XAP MemoryXtend Tutorial

Shay Hassidim , Deputy CTO

August 2014

What this tutorial is all about?

This tutorial will guide you how to experiment with XAP MemoryXtend

MemoryXtend Full documentation

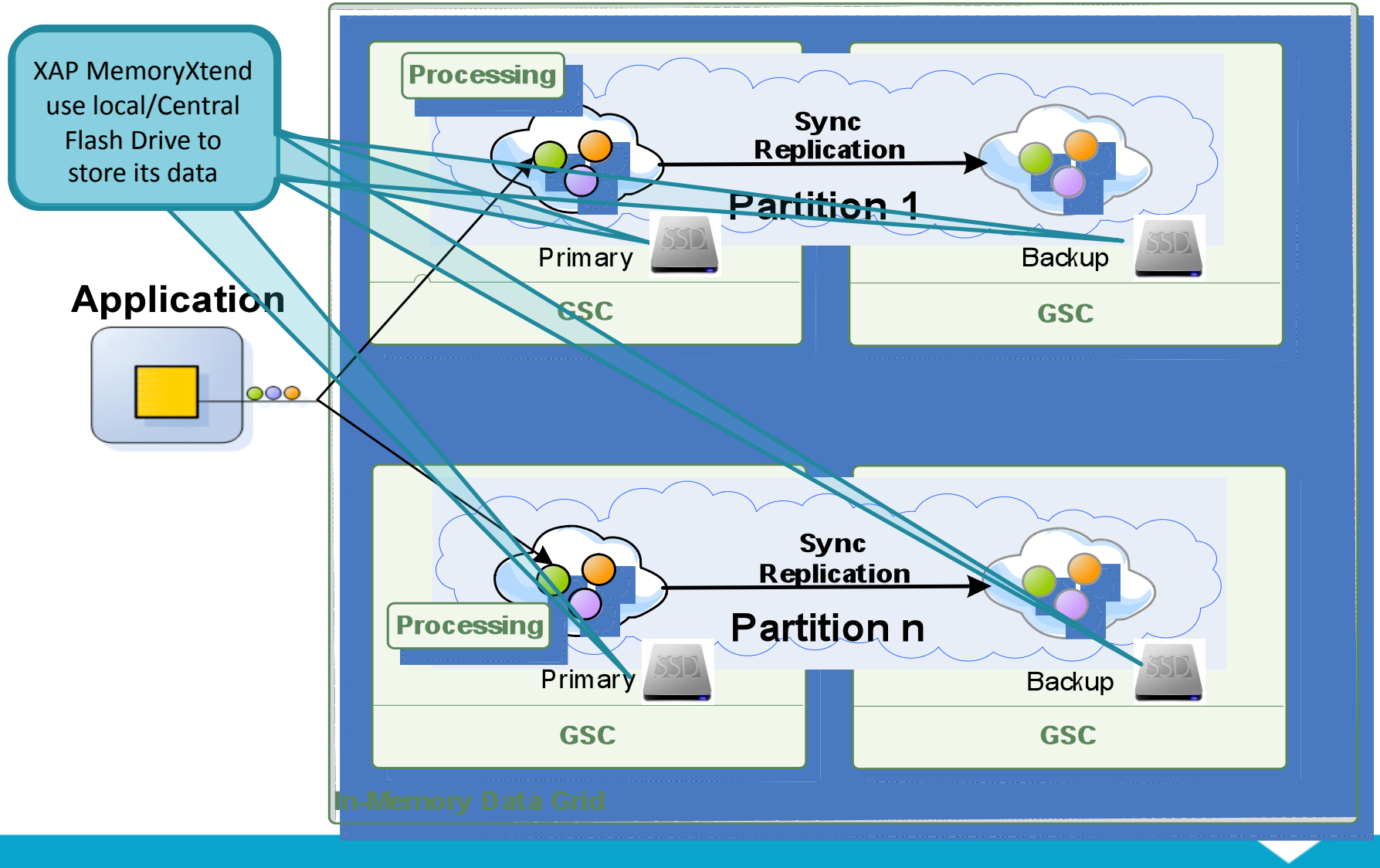
- http://docs.gigaspaces.com/xap100adm/blobstore-cache-policy.html

- http://docs.gigaspaces.com/faq/blobstore-cache-policy-faq.html

XAP MemoryXtend

- **Deploy high capacity Data Grid with minimal RAM utilization**
- **No lock-in**
  - All Enterprise flash drives are supported. SanDisk, Fusion-IO, Intel® SSD , etc are supported with the XAP IMC-SSD technology. Central SSD (RAID) devices such as Tegile, Cisco Whiptail, DSSD, and Violin Memory are also supported.
- **All data access routines supported**
  - XAP IMC data retrieval via a key or via SQL is fully supported. IMC Data grid indexes are maintained on-heap (RAM) for fast update and access.
- **Interoperability - All XAP IMC APIs are supported.**
  - Including the Space API (POJO and Document), JDBC API, JPA API, JMS API, and Map API ,Rest API , .Net API and C++ API.
- **All data-grid clustering topologies supported**
  - Allows grid based SSD storage configuration with one-click deployment , including multi-cluster multi-data center configuration across remote geographies.
- **Extensive Management**
  - Vast number of statistics available in real-time for optimized SSD utilization , and fine tuning based on the application data access pattern.
- **Intelligent multi-level DRAM caching**
  - Configurable flash management algorithms to optimize different workloads
- **Various durability levels supported**
  - both write-through or writeback (write-behind) for maximum write performance.

XAP MemoryXtend – SSD based Data-Grid



Tutorial Structure

- Cloud Instance setup - 5 min
- Cloud Instance bootstrap - 5 min
- SW download – 5 min
- SW Install – 5 min
- XAP Configuration – 5 min
- XAP Startup – 5 min
- RAM vs. SSD Data Grid Benchmarks – 10 min

Running XAP MemoryXtend on the EC2 Cloud

Quick guide:

1. Create your EC2 Account
2. Login into the AWS Management Console
3. Select the AMI to start and configure it
4. Start the Instance
5. Download XAP 10 , blobstore RPM and JDK
6. Install JDK
7. Install XAP 10
8. Install XAP blobstore RPM
9. Configure blobstore data grid
10. Start the XAP agent
11. Deploy RAM and SSD Data Grids
12. Run your tests

BlobStore Configuration

```
<blob-store:sandisk-blob-store id="sandisk"
        blob-store-capacity-GB="50"
        blob-store-cache-size-MB="20“
       devices="/dev/sdc1,/dev/sdc2,/dev/sdc3,/dev/sdc4"
       volume-dir="/data${clusterInfo.runningNumber}"
       durability-level="PERIODIC"
       blob-store-reformat="true">
 </blob-store:sandisk-blob-store>
```

Simple and elegant config

20MB will be dedicated to off-heap Cache

Device path – first one found being used

Symlink mapped to device generated file

```
<os-core:space id="space" url="/./myDataGrid">
        <os-core:blob-store-data-policy
            blob-store-handler="sandisk"
            cache-entries-percentage="10"
            avg-object-size-KB="1"/>
</os-core:space>
```

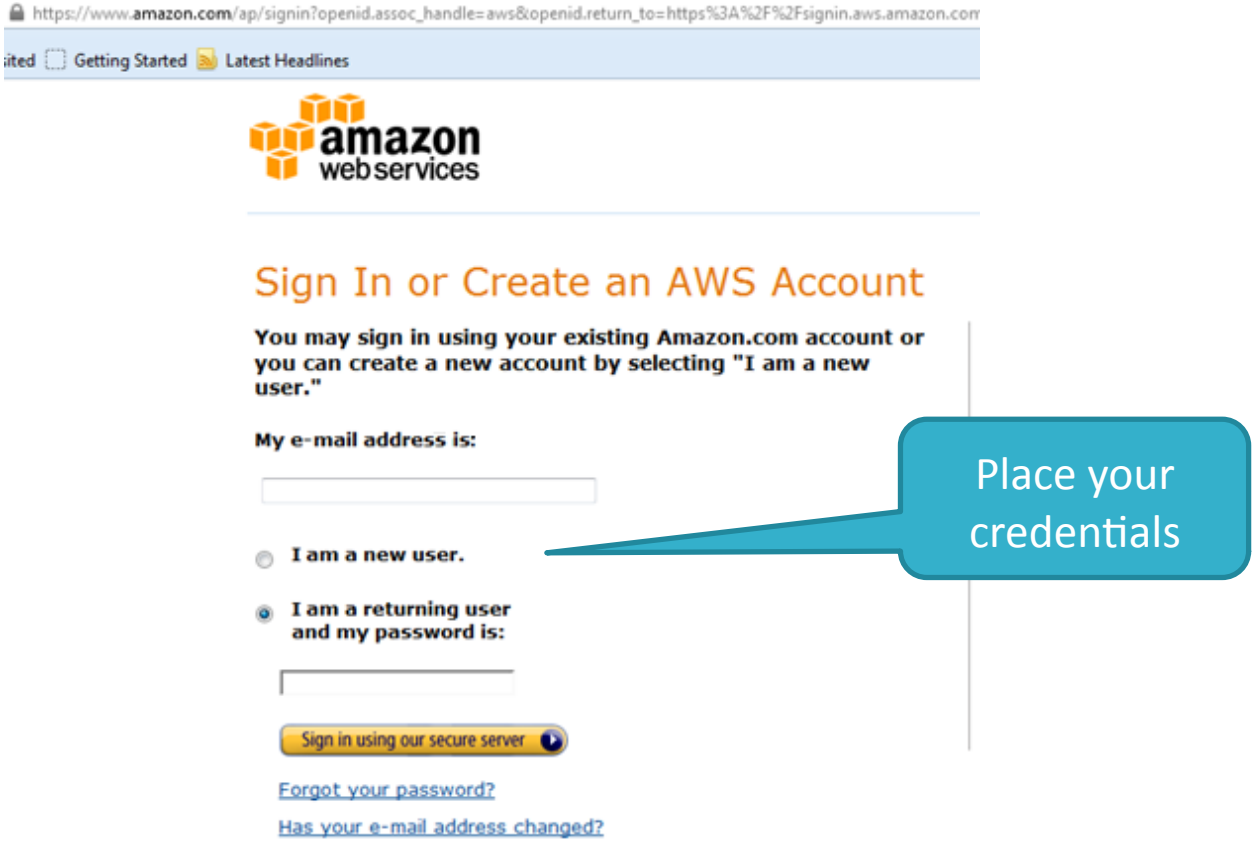
Around 10% of the heap will be dedicated to on-heap Cache

We will use EC2 to start a VM with a Flash Drive.

