Caringo Support

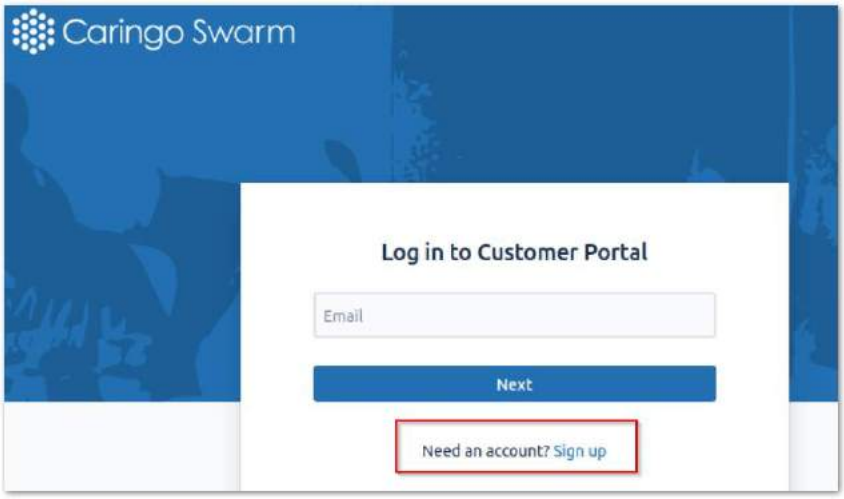
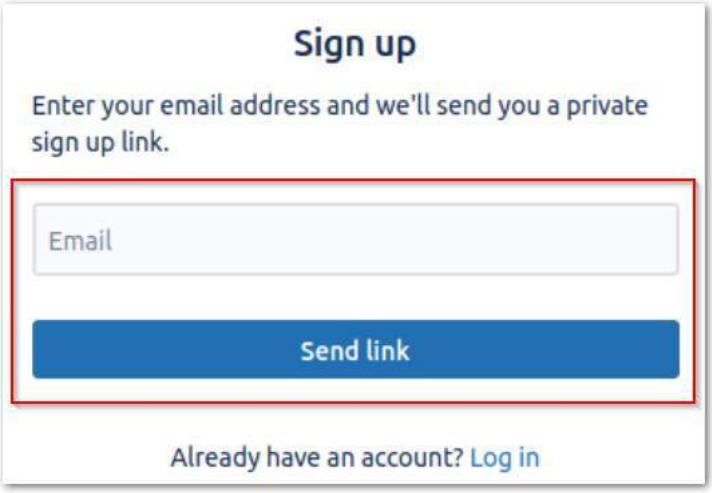
Following is guidance on how to work with Caringo Support, use the support tools provided, and get the fastest help with your product issues.

