Introduction

Automation process is implemented using Robert Framework and Jenkins, it provides a framework including:

- Download daily build
- Install build on VMs
- Execute test case
- Generate test report
- Send test result

Prerequisite

Here, take Linux OS as example, you could also set up an environment under Windows OS without much difficults.

1. Robert Framework

- Install python (2.7)
- Install pip
- Pip install robotframework
- Pip install robotframework library
    - robotframework-sshlibrary
    - robotframework-httplibrary

2. Install and configure Jenkins
3. For PXE environment, refer to the last section

If what you want is just to compose and debug automation case, then step1 is enough.

Test Case

Test case files can be organized into directories, and these directories create higher-level test suites. A test suite created from a directory cannot have any test cases directly, but it contains other test suites with test cases, instead. Currently all cases are organized as the structure of Test Link, with 2 layers of directory.

The basic grammer and semantics can refer to its UserGuide here.

All test cases are located under /work/automation-test/RF-Automation/testcase. Robert Framework supports custom keywords, something like common function. If many cases share similar steps, we can extract the steps as a keyword. For those keywords may shared by most test cases, it is suggested to append to the 00_commonkeyword.txt. 00_commonconfig.txt is used to store common configurations, e.g. IP address, username, password...

With existing library and common key word, it is very easy to add new test case. Below is an example of test case, part of 05_Virtual_Storage/04_iscsi_general.robot.

```
*** Settings ***
# Library is to import installed lib, and Resource is to
```

```
import custom keyword
Library           OperatingSystem
Library           SSHLibrary
Library           HttpLibrary.HTTP
Resource          ../00_commonconfig.txt
Resource          ../00_commonkeyword.txt

# Suite Setup is executed before all cases in this robot,
 Suite Teardown is executed after cases. Suggest clean up
 all configurations during this suite, it can reduce the
possible risk that may affect the following cases
Suite Setup       Run Keywords    Open HTTP Connection An
d Log In    @{PUBLICIP}[0]    ${UIADMIN}    ${UIPASS}
...               AND    Open All SSH Connections    ${US
ERNAME}    ${PASSWORD}    @{PUBLICIP}
Suite Teardown    Close All Connections    # Close SSH co
nnections

# Can Define some variables here, those variables are vis
ible only in this robot file
*** Variables ***
${vs_name}        Default
${default_pool}    Default
${iscsi_target_name}    iqn.2016-01.bigtera.com:auto
${iscsi_target_name_urlencoding}    iqn.2016-01.bigtera.c
om%3Aauto
${iscsi_lun_name}    lun1
${iscsi_lun_size}    5368709120
```

```
*** Test Cases ***
Add iSCSI volume
    [Documentation]    Testlink ID:
    ...    Sc-540:Add iSCSI volume
    [Tags]    RAT    # Tags are useful when analyze repo
rt or execute filtered cases
    # You can put some pre work in section [Setup](test-c
ase level) if needed.

    # Check pre-environment
    SSH Output Should Be Equal    scstadmin --list_device
 | grep vdisk_blockio | awk '{print \$2}'    -

    # Call API to finish the steps
    Add iSCSI Volume    gateway_group=${vs_name}    pool_
id=${default_pool}    target_id=${iscsi_target_name_urlen
coding}    iscsi_id=${iscsi_lun_name}    size=${iscsi_lun
_size}

    # Check result from server, the keyword "Wait Until K
eyword Succeeds" is very useful to check results that are
 not take effect immediately
    Wait Until Keyword Succeeds    30s    5s    Check If
SSH Output Is Empty    rbd showmapped    ${false}
    Wait Until Keyword Succeeds    30s    5s    SSH Outpu
t Should Match    scstadmin --list_device | grep vdisk_bl
ockio | awk '{print \$2}'    tgt*
```

```
    # Check result from client
    ${output} =    Run    sudo iscsiadm -m discovery -t s
t -p @{PUBLICIP}[0]
    Should Contain    ${output}    ${iscsi_target_name}

    # You can put some post work in section [Teardown](te
st-case level) as needed.
    [Teardown]
    ${rc} =    Run and Return RC    sudo iscsiadm -m node
 -o delete
    Should Be Equal As Integers    ${rc}    0
```

As you can see, the key thought is to first call web API to configure, then use credible way to validate the results. It makes automation cases easier to be implemented, but this way has 2 disadvantages:

- *It cannot cover UI bugs, like js error or UI display issue*
- *API may have changes in further version, which require case update*

In further, we may introduce Seleuium to do UI things, but consider the UI response time may affect the accuracy, it has lower priority. After all, web API way can cover most work that regression cycle should have, and it is usually unchanged and stable.

