Simple OpenStack Monitoring Tool

Acceptance Test Plan

Version 1.2

Team Name: Oceans11

Team Members:

- Tarun Aluguri.
- Nikhil Reddy Araga.
- J N S Sri Harsha Vardhan Kamisetty.
- Prathisrihas Reddy Konduru.
- Sai Anoop Nadella.
- Rohit Pothuraju.
- Dilip Renkila.
- Venkat Sivendra Tipirisetty.
- Sai Bhargava Ravi Teja Vedantam.
- S. Sai Srinivas Jayapala Vemula.
- Rahul Vudutha

I. PREFACE:

The main concern of the project is to develop a simple and intuitive web based drill down GUI that provides an overview of an OpenStack environment. The tool monitors the OpenStack services Nova, Neutron, Cinder, Swift, Keystone, Glance and Heat existing on the nodes. This is revised version of the document (version 1.2).

A. *Release 1.2 on 30th May 2015:*

- *Made changes regarding operation and environment of frontend , database and backend tests.*
- *Made changes regarding tests regarding database module*
- *Added tests to the backend module*

B. *Release 1.1 on 14th May 2015:*

- Made changes in D-T2 and D-T3 (Section 4.2).
- Made changes regarding open stack monitoring services (Section 5).

C. *Release 1.0 on 20th May 2015:*

- Initial Release

II. GLOSSARY AND ABBREVIATIONS:

1) API: Application Programming Interface:

An **API** is a set of routines, protocols, and tools for building software applications.

2) GUI: Graphical User Interface:

A **GUI** is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces.

3) PHP: Hypertext Preprocessor

PHP is an Open Source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

4) PERL: Practical Extraction and Report Language

Perl is a general-purpose programming language developed for text manipulation and also used for tasks including system administration, web development, network programming, GUI development, and etc.

III. ACCEPTANCE TEST PLAN:

F-T1: Authentication

Test: Login page test

Purpose: To prevent unauthorized users from gaining access to the dashboard

Requirement: Req_SYSF1, Req_SYSF2, Req_SYSNF2

Environment: Browser for rendering the webpages, PHP5

- Preinstalled php5
  - Open terminal
  - Run command: sudo apt-get install php5
- Preinstalled apache server
  - Run command: sudo apt-get install apache2
- Preinstalled MySQL database
  - Run command: sudo apt-get install mysql-server
  - Database must be created with a table containing user credentials.

Operation:

- Open web browser.
- Go to login.php from localhost and enter user credentials.

- If correct credentials are entered, it will redirect to Dashboard
- If incorrect credentials are entered, alert will be shown and access will be denied

Expected Result: Displays Tool Dashboard upon entering correct username and password

Result:

Comment: Before rendering the webpage apache2 status must be checked: "service apache2 status" on the terminal.

F-T2: Export 3rd party data

Test: REST API test

Purpose: To export data to a 3rd party using RESTful API

Requirement: Req_SYSF1, Req_SYSF2, Req_SYSNF2

Environment: Web browser, apache server

- Preinstalled php5
  - Open terminal
  - Run command: sudo apt-get install php5
- HTTP communication environment
- Preinstalled apache server
  - Run command: sudo apt-get install apache2

Operation:

- Open web browser
- In the web address bar, enter the GET request for data regarding a particular service by typing

<SERVER IP>/frontend/Oceans11_rest_api.php/?service=<name_of_service>

Expected Result: Status and uptime data, of the service requested, are displayed in JSON format.

Result:

Comment: Care must be taken not to overload the network that may slow down operation.

F-T3: Receive update on status and uptime of service

Test: Notification test

Purpose: Check the automatic update of service status and service uptime in dashboard

Requirement: Req_SYSF1, Req_SYSF2**,** Req_SYSNF2

Environment: Web browser, apache server

- Preinstalled php5
  - Open terminal
  - Run command: sudo apt-get install php5
- Preinstalled apache server
  - Run command: sudo apt-get install apache2
- Preinstalled MySQL database
  - Run command: sudo apt-get install mysql-server
  - Database must be created with tables containing fields for status and uptime for each service
  - Each service has an independent table containing the list of sub-services

Operation:

- Open web browser
- Enter the dashboard through the login page
- On the home page, to the right, a service status panel displays list of services being monitored and current status for each of them (Running/ERROR)
- Whenever a service has stopped, the status is displayed in red color
- On the statistics page, the uptime of each service is displayed. It is also updated whenever a service is restarted

Expected Result: Status changes from Running to ERROR when a service stops. When a service is restarted, the uptime is updated and reset.

