White Paper

LexisNexis® Risk Solutions
Using HtS3 to Deploy HPCC Systems® and Save and Restore Files

HPCC Systems: See Through Patterns in Big Data to Find Big Opportunities

Table of Contents

Introduction

HtS3 gives you flexibility in configuring and deploying HPCC Systems® from LexisNexis® to Amazon Web Services (AWS) without requiring you to know a lot more than you currently do. Plus, it enables you to save and restore THOR files directly from the THOR's nodes to/from S3 buckets. And, all saves and restores are done in parallel thereby the copies of THOR files to and from S3 buckets is faster than despraying them to the landing zone and then copying them from there to S3 buckets.

The main purpose of this document is to describe how to setup and use HtS3 to configure and deploy an HPCC System to AWS; as well as save and restore files on the deployed HPCC's THOR cluster. For those who want to know a little about the software, go to Appendix B where you will find a table that lists the scripts that make-up HtS3 and HPCC Charm along with a brief description of each script file.

What is HPCC Systems?

HPCC (High Performance Computing Cluster) Systems is a massive parallel-processing computing platform that solves Big Data problems. The platform is Open Source!

The HPCC Systems architecture incorporates the Thor (data refinement) and Roxie (data delivery) clusters as well as common middleware components, an external communications layer, client interfaces which provide both end-user services and system management tools, and auxiliary components to support monitoring and to facilitate loading and storing of file system data from external sources. An HPCC environment can include only Thor clusters, or both Thor and Roxie clusters.

What is Juju Charm?

HtS3 uses Juju Charm and the HPCC Charm to configure and deploy an HPCC System to AWS.

Juju is a next generation service orchestration framework. It has been likened to APT for the cloud. With Juju, different authors are able to create service formulas, called charms, independently, and make those services coordinate their communication and configuration through a simple protocol.

After the setup, the only part of Juju that you will interact the file ~/.juju/environments.yaml which sets cloud provider specific parameters.

⚠ This presentation assumes that you will run HtS3 and Juju charm on an Ubuntu 12.04 linux machine. If you don't have an Ubuntu 12.04 linux machine then you can download and install a VMware image of one and run it on a VMware Player, which can also be installed. You will find instructions in Appendix A.

About This Document

In what follows, commands given in an Ubuntu terminal window are shown with a gray background, like the following example:

```
sudo apt-get install git
```

In addition for screenshots, I will mark lines being discussed with a light blue arrow, like this one. →

When you are done with your deployed HPCC System shut it down so AWS charges don't continue to accumulate.

As you use this document to configure and deploy an HPCC System to AWS, if you get errors that I haven't talked about here, please take a screenshot showing the error and email it to me: timothy.humphrey@lexisnexis.com

Setup for Using HtS3 on Ubuntu 12.04 Linux

HtS3 uses the HPCC Juju Charm to configure and deploy an HPCC System to AWS. Also, HtS3 uses s3cmd to save and restore files to/from S3 buckets.

The following describes how to install and configure: 1) HtS3 with HPCC Charm, 2) Juju, and 3) s3cmd.

Installing HtS3 with the HPCC Charm

If you don't have git installed on your Ubuntu linux machine then the following command does that.

```
sudo apt-get install git
```

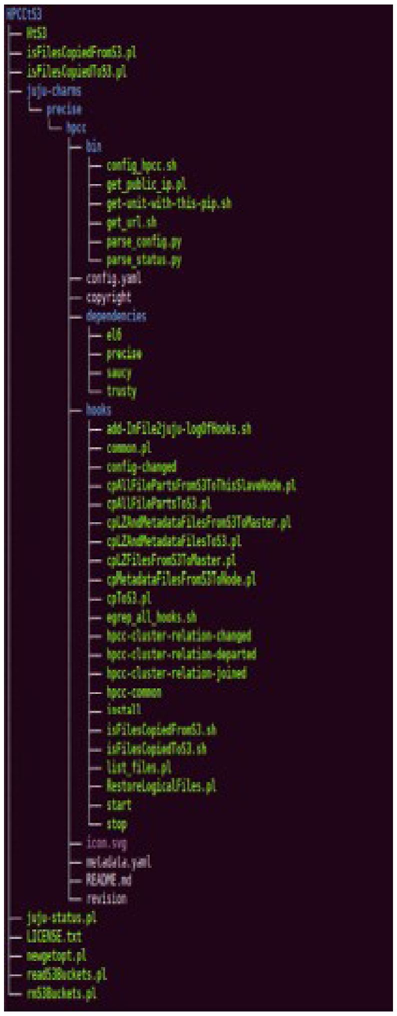
After entering a sudo command, you will be asked to enter a password. If you are using the vmware ubuntu image of Appendix A then enter the word "password" (without quotes). Plus for the above command, there may be other prompts you need to respond to during the install.

This software runs only on Ubuntu 12.04 Linux. If you don't have access to Ubuntu, follow the instructions of Appendix A to build a virtual Ubuntu 12.04 machine.

Next install the HtS3 scripts and the HPCC Charm by cloning it from github.com. From your home directory, enter the following command.

```
git clone https://github.com/tlhumphrey2/HPCCtS3.git
```

Here is a screenshot of the directory structure created by the above clone (contents of the directory HPCCtS3 created by the above command).



```
HPCCtS3
├── HtS3
├── isFilesCopiedFromS3.pl
├── isFilesCopiedToS3.pl
├── juju-charms
│   └── precise
│       └── hpcc
│           ├── bin
│           │   ├── config_hpcc.sh
│           │   ├── get_public_ip.pl
│           │   ├── get-unit-with-this-pip.sh
│           │   ├── get_url.sh
│           │   ├── parse_config.py
│           │   └── parse_status.py
│           ├── config.yaml
│           ├── copyright
│           ├── dependencies
│           │   ├── el6
│           │   ├── precise
│           │   ├── saucy
│           │   └── trusty
│           ├── hooks
│           │   ├── add-InFile2juju-LogOfHooks.sh
│           │   ├── common.pl
│           │   ├── config-changed
│           │   ├── cpAllFilePartsFromS3ToThisSlaveNode.pl
│           │   ├── cpAllFilePartsToS3.pl
│           │   ├── cpLZAndMetadataFilesFromS3ToMaster.pl
│           │   ├── cpLZAndMetadataFilesToS3.pl
│           │   ├── cpLZFilesFromS3ToMaster.pl
│           │   ├── cpMetadataFilesFromS3ToNode.pl
│           │   ├── cpToS3.pl
│           │   ├── egrep_all_hooks.sh
│           │   ├── hpcc-cluster-relation-changed
│           │   ├── hpcc-cluster-relation-departed
│           │   ├── hpcc-cluster-relation-joined
│           │   ├── hpcc-common
│           │   ├── install
│           │   ├── isFilesCopiedFromS3.sh
│           │   ├── isFilesCopiedToS3.sh
│           │   ├── list_files.pl
│           │   ├── RestoreLogicalFiles.pl
│           │   ├── start
│           │   └── stop
│           ├── icon.svg
│           ├── metadata.yaml
│           ├── README.md
│           └── revision
├── juju-status.pl
├── LICENSE.txt
├── newgetopt.pl
├── readS3Buckets.pl
└── rmS3Buckets.pl
```

The command, tree Juju-charms, will create the above directory structure. If you don't have tree then you can download it with the following command:

```
sudo apt-get install tree
```

Installing and Configuring Juju

Installing Juju

You must install juju to use HtS3. To install Juju, you simply need to grab the latest juju-core package from the PPA, update the packages on your Ubuntu system and apt-get install juju-core and juju-local. Note. Juju-local is installed because it loads python modules we need.

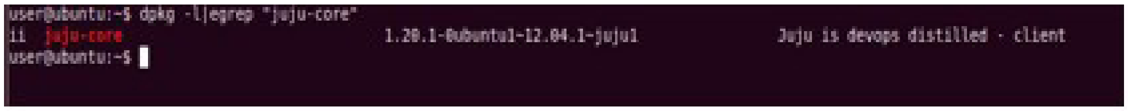
```
sudo add-apt-repository ppa:juju/stable
sudo apt-get update
sudo apt-get install juju-core
sudo apt-get install juju-local
```

The installation of juju-core and juju-local will take a few minutes and generate a lot of output lines. Also, you are asked if you want to continue. So, type in Y.

You can check if juju-core was installed by doing the following command:

```
dpkg -l|egrep "juju-core"
```

Here is a screenshot of what you should see:



```
user@ubuntu:~$ dpkg -l|egrep "juju-core"
ii  juju-core                1.20.1-0ubuntu1~12.04.1~juju1          Juju is devops distilled - client
user@ubuntu:~$
```

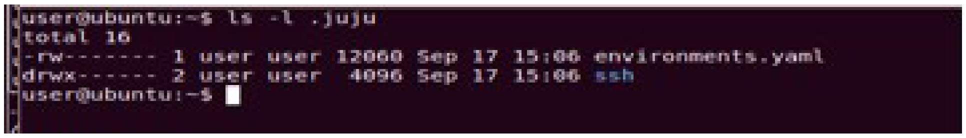
Configuring Juju

To configure juju, you only need to generate and edit the environments.yaml file, which will live in your ~/.juju directory.

To generate an initial environments.yaml file, you simply need to run:

```
juju generate-config
```

The directory, .juju, created by the above command is shown to the following screenshot.



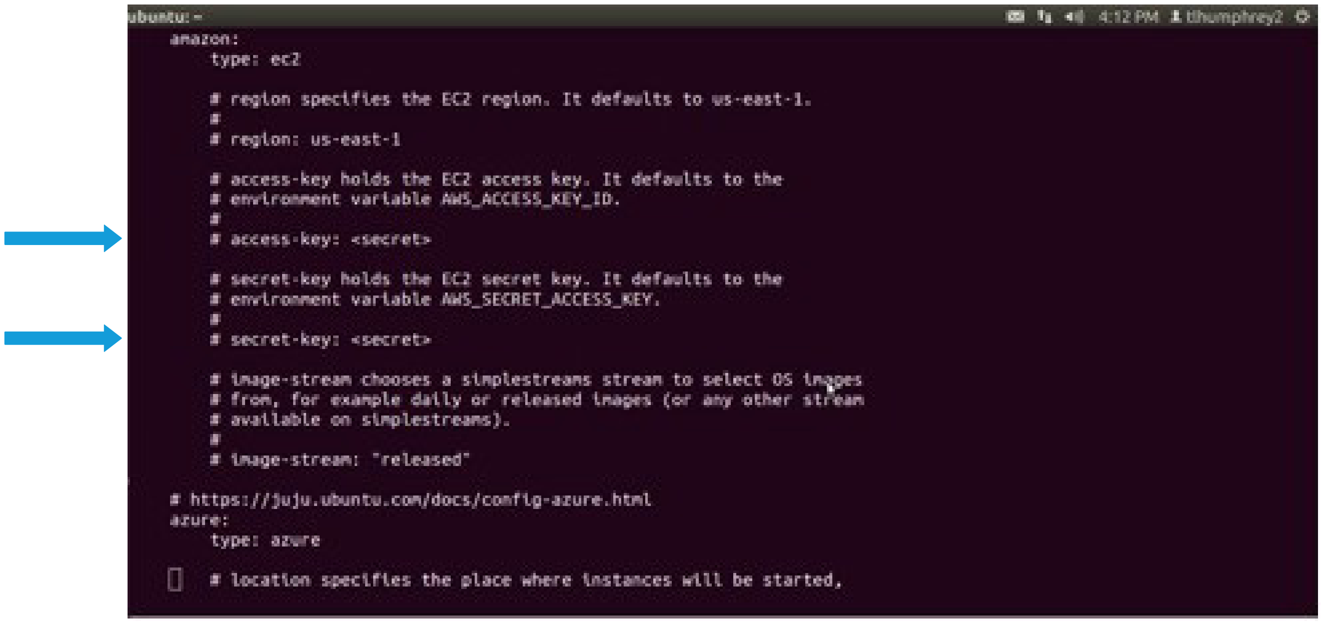
```
user@ubuntu:~$ ls -l .juju
total 16
-rw------- 1 user user 12060 Sep 17 15:06 environments.yaml
drwx------ 2 user user  4096 Sep 17 15:06 ssh
user@ubuntu:~$
```

You will notice that this directory contains the file, environments.yaml, and a sub-directory, ssh. Public and secret keys are in the ssh sub-directory. These keys are used by Juju Charm to access the AWS EC2 instances it deploys via ssh. Plus, if you use 'juju ssh' to access these instances, these same keys are used.

