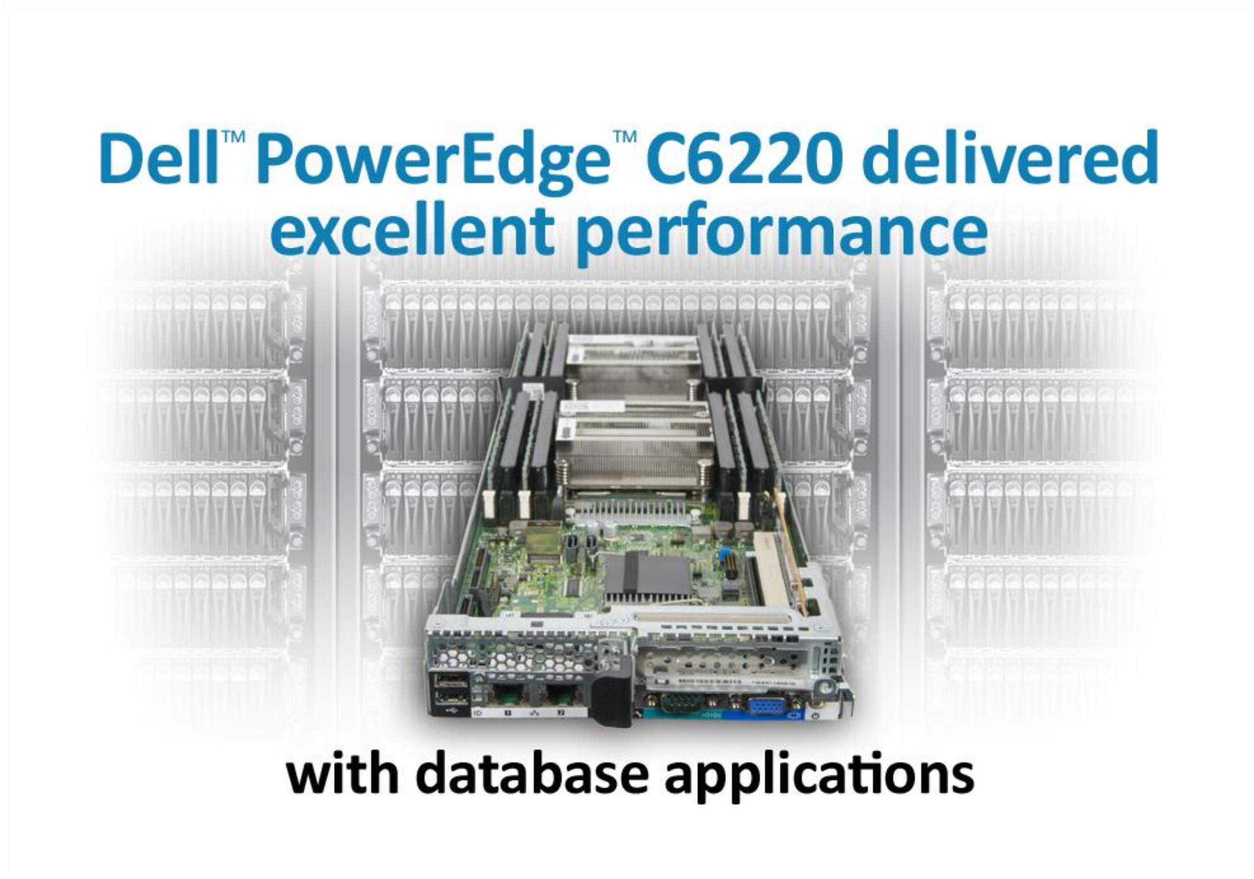
DELL POWEREDGE C6220: PERFORMANCE FOR LARGE INFRASTRUCTURES



Organizations that run large data centers to provide cloud services or software as a service (SaaS) require servers that are compact, flexible, easy to maintain, and deliver outstanding performance, along with low-cost, feature-laden software that integrates easily. Customers expect to be able to access their data on demand, so high-performing but concentrated compute fabrics are a necessity for these data centers.