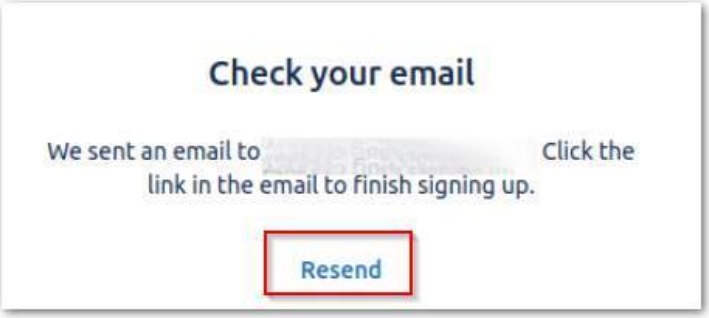
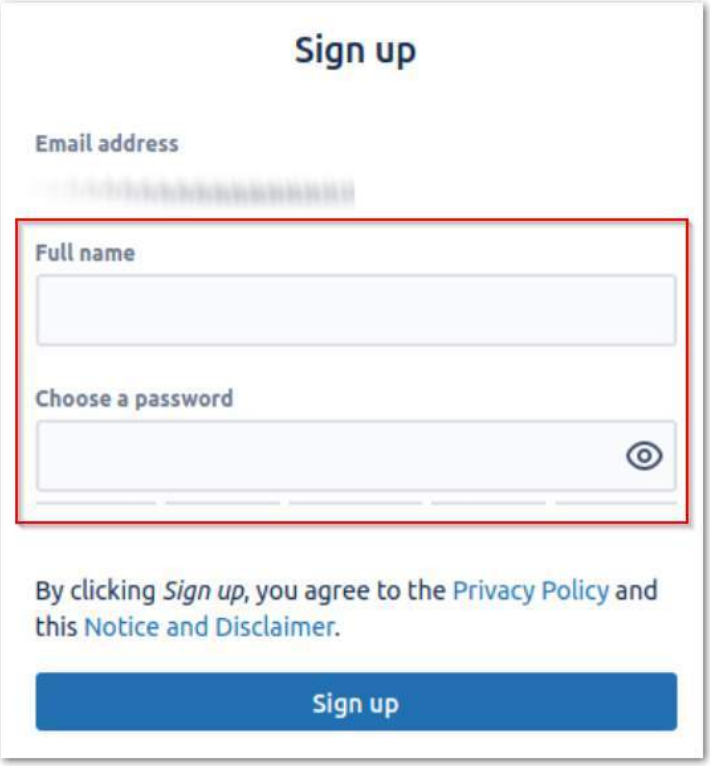
- How to open a support case
- Tips to solve your support case faster
- How to collect a support bundle
- How to upload any file to Support
- How Support uses 'Organization'
- Support Tools Training: swarmctl & swarmrestart
- How to use indexer-enumerator.sh

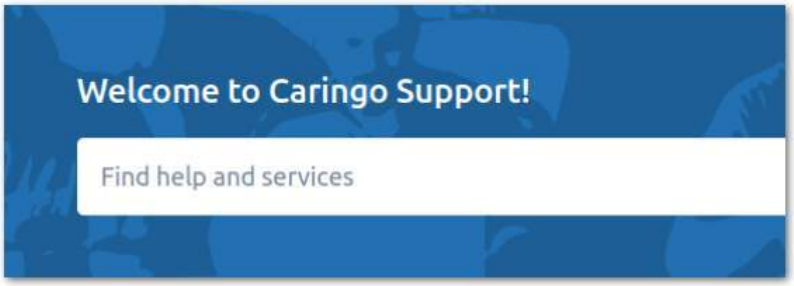
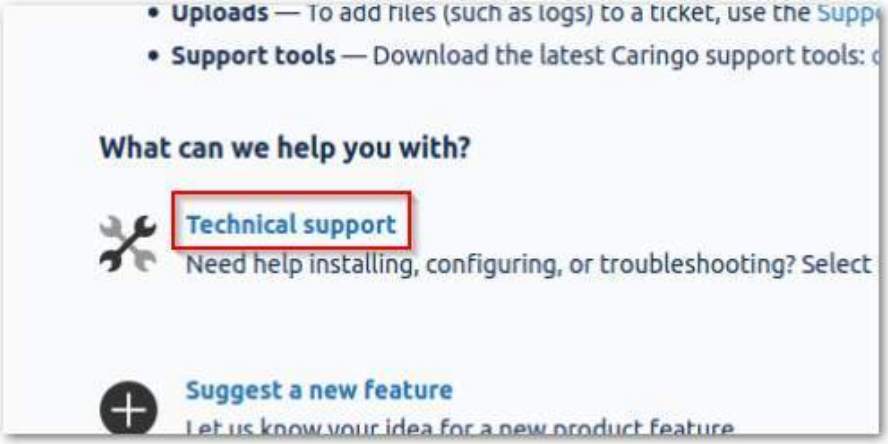
How to open a support case

Here are the steps and recommendations on how to raise a case with the Caringo support team.