Work Flow

1. First, Jenkins starts a task at 4 o’clock A.M. everyday
2. The main script is responsible for retriving daily build from TW server
3. Then, the main script calls pybot to start a root suite
4. The first suite(00_Install_ISO) is to install builds in ESXi VMs
5. The second suite(01_Create_Cluster) is to prepare env and create cluster
6. The rest test cases are followed to be executed
7. Framework generates report html and log xml
8. Jenkins sends result with build log

PXE Env Preparation

Official Ubuntu OS can be installed in PXE way, just with the online guide. There is also a good reference for automating the installation using preseeding. But our OS is a little different from official one, so the installation is a little more complicated. The PXE install with http has some troublesome issues. Thanks to David, he provides a way without http, but vblade to install OS from network CD. So we can install build unattendedly, just with a .iso file in LAN.

Below is the steps to record the preparation of the PEX server environment.

1. Install TFTP:

- apt-get install tftpd tftp xinetd

- vi /etc/xinetd.d/tftp

```
service tftp
{
    protocol = udp
    port = 69
    socket_type = dgram
    wait = yes
    user = nobody
    server = /usr/sbin/in.tftpd
    server_args = /var/lib/tftpboot
    disable = no
}
```

- service xinetd restart

2. Install DHCP:

- apt-get -y install dhcp3-server

- vi /etc/dhcp/dhcpd.conf, append:

```
ddns-update-style none;
option domain-name "home.local";
option domain-name-servers 192.168.200.1;
default-lease-time 86400;
max-lease-time 604800;
```

```
option time-offset -18000;
authoritative;
log-facility local7;
allow booting;
allow bootp;
subnet 192.168.200.0 netmask 255.255.255.0 {
        get-lease-hostnames off;
        use-host-decl-names on;
        range 192.168.200.100 192.168.200.200;
        option routers 192.168.200.1;
        option subnet-mask 255.255.255.0;
        option broadcast-address 192.168.200.255;
        filename "pxelinux.0";
        next-server 192.168.200.1;
}
```

- service isc-dhcp-server restart

3. Prepare /var/lib/tftpboot/initrd.aoecdrom.gz and /var/lib/tftpboot/vmlinuz, provided by David

4. Copy pxelinux.0 and vesamenu.c32 to /var/lib/tftpboot/ and /var/lib/tftpboot/pxelinux.cfg, from CD path: /install/netboot/ubuntu-installer/amd64/...

5. Prepare different boot file under /var/lib/tftpboot/pxelinux.cfg, named with MAC addresses of VMs, below is a sample:

```
 prompt 0
 timeout 20
 default pxelinux.cfg/vesamenu.c32

 LABEL Install Ubuntu 12.04 (Ubuntu 12.04)
     menu label Ubuntu 12.04 (Ubuntu 12.04)
     kernel bigtera/vmlinuz
     append initrd=bigtera/initrd.aoecdrom.gz root=
/dev/ram0 locale=en_US console-setup/ask_detect=fal
se vga=788 keyboard-configuration/layoutcode=us hos
tname=node-auto-1 file=tftp://172.16.146.234/ubuntu
-ezs3-auto1.seed aoecdrom=e1.0 quiet --

 LABEL Boot From Local Disk
   localboot 0
   TEXT HELP
   Boot to the local hard disk
   ENDTEXT
```

6. Modify seed file in CD and locate it under /var/lib/tftpboot/, the seed file preseeds questions during installation and presets the IP address. For the sample, please directly refer to the files on PXE server.

Reference

1. Robot Framework User Guide

2. Robot BuiltIn Library
3. Building a Fully Automated Ubuntu Installation Process
4. PXE and Preseed
5. Official Guide for Preseeding