The Dell PowerEdge C6220 server is designed for large data centers specializing in cloud computing and other massive scale-out environments. It contains up to four two-socket server nodes that provide flexibility and easy maintenance in a dense, 2U form factor. Organizations with large deployments often elect to pair this powerful hyperscale hardware with open-source Linux-based operating systems and open-source application stacks to maximize cost efficiency, performance, flexibility, and ease of management.

In our labs at Principled Technologies, we put the performance of the Dell PowerEdge C6220 server with a CentOS-based LAMP software stack to the test. We found that a single Dell PowerEdge C6220 with an open-source LAMP software stack was able to handle up to 119,758 orders per minute across multiple Web sites and databases, indicating a level of performance ideal for scale-out deployments.

A PRINCIPLED TECHNOLOGIES TEST REPORT

Commissioned by Dell Inc.; April 2012

SCALE-OUT PERFORMANCE

The performance your infrastructure delivers to end users depends upon two factors: the hardware that powers your workloads, and the software that runs them. We tested the performance of a hardware and software combination well suited for large-scale server deployments: the Dell PowerEdge C6220 server and an open-source LAMP software stack, running the CentOS operating system, Apache Web Server, MySQL databases, and PHP for the scripting language.

The size, number, and variation of customers' Web sites and databases will always vary based on your customers' needs. For our testing, we used from one to four Web sites, each with their own MySQL database instance. For testing, we used a 10GB database. We found that the LAMP stack on the Dell PowerEdge C6220 was able to process 38,793 orders per minute while running one Web site/database, 79,759 orders per minute while running two Web sites/databases, and 119,758 orders per minute while running four DVD store Web sites, each with an independent MySQL database instance. Figure 1 shows the performance, by increasing numbers of Web sites, that the server achieved, in orders per minute.

Number of DVD store Web sites	Orders per minute
1	38,793
2	79,759
4	119,758

Figure 1: Performance results, in orders per minute, for varying numbers of Web sites.

These results are comparable to another recently published single-node DVD Store report using Linux and MySQL, which offered single-node scores ranging from 36,000 to 70,000 OPM.[1]

About the Dell PowerEdge C6220 server

The Dell PowerEdge C6220 is a rack server designed for large data centers that require extensive efficiency, flexibility, performance, and maintenance features. The PowerEdge C6220 can house up to four hot-swappable server nodes, which are each powered by two sockets containing processors from

Highlights of the Dell PowerEdge C6220

- **Shared infrastructure uses less floor space, power, and cooling**
- **Dense form factor ideal for service providers, hosting platforms, and hyperscale environments**
- **Up to 36TB raw storage in a single four node chassis**
- **Embedded management control per node**

[1] http://www.principledtechnologies.com/clients/reports/Red%20Hat/LAMP_stack_0412.pdf

the Intel® Xeon® processor E5-2600 series. Using multiple hot-swappable server nodes provides the flexibility to reuse or repurpose servers when workloads change, eliminating downtime in the process. Fitting the power of four servers into just a standard 2U rack slot with a number of drive options for efficient storage allocation, the Dell PowerEdge C6220 easily meets the needs of large-scale server deployments. For more information about the PowerEdge C6220, and the entire Dell PowerEdge C series, visit http://www.dell.com/us/enterprise/p/poweredge-cloud-servers.

About the LAMP stack we tested

LAMP is an acronym for the components of a multi-tier software stack that an infrastructure may run: the Linux operating system, Apache Web server, MySQL database, and PHP scripting language. Service providers have used this application stack for years to provide powerful and flexible multi-tier environments for customers. Below, we briefly describe each component.

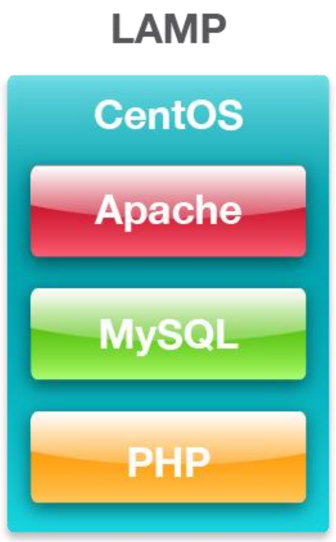


Figure 2: The LAMP stack.

