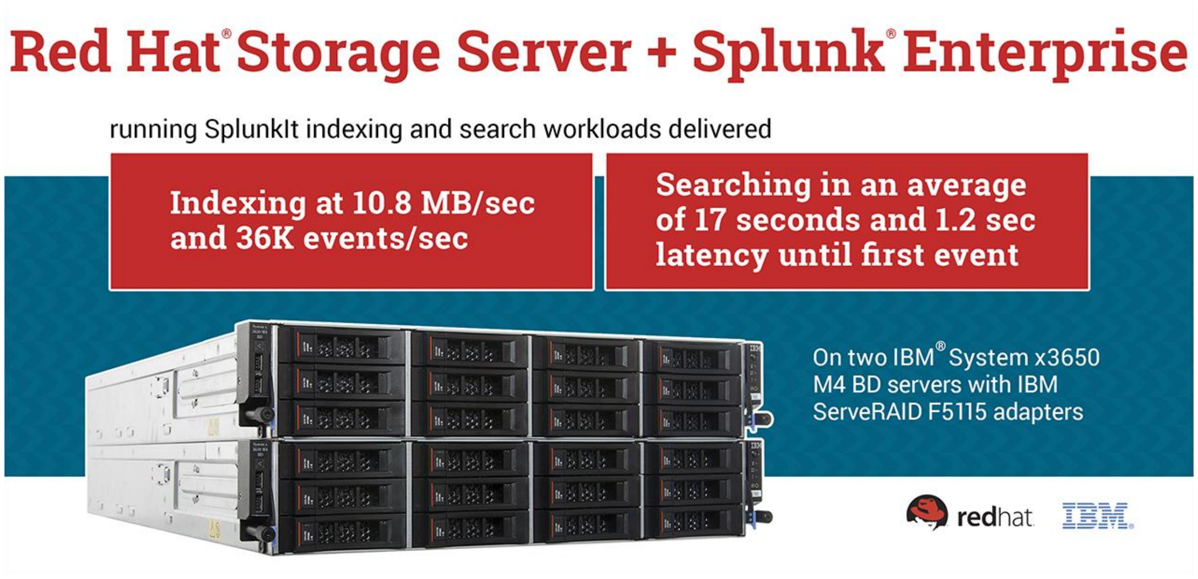
SPLUNK ENTERPRISE ON RED HAT STORAGE SERVER 2.1


Red Hat® Storage Server + Splunk® Enterprise
running SplunkIt indexing and search workloads delivered
Indexing at 10.8 MB/sec and 36K events/sec
Searching in an average of 17 seconds and 1.2 sec latency until first event
On two IBM® System x3650 M4 BD servers with IBM ServeRAID F5115 adapters
redhat
IBM

For any business, understanding and increasing operational efficiency is a major concern for management. In recent years, as connectivity and computing power have grown, the amount of machine-generated data companies must contend with has ballooned as well. Businesses are turning to robust software solutions to analyze this data, determine patterns, and make important decisions; these solutions require a storage platform that can handle enormous amounts of throughput and provide easy-to-scale capacity while remaining affordable. Red Hat Storage Server on IBM System x servers combine to provide a powerful yet cost-effective storage platform for business-analysis software such as Splunk Enterprise.

In the Principled Technologies labs, we configured two IBM System x3650 M4 BD servers with Red Hat Storage Server 2.1, operating as a storage back end to a Splunk Enterprise infrastructure. We then ran SplunkIt 2.0 indexing and search workloads, which target Splunk hot/warm data buckets requiring fast I/O capabilities. The Red Hat Storage solution delivered 11,059 average kilobytes per second throughput, while processing 36,000 average events per second during data ingest testing. Put differently, that is an ingest rate of 10.8 megabytes per second or 37.9 gigabytes per hour. Search performance across the ingested data delivered an average of 17 seconds response time to queries.

When we compared these SplunkIt results to published results of an eight-node EMC Isilon X400 storage solution, we found that Red Hat Storage achieved comparable performance in terms of both throughput and search time, running on just two IBM x-series servers, costing significantly less. It is important to note that we did not test the EMC Isilon solution in our labs and that the published EMC results do not contain specific test methodologies.[1]

[1] EMC Reference Architecture for Splunk https://community.emc.com/docs/DOC-27430

FAST ACCESS TO ANALYTICS WITH RED HAT STORAGE SERVER

Red Hat Storage Server is an open-source, software-defined storage platform for physical, virtual, and cloud environments. The Red Hat Storage Server configuration we tested consisted of two IBM x3650 M4 BD servers running Red Hat Storage 2.1 Update 2.

Splunk Enterprise helps businesses by capturing massive machine-generated data streams, indexing the information regardless of schema, and making it searchable for advanced real-time analytics. After Splunk indexes data, users can search for the information they want or need to analyze patterns and make decisions based on those findings. Splunk Enterprise software is self-contained and has no additional back-end database or specific infrastructure requirements. Our test workload, SplunkIt, consisted of an index test using a 50GB syslog data set and a search test that uses the resulting indexed data.

COST-EFFICIENT PERFORMANCE FROM RED HAT STORAGE SERVER

Index test

What this means to you…
Running Splunk with Red Hat Storage Server and IBM System x hardware can offer real-time data analysis for your business at significantly lower costs than on proprietary storage appliances..

The greater the performance capabilities of a given storage server, the faster Splunk Enterprise can complete data ingestion through indexing. The SplunkIt benchmark measures throughput in indexed events per second (EPS) and average disk reads from the storage in KBps. Figures 1 and 2 show the results of our index tests on the two-node Red Hat Storage solution. The Red Hat Storage solution, delivering 11,059.7 average KBps and 36,690.5 average events per second, was comparable to the published 10,944.1 average KBps and 38,730.8 average events per second of the EMC Isilon solution.[2]

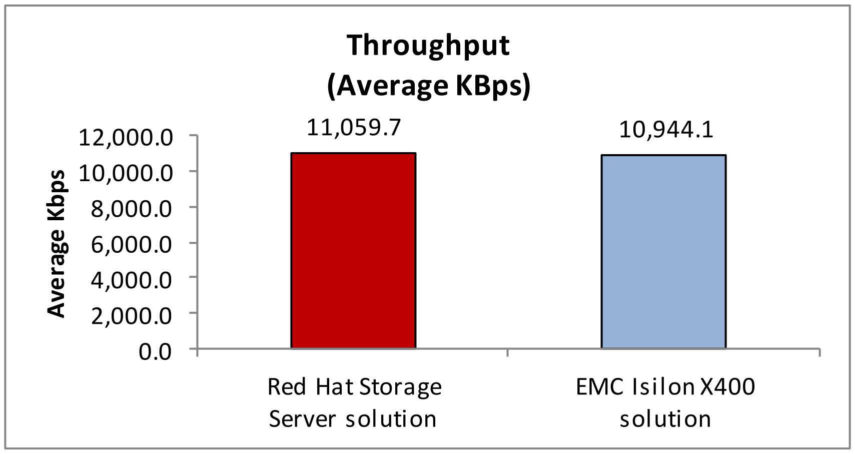


Figure 1: Index throughput in average KBps. The two solutions delivered comparable throughput. (Higher numbers are better.)

