Docker task in HPC Pack

We introduced docker task in HPC Pack 2016 Update1.
To use this feature, set the environment variable CCP_DOCKER_IMAGE of a task so that it could be run in a docker container on Linux compute node.

Environment variable CCP_DOCKER_IMAGE indicates the docker image to use by this task, the format is like running a docker command in Linux:
CCP_DOCKER_IMAGE=[Docker Registry]<Repository>[:Tag]

Besides, there are several environment variables could be used to enhance this feature.

- CCP_DOCKER_NVIDIA to indicate if using command 'nvidia-docker', instead of using 'docker', to start docker container.
- CCP_DOCKER_VOLUMES to set the directories to be mounted from host to docker container as volumes.
- CCP_DOCKER_DEBUG to indicate if leaving the container alive for debugging after the command in it finishes, the container needs to be removed manually later.

❖ To run docker task, docker should be installed on Linux compute nodes first.

❖ Currently, the docker image used in HPC Pack should have /bin/bash, or the docker task would fail. We will try to remove this restriction later.

❖ A docker task can be allocated with multiple nodes to run MPI application, nevertheless, one node should not be allocated to multiple docker tasks.

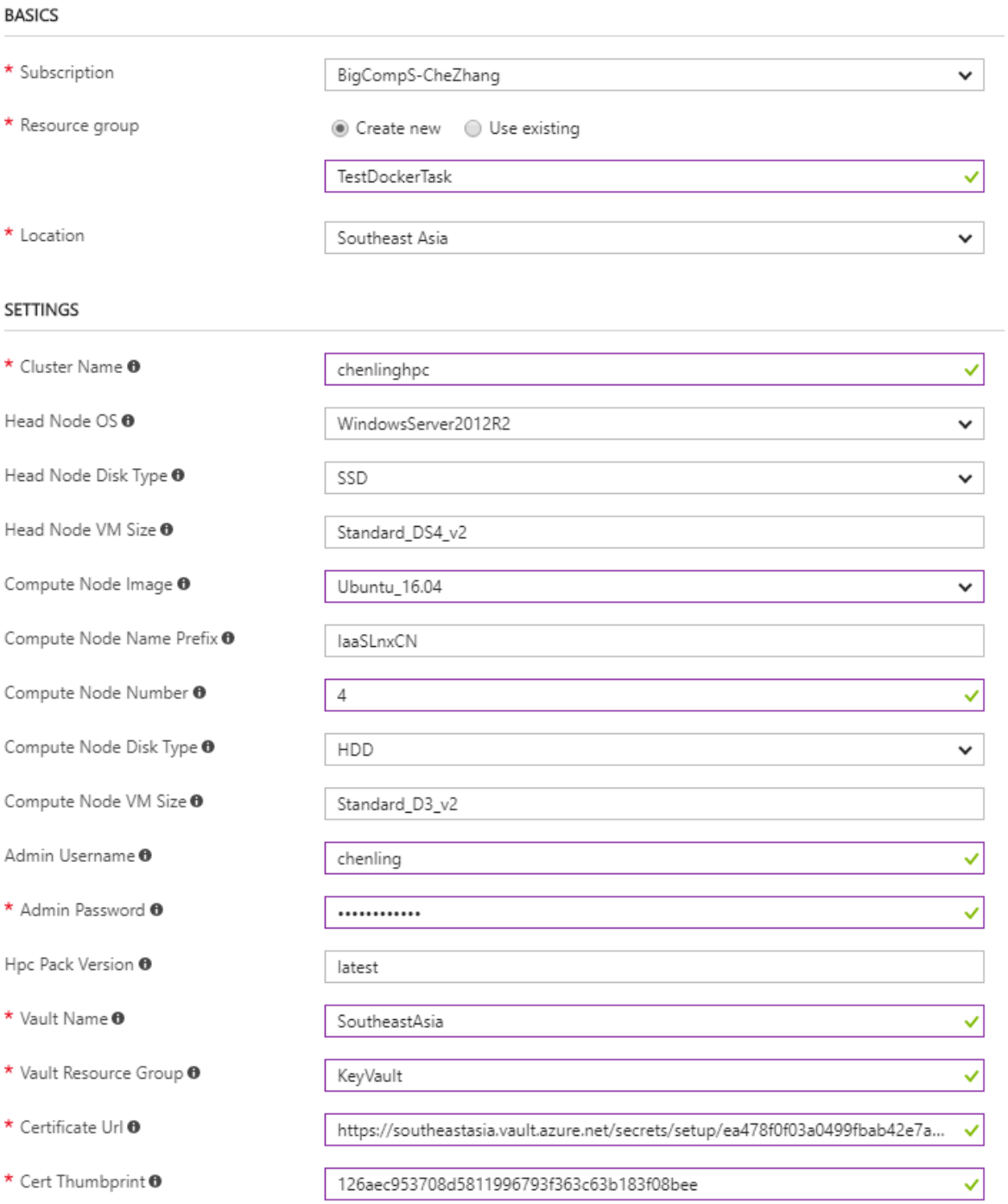
❖ To run MPI application in Linux nodes and docker containers in HPC Pack, a SSH key pair should be prepared and configured to Linux nodes for setting the mutual trust between them. The key pair could be generated by ssh-keygen on a Linux node or by puttygen.exe(PuTTY Key Generator) on a windows node.

