



Red Hat Performance Briefs

RHEV 3.1 + RHS 2.0 perf.

For a single RHEV host

Red Hat performance engineering
January 2013

Version 0.9
Jan 2013





1801 Varsity Drive™
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo and Xeon are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2013 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The information contained herein is subject to change without notice. Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc.

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Send feedback to refarch-feedback@redhat.com ***** Modify the subject and text of this link *****



Table of Contents

1 Goals.....	3
2 Results.....	4
2.1 Future performance enhancements.....	7
2.2 Evolution of RHEV/RHS virtualization.....	9
3 Workload.....	10
4 Configuration.....	10



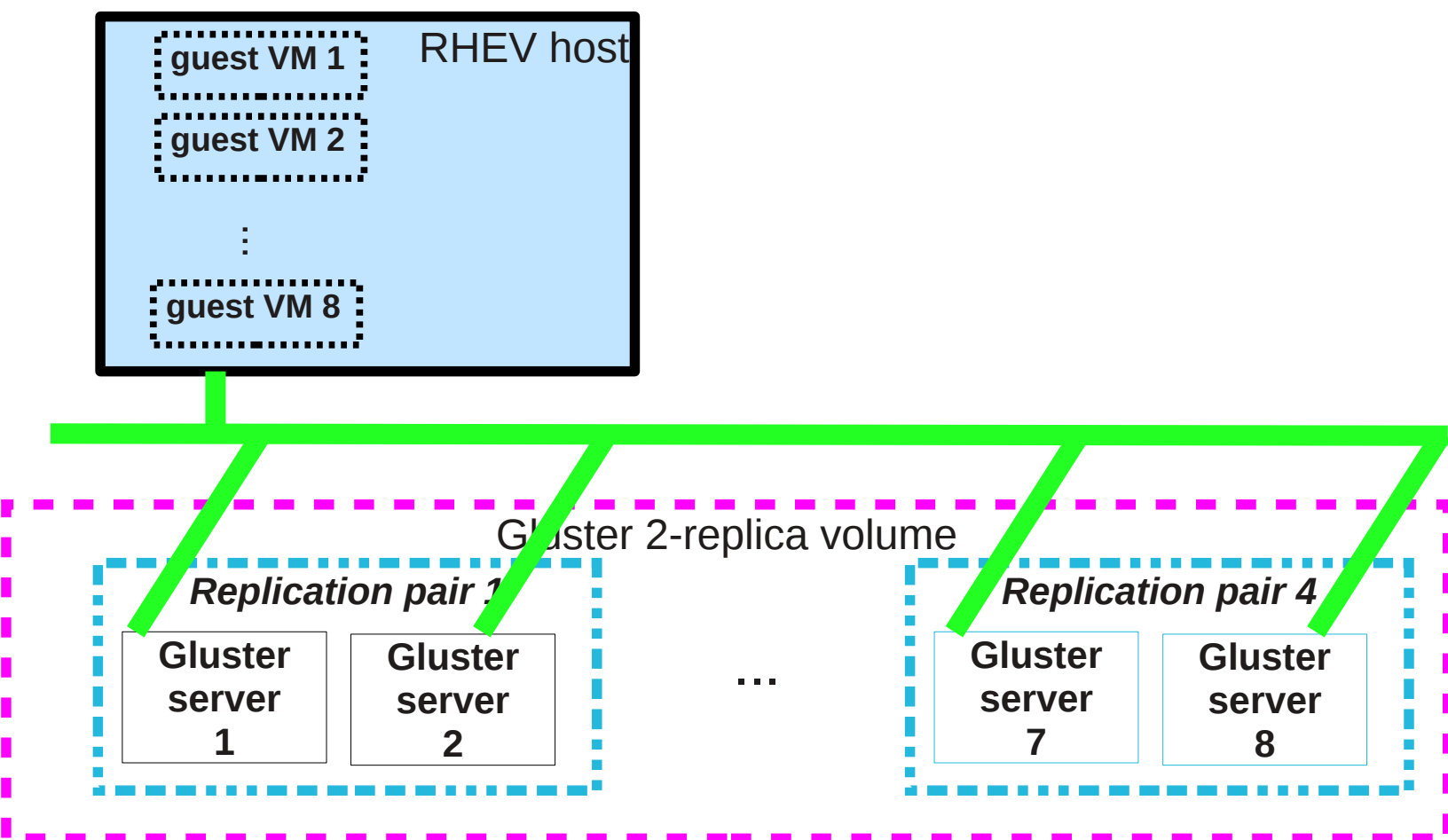
1 Goals

This performance study is one of a series intended to obtain performance data which can be used for several purposes:

- determine scalability limits for RHEV/RHS
- provide recommendations to customers about ratio of RHEV hosts to Gluster servers, number of guests per RHEV host, tuning of RHEV hosts if any, tuning of guests if any
- understand I/O bottlenecks and identify optimizations

In this article we use the terms “Red Hat Storage” (abbreviated to RHS) and “Gluster” somewhat interchangeably. Red Hat Storage is a particular instantiation of Gluster that is tuned and optimized for specific workloads and configurations.

Drawing 1 : RHEV/RHS configuration





2 Results

For first installment of results, we use a single RHEV client with virtual machine parameters similar to RHEV/RHS configuration. *We are using an “image-local” filesystem only. By “image-local” filesystem we mean a filesystem such as ext4 which is backed by the VM's virtual disk image.* In these tests, we created a separate LVM volume inside the VM's virtual disk image, formatted it with ext4, and ran all tests within that filesystem¹ We are using beta-test version of RHS 2.0+ designed to support RHEV 3.1 release, and we are focusing at this point exclusively on sequential read/write performance of an application running inside a VM. The short version of the results is:

max throughput for:	sequential read MB/s	sequential write MB/s
single guest	380 ²	404
host (aggregation of all 8 guests)	926	560

Note: it is possible for the guest to mount an external filesystem using the network (examples: Native Client or NFS (Linux), SMB (Windows), or to attach an external block device over the network (example: iSCSI). If additional performance is needed beyond what is described in this report, consider using one of these alternatives.

For sequential write performance, RHEV/RHS performance is about 33% less than network limit for a single guest, and approaches the network limit after that. The Y-axis is sized so that the top of the Y axis corresponds to network line speed. Keep in mind that 2 replicas/file are used in this test, so network throughput is actually 2x the above results. 550 MB/s x 2 replicas = 1100 MB/s ~ 10 Gbit/sec . , and in the case of 2 guests/host with server O_DIRECT off, we are over 80% of 10-GbE line speed.

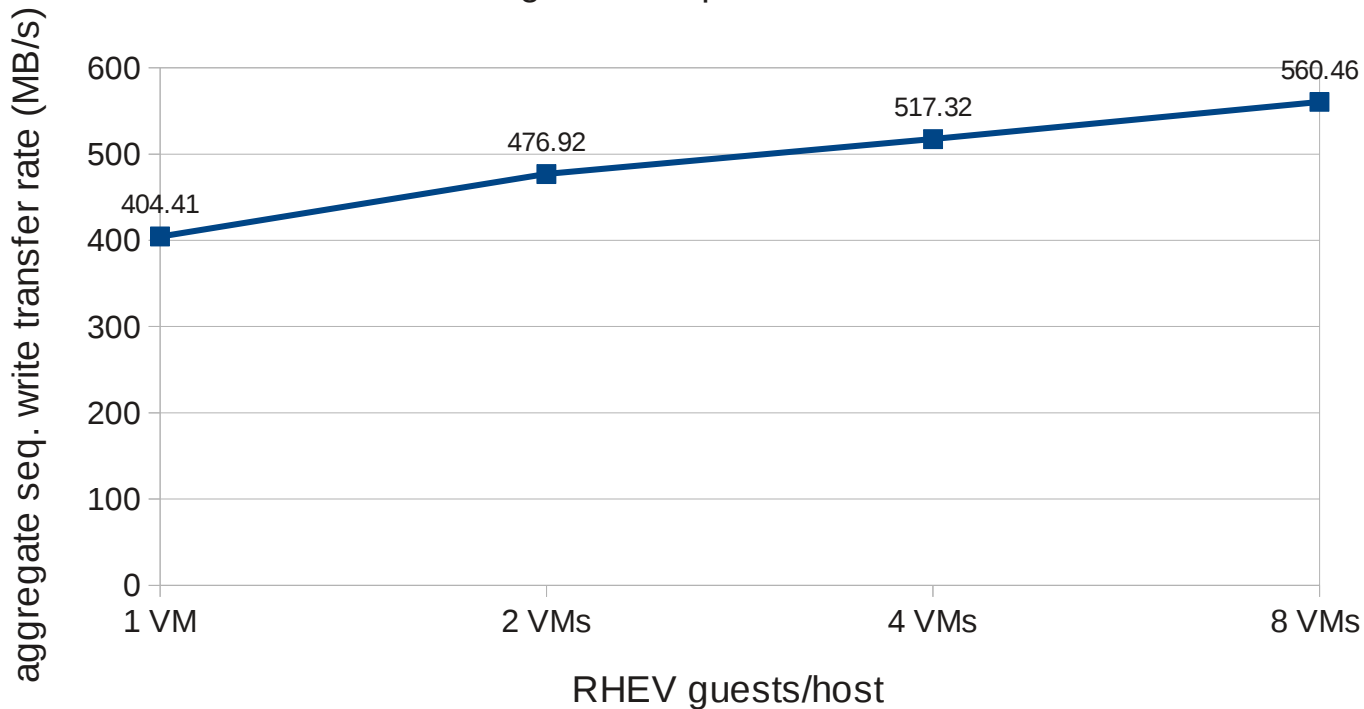
¹ It is also possible to create a separate virtual disk image, and in fact this may be a best practice for certain applications, but this did not seem to be a configuration alternative that would yield significantly more data.

