Installation, Configuration and Performance Tuning of Shifter V16 on Blue Waters

HonWai Leong, Timothy Bouvet, Brett Bode, Jeremy Enos & David King
Outline

• Background and Challenges
• Installation
• Configuration
• Integration with Workload Manager
• Scaling Performance
• Operational Issues
Background

- Shifter enables execution of container-based applications on HPC systems.
- Shifter Version 1.0 was deployed on Blue Waters since 2016.
- Limited scaling capability of Shifter V1.
- 2016 version (V16) is made available through CLE release 6.0 (but not available to Blue Waters).
- Security concerns and demand for better scalability prompted a need for upgrade on Blue Waters.
Challenges

• Official installation procedure provided by Cray is not designed for CLE 5.2.
• Ongoing integration and testing on boot image require tedious and time consuming efforts.
• “It ain’t broken, don’t fix it” rules of thumb for a five-year-old seasoned system.
Strategy

- Implemented Shifter on /ds1.
- Faster turnaround time in testing and troubleshooting.
- Installed and conducted functional tests on Test and Development System (TDS), then replicated the installation procedure on Blue Waters.
- Installation on /ds1 provides agility for performance tuning on Blue Waters.
Shifter V16 Software Stack

- Shifter 16.08.3
- Squashfs Linux kernel modules
- MongoDB 3.4.7
- Redis 3.2.8
- Python 2.7.13
- Python modules: Celery, PyMongo, Flask, redis, gunicorn
- Munge
Shifter Image Gateway Server

MongoDB  Redis

Shifter-Imagegw (Python)

Munge

Munge  Munge  Munge  Munge  Munge  Munge  Munge

Munge  shifter  Munge  shifter  Munge  shifter  Munge  shifter  Munge  shifter  Munge  shifter

Compute
Installation

- All installations are done in /dsl through xtopview utility on the boot node.

  boot:  ~# xtopview
  default:/ :/  

- Dependencies: *fdupes*, *json*, *squashfs*. Only needed for building the RPM packages of Shifter, not required for Shifter installation.

- Compilation of Shifter software also requires newer version of *Autoconf* and *Automake* tools.
Installation of Shifter 16.08.3

• The Shifter source distribution was cloned from NERSC’s github repository.

    # git clone https://github.com/NERSC/shifter.git shifter-16.08.3

• At the time when we cloned the repository, the source distribution came with GPU support, but it is now removed from the master branch.

• Source modification: Added user definable MountCmd parameter into Udi Root Config.h, Udi Root Config.c and shifter_core.c source files. This parameter is configurable in Udi Root.conf.

• Linux kernel module lookup path: kmodBasePath + `uname -r` + kernel
Installation of Shifter 16.08.3

• RPM spec files:
  a) Shifter binaries
  b) Linux kernel modules: squashfs.ko, loop.ko

• Added “Prefix” macro to the spec files, to make installation of the binaries relocatable.

• We modified the given sample spec file (initially written for CLE6) to build Linux kernel modules for CLE5.
Installation of Shifter 16.08.3

• Build procedure - Shifter RPM packages:
  # pwd
  /software/shifter-16.08.3
  # ./autogen.sh
  # cd ..
  # tar czvf shifter-16.08.3.tar.gz shifter-16.08.3
  # cp shifter-16.08.3.tar.gz /usr/src/packages/SOURCES
  # cd shifter-16.08.3
  # rpmbuild -ba shifter.spec

• Build procedure - Linux kernel modules:
  # cd extra
  # rpmbuild -bb shifter_cle5_gemc_kmod.spec
Installation of Shifter 16.08.3

```bash
# cd /usr/src/packages/RPMS/x86_64
# rpm -ivh --prefix=/opt/cray/shifter/16.08.3 \ 
  shifter_cl_e5_kmod_deps-crmean_gem_c-1.0-3.x86_64.rpm

# rpm -ivh --prefix=/opt/cray/shifter/16.08.3 \ 
  shifter-16.08.3-1.nersc.x86_64.rpm

# rpm -ivh --relocate /usr=/opt/cray/shifter/16.08.3 \ 
  shifter_imagegw-16.08.3-1.nersc.x86_64.rpm

# rpm -ivh --prefix /opt/cray/shifter/16.08.3 \ 
  shifter-runtime-16.08.3-1.nersc.x86_64.rpm
```
Installation of MongoDB

• Download RPM packages from https://repo.mongodb.org.