You may use any other machine running Linux 6.x with SSD Flash Drive with this tutorial.

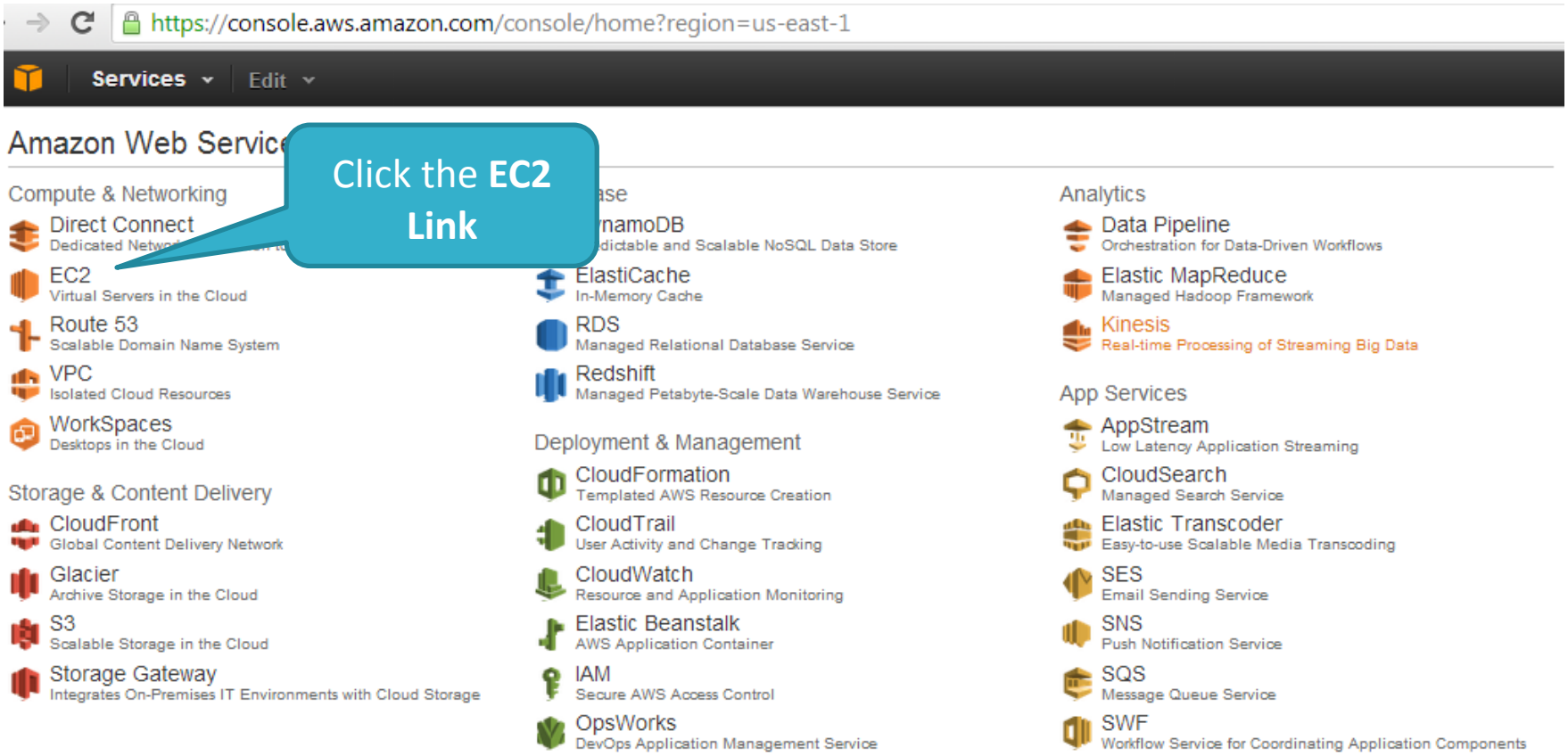
Login into the AWS Management Console

- **Open http://aws.amazon.com/console/**

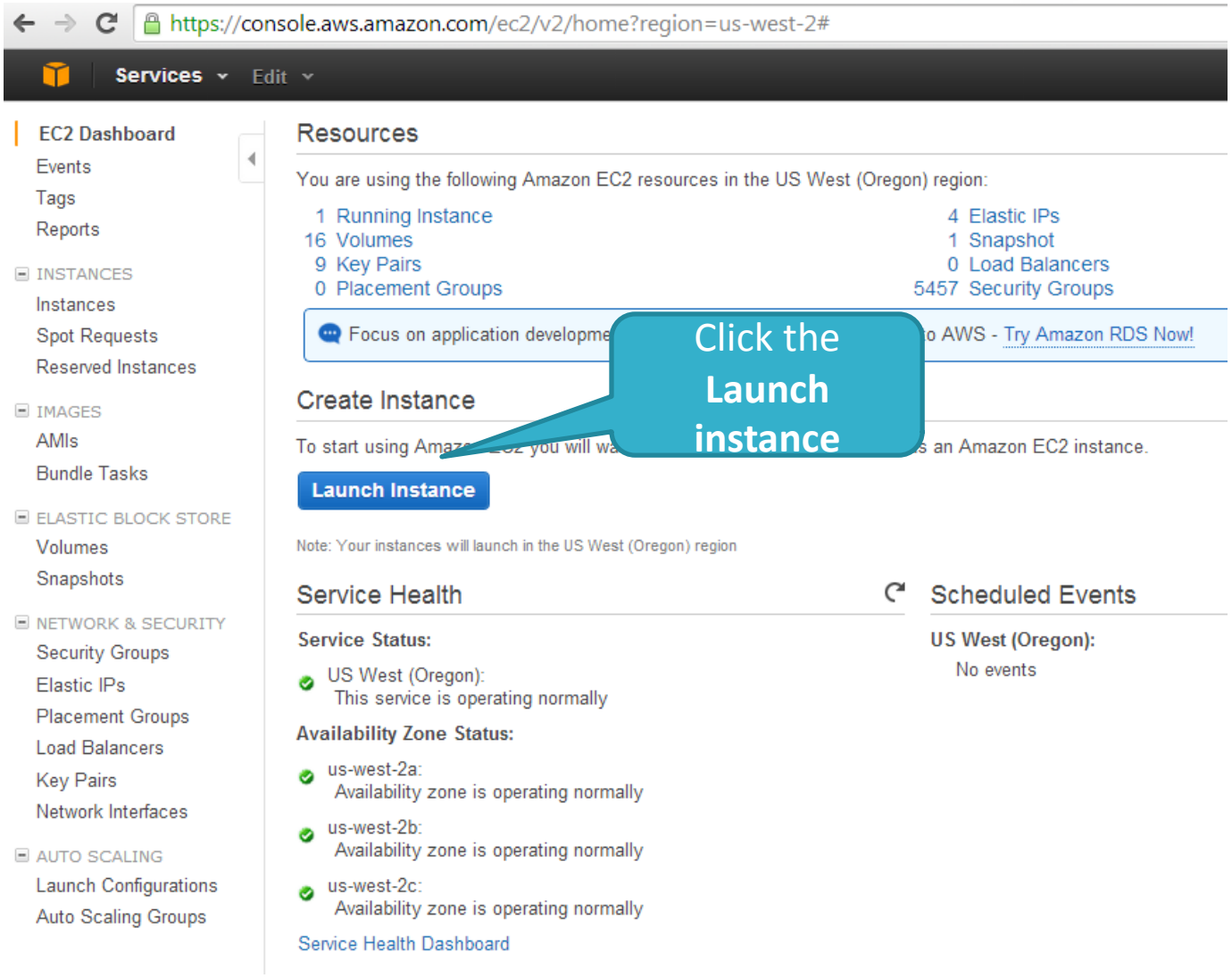


GIGASPACES

Access the EC2 Service panel

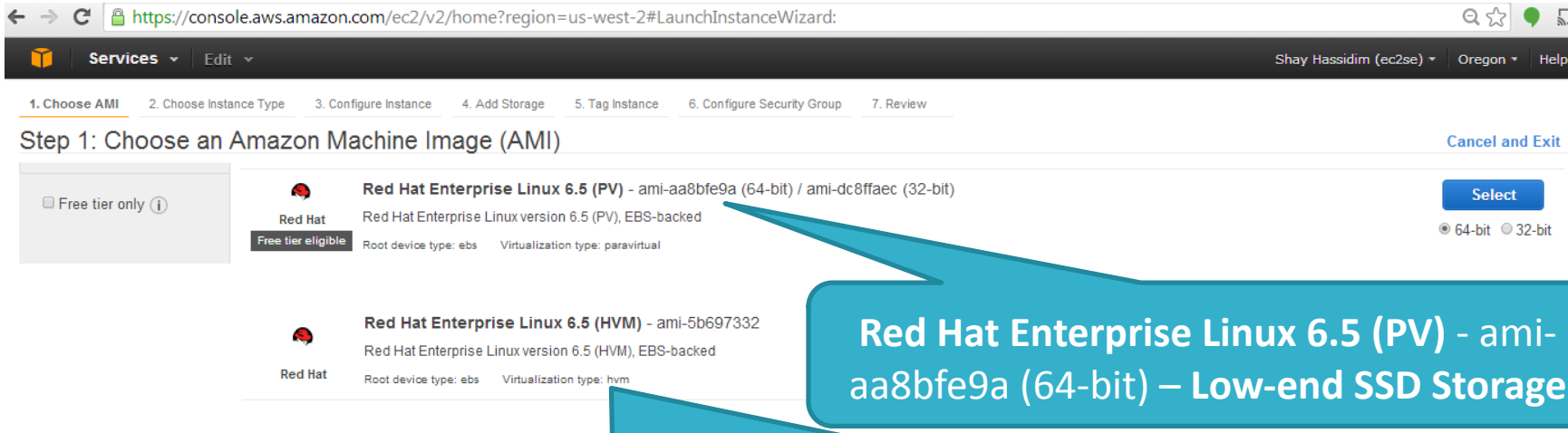