[2] EMC Reference Architecture for Splunk https://community.emc.com/docs/DOC-27430

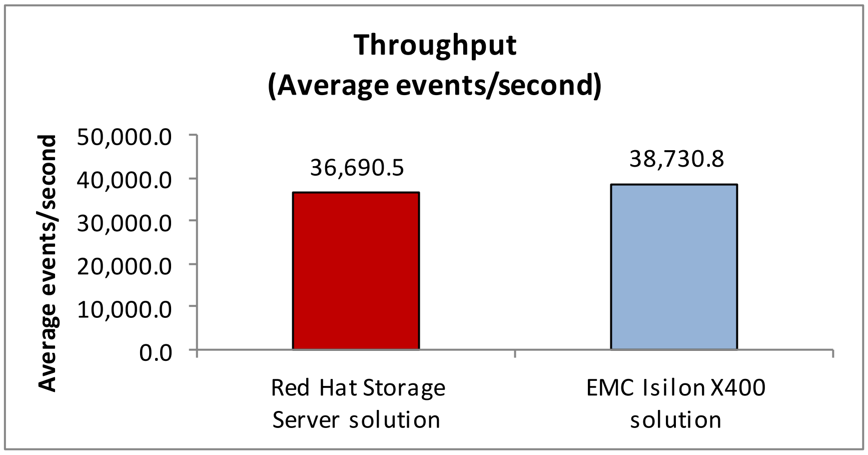


Figure 2: Index throughput achieved by the Red Hat Storage Server solution in events per second. The two solutions delivered comparable throughput. (Higher numbers are better.)[3]

Search time

What this means to you…
When Splunk can return important machine data logs quickly, management and IT staff can identify patterns before they become problems.

Identifying problematic patterns in your machine data as early as possible can help to avoid bigger issues with your business. In terms of potential security threats, the faster a problem can be identified, the sooner it can be solved. Your business and users need these threats reduced and eliminated immediately; speedy search times when running Splunk can translate to faster problem solving.

Figure 3 shows the results of our search test on the Red Hat Storage solution. The Red Hat Storage solution completed this in 17.2 seconds, comparable to the published 18.0 seconds of the eight-node EMC Isilon solution.[4]

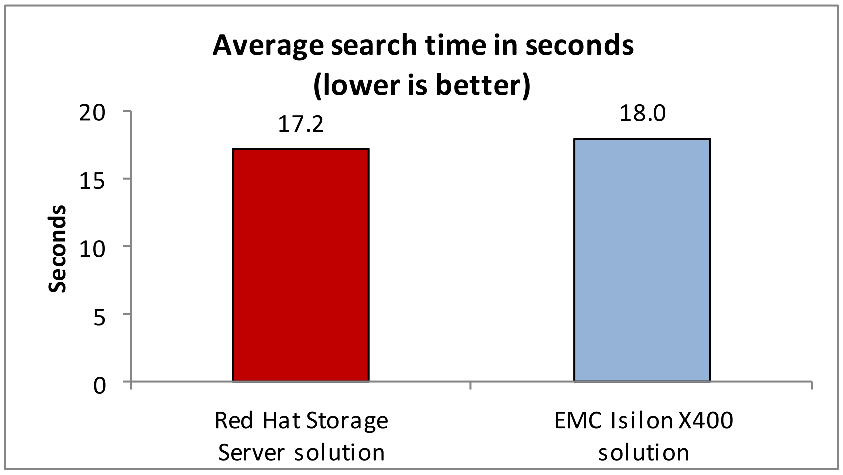


Figure 3: Time the two solutions took to complete the search.[5]

Figure 4 shows the results of our average latency until the first event in the search test on the Red Hat Storage solution. The Red Hat Storage solution completed

[3] EMC Reference Architecture for Splunk https://community.emc.com/docs/DOC-27430
[4] *Ibid*.
[5] *Ibid*.

this in 1.21 seconds, comparable to the published 2.49 seconds of the eight-node EMC Isilon solution.[6]

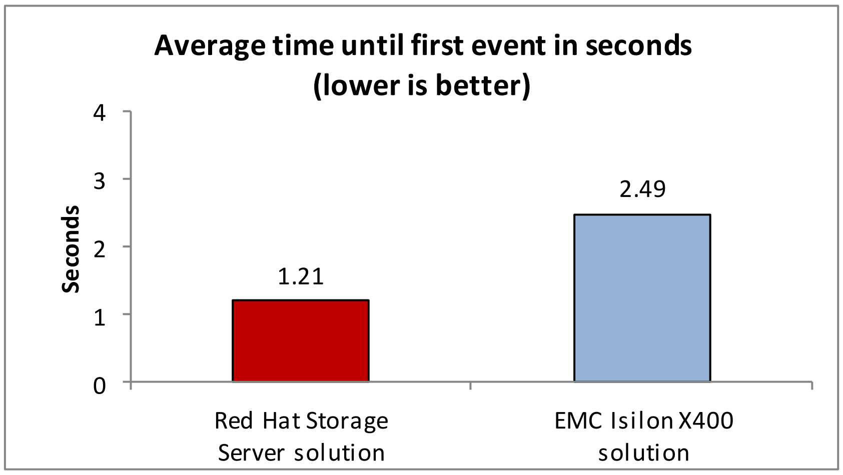


Figure 4: Time the two solutions took until the first event in the search tests.[7]

Cost

The Red Hat Storage and IBM server x solution we tested has a list price of $57,780. This is significantly less than reported prices of the EMC Isilon solution where estimates of list prices were over $600K.[8]

For detailed configuration of our test servers, see Appendix A. We configured SplunkIt using the standard single-indexer and user workload. Both Red Hat Storage and EMC benchmarks were performed by the same SplunkIt workload utility published by Splunk. We tested the Red Hat Storage Server solution in out labs. Wedid not reproduce the EMC Isilon results, and acknowledge potential differences in the specific SplunkIt test methodology for the EMC solution. We were able to sync up many factors including the Splunk server/client CPU Model (Intel E5-2660) and the Splunk indexer mount point (/opt). For our tests, we ran SplunkIt on bare metal while the EMC report indicates that they used virtual (5 index/2 search). For our testing, we used Splunk 6.0.3 and SplunkIt 3.1 on a single host and a single bare-metal client. For the EMC the versions are not reported, nor the client count and the Splunk configuration is distributed.