The environments.yaml file contains sample profiles for different types of cloud services (including Amazon's). You need to edit the file to provide specific information for Amazon's. For now, you only need to make two changes to the file -- insert your AWS access key and secret key. So, to bring up environments.yaml in vi, do the following:

```
vi ~/.juju/environments.yaml
```

Here, is a screenshot of environments.yaml in vi. It is opened to the amazon section where you will insert your AWS access key and secret key (where the green arrows are pointing). For both lines, you replace <secret> with your aws access-key and secret-key.



```
    amazon:
        type: ec2

        # region specifies the EC2 region. It defaults to us-east-1.
        #
        # region: us-east-1

        # access-key holds the EC2 access key. It defaults to the
        # environment variable AWS_ACCESS_KEY_ID.
        #
        # access-key: <secret>

        # secret-key holds the EC2 secret key. It defaults to the
        # environment variable AWS_SECRET_ACCESS_KEY.
        #
        # secret-key: <secret>

        # image-stream chooses a simplestreams stream to select OS images
        # from, for example daily or released images (or any other stream
        # available on simplestreams).
        #
        # image-stream: "released"

    # https://juju.ubuntu.com/docs/config-azure.html
    azure:
        type: azure

        # location specifies the place where instances will be started,
```

In the above screenshot there is also a place where you can change the AWS region that you HPCC System is deployed. It is currently commented out, i.e. the line with "region: us-east-1" on it. If you want to change the region, this is the place to do it.

Installing and Configuring s3cmd

To install s3cmd, execute the following command.

```
sudo apt-get install s3cmd
```

Since, the above command begins with 'sudo', it will ask you for the password of root. If you used the procedure in Appendix A to download an vmware image of ubuntu 12.04, the password is 'password' (without the quotes).

Once, s3cmd is install, do the following command to configure it. This command creates a the file, .s3cfg, in your home directory.

```
s3cmd --configure
```

The above command will prompt you for information. You will use the defaults supplied accept for those prompts given in the following table.

Prompt	Your Response
Access key	Enter your AWS access key
Secret key	Enter your AWS secret key
Encryption password	Enter password IF you want file transfers encrypted
Test access with supplied credentials	Enter yes (this assures you didn't make any mistakes when entering your access and secret keys)
Save settings	Enter yes (this saves your settings to ~/.s3cfg)

If you change your AWS access and secret keys, after doing 's3cmd –configure', you must redo 's3cmd –configure'. Just changing the keys in the file, .s3cfg, DOES NOT WORK.

Once, .s3cfg is created, do the following command to copy it to the HPCC Charm's hooks directory.

```
cp ~/.s3cfg  ~/HPCCtS3/juju-charms/precise/hpcc/hooks
```

The reason for this copy is so juju will copy .s3cfg along with all other files of the hooks directory to each node of the THOR (the copying of these files is part of juju's deployment process). If this wasn't done, then these nodes could not copy from/to S3 buckets.

You have completed the setup process.

Using HtS3 to Configure & Deploy an HPCC System to AWS and Save and Restore Files

The tasks in the following table are what will be discussed below. The table shows normal sequences of tasks that are done to a) deploy an HPCC System to AWS and b) to save and restore files of the deployed HPCC System. Plus, the table shows where errors might occur.

Normal Sequence of Tasks			
Seqno	Task	Task Type	Task Description
1	bootstrap	HtS3 & juju command	Brings up a small EC2 instance used as the juju server.
2	deploy	HtS3 & juju command	Deploys the desired HPCC System to AWS.
3	expose	HtS3 & juju command	Exposes the deployed HPCC's public address so one can use it to display its ECL Watch.
4	thor_info	HtS3 command	Get the HPCC's ESP unit-id, private IP and public id addresses.
5	get files onto deployed thor	ECL Watch function	To demonstrate that we can copy files of the THOR to S3 buckets, we use ECL Watch to upload a file to the drop zone and then spray it to the nodes of the THOR.
6	cp2s3	HtS3 command	Copies sprayed files and drop zone files to S3 buckets.
7	readS3Buckets.pl	script	Used to check to see if the files got saved to the S3 buckets.
8	destroy	HtS3 & juju command	Shuts down deployed HPCC System

Later			
Seqno	Task	Task Type	Task Description
9	restore_hpcc		Deploy another HPCC System to AWS.
10	cpfs3	HtS3 command	Copy files in S3 buckets to newly deployed HPCC System
11	look at files on thor	ECL Watch function	Use ECL Watch to see if the files were copied to the HPCC's THOR and drop zone.

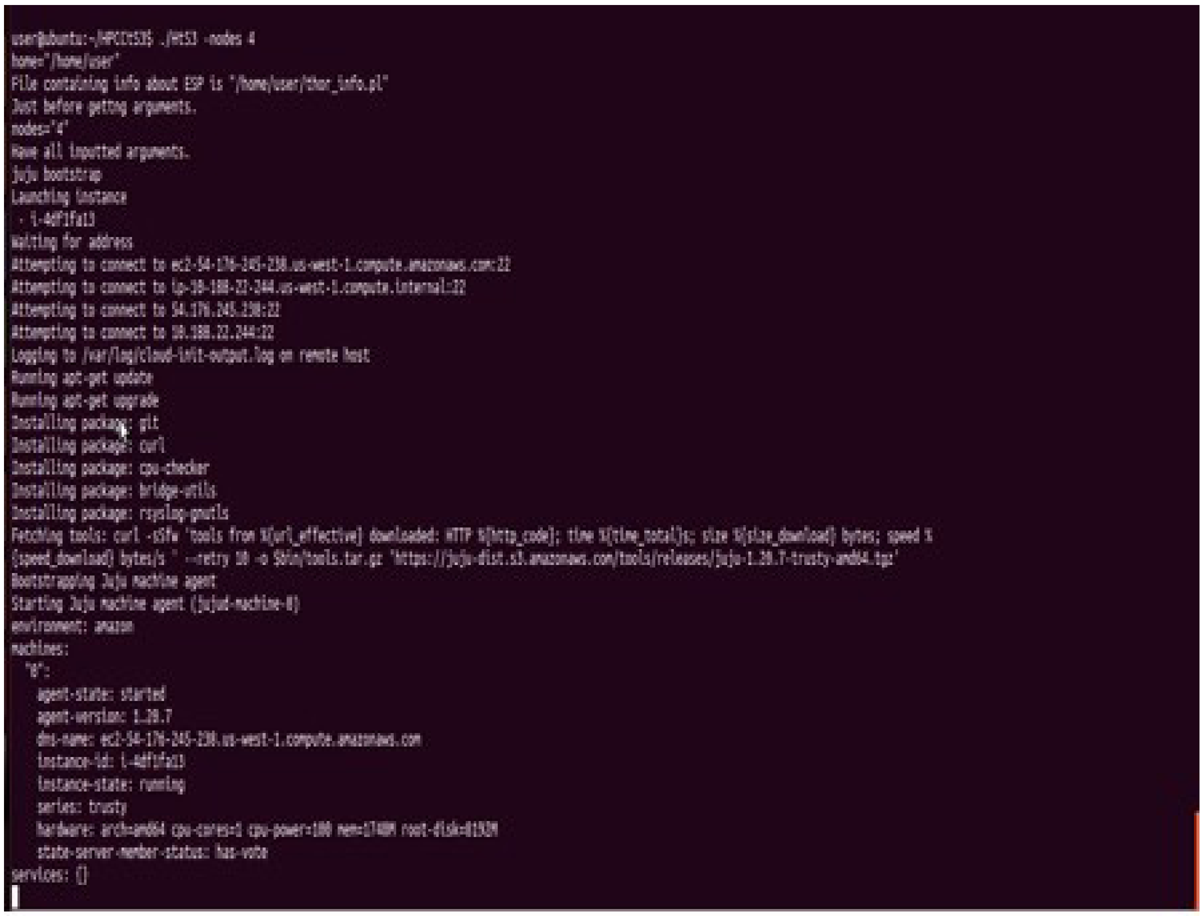
Handling Errors/Problems			
Seqno	Task	Task Type	Task Description
	bootstrap or deploy		Normally, these are the only juju performed tasks that have errors.
	thor_info		When thor_info is executed too soon after all units have the state 'started', you might get errors.

Enter the following HtS3 command from ~/HPCCtS3 directory. This command configures and deploys an HPCC System whose THOR has 4 nodes (1 master and 3 slaves) as well as a ROXIE with 3 nodes.

```
./HtS3 -nodes 4
```

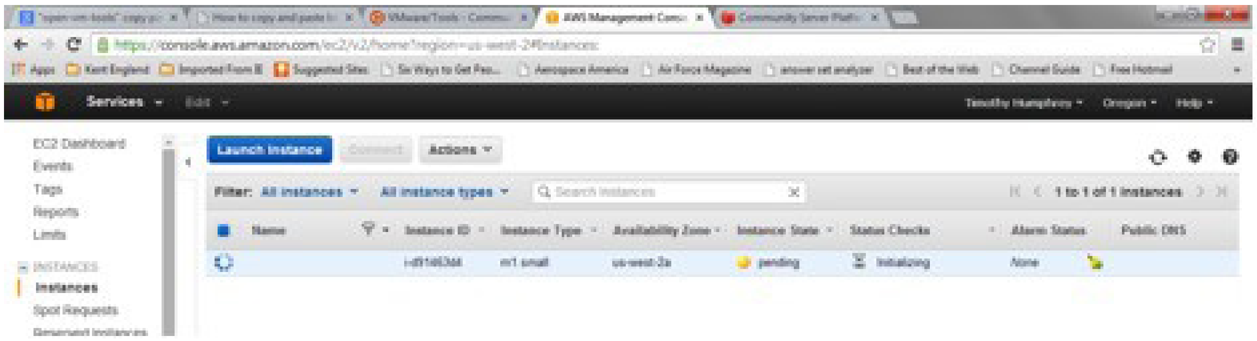
1. bootstrap

As shown in the following terminal window screenshot, the first thing HtS3 does is have juju create a bootstrap instance (the top green arrow points to where that begins).



```
user@ubuntu:~/HPCCtS3$ ./HtS3 -nodes 4
home="/home/user"
File containing info about ESP is "/home/user/thor_info.pl"
Just before getting arguments.
nodes="4"
Have all inputted arguments.
juju bootstrap
Launching instance
 - i-4df1fa13
Waiting for address
Attempting to connect to ec2-54-176-245-238.us-west-1.compute.amazonaws.com:22
Attempting to connect to ip-10-188-22-244.us-west-1.compute.internal:22
Attempting to connect to 54.176.245.238:22
Attempting to connect to 10.188.22.244:22
Logging to /var/log/cloud-init-output.log on remote host
Running apt-get update
Running apt-get upgrade
Installing package: git
Installing package: curl
Installing package: cpu-checker
Installing package: bridge-utils
Installing package: rsyslog-gnutls
Fetching tools: curl -sSfw 'tools from %{url_effective} downloaded: HTTP %{http_code}; time %{time_total}s; size %{size_download} bytes; speed %
{speed_download} bytes/s ' --retry 10 -o $bin/tools.tar.gz 'https://juju-dist.s3.amazonaws.com/tools/releases/juju-1.20.7-trusty-amd64.tgz'
Bootstrapping Juju machine agent
Starting Juju machine agent (jujud-machine-0)
environment: amazon
machines:
  "0":
    agent-state: started
    agent-version: 1.20.7
    dns-name: ec2-54-176-245-238.us-west-1.compute.amazonaws.com
    instance-id: i-4df1fa13
    instance-state: running
    series: trusty
    hardware: arch=amd64 cpu-cores=1 cpu-power=100 mem=1740M root-disk=8192M
    state-server-member-status: has-vote
services: {}
```

You can watch the bootstrapped instance being deployed by going to console.aws.amazon.com to the Instances page of the EC2 service section. In the following screenshot, the green arrow points to the instance being bootstrapped.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-d9146344	m1.small	us-west-2a	pending	Initializing	None	

You may have to hit refresh several times before the deployment starts.