About CentOS Server

Community ENTerprise Operating System, commonly known as CentOS, is an open-source Linux-based operating system designed to provide organizations with an enterprise-class computing platform that is free to use. CentOS incorporates numerous advanced operating system features such as virtualization capabilities, enhanced memory management capabilities, and ext4 file system support. For more information about CentOS, visit http://www.centos.org/.

Apache

Apache HTTP Server, initially released in 1995, has been a driving force in the growth of the World Wide Web over the past two decades. It is an open-source Web server that includes such features as Secure Sockets Layer and Transport Layer Security support, filtering support, and custom log files. For more information, visit http://www.apache.org/.

MySQL

MySQL, initially developed in 1995 and open-sourced in 2000, is one of the most common database platforms backing both external Web sites and internal database infrastructures. It delivers high performance, high reliability, and is easy to use. Running on more than 20 operating system platforms and providing a range of database tools, MySQL delivers flexibility and performance to large-scale deployments. For more information about MySQL, visit http://www.mysql.com/.

PHP

PHP, initially released to the public in 1995, is a scripting language for Web development that can be embedded into HTML source documents instead of using an external file to process data. The Web server interprets the PHP code to generate a Web page. For more information about PHP, visit http://www.php.net/.

WHAT WE TESTED

About DVD Store Version 2.1

To create our real-world multi-tier LAMP workload, we used the DVD Store Version 2.1 (DS2) benchmarking tool. DS2 models an online DVD store, where customers log in, search for movies, and make purchases. DS2 reports these actions in orders per minute (OPM) that the system could handle, to show what kind of performance you could expect for your customers. The DS2 workload also performs other actions, such as adding new customers, to exercise the wide range of database functions you would need to run your LAMP environment. DS2 supports different versions of database platforms, Web servers, and application servers. As we note above, for this evaluation, we used the MySQL database engine, Apache Web server, and PHP. We ran three test runs of each Web site/database density level, and report the median result of each scenario above. For more information about the DS2 tool, see http://www.delltechcenter.com/page/DVD+Store.

CONCLUSION

Compact, high-performing servers and streamlined software are a necessity for any data center that provides multi-tier hosted Web solutions, cloud services, or other scale-out implementations. Customers need quick access to Web applications that power their business, and the databases that power those Web applications. In our tests, we found that the Dell PowerEdge C6220 server with an open-source LAMP software stack was able to complete up to 38,793 orders per minute for one Web site, 79,759 orders per minute for two Web sites, and 119,758 orders per minute for four Web sites – all while providing the flexibility, efficiency, and maintenance features that large-scale deployments require.

APPENDIX A – SERVER CONFIGURATION INFORMATION

Figure 3 provides detailed configuration information for the test server.

System	Dell PowerEdge C6220
Power supplies (per chassis)	
Total number	2
Vendor and model number	Dell Model: D1200E-S1
Wattage of each (W)	1400
Cooling fans (per chassis)	
Total number	4
Vendor and model number	Dell FHXGY-A00
Dimensions (h x w) of each	2.5" x 2.5"
Volts	12
Amps	3.30
General	
Number of processor packages	2
Number of cores per processor	6
Number of hardware threads per core	2
System power management policy	Balanced
CPU	
Vendor	Intel
Name	Xeon
Model number	E5-2630
Stepping	7
Socket type	LGA2011
Core frequency (GHz)	2.30
Bus frequency	7.2 GT/s
L1 cache	32 KB + 32 KB (per core)
L2 cache	256 KB (per core)
L3 cache (MB)	15
Platform	
Vendor and model number	Dell PowerEdge C6220
Motherboard model number	E145483
BIOS name and version	Dell 1.0.14
BIOS settings	Defaults
Memory module(s)	
Total RAM in system (GB)	128
Vendor and model number	Hynix™ HMT31GR7BFR4C-PB
Type	PC3-12800R
Speed (MHz)	1,600
Speed running in the system (MHz)	1,600
Timing/Latency (tCL-tRCD-tRP-tRASmin)	11-11-11-35
Size (GB)	8
Number of RAM module(s)	16