• Import MongoDB PGP key.

```
# wget --no-check-certificate \
https://www.mongodb.org/static/pgp/server-3.4.asc
# rpm --import server-3.4.asc
```

• Install procedure:

```
# rpm -ivh --prefix=/opt/mongodb/3.4.7 \
  mongodb-org-3.4.7-1.suse11.x86_64.rpm \
  mongodb-org-server-3.4.7-1.suse11.x86_64.rpm \
  mongodb-org-shell-3.4.7-1.suse11.x86_64.rpm \
  mongodb-org-mongos-3.4.7-1.suse11.x86_64.rpm \
  mongodb-org-tools-3.4.7-1.suse11.x86_64.rpm
```
Installation of Redis

• Download and extract source package from http://download.redis.io.

• Install procedure:
  # export CC=gcc
  # make PREFIX=/opt/redis/3.2.8 install
Installation of Python modules

- Installation of Python modules using *pip* tool in a *virtualenv* isolated Python environment.
- *PyPI* enforces SSL enabled client connections (TLSv1.2 is mandatory now).
- Python with SSL support is required.

```
# virtualenv.py /opt/cray/shifter/16.08.3/imagegw_venv 
  --python=/opt/python/2.7.13/bin/python

# source /opt/cray/shifter/16.08.3/imagegw_venv/bin/activate

(image_venv) # pip install \\n  -r /opt/cray/shifter/16.08.3/share/shifter/requirements
```
Munge

- Munge is provided by cray-munge RPM package.
- The PRM package provides a `/etc/munge` key file.
- All compute nodes and the Shifter image manager gateway node use the same key for authentication.
Post Installation

- After exiting from `xtopvi ew`, copy files to Shifter service node persistent `/var` space.

```bash
# cd /rr/current/var/lib
# cp -Rp mongo /snv/ <nid_id>/var/lib
# cd ../log
# cp -Rp mongodb shifter_imagegw* /snv/ <nid_id>/var/log
# cd ../run
# cp -Rp mongodb /snv/ <nid_id>/var/run
```
Configuration

- Shifter image manager gateway: imagemanager.json
- Shifter runtime: udi Root.conf
- Redis: redis.conf
- MongoDB: mongod.conf
- Service init scripts: munge, mongod, redisd, shifter-imagegw
Redis

- `/etc/shifter/redis.conf`

  ```
  dir /var/lib/redis
  requirepass P@55w0rd
  ```

- `/etc/init.d/redisd` is written to read from `/etc/shifter/redis.conf` file.

- Set permission of files to be accessible only by root user.

  ```
  # chown root: /etc/shifter/redis.conf
  # chmod 640 /etc/shifter/redis.conf
  # chown root: /etc/init.d/redisd
  # chmod 750 /etc/shifter/redisd
  ```
MongoDB

- Listen to localhost only (set in `/etc/mongod.conf`).

```javascript
net:
  port: 27017
  bindIp: 127.0.0.1

- Create admin user.

  # mongo
  > use admin
  > > db.createUser(
  ... {
  ... user: "mongodbadmin",
  ... pwd: "<P@5sw0rd>",
  ... roles: [ { role: "root", db: "admin" } ]
  ...
  ...
  )
```
MongoDB

• Restart MongoDB with `--auth` option in init script (`/etc/init.d/mongod`).

```bash
CONFFILE="/etc/mongod.conf"
OPTIONS="-f $CONFFILE --auth "
```

• Create Shifter DB owner.

```bash
# mongo admin -u mongodadmin -p
> use Shifter
switched to db Shifter
> db.createUser(
 ... {
 ... user: "shifteradmin",
 ... pwd: "<P@55w0rd>",
 ... roles: [ { role: "dbOwner", db: "Shifter" } ]
 ... }
 ... )
```
Shifter Image Manager Gateway Init Script