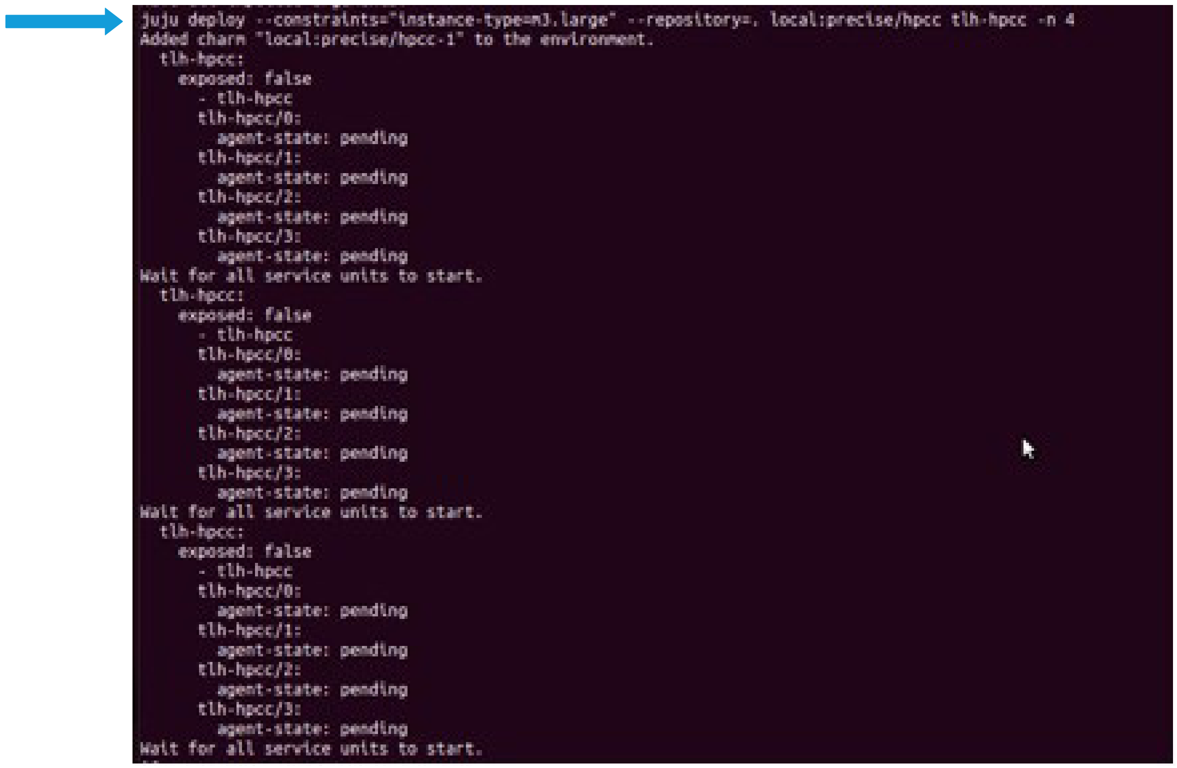
The bootstrap will take a few minutes. And, it outputs several lines to the Ubuntu terminal window (In the above terminal window screenshot , the bootstrap's output is everything down to the 2nd green arrow).

Once the bootstrap is done, HtS3 does a "juju status" which displays the status of the current state of the deployment (In the above terminal window screenshot , the status starts at the 2nd green arrow.)

2. deploy

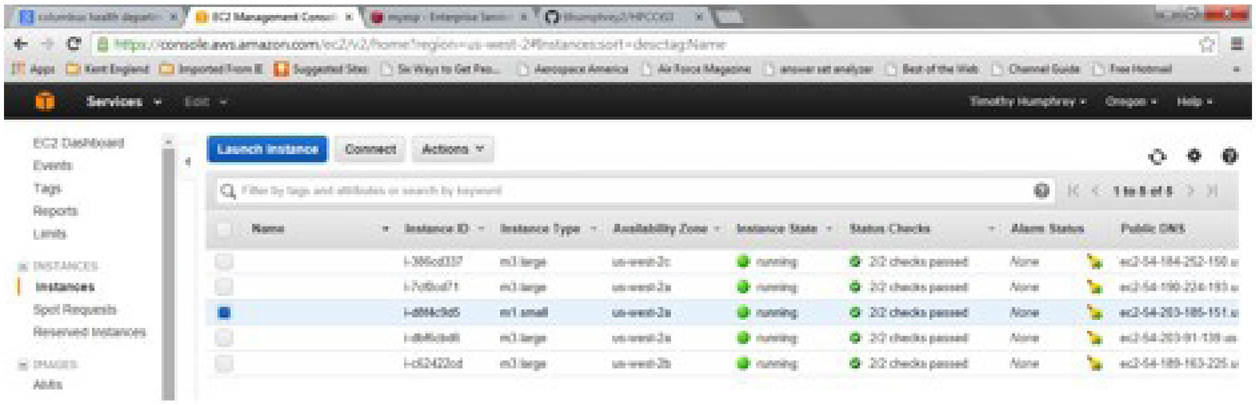
HtS3 continues to display the bootstrap's status until its state is 'started'. Then, and only then does HtS3 have juju deploy the HPCC to AWS.

The following terminal window screenshot shows HtS3's terminal output during the deployment (the green arrow marks the beginning of the deployment process). Also, the screenshot shows HtS3 displaying the status of the deployment while it waits for the state of all nodes to be 'started'.



```
juju deploy --constraints="instance-type=m3.large" --repository=. local:precise/hpcc tlh-hpcc -n 4
Added charm "local:precise/hpcc-1" to the environment.
  tlh-hpcc:
    exposed: false
      - tlh-hpcc
      tlh-hpcc/0:
        agent-state: pending
      tlh-hpcc/1:
        agent-state: pending
      tlh-hpcc/2:
        agent-state: pending
      tlh-hpcc/3:
        agent-state: pending
Wait for all service units to start.
  tlh-hpcc:
    exposed: false
      - tlh-hpcc
      tlh-hpcc/0:
        agent-state: pending
      tlh-hpcc/1:
        agent-state: pending
      tlh-hpcc/2:
        agent-state: pending
      tlh-hpcc/3:
        agent-state: pending
Wait for all service units to start.
  tlh-hpcc:
    exposed: false
      - tlh-hpcc
      tlh-hpcc/0:
        agent-state: pending
      tlh-hpcc/1:
        agent-state: pending
      tlh-hpcc/2:
        agent-state: pending
      tlh-hpcc/3:
        agent-state: pending
Wait for all service units to start.
```

After the nodes of the HPCC have been deployed, the Instances page of the EC2 section of console.aws.amazon.com looks like the following screenshot.



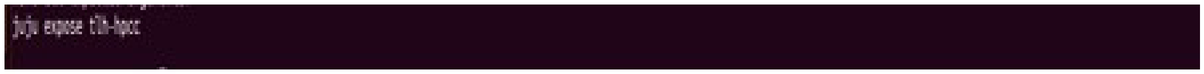
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-386cd337	m3.large	us-west-2c	● running	2/2 checks passed	None	ec2-54-184-252-150.u
	i-7c6bcd71	m3.large	us-west-2a	● running	2/2 checks passed	None	ec2-54-190-224-193.u
	i-d884c9d5	m1.small	us-west-2a	● running	2/2 checks passed	None	ec2-54-203-185-151.u
	i-db4fcbd8	m3.large	us-west-2a	● running	2/2 checks passed	None	ec2-54-203-91-139.us
	i-c62422cd	m3.large	us-west-2b	● running	2/2 checks passed	None	ec2-54-189-163-225.u

Be aware that seeing the instances on the Instances page does not mean that the HPCC System is fully operational. As the above terminal window screenshot shows, the nodes might not be in the 'started' state (the above terminal window screenshot shows the nodes are in the 'pending' state. There are three states that you might see during deployment: pending, installed, and started.

The HPCC System won't be fully operational until a) all nodes are in the 'started' state and b) the HPCC platform has completed its starting procedure.

3. expose

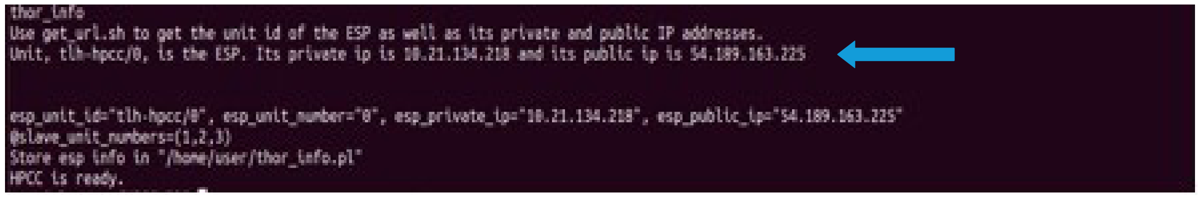
Once, all nodes are in the 'started' state, HtS3 will have juju expose its public IP address which is needed to bring up the HPCC's ECL Watch page in your browser and to initialize your IDE.



```
juju expose tlh-hpcc
```

4. thor_info

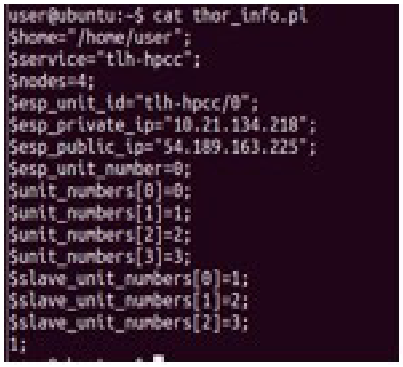
Once the public IP address is exposed, HtS3 uses a script called get_url.sh, part of the HPCC Charm, to get the ESP's unit id, local IP and public IP addresses. The following terminal window screenshot shows what HtS3 outputs when it does thor_info. The green arrow points to the line having the unit id, private IP, and public IP. You enter the public IP address followed by :8010 into your browser's address field to bring up the deployed HPCC System's ECL Watch page. So, for this presentation, ECL Watch can be displayed in your browser using the following address: http://54.189.163.225:8010, which is the ESP's public IP address with ':8010' appended on the end.



```
thor_info
Use get_url.sh to get the unit id of the ESP as well as its private and public IP addresses.
Unit, tlh-hpcc/0, is the ESP. Its private ip is 10.21.134.218 and its public ip is 54.189.163.225

esp_unit_id="tlh-hpcc/0", esp_unit_number="0", esp_private_ip="10.21.134.218", esp_public_ip="54.189.163.225"
@slave_unit_numbers=(1,2,3)
Store esp info in "/home/user/thor_info.pl"
HPCC is ready.
```

When the thor_info command executes too soon after the expose command, it isn't always successful at returning the unit-id, private IP, and public IP: 1) it may return parse errors and 2) it may return the wrong unit-id, private IP, and public IP. To learn how to handle these possible problems, see ***Handling Errors***, below.

