Administration & Operations

Exercise and Useful Tools

AEROSPIKE

asadm

asadm

asadm is a command line tool used to track the health of an Aerospike cluster.

Typical syntax:

- `asadm [-h <host>[:<port>]] [-p <port>]`

This will put you into the asadm command line which looks like this:

`Admin>`

Hitting <TAB> will show you possible options.

Command: `help`

Displays the full syntax of the asadm command.

asadm – Commonly Used Commands

Command: `info`

Displays cluster info similar to the dashboard on the AMC.

```
Admin> info
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Service Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node    Build   Cluster      Cluster     Cluster    Free    Free   Migrates   Principal      Objects      Uptime
   .        .      Size   Visibility   Integrity   Disk%   Mem%          .           .            .           .
i       3.5.9         1   True         True            0     99    (0,0)      i            248.787 K   24:18:09
Number of rows: 1

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Network Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node              Node                             Fqdn                Ip  Client      Current       HB        HB
   .                Id                                .                 .   Conns         Time     Self   Foreign
i       *BB94FB6A4647106   ip-172-31-59-3.ec2.internal:3000   172.31.59.3:3000        2   170552385   581239         0
Number of rows: 1

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Namespace Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node    Namespace   Evictions     Objects      Repl     Stop    HWM          Mem      Mem    HWM       Stop
   .            .           .           .    Factor   Writes   Disk%        Used    Used%   Mem%   Writes%
i       bar                 0     0.000           1    false   50       0.000 B         0     60         90
i       test                0   248.787 K         1    false   50      43.901 MB        2     60         90
Number of rows: 2
```

asadm – Commonly Used Commands

Command: `show stat`

Displays node stats for each node in the cluster. You can select for a single set of statistics by choosing the statistic type:

- bins
- namespace
- service
- sets
- xdr (for Enterprise Edition)

```
Admin> show stat sets
~~~~~~~~test longevity Set Statistics~~~~~~~~
NODE                :   u10             u13
deleting            :   false           false
disable-eviction    :   false           false
memory_data_bytes   :   0               0
ns                  :   test            test
objects             :   55976084        57480531
set                 :   longevity       longevity
set-enable-xdr      :   use-default     use-default
stop-write-count    :   0               0
```

asadm – Commonly Used Commands

Command: `show stat`

Displays node stats for each node in the cluster. The output can be very long, so filter with the “like” modifer.

```
Admin> show stat like memory
~~~~~~~~~~~~~~~test testset Set Statistics~~~~~~~~~~~~~~~
NODE              :   ip-172-31-24-237.ec2.internal:3000
memory_data_bytes:   11893931