• /etc/init.d/shifter-imagegw

```bash
command=/opt/cray/shifter/16.08.3/sbin/shifter-imagegw
start() {
  lockfile -r 0 $lockfile || {
    echo shifter-imagegw service is already running. && exit -1;
  }
  echo -n "Starting "$util": ">
  ulimit -S -n 8192; ulimit -H -n 16384
  startproc -u shifter $command >> /var/log/$util.log 2>&1
  pidofproc $command > $PIDFILE 2>/dev/null
  rc_status -v
}
```
#!/bin/bash
ROOT_TREE='/opt/cray/shifter/16.08.3'
PYTHON_VENV='imagegw_venv'
SHIFTER_SYSTEM_NAME='bluewaters'
QA="${SHIFTER_SYSTEM_NAME}"
cd ${ROOT_TREE}
source ${PYTHON_VENV}/bin/activate
echo "Starting Celery Queue $QA"
celery -A shifter_imagegw.imageworker worker -Q $QA --loglevel=WARNI NG -n \
worker.queue.$QA -E --concurrency=24 &
echo "Starting imagegw API"
python lib64/shifter/imagegwapi.py &
python lib64/shifter/imagegwapi1.py &
python lib64/shifter/imagegwapi2.py &
wait
Shifter Image Manager

- /etc/shifter/imagemanager.json

  
  "MongoDBURI": "mongodb://shifteradmin:<P@55w0rd>@localhost/Shifter?authMechanism=SCRAM-SHA-1",

  "MongoDB": "Shifter",

  "Broker": "redis://:<P@55w0rd>@localhost/",

- Owned by “shifter” user. Accessible by “shifter” and root user only.
- Specialized to utility class service nodes.
Shifter Runtime

• /etc/shifter/udi Root.conf
  udi Mbunt =/var/udi Mbunt
  loopMbunt =/var/udi LoopMbunt
  udi Root Path=/opt/cray/shifter/16.08.3
  sitePreHookMbunt=/opt/cray/shifter/16.08.3/sbin/premount.sh
  mountCmd=/opt/cray/shifter/16.08.3/lib64/shifter/mount
  kmodBasePath=/opt/cray/shifter/16.08.3/modules
  siteResources=/opt/shifter/site-resources
  allowLibcPwdCalls=1
```
#!/bin/bash

# Create directories
mkdir -p mnt/a
mkdir -p mnt/b
mkdir -p mnt/c

# Mount directories
mount --bind /mnt/a mnt/a
mount --bind /mnt/b mnt/b
mount --bind /mnt/c mnt/c

# Create additional directories
mkdir -p ufs
mount --bind /ufs ufs
mkdir -p var/opt/cray/alps
mount --bind /var/opt/cray/alps var/opt/cray/alps
mkdir -p dsl/opt
mount --bind /dsl/opt dsl/opt

# Create symbolic links
ln -s mnt/a/u u
ln -s mnt/b/projects projects
ln -s mnt/c/scratch scratch
```
Services

- Start services
  
  # service munge start
  # service mongod start
  # service redis start
  # service shifter-imagegw start
Integration with Moab/Torque

```
qsub -l nodes=1:ppn=32,gres=shifter16 -v UDI=<image:tag> jobscript.sh
```

```
export CRAY_ROOTFS=SHIFTER
aprun -b -- command
```

```
qsub -l nodes=1:ppn=32,gres=shifter16 jobscript.sh ➔ aprun -b -- shifter --image=<image:tag> -- command
```
Shifter Job Submission

• Invoking “shifter” command in job script.

```bash
#!/bin/bash

#PBS -l nodes=1:ppn=32
#PBS -l gres=shifter16

module load shifter/16.08.3

cd $PBS_O_WORKDIR

module load shifter/16.08.3

aprun -n 1 -b -- shifterimg pull <image:tag>
aprun -n 1 -b -- shifter --image=<image:tag> -- command

$ qsub jobscript.sh
```
Shifter Job Submission

- Invoking “set upRoot” command through Torque prologue script.

```bash
$ cat jobscript.sh
#!/bin/bash

#PBS -l nodes=1:ppn=32
#PBS -l gres=shifter16
#PBS -v UDI=<image:tag>

cd $PBS_O_WORKDIR

export CRAY_ROOTFS=SHIFTER

aprun -n 1 -b -- command

$ qsub jobscript.sh
```
Prologue

