Design of Generic Web Based Automation Framework for Network Testing

S. Balamurugan
Assistant Professor, Department of Information Technology,
Perunthalaivar Kamarajar Institute of Engineering and Technology,
Nedungadu (PO), Karaikal-609603, India

Abstract **- Network software and hardware test applications used by network administrators are available to manage and test network of all sizes, from smallest home network to largest organizational network but these test applications are not automated. Manual work is needed to perform network test, maintenance and monitoring of active data network or converged infrastructure and related network devices like switches and routers. When testing an organizational network environment many aspects need to be taken into consideration. For example different software need to be deployed at different nodes, different test cases are to be executed using different topologies etc. The manual execution of these test cases is very time consuming and also it is tiresome task to analyze network data and traffic. To overcome these difficulties we proposed a web based GUI framework to automate network testing. With the use of the web framework, we efficiently improve the extensibility and reusability of automated test cases. The results shows that the new web framework improves testing quality, test efficiency with reduced man power it also provides a user friendly environment to execute the network testing**.

1. INTRODUCTION

Network testing is performed to test, analyze the network information and to manage the network. But network consist of voluminous data which cannot be identified by an individual there may be high percentage of human error that can occur easy. Also consumes more time to conduct test.

Test automation may be able to reduce or eliminate the cost of actual testing. It automate previous repetitive but necessary testing in a formalized testing process already in place, or add additional testing that would be difficult to perform manually. An automated system can follow a sequence of steps more quickly than a person, and it can run the tests overnight to present the results in the next day. However, the labour that is saved in actual testing must be spent instead authoring the test program.

Our implementation of automation framework is a generic and user friendly web based framework to carry out sanity, functional, regression and performance testing. It also provides a place holder for different type of network devices and device under test. User can also develop his own test cases and test topologies and integrate them to run the suite collect and analyze the results.

2. NEED FOR GENERIC FRAMEWORK

In smaller organization, network testing is done by network administrator and he is technically involved in maintenance of servers, desktop computers, servers, routers, firewall etc. In certain cases the organization will employ a separate team for manually testing the developed software. Organization has to invest more money and time to successfully complete the network testing.

Network Administrators are generally mid-level support staff within an organization and don't typically get involved directly with users.

They focus upon network components within a company's LAN/WAN infrastructure ensuring integrity. Depending on the company and its size, the network Administrator may also design and deploy networks. An organization may design their own framework for their network and automate them. Generally this increases the cost and no common framework to do network testing and managing network. Proposed generic framework can be adopted according to the organizational needs and then it can be tested with or without any additional tools. It ensures that this test automation framework can be compatible with the same type of many applications, without any major modification.

3. FRAMEWORK DESIGN

We have designed our test automation framework to support wide variety of network tools and web browsers primarily for solving web application compatibility across the browsers.

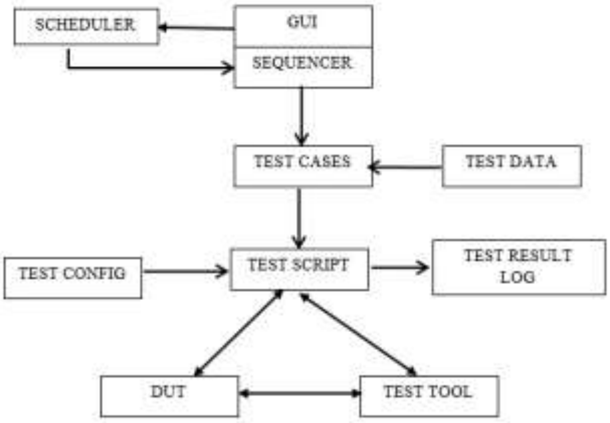

Figure 1. Core Framework

It provides testers with simple web interface and allows the testers to choose and configure network topology which is used to run the automated test cases. In order to provide testers a convenient and flexible test environment, out test automation framework uses a web graphical user interface. The automated testing framework designed in this paper includes nine components as shown in Figure 1.

3.1 Scheduler

This component is used to schedule the test according to the user convenient. This module will be helpful when running the regression test cases. Regression test cases may take longer time to complete the execution. It might be a day or more. So these types of test cases should be run during off working hours of an organization to avoid idle hours of employees when regression is running. With the help of the scheduler the tester can select date and time on which the test cases should be executed and schedule to runs at that particular time.