² This result requires guest readahead tuning of 2 MB on block device. Using Linux kernel default readahead of 128 KB in guest, you only get 180 MB/s. See configuration section below for instructions on how to get **rhs-guest** tuned profile.



RHEV guest sequential write throughput

1 RHEV host, 1 iotest thread/guest, 2-replica volume, 16-GB file, 4-KB write size

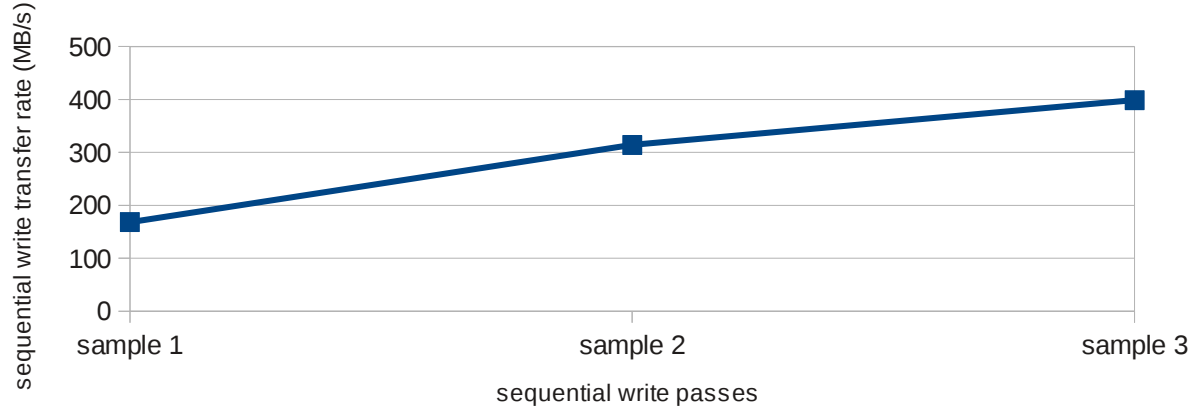


Note that the first time the VM is used, there is some overhead because of the copy-on-write (cow) behavior of the VM image for the first sequential write, and this overhead then diminishes.



effect of thin provisioning on sequential write perf.