- `/var/spool/torque/mom_priv/prologue`

```bash
if [ $(qstat -f ${BATCH_JOB_ID} | grep Resource_List.gres | grep -c '\bshifter16\b') -gt 0 ]; then
    shifter_prologue=/opt/cray/shifter/16.08.3/wlm/torque/cray-shifter-prologue
    if [[ -x $shifter_prologue ]]; then
        $shifter_prologue ${BATCH_JOB_ID} $USER $GROUP ${RESV_ID} ${NIDS}
    fi
fi
```

**Diagram**

- `prologue (mom node)`
- `gres=shifter16`
- `cray-shifter-prologue (mom node)`
- `start (pcmd)`
- `cray-shifter-extra-service (compute node)`
- `udiRoot-prologue (mom node)`
- `if UDI=<image:tag>`
- `setupRoot (compute node)`

Presentation Title
cray-shifter-prologue

- Start munge and nscl services on compute nodes (invoking cray-shifter-extra-service start)
- If UDI is defined in job submission, execute setupRoot on compute nodes to mount UDI (invoking udi Root - prologue).
Epilogue

• /var/spool/torque/mom_priv/epilogue

if [ $(qstat -f ${BATCH_JOB_ID} | grep Resource_List.gres | grep -c ^\bshifter16\b) -gt 0 ]; then
  shifter_epilogue=/opt/cray/shifter/16.08.3/wlm/torque/cray-shifter-epilogue
  if [[ -x $shifter_epilogue ]]; then
    $shifter_epilogue ${BATCH_JOB_ID} $USER $GROUP ${RESV_ID} ${NIDS}
  fi
fi

epilogue
(mom node)

---

cray-shifter-epilogue
(mom node)

gres=shifter16

---

stop
(pcmd)

---

cray-shifter-extra-service
(compute node)

---

udiRoot-epilogue
(mom node)

---

if UDI=<image:tag>

---

unsetupRoot
(compute node)

---

pcmd
cray-shifter-epilogue

• Stop munge and nsed services on compute nodes (invoking cray-shifter-extra-service stop)
• If UDI is defined in job submission, execute unset upRoot on compute nodes to un-mount UDI (invoking udi Root - epilogue).
Compute Node Root Runtime Environment (CNRTE)

• Shifter is installed in /ds1 and executes from /ds1.
• In udi Root.conf, udi Mbunt = /var/udi Mbunt
• Absolute root path as seen from compute node => /ds1/var/udi Mbunt
• /etc/opt/cray/cnrte/roots.conf
  
  SHIFTER=/ds1/var/udi Mbunt
• When using setupRoot in job prologue and setting CRAY_ROOTFS=SHIFTER in job script, alps tasks land on /ds1/var/udi Mbunt.
shifter vs. setupRoot (80 nodes)

```bash
# Shifter Command
aprun -n 1280 -N 16 -- shifter --image=centos:latest -- cat /etc/centos-release

# Prologue Command
aprun -n 1280 -N 16 -- cat /etc/centos-release
```

![Shifter Image Start-Up Time Graph](image.png)

- CLI
- Prologue
Initial Scaling Test Results - shifter

$ qsub -l nodes=4096:ppn=16,gres=shifter16

> aprun -n 4096 -N 1 -b -- shifter --image=centos:latest -- cat /etc/centos-release

• 2000+ nodes success.
  ➢ FAILED to lookup docker image
  ➢ Failed to lookup username or attempted to run as root
Initial Scaling Test Results - \texttt{setupRoot}

$\texttt{qsub -l\ nodes=2048:\ ppn=16,\ gres=\text{shifter16}\ -v\ UDI=\text{centos:latest,\ CRAY\_ROOTFS=\text{SHIFTER}}}$

$\texttt{aprun -n\ 2048\ -N\ 1\ -b\ --\ cat\ /etc/\text{centos\_release}}$

- 700+ nodes success.
  - \texttt{FAILED to get groups correctly}
  - \texttt{FAILED to lookup auxiliary gids. Exiting}
Username, groups, gids related issues