System	**Dell PowerEdge C6220**
Chip organization	Double-sided
Rank	Dual
Operating system	
Name	CentOS 6.2
Build number	ext4
File system	ext4
Kernel	2.6.32-220.13.1.el6.x86_64
Language	English
Graphics	
Vendor and model number	ASPEED VGA Controller
Graphics memory (MB)	8
Driver	N/A
RAID controller	
Vendor and model number	LSI™ MegaRAID® SAS 9265-8i
Firmware version	3.120.5-1169
Driver and version	LSI MegaRAID SAS Driver: megaraid_sas, 00.00.05.40-rh2
Cache size (GB)	1
Hard drive	
Vendor and model number	Toshiba MBF2600RC
Number of disks in system	24 (6 disks per node)
Size (GB)	600
Buffer size (MB)	16
RPM	10K
Type	6Gb SAS
Ethernet adapters	
Vendor and model number	I350 Gigabit Network Connection
Type	Integrated
Driver	Intel Gigabit Ethernet Network Driver: igb, 3.0.6-k
USB ports	
Number	2 external, 1 internal
Type	2.0

Figure 3: Configuration details for the test server.

APPENDIX B – HOW WE TESTED

We used the Dell PowerEdge C6220 server as the server under test and installed CentOS 6.2 on it, as we specify below. For the DVD Store clients, we used eight machines running Windows Server 2003 R2. We cabled all machines into a 1Gbps switch for testing.

Installing the CentOS 6.2 operating system

1. Insert and boot from the CentOS-6.2-x86_64-bin-DVD1 installation DVD.
2. At the welcome screen, select Install or upgrade an existing system, and press Enter.
3. At the Media test screen, select Skip, and press Enter.
4. At the CentOS 6 title screen, click Next.
5. At the Choose an Installation Language screen, select English, and click Next.
6. At the Keyboard Type screen, select U.S. English, and click Next.
7. At the Storage Devices screen, select Basic Storage Devices, and click Next.
8. If a warning for device initialization appears, select Yes, discard any data.
9. At the Name the Computer screen, type the host name, and click Configure Network.
10. At the Network Connections screen, select the server's main or management network interface, and click Edit.
11. At the Editing network interface screen, check Connect Automatically.
12. On the same screen, Select the IPv4 Settings tab, change the Method to Manual, and click Add.
13. On the same screen, enter the IP address, Netmask, Gateway, and DNS server. Click Apply.
14. Click Close on the Network Connections screen, and click Next on the Name the Computer screen.
15. At the Time zone selection screen, select the appropriate time zone, and click Next.
16. Enter the root password in the Root Password and Confirm fields, and click Next.
17. At the Partition selection screen, select Replace Existing Linux System(s), and click Next.
18. If a warning appears, click Write changes to disk.
19. At the default installation screen, click Next to begin the installation.
20. At the Congratulations screen, click Reboot.
21. After the system reboots, log in as root.
22. Disable SELinux by editing the file /etc/selinux/config, and change the line `SELINUX=enforcing` to `SELINUX=disabled`. These changes take effect after rebooting.
23. Update the system using yum:

    ```
    yum update
    ```

24. Disable unused services by running the following command-line script:

    ```
    CHK_OFFs="auditd ip6tables iptables netfs postfix qpidd "
    for i in ${CHK_OFFs}; do
      chkconfig $i off
      service $i stop
    done
    ```

25. Enable and configure the cpuspeed service. Edit the file /etc/sysconfig/cpuspeed and modify the line containing `GOVERNOR` to `GOVERNOR=performance`. Run this command-line script:

    ```
    chkconfig cpuspeed on
    service cpuspeed restart
    ```

26. Configure network adapter connected to the DVD Store clients, for example em2, to listen on four IP addresses (aliases). Modify the file /etc/sysconfig/network-scripts/ifcfg-em2, changing the line ONBOOT=”no” to ONBOOT=yes, and create the file /etc/sysconfig/network-scripts/ifcfg-em2-range0 with this content:

    ifcfg-em2-range0:
    ```
    IPADDR_START=192.168.3.21
    IPADDR_END=192.168.3.24
    CLONENUM_START=1
    ```