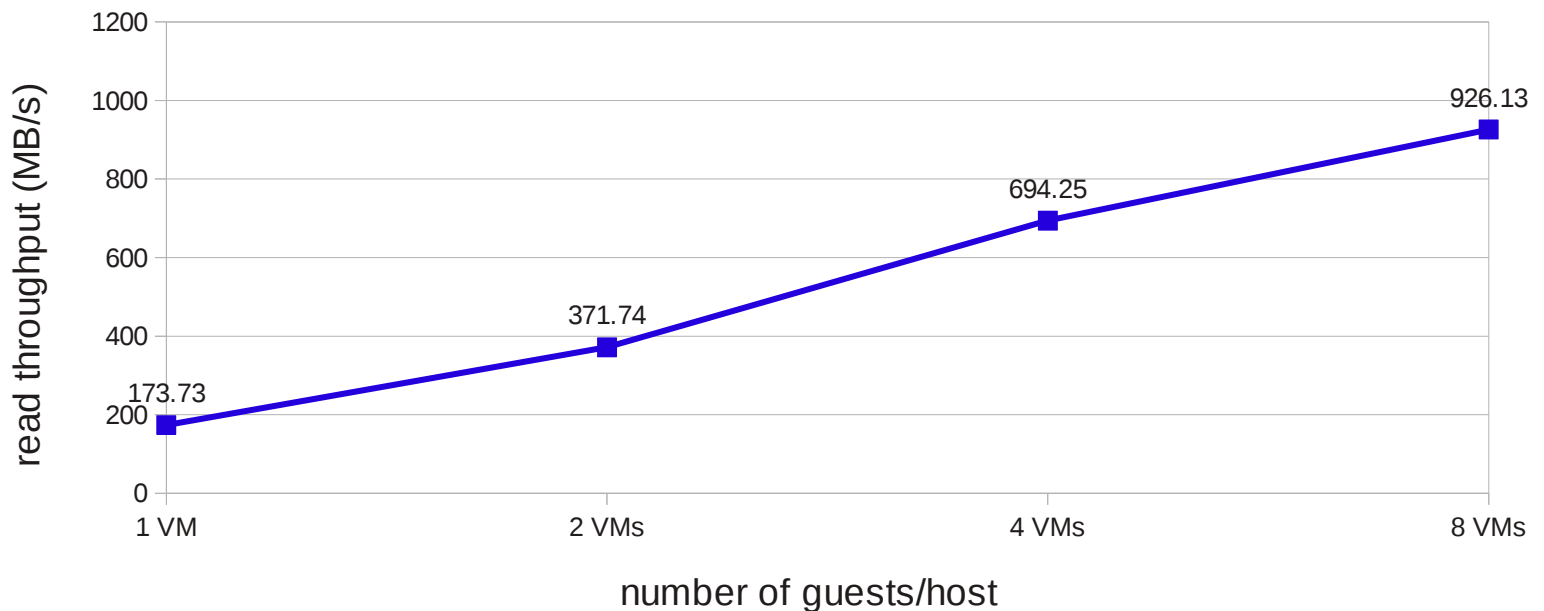
successive sequential writes in same VM to same file, first sample is with previously unused VM



For reads, only 1 copy of data will be transmitted to the Gluster client so the top of the Y-axis again corresponds to network line speed. This result shows that we get about 15% of network line speed for a single guest, and about 80% of network line speed for 8 guests.