3.2 GUI

It provides user friendly interface to build, modify and run test suites. In our case web interface is developed to attain full functionality of GUI. The web graphical user interface developed for this automation test process the following characteristics,

a. Easy Accessibility
b. Ease of use
c. Increased productivity
d. Understandable by non-technical users

3.3 DUT

Device Under Test is the actual test device undergoing testing. It may be referred as Equipment Under Test or Unit under Test. The DUT is checked for defects to make sure that all the features in the device are working correctly. DUT is usually tested manually; nowadays automatic and automatic test equipments are used to conduct complex testing. In context of networking the device can be newly built routers, switches, gateways etc.

3.4 Sequencer

Sequencer is designed to accommodate the test case in sequence of events to be executed automatically. This component will execute the test case in sequence order of the test case selected in the suite. The user is able to add, delete or modify the test sequence according to his requirement. There might be certain conditions where one test case depends on other test case, in such situations the sequencer will give us greater support to execute the test cases in user selected order.

3.5 Test case

Test case is the core part of testing various test cases are selected according to the network to be tested. A formal written test-case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition. The Test cases are classified into following categories,

a. Sanity test
b. Functional test
c. Regression test
d. Performance test
e. Stress test

Each of the classification in will have test scripts which are written in any scripting languages like TCL or PERL.

3.6 Test data

Test data plays a major role in automation testing, it is data which has been identified for use in the test case to confirm the expected output and declare the test results. Some test data may be used to check the ability of the software to respond in certain exceptional and unusual cases. Test data is generated by manual testers or by automation engineers who support testing. In regression testing the test data is re-used, and it is always a good practice to verify the test data before re-using it in any kind of test. All the test data must be validated and verified by the higher management within the testing team before it is released to the test execution. The team members are not authorised to change the test data without the approval of manager.

3.7 Test configure

Test configuration file is used to configure the network topology according to the given test case. For example for each test case the peer IP addresses, port numbers, protocol configuration etc. need to be configured before the test run starts. All these configuration details are placed in a separate script file called test configuration. There may be one to one mapping between test configuration and test case or many test cases can be mapped to a single test configuration file. For each test script inside the test suite, calls test configuration procedure before the actual testing and sending data traffic.

3.8 Test script

Test scripts are the step by step instructions which are written in any scripting language like TCL or PERL to test the DUT. Each test script file will have the following modules, which are locking test bed(s), test bed configuration, traffic generation, test analysis, logging, unlocking test bed(s) and result declaration. The scenarios for simple test scripts are as given below

a. Configure the subnet and check for connectivity
b. Establish TCP connection between two nodes
c. Verification of routing table
d. Configure routing protocol and pump IPV4 Traffic.

3.9 Test results

The tested network results are stored in log file and maintained in database. The tested data are converted into graphical from which is easy for user to analyze and troubleshoot the problems in the network. This log used for the future reference. There will be a common template for displaying the summary of test runs. For each test suite, the summary displays the following details,

a. Test Suite Name
b. Test Suite ID
c. Test Initiator name
d. Test Start time
e. Test End time
f. No. of Test cases
g. No. of Test cases Passed
h. No. of Test cases Failed
i. No. of Test cases aborted

Also, for each test case detailed log file is maintained in the log directory, this file is generated while the test case is in execution. This log file is used by the test engineers to debug the failed test cases. Once the test results are published the test summary is automatically mailed to the higher management and all other team members who involved in that specific project.

4. WEB REFERRAL ARCHITECTURE

Web referral architecture is a challenging task when we think about network testing. Many test automated test tools are available for application level testing. But, for network testing our architecture is going to create a new point of reference in network test automation. The web based testing increases more user level interaction to manipulate test cases and no extra knowledge is needed to perform the test.