Appendix B shows how we performed our tests on the Red Hat Storage solution. For more information on the components we used in testing, see Appendix C.

[6] EMC Reference Architecture for Splunk https://community.emc.com/docs/DOC-27430
[7] *Ibid.*
[8] Pricing data provided by Red Hat and IBM as of 6/3/2014. Price of the Red Hat Storage solution includes Red Hat Storage Server for On-premise, Standard (2 Nodes) 1-year subscription cost.

IN CONCLUSION

With the speed and capacity that Red Hat Storage Server on IBM System x servers can offer, analyzing your business's operational data stores can lead to a better understanding of machine data patterns, presenting crucial, real-time opportunities that allow management to make the difficult decisions that improve operational efficiency. In tests with the SplunkIt 2.0 benchmark, we found that our Red Hat Storage solution achieved index test throughput of up to 36,690.47 EPS and up to 11,059.68 KBps. When we ran the search test, the Red Hat Storage Server solution had an average search time of 17.2 seconds with an average latency of 1.2 second until the first event. If your business runs needs to analyze its growing data stores or identify problems and patterns within your infrastructure, Red Hat Storage Server on IBM System x is a cost-efficient storage option for Splunk Enterprise software that can deliver strong performance.

APPENDIX A – SYSTEM CONFIGURATION INFORMATION

Figure 4 provides detailed configuration information for the test systems.

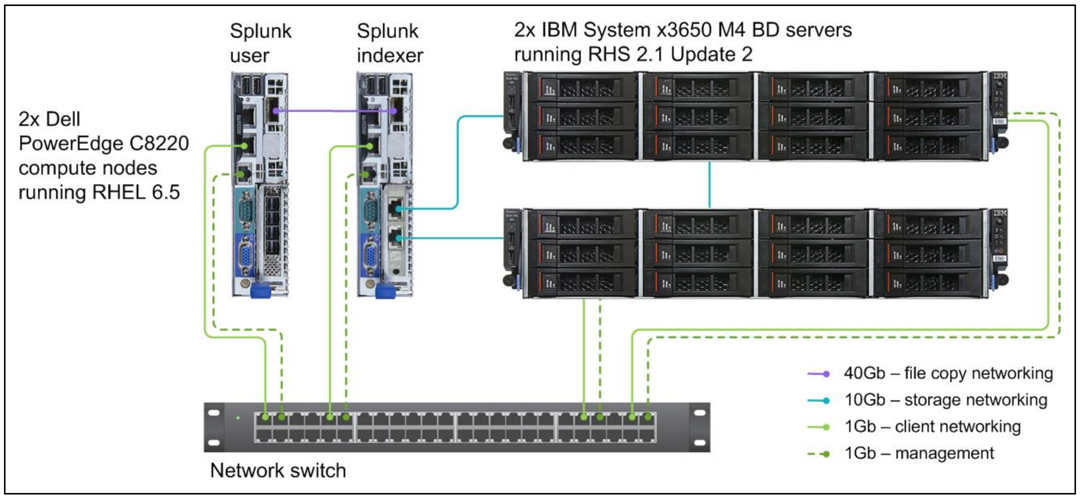
System	IBM System x3650 M4 BD
Power supplies	
Total number	2
Vendor and model number	IBM 00AL742
Wattage of each (W)	750
General	
Number of processor packages	2
Number of cores per processor	10
Number of hardware threads per core	2
System power management policy	Max performance
CPU	
Vendor	Intel
Name	Xeon
Model number	E5-2680 v2
Stepping	M1
Socket type	FCLGA2011
Core frequency (GHz)	2.8
Bus frequency	8 GT/s
L1 cache	32 KB I + 32 KB D (per core)
L2 cache	256 KB on chip (per core)
L3 cache	25 MB (shared)
Platform	
Vendor and model number	IBM System x3650 M4 BD
BIOS settings	Max performance
Memory module(s)	
Total RAM in system (GB)	64
Vendor and model number	Hynix HMT41GR7AFR4A-PB / Micron MT18KSF1G72PZ-1G6E1HE
Type	PC3L-12800R
Speed (MHz)	1600 MHz
Speed running in the system (MHz)	1600 MHz
Timing/Latency (tCL-tRCD-tRP-tRASmin)	11-11-11-35
Size (GB)	8
Number of RAM module(s)	8
Chip organization	Dual
Rank	Single
Operating system	
Name	Red Hat Storage Server
Build number	2.1 Update 2
File system	xfs
Kernel	2.6.32-358.41.1.el6.x86_64
Language	English

System	**IBM System x3650 M4 BD**
RAID controller	
Vendor and model number	ServeRAID F5115-200
Firmware version	3.300.45-0011
Driver version	06.504.01.00-rh1
Cache size (MB)	1GB
NytroCache size (GB)	184.4
NytroCache type	R1-EC
Hard drives	
Vendor and model number	Western Digital WD4000FYYZ-23UL1B0
Number of drives	14
Size	4 TB
RPM	7200
Type	6Gb SATA
Ethernet adapters	
First network adapter	
Vendor and model number	Intel Quad Port I350-AM4 GbE
Type	Integrated
Second network adapter	
Vendor and model number	Intel Ethernet Converged Network Adapter X540-T2
Type	PCIe
USB ports	
Number	6
Type	2.0

Figure 4: Detailed information of our test system.

APPENDIX B – HOW WE TESTED

For our testing, we configured the Splunk Enterprise infrastructure using the default SplunkIt workload recommendations: a single Splunk user system to generate the workload and a single Splunk indexer, each running Red Hat Enterprise Linux 6.5 on a Dell PowerEdge C8220 server node. Figure 5 shows our test components and configuration.