Registering to use the portal

1	Browse to the support portal:	**support.caringo.com**
2	If you have never opened a case before, create an account by selecting the **Sign up** link:	
3	When prompted, enter your email address and click **Send link**:	

4	Look for an email from **jira@caringo.atlassian.net**. When you receive the email, click on the link provided.	
5	If you don't receive it, check your spam/junk folders. If it's not there, select the **Resend** option.	
6	At the **Sign up** prompt, enter your name and a password (which will have to pass a strength check):	

7	If you see **Welcome**, your registration is complete.	
8	Now that you are logged into the support portal, create a new case by selecting the **Technical support** link:	
9	To search across all documentation and knowledge base articles, type a phrase into the search field at top:	

Entering a new case

> **Important**
>
> *Before* contacting Support, whoever submits the ticket should arrange to have remote access to the environment (such as “root” access on the Platform/CSN or Gateway systems).
>
> If you cannot access the system, Support will be severely limited in their ability to assist you.

Make sure that your ticket covers the vital information:

- Describe **your environment**. At minimum, list the versions of the Swarm components that are associated with the account (such as “Swarm 11.2.0, CSN 8.3.2, …”).
- Include a **detailed explanation** of the problem.
- List any **troubleshooting steps** attempted (if any).
- Include the **name of the cluster** so we can identify it in our health report listing, if possible.

- Include the **name of the end customer**, if you are acting on someone's behalf.

Once you have created a case, note the case number and create and upload a log bundle to link to it:

- **How to collect a support bundle**

Tips to solve your support case faster

Here are tips to improve the speed and effectiveness of the support you receive from Caringo:

- **Start with specifics** — Frequently we receive support tickets with vague problem descriptions (such as "Swarm is not replicating" or "One node won't come up") and no information about the version of software involved, the configuration, the exact messages or symptoms received. Without that information, all we can do is to respond and ask you for more information. You can save a lot of time by being precise and detailed and by including the maximum possible amount of configuration information.
- **Configuration** — To diagnose almost any problem, we need to know the basics about your configuration:
    1. Basic hardware architecture (number and type of nodes, connectivity, etc.)
    2. Software versions (Swarm, CSN, etc.)
    3. Swarm configuration (your `cluster.cfg/node.cfg` file)
- **Logging** — For most problems, being able to see what happened *over time* is essential to understanding behavior. This means that we will likely need to see your logs.
    1. *Enable logging*: Please make sure that you have logging enabled.
    2. *Retain logs*: If you are not logging to a syslog server somewhere or if you have not retained those logs, it is quite likely that it will be impossible to definitively say what went wrong.
    3. *Target the timeframe*: Please provide us logs for the period during which the problem you are trying to resolve occurred.

Thank you! We are much more likely to be able to diagnose your problem quickly if you send us these details early.

How to collect a support bundle

The **Caringo Support Tools** bundle is essential for all users of Caringo products. It contains scripts for many purposes. Inside the bundle are support scripts as well as PDFs that describe in detail how to use them:

- `Tech-Support-Scripts-Bundle.pdf`
- `Tech-Support-Scripts-Bundle-swarmctl.pdf`

The primary tool in this bundle is `techsupport-bundle-grab.sh`, a script that snapshots your Swarm configuration and log data into a tarball, which you attach to the ticket to help troubleshoot your case.

Creating your support bundle

When you open a support ticket with Caringo, best practice is to proactively collect and upload a support tarball with your technical information. You create this tarball by running the `techsupport-bundle-grab.sh` command from the extracted tools bundle.

The first time you have an issue to raise with Support, follow this process to give them your support bundle:

1. Open a support ticket, and note the **ticket number** (such as `SUP-1234`), to use when uploading your file.
2. Download the bundle here: https://support.cloud.caringo.com/tools/caringo-support-tools.tgz
3. Extract the tools in the `/root` directory on the server (CSN, Elasticsearch, SwarmNFS, or Gateway).
4. Navigate to the tools directory, which is usually `/root/dist`.
5. Run the script by typing: `./techsupport-bundle-grab.sh`
6. Note the name and location of the tarball it created.

Uploading the bundle (internet access)

If your server has internet access, you can upload directly from your server:

1. To enable the uploadtosupport function in your shell, type the following:
   `source /root/dist/bashrcforcustomers`
2. To upload the bundle to Caringo Support, type this:
   `uploadtosupport [new-bundle-filename]`
3. When prompted for a customer name and ticket number, enter the number you noted above.
4. On success, the uploader automatically adds your bundle to the ticket and notifies Support; you do not need to email Support or make any changes to the ticket.