❖ To run MPI application in docker containers on Linux nodes, the docker image should have sudo, ssh service and MPI installed.

Run MPI in docker container step by step

1. Deploy cluster with ARM template

Use the template "NOT Active Directory Domain integrated - Single head node cluster for Linux workloads"
Use WindowsServer2012R2 as Head Node OS and use Ubuntu_16.04 as Compute Node Image.



2. Install docker on Linux compute nodes by clusrun

Connect to the cluster, open Command Prompt and type below command:

```
clusrun /nodegroup:linuxnodes apt update^; apt -y install docker.io
```

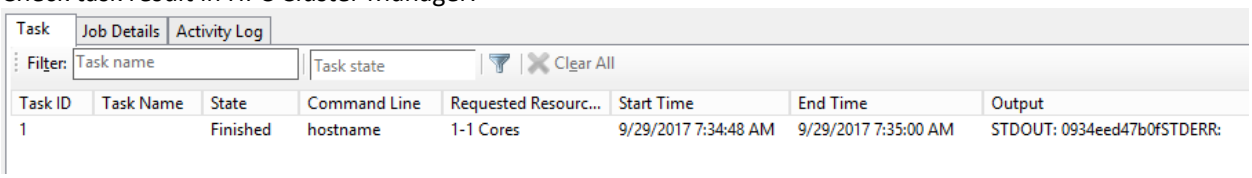
3. Run command in container as docker task

- ### Submit a job containing 1 docker task:

Take the Linux compute nodes online and type below command in Command Prompt:

```
job submit /env:ccp_docker_image=docker.io/library/ubuntu:16.04 hostname
```

Check task result in HPC Cluster Manager:

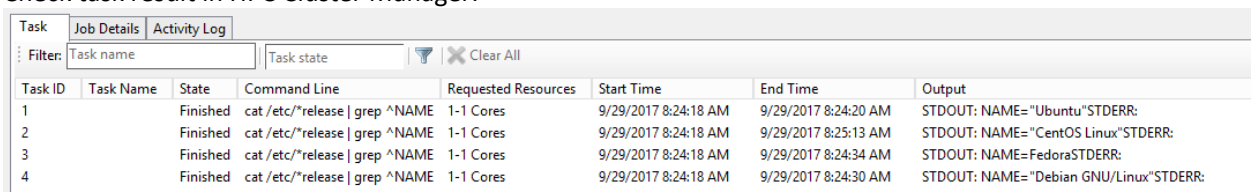


- ### Submit a job containing multiple docker tasks:

Type below command in Command Prompt:

```
job new
job add !! /env:ccp_docker_image=ubuntu cat /etc/*release ^| grep ^^NAME
job add !! /env:ccp_docker_image=centos cat /etc/*release ^| grep ^^NAME
job add !! /env:ccp_docker_image=debian cat /etc/*release ^| grep ^^NAME
job add !! /env:ccp_docker_image=fedora cat /etc/*release ^| grep ^^NAME
job submit /id:!!
```

Check task result in HPC Cluster Manager:

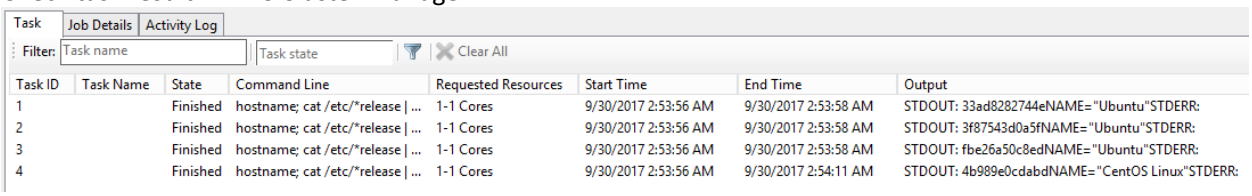


❖ Tasks would inherit the environment variables of their job if they don't have the same ones, thus the docker image can also be assigned in job environment variables.