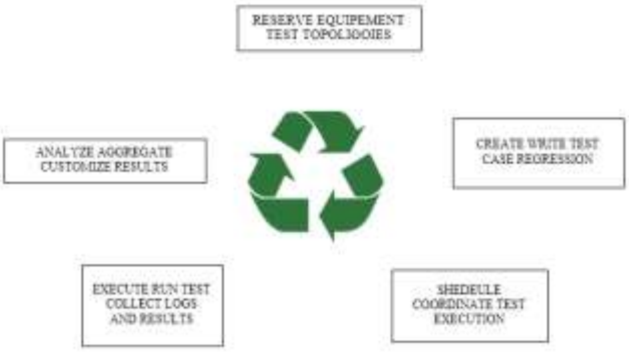
RESERVE EQUIPEMENT TEST TOPOLOGOIES
CREATE WRITE TEST CASE REGRESSION
SHEDEULE COORDINATE TEST EXECUTION
EXECUTE RUN TEST COLLECT LOGS AND RESULTS
ANALYZE AGGREGATE CUSTOMIZE RESULTS

Figure 2. Web Architecture

General Structure of the Integrated Automation Testing Framework .The integrated automated testing framework designed in this paper includes five components listed the following Figure 2.

Reserve equipment topology**:** This component selects and reserves the topology to perform the test. This module logically locks the test bed till the completion of the test case so that other test cannot be performed until reserved node is tested and the topology is unlocked or released by this module.

Create test case regression: This function is to create a new test suite by adding, deleting and modifying the existing test case. Normally this will be done only when newer version of the software is released.

Schedule coordinate test execution: Test runs are scheduled according to the user convenience. With help of this module, the user can set the date and time for each test suite.

Execute test collect logs: This module will start the test run once the date and time is reached. On completion of the test run the test results are stored in the log file which records of all tested activities. It also sends out summary of test results to the user and higher management.

Aggregate Analyze results: After testing the network the results are analyzed and compared with the previous results to find new bugs in the software.

In this web framework, the network environment to be tested is first identified (i.e.) the topology of the network (LAN topology, single host) is selected. Then the topology is locked until current test completes the execution. The test run can also be scheduled at different time based on user preference. After performing the test the results are gathered and analyzed for real bugs or any network issues.

5. INTEGRATION METHODOLOGY

Integration is the process of linking together different modules to give single functionality. The following deployment diagram shows the various modules of framework and the interaction between the modules within the web server, database, GUI and the user.

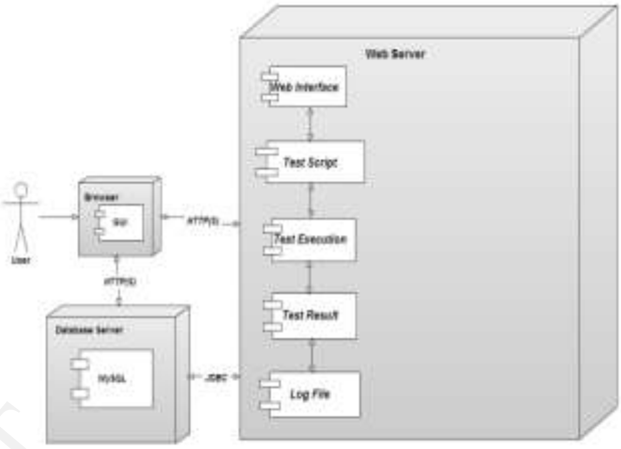

Figure 3. Deployment Diagram

In the web framework the web page is designed using web designing language and stored in local web server. The dynamic web page is created for user to interact easily with the test suite. Different types of testing like sanity, regression, function, performance etc. are considered to be the test suites. For each of the test suite, the identified test cases are scripted and the script files are stored in the web server. The test scripts are written using the scripting language like TCL and PERL which supports networking functionality.

Integrate the test suite with the web server. After integration, the user can select the test suite and start the execution, and the test suite will perform the execution of test case and the logs are displayed in the browser and also stored in the log files.

6. STEPS TO INITIATE AUTOMATION

The following are the steps to initiate a test run in our automation framework.

- Open up the automation webpage present in local machine or from a remote web server.
- Use, Modify or write new test case for the given feature and update the test suite.
- Select one or more test bed upon which the test case needs to be executed.
- Select the test tool.
- Lock the test bed and test tool upon which the testing is to be done.
- Select the individual test suites and test cases within the test suites, depends upon the type of testing need to run.

- Schedule date and time for the execution, if needed.
- Execute the test case and collect logs.
- Analyze the failures and troubleshoot the problems.