Splunk user
Splunk indexer
2x IBM System x3650 M4 BD servers running RHS 2.1 Update 2
2x Dell PowerEdge C8220 compute nodes running RHEL 6.5
40Gb – file copy networking
10Gb – storage networking
1Gb – client networking
1Gb – management
Network switch

Figure 5: The configuration of components we used in testing.

The Red Hat Storage Server configuration we tested consisted of two IBM x3650 M4 BD servers running Red Hat Storage 2.1 Update 2, each with the following:

- 64 GB of RAM
- One IBM ServeRAID F5115 RAID adapter with a 2 x 100GB mirrored SSD Nytro cache volume
- 14 x 4TB 7200RPM NL-SAS drives (2 drives in RAID 1 for OS, 12 drives in RAID 6 for Splunk data)

We installed Red Hat Enterprise Linux 6.5 on the PowerEdge C8220 servers, to be used for Splunk user and index compute nodes, and a controller node.

Installing Red Hat Storage 2.1 Update 2 on the IBM System x3650 M4 BD servers

Prior to these steps, we created two virtual drives on each server: A two-drive RAID 1 virtual drive for the Red Hat Storage OS, and a twelve-drive RAID 6 for the data volumes. We also configured the ServeRAID F5115 NytroCache volume in a RAID 1 and assigned it solely to the RAID 6 data volume. The ServeRAID F5115 RAID adapter we tested did not come with a BBU, so we configured all volumes under an always write back write policy since this would be a more typical configuration. Following installation of Red Hat Storage 2.1 Update 2, we registered both nodes with Red Hat Network (RHN) and ran all available software updates. Repeat the following for each IBM System x3650 M4 BD server.

1. Insert the Red Hat Storage Linux 2.1 installation media into the CD/DVD drive, and restart the server.
2. When the Welcome to Red Hat Storage 2.1! screen appears, select Install or upgrade an existing system, and press Enter.
3. At the Disc Found screen, select Skip, and press Enter.
4. At the Red Hat Storage Server splash screen, click Next.
5. At the language selection screen, click Next.

6. At the keyboard selection screen, click Next.
7. At the next screen, leave Basic Storage Devices selected, and click Next.
8. At the hostname configuration screen, provide an appropriate name, and click Next. For our servers, we named them `rhs#.test.local`, replacing the # with the number of the server we were installing RHS on, with the first server being `rhs1.test.local`, and the second server `rhs2.test.local`.
9. At the time zone configuration screen, select the correct time zone, and click Next.
10. At the root account screen, provide a root password, retype it in the Confirm field, and click Next. If a Weak Password popup appears, click Use Anyway.
11. At the installation type screen, select Use All Space, and click Next.
12. At the Format Warnings window, click Format.
13. At the Writing storage configuration to disk window, click Write changes to disk.
14. When installation finishes, remove the Red Hat Storage 2.1 DVD, and click Reboot.

Installing Red Hat Enterprise Linux 6.5 on PowerEdge C8220

1. Repeat the following steps for each PowerEdge C8220 server. Insert the Red Hat Enterprise Linux 6.5 installation DVD into the server's DVD drive.
2. On the Welcome to Red Hat Enterprise Linux 6.5 screen, press Enter to boot the installer.
3. On the Welcome to Red Hat Enterprise Linux for x86_64 screen, select SKIP, and press Enter.
4. On the Red Hat Enterprise Linux 6 screen, click Next.
5. On the installation-language screen, select your language (English), and click Next.
6. On the keyboard-selection screen, select the correct format, and click Next.
7. On the Storage Devices screen, select Basic Storage Devices, and click Next.
8. If a pop-up window appears with a Storage Device Warning, click Yes, discard any data.
9. On the Name This Computer screen, enter the server's name, and click Configure Network.
10. On the Network Connections pop-up screen, select your management interface (e.g., eth0), and click Edit.
11. On the Editing eth0 pop-up screen, check the Connect automatically box, and select the IPv4 Settings tab. Change Method to Manual. Click Add, and enter the IP address, Netmask, and Gateway. Enter the IP address of your DNS server. Click Apply to close the pop-up screen.
12. Back on the Network Connections pop-up screen, select the cluster-interconnect interface (e.g., eth1), and click Edit.
13. On the Editing eth1 pop-up screen, check the Connect automatically box, and select the IPv4 Settings tab. Change Method to Manual. Click Add, and enter the IP address, Netmask, and Gateway. Enter the IP address of your DNS server. Click Apply to close the pop-up screen.
14. On the Network Connections pop-up screen, click Close, and click Next.
15. On the Time Zone screen, select your time zone, and click Next.
16. On the administrator's password page, enter the root password, and click Next.
17. On the Installation Type screen, keep the default installation type, check the Review and modify partitioning layout box, and click Next.
18. On the Device assignment screen, select the server's OS disk from the list in the left column, and click the upper arrow to designate the disk as the target device. Click Next.
19. On the Review and Modify Partitioning screen, delete the /home logical volume and assign the resulting free space to the root partition. When finished, click Next.
20. On the Format Warnings pop-up screen, click Format.
21. On the Writing storage configuration to disk screen, click Write changes to disk.
22. On the Boot loader selection screen, keep the defaults, and click Next.
23. On the Red Hat Enterprise Linux software installation screen, select Basic Server, and click Next.
24. When the installation completes, click Reboot.

Configuring the Splunk indexer host

Installing base packages and configure the OS

The following commands install base packages and configure the server:

1. Install packages:

```
    yum install -y acpid cpuspeed wget vim nfs-utils openssh-clients man unzip
smartmontools numactl ipmitool OpenIPMI sysstat xfsprogs tuned irqbalance
```

2. Configure services:

```
chkconfig ipmi on
chkconfig ipmievd on
chkconfig smartd on
```

3. Disable firewall:

```
chkconfig iptables off
chkconfig ip6tables off
```

Mount the RHS volume

The following commands to configure the RHS volume mount options using NFS:

1. Add the following line to /etc/fstab:

```
    rhs1-frontend:/splunk-repl /opt/splunk nfs
tcp,hard,port=2049,noatime,nodiratime,rsize=1048576,wsize=1048576,nfsvers=3,_netdev 0 0
```

2. Mount the volume:

```
mkdir -p /opt/splunk
mount –v /opt/splunk
```

Installing Splunk

The following commands install splunk and configure it to use init.d:

1. Install Splunk packages:

```
rpm -ivh splunk-6.0.*-linux-2.6-x86_64.rpm
```

2. Configure Splunk service:

```
/opt/splunk/bin/splunk start --accept-license
/opt/splunk/bin/splunk enable boot-start
```