Type below command in Command Prompt:

```
job new /jobenv:ccp_docker_image=ubuntu
job add !! hostname^; cat /etc/*release ^| grep ^^NAME
job add !! hostname^; cat /etc/*release ^| grep ^^NAME
job add !! hostname^; cat /etc/*release ^| grep ^^NAME
job add !! /env:ccp_docker_image=centos hostname^; cat /etc/*release ^| grep ^^NAME
job submit /id:!!
```

Check task result in HPC Cluster Manager:



Task ID	Task Name	State	Command Line	Requested Resources	Start Time	End Time	Output
1		Finished	hostname; cat /etc/*release | ...	1-1 Cores	9/30/2017 2:53:56 AM	9/30/2017 2:53:58 AM	STDOUT: 33ad8282744eNAME="Ubuntu"STDERR:
2		Finished	hostname; cat /etc/*release | ...	1-1 Cores	9/30/2017 2:53:56 AM	9/30/2017 2:53:58 AM	STDOUT: 3f87543d0a5fNAME="Ubuntu"STDERR:
3		Finished	hostname; cat /etc/*release | ...	1-1 Cores	9/30/2017 2:53:56 AM	9/30/2017 2:53:58 AM	STDOUT: fbe26a50c8edNAME="Ubuntu"STDERR:
4		Finished	hostname; cat /etc/*release | ...	1-1 Cores	9/30/2017 2:53:56 AM	9/30/2017 2:54:11 AM	STDOUT: 4b989e0cdabdNAME="CentOS Linux"STDERR:

4. Run MPI task in container as docker task

• Build customized docker image with mpich installed

❖ Perform this step in any Linux node with docker installed.

Start a container with docker image ubuntu:

```
docker run -it ubuntu
```

Install sudo, ssh, vim and mpich with apt-get:

```
apt update; apt -y install sudo ssh vim mpich
```

Write a simple mpi program:

```
mkdir /mpisample
chmod o+w /mpisample
cd /mpisample
vim helloMpi.c
```

Edit helloMpi.c with below content:

```
#include<mpi.h>
#include<stdio.h>

int main(int argc, char** argv)
{
    int rank, size, processor_name_length;
    char processor_name[1000];
    MPI_Init(NULL, NULL);
    MPI_Comm_rank (MPI_COMM_WORLD, &rank);
    MPI_Comm_size (MPI_COMM_WORLD, &size);
    MPI_Get_processor_name(processor_name, &processor_name_length);
    printf("Hello from %s, rank %d out of %d processors.\n", processor_name, rank, size);
    MPI_Finalize();
}
```

Compile helloMpi.c with mpicc and create a shell script run.sh:

```
mpicc helloMpi.c -o helloMpi
vim run.sh
```

Edit run.sh with below content:

```
#!/bin/bash
echo $CCP_NODES | tr " " "\n" | sed "1d;n;d" | cat > host_file
```

```
num=$1
[ -z "$num" ] && num=4
mpirun -n $num -f host_file ./helloMpi
```

Set the execution permission of run.sh and exit the docker container:

```
chmod +x run.sh
exit
```

❖ A docker hub account is needed to perform this operation.
Commit and push the docker image to docker hub:

```
docker commit $(docker ps -qa -n 1) <docker hub account>/mpich
docker login -u <docker hub account> -p <password>
docker push <docker hub account>/mpich
```

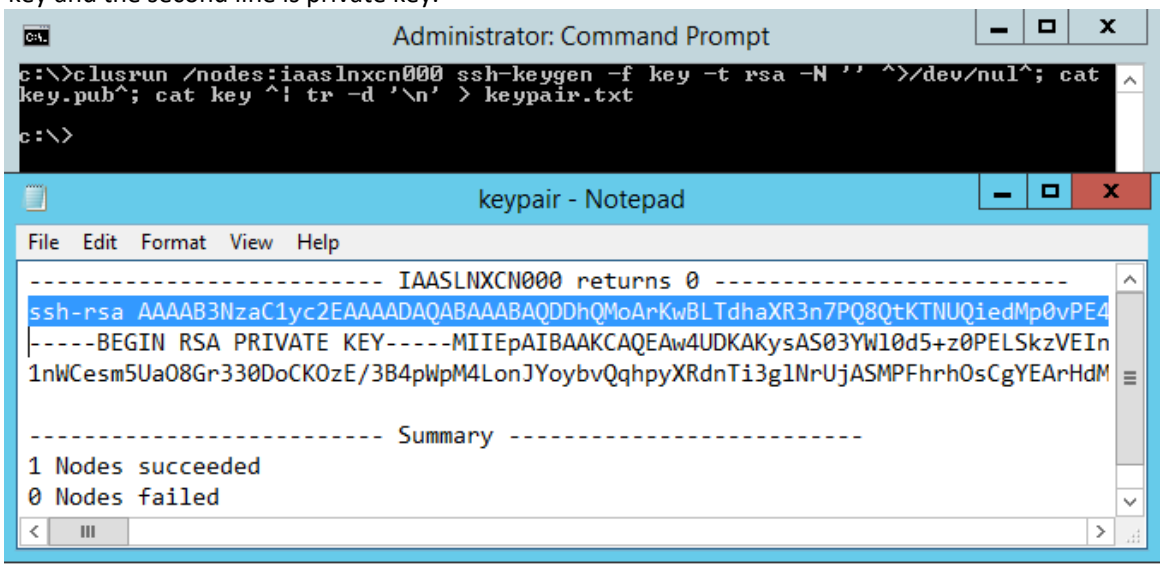
❖ This is a way to build docker image manually, alternative is using Dockerfile.