Start the New Instance Wizard



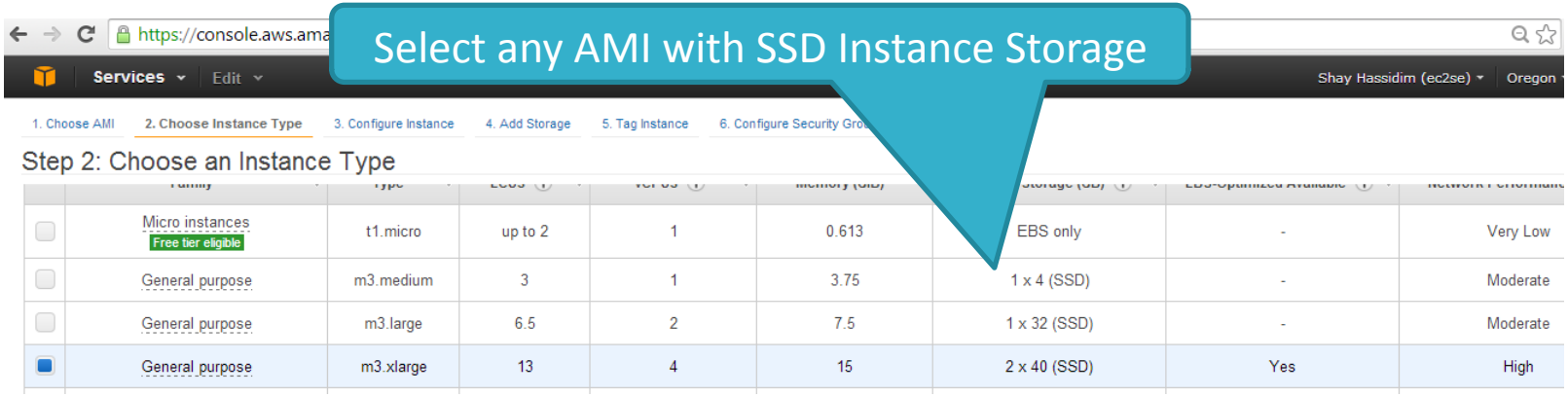
GIGASPACES

Choose an Amazon Image



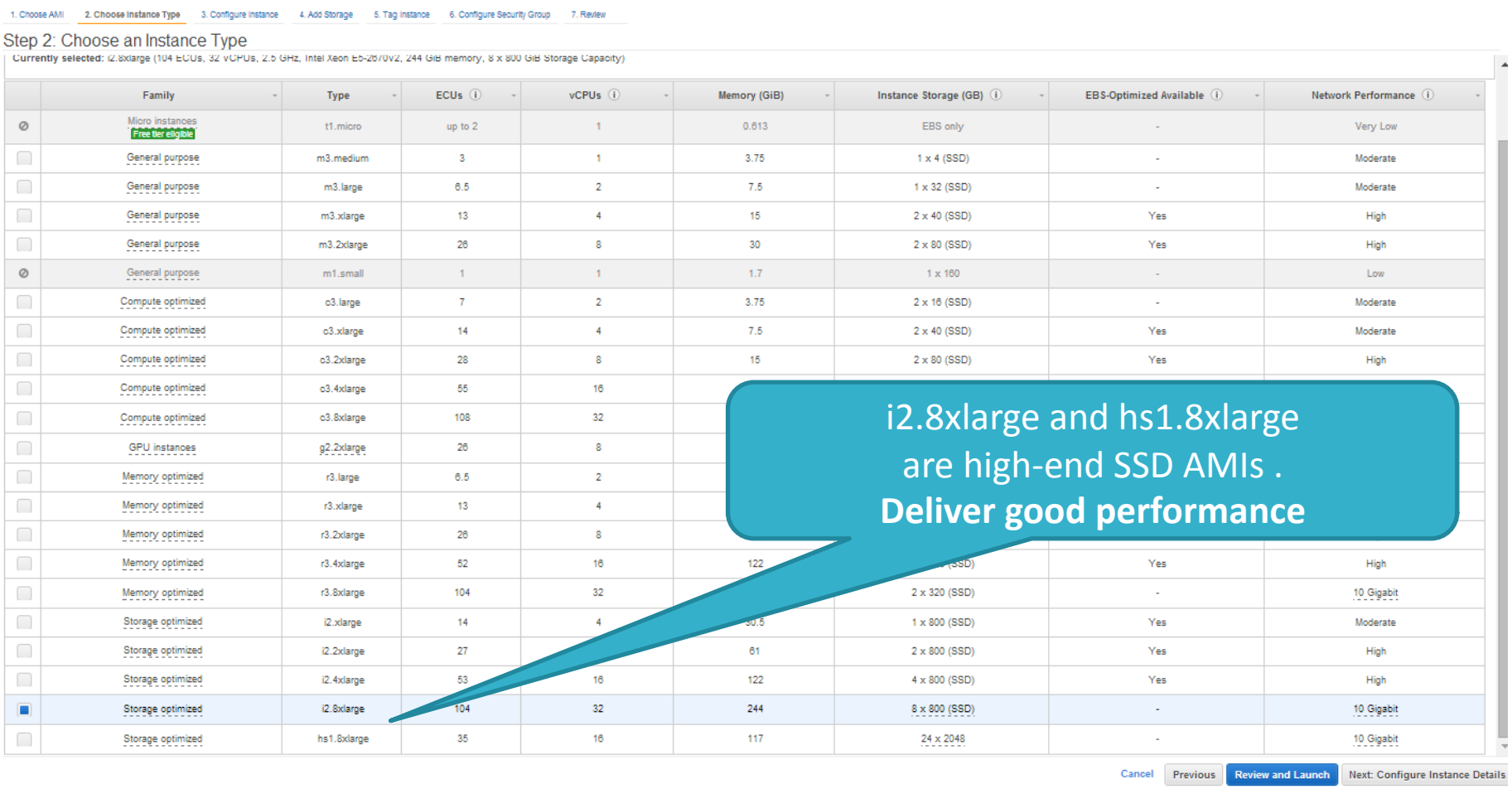
Red Hat Enterprise Linux 6.5 (PV) - ami-aa8bfe9a (64-bit) – **Low-end SSD Storage**

Red Hat Enterprise Linux 6.5 (HVM) - ami-5b697332 – **High-End SSD storage** – require prior registration