~~~~~~~~~~~~~~~~~~test Namespace Statistics~~~~~~~~~~~~~~~~~~
NODE                     :   ip-172-31-24-237.ec2.internal:3000
high-water-memory-pct    :   60
memory-size              :   4294967296
memory_free_pct          :   99
memory_used_bytes        :   18290850
memory_used_data_bytes   :   11894050
memory_used_index_bytes  :   6396800
memory_used_sindex_bytes:   0
sindex.data-max-memory   :   ULONG_MAX
```

asadm – Commonly Used Commands

Command: `show config`

Displays node configurations for each node in the cluster. The output can be made specific to specific areas:

- namespace
- network
- service
- xdr (for Enterprise Edition)

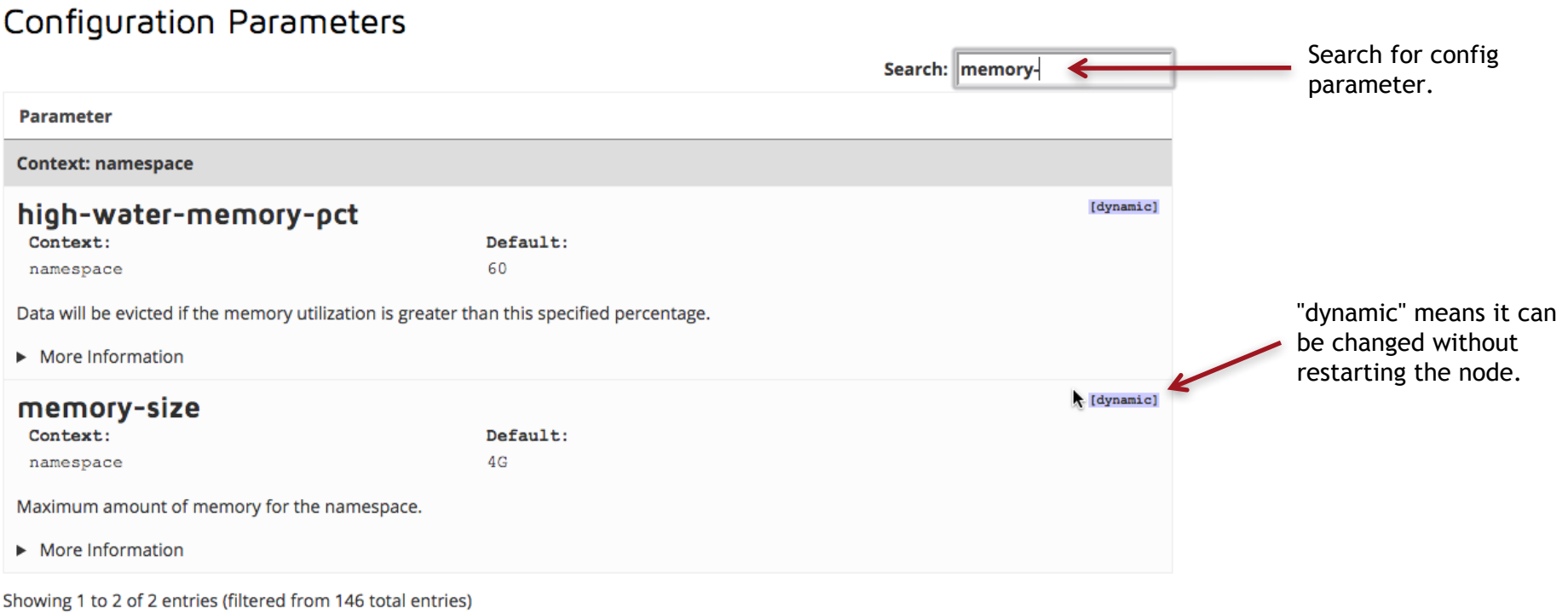
```
Admin> show config namespace
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ test Namespace Configuration~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
NODE                                    :   u12
allow-nonxdr-writes                     :   true
allow-xdr-writes                        :   true
cold-start-evict-ttl                    :   4294967295
conflict-resolution-policy              :   generation
data-in-index                           :   false
default-ttl                             :   2592000
disallow-null-setname                   :   false
enable-benchmarks-batch-sub             :   false
enable-benchmarks-read                  :   false
enable-benchmarks-storage               :   false
enable-benchmarks-udf                   :   false
enable-benchmarks-udf-sub               :   false
enable-benchmarks-write                 :   false
enable-hist-proxy                       :   false
enable-xdr                              :   false
evict-hist-buckets                      :   10000
evict-tenths-pct                        :   5
geo2dsphere-within.earth-radius-meters:   6371000
geo2dsphere-within.level-mod            :   1
geo2dsphere-within.max-cells            :   12
geo2dsphere-within.max-level            :   30
geo2dsphere-within.min-level            :   1
geo2dsphere-within.strict               :   true
high-water-disk-pct                     :   50
high-water-memory-pct                   :   60
...
```

asadm – Commonly Used Commands

Command: `asinfo -v`

Dynamically alters the configuration of the nodes in the cluster. The context will match the area in the configuration file. Not all variables are dynamically changeable.

Go to http://www.aerospike.com/docs/reference/configuration/



asadm – Commonly Used Commands

Example

Command: `asinfo -v`

To update the amount of memory (RAM) used by the namespace "test" to 2 GB without restarting the nodes in the cluster. Issue the following command. Note that all nodes will be changed. The configuration file will NOT be altered.

```
Admin> asinfo -v "set-config:context=namespace;id=test;memory-size=2G"
u12 (192.168.120.112) returned:
ok
u13 (192.168.120.113) returned:
ok
u10 (192.168.120.110) returned:
ok
```

asadm – Commonly Used Commands

Use **asinfo –v** within asadm to get configuration of all nodes, filter using **like**

- Admin> **asinfo -v "get-config" like nsup**
- 192.168.183.172:3000 (192.168.183.172) returned:
- nsup-delete-sleep=100;nsup-period=15;nsup-startup-evict=true

- Admin> **asinfo -v "namespace/test" like defrag**
- 192.168.183.172:3000 (192.168.183.172) returned:
- storage-engine.defrag-lwm-pct=50;storage-engine.defrag-queue-min=0;storage-engine.defrag-sleep=1000;storage-engine.defrag-startup-minimum=10

Try:

- Admin> asinfo -v "get-config" like evict
- Admin> asinfo -v "namespace/test" like evict

Exercises

Exercise 1: Namespace

- Add a namespace with data persisted on file. Here are the requirements:
  - Total number of records: 300,000.
  - Average record size: 2048B.
  - Let's keep it simple, only 1 bin per record.
- Insert all those records in your new namespace using the Java Benchmark tool.
- Verify RAM and Disk usage
  - Using AMC
  - Using asadm

Reminder - Capacity Planning – Quick Estimate

Area	How stored	Formula	Note
Primary Index	RAM	n * r * 64	The amount of RAM needed for the primary index is fixed at 64 bytes.
Data storage	RAM	n * r * (2 + (17 * b) + v)	Every objects needs 2 bytes for overhead, 17 bytes per bin, and the actual data
Data storage	Flash/SSD	n * r * p Where p is ((64 + (9 + s) + (28 + 2 or 5)* b) + v) -> round up to nearest 128 bytes	Every object needs to store the index (64 bytes), set overhead (9 +s bytes) , general overhead (28 bytes),) type info (2(int) or 5(str) bytes per bin), and the actual data. Because Aerospike stores data in 128 byte blocks, you must round up to the nearest 128 byte amount.

- n = number of records
- r = replication factor
- v = average size of records
- b = number of bins
- s = avg set name length

- Exercise:
  - 300,000 records
  - 2048B per record
  - Data is persisted on disk only
  - 1 bin per record

Sizing

- Sizing
  - RAM
    - (300,000 * 64) / (1024*1024) = ~ 18.3MiB
    - If replication factor 2, then *2: 36.6MiB
    - Memory High Water Mark 60%: 61MiB (36.6/0.6)
    - Let's be generous, and go with **100MiB** ☺
    - Also, in this special case, as we are running on a single node, we will default to replication factor 1.
  - SSD
    - p = (64 + (9 + 7) + (28 + 5)*1 + 2048 = 2161 rounded up to next 128 bytes -> 2176
    - (300,000 * 2176) / (1024*1024*1024) = ~ 0.61GiB
    - * 2 (rep. factor) = 1.2GiB
    - * 2 (defrag) = 2.4GiB
    - Again, replication factor 1, so we should be using ~0.61GiB, but to avoid evictions, we should size for 1.2GiB. But let's be generous again and give it **1.5GiB** ☺

Configuration

- Configuration:

```
namespace ns1 {
        replication-factor 2
        memory-size 100M
        default-ttl 30m # 30minutes


        storage-engine device {
                file /opt/aerospike/data/ns1.dat
                filesize 1500M
                # write-block-size 1M
        }
}
```

Exercise 2: Change TTL

Let's spread those records, 1 hour apart starting 3 hours in the future, for the sake of the exercise.

- **asinfo**
  - Tool to dynamically change configuration (among other things).
  - 2 useful links to bookmark:
    - Info commands reference: http://www.aerospike.com/docs/reference/info/
    - Configuration reference: http://www.aerospike.com/docs/reference/configuration/
- We will change the default-ttl for the namespace between each 100,000 records insert.

```
$asinfo -v 'set-config:context=namespace;id=ns1;default-ttl=3h'

$./run_benchmarks -n ns1 -s testset -k 100000 -S 1 -o S:2048 -w I -z 8

$asinfo -v 'set-config:context=namespace;id=ns1;default-ttl=4h'

$./run_benchmarks -n ns1 -s testset -k 100000 -S 100001 -o S:2048 -w I -z 8

$asinfo -v 'set-config:context=namespace;id=ns1;default-ttl=5h'

$./run_benchmarks -n ns1 -s testset -k 100000 -S 200001 -o S:2048 -w I -z 8
```

Usage

- Verify RAM and Disk usage using asadm:

```
$ asadm
Aerospike Interactive Shell, version 0.1.4
Found 1 nodes
Online:  172.31.59.3:3000

Admin> info
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Network Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node                Node                     Ip      Build   Cluster            Cluster     Cluster          Principal   Client      Uptime
   .                  Id                      .          .      Size                Key   Integrity                  .    Conns           .
i      *BB94FB6A4647106   172.31.24.237:3000   E-3.9.0          1   78432267FA2F66BF   True        BB957C01E72660E        3   00:03:39
Number of rows: 1

Namespace Node    Avail%   Evictions       Master    Replica      Repl     Stop      Pending        Disk     Disk      HWM          Mem      Mem     HWM      Stop
        .    .         .           .      Objects    Objects    Factor   Writes     Migrates        Used    Used%    Disk%         Used    Used%    Mem%    Writes%
        .    .         .           .            .          .         .        .    (tx%,rx%)           .        .        .            .        .       .          .
bar          i   N/E         0.000        0.000 K    0.000     1          false     (0,0)               N/E   N/E      50         0.000 B    0         60       90
ns1          i   58          0.000      300.000 K    0.000     1          false     (0,0)        622.559 MB   42       50        18.311 MB   19        60       90
test         i   N/E         0.000        0.000 K    0.000     1          false     (0,0)               N/E   N/E      50         0.000 B    0         60       90
Number of rows: 6
```

- We are using 42% of the disk, have 58% avail_pct and are using 19% of RAM.
- Notice that Disk Used and Avail% add up to exactly 100% (in this very particular case!).

Histogram: TTL

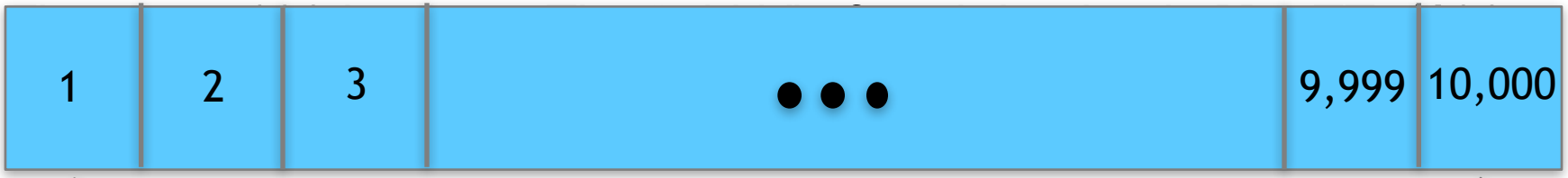
ttl histogram:

```
asinfo -v 'hist-dump:ns=ns1;hist=ttl'
ns1:ttl=100,180,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,100000,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,100000,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,100000;
```

180s

- As expected, records spread in 3 buckets. 180s is the 'width' of each bucket.
- (180*100) / 3600 = 5 hours.

Records will be grouped by buckets. Eviction will expire first all records in the first bucket, then will move on to the second one, etc…



Data that will expire soonest will be the first to be evicted.

WARNING: Watermark breached!!!

Data that will expire latest.

Histogram: Object size

- **Object size histogram:**