Uploading the bundle (no internet access)

If your server does not have access to the internet, you will need to move the file to a location that does.

1. Securely connect (such as with WinSCP) to the server (CSN or other product) from an accessible location.
2. Download your new support bundle.
3. Browse to the Uploader URL: https://support.cloud.caringo.com/uploader/uploader
4. Upload your bundle from your download location.

5. When prompted for a customer name and ticket number, enter the number you noted above.
6. On success, the uploader automatically adds your bundle to the ticket and notifies Support; you do not need to email Support or make any changes to the ticket.

How to upload any file to Support

Uploader (preferred)

If you are using a modern browser (Chrome, Firefox, not IE), you can browse to the uploader and transfer the file directly.

- https://support.cloud.caringo.com/uploader/uploader

Caringo Support will automatically be notified of the link to the file and the link will be attached to your ticket. This uploader will tag the file with metadata specific to support, and is by far our preferred method for transferring data to Support.

cURL

You may use curl to upload logs, configurations, and other large files to Caringo Support. In order to do so, you must have curl installed.

There is a function called *`uploadtosupport`* included in the file "`bashrcforcustomers`" in the Support bundle:

- https://support.cloud.caringo.com/tools/caringo-support-tools.tgz

If you have downloaded this bundle, add the following to your /root/.bashrc file:

```
source /root/dist/bashrcforcustomers
```

That way, you can upload a file to support using the following syntax (after sourcing .bashrc again):

```
source /root/.bashrc
uploadtosupport [filename]
```

This will put the file in a bucket called `cli-uploader`. We will get automatically notified within 10 minutes.

> **Tip**
> For your convenience, a reference to the support tools is located in a PDF within the bundle itself.

How Support uses 'Organization'

Filling in the **Organization** field in the support portal is necessary to associate you with an account that holds a valid support contract.

Notifications

All support users get notifications about cases that are related to their *own* organization:

- They will be notified each time a ticket has been *created*.
- They will be notified when the organization has been *added* to an existing ticket.
- They will be able to 'opt in' to see *updates* about each ticket. If they do not opt in, they won't get any updates on the case.

How are people removed?

If there are others within your organization that were previously copied on tickets who no longer need access to your support tickets, ask Support to remove them from the organization.

Support Tools Training: swarmctl & swarmrestart

What We'll Cover

- **snmp-castor-tool.sh**
- **swarmctl**
- **swarmrestart**

snmp-castor-tool.sh

- snmp-castor-tool.sh has been around for many years
- uses SNMP for most operations (although not all)
- swiss army knife for many support operations (38 different option flags!)
- can query the cluster AND perform operations against the cluster
- can change cluster settings in real-time
- can rolling reboot a cluster
- can collect snmp oid output
- relies on SNMP which is very slow, especially in larger clusters
- snmp could eventually be removed from Swarm in favor of the better tools

swarmctl Overview

- Like `snmp-castor-tool.sh`, swarmctl lives in the support tools bundle (support.cloud.caringo.com/tools/caringo-support-tools.tgz)
- Uses the management API for all operations
- Swiss army knife for many support operations
- Can query the cluster AND perform operations against the cluster
- Can change cluster settings in real-time
- Cannot rolling reboot a cluster (see the `swarmrestart` script)
- Does not collect snmp oid output
- Relies on the mgmt api, which is very fast compared to SNMP
- Is the go-forward support tool for the majority of support operations (Swarm 10+)
- If SCSP_HOST is set in the environment, -d [ip address] is not necessary
- `-x` is commonly used to get more output, in some cases MUCH more
- `-x` is used to output to a csv file automatically for more analysis
- By default, will only take action or get information from the node specified in SCSP_HOST or -d [ip address] options.
- `-a` takes action or gets information from ALL nodes in the cluster as seen in the cluster status page
- `-a` is not necessary to change a value that affects the PSS as the PSS is automatically read by all nodes
- `-n` is used to take action or query node IPs specified in a file you create called NODES.csv in the directory from which you run the swarmctl script (same note regarding -a applies here this works the same as in the snmp-castor-tool.sh script)