27. Allocate 21000 (about 41GB) standard Hugepages for MySQL by modifying the parameter, vm.nr_hugepages, in the file /etc/sysctl.conf:

    ```
    vm.nr_hugepages=21000
    ```

28. Reboot the server:

    ```
    shutdown -r now
    ```

29. After the system reboots, log in as root.

30. Create four pairs of disk partitions (one set for each instance of MySQL) for the MySQL data and log files, using, for example, the disk /dev/sdb, whose contents will be destroyed, and is at least 120GB in size. The size of the partitions is 20GB for the data files, and 5GB for the log files. Run this command-line script:

    ```
    parted /dev/sdb mklabel gpt
    parted /dev/sdb mkpart primary "1    20GB"
    parted /dev/sdb mkpart primary "20GB 40GB"
    parted /dev/sdb mkpart primary "40GB 60GB"
    parted /dev/sdb mkpart primary "60GB 80GB"
    parted /dev/sdb mkpart primary "80GB 85GB"
    parted /dev/sdb mkpart primary "85GB 90GB"
    parted /dev/sdb mkpart primary "90GB 95GB"
    parted /dev/sdb mkpart primary "95GB 100GB"
    parted /dev/sdb name 1 dat01
    parted /dev/sdb name 2 dat02
    parted /dev/sdb name 3 dat03
    parted /dev/sdb name 4 dat04
    parted /dev/sdb name 5 log01
    parted /dev/sdb name 6 log02
    parted /dev/sdb name 7 log03
    parted /dev/sdb name 8 log04
    ```

31. Create one more 20GB disk partition to hold the master DVD Store data, used to reset the system before each benchmark run:

    ```
    parted /dev/sdb mkpart primary "100GB 120GB"
    parted /dev/sdb name 9 ds2
    ```

32. Create an ext4 filesystem on each of the nine partitions:

    ```
    for i in $(seq 19); do
      mkfs.ext4 /dev/sdb${i}
    done
    ```

33. Create mount points for these nine file systems:

```
mkdir /ds2 /mysql
mkdir /mysql/{dat,log}0{1,2,3,4}
```

34. To mount these file systems at boot time, add the following lines to /etc/fstab:

```
/dev/sdb9 /ds2         ext4 defaults,noatime 1 2
/dev/sdb1 /mysql/dat01 ext4 defaults,noatime 1 2
/dev/sdb2 /mysql/dat02 ext4 defaults,noatime 1 2
/dev/sdb3 /mysql/dat03 ext4 defaults,noatime 1 2
/dev/sdb4 /mysql/dat04 ext4 defaults,noatime 1 2
/dev/sdb5 /mysql/log01 ext4 defaults,noatime 1 2
/dev/sdb6 /mysql/log02 ext4 defaults,noatime 1 2
/dev/sdb7 /mysql/log03 ext4 defaults,noatime 1 2
/dev/sdb8 /mysql/log04 ext4 defaults,noatime 1 2
```

35. Mount them for this session as well:

```
mount /ds2 /mysql/dat* /mysql/log*
```

36. Install the software packages for Apache Web Server, PHP, and MySQL database as well their dependencies using yum:

```
yum install @web-server @mysql @mysql-client @php php-mysql
```

Installing Apache Web server and PHP on CentOS 6.2

In our testing, the benchmark used up to four independent DVD Store Web sites. The client machines accessed them through one instance of the Apache Web server configured with four virtual hosts listening on TCP port 80 on four separate IP addresses: 192.168.3.21, 192.168.3.22, 192.168.3.23, and 192.168.3.24.

1. Copy the Apache configuration file /etc/httpd/conf/httpd.conf to /mysql/httpd-template.conf to create a template for the Apache Web server instances.

```
cp /etc/httpd/conf/httpd.conf /mysql/httpd-template.con
```

2. For each row in the next table, replace the line in the Apache configuration template (the entry in the left column) with the contents on the right.