Tuning Splunk server

The following commands tune the OS for running splunk:

1. Configure tuned:

```
tuned-adm profile enterprise-storage
```

2. Disable SELinux.
3. Add the following to /etc/sysctl.conf:

```
net.ipv4.tcp_timestamps = 0
net.ipv4.tcp_sack = 1
net.core.netdev_max_backlog = 250000
net.core.rmem_max = 4194304
net.core.wmem_max = 4194304
net.core.rmem_default = 4194304
net.core.wmem_default = 4194304
net.core.optmem_max = 4194304
net.ipv4.tcp_rmem = 4096 87380 4194304
net.ipv4.tcp_wmem = 4096 65536 4194304
```

```
    net.ipv4.tcp_low_latency = 1
```

Configuring the Splunk user host

Installing base packages and configure the OS

The following commands install base packages and configure the server:

1. Install packages:

```
    yum install -y acpid cpuspeed wget vim nfs-utils openssh-clients man unzip
smartmontools numactl ipmitool OpenIPMI sysstat xfsprogs tuned irqbalance xorg-x11-utils
xorg-x11-xauth
```

2. Configure services:

```
chkconfig ipmi on
chkconfig ipmievd on
chkconfig smartd on
```

3. Disable firewall:

```
chkconfig iptables off
chkconfig ip6tables off
```

Tuning server

The following commands tune the OS:

1. Configure tuned:

```
tuned-adm profile enterprise-storage
```

2. Disable SELinux:
3. Add the following to /etc/sysctl.conf:

```
net.ipv4.tcp_timestamps = 0
net.ipv4.tcp_sack = 1
net.core.netdev_max_backlog = 250000
net.core.rmem_max = 4194304
net.core.wmem_max = 4194304
net.core.rmem_default = 4194304
net.core.wmem_default = 4194304
net.core.optmem_max = 4194304
net.ipv4.tcp_rmem = 4096 87380 4194304
net.ipv4.tcp_wmem = 4096 65536 4194304
net.ipv4.tcp_low_latency = 1
```

Configuring the RHS hosts

Registering with RHN and installing updates:

These commands register with RHN and install updates:

1. Register with RHN:

```
rhn_register
rhn-channel --add --channel=rhel-x86_64-server-6-rhs-2.1
rhn-channel --add --channel=rhel-x86_64-server-sfs-6.4.z
```

2. Install updates and reboot:

```
yum update –y
reboot
```

3. Disable firewall:

```
chkconfig iptables off
chkconfig ip6tables off
```

Installing additional packages and configure the OS

The following commands install base packages and configure the server:

1. Install packages:

```
    yum install -y acpid cpuspeed wget vim nfs-utils openssh-clients man numactl
ipmitool OpenIPMI sysstat xfsprogs tuned irqbalance
```

2. Configure services:

```
    chkconfig ipmi on
    chkconfig ipmievd on
```

3. Disable firewall:

```
    chkconfig iptables off
    chkconfig ip6tables off
```

Configuring RHS storage brick

Use the following commands to partition and format the RHS storage brick device:

1. Partition using LVM:

```
    pvcreate --dataalignment 2560K /dev/sdb
    vgcreate rhs /dev/sdb
    lvcreate -l 100%VG -n brick rhs
```

2. Format using XFS and mount with specified fstab parameters:

```
    mkfs.xfs -f -i size=512 -d su=256k,sw=10 /dev/rhs/brick
    echo -e "UUID=`blkid -s UUID -o value
/dev/rhs/brick`\t/rhs/brick\txfs\tdefaults,nobarrier,nodiratime,noatime,logbufs=8,logbsi
ze=256k,allocsize=4096,inode64\t0 0" >> /etc/fstab
    mkdir -p /rhs/brick
    mount -v /rhs/brick
```

Configuring Gluster storage volume

Use the following commands to configure the Gluster volume.

1. Configure gluster peers:

```
    gluster peer probe 192.168.4.102
    gluster peer status
```

2. Configure gluster volume:

```
    gluster volume create splunk-repl replica 2 transport tcp
192.168.4.101:/rhs/brick/brick1 192.168.4.102:/rhs/brick/brick2
    gluster volume set splunk-repl group small-file-perf
    gluster volume start splunk-repl
    gluster volume status splunk-repl
    gluster volume info splunk-repl
```

Tuning server

The following commands tune the OS:

1. Configure tuned:

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

2. Disable SELinux.
3. Add the following to /etc/sysctl.conf:

```
    net.ipv4.tcp_timestamps = 0
```

```
net.ipv4.tcp_sack = 1
net.core.netdev_max_backlog = 250000
net.core.rmem_max = 4194304
net.core.wmem_max = 4194304
net.core.rmem_default = 4194304
net.core.wmem_default = 4194304
net.core.optmem_max = 4194304
net.ipv4.tcp_rmem = 4096 87380 4194304
net.ipv4.tcp_wmem = 4096 65536 4194304
net.ipv4.tcp_low_latency = 1
```

Running SplunkIT

Benchmark script:

We used the following script to prepare and run all of the SplunkIT benchmark on the splunk user host.