- `-j` **removes the text table lines to declutter the output**
- `-h` **is used for help (usage)**
- **most query output will show you the default value, the current value, whether it is changeable, and scope of the setting's effect (node/ cluster/ etc)**

swarmctl -A Announcements

- **swarmctl -A [show|clear] shows (default) and allows you to clear announcements.**
- **Add -a for all nodes.**

swarmctl -b Largest streams

- swarmctl -b shows the biggest streams and their rep counts.
    - Add -a for all nodes.
    - -x for much more output to file.
- If the largest stream on a particular disk is a segment, that is noted as "seg"
- In the graphic, /dev/sda is retired:

```
root@c-csn1:~/tmp>~/Support/swarmctl -b
+----------------+----------+--------------+-------------+---------+----------+-------------------+----------------------------------+-------------------+
| nodeIPAddress  |   name   | availPercent | streamCount | volErrs | maxSpace | largestStreamSize |        largestStreamUuid         | largeUuidRepCount |
+================+==========+==============+=============+=========+==========+===================+==================================+===================+
| 192.168.201.85 | /dev/sda | 0%           | 0           | 0       | 0.00MB   | 0                 |                                  | -                 |
+----------------+----------+--------------+-------------+---------+----------+-------------------+----------------------------------+-------------------+
| 192.168.201.85 | /dev/sdb | 63%          | 21,560      | 0       | 10.12GB  | 102               | 7443667259088a0abc171f06b06f41b3 | seg               |
+----------------+----------+--------------+-------------+---------+----------+-------------------+----------------------------------+-------------------+
| 192.168.201.85 | /dev/sdc | 61%          | 8,668       | 0       | 10.12GB  | 102               | 20286435ed16148ec949468876b214fe | seg               |
+----------------+----------+--------------+-------------+---------+----------+-------------------+----------------------------------+-------------------+
| 192.168.201.85 | /dev/sdd | 50%          | 6,196       | 0       | 10.12GB  | 1048              | b191f88b765f52ee04293d54402c1101 | 3                 |
+----------------+----------+--------------+-------------+---------+----------+-------------------+----------------------------------+-------------------+
```

swarmctl -C See and modify configuration

- **swarmctl -C [option] this shows a particular mgmt api configuration endpoint**
- **think OID in snmp parlance**
- **By default, this shows a single node's current value…**
    - **-a to see all nodes' values.**
- **Add -V [option] to change the value if Readonly is False.**
- **By far one of the most commonly used flags.**

```
root@c-csn1:~/tmp>swarmctl -C disk.obsoleteTimeout
+----------------+----------------------+---------+------------+---------+----------+
|      Node      |       Setting        |  Value  | Changed By |  Scope  | Readonly |
+================+======================+=========+============+=========+==========+
| 192.168.201.85 | disk.obsoleteTimeout | 1209600 |            | cluster | False    |
+----------------+----------------------+---------+------------+---------+----------+
root@c-csn1:~/tmp>swarmctl -C disk.obsoleteTimeout -V 604800
+----------------+----------------------+--------+----------------+----------------+-----------------+---------+----------+
|      Node      |       Setting        | Value  |   Changed By   | Previous Value | Prev Changed By |  Scope  | Readonly |
+================+======================+========+================+================+=================+=========+==========+
| 192.168.201.85 | disk.obsoleteTimeout | 604800 | Management API | 1209600        |                 | cluster | False    |
+----------------+----------------------+--------+----------------+----------------+-----------------+---------+----------+
```

swarmctl -d Specify the node to query

- swarmctl -d [ip address or DNS name]
- this option is used in conjunction with other options
- this option specifies the initial IP address to use when querying the cluster.
- swarmctl can use this IP address to collect the complete list of IPs in the cluster
- the complete list of IPs can be queried using -a once -d [ip] has been specified
- this option is not necessary if using the -n option
- you can forgo this option if you set SCSP_HOST in your environmental variables (which is why the examples in this deck are all missing the -d [ip] options)

```
root@c-csn1:~/tmp>echo $SCSP_HOST
192.168.201.85
```