Original Apache Web Server setting	Updated setting
`KeepAlive Off`	`KeepAlive On`
`KeepAliveTimeout 15`	`KeepAliveTimeout 2`
`Listen 80`	`Listen 192.168.1.2XyZZY:80`
`#ServerName www.example.com:80`	`ServerName 191.168.30.2XYZZY`
`AccessFileName .htaccess`	`#AccessFileName .htaccess`
`ErrorLog logs/error_log`	`ErrorLog logs/error0XYZZY_log`
`CustomLog logs/access_log combined`	`CustomLog logs/access0XYZZY_log combined`
`ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"`	`#ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"`

`<Directory "/var/www/cgi-bin">`
`    AllowOverride None`
`    Options None`
`    Order allow,deny`
	`#<Directory "/var/www/cgi-bin">`
	`#    AllowOverride None`
	`#    Options None`
	`#    Order allow,deny`

`    Allow from all`
`</Directory>`
	`#    Allow from all`
	`#</Directory>`

`ReadmeName README.html`
`HeaderName HEADER.html`
	`#ReadmeName README.html`
	`#HeaderName HEADER.html`

`AddType text/html .shtml`
`AddOutputFilter INCLUDES .shtml`
	`#AddType text/html .shtml`
	`#AddOutputFilter INCLUDES .shtml`

3. Insert the following lines at the end of the file /mysql/httpd-template.conf to create the one independent virtual Web sites and to specify how PHP is to connect to the site's MySQL instance:

```
NameVirtualHost 192.168.3.2XYZZY:80

# separate DVD Stores
<VirtualHost 192.168.3.2XYZZY:80>
    DocumentRoot /www/html0XYZZY
    php_value mysql.default_socket  /mysql/log0XYZZY/mysql.sock
    php_value mysqli.default_socket /mysql/log0XYZZY/mysql.sock
</VirtualHost>
```

4. Disable unneeded Apache Web Server extensions:

```
cd /etc/httpd/conf.d
for i in *conf; do
  mv ${i} ${i}-orig
done
mv php.conf-orig php.conf
```

5. Modify the default PHP configuration file /etc/php.ini. For each row in the following table, replace the line in the PHP configuration file (the entry in the left column) with the contents on the right.

Original PHP setting	Updated setting
error_reporting = E_ALL & ~E_DEPRECATED	error_reporting = E_COMPILE_ERROR|E_ERROR|E_CORE_ERROR
date.timezone =	date.timezone ='America/New_York'
display_errors = Off	display_errors = On
mysql.default_host =	mysql.default_host = localhost
mysql.default_user =	mysql.default_user = web
mysql.default_password =	mysql.default_password = web
mysqli.default_port = 3306	mysqli.default_port =
mysqli.default_user =	mysqli.default_user = web
mysqli.default_password =	mysqli.default_password = web

6. Create the directories that will contain the Web content:

```
mkdir /www
mkdir /www/html0{1,2,3,4}
mkdir /www/html0{1,2,3,4/ds2
```

7. Generate the configuration file for each instance by replacing the tag XYZZY with the instance number:

```
for i in 1 2 0 4; do
  sed "s/XYZZY/$i/g" < /mysql/httpd-template.conf > /mysql/h${i}.conf
done
```

8. Stop the default Apache Web instance and prevent it from automatically starting :

```
service httpd stop
chkconfig httpd on
```

Installing MySQL database on CentOS 6.2

The MySQL database software was installed as part of the operating system installation. We configure four independent instances, and start and stop them from scripts rather than use the init/services mechanism.