• Run MPI task as root

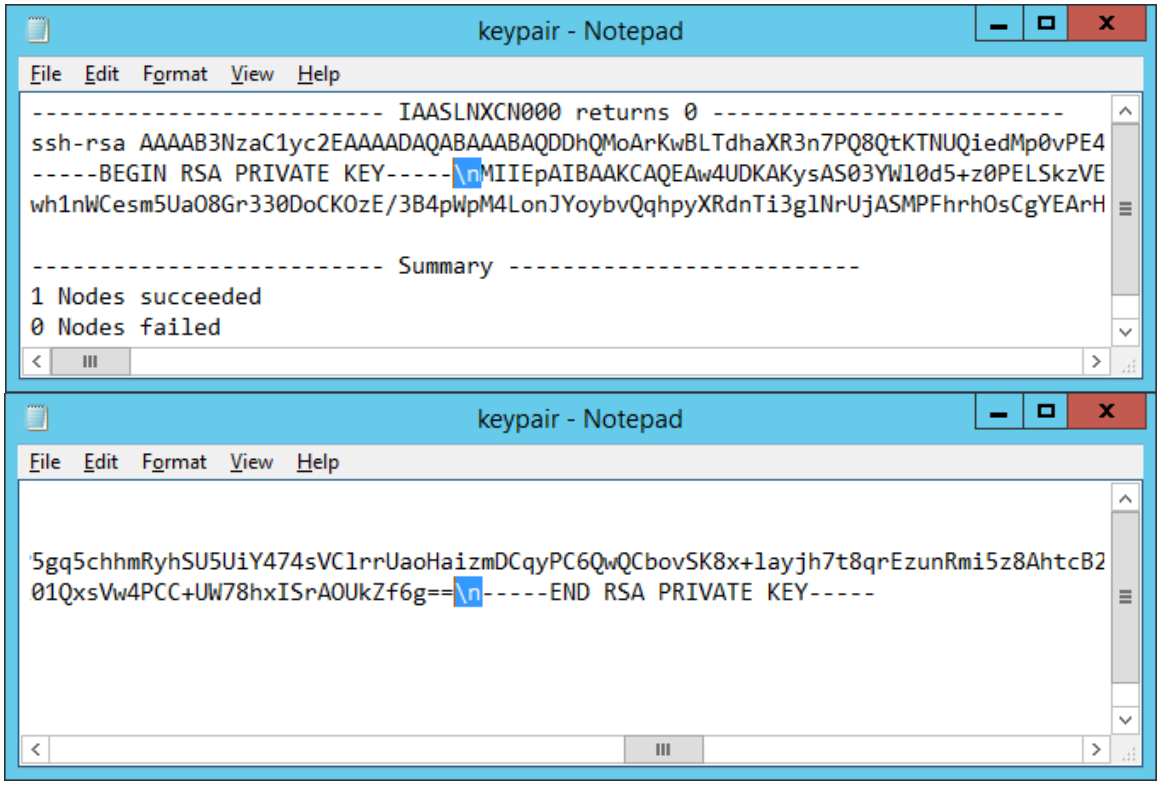
Generate key pair by command ssh-keygen on Linux node with clusrun.
Open a Command Prompt and run below command:

```
clusrun /nodes:iaaslnxcn000 ssh-keygen -f key -t rsa -N '' ^>/dev/nul^; cat key.pub^; cat key ^| tr -d '\n' > keypair.txt
```

A key pair will be generated and stored in the file keypair.txt, the first line of clusrun result is public key and the second line is private key.



Insert 2 newline characters '\n' in public key, one after "-----BEGIN RSA PRIVATE KEY-----" and the other before "-----END RSA PRIVATE KEY-----".



Set mutual trust between Linux compute nodes for root.
Creating ssh key file id_rsa with clusrun in Command Prompt:

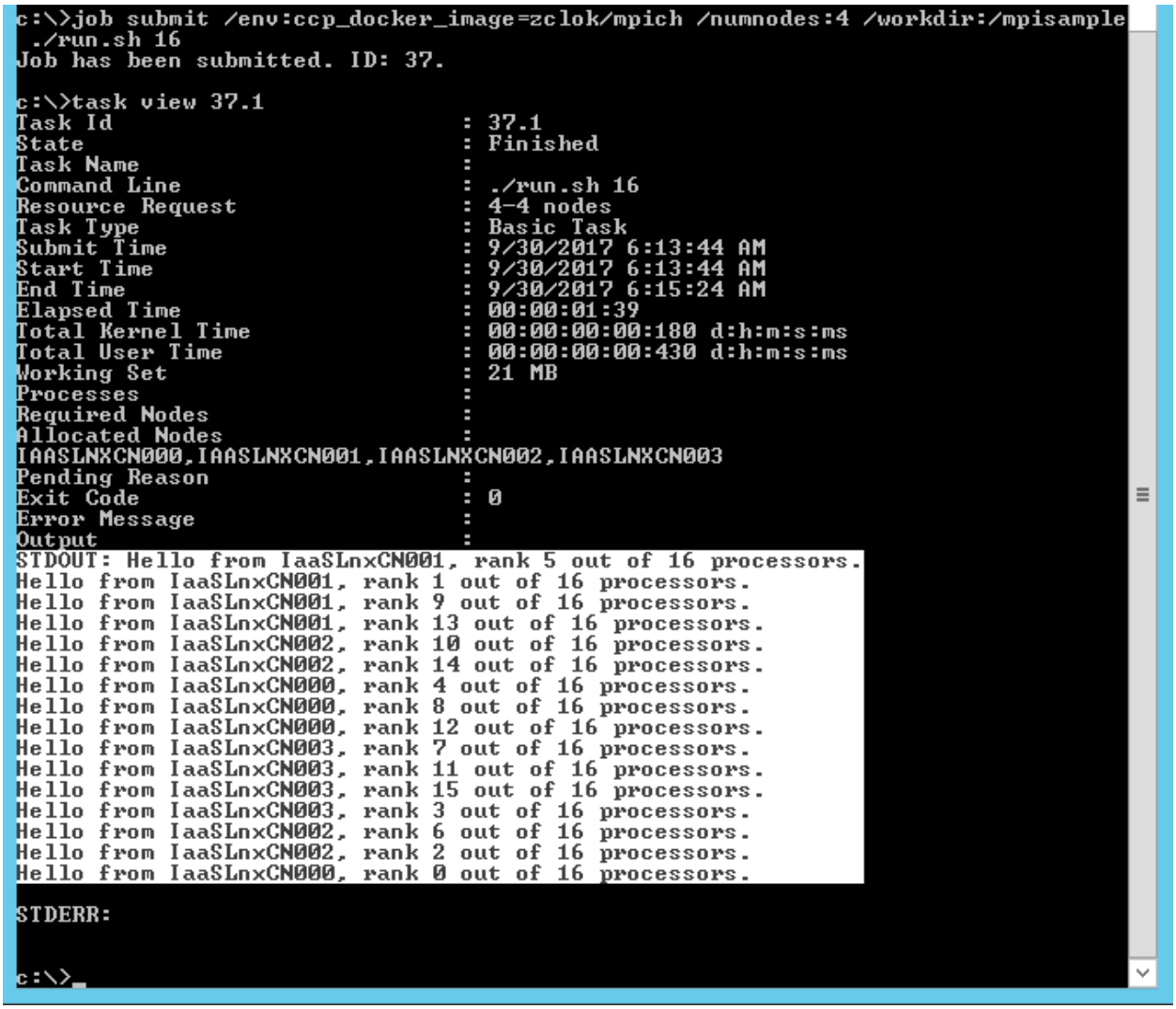
```
clusrun /nodegroup:linuxnodes mkdir -p /root/.ssh
clusrun /nodegroup:linuxnodes echo '<public key>' ^> /root/.ssh/authorized_keys
clusrun /nodegroup:linuxnodes echo -e '<private key>' ^> /root/.ssh/id_rsa
clusrun /nodegroup:linuxnodes chmod 600 /root/.ssh/id_rsa
```

- The mutual trust could also be set by other ways like login Linux compute nodes with putty and configuring manually. For normal user, a credential file containing the key pair could be passed with command hpccred so that the mutual trust could be set automatically by HPC Pack.

Submit a job including a docker task to run the MPI application in the docker image we built in above step:

```
job submit /env:ccp_docker_image=<docker hub account>/mpich /numnodes:4
/workdir:/mpisample ./run.sh 16
```

