Gentoo Servers, Volunteer computers - we need to have identical build conditions for all of them. For this, Docker seems to be an ideal candidate, since it runs everywhere (including Github's Travis CI), is open source, and provides the identical environment to all builds irrespective of where they are being run.

2. **USE Flags:**
   If a package has 'n' number of USE flags, to build the package with all possible combinations of USE flags would require 2^n number of builds. For the packages with a lot of USE flags, this results in a VERY unrealistic number of total builds. So for a more realistic number of builds, we build the packages for a certain few combinations of USE flags.

a. Without any USE flag turned on
b. With all USE flags turned on
c. Few random combinations based on default flags, or inverse of default flags or those generated by tatt.

3. **Post Install Test Framework:**
   Currently, all testing after building and installing a package is done manually. There isn't a way for the maintainer to provide scripts for testing of a package post installation, without human intervention. This test framework would be a wrapper that would provide a convenient way to write test suites. In this framework, maintainers would be able to declare dependencies and commands required to run the test suite as a part of the ebuild, as a separate function like post_install_test() that would be run only if the package is being installed as a part of a stabilization job. By allowing the maintainers to come up with suitable test suites, it should be possible to deliver a completely self stabilizing Gentoo system.

4. **Dependencies:**
   For a package to be stabilized, it can run against the current stable versions of its dependencies, or trigger the stabilization of the dependencies first, and resume its own job once all the dependencies have been stabilized.

5. **Triggering of stabilization jobs:**
   Triggering of stabilization jobs can occur whenever a pull request is made that changes the status of KEYWORDS in any package. However, apart from that the Gentoo Bugzilla can also be monitored for changes and a record of the last reported bug for each package (+version) can be kept. The stabilization job can also be run when no bug is reported for that package in sufficiently long period of time (either set explicitly by the maintainer or default to a value like 30 days).

6. **Infrastructure and Computing Power:**
   The main point in using docker for the builds is that the build process can take place anywhere. The three places where this would be done is:
   a. Travis CI (Github): This would be triggered by keyword changes. The script run for CI will determine how many packages (including dependencies) need to be stabilized and how many of those can be stabilized concurrently. If the number is small, it will try to perform the stabilization and report to the main server. If the stabilization times out or the number is large, the task will be transferred to the server for distribution via BOINC(volunteer computing).
   b. Volunteers' Computers: Using BOINC, a volunteer's computer can be used for running the jobs, which is ideal if there are a large number of jobs that can be stabilized concurrently.
   c. Gentoo Infrastructure (BOINC and main server): This would manage the distribution of jobs to clients and be responsible for updating the repository for the stabilized packages.

Timeline:

April 22 - May 22 (Community Bonding Period)	Get familiar with the intricacies in Architecture testing. Communicate with current Arch testers and get their opinions on the plan of action. Familiarise with the accepted conventions in the organisation as well as the technologies to be used in the project. (Coding conventions, documentation, communication etc)
May 23 - May 30 (Week 1)	Create a script for the evaluation of all dependencies of gentoo packages. The scripts would also need to resolve the order in which stabilization would require the minimum number of jobs.

May 30 - June 5 (Week 2)	Prepare Docker images for minimal Gentoo builds that would be used for stabilization using both Travis CI and volunteer computing.
June 5 - June 12 (Week 3)	Setup the server side scripts to receive data from CI (as mentioned above in 6a) and update the now stabilized packages. Also, if the CI was unsuccessful, either due to timeout or due to build failures, report to the person concerned.
June 12 - June 19 (Week 4)	Start work on adding support for custom post-install test frameworks. Create wrappers for various test frameworks such that the code for multiple frameworks may be written in a similar syntax. This would be converted to respective framework's code during the CI run.
June 19 - June 26 (Week 5)	(Buffer Period) : Catch up with any work left from previous weeks. **--------Mid Term Evaluation------**
June 26 - July 3 (Week 6)	Set up a BOINC server on a cloud hosted Virtual Machine. Start writing server side scripts for management of job distribution for the package builds. *Please note that this will require access to either Gentoo Infrastructure or an Amazon AWS account.*
July 3 - July 10 (Week 7)	Continue and complete server side scripts for the BOINC server. Continue with adding support for test frameworks.
July 10 - July 17 (Week 8)	Complete work on support for test framework. This would include adding of a new methods to ebuilds for CI testing as well as macros for the common procedures of running tests on common test automation frameworks. Start implementing a docker based method