• Scalability issue with LDAP.
• Too many query requests to LDAP server.
• Workarounds:
  - Start NSCD service on compute nodes => random caching
  - Perform "id $USER" on compute nodes => taking too long
Resolution

i. Perform “id $USER” on mom node.

ii. Copy /var/run/ncsd/passwd and /var/run/ncsd/group files from the mom node to a cluster shared location.

shifter-crroy-prologue

id $USER

cp /var/ncsd/run/ncsd/passwd /scratch/system/shifter/jobs/passwd.$JOBID

cp /var/ncsd/run/ncsd/group /scratch/system/shifter/jobs/group.$JOBID

pcmd -r -q -n $NIDS "/dsl/usr/bin/chroot /dsl cray-shifter-extra-service start $JOBID"
iii. On each compute node, copy nscd passwd and group files from cluster shared location to local `/var/run/nscd` directory.

iv. Start nscd service on compute node.

```bash
cray-shifter-extra-service start $JOBID

cp /scratch/system/shifter/jobs/passwd.$JOBID /var/run/nscd

cp /scratch/system/shifter/jobs/group.$JOBID /var/run/nscd
/etc/init.d/nscd start
```
Failed to lookup docker image

- Too many query requests sent to Shifter image manager gateway.
- Six celery threads were started by default.
- imagegwapi.py listen to port 5000.

```
shifter-imagegw

celery -A shifter_imagegw.imageworker worker -Q $QA -n worker.queue.$QA &
python imagegwapi.py
```
Workarounds

• Increase number of threads to 24.
• Duplicate `imagegwapi.py`, listening to port 5000, 5001 and 5002.

```
shifter-imagegw

celery -A shifter_imagegw.imageworker worker -Q $QA -n worker.queue.$QA -E concurrency=24 &
python imagegwapi.py &
Python imagegwapi1.py &
python imagegwapi2.py &
```
Improved Scaling Performance

- 4096 nodes using `shifter` CLI.
- 2048 nodes using `setupRoot` through job prologue.
- `setupRoot` is limited to 2048 nodes due to prologue timeout duration set at 300 seconds.
aprung -N >1

$ qsub -l nodes=256:ppn=16,gres=shifter16

> aprun -n 4096 -N 16 -b -- shifter --image=centos:latest -- cat /etc/centos-release

- Preload loop.ko and squashfs.ko in job prologue.
- Set max_loop=128

cray-shifter-extra-service start $JOBID

/sbin/insmod $KMODPATH/drivers/block/loop.ko max_loop=128
/sbin/insmod $KMODPATH/fs/squashfs/squashfs.ko
Encoding and Decoding Issues

```
import sys
reload(sys)
sys.setdefaultencoding('utf8')
```

```
@@ -625,6 +625,9 @@
tfp = tar_file_refs[layer_idx]
    members = layer_paths[layer_idx]
    
    # Change encoding to 'utf8' to take care of unicode character in file paths.
+    base_path = base_path.encode('utf8')
    tfp.extractall(path=base_path, members=members)
```
Untracked process in SSH session

```
user@mom ~> cat .shifter/config
Host * 
Port 1204
IdentityFile ~/.shifter/id_rsa
StrictHostKeyChecking no
UserKnownHostsFile /dev/null
LogLevel error

user@mom ~> ssh -F .shifter/config nidxxxxx
```

- Background/daemon processes are left running on compute nodes even after job ends.
- An epilogue script is written to cleanup these stray processes
/etc/shifter/shifter_etc_files

- Files under /etc/shifter/shifter_etc_files directory are copied from host into container.
- passwd, group, nsswitch.conf
- Some applications (e.g. spark) validates user/group information of execution user before launching.
- A cron script is written to update the passwd and group files in the directory regularly.
Conclusions

• Lots of works needed to maintain Shifter software stack.
• To do:
  ➢ Use distributed MongoDB servers for scale-out performance.
  ➢ Use multiple service nodes to host Shifter image manager gateways.
• Shifter on Blue Waters provides an platform for researchers to develop and test container-based applications, in preparation for next generation HPC systems.
Acknowledgement

- US National Science Foundation (awards OCI-0725070 and ACI-1238993)
- US state of Illinois.
- University of Illinois at Urbana-Champaign
- Mr. Mark Dalton of Cray Inc.
- Shifter open source community