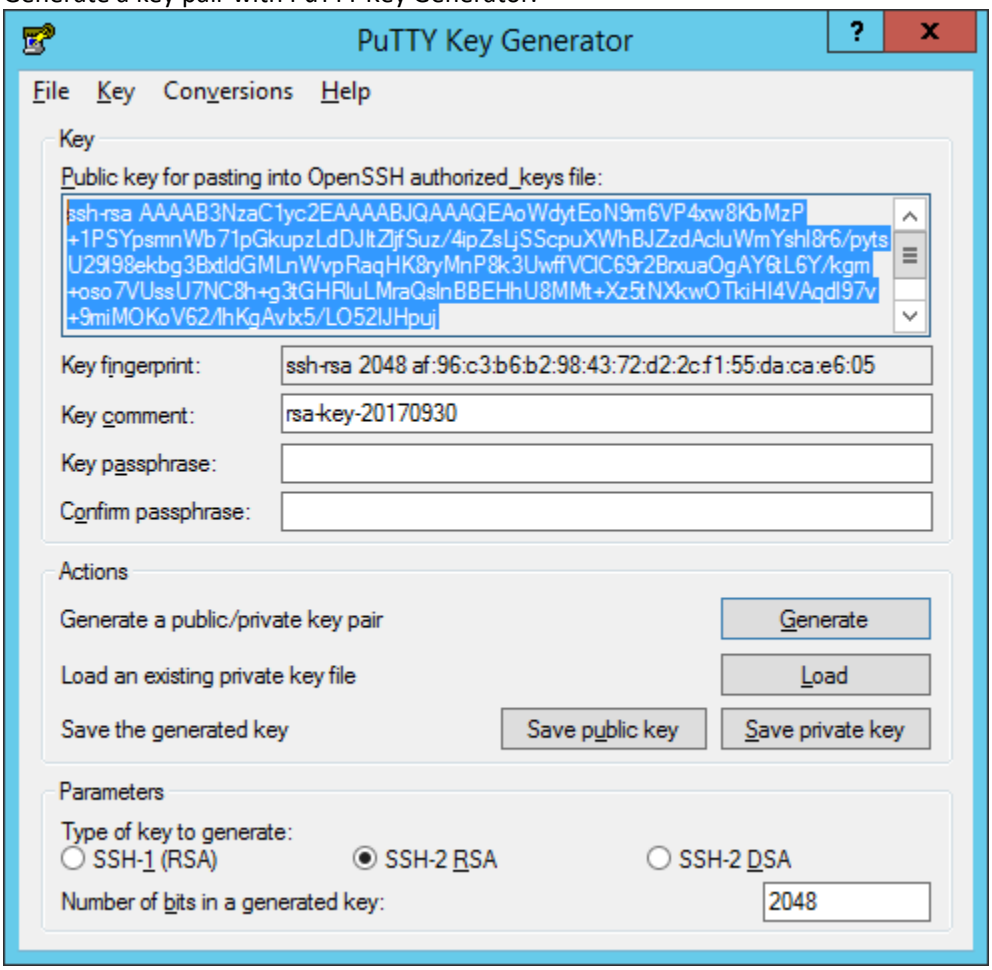

Select Administrator: Command Prompt
c:\>clusrun /nodes:iaaslnxcn000 ssh-keygen -f key -t rsa -N '' ^>/dev/nul^; cat key.pub^; cat key ^| tr -d '\n' > keypair.txt
c:\>clusrun /nodegroup:linuxnodes mkdir -p /root/.ssh
IAASLNXCN002 returns 0
IAASLNXCN003 returns 0
IAASLNXCN000 returns 0
IAASLNXCN001 returns 0
Summary
4 Nodes succeeded
0 Nodes failed
c:\>clusrun /nodegroup:linuxnodes echo 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDD
hQMoArKwBLTdhaXR3n7PQ8QtKTNUQiedMp0vPE4DkPSmMMFW/REZgo1yyh73BUIq3AULdmDEUu0gScAG
/ppmfKhjT1tLKrPbvwax93+4kZ/oFwIsKfeNducDduFq1lLbc8dpdXqo1df0gLpiTU+Hx06TBMzwQ9fH
QeAtyrmFeklfOcUZXFb3n56NHgwuB0a8WPsmzwdRHiWzFR9Ah6moQh4PpbGe10+6tSWqkXYSp5flSVJX
dQReso+ebWEVmLfM7VZlpcEMlJso/jKH4hyU1tVpcyeghMhBF64KbMB4XAFZCv9C2iMp321bIyI7iQSU
JpqD6yH/eNa9cCWX2eS3 root@IaaSLnxCN000' ^> /root/.ssh/authorized_keys
IAASLNXCN000 returns 0
IAASLNXCN001 returns 0
IAASLNXCN003 returns 0
IAASLNXCN002 returns 0
Summary
4 Nodes succeeded
0 Nodes failed
c:\>clusrun /nodegroup:linuxnodes echo -e '-----BEGIN RSA PRIVATE KEY-----\nMIIE
pAIBAAKCAQEAw4UDKAKysAS03YW10d5+z0PELSkzVEInnTKdLzxOA5D0pjDBVv0RGYKNcsoe9wVCKtwF
S3ZgxFbtIEnABv6aZnyoY09bSyqz278Gsfd/uJGf6BcCLCn3jXbnA3bhatZS23PHaXV6qNXX9IC6Yk1P
h8dOkwTM8EPXx0HgLcq5hXpJXznFGUxW95+ejR4MLgdGvFj7Js8HUR4lsxUfQIepqEIeD6WxnpdPurUl
qpF2EqeX5U1SV3UEXrKPnm1hFZi3zO1WZaXBDJSbKP4yh+IclJbVaXMnoITIQReuCmzAeFwBWQr/Qtoj
Kd9tWyMiO4kE1Caag+sh/3jWvXAll9nktwIDAQABAoIBADEjgdqRawVP+YDfMQCg8a/3sRtEipD3m2sn
qFeRyW2eBc6q3DyX16+YnyIlaEje+nU5HN+NXd6aljgSMP/RB08Qr2VeMWyfy9pDlUQvBtuFRfIXrkUZ
Ow1Dv+WgACFQwjxDSe/YRb6fGApsO+uu0Hkofzf18efEtTtv8/6uP5gq5chhmRyhSU5UiY474sVClrrU
aoHaizmDCqyPC6QwQCbovSK8x+layjh7t8qrEzunRmi5z8AhtcB2OhPcAObs8h+sW7ksZIvs7vfzRAsI
c8b0Vhp0q4TxjUvz0gyywlbedW+K5F/14AjTs/t0ajmQdErYYqzQyM4fS2xAA34SKFkCgYEA9bXA+K5C
5EbAzeDe0Wr1JFPcYjPg1385xWNTp4pfGGOohZiQLt2Ymdr1SEn2LatpVidsM7rsOPLoWTEMTLo1UR7C
alsEsUIg7gfvFM1LDc5pOQig0uttYqTPJJPFNKb6mmNTEBvb1K28UKt0Y0171topmcAArBbygjdGXHJ+
XGUCgYEAy7UrX7H7f6jGckGCjze+LlKdvoq/2tQ0KxmwFo1R86Y5UydmKoRmRsm6IdIFpDeFGTD7vxKi
aRDN+yWeThUza7Tb5eDhWYT0Oszwh1nWCesm5UaO8Gr330DoCKOzE/3B4pWpM4LonJYoybvQqhpyXRdn
Ti3g1NrUjASMPFhrhOsCgYEArHdMIQGoQDUNhjUOEW53s10Otvt07MHHmi5S/KmagDMLieKBafPu9id4
gpovqvTDNk7rmw/cWUjfICPWX1DgtbTif+atT4J8PLfi7/MutF0q7HaK9CF+IRxPrXvvkzqVcqK/HfJ/
2UdthC4kvsMXOdaeFqZ3klN1G0OTJrbSzSUCgYEAgGTZSMCqrwTWx48I1nIg+Zty93UGhcu4ce4H8bxt