Besides displaying the unit-id, private IP and public IP of the ESP, the thor_info HtS3 command, also, creates the file, thor_info.pl, in your home directory. The following terminal window screenshot shows the contents of this file.

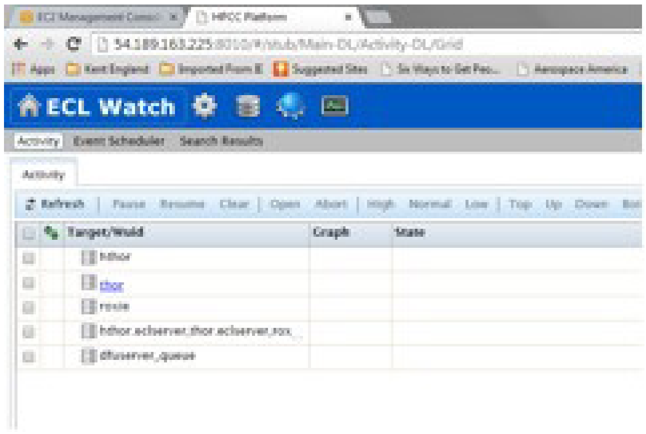


```
user@ubuntu:~$ cat thor_info.pl
$home="/home/user";
$service="tlh-hpcc";
$nodes=4;
$esp_unit_id="tlh-hpcc/0";
$esp_private_ip="10.21.134.218";
$esp_public_ip="54.189.163.225";
$esp_unit_number=0;
$unit_numbers[0]=0;
$unit_numbers[1]=1;
$unit_numbers[2]=2;
$unit_numbers[3]=3;
$slave_unit_numbers[0]=1;
$slave_unit_numbers[1]=2;
$slave_unit_numbers[2]=3;
1;
```

5. Get Files onto Deployed Thor

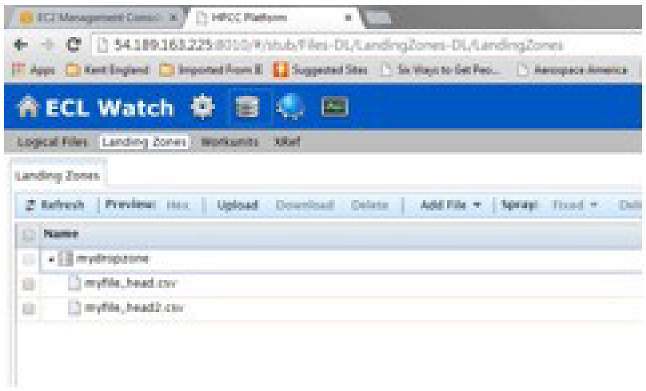
To demonstrate saving and restoring files, I must have files on my deployed HPCC's THOR. I use ECL Watch to do this.

First, I must bring-up ECL Watch in my browser. So, I enter into my browser address box http://54.189.163.225:8010 (instead of the IP address shown here, you will use the public IP address of your deployed HPCC's ESP), where 8010 is the port number for ECL Watch. The following screenshot snippet shows what this should look like.



Target/Wuid	Graph	State
hthor		
thor		
roxie		
hthor.eclserver,thor.eclserver,rox...		
dfuserver_queue		

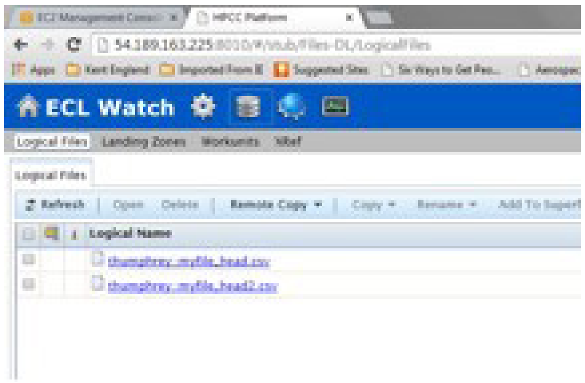
Next, I upload two files from my local machine to the drop zone. The following screenshot snippet shows those files in the drop zone.



54.189.163.225:8010/#/stub/Files-DL/LandingZones-DL/LandingZones

mydropzone
myfile_head.csv
myfile_head2.csv

Then, I used ECL Watch's spray delimited function to spray these two files onto the deployed THOR. The following screenshot snippet shows those sprayed files as logical files on the deployed THOR.



54.189.163.225:8010/#/stub/Files-DL/LogicalFiles

thumphrey::myfile_head.csv
thumphrey::myfile_head2.csv

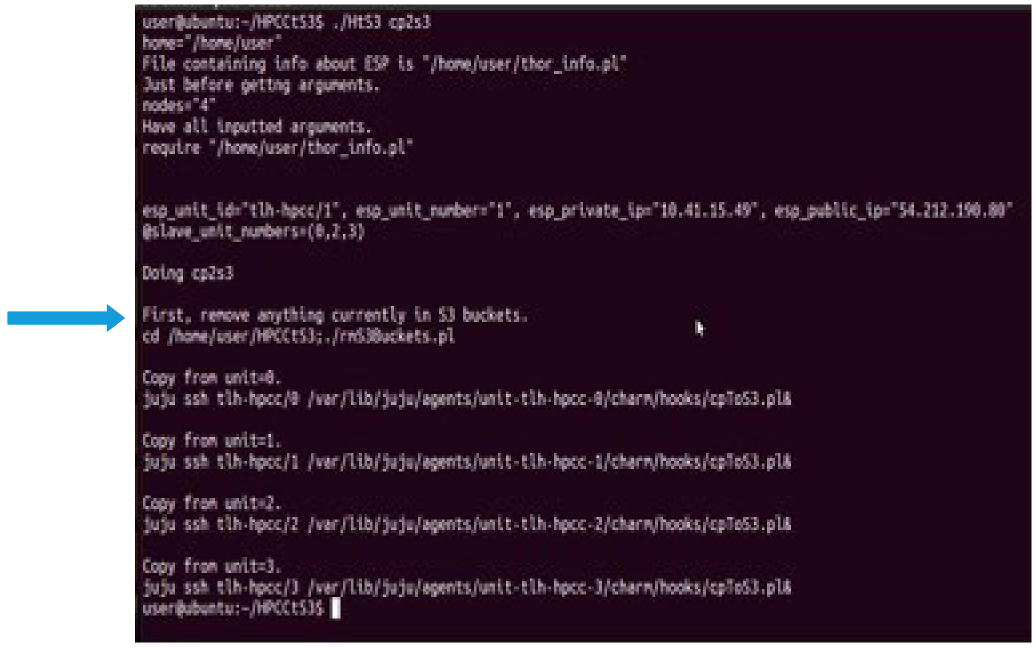
6. cp2s3

Now that we have files on the drop zone and on the deployed HPCC's THOR, we can copy these files to S3 buckets using the following command.

```
HtS3 cp2s3
```

Notice, in the above command we don't have '-nodes 4', like we previously did. Why? Because the number of nodes is taken from the file created by the thor_info HtS3 command.

The following terminal window screenshot shows what the above command outputs IF S3 buckets are currently empty. But, if they aren't empty then they are deleted where the green arrow is pointing in this screenshot.



```
user@ubuntu:~/HPCCtS3$ ./HtS3 cp2s3
home="/home/user"
File containing info about ESP is "/home/user/thor_info.pl"
Just before getting arguments.
nodes="4"
Have all inputted arguments.
require "/home/user/thor_info.pl"


esp_unit_id="tlh-hpcc/1", esp_unit_number="1", esp_private_ip="10.41.15.49", esp_public_ip="54.212.190.80"
@slave_unit_numbers=(0,2,3)

Doing cp2s3

First, remove anything currently in S3 buckets.
cd /home/user/HPCCtS3;./rmS3Buckets.pl

Copy from unit=0.
juju ssh tlh-hpcc/0 /var/lib/juju/agents/unit-tlh-hpcc-0/charm/hooks/cpToS3.pl&

Copy from unit=1.
juju ssh tlh-hpcc/1 /var/lib/juju/agents/unit-tlh-hpcc-1/charm/hooks/cpToS3.pl&

Copy from unit=2.
juju ssh tlh-hpcc/2 /var/lib/juju/agents/unit-tlh-hpcc-2/charm/hooks/cpToS3.pl&

Copy from unit=3.
juju ssh tlh-hpcc/3 /var/lib/juju/agents/unit-tlh-hpcc-3/charm/hooks/cpToS3.pl&
user@ubuntu:~/HPCCtS3$
```

The above screenshot shows that HtS3 asked for files to be copied from each of the 4 nodes of our THOR. The copying actually occurs on the several nodes of the deployed HPCC's THOR. So, seeing the above output doesn't mean the copying has completed.

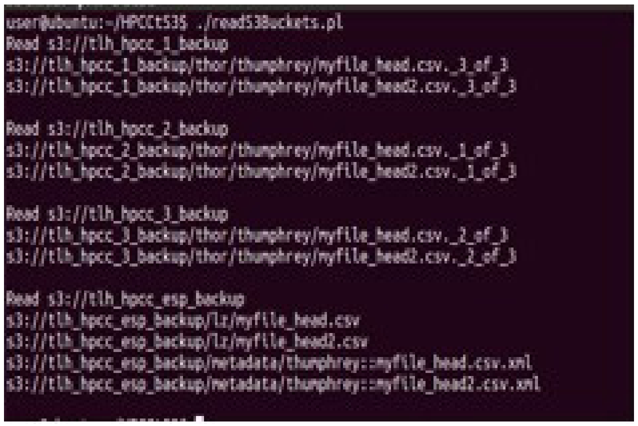
You will see additional output during the cp2s3 process over-and-above what you see in the above terminal window screenshot. The additional output comes from the copy processes on the several nodes of your deployed HPCC's THOR.

7. readS3Buckets.pl

To verify that the files were copied to S3 buckets, do the following:

```
./readS3Buckets.pl
```

The following terminal window screenshot shows what the output is of the above command.