7. IMPLEMENTATION

The framework is implemented using PHP, TCL, and SQL. PHP is popular for its open, simple, easy transplantation stability and high running speed. It is used for building Web based applications of all types. In the context of Web-based application development, a technological platform is a programming language and the set of reusable technological components used in conjunction with that language, such as components, frameworks, libraries, tools, and auxiliary languages. Web development platforms share a number of auxiliary languages, in particular HTML, CSS, and JavaScript. PHP is an open source, server-side, cross-platform, object-oriented scripting language for creating dynamic Web pages. Users insert the PHP code into a page's HTML code. The PHP Data Objects extension defines lightweight and consistent interfaces for accessing various relational databases.

SQL is a compact and powerful language for working with databases. Used for formulating statements that is processed by a database server. A database server enables users to process data stored in a database.

All the test scripts are scripted using a well known command language TCL. It is a simple scripting language for controlling and extending application. It provides generic programming facilities, such as variables, loops, procedures. The TCL library consists of a parser for the language, routines to implement the TCL built-in commands, and procedures that allow each application to extend TCL with additional commands specific to that application. TCL provides two commands that support string matching using egrep-style regular expressions: regexp and regsub. Another important package in TCL is expect package, which is used to connect to the remote computers and execute the commands. Because of these functionalities we used TCL as our scripting language.

The framework is implemented to check the compatibility, integrity, speed and quality of testing. Framework is deployed at particular host with a processor speed equivalent or more 2.6MHz, 4GB of main memory. Linux OS is installed in the host with web server and TCL and expect package. The webpage design is almost compatible with any of the recent web browsers.

8. RESULTS

Whenever a user requests the server for conducting a test execution, the server calls the appropriate test suite and execute, results are displayed in the web page for immediate reference. However all the results are stored in the log file for future reference. The framework is implemented and tested in local network successfully.

The result page shows the tested results. In Figure 4, the IP address of the system is identified and then connectivity is checked using ping. Also list the active and non-active ports using net stat, list of services, system load, no of users, uptime for the system, port and services upon it. These results are written in log file shown in Figure 5 and stored in database.

After performing the test, a set of test result are obtained in web page as shown in Figure 4.

Figure 4.Test result page

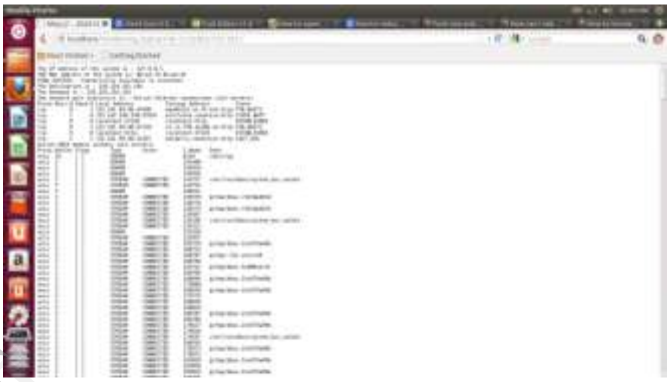

Figure 5. Result log file

9. CONCLUSION

The new network testing automation integrated with web can share the test steps and test data among different testing, such as UI testing, router testing, loading test and so on. It is convenient to switch in various types of testing for network environment. It supports multiple browsers and a variety of operating system. It can be widely used in web network test automation.

In order to ensure uniformity, the test automation framework designed in our work not only provides testers with unified test cases, but also leaves the extensibility space for introducing other test tools in the future. Using this test automation framework, the testers can configure a test type easily just through modifying a test type and can switch among different test by altering the elements related to test cases.

10. REFERENCES

[1] Feg Wand, Wencai Du ," A Test Automation Framework Based on WEB",2012 IEEE/ASIC 11th International Conference on Computer and Information Science pp.683-687

[2] C.F. Coombs, Jr., and C.A. Coombs, Communications Network Test & Measurement, PUBLISHER'S LOCATION: McGraw Hill, 1998, ISBN 0-07-012617-8.

[3] http://en.wikipedia.org/wiki/Network_Test_Automation_Forum

[4] http://www.tmworld.com/design/design-and–protottping/ 4389164,Automating-network-test

[5] http://en.opensuse.org/SDB:Linux_Apache_MySQL_PHP