1. Stop the default MySQL instance and prevent it from automatically starting:

```
service mysqld stop
chkconfig mysqld off
```

2. The configuration file for each instance will generate from a template. Create the file /mysql/my-template.cnf with the content:

```
[mysql]
socket                          = /mysql/logXYZZY/mysql.sock

[mysqld]
user                            = mysql
default_storage_engine          = InnoDB
socket                          = /mysql/logXYZZY/mysql.sock
pid_file                        = /mysql/logXYZZY/mysql.pid
skip-networking
large_pages                     = true
key_buffer_size                 = 32M
myisam_recover                  = FORCE,BACKUP
max_allowed_packet              = 16M
max_connect_errors              = 1000000
innodb                          = FORCE
datadir                         = /mysql/datXYZZY
tmp_table_size                  = 32M
max_heap_table_size             = 32M
query_cache_type                = 0
query_cache_size                = 0
max_connections                 = 20000
thread_cache_size               = 50
open_files_limit                = 65535
table_definition_cache          = 1024
table_open_cache                = 2048
innodb_flush_method             = O_DIRECT
innodb_log_files_in_group       = 2
innodb_log_file_size            = 512M
innodb_flush_log_at_trx_commit  = 2
innodb_file_per_table           = 1
innodb_buffer_pool_size         = 10G
innodb_thread_concurrency       = 0
innodb_log_group_home_dir       = /mysql/logXYZZY
innodb_data_home_dir            = /mysql/datXYZZY
log_error                       = /mysql/logXYZZY/mysql-error.log
log_queries_not_using_indexes   = 1
slow_query_log                  = 1
slow_query_log_file             = /mysql/logXYZZY/mysql-slow.log
```

```
default-storage-engine          = InnoDB
#### from DVD Store installation instructions
ft_min_word_len                 = 3
ft_stopword_file                =
[mysqldadmin]
socket                          = /mysql/logXYZZY/mysql.sock
```

3. Generate the configuration file for each instance by replacing the tag XYZZY with the instance number:

```
for i in 01 02 03 04; do
  sed "s/XYZZY/$i/g" < /mysql/my-template.cnf > /mysql/my-${i}.cnf
done
```

4. Create the script files /mysql/MySQLstart, /mysql/MySQLstop and /mysql/MySQLdata with the following contents:

```
# MySQLstart
for i in 01 02 03 04; do
   echo Starting Instance ${i}
   sleep 5
   mysqld_safe --defaults-file=/mysql/my-${i}.cnf\
       --basedir=/usr --user=mysql --datadir=/mysql/dat${i}\
       --socket=/mysql/log${i}/mysql.sock   >/dev/null 2>&1 &
done
# end of MySQLstart

# MySQLstop
for i in 01 02 03 04; do
   echo Stopping Instance ${i}
   mysqladmin -uroot -pPassword1 \
              -S/mysql/log${i}/mysql.sock shutdown
done
# end of MySQLstop

# MySQL Data
for i in 01 02 03 04; do
   mkdir /mysql/{dat,log}${i} >/dev/null 2>&1
   rm -rf /mysql/{dat,log}${i}/*
   cp -rp /ds2/MySQL-GOLD/dat/* /mysql/dat${i} &
   cp -rp /ds2/MySQL-GOLD/log/* /mysql/log${i} &
   echo Restoring dataset $i
   wait
   chown -R mysql:mysql /mysql/{dat,log}${i}
done
# end of MySQLdata
```

5. Create the default MySQL database and user permissions for the first instance:

```
chown mysql:mysql /mysql/dat* /mysql/log*
mysql_install_db --defaults-file=/mysql/my-01.cnf --basedir=/usr \
   --user=mysql --datadir=/mysql/dat01 --socket=/mysql/log01/mysql.sock
```

6. Start one MySQL server and set the user password:

```
mysqld_safe --defaults-file=/mysql/my-01.cnf --basedir=/usr --user=mysql \
            --datadir=/mysql/dat01 --socket=/mysql/log01/mysql.sock
mysqladmin -S=/mysql/log01/mysql.sock -uroot password Password1
```

7. Create one MySQL user with full permissions for the DVD Store workload by starting a mysql shell with the command, mysql -uroot -pPassword1, and then entering the following commands:

```
grant all privileges on *.* to web@lamp001 identified by 'web';
grant all privileges on *.* to web@localhost identified by 'web';
grant all privileges on *.* to web@'%' identified by 'web';
grant all privileges on *.* to 'apache'@'localhost';
delete from mysql.user where User='';
```

8. Stop the MySQL server

```
mysqladmin -uroot -pPassword1 -S/mysql/log01/mysql.sock shutdown
```

Installing and configuring DVD Store 2.1 on the Linux server

Download the DVD Store 2.1 software from the Dell repository, http://linux.dell.com/dvdstore, to the CentOS server. You need both ds21.tar.gz and ds21_mysql.tar.gz archives.