ne+INCQiHexDdpyRJ93zV1T0HDOmiQEawrQn0AMe60IrndhpvVNnVgAZ27HubLEv2iKP6nT1hQ023E1R
sphTN+d6O/Fuqxv4tzFh1fz+ASkYKhq1Lz51Tf07eGEpbqOgaS8CgYACm6+4AoAiYRbYcOsduXSWnPco
DSFmcVzZGQc6nkh/33L4PG8XmnDOm0DSMC49mKUENIGCBONuVcoLYIvTNj4PERbJWVCcqY30AdivzCzn
0IFJ8/t//KxtSVerHj35po2GJwtWKab69HHNz01QxsVw4PCC+UW78hxISrAOUkZf6g==\n-----END R
SA PRIVATE KEY-----' ^> /root/.ssh/id_rsa
IAASLNXCN000 returns 0
IAASLNXCN002 returns 0
IAASLNXCN003 returns 0
IAASLNXCN001 returns 0
Summary
4 Nodes succeeded
0 Nodes failed
c:\>clusrun /nodegroup:linuxnodes chmod 600 /root/.ssh/id_rsa
IAASLNXCN003 returns 0
IAASLNXCN000 returns 0
IAASLNXCN001 returns 0
IAASLNXCN002 returns 0
Summary
4 Nodes succeeded
0 Nodes failed
c:\>job submit /env:ccp_docker_image=zclok/mpich /numnodes:4 /workdir:/mpisample
./run.sh 16
Job has been submitted. ID: 37.
c:\>task view 37.1


c:\>job submit /env:ccp_docker_image=zclok/mpich /numnodes:4 /workdir:/mpisample
./run.sh 16
Job has been submitted. ID: 37.
c:\>task view 37.1
Task Id : 37.1
State : Finished
Task Name :
Command Line : ./run.sh 16
Resource Request : 4-4 nodes
Task Type : Basic Task
Submit Time : 9/30/2017 6:13:44 AM
Start Time : 9/30/2017 6:13:44 AM
End Time : 9/30/2017 6:15:24 AM
Elapsed Time : 00:00:01:39
Total Kernel Time : 00:00:00:00:180 d:h:m:s:ms
Total User Time : 00:00:00:00:430 d:h:m:s:ms
Working Set : 21 MB
Processes :
Required Nodes :
Allocated Nodes :
IAASLNXCN000,IAASLNXCN001,IAASLNXCN002,IAASLNXCN003
Pending Reason :
Exit Code : 0
Error Message :
Output :
STDOUT: Hello from IaaSLnxCN001, rank 5 out of 16 processors.
Hello from IaaSLnxCN001, rank 1 out of 16 processors.
Hello from IaaSLnxCN001, rank 9 out of 16 processors.
Hello from IaaSLnxCN001, rank 13 out of 16 processors.
Hello from IaaSLnxCN002, rank 10 out of 16 processors.
Hello from IaaSLnxCN002, rank 14 out of 16 processors.
Hello from IaaSLnxCN000, rank 4 out of 16 processors.
Hello from IaaSLnxCN000, rank 8 out of 16 processors.
Hello from IaaSLnxCN000, rank 12 out of 16 processors.
Hello from IaaSLnxCN003, rank 7 out of 16 processors.
Hello from IaaSLnxCN003, rank 11 out of 16 processors.
Hello from IaaSLnxCN003, rank 15 out of 16 processors.
Hello from IaaSLnxCN003, rank 3 out of 16 processors.
Hello from IaaSLnxCN002, rank 6 out of 16 processors.
Hello from IaaSLnxCN002, rank 2 out of 16 processors.
Hello from IaaSLnxCN000, rank 0 out of 16 processors.
STDERR:
c:\>

