


CITRIX®

Test-as-a-Service and XenRT

Alex Brett (alex.brett@citrix.com)

Xen Project Developer Summit

October 24th, 2013

Agenda

- What is Test-as-a-Service?
- Introduction to XenRT
- Demo
- Test-as-a-Service and the Xen Project
- Q & A

What is Test-as-a-Service?

An automated test platform, with the following characteristics:

- Self service
- Contains a good sized library of tests
- Performs on-demand provisioning
- For Xen, provides a variety of hardware

Not the ‘Citrix Auto Support’ tool, which also goes by the acronym TaaS, but is *Tools*-as-a-Service

CITRIX®

Introduction to XenRT (Xen Regression Test)

- Automated test platform used for all XenServer testing
- Python based
- Originally created in 2005 by James Bulpin @ XenSource
- Celebrated execution of its 500,000th test job in August 2013
- Open sourced in September 2013

XenRT Capabilities

- Job scheduling and results management
- Full host provisioning support
- Integration with storage and network devices
- Performs Functional, Scalability, Stress, Soak and Performance testing
- Supports large variety of guest operating systems (currently 31 Linux and 26 Windows)
- Many workloads available (e.g. BurnInTest, NetPerf, Lmbench)

XenRT Architecture

Submit Job

Scheduler

Check for jobs

Site 1

Controller

Execute Job

Test Hosts

Storage

Networking

Select Machines & controller

Machines & Jobs

Site 2

Controller

Test Hosts

Storage

Networking

CITRIX

XenRT Object Model

- All real world objects (hosts, guests etc) modelled
- Provides standard interface and methods for testcases, independent of toolstack in use
- Objects provide ‘check’ methods – ensure actual state is as expected
- Inheritance used to group common host / guest methods

Object model example

libvirt

```
 def createNetwork(self, name="XenRT bridge"):
        self.execvirt("virsh iface-bridge %s %s --no-stp 10" %
(self.getDefaultInterface(), name))
```

XenServer / xapi

```
 def createNetwork(self, name="XenRT bridge"):
        cli = self.getCLIInstance()
        args = []
        args.append("name-label=\"%s\"" % (name))
        args.append("name-description=\"Created by XenRT\"")
        nwuuid = cli.execute("network-create", string.join(args)).strip()

        return nwuuid
```

Some definitions

Test case: Python class derived from `xenrt.TestCase`, implementing (at minimum) a `run` method

Sequence file: XML file grouping test cases into serial / parallel blocks, together with a `<prepare>` section defining required environment for tests (number of hosts, guests etc)

Job: Execution of a sequence file against a specified input (build)

Sample Test Case

```
class TCHfx745(xenrt.TestCase):
    """Test to check that packets larger than 64 KiB sent from a guest
       to Dom0 are dropped at netback and the guest does not lose
       connectivity"""
    DISTRO = "centos57"
    def prepare(self, arglist=None):

        self.host = self.getDefaultHost()
        self.guest = self.host.createBasicGuest(self.DISTRO)

        self.host.installIperf()
        self.guest.installIperf()

        self.guest.checkReachable()
```

CITRIX

Sample XenRT Test Case

```
def run(self, arglist=None):

        step("Set the Guest MTU to 100")
        self.guest.execguest("ifconfig eth0 mtu 100 up")

        step("Start iperf server on host")
        self.host.execdom0("service iptables stop")
        self.host.execdom0("./iperf/iperf -s -w 256K > /dev/null 2>&1 < /dev/null &")

        step("Send iperf packets of size 65 KiB from guest to host")
        self.guest.execguest("iperf -c %s -w 256K" % self.host.getIP())

        step("Verify if guest is reachable")
        self.guest.checkReachable()
        log("The guest did not lose connectivity")
```

CITRIX®

Demo: clearwaterHfx745.seq

```
<xenrt>
  <variables>
    <PRODUCT_VERSION>Clearwater</PRODUCT_VERSION>
  </variables>

  <prepare>
    <host />
  </prepare>

  <testsequence>
    <testcase id="testcases.xenserver.tc.network.TCHfx745" tc="TC-19178" />
  </testsequence>
</xenrt>
```

CITRIX

Test-as-a-Service and the Xen Project

CITRIX®

Goal

The Advisory Board is willing to fund a Test-as-a-service infrastructure for the Xen Community to:

> “Increase upstream Xen Hypervisor quality including quality of latest CPU/platform features. Address problems with the code in a timely and proactive manner, including defects, security vulnerabilities and performance problems”
>
> *Xen Project Advisory Board August 2013 Meeting Minutes*

CITRIX®

Challenges

Security

- Prevent abuse of resources
- Prevent exploitation of (often unpatched) guests

Resources

- Hardware costs
- Hosting costs (space, power, connectivity etc)

Licensing

- Windows
- Commercial benchmarks and test tools

Management and maintenance

- Hardware issues
- Configuration of new equipment
- Maintenance of TaaS software

Proof of Concept

Is XenRT the right choice?

- Working with Oregon State University's Open Source Lab (OSUOSL)
- Three host environment
- Access via SSH gateway
- Limited set of tests
- TODO: Implement support for xl toolstack

Next Steps

- Complete Proof of Concept (PoC) implementation
- First meeting of Test Framework Working Group
- Encourage Xen Developer Community to evaluate the PoC

Assuming community buy-in for XenRT:

- Iterate and address any remaining challenges
- Publish code and live environment

Q&A

CITRIX®

Links

- XenRT documentation (work in progress!): http://wiki.xenproject.org/wiki/Category:XenRT
- Details of the PoC will go out on xen-devel
- xenrt-users mailing list: https://lists.xenserver.org/sympa/info/xenrt-users
- My email address: alex.brett@citrix.com

CITRIX®


CITRIX®