1. Extract the DVD Store software to the /ds2 file system:

```
cd /ds2
tar zxf ds21.tar.gz
tar zxf ds21_mysql.tar.gz
```

2. Modify the database creation SQL scripts to load the data from the local file system, rather than through the network:

```
cd /ds2/mysqlds2/build
cp mysqlds2_create_db.sql mysqlds2_create_db.sql-orig
sed -i 's/TYPE=/ENGINE=/g ' sql mysqlds2_create_db.sql
cd /ds2/mysqlds2/load
for i in */*.sql; do
   cp ${i} ${i}-orig
   sed -i 's/ LOAD DATA LOCAL INFILE/ LOAD DATA INFILE/' ${i}
done
```

3. Create the master data files for the tests; that is a 10GB DVD Store database for MySQL on Linux:

```
cd /ds2
perl Install_DVDStore.pl
```

4. Start the first MySQL instance and load the DVD Store data:

```
mysqld_safe --defaults-file=/mysql/my-01.cnf --basedir=/usr --user=mysql \
            --datadir=/mysql/dat01 --socket=/mysql/log01/mysql.sock
sleep 10
cd /mysql/mysqlds2
sh mysqlds2_create_all.sh
```

5. Stop MySQL and copy its data and log file to the gold or master file location to speed resetting DVD Store to its original state:

```
sh /mysql/MySQLstop
mkdir /ds2/MySQL-GOLD/dat/
mkdir /ds2/MySQL-GOLD/log
cp -rp /mysql/log01/* /ds2/MySQL-GOLD/log &
cp -rp /mysql/dat01/* /ds2/MySQL-GOLD/dat
```

6. Finally, copy the DVD Store Web and PHP files to the three web-root directories:

```
cd /ds2/mysqlds2/web/php5
cp dsnewcustomer.php.sp dsnewcustomer.php
cp index.html ds*.php ds*.inc /www/html01/ds2
cp index.html ds*.php ds*.inc /www/html02/ds2
cp index.html ds*.php ds*.inc /www/html03/ds2
cp index.html ds*.php ds*.inc /www/html04/ds2
```

Configuring the DVD Store client servers

Each DVD Store client simulates 24 simultaneous users of one DVD Store Web site. In order to generate an appropriate workload, we assigned two clients to each DVD Store Web site, or 48 simultaneous users. Accordingly, we prepared eight client servers, each running 32-bit Microsoft Windows Server 2003 Enterprise with Service Pack 2.

We copied the DVD Store Web client for MySQL, ds2webdriver.exe, to the directory c:\ds2\bin on each client. We created a DVD Store parameter file, params.txt, on each client with the following contents (the target IP address changed to that client's target Web site):

```
target=<target ip address>
n_threads=24
ramp_rate=1
run_time=30
db_size=10GB
warmup_time=10
think_time=0
pct_newcustomers=20
n_searches=3
search_batch_size=5
n_line_items=5
virt_dir=ds2
page_type=php
windows_perf_host=
linux_perf_host=
detailed_view=n
```

Testing procedure

To perform one test, we used a batch file, which we executed from the client machine. We stored the batch file in the C:\ClientShare folder. The testing procedure consisted of the following steps:

1. Execute the batch file.
2. Stop the MySQL services.
3. Delete all prior database files.

4. Copy all original database files from the backup utility partition.
5. Reboot the server under test.
6. Reboot the client machine.
7. Wait for a ping response from the physical server machine.
8. Wait 10 additional minutes for any background tasks to complete.
9. Mount all necessary partitions.
10. Start the MySQL services.
11. Start the workload ramp up period.
12. Stop the workload.
13. Start the workload.
14. Stop the workload.
15. Copy all output files.

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.