Compare performance of virtual agents with real ones

Run Rational Performance Tester agents on a VM to compare the agent performance levels

Andrew Citron
Gaurav Bhagat
Marshall Massengill

August 17, 2012

In this article, the authors compare running Rational® Performance Tester (RPT) agents on a virtual machine with running RPT agents on a "real" machine. The initial results show that running on a virtual machine has more variability in reported response times. Applying network tuning to the virtualized machines reduces the variability and make using virtualized RPT agents acceptable as long as a VM migration does not occur during a performance run.

A key foundation of cloud computing is virtualization. One question cloud-oriented designers, developers, and administrators need to ask themselves is "how does the performance level of virtualized components stack up against their 'real' physical counterparts?" "And if there is a negative gap, how can I overcome it?"

This article describes the results of a series of experiments on IBM® Rational® Performance Tester agents running on virtual machines (VMs) and offer some basic insights into the virtual vs. "real" argument.

Behind the experiments

We explain a series of experiments in which Rational Performance Tester agents run on virtual machines and compare those results to running the same tests on non-virtualized hardware. Before examining the virtualized environment we used, let's start with a background on the terminology we used, the methodology we used, and a results summary.

Terminology

Quite simply, you'll be able to follow the experiments by understanding these terms:

**Rational Performance Tester agent**
Software that simulates thousands of users making requests to a server.
Vuser
Virtual user. Each virtual user simulates many users over the course of the test. A running vuser starts by executing the script with a unique user identity. After the script completes, the vuser selects another unique user identity and executes the script again as that new, unique user identity. The vuser repeats this process of selecting a new user identity and executing the script as long as the vuser is running.

Real agent
RPT agent running on non-virtualized hardware.

VM
Virtual machine. A virtual agent is an RPT agent running on a VM.

vMotion
When the VM manager moves the VM image from one physical machine to another.

Plateau
Represents a period of time in which the same number of vusers are active during a test.

TPS
Transactions per second. TPS and PPS (pages per second) are used interchangeably as a measure of throughput.

Paravirtualization
A virtualization technique that presents a software interface to virtual machines that is similar but not identical to that of the underlying hardware.

Methodology

The same performance test was run a number of times using real RPT agents and virtualized RPT agents.

The performance benchmark involved a number of different HTTP requests performed by different simulated users. The benchmark and the system under test are not the focus of this article. The focus of this article is the RPT agents that generated the simulated load.

Each test consisted of three 5-minute runs simulating 2,000 vusers and three 5-minute runs simulating 2,500 vusers. Each test was run two times using the real agents and two times using the virtual agents.

In each test there were five machines running RPT agents. One of those machines was a real machine and ran just 100 vusers. That machine was used as a point of comparison; it never ran enough of a load to cause contention for its resources. However it did run enough vusers to generate a usable sample size. This allowed us to compare a lightly loaded real machine with heavily loaded VMs and real machines.

Analysis was performed on the 5-minute runs. The three 5-minute runs were also combined with one 15-minute plateau. Longer runs provide more data points for each interval. More data points mean more stability of averages.

We also did an experiment in which the virtual agent was moved from one physical machine to another during a load test. We call that process vMotion.
A summary of the overall results

In the initial set of tests, response times and throughput reported by the virtual agents varied more than the response times reported by the real agents. For benchmarks, where there are response time criteria, this variation causes a problem. Subsequent analysis and experimentation led to a VM network-tuning setting that reduced the response-time variation. With that tuning in place, the VMs behaved acceptably as RPT agents.

The experiment where the virtual agent was migrated (vMotion) showed a spike in reported response time while the VM was in the process of moving. The conclusion here is that VM migration must be an uncommon occurrence; the analyst must know when it has occurred because it might impact the test result.

Now let's look at the virtualized environment.

Description of virtualized environment

In the IT field, there is always more than one way to solve a given problem. This is particularly true in the area of designing a good virtualization infrastructure where the possibilities are practically limitless. It is important to understand that the infrastructure design described here is not a "recommended best practice" or a "model environment," but rather an attempt to create a solution that works for the lab environment that it provides hosting for.

There are three major parts to any virtualization infrastructure — servers, storage, and the network. The environment used for these tests:

- Servers. The servers consisted of IBM System x3850 X5s. These servers are each loaded with four Intel Xeon X7560 CPUs and 512GB of RAM. The servers are grouped both physically and logically into clusters of eight machines. This clustering of machines is described in depth later.
- Storage infrastructure. The storage for these servers consists of redundant fibre channel SAN fabrics. The fabrics are made redundant through dual-port host bus adapters on the servers and redundant top-of-rack SAN switches. These switches are then connected into a larger fabric consisting of multiple uplinks to multiple IBM DS5300 Storage attached networks (SAN). Each set of eight servers is placed into a hostgroup that can only reach certain Logical Unit Numbers (LUNs) on the SANs. Each host within a hostgroup can see only the storage made available to that hostgroup.
- Network infrastructure. The network for these servers consists of one 100MB management link and four 1GB links, for a total of five Ethernet connections to each server.
  - The 100MB management link is used for communication with the onboard Integrated Management Module which enables remote console access to the server as well as the ability to monitor the status of the hardware.
  - The four 1GB links are split into two groups of two connections each:
    - The first group is used for data traffic for the virtual machines.
    - The second group is used for management traffic to the host server.