GIGASPACES

Choose an Instance Type

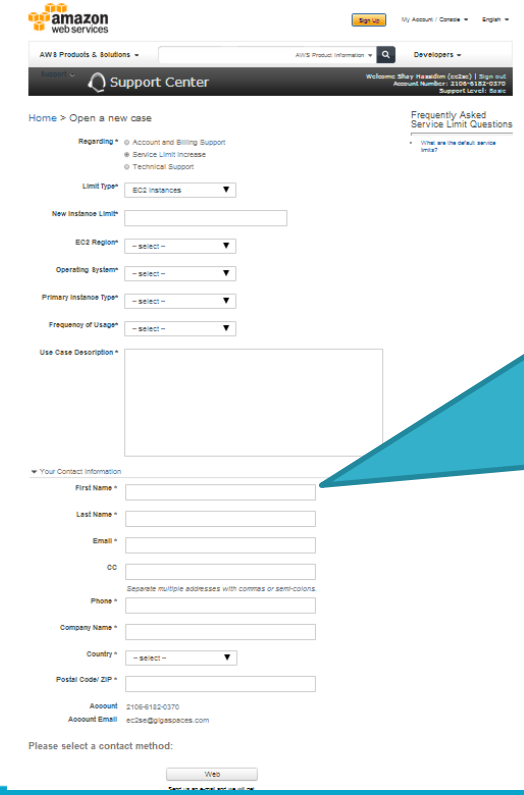


i2.8xlarge and hs1.8xlarge
are high-end SSD AMIs .
Deliver good performance

GIGASPACES

Increase your i2.8xlarge and hs1.8xlarge AMI Service limit

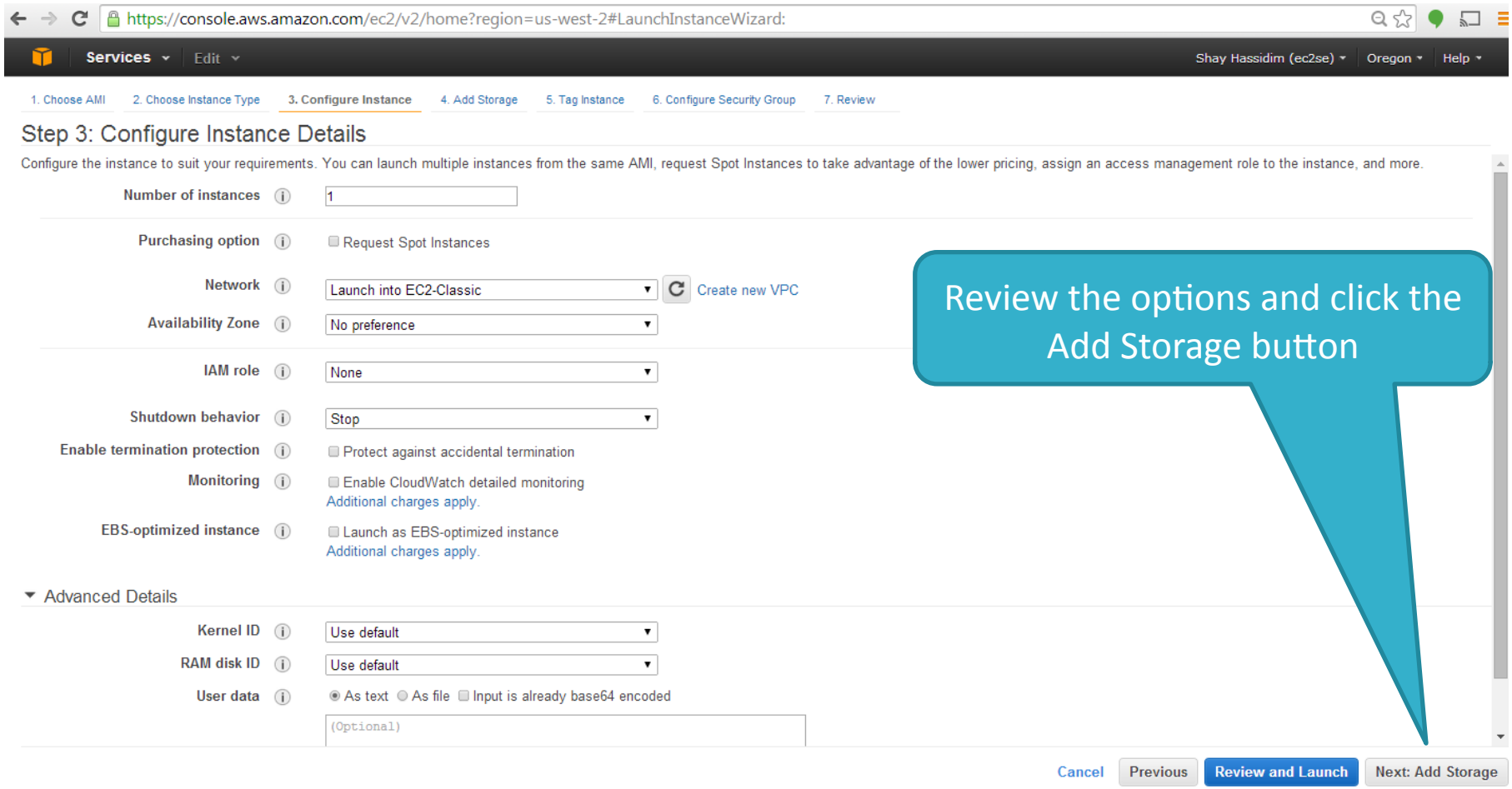
- **https://aws.amazon.com/support/createCase?serviceLimitIncreaseType=ec2-instances&type=service_limit_increase**



By default the i2.8xlarge and hs1.8xlarge are not available. You will need to fill in the form and request several instances. It takes about 2 days getting the approval.

GIGASPACES

Configure Instance Details



GIGASPACES

Add Storage

1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | **4. Add Storage** | 5. Tag Instance | 6. Configure Security Group | 7. Review

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

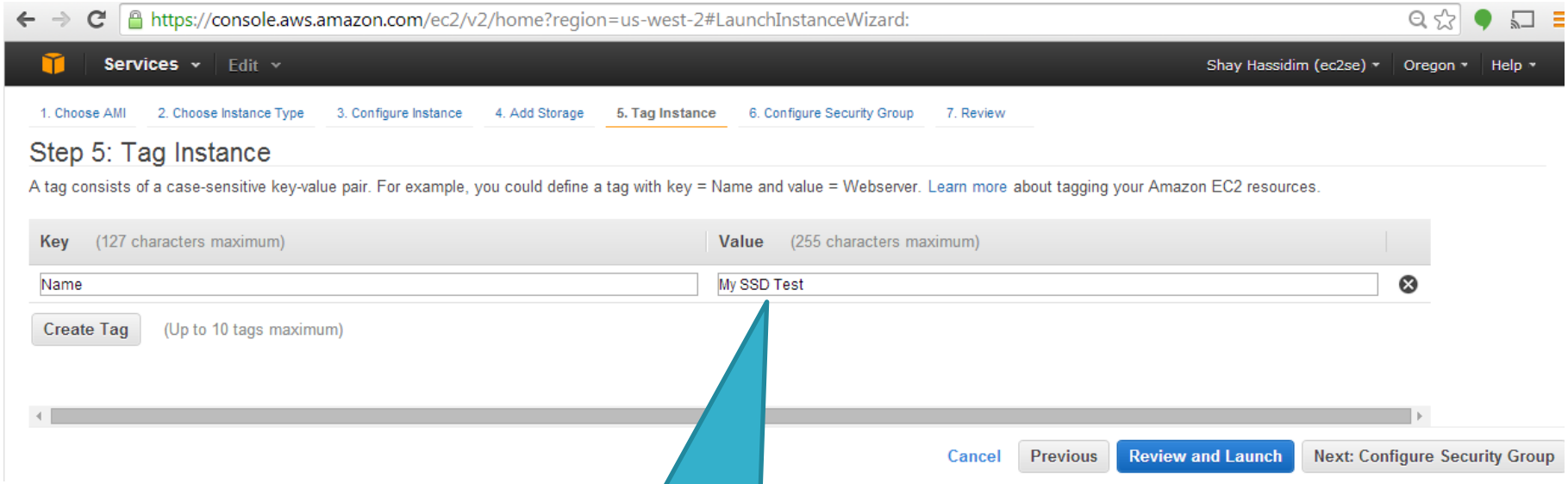
Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted	
Root	/dev/sda1	snap-d353ee0c	10	Standard	N/A	☑	Not Encrypted	
Instance Store 0	/dev/sdb	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 1	/dev/sdc	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 2	/dev/sdd	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 3	/dev/sde	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 4	/dev/sdf	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 5	/dev/sdg	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 6	/dev/sdh	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗
Instance Store 7	/dev/sdi	N/A	N/A	N/A	N/A	☐	Not Encrypted	⊗