splunkit-test.sh:

```
#!/bin/bash
INTERVAL=30
TIMESTAMP=$(date +%s)

SPLUNKIT_DIR=/mnt/splunkit
SPLUNKIT_SYSLOG=${SPLUNKIT_DIR}/splunkit-server/data/static/syslog_50g.log
SPLUNKIT_TESTLIST="index search"

SPLUNK_DIR=/opt/splunk
#INDEX_DIR=/opt/splunk/var/lib
INDEX_DIR=/opt/splunk
SPLUNK_BACKUP=/opt/splunk.tar.gz
SPLUNK_ADMIN="admin"
SPLUNK_PASSWORD="Password1"
SPLUNK_CONF=/opt/splunk-conf

RHS1="rhs1"
RHS_OTHER="rhs2"
SPLUNK_SERVER="splunk-server"
SPLUNK_USER="splunk-user"
HOSTS="$RHS1 $RHS_OTHER $SPLUNK_SERVER $SPLUNK_USER"
RHS_HOSTS="$RHS1 $RHS_OTHER"
GLUSTER_VOL=splunk-repl
#GLUSTER_VOL=splunk-distrib

SAVEDSEARCH_LIST="
splunkit_aggqueue_chart_idxtest
splunkit_aggqueue_chart_searchtest
splunkit_index_chart_idxtest
splunkit_index_chart_searchtest
splunkit_index_stats_idxtest
splunkit_index_stats_searchtest
splunkit_indexqueue_chart_idxtest
splunkit_indexqueue_chart_searchtest
splunkit_load_chart_idxtest
splunkit_load_chart_searchtest
splunkit_parsingqueue_chart_idxtest
splunkit_parsingqueue_chart_searchtest
splunkit_queue_chart_idxtest
```

```
splunkit_queue_chart_searchtest
splunkit_search_chart
splunkit_search_stats
splunkit_typingqueue_chart_idxtest
splunkit_typingqueue_chart_searchtest"

RESULTSEARCH_LIST="
splunkit_index_stats_idxtest
splunkit_search_stats"

SSEXT_LIST="png html pdf"

for TEST in ${SPLUNKIT_TESTLIST}; do

TESTNAME="${TEST}test"
RESULTS_NAME=splunkit_${TIMESTAMP}_${TESTNAME}
RESULTS_DIR=${SPLUNKIT_DIR}/results/${RESULTS_NAME}
RESULTS_STATS=${RESULTS_DIR}/perflogs
RESULTS_INFO=${RESULTS_DIR}/sysinfo
RESULTS_SS=${RESULTS_DIR}/dashboards
RESULTS_LOGS=${RESULTS_DIR}/logs
RESULTS_CSV=${RESULTS_DIR}/csv

mkdir -p $RESULTS_DIR
mkdir -p $RESULTS_STATS
mkdir -p $RESULTS_INFO
mkdir -p $RESULTS_SS
mkdir -p $RESULTS_LOGS
mkdir -p $RESULTS_CSV
cp $0 $RESULTS_DIR/

# Cleanup
if [ "${TEST}" == "index" ]; then
  ssh $SPLUNK_SERVER "service splunk stop; sync ; rm -rf ${SPLUNK_DIR}/* ; sync ;
umount ${INDEX_DIR} ; mount ${INDEX_DIR} ; rm -rf ${SPLUNK_DIR}/* ; sync ; tar -
zxf ${SPLUNK_BACKUP} -C ${SPLUNK_DIR}/../ ; cp -rvf ${SPLUNK_CONF}/*
${SPLUNK_DIR}/ ; sync"
fi

# Drop caches and swap
echo "Dropping caches and swap..."
for HOST in $HOSTS; do
  ssh $HOST "swapoff -a ; sync ; echo 1 > /proc/sys/vm/drop_caches ; swapon -a" &
  wait
done

# Fetch pre-test system info
echo "Gathering system information..."
for HOST in $HOSTS; do
  ssh $HOST "set -x ; uname -a ; ifconfig ; mount -v ; df ; cat /etc/fstab ;
lsblk ; lvs ; cat /proc/meminfo ; cat /proc/cpuinfo ; sysctl -e -p ; rpm -qa |
sort" > $RESULTS_INFO/info_${HOST}.txt 2>&1 &
done
for RHS_HOST in $RHS_HOSTS; do
  ssh $RHS_HOST 'PATH=$PATH:/opt/MegaRAID/storcli; set -x; storcli64 /cALL show
all ; storcli64 /cALL/vALL show all' > $RESULTS_INFO/storcli_${RHS_HOST}.txt 2>&1
&
```

```
done
[ ${RHS1} ] && ssh $RHS1 "set -x ; gluster volume info ${GLUSTER_VOL} ; gluster
volume status ${GLUSTER_VOL} ; gluster volume heal ${GLUSTER_VOL} info" >
$RESULTS_INFO/gluster_info.txt 2>&1
wait
sleep 1

# Preload syslog file into memory
#if [ "${TEST}" == "index" ]; then
#  echo "Preloading syslog testfile..."
#  ssh $SPLUNK_SERVER "dd if=${SPLUNKIT_SYSLOG} of=/dev/null bs=1M skip=25600" &
#  for i in `seq 1 5`; do
#    ssh $SPLUNK_SERVER "dd if=${SPLUNKIT_SYSLOG} of=/dev/null bs=1M ; sync"
#    wait
#  done
#  sleep 1
#fi

# Restart gluster and start profile logging
if [ ${RHS1} ]; then
  echo "Restart gluster and start profile logging..."
  ssh $SPLUNK_SERVER "service splunk stop ; sync ; umount ${INDEX_DIR}"
  ssh $RHS1 "gluster volume profile ${GLUSTER_VOL} stop ; echo y | gluster volume
stop ${GLUSTER_VOL} ; gluster volume profile ${GLUSTER_VOL} start ; gluster volume
start ${GLUSTER_VOL}"
  sleep $INTERVAL
  ssh $RHS1 "echo -e '#!/bin/bash\nwhile [ 1 ]; do gluster volume profile
${GLUSTER_VOL} info ; gluster volume profile ${GLUSTER_VOL} info nfs 1>&2 ; sleep
${INTERVAL}; done' > /tmp/gv_prof.sh ; chmod +x /tmp/gv_prof.sh ; /tmp/gv_prof.sh"
1> $RESULTS_STATS/gluster_profile.txt 2> $RESULTS_STATS/gluster_profile_nfs.txt &
  ssh $SPLUNK_SERVER "mount -v ${INDEX_DIR} ; service splunk start ; sync"
  sleep 1
fi

# Start statistics collection
echo "Starting stat collection..."
for HOST in $HOSTS; do
  ssh $HOST "pkill vmstat ; vmstat -n ${INTERVAL} | sed -e 's/^[ \t]*//' -e
'3d'" > ${RESULTS_STATS}/vmstat_${HOST}.log &
  ssh $HOST "pkill sar ; rm -f /tmp/sar_*.bin ; sar -o /tmp/sar_${HOST}.bin
${INTERVAL} > /dev/null 2>&1" &
  ssh $HOST "pkill top ; top -b -i -H -d ${INTERVAL}" >
${RESULTS_STATS}/top_${HOST}.log &
done
for RHS_HOST in $RHS_HOSTS; do
  echo 'Hit Count (Rd),Hit Count (Wr),Miss Count (Rd),Miss Count (Wr),Cache Fills
(Rd),Cache Fills (Wr),Cache Flushes,Total CWs,Non Dirty CWs,Dirty CWs,Host IO
Count (Rd),Host IO Count (Wr),Disk IO Count (Rd),Disk IO Count (Wr)' >
$RESULTS_CSV/nytrocache_${RHS_HOST}.csv
  ssh $RHS_HOST "/opt/MegaRAID/storcli/storcli64 /cALL set stats=off 2>&1 >
/dev/null ; /opt/MegaRAID/storcli/storcli64 /cALL set stats='on
interval=${INTERVAL}' 2>&1 > /dev/null ; echo -e '#!/bin/bash\nwhile [ 1 ]; do