```
asinfo -v 'hist-dump:ns=ns1;hist=objsz'
ns1:objsz=100,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,300000,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0;
```

- Bucket #17 has all the records.
- 17 * 128 = 2176 (2048 + 113B overhead rounded up to next 128B)

- Available for storage-engine device
- Optionally available at set level
- `asinfo -v "hist-dump:ns=<NAMESPACE>;set=<SET>;hist=objsz"`

Exercise 3: Breach high water mark

- Add another 100,000 records (same size 2048B).
- Let's insert them in 200 batches of 500, 1 minute apart

```
for i in {1..200}; do
>   asinfo -v "set-config:context=namespace;id=ns1;default-ttl=${i}m"
>   ./run_benchmarks -n ns1 -s testset -k 500 -S $((300000+500*i)) -o S:2048 -w I -z 8
> done
```

(this script is in ~aerotraining/packages/aerospike/insert_records.sh for your convenience)

- Observe what happened when looking at asadm or AMC.
  - Look at the Used Disk column in asadm.
  - AMC throws an alert notification.

- Let's check the ttl histogram again:

```
asinfo -v 'hist-dump:ns=ns1;hist=ttl'
ns1:ttl=100,176,2500,1000,1500,1500,1500,1500,1500,1000,1500,1500,1500,1500,100
0,1500,1500,1500,1500,1500,1000,1500,1500,1500,1500,1500,1000,1500,1500,1500,15
00,1500,1000,1500,1500,1500,1500,1500,1000,1500,1500,1500,1500,1000,1500,1500,1
500,1500,1500,1000,1500,1500,1500,1500,1500,1000,1500,1500,1500,1500,101000,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28092,71908,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,100000;
```

Evictions in the Log

- Let's take a look at the logs to see how many records are evicted during each nsup cycle:
  - `grep thr_nsup /var/log/aerospike/aerospike.log`

```
Jul 20 2016 15:41:59 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 384000, 0 0-vt,
0(2000) expired, 1500(1500) evicted, 0(0) set deletes. Evict ttl: 110. Waits: 0,0,0. Total
time: 119 ms
```

- 1500 records evicted this cycle, 1500 total (first cycle)
- All records in all buckets less than 110s were evicted

```
Jul 20 2016 15:45:59 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 395000, 0 0-vt,
0(2000) expired, 1500(4500) evicted, 0(0) set deletes. Evict ttl: 242. Waits: 0,0,0. Total
time: 122 ms
```

- 1500 records evicted in this cycle, 4500 records total
- Eviction bucket is now at 242s

Eviction time

- This is slow… **how long** is it going to take?
  - 1500MiB / 2 = 750MiB which represent (750 *1024 *1024) / 2176 = 361,411 records.
  - Would need to evict ~40,000 records.

- Each cycle has a **limit** for how many records can be evicted:
  - evict-tenths-pct: default value is 5 (5/10 = 0.5%)
  - 400,000 * 0.5% = 2,000
  - We see 1,500 records evicted:
    - Each bucket contains 500 records
    - Aerospike finds 4 buckets which contains the 2,000 records, but doesn't evict the last bucket it finds.

- Would take (40000 / 2000) * 120 s = 40minutes.
- Let's speed this up!

- Reduce the nsup-period to 30 seconds:

```
asinfo -v 'set-config:context=service;nsup-period=30'
```

- Increase the evict-tenths-pct to 20:

```
asinfo -v 'set-config:context=namespace;id=ns1;evict-tenths-pct=20'
```

Review

```
Jul 20 2016 15:47:48 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 392000, 0 0-
vt, 0(2000) expired, 7500(13500) evicted, 0(0) set deletes. Evict ttl: 1254. Waits:
0,0,0. Total time: 119 ms

Jul 20 2016 15:48:19 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 384500, 0 0-
vt, 0(2000) expired, 7500(21000) evicted, 0(0) set deletes. Evict ttl: 2156. Waits:
0,0,0. Total time: 121 ms

Jul 20 2016 15:48:48 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 377000, 0 0-
vt, 0(2000) expired, 7500(28500) evicted, 0(0) set deletes. Evict ttl: 3062. Waits:
0,0,0. Total time: 116 ms
...
...

Jul 20 2016 15:49:49 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 362500, 0 0-
vt, 0(2000) expired, 7000(42500) evicted, 0(0) set deletes. Evict ttl: 4744. Waits:
0,0,0. Total time: 110 ms

Jul 20 2016 15:50:18 GMT: INFO (nsup): (thr_nsup.c:1096) {ns1} Records: 355500, 0 0-
vt, 0(2000) expired, 0(42500) evicted, 0(0) set deletes. Evict ttl: 0. Waits: 0,0,0.
Total time: 54 ms
```

Review

Let's check the ttl histogram again:

```
asinfo -v 'hist-dump:ns=ns1;hist=ttl'
ns1:ttl=100,170,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,1000,1000,1500,1500,1500,1000,1500,1500,1500,1000,15
00,1500,1000,1500,1500,1500,1000,1500,1500,1500,1000,1500,150
0,1500,1000,1500,1500,1000,1500,1500,101500,1000,1500,1500,15
00,1000,1500,1500,1000,1500,1500,0,0,0,0,0,0,0,0,0,0,100000,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,100000;
```

Exercise 4: Defrag

- Look at disk used and available percent (avail_pct).
  - Notice they don't add up to 100% anymore.
  - 44 + 50 = 94%
  - Why?