Add New Volume

Instance Store 0-7– will be used with the **blobstore device configuration**

GIGASPACES

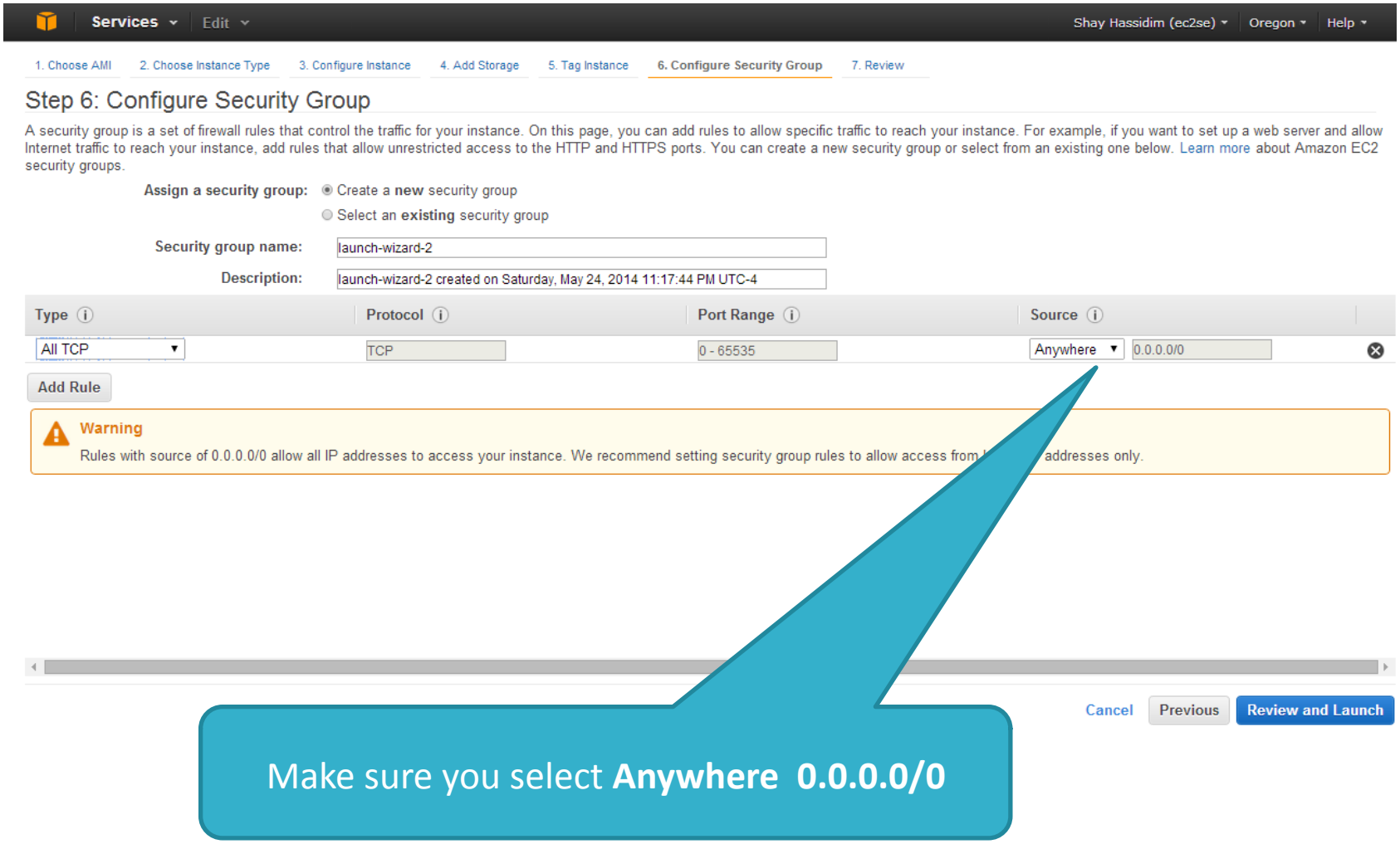
Tag the Instance



Will allow you to identify your instances

GIGASPACES

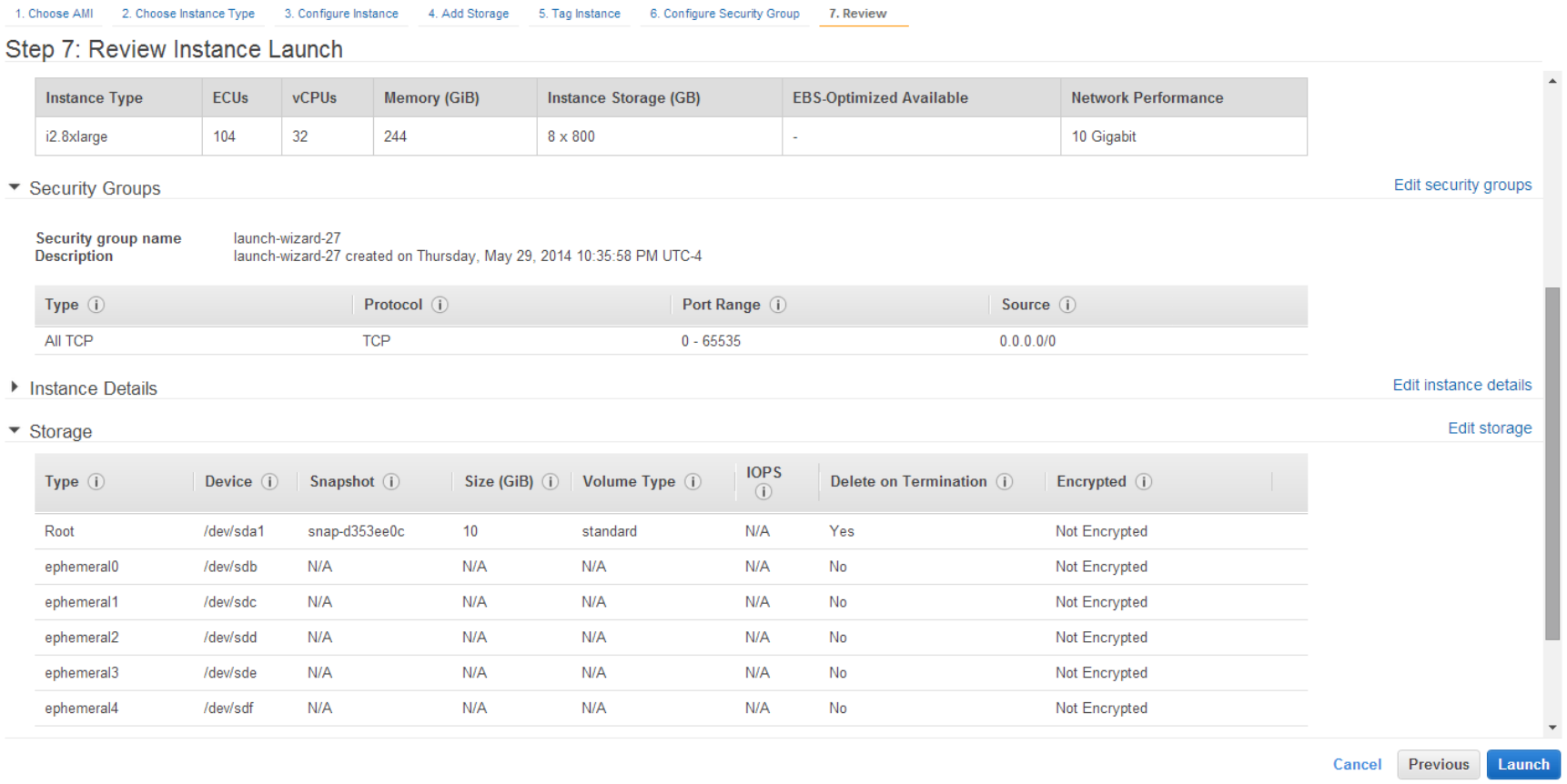
Configure Security Group



Make sure you select **Anywhere 0.0.0.0/0**

GIGASPACES

Review and Launch



1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Tag Instance | 6. Configure Security Group | 7. Review

Step 7: Review Instance Launch

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
i2.8xlarge	104	32	244	8 x 800	-	10 Gigabit

Security Groups — Edit security groups

Security group name launch-wizard-27
Description launch-wizard-27 created on Thursday, May 29, 2014 10:35:58 PM UTC-4

Type	Protocol	Port Range	Source
All TCP	TCP	0 - 65535	0.0.0.0/0

Instance Details — Edit instance details

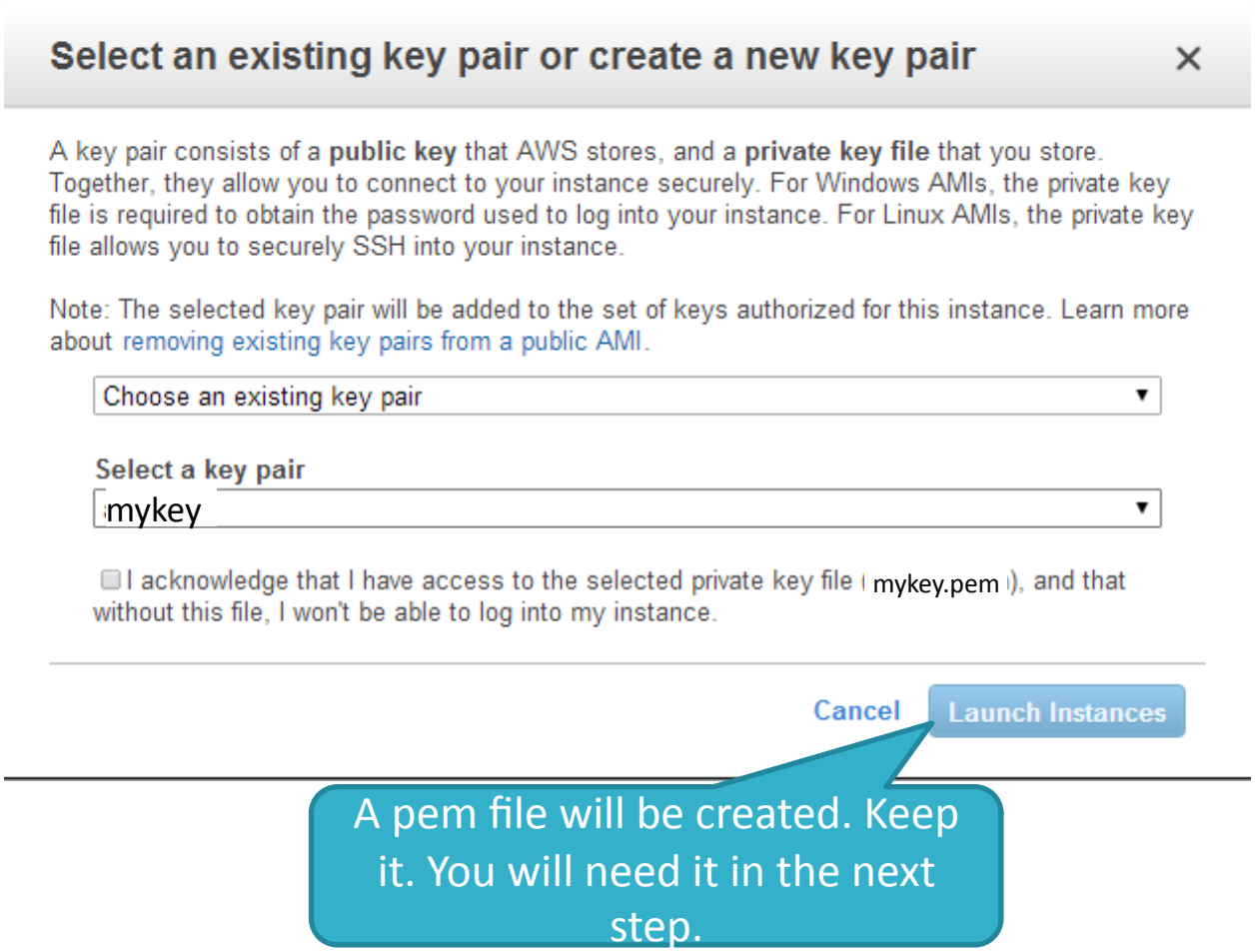
Storage — Edit storage

Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/sda1	snap-d353ee0c	10	standard	N/A	Yes	Not Encrypted
ephemeral0	/dev/sdb	N/A	N/A	N/A	N/A	No	Not Encrypted
ephemeral1	/dev/sdc	N/A	N/A	N/A	N/A	No	Not Encrypted
ephemeral2	/dev/sdd	N/A	N/A	N/A	N/A	No	Not Encrypted
ephemeral3	/dev/sde	N/A	N/A	N/A	N/A	No	Not Encrypted
ephemeral4	/dev/sdf	N/A	N/A	N/A	N/A	No	Not Encrypted