The management traffic is kept isolated to a private internal network while the data traffic is allowed onto a larger public network. This was done to reduce the amount of traffic on the data
network, as well as to ensure the security of the management traffic. Each group of links is further split into an active and a standby link. This combination of active and standby links ensures redundancy.

To enable a multitude of private and public networks to be connected, VLAN tagging is utilized. Each packet leaving from any given virtual machine is tagged with the appropriate VLAN number. This enables access to a large number of subnets across the environment. It also enables virtual machines to rove from one cluster of machines to another cluster through a process known as migration.

Figure 1 diagrams the virtualized environment.

**Figure 1. The virtualized environment**

The real agents ran on Intel® Xeon® machines that had two 2.8GHz processors and 2GB of RAM. Those machines had a 1GB Ethernet cards.

The virtual agents ran RedHat Linux®. The real agents ran Windows® Server 2003. It would have been better if the virtual and real agents ran on the same hardware and software, but that wasn't an option. We do have other evidence that indicates that real agents running on Linux show less than a 1 percent variation for throughput between runs. That is similar to what we saw with the real Windows agents.

The machine we used as the control real machine was an Intel Xeon, running Windows Server 2003. It had two 3.2GHz processors. Hyperthreading was enabled. There was 3.5GB of RAM. It had a 100MB Ethernet card. It ran only 100 vusers. It didn't need to be as powerful as the other agents.
Illustrating the results

Now we'll show you some graphs that illustrate the numbers behind the conclusions we reached; we'll show you throughput and response times before and after network tuning was applied.

Throughput and response times for the untuned network

The first graph (Figure 2) shows the throughput and response times reported by the virtualized agents before network tuning was applied. This can be compared to the second graph (Figure 3) which shows the same information for real agents. These graphs are from the 2,500 vusers workload plateaus.

Figure 2. Virtual agents, untuned network, 2,500 vusers

Figure 3. Real agents, untuned network, 2,500 vusers

The graphs illustrate that the virtualized agents' reports of response times and throughput varied more than the real agents:

- The response times reported by the virtual agents varied from 188.6ms to 104.5ms; that's a 48.1 millisecond range.
- The response times reported by the real agents vary from 148.4ms to 120.6ms; that's a 27.8 millisecond range.

The page-hit rate reported by the VM agents ranged from 215.9 PPS to 219 PPS. That's a 3.1 PPS difference for the virtualized agents. The real agents reported a range of 218.5 PPS to 219.3 PPS. That's a difference of 0.8 PPS. Again, this illustrates that the virtual agents showed a higher variance than real agents in the data they reported.
After tweaking the network adapter on the VMs

After the first set of tests showed high variability on the response times reported by the virtual agents, we changed the type of virtual network adapter that the machine was using.

You can think of it as switching out the Ethernet adapter of a physical system.

Originally, the virtual machines were configured to use a "flexible" adapter which is basically VMware's way of allowing the adapter to be chosen based on the needs of the system. These adapters work great in theory and not-so-well in practice. VMware has dropped them in favor of something called VMXNet3 or VMXNet2 adapters. There is also the option to use an E1000 virtual network adapter which is basically an emulated E1000 Intel Network Card.

We switched out the flexible adapter in favor of the VMXNet3 adapter. This provided a performance boost to all network traffic for the system. The VMXNet3 adapter is technically a paravirtual network adapter. Paravirtual hardware works on the idea of offloading tasks from the host CPU to the physical resources of the system. In the case of these paravirtual Ethernet adapters, the workload is being handled by the actual Intel and Broadcom physical Ethernet adapters on the hosts rather than the hosts CPU. This leads to improved response times and throughput.

Along with making this change to the hardware, the change was also made to the guest operating system and the driver it uses to communicate to the adapter. This is similar to installing new drivers on a physical machine after installing a new network adapter. The same has to be done on a virtual machine.

Figure 4 shows that after applying the VMXNet3 adapter, response times were lower than before. In fact results with VMXNet3 adapter were lower than all the other tests. The reported PPS was also slightly higher.

**Figure 4. Comparing real agents, VMs with/without the VMXNet3 adapter**

![Graph showing comparison of real agents vs. VMs with/without VMXNet3 adapter](image)
Not shown in this chart is that the variation of reported response times was also less with the VMXNet3 adapter. This is important for being able to reproduce results from run to run.

The conclusion is that with the right network settings, a virtualized agent is as reliable as a real agent for reporting response times and throughput.

**Clusters and vMotion**

vMotion is VMware technology that enables virtual machines to be migrated live from server to server. This is typically done for load balancing reasons. In our experiment we forced a migration of one of the VMs during one of the 5-minute plateaus. We did that to determine the effect that would have on the data reported by the virtualized agents.

Each cluster of servers shares some characteristics. Each grouping of eight servers shares the same hardware type. The CPUs, memory, disk, and adapters are all the same in each machine within a cluster. This commonality is key for the process of vMotioning to work properly.

