Bug Insertion

Gema Rodríguez Pérez

Abstract

The objective of this paper is to prove that the bug can be not caused by the previous commit. Currently, the premise establishes that all the bugs were introduced by the previous commit. We have carried out an experiment which has helped us show results pointing out a minor disagreement with the present premise. We have observed three stages totally separable throughout the execution of this experiment. The first stage considers a classification, based on our knowledge of the ticket, in three groups: 1.It is an error. 2.It is not an error. 3.Unknown. The second stage involves the analysis of it as an error. In this stage where, with some parameters of interest, we are able to give a percentage of how many prior commits were responsible for the bug. The third stage is the implementation of an analytic tool to help us with the first and second stages. In this paper we focus in two first stages. Explaining clearly and concisely the methodology used in our experiment.

1 The experiment

Our experiment consisted of the analysis of one hundred tickets which were taken randomly from Cinder repository. We will call the first stage ‘the filtering’ as we will classify them into three categories:

1. It is an error.
2. It is not an error.
3. Unknown

Two different people with knowledge about programming were responsible for classifying these tickets in accordance with their opinion of the three categories mentioned above. The two people worked in parallel during the filtering. Only when all the tickets were classified were both classifications compared. Those whose classification were different, were compared with each other, thus reaching an agreed classification. After filtering, we continued to the next stage, in which we only kept in mind those tickets belonging to 'It is an error' group. In this phase we were able to conclude whether the bug was added in an earlier commit, or by contrast, if it was not. This stage is still being validated on account of a shortage of tickets analyzed. At this moment, we can design an efficient algorithm that involves relevant parameters to reach the conclusion that we are looking for. So, for the moment, we are using these parameters as a support.

2 Several questions we have to answer

To provide a context, we have to answer some questions referring to the project from which we have taken the information. Likewise, we are going to describe the specific vocabulary.

- What is OpenStack?

  It is a combination of software tools for building and managing cloud computing platforms for public and private clouds. Principally, users deploy it as an infrastructure as a service (IaaS) solution. The technology consists of many different moving parts. In particular, OpenStack has nine key components that can be identified as part of the "core". Officially, OpenStack community maintained these system.

  - Compute (Nova) is the primary engine of an IaaS system. It is used for deploying and managing virtual machines.
  - Object Storage (Swift) is a scalable redundant storage system for objects and files.
  - Block Storage (Cinder) manages the creation, attaching and detaching of the block devices to servers, being able to access specific locations on a disk drive.
  - Networking (Neutron) provides the networking capability for OpenStack managing IP addresses.
  - Dashboard (Horizon) is the only graphical interface to OpenStack whereby administrators access.
  - Identity (Keystone) provides a central directory of users mapped to the OpenStack services they can access.
  - Image Service (Glance) provides virtual copies, services to OpenStack. It allows to use these images as templates when deploying new virtual machine instances.
  - Telemetry Service (Ceilometer)
  - Orchestration (Heat) helps to manage the infrastructure needed for a cloud service to run through both a REST API and a Query API.

  Each one of them has its own API to achieve integration. Furthermore, all of them have a repository where the community implements improvements; fixing bugs ... etc.

- How can OpenStack help us?

  OpenStack is of an open nature, anyone can add additional components to OpenStack to help it to meet their needs. Because of the high scope of OpenStack we have at our disposal several bugs that we are interested in the first of which is in Cinder.

- What is Cinder in OpenStack?
- What would we like to achieve by analyzing this repository?
- What do we have to analyze in this repository
- What is a ticket?
- Where will we find a ticket?
- What gives us the title and description of a commit?
- What criteria will be used?
- What kind of code is it?
- What number of commits are involved?

- What will happen if the number of folders is greater than one?
- Is it only the previous commit that is involved?
- Is this commit the responsible?

3 The resuts

4 Conclusions