Cancel | Previous | Launch

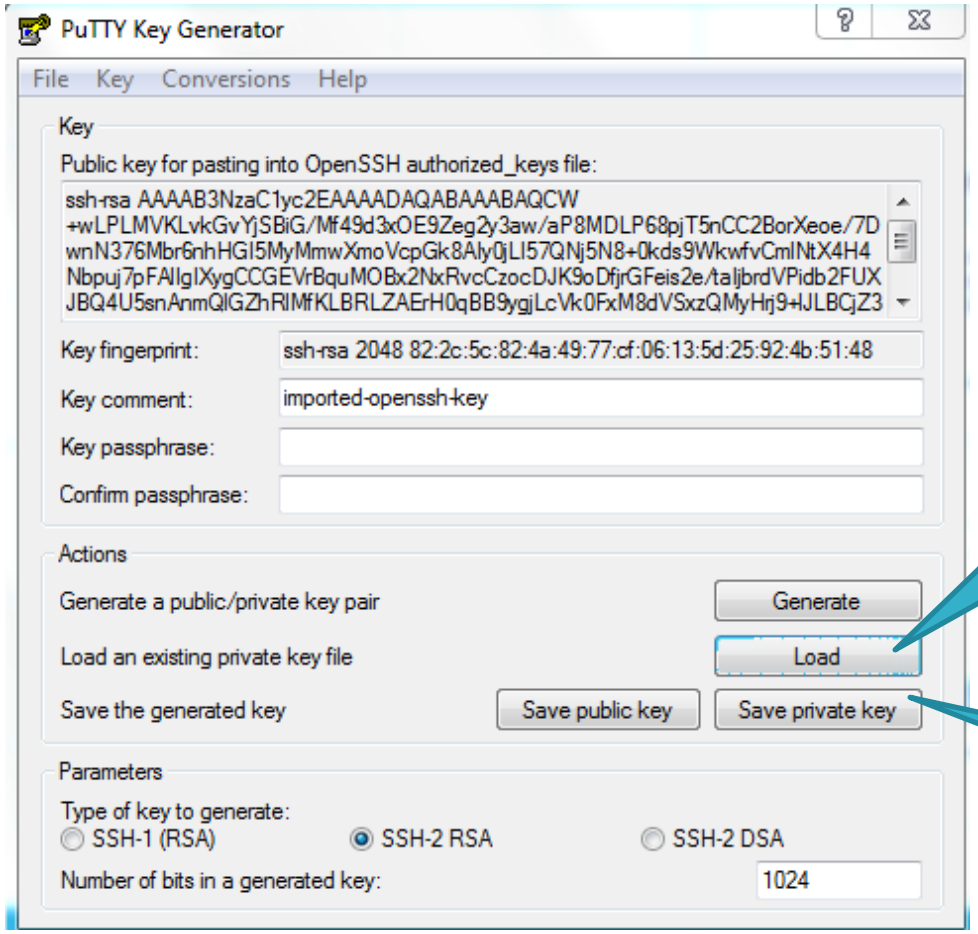
GIGASPACES

Keep your key pair



GIGASPACES

Create ppk file

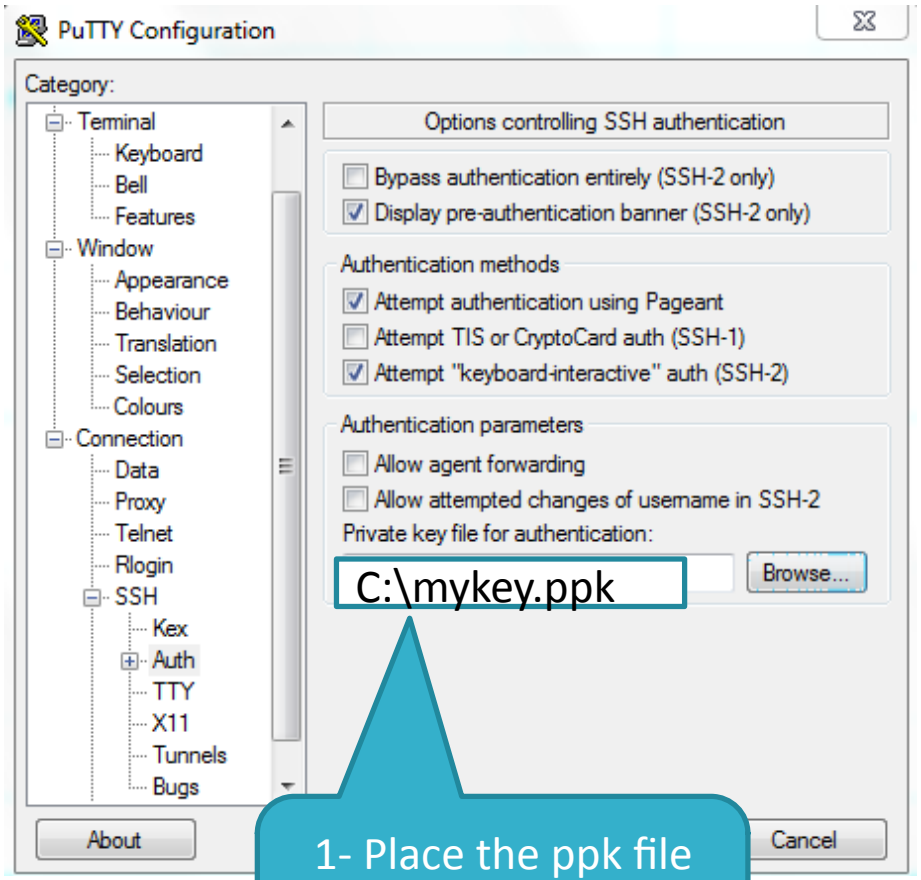


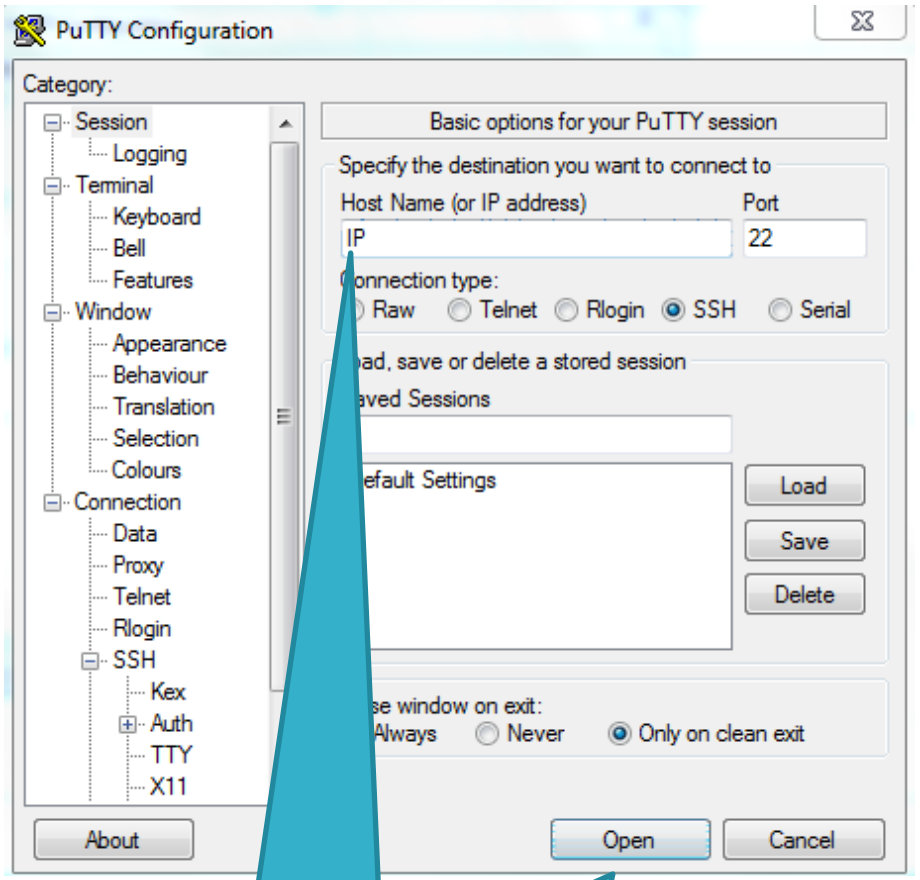
1- Click to load the pem file

2 - Click to generate a ppk file. Save the file.

GIGASPACES

Access the Instance





1- Place the ppk file location generated earlier

2 – place the instance public IP and click Open

GIGASPACES

What do you need to install?

- **One you have access to the instance on the cloud you will need to install the following:**
  - JDK 1.7 , 64 bit
  - XAP 10 - Java distribution
  - XAP 10 SanDisk ZetaScale libraries RPM

Sudo

- **Before installing move to root user using:**

> sudo -s

Download and Install JDK 1.7

- wget http://download.oracle.com/otn-pub/java/jdk/7u55-b13/jdk-7u55-linux-x64.tar.gz
- tar zxf jdk-7u55-linux-x64.gz
- Have the JDK installed under :

/home/ec2-user/jdk-7u55

See :

http://tecadmin.net/steps-to-install-java-on-centos-5-6-or-rhel-5-6

Download and Install XAP 10

- **wget http://www.gigaspaces.com/download_files/10/ga/gigaspaces-xap-premium-10.0.0XXX.zip**
- **unzip gigaspaces-xap-premium-10.0.0XXX.zip**
- **Change the *<u>/home/ec2-user/</u>*gigaspaces-xap-premium-10.0.0XXX/bin/setenv.sh to include:**

  export JAVA_HOME=/home/ec2-user/jdk-7u55

Download and Install XAP 10 BlobStore RPM

 # wget
http://www.gigaspaces.com/download_files/10/ga/blobstore-10.0.0XXX.noarch.rpm

 # sudo XAP_HOME=/home/ec2-user/gigaspaces-xap-premium-10.0.0XXX sh -c "rpm -ivh blobstore-10.0.0XXX.noarch.rpm"

Preparing... ### [100%]
1:blobstore ### [100%]

Getting the Device List – Low End SSD

 # fdisk -l

Disk /dev/xvda1: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

This is the OS drives. Please **DO NOT** use it with the blobstore configuration

Disk /dev/xvdb: 40.3 GB, 40256929792 bytes
255 heads, 63 sectors/track, 4894 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

This is the **device you should use** with the blobstore configuration

Getting the Device List – High End SSD

```
sudo –s
# fdisk -l

Disk /dev/xvda: 10.7 GB, 10737418240 bytes
97 heads, 17 sectors/track, 12717 cylinders
Units = cylinders of 1649 * 512 = 844288 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0003b587

   Device Boot      Start         End      Blocks   Id  System
/dev/xvda1   *           2        7632     6291456   83  Linux

Disk /dev/xvdb: 800.2 GB, 800165027840 bytes
255 heads, 63 sectors/track, 97281 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/xvdc: 800.2 GB, 800165027840 bytes
…
Disk /dev/xvdd: 800.2 GB, 800165027840 bytes
….
Disk /dev/xvde: 800.2 GB, 800165027840 bytes
…
Disk /dev/xvdf: 800.2 GB, 800165027840 bytes
…
Disk /dev/xvdg: 800.2 GB, 800165027840 bytes
…
Disk /dev/xvdh: 800.2 GB, 800165027840 bytes
…
Disk /dev/xvdi: 800.2 GB, 800165027840 bytes
…
```

These are the OS drives. Please **DO NOT** use these with the blobstore configuration

These are the available **devices when adding storage instance. These should be used** with the blobstore configuration

GIGASPACES

The Blobstore PU

- XAP 10 Blobstore RPM comes with a blobstore PU template.
- You will find it at:

/home/ec2-user/gigaspaces-xap-premium-10.0.0XXX/deploy/templates/blobstoreDataGrid folder

- Copy this folder into */home/ec2-user/gigaspaces-xap-premium-10.0.0XXX*/deploy/SSD-DataGrid to customize it.