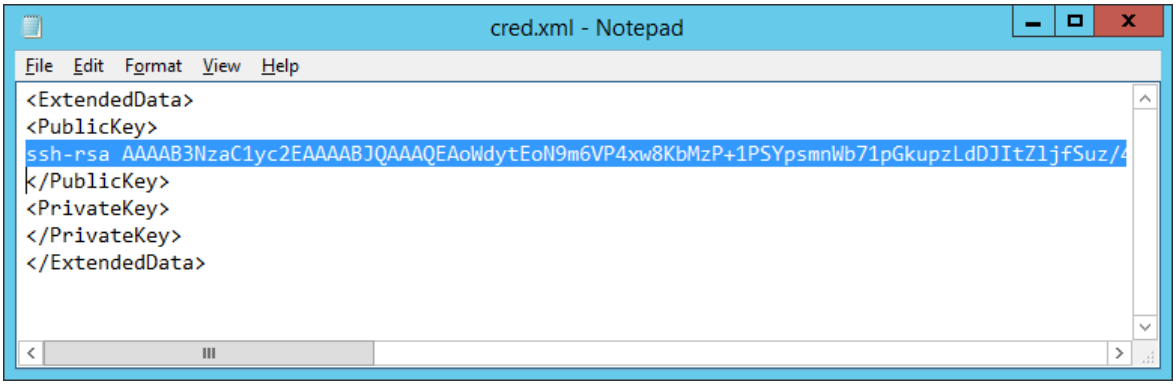
- ## Run MPI task as a normal user

create a normal user, like “user1”.

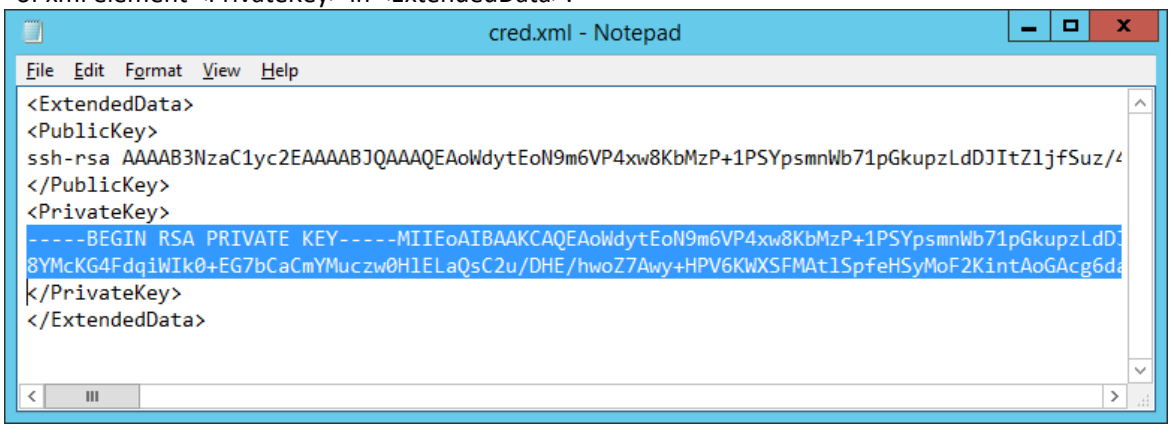
Generate a key pair with PuTTY Key Generator.



Create a xml file named “cred.xml” in C:\.
Copy public key from the window to file “cred.xml” as the content of xml element <PublicKey> in <ExtendedData>.



Export OpenSSH Key in PuTTY Key Generator as private key, and copy it to “cred.xml” as the content of xml element <PrivateKey> in <ExtendedData>.



Set credential for normal user ‘user1’ in Command Prompt:

```
hpccred setcreds /user:user1 /extendeddata:C:\cred.xml
```

Submit job to run MPI application in docker task as user1:

```
job submit /env:ccp_docker_image=<docker hub account>/mpich /numnodes:4 /user:user1 /workdir:/mpisample ./run.sh
```