```
$ asadm
Aerospike Interactive Shell, version 0.1.4
Found 1 nodes
Online:  172.31.59.3:3000


Admin> info
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Network Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 Node               Node                   Ip      Build   Cluster            Cluster     Cluster         Principal   Client      Uptime
    .                 Id                    .          .      Size                Key   Integrity                 .    Conns           .
i      *BB957C01E72660E   172.31.24.237:3000   E-3.9.0         1   80752D13B5196C2E   True         BB957C01E72660E       13    00:23:42
Number of rows: 1
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Namespace Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node   Namespace   Avail%   Evictions     Objects     Repl     Stop         Disk     Disk     HWM          Mem     Mem    HWM      Stop
   .           .        .           .           .   Factor   Writes         Used    Used%   Disk%         Used   Used%   Mem%   Writes%
i      bar            N/E           0     0.000          1   false           N/E      N/E      50   0.000 B          0     60        90
i      ns1             44      42.500   355.500 K        1   false    737.732 MB       50      50   21.557 MB       22     60        90
i      test           N/E           0     0.000          1   false           N/E      N/E      50   0.000 B          0     60        90
Number of rows: 3
```

Exercise 4 cont..

- Evictions have fragmented the storage.
- Let's look at some info in the logs again about defrag:

```
grep defrag /var/log/aerospike/aerospike.log
```

```
Jul 20 2016 16:01:14 GMT: INFO (drv_ssd): (drv_ssd.c:2072) device
/opt/aerospike/data/ns1.dat: used 773568000, contig-free 667M (667 wblocks), swb-free 16,
w-q 0 w-tot 831 (0.0/s), defrag-q 0 defrag-tot 0 (0.0/s) defrag-w-tot 0 (0.0/s)
```

- We have written 831 wblocks, and defragged 0.
- Let's see what happens when we start updating the records in our namespace.
  - Start a Read/Update workload at 50/50 using the Java Benchmark Tool.

```
./run_benchmarks -n ns1 -s testset -k 400000 -S 1 -o S:2048 -w RU,50 -z 8
```

- Let's look at the logs for speed of writes vs. speed of defrag.

```
Jul 20 2016 16:04:34 GMT: INFO (drv_ssd): (drv_ssd.c:2072) device
/opt/aerospike/data/ns1.dat: used 848707456, contig-free 378M (378 wblocks), swb-free 15,
w-q 0 w-tot 5260 (46.8/s), defrag-q 13 defrag-tot 4062 (48.0/s) defrag-w-tot 1932 (23.3/s)
```

- At this point in the system this log line is copied from, defrag is not keeping up (46.8 > 23.3).
- Let's stop the benchmark.

Exercise 4 Cont…

- We can **speed up defrag** by tuning defrag-sleep.
  - **defrag-sleep**: how much to sleep in between each block being consumed out of the defrag queue.
  - Default: 1000µs (micro seconds). May impact performance if decreased too much.

- Let's misconfigure the server to pretty much stop defrag.
  - defrag-lwm-pct: default 50%.
  - Let's make it 5%:

```
asinfo -v 'set-config:context=namespace;id=ns1;defrag-lwm-pct=5'
```

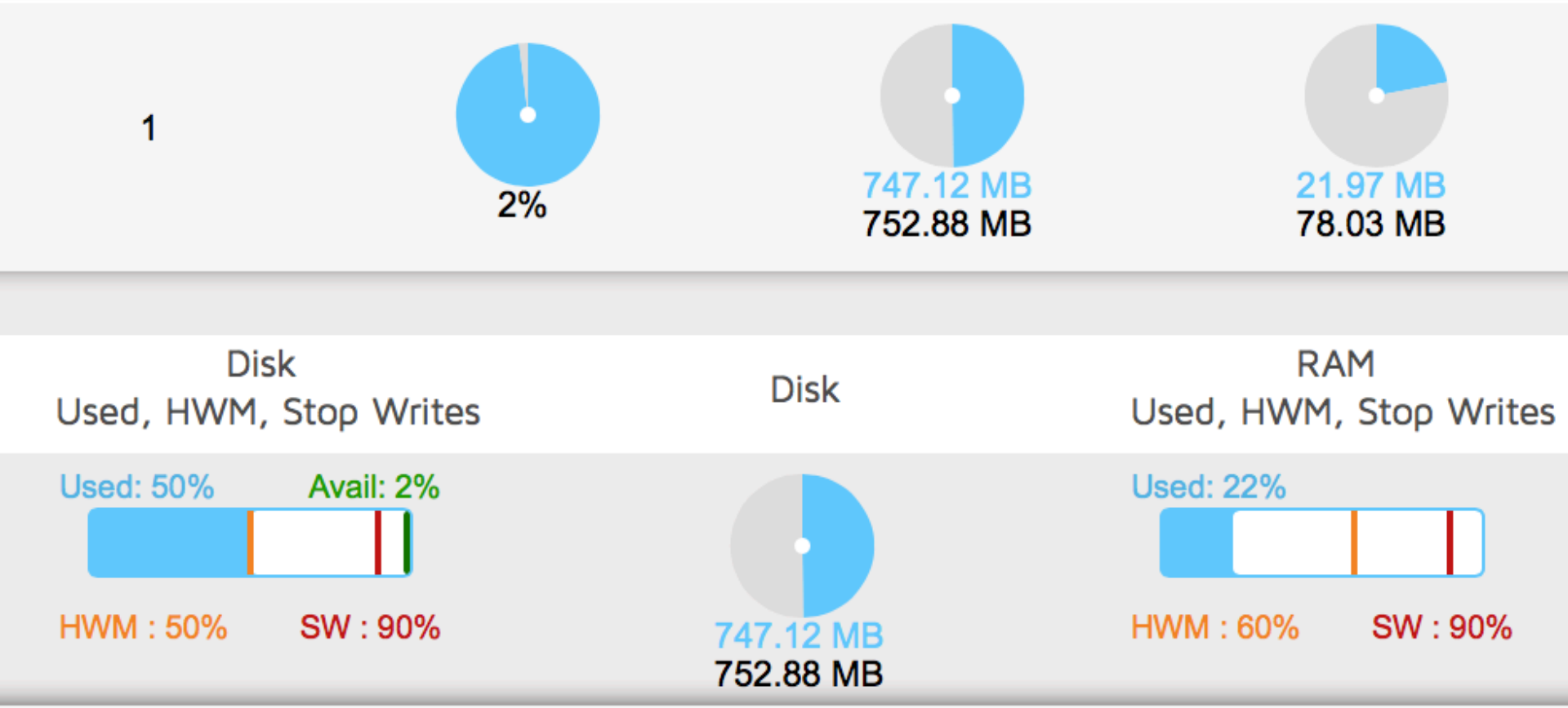
- Let's continue our benchmark workload:

```
./run_benchmarks -n ns1 -s testset -k 400000 -S 1 -o S:2048 -w RU,50 -z 8
```

- Observe the avail % in asadm or AMC.
- What happens after a few moments?

Exercise 5: Stop writes

- We hit stop writes.
- The key value store is now operating in read only mode.
- Notice the errors from the benchmark tool for the writes.



Exercise 5 Cont..

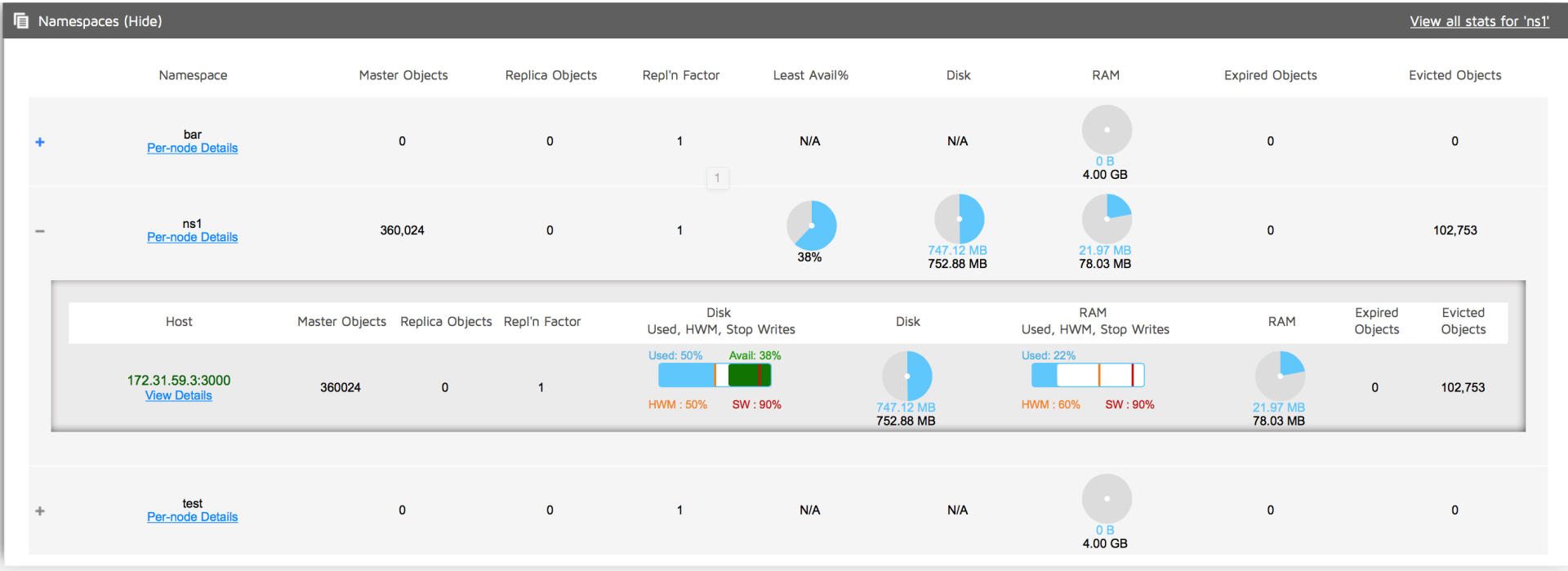
- Same state now from asadm:

```
asadm
Aerospike Interactive Shell, version 0.0.9
Found 1 nodes
Online:  172.31.59.3:3000


Admin> info
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Network Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Node               Node                   Ip      Build   Cluster            Cluster     Cluster          Principal    Client      Uptime
   .                 Id                    .          .      Size                Key   Integrity                  .     Conns           .
I      *BB957C01E72660E   172.31.24.237:3000   E-3.9.0         1   80752D13B5196C2E   True         BB957C01E72660E       22    00:36:06
Number of rows: 1

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~Namespace
Information~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Namespace  Node   Avail%   Evictions       Master   Replica     Repl     Stop     Pending         Disk     Disk     HWM         Mem     Mem    HWM
Stop
        .     .        .           .      Objects   Objects   Factor   Writes    Migrates         Used    Used%    Disk%        Used   Used%   Mem%
Writes%
        .     .        .           .           .         .        .        .    (tx%,rx%)           .        .        .           .       .      .
.
bar          i   N/E        0.000        0.000     0.000     1        false     (0,0)              N/E   N/E      50        0.000 B     0        60      90
ns1          i   2        117.889 K   371.155 K   0.000     1        true      (0,0)       770.219 MB   52       50       22.654 MB   23       60      90
test         i   N/E        0.000        0.000     0.000     1        false     (0,0)              N/E   N/E      50        0.000 B     0        60      90
Number of rows: 3
```

Getting out of stop writes

- Let's recover from stop writes
  - Set defrag-lwm-pct back to 50%:

```
asinfo -v 'set-config:context=namespace;id=ns1;defrag-lwm-pct=50'
```



Exercise 6: Insights

- More insights from the logs:
  - **Cache-read pct**: percentage of reads served from memory and not hitting the disk.
  - **Post write queue** keeps some records (blocks to be precise) in memory.

```
grep cache /var/log/aerospike/aerospike.log
```

```
Jul 20 2016 16:13:52 GMT: INFO (info): (ticker.c:433) {ns1} device-usage: used-bytes
779282176 avail-pct 36 cache-read-pct 29.61
```

- Post write queue:
  - After flushing swb blocks to the device, this config parameters tunes how many **blocks** to keep **in memory** for fast read access.
  - Will help any use case where records are **read soon after** they are **inserted/updated**.
  - Very beneficial if XDR is running.
  - **Default: 256**. This is per device and measured in number of blocks (write-block-size will impact how much memory will be used by the post write queue).
  - Blocks still referenced in post write queue are not eligible to be defragged.

asloglatency

asloglatency

`asloglatency` is a command line tool used find the latency of the server in log files for specific types of transactions.

Typical syntax

```
> asloglatency -h <histogram> -l <log_file> -f <time_from> -d <duration>
```

Option	Default	Description
-l	/var/log/aerospike/aerospike.log	Log file to read from. Can be used to read from logs that have been rotated out.
-h	[none]	(required) One of `read, batch-index, query, query-rec-count, read, udf, write`
-t	10	Analysis slice interval in seconds or time format. Time format is “HH:MM:SS”
-f	tail	Time_from may be in either form “Aug 6 2014 22:10:13”, “-3600”, “-1:00:00”. Default is to tail the file.
-d		Maximum duration from which to analyze. Duration is in either form “3600” or “HH:MM:SS”
-n	3	Number of buckets to display.
-e	3	Show the 0-th and then every e-th bucket. Lower numbers show finer granularity. Examples: n=3, e=3 will show (ms): 1,8,64 n=7, e=1 will show (ms): 1,2,4,8,16,32,64
-N	[none]	Namespace for which to display statistics.

Asloglatency options

- Namespace specifier (optional, 3.9+)
  - As a separate parameter: -N namespace
  - As part of the histogram speficier: -h {namespace}-read
- Auto-enabled benchmarks:
  - Batch-index, {ns}-query, {ns}-query-rec-count, {ns}-read, {ns}-udf, {ns}-write
    - Will only show results if appropriate transactions are occurring
- Configuration-enabled benchmarks:
  - Enable with asinfo, eg

```
asinfo -v 'set-config:context=service;svc-queue=true'
```

  - Read transaction analysis, write transaction analysis, UDFs, etc
- See http://www.aerospike.com/docs/operations/monitor/latency

asloglatency - example

Suppose there was an issue in read latency 12 hours ago that lasted for an hour. You wish to review the read latencies from 12 hours ago to 10 hours ago. You can issue the command:

```
> asloglatency -h read -f -12:00:00 -d 2:00:00
Read

Aug 6, 2014 01:58:58
% > (ms)
slice-to (sec)       1      8     64   ops/sec
-------------- ------ ------ ------ --------
01:59:08    10   1.13   0.04   0.00   4661.8
01:59:18    10   1.13   0.04   0.00   4661.8
01:59:28    10   1.13   0.04   0.00   4661.8
...
03:58:58    10   1.13   0.04   0.00   4661.8
03:59:08    10   1.13   0.04   0.00   4661.8
-------------- ------ ------ ------ --------
avg              0.97   0.04   0.00   4188.0
max              1.34   0.05   0.00   4661.8
```

asloglatency can also be run without –f and –d argument to see current latency. Try it!

asloglatency – micro / storage benchmarks

Details on our documentation site:

http://www.aerospike.com/docs/tools/asloglatency/

asadm – show latency

asadm can also be used to show current latencies for the main histograms across all nodes in the cluster.

Command: `show latency`

Displays latency stats for how long requests take to be filled as measured on the server. This may differ significantly from the client latency measures. There are additional parameters to take a look back at a specific time or gather other metrics. Useful for determining throughput.