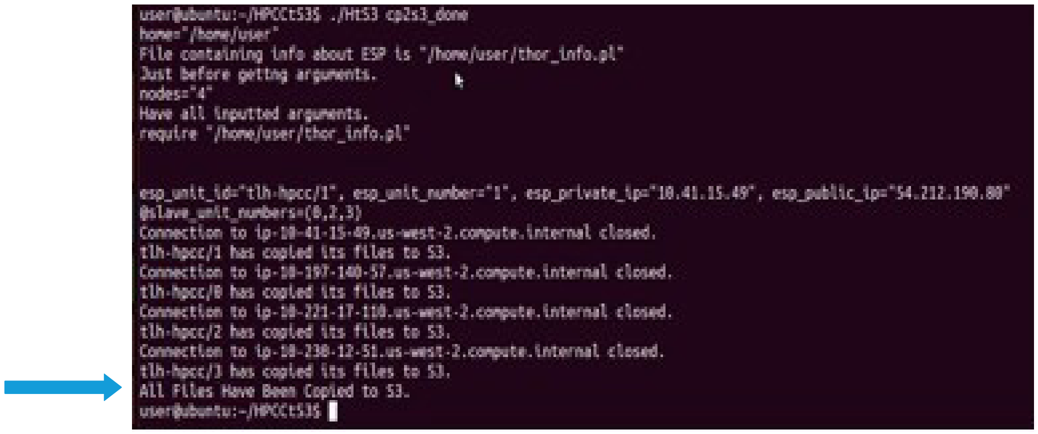
```
user@ubuntu:~/HPCCtS3$ ./readS3Buckets.pl
Read s3://tlh_hpcc_1_backup
s3://tlh_hpcc_1_backup/thor/thumphrey/myfile_head.csv._3_of_3
s3://tlh_hpcc_1_backup/thor/thumphrey/myfile_head2.csv._3_of_3

Read s3://tlh_hpcc_2_backup
s3://tlh_hpcc_2_backup/thor/thumphrey/myfile_head.csv._1_of_3
s3://tlh_hpcc_2_backup/thor/thumphrey/myfile_head2.csv._1_of_3

Read s3://tlh_hpcc_3_backup
s3://tlh_hpcc_3_backup/thor/thumphrey/myfile_head.csv._2_of_3
s3://tlh_hpcc_3_backup/thor/thumphrey/myfile_head2.csv._2_of_3

Read s3://tlh_hpcc_esp_backup
s3://tlh_hpcc_esp_backup/lz/myfile_head.csv
s3://tlh_hpcc_esp_backup/lz/myfile_head2.csv
s3://tlh_hpcc_esp_backup/metadata/thumphrey::myfile_head.csv.xml
s3://tlh_hpcc_esp_backup/metadata/thumphrey::myfile_head2.csv.xml
```

This screenshot shows there were 4 buckets created (one for each node of the THOR) where the 1st bucket contains the 3rd part of the files myfile_head.csv and myfile_head2.csv. And the 2nd and 3rd buckets contain the 1st and 2nd parts of the same two files, respectively. Lastly, the 4th bucket contains the files from the drop zone: myfile_head.csv, and myfile_head2.csv. Plus, it contains the metadata (xml) files for the two files sprayed to the THOR (Note. These metafiles are used in the last step of the file restoration process.)

But, the above still may not convince you that the files have been completely copied to S3 buckets. So, you can execute the following command to get additional assurance.

```
./HtS3 cp2s3_done
```

The following terminal window screenshot shows what this command outputs. The green arrow shows where it says, “All Files Have Been Copied to S3”. The above command determines that all files have been copied by checking on each node of the THOR to see if the ‘done copying’ file exists. This file is the last thing the copy process does after copying its files to S3 buckets.



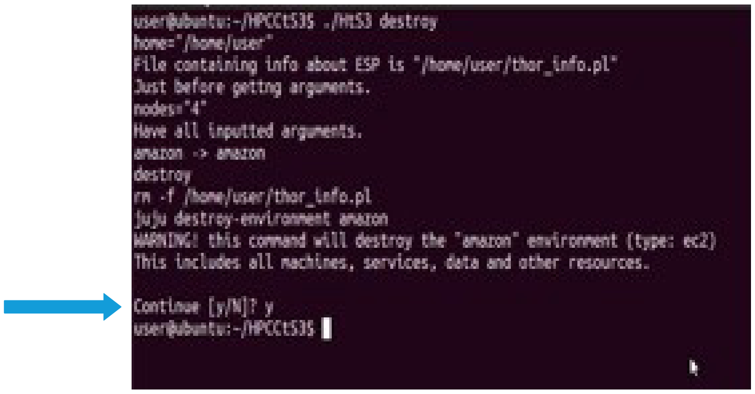
```
user@ubuntu:~/HPCCtS3$ ./HtS3 cp2s3_done
home="/home/user"
File containing info about ESP is "/home/user/thor_info.pl"
Just before getting arguments.
nodes="4"
Have all inputted arguments.
require "/home/user/thor_info.pl"


esp_unit_id="tlh-hpcc/1", esp_unit_number="1", esp_private_ip="10.41.15.49", esp_public_ip="54.212.190.80"
@slave_unit_numbers=(0,2,3)
Connection to ip-10-41-15-49.us-west-2.compute.internal closed.
tlh-hpcc/1 has copied its files to S3.
Connection to ip-10-197-140-57.us-west-2.compute.internal closed.
tlh-hpcc/0 has copied its files to S3.
Connection to ip-10-221-17-110.us-west-2.compute.internal closed.
tlh-hpcc/2 has copied its files to S3.
Connection to ip-10-230-12-51.us-west-2.compute.internal closed.
tlh-hpcc/3 has copied its files to S3.
All Files Have Been Copied to S3.
user@ubuntu:~/HPCCtS3$
```

8. destroy

Once, all files have been copied to S3 buckets, we can shut down the deployed HPCC System with the following command.

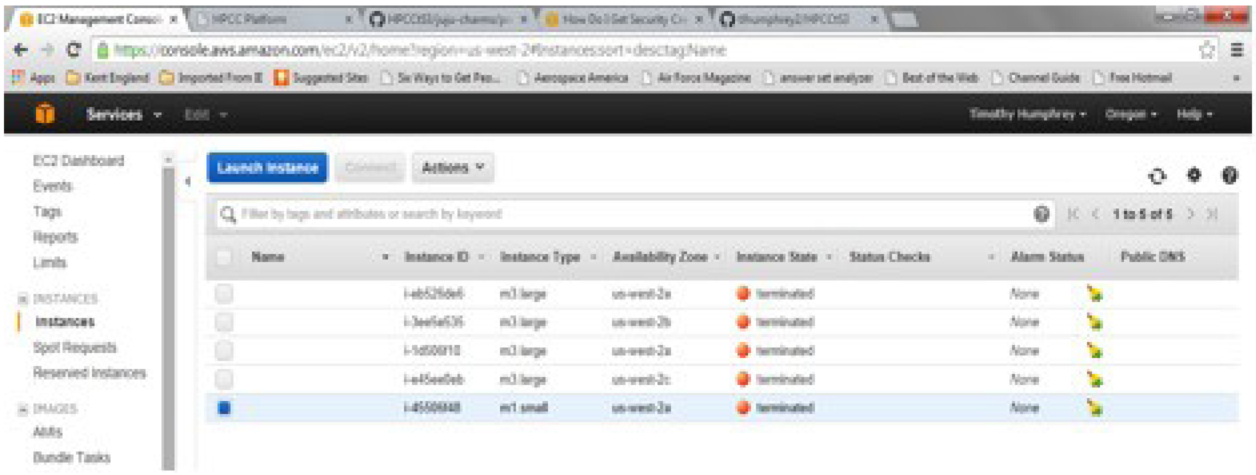
```
./HtS3 destroy
```

Its output looks like the following terminal window screenshot. You will notice that it asked you if you really want to do this, since it totally shots down the HPCC System (where the green arrow points).



```
user@ubuntu:~/HPCCtS3$ ./HtS3 destroy
home="/home/user"
File containing info about ESP is "/home/user/thor_info.pl"
Just before gettng arguments.
nodes="4"
Have all inputted arguments.
amazon -> amazon
destroy
rm -f /home/user/thor_info.pl
juju destroy-environment amazon
WARNING! this command will destroy the "amazon" environment (type: ec2)
This includes all machines, services, data and other resources.

Continue [y/N]? y
user@ubuntu:~/HPCCtS3$ 
```

The following is a screenshot of the Instances page of the EC2 section of console.aws.amazon.com showing all the instances of the HPCC System terminated.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-eb526de6	m3.large	us-west-2a	terminated		None	
	i-3ee5e535	m3.large	us-west-2b	terminated		None	
	i-5d500810	m3.large	us-west-2a	terminated		None	
	i-e45ee0eb	m3.large	us-west-2c	terminated		None	
	i-45506848	m1.small	us-west-2a	terminated		None	

9. restore_hpcc

Sometime later, you can deploy another HPCC System and have all the files of the previously deployed THOR restored to your newly deployed THOR. The following command begins the process of configuring and deploying another

HPCC System with the same number of nodes as the first one (i.e. 4). (Note. Currently this software wants you to have the same number of nodes as the HPCC System we copied the files from.)

```
./HtS3 restore_hpcc
```

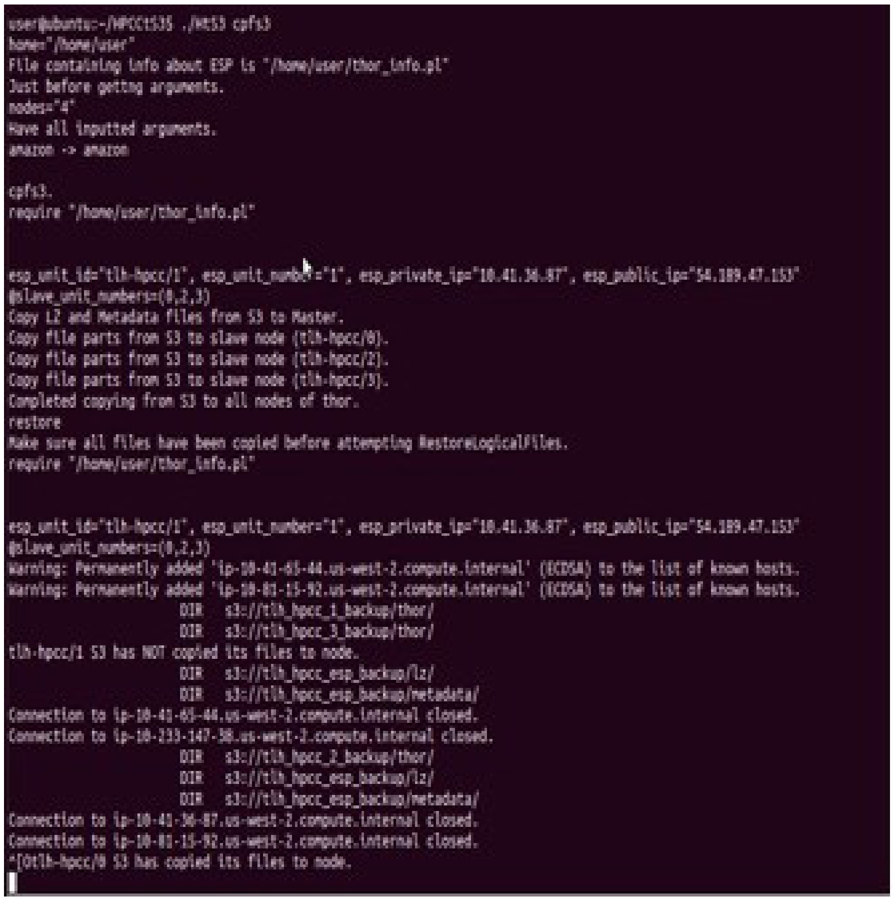
The above command will do steps 1 through 4, discusses above, that is, bootstrap, deploy expose, and thor_info.