swarmctl -D Query and control drive lights

- **swarmctl -D [{on,off,1,2,5,10,25,50}]**
- **this option is used to turn on and off the drive lights on a particular node/ disk**
- **the number variables indicate number of minutes to turn the light on, or set "on"**
- **use with -V [drive] to specify a particular drive**

```
root@c-csn1:~/tmp>swarmctl -D 1 -V /dev/sda
API reports success setting drive light /dev/sda on 192.168.201.85 to on with timeout: 1
```

swarmctl -E Errors

- **swarmctl -E [show|clear] shows (default) and allows you to clear errors.**
- **Add -a for all nodes.**
- **this is similar to -A, except for errors instead of announcements**

swarmctl -e HP cycle information

- **swarmctl -e shows you the current, ongoing HP cycle information.**
- **most commonly used with -x to export much more information for analysis**

```
root@c-csn1:~/tmp>swarmctl -e
+----------------+----------------------+-------------------------+-----------+-----------------------+-----------------------+-----------------------+-----------------------+-----------------------+
| nodeIPAddress  | HP Exam queue count  |  HP Replication queue   |  HP State |   HP ongoing cycle:   |   HP ongoing cycle:   | HP last cycle: Streams |  HP ongoing cycle,    |  HP last cycle, whole |
|                |                      |          count          |           |     Cycle number      |  Streams examined     |       examined        |  whole reps: Trims    | reps: Trims requested |
|                |                      |                         |           |                       |                       |                       |       requested       |                       |
+================+======================+=========================+===========+=======================+=======================+=======================+=======================+=======================+
| 192.168.201.85 | 0                    | 0                       | running (4) | 2                     | 1,048                 | 24,425                | 82                    | 3491                  |
+----------------+----------------------+-------------------------+-----------+-----------------------+-----------------------+-----------------------+-----------------------+-----------------------+
```

swarmctl -F format stale disks

- **swarmctl -F [stale volume]**
- **only applies to disks that have been marked as stale (offline for more than 2 weeks by default)**
- **leave off the volume option in order to format all stale disks on the node (likely the more common option)**
- **this prevents you from having to go to the terminal console, stop the storage processes, format the drives, and then reboot the chassis**

swarmctl -i Log level

- **swarmctl -i [{0,5,10,15,20,30,40,50}]**
- **display the log level (without variable) or use a variable to change the log level for all nodes in the cluster**

```
root@c-csn1:~/tmp>swarmctl -i
+----------------+-----------+-------+------------------+---------+----------+
|      Node      |  Setting  | Value |    Changed By    |  Scope  | Readonly |
+================+===========+=======+==================+=========+==========+
| 192.168.201.85 | log.level | 30    | cluster settings | cluster | False    |
+----------------+-----------+-------+------------------+---------+----------+
root@c-csn1:~/tmp>swarmctl -i 20
+----------------+-----------+-------+----------------+----------------+------------------+---------+----------+
|      Node      |  Setting  | Value |   Changed By   | Previous Value | Prev Changed By  |  Scope  | Readonly |
+================+===========+=======+================+================+==================+=========+==========+
| 192.168.201.85 | log.level | 20    | Management API | 30             | cluster settings | cluster | False    |
+----------------+-----------+-------+----------------+----------------+------------------+---------+----------+
```

swarmctl -I Remount stale volume

- **swarmctl -I [stale-vol-to-mount [stale-vol-to-mount ...]]**
- **leave blank to apply this to every stale disk on the chassis**
- **this option allows you to remount the stale volumes as opposed to -F which allows you to reformat stale volumes**
- **if you are sure the data on the disks is valid and necessary, use this option to resurrect those streams**
- **this will likely cause over-replication as the streams likely have already been recovered in the 2+ weeks since the drives were not stale.**