/opt/MegaRAID/storcli/storcli64 /cALL show stats type=NytroCache all 1>&2 ;
/opt/MegaRAID/storcli/storcli64 /cALL show stats type=NytroCache all csv | grep ,
2>&1 ; sleep ${INTERVAL}; done' > /tmp/nytro_stats.sh ; chmod +x
/tmp/nytro_stats.sh ; /tmp/nytro_stats.sh" 2>
$RESULTS_STATS/nytrocache_${RHS_HOST}.txt 1>>
```

```
$RESULTS_CSV/nytrocache_${RHS_HOST}.csv &
done
[ ${RHS1} ] && ssh $RHS1 "top -H -b -d 1 -p \`gluster volume status ${GLUSTER_VOL}
| awk '/NFS/&&/localhost/{print \$7}'\`" > ${RESULTS_STATS}/top_glusterfs.log &

# Run test
sleep `expr $INTERVAL \* 2`

if [ "${TEST}" == "index" ]; then
echo "Running splunkit ${TESTNAME}..."
  ssh $SPLUNK_SERVER "cd ${SPLUNKIT_DIR}/splunkit-server ; python
bin/indextest.py" 2>&1 | tee $RESULTS_LOGS/output.log
elif [ "${TEST}" == "search" ]; then
echo "Running splunkit ${TESTNAME}..."
  ssh $SPLUNK_SERVER "cd ${SPLUNKIT_DIR}/splunkit-server ; python
bin/searchtest.py" 2>&1 > $RESULTS_LOGS/server_output.log &
  cd ${SPLUNKIT_DIR}/splunkit-user
  python bin/searchtest.py 2>&1 | tee $RESULTS_LOGS/user_output.log
fi

echo "Test complete!"
sleep `expr $INTERVAL \* 2`

# Stop statistics collection
echo "Stopping stat collection..."
[ ${RHS1} ] && ssh $RHS1 "pkill gv_prof"
for RHS_HOST in $RHS_HOSTS; do
  ssh $RHS_HOST 'pkill nytro_stats'
done
for HOST in $HOSTS; do
  ssh $HOST "pkill vmstat ; pkill iostat ; pkill sar ; pkill top"
done
wait
sleep 1

# Copy stat files
for HOST in $HOSTS; do
  scp $HOST:/tmp/sar_*.bin $RESULTS_STATS/
done

if [ "${TEST}" == "index" ]; then
  scp $SPLUNK_SERVER:$SPLUNKIT_DIR/splunkit-server/log/indextest.log
$RESULTS_LOGS/
  ssh $SPLUNK_SERVER "cat $SPLUNKIT_DIR/splunkit-server/bin/indexRecord.log" | awk
-F'='
'/starttimeu/{printf("starttimeu=%.2f\nendtimeu=%.2f\ntotaltimeu=%.2f\ntotaltimemi
n=%f\n",$2,$3,($3-$2),(($3-$2)/60))}' > $RESULTS_LOGS/indexRecord.log
  ssh $SPLUNK_SERVER "${SPLUNK_DIR}/bin/splunk search 'index=_internal
per_index_thruput series=\"splunkit_idxtest\" | stats avg(kbps), avg(eps)' -auth
${SPLUNK_ADMIN}:${SPLUNK_PASSWORD}" >> $RESULTS_LOGS/indexRecord.log
  ssh $SPLUNK_SERVER "${SPLUNK_DIR}/bin/splunk search 'index=_internal
per_index_thruput series=\"splunkit_idxtest\"' -maxout 0 -output csv -wrap 0 -auth
${SPLUNK_ADMIN}:${SPLUNK_PASSWORD}" > $RESULTS_CSV/per_index_thruput_idxtest.csv
  cp $RESULTS_LOGS/indexRecord.log $RESULTS_DIR/results.txt
elif [ "${TEST}" == "search" ]; then
  scp $SPLUNK_SERVER:$SPLUNKIT_DIR/splunkit-server/log/*.log $RESULTS_LOGS/
  scp $SPLUNK_SERVER:$SPLUNKIT_DIR/splunkit-server/bin/*.log $RESULTS_LOGS/
  scp $SPLUNK_SERVER:$SPLUNKIT_DIR/splunkit-server/results/*.log $RESULTS_LOGS/
  scp $SPLUNK_SERVER:$SPLUNKIT_DIR/splunkit-server/results/*.csv $RESULTS_CSV/

  scp $SPLUNK_USER:$SPLUNKIT_DIR/splunkit-user/log/*.log $RESULTS_LOGS/
  scp $SPLUNK_USER:$SPLUNKIT_DIR/splunkit-user/results/*.log $RESULTS_LOGS/
  scp $SPLUNK_USER:$SPLUNKIT_DIR/splunkit-user/results/*.csv $RESULTS_CSV/

  ssh $SPLUNK_SERVER "cat $SPLUNKIT_DIR/splunkit-server/bin/searchRecord.log" |
awk -F'='
'/starttimeu/{printf("starttimeu=%.2f\nendtimeu=%.2f\ntotaltimeu=%.2f\ntotaltimemi
n=%f\n",$2,$3,($3-$2),(($3-$2)/60))}' > $RESULTS_LOGS/searchRecord.log
  ssh $SPLUNK_SERVER "${SPLUNK_DIR}/bin/splunk search 'index=splunkit_searchtest
sourcetype=splunkit_results | stats avg(time_to_first_event) AS \"Average Time to
First Event (s)\", avg(time_to_search) AS \"Average Time to Search (s)\"' -maxout
0 -auth ${SPLUNK_ADMIN}:${SPLUNK_PASSWORD}" >> $RESULTS_LOGS/searchRecord.log

  for SAVEDSEARCH in $SAVEDSEARCH_LIST; do
    ssh $SPLUNK_SERVER "${SPLUNK_DIR}/bin/splunk search '| savedsearch
\"${SAVEDSEARCH}\"' -maxout 0 -output csv -wrap 0 -auth
${SPLUNK_ADMIN}:${SPLUNK_PASSWORD}" > $RESULTS_CSV/${SAVEDSEARCH}.csv
  done

  for RESULTSEARCH in $RESULTSEARCH_LIST; do
    ssh $SPLUNK_SERVER "${SPLUNK_DIR}/bin/splunk search '| savedsearch
\"${RESULTSEARCH}\"' -maxout 0 -output table -wrap 0 -auth
${SPLUNK_ADMIN}:${SPLUNK_PASSWORD}" >> $RESULTS_DIR/_result_${RESULTSEARCH}.txt
  done
  paste -d' ' $RESULTS_DIR/_result*.txt > $RESULTS_DIR/results.txt
  mv -f $RESULTS_DIR/_result*.txt /tmp/

  for SSEXT in $SSEXT_LIST; do
    CutyCapt --url="http://${SPLUNK_SERVER}:8000/en-
US/account/insecurelogin?username=${SPLUNK_ADMIN}&password=${SPLUNK_PASSWORD}&retu
rn_to=%2Fen-US%2Fapp%2Fsearch%2Fsplunkit_results" --
out=$RESULTS_SS/splunkit_results.${SSEXT} --delay=5000
    CutyCapt --url="http://${SPLUNK_SERVER}:8000/en-
US/account/insecurelogin?username=${SPLUNK_ADMIN}&password=${SPLUNK_PASSWORD}&retu
rn_to=%2Fen-US%2Fapp%2Fsearch%2Fsplunkit_indexing_details%3Fform.test%3Didxtest" -
-out=$RESULTS_SS/splunkit_indexing_details_idxtest.${SSEXT} --delay=5000
    CutyCapt --url="http://${SPLUNK_SERVER}:8000/en-
US/account/insecurelogin?username=${SPLUNK_ADMIN}&password=${SPLUNK_PASSWORD}&retu
rn_to=%2Fen-
US%2Fapp%2Fsearch%2Fsplunkit_indexing_details%3Fform.test%3Dsearchtest" --
out=$RESULTS_SS/splunkit_indexing_details_searchtest.${SSEXT} --delay=5000
  done
fi

echo "${TESTNAME} DONE!!!"
done

cd ${SPLUNKIT_DIR}/results
tar -Jcf splunkit_${TIMESTAMP}.tar.xz splunkit_${TIMESTAMP}*
```