10. cpfs3

The following command will copy the files stored in S3 buckets to the nodes of the deployed THOR. This includes files on the drop zone.

```
./HtS3 cpfs3
```

The following terminal window screenshot shows the output of cpfs3.



```
user@ubuntu:~/HPCCtS3$ ./HtS3 cpfs3
home="/home/user"
File containing info about ESP is "/home/user/thor_info.pl"
Just before getting arguments.
nodes="4"
Have all inputted arguments.
amazon -> amazon

cpfs3.
require "/home/user/thor_info.pl"


esp_unit_id="tlh-hpcc/1", esp_unit_number="1", esp_private_ip="10.41.36.87", esp_public_ip="54.189.47.153"
@slave_unit_numbers=(0,2,3)
Copy LZ and Metadata files from S3 to Master.
Copy file parts from S3 to slave node (tlh-hpcc/0).
Copy file parts from S3 to slave node (tlh-hpcc/2).
Copy file parts from S3 to slave node (tlh-hpcc/3).
Completed copying from S3 to all nodes of thor.
restore
Make sure all files have been copied before attempting RestoreLogicalFiles.
require "/home/user/thor_info.pl"


esp_unit_id="tlh-hpcc/1", esp_unit_number="1", esp_private_ip="10.41.36.87", esp_public_ip="54.189.47.153"
@slave_unit_numbers=(0,2,3)
Warning: Permanently added 'ip-10-41-65-44.us-west-2.compute.internal' (ECDSA) to the list of known hosts.
Warning: Permanently added 'ip-10-81-15-92.us-west-2.compute.internal' (ECDSA) to the list of known hosts.
                        DIR   s3://tlh_hpcc_1_backup/thor/
                        DIR   s3://tlh_hpcc_3_backup/thor/
tlh-hpcc/1 S3 has NOT copied its files to node.
                        DIR   s3://tlh_hpcc_esp_backup/lz/
                        DIR   s3://tlh_hpcc_esp_backup/metadata/
Connection to ip-10-41-65-44.us-west-2.compute.internal closed.
Connection to ip-10-233-147-38.us-west-2.compute.internal closed.
                        DIR   s3://tlh_hpcc_2_backup/thor/
                        DIR   s3://tlh_hpcc_esp_backup/lz/
                        DIR   s3://tlh_hpcc_esp_backup/metadata/
Connection to ip-10-41-36-87.us-west-2.compute.internal closed.
Connection to ip-10-81-15-92.us-west-2.compute.internal closed.
^[Otlh-hpcc/0 S3 has copied its files to node.
```

11. look at files on THOR

Once HtS3 cpfs3 completes copying files from the S3 buckets, go to your ECL Watch and check to see if the files have been copied to the nodes of the deployed THOR (see step 5 to see what ECL Watch should display if the files were copied).

It takes several minutes after HtS3 cpfs3 completes before the files are completely copied from the S3 buckets to the nodes of the THOR cluster.

Handling Errors

The following table shows where errors/problems can occur (that I know about).

1	bootstrap or deploy	Normally, these are the only juju performed tasks that have errors.
2	thor_info	When thor_info is executed too soon after all units have the state 'started', you might get errors.

bootstrap or deploy

If bootstrap gets an error, first CTRL-C out of HtS3. Most often juju will terminate the bootstrap instance when you CTRL-C. But, make sure this is the case by looking at the Instances page of console.aws.amazon.com. If it hasn't terminated then do the following command to do so.

```
./HtS3 destroy
```

Sometimes, during a bootstrap, you will get a warning message telling you the file, amazon.jenv, exists in the .juju/environments directory. If you get this message CTRL-C out of HtS3 and remove this file with the following command.

```
rm ~/.juju/environments/amazon.jenv
```

After deleting it, you can start the boot with the following command.s

```
./HtS3 -nodes 4
```

handling errors during deploy

If errors occur during a deploy, first CTRL-C out of HtS3. Then, check console.aws.amazon.com to see if any of the THOR's instances are running. If they are running, terminate all of them except the bootstrap instance which will be the one with instance type m1.small. Next, remove any files in ~/.juju/environments. Then, do the command above, again, to deploy the HPCC System.

thor_info

I've seen two errors occur when thor_info is executed too soon after the juju expose: 1) a parse error can occur, or 2) the public IP displayed isn't correct. For both errors you will rerun thor_info as so:

```
rm ~/thor_info.pl
./HtS3 –nodes 4 thor_info_only
```

The above, reruns only thor_info. So, the above first deletes the file created by thor_info .Then, it executes only thor_info. (note. If you get a parse error, thor_info.pl won't be created. So, rm ~/thor_info.pl isn't needed).

To determine if the public IP works, attempt to bring-up ECL Watch for the deployed HPCC System in your browser by entering into your browser address box http://54.189.163.225:8010, where 8010 is the port number for ECL Watch (note. You will enter the public IP displayed to you, instead of 54.189.163.225). If ECL Watch doesn't come up, possibly thor_info didn't give you the correct IP address. But, it is also possible that the deployed HPCC System hasn't had enough time to fully deploy.

Sometimes the displayed public IP is correct, but you still can't get ECL Watch to display in your browser. This happens when the HPCC System isn't fully deployed. And, the only fix is to wait awhile and try again.

More information about problems encountered might be found in the logs created on the nodes of the deployed HPCC. Juju has its log files in /var/log/juju and HtS3 places its log files in /home/ubuntu, which is the directory you are in when you ssh into a node. Also, Juju has simplified the syntax for doing ssh. For example, to ssh into the node with juju unit-id, tlh-hpcc/0, use following command.

```
juju ssh tlh-hpcc/0
```

Final Words

When you are done with your deployed HPCC System shut it down so AWS charges don't continue to accumulate. You can do this with the following command.

```
./HtS3 destroy
```

Also, if you don't want the files saved in the S3 buckets then to avoid additional AWS charges, delete them with the following command.

```
./rmS3Buckets.pl
```

All the juju commands executed by HtS3 can be executed from the command line outside of HtS3. Example, you can find out the status of a deployed system by doing the following from the command line.

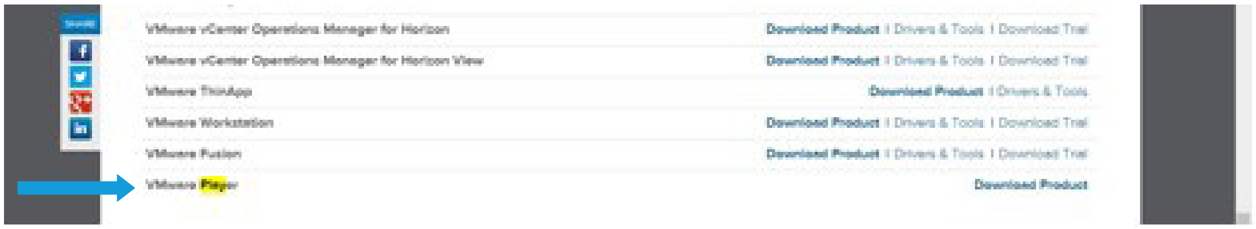
```
juju status
```

Appendix A. Setting Up Ubuntu 12.04 Linux VMware Machine

Download and Install VMware Player

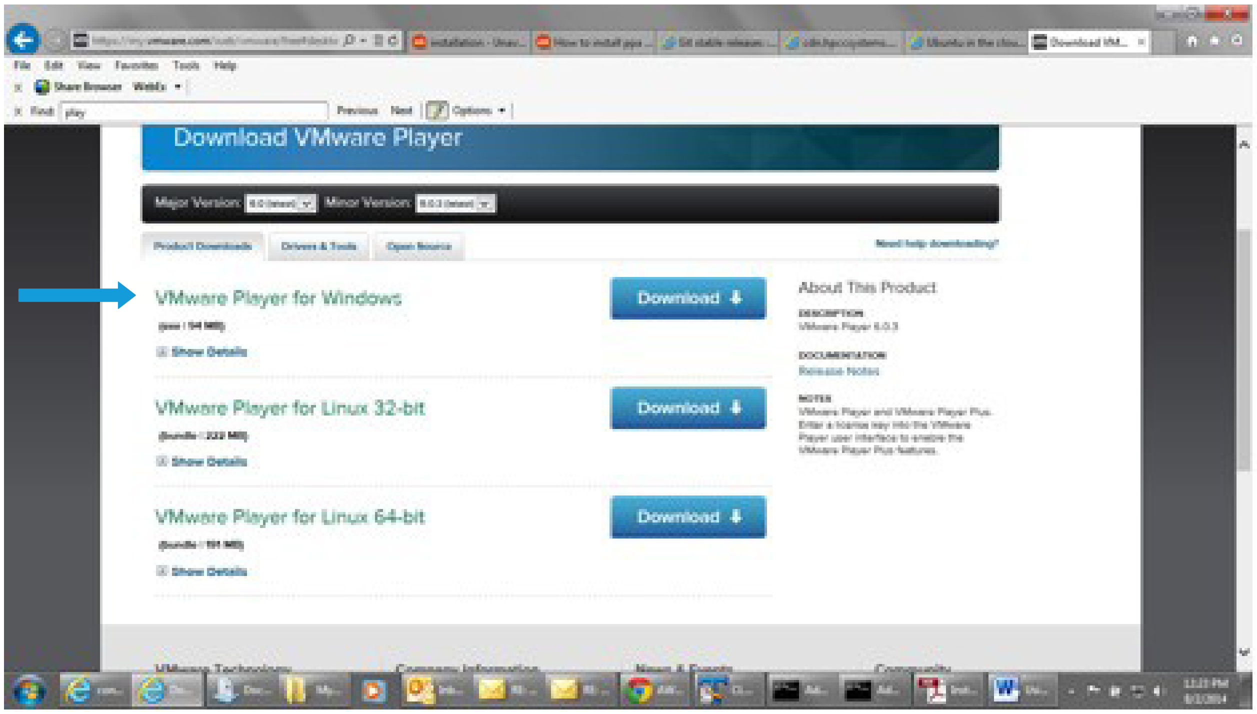
First, download a VMware Player at https://my.vmware.com/web/vmware/free.

As you can see from the following screenshot of the site, there are many products that you can download. The one that you want to download is marked with a green arrow, below, i.e. VMware Player.



VMware vCenter Operations Manager for Horizon	Download Product | Drivers & Tools | Download Trial
VMware vCenter Operations Manager for Horizon View	Download Product | Drivers & Tools | Download Trial
VMware ThinApp	Download Product | Drivers & Tools
VMware Workstation	Download Product | Drivers & Tools | Download Trial
VMware Fusion	Download Product | Drivers & Tools | Download Trial
VMware Player	Download Product

You will get the following web page after clicking on the VMware Player link shown above.



Download VMware Player

Major Version: 6.0 (latest) Minor Version: 6.0.3 (latest)

Product Downloads | Drivers & Tools | Open Source | Need help downloading?