	to cache downloaded packages without affecting the actual gentoo docker image.
July 17 - July 24 (Week 9)	Create client side scripts to execute the jobs obtained from the BOINC server. Integrate BOINC server with the server needed by Travis CI, for updation of packages after stabilization. Complete the package caching implementation.
July 24 - July 31 (Week 10)	Testing Phase. Test with various packages, build scenarios, and stress testing of CI and distributed builds, along with logging of performance.
July 31 - Aug 7 (Week 11)	Testing Phase.
Aug 7 - Aug 16 (Week 12)	Refine the code to make it more presentable. Add final touches to the documentation, and ready the project for submission.

Biography:

I am a sophomore Computer Science Undergraduate studying at the Indian Institute of Technology, Kanpur.
I'm a Linux enthusiast, an avid Developer; and have a plethora of experience with development in its various manifestations, viz Web, App, Package writing to name a few. I have above-par knowledge about Linux systems due to my intensely concentrated experience with Linux since the past few years.

I've been a long time user of various Linux distros (starting with Ubuntu, I have since used Debian, Fedora, Arch, Gentoo, and NixOS). I even went as far as to compile the whole operating system without a package manager while following the LinuxFromScratch (http://www.linuxfromscratch.org/lfs/) project. I have been fascinated by Gentoo and Portage, ever since I forced its (Portage's) installation in a completed LFS system just to

see if it would work
(sort of something like this https://forums.gentoo.org/viewtopic-t-28559-start-0.html).

My previous experience includes contributing build recipe for WPS Office package for NixOS (https://github.com/NixOS/nixpkgs/pull/13155) and writing the debian packaging scripts for Pathpicker (https://github.com/facebook/PathPicker/pull/117).

Some of the things I have worked on before this, include:

1. Summer project on interpretation and visual simulation of a C program.
   (Used: Python, knowledge of C compiler)
   a. Github link: https://github.com/pallavagarwal07/its
      https://github.com/pallavagarwal07/its_server
   b. Demo:
      http://www.varstack.com/2015/10/06/Introduction-to-Cimulator/
2. Shuttle cock tracking and trajectory prediction as a part of International Robotics competition Robocon. (Used : C++, OpenCV)
   a. Bitbucket link https://bitbucket.org/robocon15/rcon
3. My Blog (Used: HTML, CSS, JS, Ruby)
   a. Github Link: https://github.com/pallavagarwal07/pallavagarwal07.github.io
   b. Blog Link: http://www.varstack.com

I was part of the team that came first in Code.Fun.Do, a one day hackathon conducted by Microsoft in our campus, in both our first and second year.

I have also qualfied the regionals of ACM ICPC (https://icpc.baylor.edu/), as well as the first level of Build The Shield (https://buildtheshield.microsoft.com/india/).

Contributions:

As a preparation for this project, on communication with the mentors, I have written basic scripts that estimate the number of stabilization jobs the current tree would require (http://paste.ubuntu.com/15356039/), as well as evaluate the dependencies of Gentoo packages and attempts to build it ((http://paste.ubuntu.com/15356041/)).
I also wrote and contributed an ebuild for a simple package

(https://bugs.gentoo.org/show_bug.cgi?id=578116).
For the dependency script, I also built a custom Gentoo Docker Image and ran the above scripts as a part of Travis CI as a proof of concept.
(https://travis-ci.org/pallavagarwal07/gentoo/builds/114632357)
I have also looked into various helper tools for the packaging systems including, but not limited to eix, egrep, and tatt.
I have also made a pull request to the github repository gentoo/gentoo to help resolve Bug#557308 (https://bugs.gentoo.org/show_bug.cgi?id=557308)

Working Hours:

My semester vacations would start from April 29, and end on July 20. During the vacation, I would be available for work and be online on IRC any time I am awake. However, when the semester resumes, on July 21, the working hours would be determined by the schedule of the semester.

Contact:

Phone:	+91 7408997854
Email:	pallavagarwal07@gmail.com pallavag@iitk.ac.in
Github:	pallavagarwal07
Skype ID:	pallavagarwal07
IRC Nick: (freenode)	pallav
Blog:	http://www.varstack.com/
Address:	A-311 Hall of Residence 2 Indian Institute of Technology Kanpur, Uttar Pradesh India - 208016