Clusters of machines also share configuration characteristics, such as high availability and dynamic resource scheduling:

- **High availability** is the process by which virtual machines are kept running to minimize downtime in the event of hardware failure. If one server within a cluster experiences a failure, the virtual machines from that server are immediately brought back online on a different server within that cluster. This minimizes any downtime as a result of a service disruption.
- **Dynamic resource scheduling** is the process by which virtual machines are dynamically moved between servers within a cluster based on their resource needs and the availability of resources for them to use. If a virtual machine is in need of CPU resources on a host that has very little resources available, that virtual machine will be migrated off to a different host where there are CPU resources available. Alternatively, virtual machines that have less need of resources are often migrated off to make room for virtual machines that need to consume more resources.

The process of vMotioning works by migrating a virtual machine in a current running state from one host to another within a cluster. The virtual machine is kept running while this process is happening and as a result, the virtual machines experiences virtually no disruption.

In our testing, we have observed that the end users experience at worst a short hangup, no different than one or two packets being lost by network transmission failures. This is an acceptable interruption for carrying out integration, usability, functionality, and other types of testing, but not for performance testing where a single dropped packet can distort the outcome. Frequent vMotion can also be a problem in a production environment that has strict service level agreements.

Dynamic resource scheduling and high availability can both use the vMotion process to migrate virtual machines running on one host to another to provide failure protection or to meet the immediate resource needs of a virtual machine. Figure 5 shows our virtualized environment with vMotion migration in process.
Under normal circumstances vMotion does not occur frequently. We felt a realistic experiment was to migrate one of four VMs we used during the test. We also had a fifth real agent that was running 100 vusers. We used that agent as a control to get some idea of response times reported by an agent that should be unaffected by the vMotion.

The effect of the VM migration was apparent. The effect lasted about a minute. The following few graphs illustrate this.

Figure 6 shows the report from one VM agent during the migration. This was taken at the 2,000 vusers plateau.
Compare performance of virtual agents with real ones

Figure 6. Page response time during migration; spike is bad

The spike in the response time where the vMotion occurred is apparent. This sort of spike, caused by the virtualization infrastructure, during a performance test run is unacceptable.

Next, look at vMotion when analyzed as 15-minute plateaus (Figure 7).

Figure 7. A longer plateau makes for more stable average

The longer plateau gives a more stable average and minimizes the effect of the VM migration. However you can still see that the vMotion caused higher response times to be reported at the virtualized agents. This graph reports overall average response time compared to the response time at the one real agent running 100 vusers.

The Figure 7 graph shows that the one VM migrating had an effect on the reported average response times. Even though three of the four VMs did not migrate, the higher response time reported by the migrating VM caused the average response time to be higher than that reported by the control group's real agent.
What can't be seen in Figure 7 is that the impact of the migrating the VM did have an impact on response times reported by all agents. Presumably that was caused by a bottleneck from the server-under-test not being able to complete the requests that were initiated by the migrating VM until that VM finished migrating.

In general, it is best if vMotion can be disabled on the virtualized agents. When running a performance test, any uncontrolled source of variance is a problem.

If you can't disable it, it is important to be able to determine if vMotion has occurred during a test. The virtualization infrastructure does provide logs that can be examined to determine that a migration has occurred. The screen capture (Figure 8) is from the view inside the virtual infrastructure client software of the Tasks and Events for one of the VMs that the migration process was performed on. The logs clearly indicate that a migration task was performed along with who initiated the task and when the task was completed.

Figure 8. Logs as a source of determining whether vMotion takes place (if you can't disable it)

(See a larger version of Figure 8.)

If the system had been dynamically moving the virtual machines around based on the load, then the task would have been initiated by the System rather than by a user (in this case, VISRTP/marshall). This log information can also be gathered or checked programmatically with VMware's APIs and scripting utilities.

It's important to note that dynamic resource scheduling can be disabled for certain virtual machines in a given environment. This does increase the risk of a virtual machine's uptime being impacted based upon a host outage but for a performance team, maintaining constant performance is more important than the possibility of an outage.
In conclusion

Virtualizing Rational Performance Tester agents is a viable configuration; however, it is important to make sure that the virtualized environment is properly tuned. The network card setup was critical in our environment.

For a comparison evaluation, it is best to have non-virtualized agents that are running equivalent hardware and operating system as the virtualized agents. For problem determination though, it is a good practice to have a set of real agents, preferably running a different operating system than the virtualized agents. We have found the real agents to be a useful control group when problems or bottlenecks occurred. By running a test with the real agents, we were able to rule out that the VM was a cause of problems.

If a VM migration has occurred during a test it is important for the analyst to be aware of it. Automated monitoring of the VMs is preferable. The best alternative is to disable vMotion on the servers that are hosting the RPT agents.
Related topics

• For more resources on the topics in this article:
  • Learn more about vMotion.
  • Learn more about Rational Performance Tester.
  • Learn more about Logical Unit Numbers.
• In the developerWorks cloud developer resources, discover and share knowledge and experience of application and services developers building their projects for cloud deployment.
• Download a trial version of Rational Performance Tester.

© Copyright IBM Corporation 2012
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)