VMware Player for Windows
(exe | 94 MB)
Show Details
Download

VMware Player for Linux 32-bit
(bundle | 222 MB)
Show Details
Download

VMware Player for Linux 64-bit
(bundle | 191 MB)
Show Details
Download

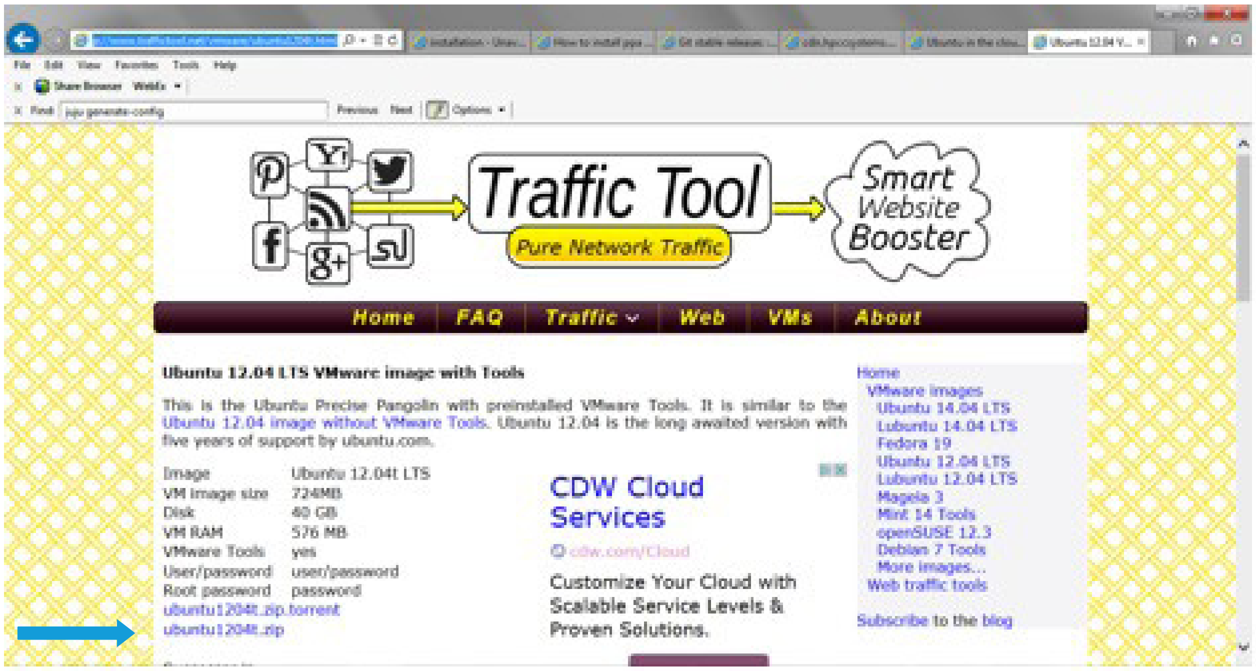
About This Product
DESCRIPTION
VMware Player 6.0.3
DOCUMENTATION
Release Notes
NOTES
VMware Player and VMware Player Plus. Enter a license key into the VMware Player user interface to enable the VMware Player Plus features.

Click on the Windows VMware Player download button, marked with the green arrow, i.e. VMware Player for Windows. The download takes a few minutes.

Download and Install Ubuntu 12.04 VM Image with VMware Tools.

Secondly, download the VMware image for Ubuntu 12.04 Linux machine with VMware Tools at http://www.traffictool.net/vmware/ubuntu1204t.html

When you load this site into your browser, it should look like the following:



Ubuntu 12.04 LTS VMware image with Tools

This is the Ubuntu Precise Pangolin with preinstalled VMware Tools. It is similar to the Ubuntu 12.04 image without VMware Tools. Ubuntu 12.04 is the long awaited version with five years of support by ubuntu.com.

Image	Ubuntu 12.04t LTS
VM image size	724MB
Disk	40 GB
VM RAM	576 MB
VMware Tools	yes
User/password	user/password
Root password	password

ubuntu1204t.zip.torrent
ubuntu1204t.zip

CDW Cloud Services
cdw.com/Cloud
Customize Your Cloud with Scalable Service Levels & Proven Solutions.

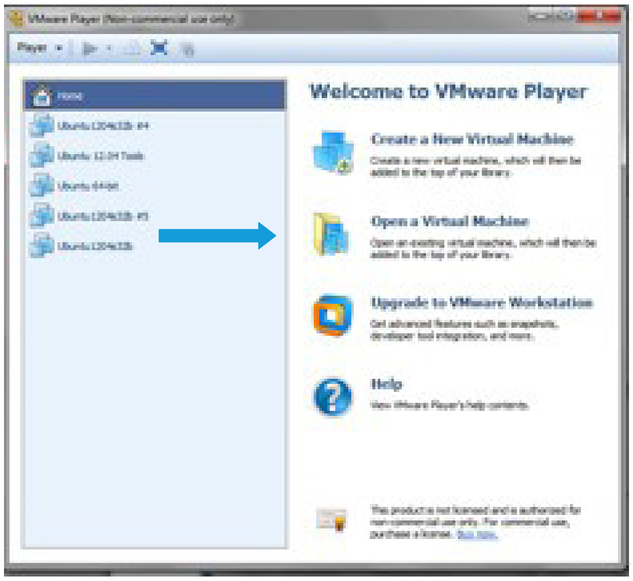
Home
VMware images
Ubuntu 14.04 LTS
Lubuntu 14.04 LTS
Fedora 19
Ubuntu 12.04 LTS
Lubuntu 12.04 LTS
Mageia 3
Mint 14 Tools
openSUSE 12.3
Debian 7 Tools
More images...
Web traffic tools
Subscribe to the blog

To download Ubuntu 12.04 with the VMware Tools, click on the link pointed to by the green arrow, above. Save the downloaded zip file where you can find it. The download may take several minutes.

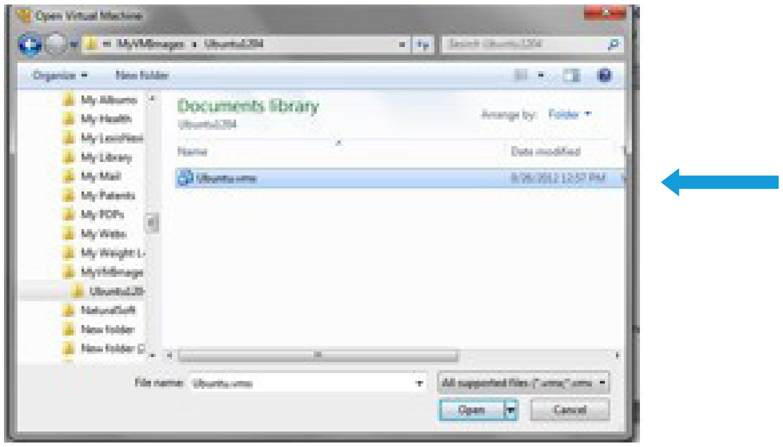
After the download is complete, extract the contents of the zip file and place it in a folder where you can find it. I've stored mine in Documents\MyVMImages.

Setup Ubuntu 12.04 and VMware Tools on the VMware Player

Open VMware Player and click on Open a Virtual Machine (marked with a green arrow below).

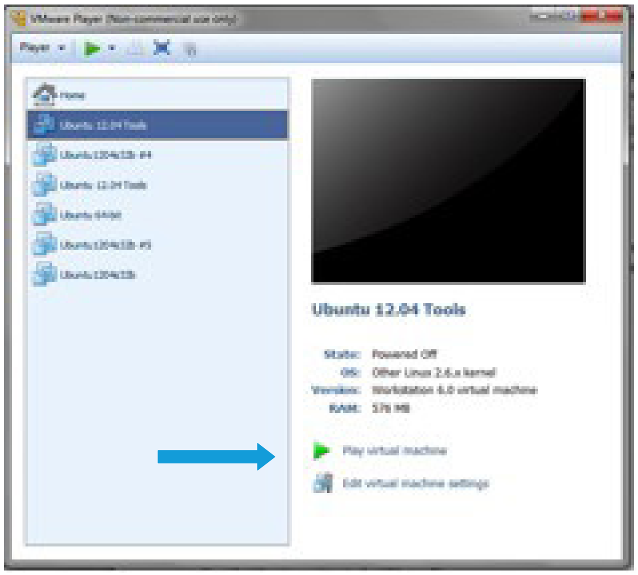


This will open the Windows Explorer so you can open the folder containing the Ubuntu 12.04 VM image. Once you have found it, you should see the file Ubuntu.vmx (see below screenshot).

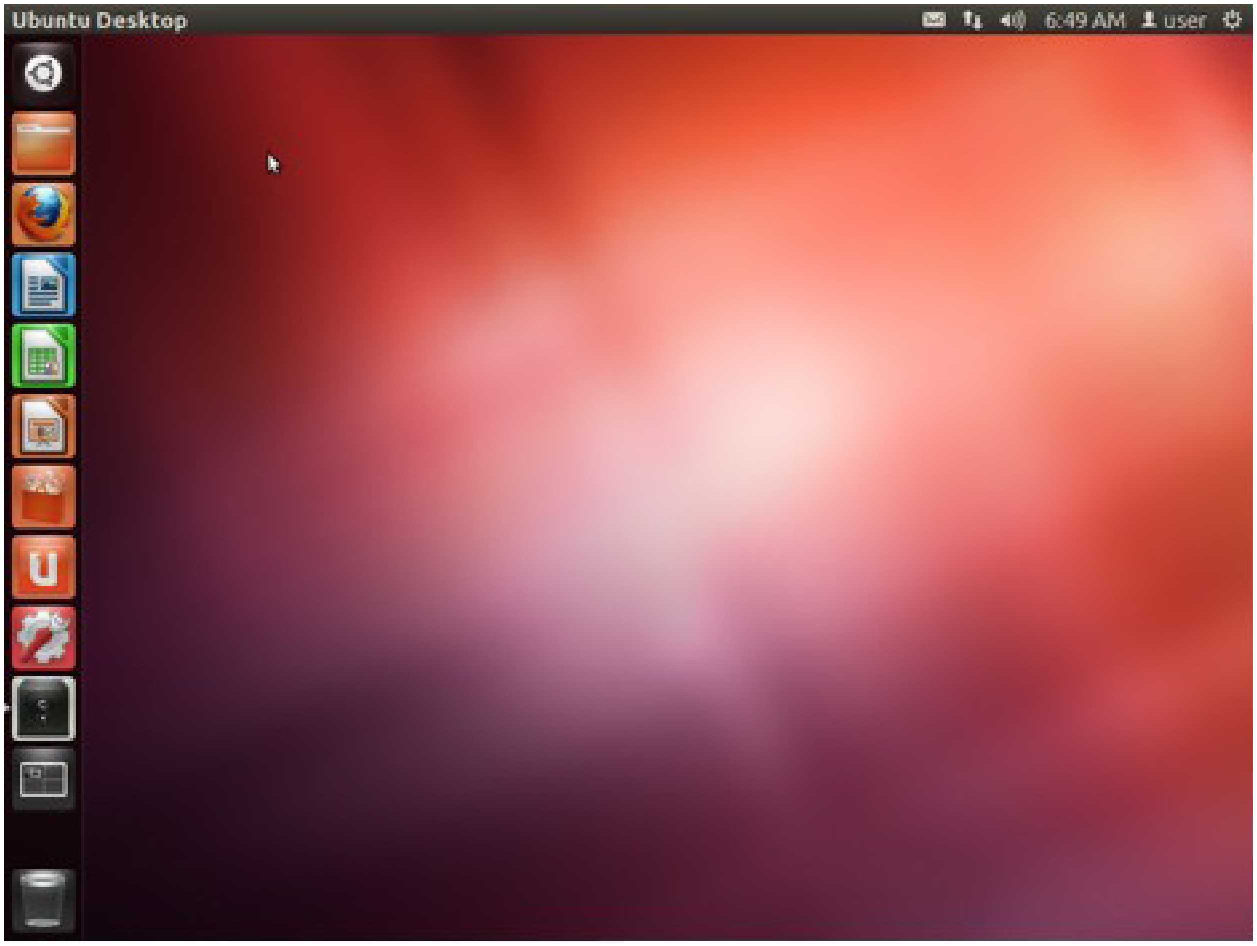


Click on it so it will be listed in the VMware Player's list of virtual machines (see the top green arrow in the following screenshot). As you can see, it has already been selected. So, you can click on Play virtual machine to start the virtual machine (see bottom green arrow in the following screenshot).