RHEV host sequential read throughput as guests/host increase

1 RHEV host, 1 iotune thread/guest VM, 8 Gluster servers, 2-replica volume



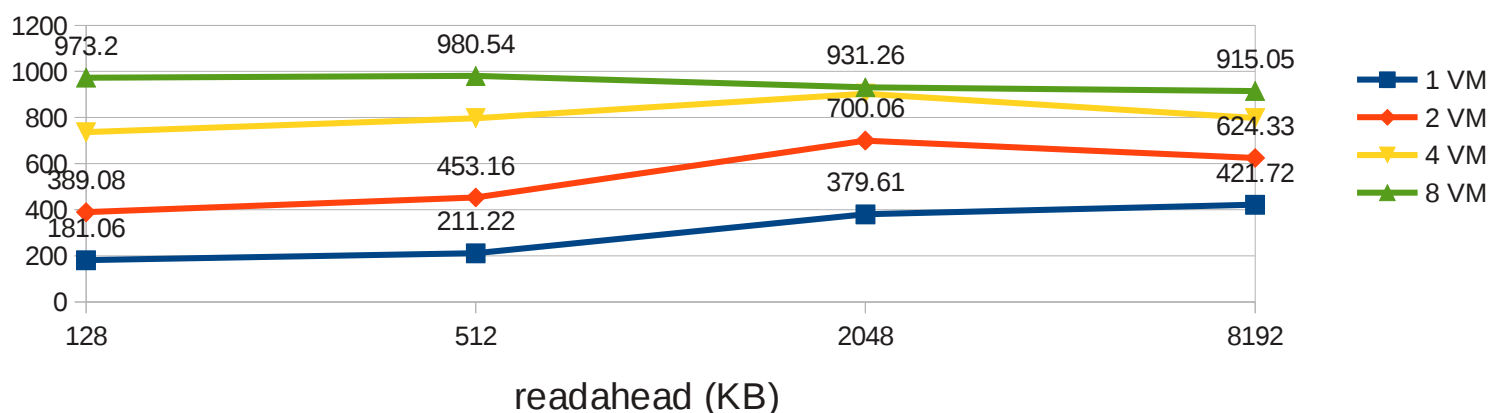


If the RHEV/RHS user has only 1 guest per host, and if sequential read performance for that guest is critical, the next graph shows an alternative that in some cases can give better results: guest readahead tuning. To do this, we set `/sys/block/$dev/queue/read_ahead_kb` sysfs parameter inside the guest where `$dev` is the appropriate block device. Note: for LVM devices, you must use device mapper block device to set readahead.

effect of guest readahead value on RHEV host read throughput

1 RHEV host, 1 iozone thread/guest, 16 GB data total, 4-KB iozone record size

RHEV host read transfer rate (MB/s)



This graph shows that 2048 KB is a “sweet spot” that gives 100% performance improvement in the 1-VM case but only 6% performance degradation in the 8 VM case.

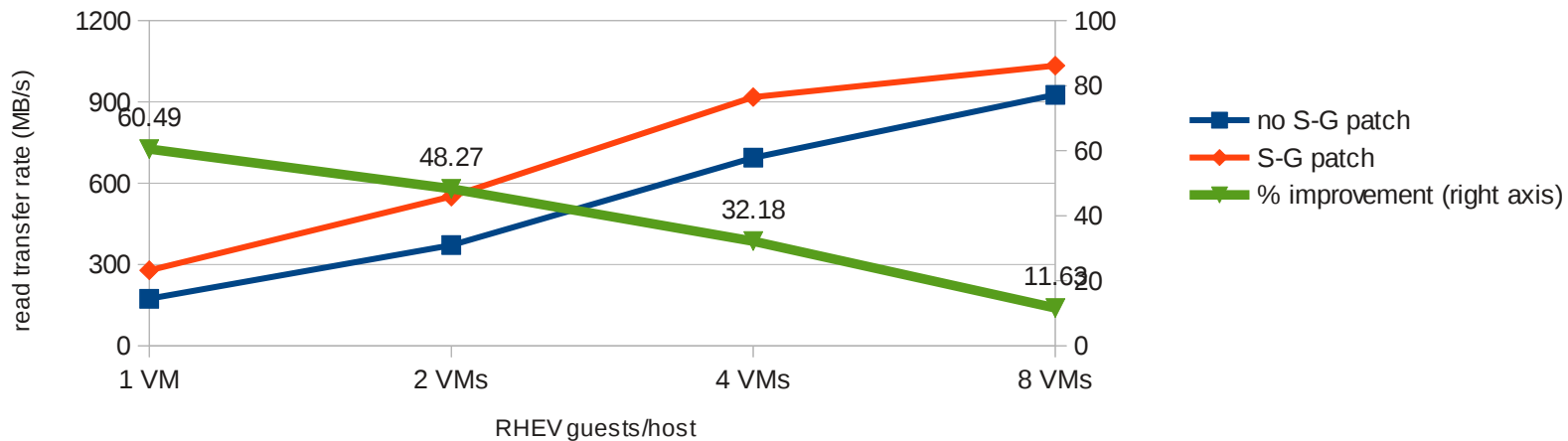


2.1 Future performance enhancements

The next graph shows what the FUSE scatter-gather patch (in RHEL6.4, details in bugzilla 858850) can do:

effect of RHEL6.4 FUSE Scatter-Gather enhancement for guest seq. read

1 RHEV host, 1 iozone thread per RHEV guest, 8 RHS servers, 2-replica volume

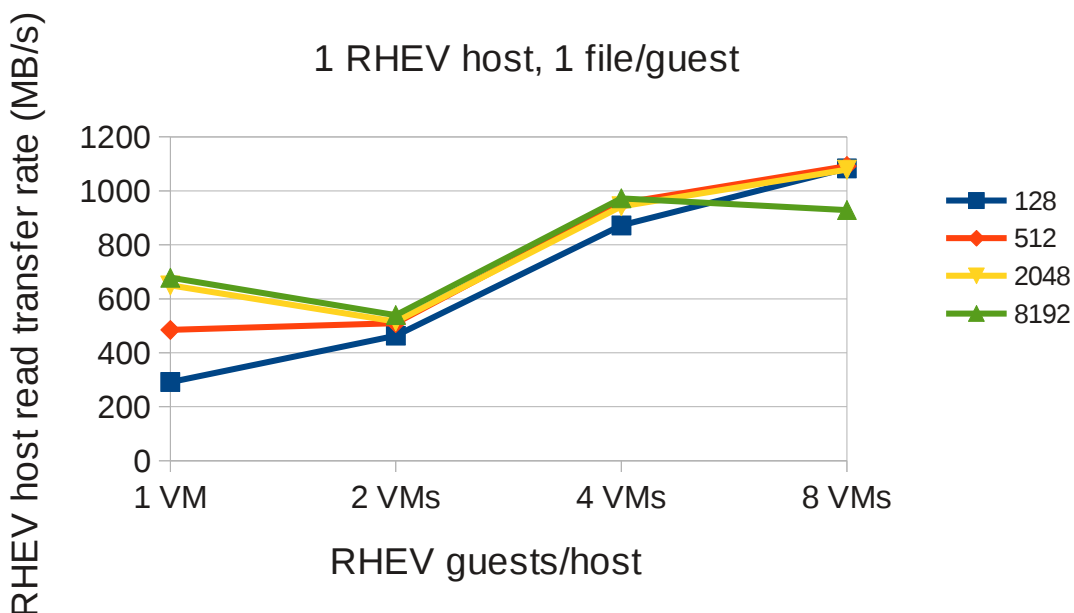


The preceding graph shows effect of two variables, whether scatter-gather patch is in use, and number of RHEV guests/host. This enhancement provides a significant boost to per-guest throughput, as shown by the green curve, until you reach 8 guests, at which point performance is constrained by other factors such as network capacity.

If you combine this enhancement with guest readahead tuning, you can significantly improve sequential read performance as shown by the below graph, to 50% of line speed.

Effect of guest readahead on aggregate guest sequential read performance with FUSE Scatter-Gather enhancement

1 RHEV host, 1 file/guest





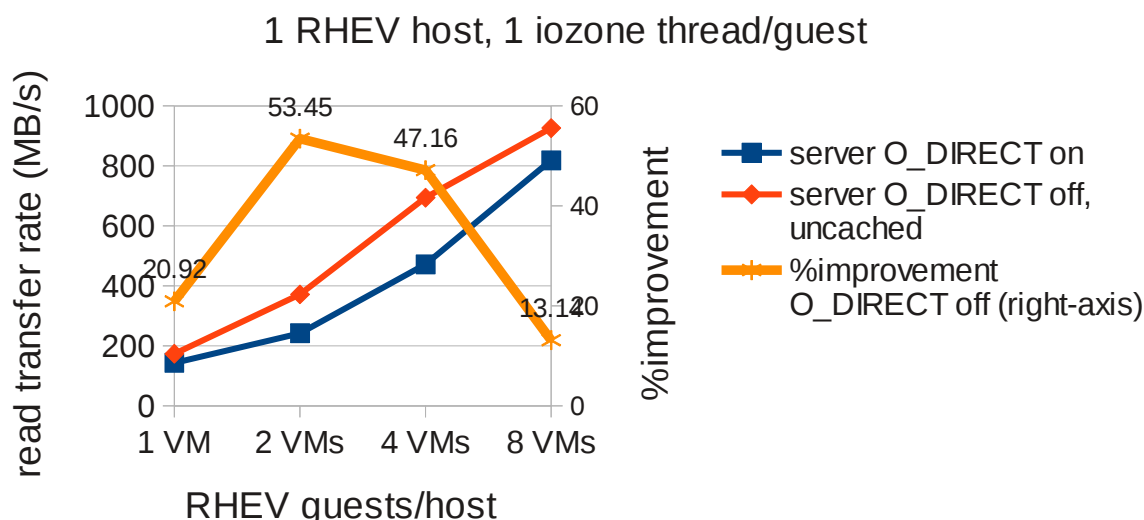
2.2 Evolution of RHEV/RHS virtualization

The next graph shows the effect of an enhancement to Gluster specifically for support of virtualization: the `network.remote-dio` volume parameter. When you configure the Gluster volume with the “group virt” command shown in the Configuration section, this `network.remote-dio` parameter will be set to “on” by default. This parameter is somewhat counter-intuitive – if you turn it “on”, you disable use of `O_DIRECT` flag on the Gluster server, and if you turn it “off”, you are causing the server to use `O_DIRECT` flag if and only if the client application opens the file with `O_DIRECT`.