```
Admin> show latency
~~~~~~~~~~~~~~~~~~~~~~~~proxy Latency~~~~~~~~~~~~~~~~~~~~~~~~
Node                      Time    Ops/Sec    >1Ms    >8Ms    >64Ms
   .                      Span          .       .       .        .
u10    22:59:33-GMT->22:59:43        0.0     0.0     0.0      0.0
u12    22:59:30-GMT->22:59:40        0.0     0.0     0.0      0.0
u13    22:59:35-GMT->22:59:45        0.0     0.0     0.0      0.0
Number of rows: 3

~~~~~~~~~~~~~~~~~~~~~~~~query Latency~~~~~~~~~~~~~~~~~~~~~~~~~
Node                      Time    Ops/Sec     >1Ms     >8Ms    >64Ms
   .                      Span          .        .        .        .
u10    22:59:33-GMT->22:59:43     1661.7    99.99     52.3    38.96
u12    22:59:30-GMT->22:59:40     1332.5    100.0    13.75     1.06
u13    22:59:35-GMT->22:59:45     1398.5    100.0    22.53      0.0
Number of rows: 3

~~~~~~~~~~~~~~~~~~~~~~~~reads Latency~~~~~~~~~~~~~~~~~~~~~~~~~
Node                      Time    Ops/Sec     >1Ms    >8Ms    >64Ms
   .                      Span          .        .       .        .
u10    22:59:33-GMT->22:59:43        0.0      0.0     0.0      0.0
u12    22:59:30-GMT->22:59:40      152.1    37.28     0.0      0.0
u13    22:59:35-GMT->22:59:45      157.8    47.59     0.0      0.0
Number of rows: 3

…

~~~~~~~~~~~~~~~~~~~~~~writes_master Latency~~~~~~~~~~~~~~~~~~~~~~
Node                      Time    Ops/Sec     >1Ms     >8Ms    >64Ms
   .                      Span          .        .        .        .
u10    22:59:33-GMT->22:59:43        4.0    100.0    100.0    100.0
u12    22:59:30-GMT->22:59:40      357.9    76.73      2.1     0.08
u13    22:59:35-GMT->22:59:45      334.8    75.84     1.67     0.06
Number of rows: 3

~~~~~~~~~~~~~~~~~~~~~~writes_reply Latency~~~~~~~~~~~~~~~~~~~~~~
Node                      Time    Ops/Sec     >1Ms     >8Ms    >64Ms
   .                      Span          .        .        .        .
u10    22:59:33-GMT->22:59:43        4.0    100.0    100.0    100.0
u12    22:59:30-GMT->22:59:40      357.8    76.69     2.07     0.08
u13    22:59:35-GMT->22:59:45      334.8    75.84     1.67     0.06
Number of rows: 3
```

Collectinfo

To **gather information for Aerospike support** use the collectinfo command.

This command still uses the precursor to asadm called asmonitor.

Note: sudo/root privileges required.

```
[root@v15 ~]# sudo asadm -e "collectinfo"

Enter help for commands

3 hosts in cluster: 192.168.120.143:3000,192.168.120.144:3000,192.168.120.145:3000
Data collection for collect_asdcheck in progress..
Data collection for collect_params in progress..
Data collection for collect_loginfo in progress..
Data collection for collect_readlogs in progress..
Data collection for collect_sys in progress..
Data collection for collect_shell in progress..
running shell command: tar -czvf /tmp/as_log_1408404265.16.log.tgz /tmp/as_log_1408404265.16.log
tar: Removing leading `/' from member names
/tmp/as_log_1408404265.16.log


FILE /tmp/as_log_1408404265.16.log and /tmp/as_log_1408404265.16.log.tgz saved. Please send it to support@aerospike.com
END OF ASCOLLECTINFO
```

Backup and Restore

asbackup & asrestore

- Aerospike (Ver 3.0+) provides asbackup to serially backup your data to file(s) and asrestore to restore data from backup files.
- http://www.aerospike.com/docs/tools/backup

- Installed as part of aerospike-tools package
- asbackup (http://www.aerospike.com/docs/tools/backup/asbackup.html)
  - Backup all data from a specific namespace or set
  - Data stored in files in specified directory, max 250Mb each file (default).
  - Estimate backup size, select what to backup, output options, connection options
- asrestore (http://www.aerospike.com/docs/tools/backup/asrestore.html)
  - Restore data from backup directory
  - Select what to restore, write policy, connection options.