APPENDIX C – WHAT WE TESTED

About Red Hat Storage Server

Red Hat Storage Server is a software-based, or according to Red Hat "software-defined," storage platform to manage big, semi-structured, and unstructured data growth while maintaining performance, capacity, and availability to meet demanding enterprise storage requirements. Along with the ability to deploy on-premises or in a cloud environment, Red Hat Storage Server has flexible deployment options to meet various business needs.

For more information about Red Hat Storage, visit www.redhat.com/products/storage-server/.

About IBM x3650 M4 BD

According to IBM, the IBM System x3650 M4 BD can provide "the capacity, performance, and efficiency you need for Big Data workloads" and can "rapidly analyze enormous volumes of data. This System x server supports Big Data applications from IBM and independent software vendors (ISVs)." Powered by Intel Xeon E5-2600 v2 processors, the x3650 M4 BD has up to fourteen 3.5-inch HDDs, which includes two rear drives with their own RAID controller to separate the OS from the business data; up to 56 TB of internal storage; up to 12 cores per CPU; and 16 DIMMS.

For more information about the IBM System x3650 M4 BD, visit www-03.ibm.com/systems/x/hardware/rack/x3650m4bd/.

About Dell C8220

According to Dell, the two-socket compute node PowerEdge C8220 powered by the Intel Xeon processor E5-2600 and E5-2600v2 product families can run your data center "with density and versatility." The server features 16 DIMM slots for up to 512GB memory per node; 2TB SATA II or 1.6TB SATA SSD for internal storage; 2 x 2.5" drive options; 1 x8 PCI Express 3.0 express mezzanine and 1 x16 PCI Express 3.0 I/O slots; and one Intel i350 quad-port 1GB low-profile NIC adapter.

For more information about the Dell PowerEdge C8220, visit www.dell.com/us/business/p/poweredge-c8220/pd.

About Splunk

Splunk monitors and evaluates "everything from customer clickstreams and transactions to network activity and call records" to turn your machine data into searchable and useful information. It can collect and index machine-generated data from most sources or locations, including data streams from packaged and custom applications, application servers, web servers, databases, networks, virtual machines, telecoms equipment, operating systems, and sensors.

For more information about Splunk Enterprise, visit www.splunk.com/view/splunk/SP-CAAAG57.

About SplunkIt 2.0

The SplunkIt 2.0 benchmark produces data for Splunk to index and search and then measures performance in terms of throughput and search times. There are three categories of information for the data returned from a search: host (which machine generated the data), source (which program generated the data), and source type. Searches in the benchmark are conducted under an indexing load and follow a pattern so that data replication is consistent. This pattern means that SplunkIt can measure performance in a consistent and repeatable manner. Consistency and repeatability translates to reliable evaluation of different Splunk configurations involving operating systems, storage, and servers. We ran both tests three times and report the median in the following sections.

For more information about SplunkIt, visit apps.splunk.com/app/749/.

ABOUT PRINCIPLED TECHNOLOGIES

Principled Technologies, Inc.
1007 Slater Road, Suite 300
Durham, NC, 27703
www.principledtechnologies.com

We provide industry-leading technology assessment and fact-based marketing services. We bring to every assignment extensive experience with and expertise in all aspects of technology testing and analysis, from researching new technologies, to developing new methodologies, to testing with existing and new tools.

When the assessment is complete, we know how to present the results to a broad range of target audiences. We provide our clients with the materials they need, from market-focused data to use in their own collateral to custom sales aids, such as test reports, performance assessments, and white papers. Every document reflects the results of our trusted independent analysis.

We provide customized services that focus on our clients' individual requirements. Whether the technology involves hardware, software, Web sites, or services, we offer the experience, expertise, and tools to help our clients assess how it will fare against its competition, its performance, its market readiness, and its quality and reliability.

Our founders, Mark L. Van Name and Bill Catchings, have worked together in technology assessment for over 20 years. As journalists, they published over a thousand articles on a wide array of technology subjects. They created and led the Ziff-Davis Benchmark Operation, which developed such industry-standard benchmarks as Ziff Davis Media's Winstone and WebBench. They founded and led eTesting Labs, and after the acquisition of that company by Lionbridge Technologies were the head and CTO of VeriTest.