Result:

F-T4: Plot graphs on uptime of each service

Test: Graph test

Purpose: Check generation of graphs by Perl script using data in MySQL database

Requirement: Req_SYSF1, Req_SYSF2**,** Req_SYSNF2

Environment: Web browser, apache server

- Preinstalled php5
  - Open terminal
  - Run command: sudo apt-get install php5
- Preinstalled apache server
  - Run command: sudo apt-get install apache2
- Preinstalled MySQL database
  - Run command: sudo apt-get install mysql-server
  - Database must be created with tables containing fields for status and uptime for each service
  - Each service has an independent table containing the list of sub-services
- Preinstalled GD::Graph perl module
  - Open terminal
  - Run command: sudo apt-get install libgd-graph-perl

Operation: Creates graphs on Uptime of each service and passes it to frontend

- Open web browser
- Enter the tool dashboard
- In the Statistics tab, list of services are displayed with their respective uptimes.

- Select a particular service, the Uptime graph for that service is displayed

Expected Result: Uptime graph is plotted for each service based on data stored in MySQL database

Result:

D-T1: MySQL database contains user credentials

Module: PHP::MySQL

Purpose: For interaction between PHP script and MySQL

Requirement: Req_SYSF1, Req_SYSF2, Req_USRF1, Req_USRF2

Environment: MySQL database contains a table in the database for user credentials

- Preinstalled php5
  - Open terminal
  - Run command: sudo apt-get install php5
- Preinstalled MySQL database
  - Run command: sudo apt-get install mysql-server
  - Database must be created with tables containing fields for status and uptime for each service
  - Each service has an independent table containing the list of sub-services

Operation:

- Open web browser
- Enter the tool dashboard, click on "Register new user". Enter the requested credentials. A new user is created
- Open mysql in terminal by entering $ mysql –u root -p
- Check 'users' table in the tool's database for the registered user credentials

Expected Result: The table contains user credentials as created from the frontend

Result:

D-T2: MySQL database contains status and Uptime data

Module: DBI, DBD::mysql

Purpose: To ensure that database contains status and uptime

Requirement: Req_SYSF1, Req_SYSF2, Req_USRF1, Req_USRF2

Environment: MySQL database contains a table in the database for user credentials

- Preinstalled perl DBI
- Preinstalled MySQL database
  - Run command: sudo apt-get install mysql-server
  - Database must be created with tables containing fields for status and uptime for each service
  - Each service has an independent table containing the list of sub-services

Operation:

- Run backend script so that the script can monitor the OpenStack environment and insert status and uptime into the corresponding tables.
- Open mysql in terminal by entering $ mysql –u root –p
- Check '<service_name>' table in the tool's database for the status and uptime of each sub-service.

Expected Result: The '<service name>' table contains the status and uptime data.

Result:

B-T1: Ensure establishment of secure connection between user and server

Module: Net::OpenSSH

Purpose: To execute commands on remote server's terminal

Requirement: Req_SYSF3, Req_SYSNF1

Environment:

- OpenStack node.
- SSH public key to be placed in the authorized_keys directory in /root/.ssh of the host.
- The corresponding public and private key pairs are to be placed in the keys directory of the tool.
- Update the key passphrase in db.conf

Operation:

- Open terminal
- Execute backend.pl which contains perl files that need to be running in the background continuously
- As sshcon() is a subroutine in sshcon.pl, the successful execution of backend.pl ensures the successful establishment of a ssh connection between tool and OpenStack node

Expected Result: Successfully establish a secure connection between OpenStack node and backend

Result:

B-T2: Status and Uptime of services are inserted

Module: DBI, DBD::mysql

Purpose: To insert status and uptime in mysql database that is later fetched by frontend to be displayed on web page

Requirement: Req_USRF7, Req_SYSF2, Req_SYSF3

Environment:

- OpenStack node.
- Established SSH connection between backend and OpenStack node
- MySQL database containing tables for each service

Operation:

- Open terminal
- Execute backend.pl which contains perl files that need to be running in the background continuously
- As status() is a subroutine in status.pl, the successful execution of the background perl files ensures the retrieval of status and uptime of services and their insertion into their respective tables.

Expected Result: Successfully establish a secure connection between OpenStack node and backend

Result:

B-T3: Service restart

Module: Net::OpenSSH, DBI, DBD::mysql

Purpose: To restart a service upon demand and to show number of restarts of each service

Requirement: Req_USRF2, Req_SYSF2, Req_SYSF3

Environment:

- OpenStack node.
- Established SSH connection between backend and OpenStack node
- MySQL database containing tables for each service

Operation:

- Open terminal
- Execute backend.pl which contains perl files that need to be running in the background continuously
- As restart() is a subroutine in restart.pl, the successful execution of the background perl files ensures the restart of a service and the update of number of restarts in the respective tables

Expected Result: Successful restart of service and update of number of restart in the respective service table.

Result:

IV. REFERENCES:

- http://php.net/manual/en/book.mysql.php
- http://search.cpan.org/~capttofu/DBD-mysql-4.031/lib/DBD/mysql.pm
- https://metacpan.org/source/SALVA/Net-OpenSSH-0.64/lib/Net/OpenSSH.pm