The SSD-DataGrid pu.xml

Edit the *home*\ec2-user\gigaspaces-xap-premium-10.0.0XXX*\deploy\SSD-DataGrid\META-INF\spring\pu.xml to include the device list:

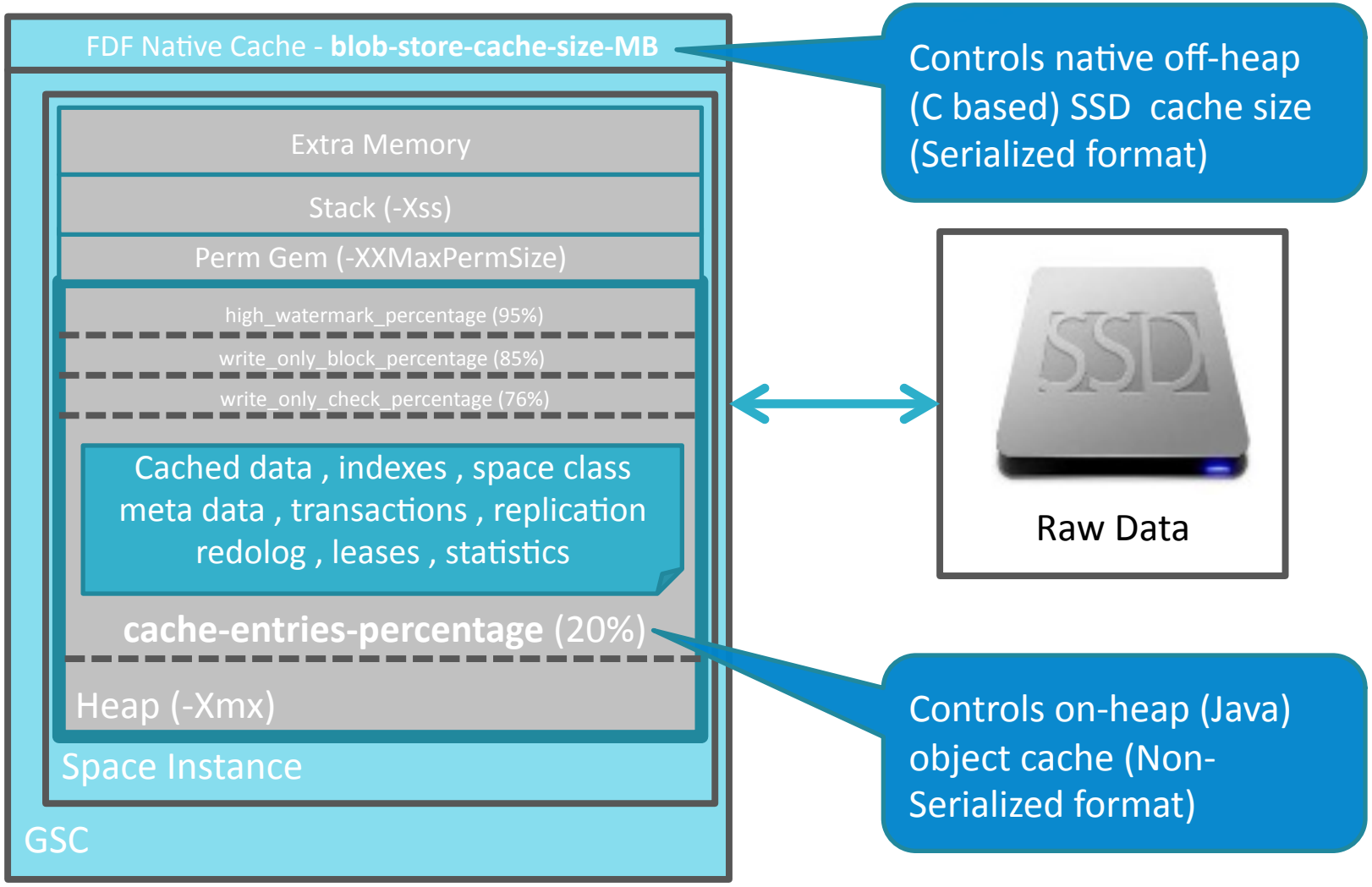
```
<blob-store:sandisk-blob-store id="sandiskBlobStore"
        blob-store-capacity-GB="100"
        blob-store-cache-size-MB="100"
        devices="/dev/xvdb,/dev/xvdc,/dev/xvdd,/dev/xvde, /dev/xvdf,/
dev/xvdg, /dev/xvdh,/dev/xvdi"
        volume-dir="/tmp/blobstore/data${clusterInfo.runningNumber}"
        durability-level="SW_CRASH_SAFE">
</blob-store:sandisk-blob-store>

<os-core:space id="space" url="/./SSD-DataGrid" >
  <os-core:blob-store-data-policy
        blob-store-handler="sandiskBlobStore"
        cache-entries-percentage="1"
        avg-object-size-KB="10"
        recover-from-blob-store="false"/>
</os-core:space>
```

The device instance drives

We will allocate 1% for on-heap cache

We will use 10K objects with our benchmarks

BlobStore Main Configuration



GIGASPACES

SanDisk BlobStore specific Properties

Property	Description	Default	Use
devices	Flash devices. Comma separated available devices. The list used as a search path from left to right. The first one exists will be used.		required
volume-dir	Directory path contains a symbolic link to the the SSD device.		required
blob-store-capacity-GB	Flash device allocation size in Gigabytes.	200	optional
blob-store-cache-size-MB	ZetaScale internal LRU based off-heap in-process cache size in Megabytes. Keeps data in serialized format.	100	optional
write-mode	WRITE_THRU - the data grid writes the data immediately into the blobstore and synchronously acknowledge the write after ZetaScale fully commits the operation.WRITE_BEHIND - the data grid writes the data immediately into the blobstore. ZetaScale asynchronously commits the operation to the SSD. This option improves write performance but may have a consistency issue with a sudden hardware failure.	WRITE_THRU	optional
enable-admin	ZetaScale admin provides a simple command line interface (CLI) through a TCP port. ZetaScale CLI uses port 51350 by default. This port can be changed through the configuration parameter FDF_ADMIN_PORT.	false	
statistics-interval	Applications can optionally enable periodic dumping of statistics to a specified file (XAP_HOME/logs). This is disabled by default.		optional
durability-level	SW_CRASH_SAFE - Guarantees no data loss in the event of software crashes. But some data might be lost in the event of hardware failure. HW_CRASH_SAFE- Guarantees no data loss if the hardware crashes.Since there are performance implication it is recommended to work with NVRAM device and configure log-flash-dir to a folder on this device.	SW_CRASH_SAFE	optional
log-flush-dir	When HW_CRASH_SAFE used , point to a directory in a file system on top of NVRAM backed disk. This directory must be unique per space, you can add ${clusterInfo.runningNumber} as suffix to generate a uniqee name	as volume-dir	optional
Property	Description	Default	Use

IMDG generic BlobStore settings

Property	Description	Default	Use
blob-store-handler	BlobStore implementation		required
cache-entries-percentage	On-Heap cache stores objects in their native format. This cache size determined based on the percentage of the GSC JVM max memory(-Xmx). If -Xmx is not speficied the cache size default to10000 objects. This is an LRU based data cache.	20%	optional
avg-object-size-KB	Average object size.	5KB	optional
recover-from-blob-store	Whether to recover from blob store or not		required

Start GigaSpaces blobstore agent and Web UI server

- cd */home/ec2-user/gigaspaces-xap-premium-10.0.xxx*/bin

- Edit the *gs-agent-blobstore.sh and* set the GSC heap size:

  GSC_JAVA_OPTIONS="-Xmx30g –Xms30g -Dcom.gigaspaces.grid.gsc.serviceLimit=1"; export GSC_JAVA_OPTIONS

- Start the *blobstore* agent with 2 GSCs:

  ./gs-agent-blobstore.sh gsa.gsc 2 &

Start the We UI server:

./gs-webui.sh &

Login to XAP Web-Console

Start your browser and point it to: Public-Instance-IP:8099

Hit the **Login** button to access the XAP dashboard.

If you have problem you might haven't set the right security group correctly

Deploy RAM and SSD Data Grid via CLI

- *> cd /home/ec2-user/gigaspaces-xap-premium-10.0.0XXX/*bin
- > gs.sh **deploy-space** RAM-DataGrid
- > gs.sh **deploy** SSD-DataGrid

Check the symlink created

Once deployed - each blobstore data grid instance will have a symlink created mapped to available drive:

[root@zeppo bin]# ls /tmp/blobstore/data0 -il

total 0

135048 lrwxrwxrwx. 1 root root 9 Aug 1 08:58 SSD-DataGrid_container-SSD-DataGrid -> /dev/xvdb

Run Tests – The benchmark application

- **Access the benchmark application**

  cd /home/ec2-user/gigaspaces-xap-premium-10.0.0XXX/tools/benchmark/bin

- **Create 4 copies of runTest.sh:**

  > cp runTest.sh runSSD-DataGridWrite.sh

  > cp runTest.sh runSSD-DataGridRead.sh

  > cp runTest.sh runRAM-DataGridWrite.sh

  > cp runTest.sh runRAM-DataGridRead.sh

Write Benchmark Command

- **Modify the runSSD-DataGridWrite.sh to have:**

  Java com…..BenchmarkTest "jini://localhost//**SSD-DataGrid**" -execute **first** -s 10000 -showrate 50000 -tr 10 -i 200000 $**

- **Modify the runRAM-DataGridWrite.sh to have**

  Java com…..BenchmarkTest "jini://localhost//**RAM-DataGrid**" -execute **first** -s 10000 -showrate 50000 -tr 10 -i 200000 $**

Read Benchmark Command

- **Modify the runSSD-DataGridRead.sh to have:**

  Java com…..BenchmarkTest "jini://localhost//**SSD-DataGrid**" -execute **second** -s 10000 -showrate 50000 -tr 10 -i 200000 $**