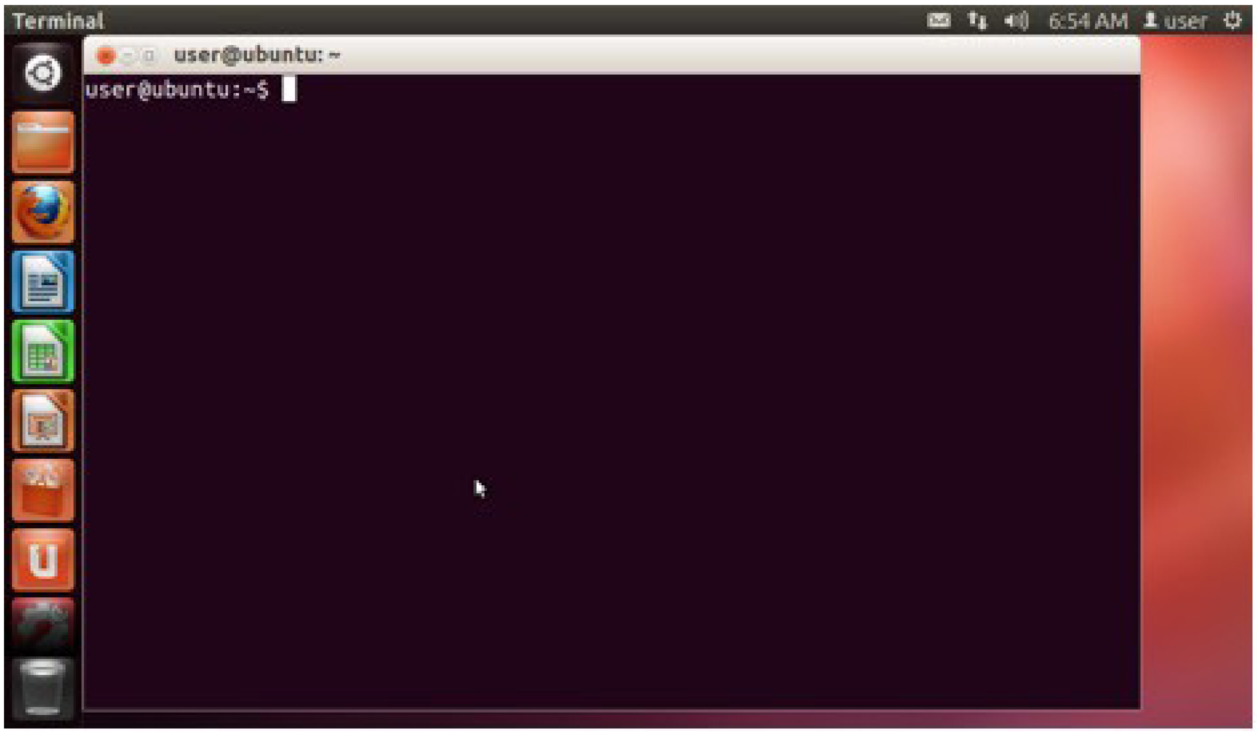
When the machine starts, you may see a popup asking to check for updates. Click the X to ignore it.



When the machine is ready for use, its screen should look like the following screenshot.



So, at this point, you can open an Ubuntu Terminal window with CRTL-ALT-T or by selecting in from the Dashboard (top icon on the left). The machine with an open Terminal Windows looks like the following.



```
user@ubuntu:~$ 
```

Appendix B. HtS3 and HPCC Charm Scripts

The following table briefly describes the files and directories of HtS3 that were cloned from the github repository. The table has four columns. The first column gives the name of all files and directories shown in the directory tree presented early in this document. This column is basically a duplication of that tree and therefore shows the parent/child relationship of all files and directories. The **Type** column indicates whether the entry of that row is a file (F) or directory (D). The **Part Of** column tells which system the entry is part of, either HtS3 or HPCC Charm. And, the **Short Description** column gives a short description of the entry. But, it gives no description for any of the directory entries.

You will notice that there are HtS3 scripts in the HPCC Charm's hooks directory. They are placed here because Juju loads anything in the hooks directory onto each of the nodes of the deployed HPCC. These scripts are placed onto each node so that the copying from/to S3 is done in parallel.

File or Directory Name	Type	Part Of	Short Description
+ HtS3	F	HtS3	The main of HtS3
+ isFilesCopiedFromS3.pl	F	HtS3	Checks to see if all files copied from S3.
+ isFilesCopiedToS3.pl	F	HtS3	Checks to see if all files copied to S3.
+ juju-charms	D	HPCC Charm	
¦ + precise	D	HPCC Charm	
¦ + hpcc	D	HPCC Charm	
¦ + bin	D	HPCC Charm	
¦ ¦ + config_hpcc.sh	F	HPCC Charm	Manually configures & deploys HPCC if juju can't.
¦ ¦ + get_public_ip.pl	F	HPCC Charm	Part of get_url.sh
¦ ¦ + get-unit-with-this-pip.sh	F	HPCC Charm	Part of get_url.sh
¦ ¦ + get_url.sh	F	HPCC Charm	Gets ESP's unit-id, private & public IP addresses.
¦ ¦ + parse_config.py	F	HPCC Charm	Part of get_url.sh
¦ ¦ + parse_status.py	F	HPCC Charm	Part of get_url.sh
¦ + config.yaml	F	HPCC Charm	HPCC Charms configuration options.
¦ + copyright	F	HPCC Charm	Copyright
¦ + dependencies	D	HPCC Charm	
¦ ¦ + el6	F	HPCC Charm	Dependencies for el6
¦ ¦ + precise	F	HPCC Charm	Dependencies for ubuntu precise.
¦ ¦ + saucy	F	HPCC Charm	Dependencies for ubuntu saucy.
¦ ¦ + trusty	F	HPCC Charm	Dependencies for ubuntu trusty.
¦ + hooks	D	HPCC Charm	
¦ ¦ + common.pl	F	HtS3	Functions and global variables used by all HtS3 scripts in the hooks directory.
¦ ¦ + config-changed	F	HPCC Charm	Tasks done when HPCC configuration changes
¦ ¦ + cpAllFilePartsFromS3ToThisSlaveNode.pl	F	HtS3	Copies all file parts from S3 to THOR slave node which its ran on

File or Directory Name	Type	Part Of	Short Description
¦ ¦ + cpAllFilePartsToS3.pl	F	HtS3	Copies all file parts to s3 from THOR node which its ran on
¦ ¦ + cpLZAndMetadataFilesFromS3ToMaster.pl	F	HtS3	Copies landing zone and metadata files from S3 to the Master (i.e. ESP).
¦ ¦ + cpLZAndMetadataFilesToS3.pl	F	HtS3	Copies landing zone and metadata files of Master (i.e. ESP) to s3. (Part of cpToS3.pl)
¦ ¦ + cpLZFilesFromS3ToMaster.pl	F	HtS3	Copies landing zone files from S3 to Master (i.e. ESP).
¦ ¦ + cpMetadataFilesFromS3ToNode.pl	F	HtS3	Copies metadata files from S3 to Master (i.e. ESP). (part of cpToS3.pl)
¦ ¦ + cpToS3.pl	F	HtS3	Copied all files to S3.
¦ ¦ + hpcc-cluster-relation-changed	F	HPCC Charm	Tasks done when the hpcc-cluster relationship changes.
¦ ¦ + hpcc-cluster-relation-departed	F	HPCC Charm	Tasks done when a node departs the hpcc-cluster relationship.
¦ ¦ + hpcc-cluster-relation-joined	F	HPCC Charm	Tasks done when a node joins the hpcc-cluster relationship.
¦ ¦ + hpcc-common	F	HPCC Charm	Functions used by all the HPCC Charm scripts in the hooks directory.
¦ ¦ + install	F	HPCC Charm	Tasks done when a node is installed.
¦ ¦ + isFilesCopiedFromS3.sh	F	HtS3	Using ssh, HtS3 runs this script on a specific node to check to see if all files were copied from S3.
¦ ¦ + isFilesCopiedToS3.sh	F	HtS3	Using ssh, HtS3 runs this script on a specific node to check to see if all files were copied to S3.
¦ ¦ + list_files.pl	F	HtS3	Using ssh, HtS3 runs this which uses dfuplus to get a list of files on the THOR. This is an indication that the HPCC is fully deployed (DOESN'T ALWAYS WORK).
¦ ¦ + RestoreLogicalFiles.pl	F	HtS3	When copying files from S3 to all nodes of the deployed HPCC, this is the last task complete ONLY AFTER all files have been copied.
¦ ¦ + start	F	HPCC Charm	Tasks done when a node is first started.
¦ ¦ + stop	F	HPCC Charm	Tasks done when a stops.
¦ + icon.svg	F	HPCC Charm	Icon that shows up in Charm Store for HPCC Charm
¦ + metadata.yaml	F	HPCC Charm	Only file that is required when making a charm.
¦ + README.md	F	HPCC Charm	Tells about the HPCC Charm.
¦ + revision	F	HPCC Charm	Version of the HPCC Charm.
+ juju-status.pl	F	HtS3	Runs 'juju status', but only shows the 'Services' part of the status.
+ LICENSE.txt	F	HtS3	License for HPCCtS3. Highly recommended for github repositories.
+ newgetopt.pl	F	HtS3	Perl functions that handles options entered into HtS3, i.e. anything given on command line right of HtS3 that begins with dash ('-').
+ readS3Buckets.pl	F	HtS3	Reads s3 buckets that user specify on command line. If none given then it reads all buckets beginning with 'tlh'.
+ rmS3Buckets.pl	F	HtS3	Deletes s3 buckets that user specify on command line. If none given then it deletes all buckets beginning with 'tlh'.

HPCC SYSTEMS®

Have questions or need more information?

Visit HPCCSytems.com or contact 1.877.316.9669 or info@hpccsystems.com

About HPCC Systems®
HPCC Systems® from LexisNexis® Risk Solutions offers a proven, data-intensive supercomputing platform designed for the enterprise to process and deliver Big Data analytical problems. As an alternative to Hadoop and mainframes, HPCC Systems offers a consistent data-centric programming language, two processing platforms and a single architecture for efficient processing. Customers, such as financial institutions, insurance carriers, insurance companies, law enforcement agencies, federal government and other enterprise-class organizations leverage the HPCC Systems technology through LexisNexis® products and services. For more information, visit http://hpccsystems.com.

About LexisNexis® Risk Solutions
LexisNexis® Risk Solutions (www.lexisnexis.com/risk/) is a leader in providing essential information that helps customers across all industries and government predict, assess and manage risk. Combining cutting-edge technology, unique data and advanced scoring analytics, Risk Solutions provides products and services that address evolving client needs in the risk sector while upholding the highest standards of security and privacy. LexisNexis Risk Solutions is part of Reed Elsevier, a leading publisher and information provider that serves customers in more than 100 countries with more than 30,000 employees worldwide.

LexisNexis®