RHEV default configuration for guests will assign the KVM parameter “cache=none” to the KVM guests that it manages, and this in turn causes the KVM qemu process to open the guest virtual disk image with the `O_DIRECT` flag. In RHS 2.0, this would result in the **server** using direct I/O to the XFS filesystem on the brick. But in the RHS 2.0+ release targeted at virtualization, the recommended volume tuning command in configuration section below will result in several virtualization-specific customizations to the Gluster volume, including setting the `network.remote-dio` parameter to on, which will disable `O_DIRECT` on the server, allowing server-side caching to take place.

Note that this test was run without the Scatter-gather patch discussed in the preceding graph.

effect of new `network.remote-dio` volume parameter



This parameter doesn't have as large an impact when using a single guest per host, but when



using more than one guest, the disabling of O_DIRECT flag on the server results in higher throughput, perhaps because of enabled Linux readahead functionality in the servers. When we get up to 8 VMs, we have reached line speed anyway so the effect cannot be as pronounced.

This optimization is critical for scaling out a RHEV guest pool to large numbers of VMs, because with KVM “cache=none” attribute of RHEV guests, the RHEV host will not cache data from the backing image for the RHEV guest template, so the Gluster servers with the template replicas will be forced to serve the template image to ALL of the guests over the network. Fortunately the guests do cache this data so over time the load on the Gluster servers decreases, and writes to the guest filesystem go to VM-specific qcow2 images and are hence distributed across the gluster volume automatically. A future paper will address this topic.

3 Workload

This kind of workload is at present modeled using iotest benchmark with the “-+m” option that enables distributed workload generation. We can use both random I/O and sequential I/O options in iotest, as well as direct I/O. For example, this command will initiate I/O using a separate thread on each of 2 guest VMs on the single RHEV host, writing 8 GB/thread in 4-KB transfer size, with no rewrite test:

```
iotest -+m 2vm.iotest -w -c -e -i 0 -+n -r 4k -s 8g -t 2
```

where 2vm.iotest file records such as:

```
gprfv-0003-10ge /mnt/test /usr/local/bin/iotest  
gprfv-0007-10ge /mnt/test /usr/local/bin/iotest
```

4 Configuration

The elements of the hardware configuration for these tests are:

- 8 RHS servers – each 48 GB RAM, 2 6-core Intel westmere CPUs,
 - PERC H700 (MegaRAID) controller with 12 drives in a RAID6 with 256-KB stripe size
 - BIOS – hyperthreading enabled, virtualization disabled, power set to “OS control”
- 1 RHEV host – 48 GB RAM, 2 6-core Intel westmere CPUs
 - BIOS – hyperthreading + virtualization enabled, power set to “OS control”
- 1 Intel 82599EB 2-port 10-GbE network card on all clients and servers.
- Cisco Nexus 7010 switch with 4 48-port line cards, 5-meter thru 10-meter twinax cables
 - jumbo frames (MTU=9000)

The elements of the software configuration are:

- on servers, RHS 2.0 base install, upgraded with Gluster virtualization support baselevel rhvirt1-9) (in one graph, the RHEL6.4 kernel S-G patch was used)
- kernel tuning done on each server with:



```
# tuned-adm profile rhs-high-throughput
```

- Gluster – 2-replica volume, volume tuning is:

```
# gluster volume set your-volume group virt
```

- on clients, RHEL6.3 + RHEV 3.1 host RPMs, rhsvirt1-9 glusterfs, and glusterfs-fuse RPMs,
- On RHEV-M, version 3.1.0-22
- Tune your guest for better readahead following these steps:
 - Get the “rhs-guest” tuned profile from /etc/tune-profiles on a RHEL host with glusterfs RPMs installed.
 - Copy this to /etc/tune-profiles/ directory on your virtual guest.
 - Configure the guest to continue to use this tuning profile with command:
 - `tuned-adm profile rhs-guest`
 - make a template out of it, and then create your pool from that template.