- **Modify the runRAM-DataGridRead.sh to have**

  Java com…..BenchmarkTest "jini://localhost// **RAM-DataGrid**" -execute **second** -s 10000 -showrate 50000 -tr 10 -i 200000 $**

Run the Write Benchmark

>./runSSD-DataGridWrite.sh

main - This Test will perform WRITE

main - MASTER SPACE URL: jini://localhost/*/SSD-DataGrid

…

main - ---------- WRITE SUMMARY ----------

main - WRITE AVG TEST TIME for all threads = 116381.100 ms

main - WRITE AVG TP for all threads = 1718.493 msg/sec

main - WRITE TOTAL TP for all threads = **17184.928** msg/sec , 163.888 MB/sec

>./runRAM-DataGridWrite.sh

main - This Test will perform WRITE

main - MASTER SPACE URL: jini://localhost/*/RAM-DataGrid

…

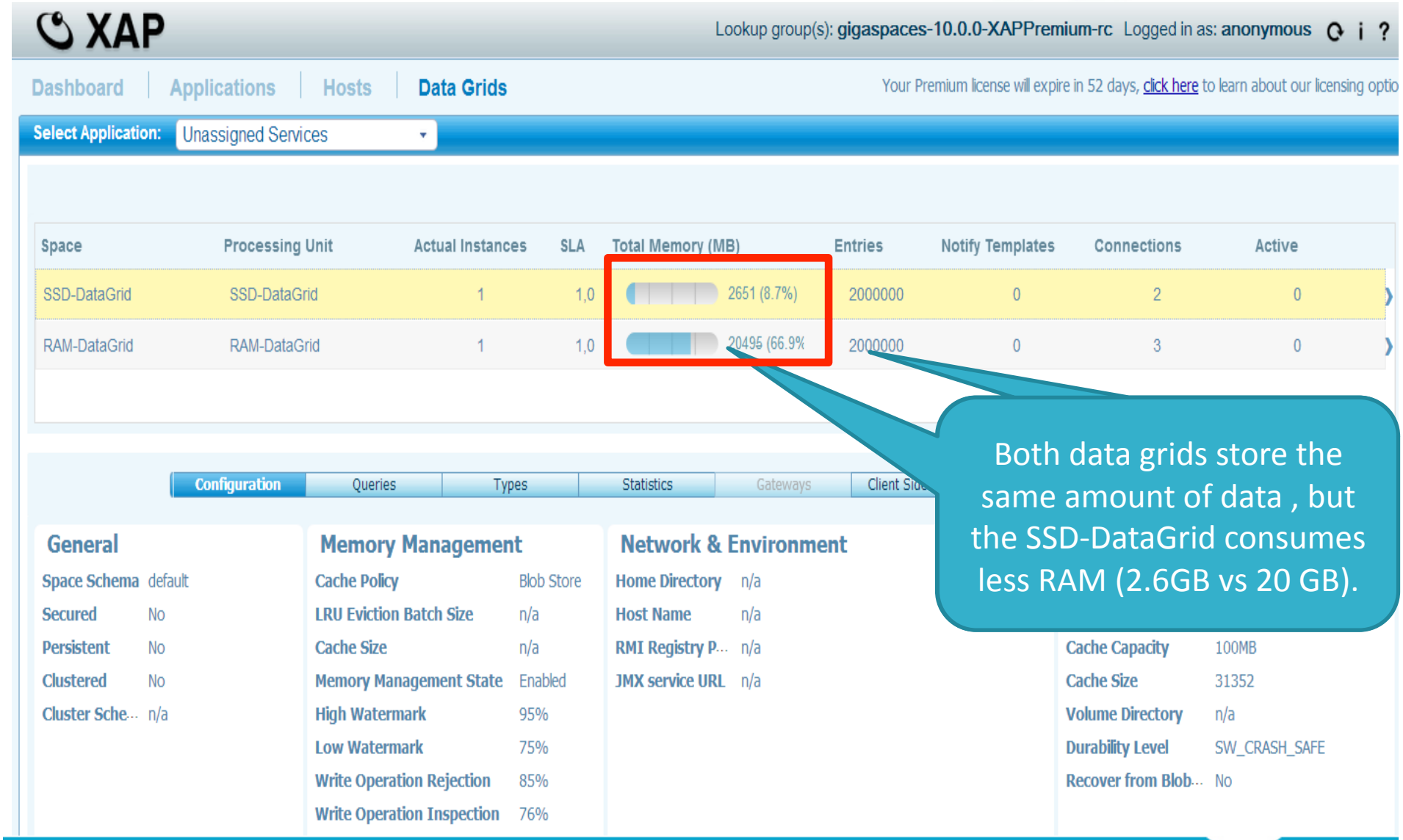
main - ---------- WRITE SUMMARY ----------

main - WRITE AVG TEST TIME for all threads = 91759.000 ms

main - WRITE AVG TP for all threads = 2179.627 msg/sec

main - WRITE TOTAL TP for all threads = **21796.270** msg/sec , 207.865 MB/sec

Compare RAM Data Grid to SSD DataGrid



Both data grids store the same amount of data , but the SSD-DataGrid consumes less RAM (2.6GB vs 20 GB).

GIGASPACES

Run the Read Benchmark

>./runSSD-DataGridRead.sh

main - This Test will perform READ

main - MASTER SPACE URL: jini://localhost/*/SSD-DataGrid

…

main - ---------- READ SUMMARY ----------

main - READ AVG TEST TIME for all threads = 194574.500 ms

main - READ AVG TP for all threads = 1027.884 msg/sec

main - READ TOTAL TP for all threads = **10278**.843 msg/sec , 98.027 MB/sec

>./runRAM-DataGridRead.sh

main - This Test will perform READ

main - MASTER SPACE URL: jini://localhost/*/RAM-DataGrid

…

main - ---------- READ SUMMARY ----------

main - READ AVG TEST TIME for all threads = 85467.700 ms

main - READ AVG TP for all threads = 2340.074 msg/sec

main - READ TOTAL TP for all threads = **23400.741** msg/sec , 223.167 MB/sec

Benchmark options

- **For all benchmark options:**

  > runTest.sh –h

- **Popular options:**

-i [number of iterations]	number of iterations; default is 1000
-tr [number of threads] operation	number of threads performing each
-s	payload size in bytes
-execute first second (without removing data)	(instead of –all) – will perform write and read
- clean	clear data before running benchmark

Automatic Data Recovery from SSD

Automatic Data Recovery and Re-Indexing

- You may un-deploy the data grid , deploy and reload Indexes by enabling *recover-from-blob-store property.*
- You should construct sla.xml that lists the machines running SSD and the data grid nodes.
- With 8 cores server running 4 partitions with four drives , 100,000 items / second (1K payload) may be scanned and indexed.

Enabling Data Recovery and Re-Indexing

Modify the SSD-Data-Grid pu.xml to enable the recover-from-blob-store:

```
<os-core:space id="space" url="/./SSD-DataGrid" >
  <os-core:blob-store-data-policy
        blob-store-handler="sandiskBlobStore"
        cache-entries-percentage="1"
        avg-object-size-KB="10"
        recover-from-blob-store="true"/>
</os-core:space>
```

Create sla.xml

- The sla.xml should list all instances you have and their host.
- If you are running your tests with a single instance simply specify the same host for all instances.

Partitioned with a backup SLA – one partition

```
<os-sla:sla>
      <os-sla:instance-SLAs>
        <os-sla:instance-SLA instance-id="1">
          <os-sla:requirements>
            <os-sla:host ip="HostIP"/>
          </os-sla:requirements>
        </os-sla:instance-SLA>
                    <os-sla:instance-SLA instance-id="1" backup-id="1">
          <os-sla:requirements>
            <os-sla:host ip="HostB"/>
          </os-sla:requirements>
        </os-sla:instance-SLA>
      </os-sla:instance-SLAs>
</os-sla:sla>
```

Partitioned data grid sla.xml

```
<os-sla:sla>
    <os-sla:instance-SLAs>
        <os-sla:instance-SLA instance-id="1">
            <os-sla:requirements>
                <os-sla:host ip="HostIP"/>
            </os-sla:requirements>
        </os-sla:instance-SLA>

...     <os-sla:instance-SLA instance-id="4">
            <os-sla:requirements>
                <os-sla:host ip="HostIP"/>
            </os-sla:requirements>
        </os-sla:instance-SLA>
    </os-sla:instance-SLAs>
</os-sla:sla>
```

Place the Instance IP

Lets test data reload and ReIndexing

- Undeploy the existing SSD-DataGrid
- Deploy the SSD-DataGrid using the sla.xml
- Write some data via the benchmark

  runSSD-DataGridWrite.sh
- See the object count, check footprint
- Undeploy and terminate the agent
- Start the agent , Deploy the SSD-DataGrid
- Monitor the data reload process

BlobStore Available Statistics

- Counts of FDF access types
- Counts of various flash activities
- Histogram of key sizes
- Histogram of data sizes in bytes
- Histogram of access latencies in microseconds
- Number of events , Minimum , Maximum , Average , Geometric mean , Standard deviation
- Overwrite/Write---Through Statistics
- Total number of created objects
- Number of get/put/delete operations
- Number of hash/flash/invalid evictions
- Number of objects in flash
- Number of soft/hard overflows in hast table
- Number of pending IO's
- Flash space allocated/consumed in bytes
- Number of overwrites
- Number of hash collisions for get/set operations

Useful SSD activity monitoring tool - iostat

http://linuxcommand.org/man_pages/iostat1.html

EXAMPLES

iostat

Display a single history since boot report for all CPU and Devices.

iostat -d 2

Display a continuous device report at two second intervals.

iostat -d 2 6

Display six reports at two second intervals for all devices.

iostat -x hda hdb 2 6

Display six reports of extended statistics at two second inter-vals for devices hda and hdb.

iostat -p sda 2 6

Display six reports at two second intervals for device sda and all its partitions (sda1, etc.)

Thank You