CRANstats User Manual

Timothy P. Jurka

August 17, 2012

1 Overview

1.1 Introduction

CRANstats is a web service that aggregates Apache access logs from CRAN mirrors and presents the data to the R community. An R script running on each mirror parses the logs, removes user identifiers, and submits them to the server. The logs are then stored in a database that is accessible via the web interface. The service was built as part of Google Summer of Code 2012.

1.2 Technologies

CRANstats was written primarily in Python using the web.py framework. The application is hosted on Heroku with a PostgreSQL database. memcached is used to reduce request time under heavy load. On the frontend, visualizations are generated using flot and jQuery. The site layout is built using the Twitter Bootstrap framework.

2 Installation

2.1 Prerequisites

CRAN mirrors must have R and run on Apache with logging enabled.

2.2 Apache

Although most of these settings are the defaults for Apache, some configurations may need to be changed to work with cranstats.

2.2.1 Log Format

Logs must be formatted using the combined log format.

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-agent}i\"" combined
CustomLog log/acces_log combined
```

2.2.2 Log Rotation

If using log rotation, the logs must be named using the standard convention. All access logs should have names that begin with “access.log”. Logs can be gzipped as long as the gzipped file abide by the aforementioned naming convention.

2.3 R

Two scripts are provided to CRAN mirror maintainers. “install.R” is a simple script that installs the required packages for cranstats. “cranstats.R” sends the logs to the cranstats server. “cranstats.R” needs minor configuration before it can be used.

2.3.1 APACHE_LOGS_PATH

This variable should point to the directory that contains the access logs.

2.3.2 KEY

This variable should hold the API key assigned to you by the cranstats maintainers.

2.3.3 SERVER_URI

This variable should not change unless instructed by the cranstats maintainers. It points to the cranstats file server where access logs are uploaded.

2.3.4 VERIFY_URI

This variable should not change unless instructed by the cranstats maintainers. It points to the cranstats node manager that tracks the status of each mirror.

2.4 Job Scheduler

Once configured, the “cranstats.R” script should be scheduled to run once a day using a job scheduler such as cron.

3 Workflow

3.1 CRAN Mirror

The CRAN mirror runs “cranstats.R” once a day using a job scheduler. The R script contacts the cranstats server to get information about the last synced entry from the mirror. The script retrieves all access logs and filters out any entries older than the last synced entry. This process is also performed server-side. The following parameters are then parsed from the logs: date downloaded, package name, package version, operating system, R version, and architecture. The parsed data is then encoded as JSON and compressed using gzip. The compressed data is then uploaded to Amazon S3, which notifies cranstats to process the logs upon successful upload.

3.2 CRANstats

3.2.1 Backend

CRANstats pulls the uploaded data from Amazon S3, unzips it, and adds it to the PostgreSQL database. The file is subsequently deleted from Amazon S3.

3.2.2 Frontend

A user arrives at a package page, and the browser issues a GET request for the package data within the specified date range. The server checks memcache to see if this request has already been cached. If not, it issues a SELECT query to PostgreSQL and retrieves the results. The server then calculates the frequencies for each category (downloads, architecture, package version, operating system, and R version), and returns the frequencies to the browser. The browser then generates the visualizations using flot. Code is also available to calculate the frequencies client-side, however this does not significantly reduce load on the server, and requires much more data to be downloaded by the client (i.e. the raw SQL query results).