swarmctl -k_ Kernel modules

- **swarmctl -k will show the loaded kernel modules**
- **this is a lot of output- use with -x to output to a file, otherwise it is unusable**
- **partial output below**

```
root@c-csn1:~/tmp>swarmctl -k
Found 511 module descriptions.
+-----------------+-----------------+-------------------------------------------+-------------------------------------------------------------+-----------------+
| nodeIPAddress   |       name      |                  description              |                          filename                           |      version    |
+=================+=================+===========================================+=============================================================+=================+
| 192.168.201.85  | acpi_cpufreq    | ACPI Processor P-States Driver            | /lib/modules/4.19.56/kernel/drivers/cpufreq/acpi-cpufreq.ko |                 |
+-----------------+-----------------+-------------------------------------------+-------------------------------------------------------------+-----------------+
| 192.168.201.85  | ahci_platform   | AHCI SATA platform driver                 | /lib/modules/4.19.56/kernel/drivers/ata/ahci_platform.ko    |                 |
+-----------------+-----------------+-------------------------------------------+-------------------------------------------------------------+-----------------+
| 192.168.201.85  | tpm_tis         | TPM Driver                                | /lib/modules/4.19.56/kernel/drivers/char/tpm/tpm_tis.ko     | 2               |
+-----------------+-----------------+-------------------------------------------+-------------------------------------------------------------+-----------------+
| 192.168.201.85  | acpi_power_meter | ACPI 4.0 power meter driver               | /lib/modules/4.19.56/kernel/drivers/hwmon/acpi_power_meter.ko |                 |
+-----------------+-----------------+-------------------------------------------+-------------------------------------------------------------+-----------------+
| 192.168.201.85  | amd_xgbe        | AMD 10 Gigabit Ethernet Driver            | /lib/modules/4.19.56/kernel/drivers/net/ethernet/amd/xgbe/amd-  | 1.0.3           |
|                 |                 |                                           | xgbe.ko                                                     |                 |
```

swarmctl -K Display all cluster settings

- **swarmctl -K shows all currently active configuration parameters in the cluster!**
- **this shows you the scope of all of the options, where they were set, whether you can change them dynamically**
- **use with -x to export to file for ease of use**

```
root@c-csn1:~/tmp>swarmctl -K
Found 93 module descriptions.
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| nodeIPAddress   |                name            |                                                 value                                           | scope   | modified-by   | readonly  |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cache.expirationTime          | 600                                                                                             | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cache.maxCacheableSize        | 1048576                                                                                         | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cache.percentage              | 10                                                                                              | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cache.realmStaleTimeout       | 600                                                                                             | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cip.group                     | 224.0.10.100                                                                                    | cluster | config file   | readonly  |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cip.queryTimeout              | 0.030                                                                                           | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
| 192.168.201.85  | cip.readBufferSize            | 1048576                                                                                         | node    |               | writeable |
+-----------------+-------------------------------+-------------------------------------------------------------------------------------------------+---------+---------------+-----------+
```

swarmctl -L Node log level

- **swarmctl [-L {0,5,10,15,20,30,40,50}] Allows you to view and change a particular node's log level.**
- **This is useful when you want to change only a single node's log level instead of EVERY node's log level**
- **This value does not persist after a reboot**

```
root@c-csn1:~/tmp>swarmctl -L 30 -d 192.168.201.85
+----------------+------------------+-------+----------------+----------------+------------------+-------+----------+
|      Node      |     Setting      | Value |   Changed By   | Previous Value | Prev Changed By | Scope | Readonly |
+================+==================+=======+================+================+==================+=======+==========+
| 192.168.201.85 | log.nodeLogLevel | 30    | Management API | 0              | Management API  | node  | False    |
+----------------+------------------+-------+----------------+----------------+------------------+-------+----------+
```

swarmctl -m detailed running statistics

- **swarmctl [-m {commstats,hpstats,networktest,meminfo,features}]**
- **this option includes all kinds of details statistics from a running cluster**
- **typically used with -x to output automatically to a file where you will see much more output**

swarmctl -m commstats

- **swarmctl -m commstats**
- **shows bidding and histogram information**
- **typically used with -a AND -x to output all nodes' information automatically to a file where you will see much more output**

```
root@c-csn1:~/tmp>swarmctl -g -a
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
| nodeIPAddress  | SCSP: Last read bid | SCSP: Last rep bid | SCSP: Last write bid | Response Histogram: Maximum | Response Histogram: Mean | Response Histogram: Minimum | Response Histogram: Tail |
|                |                   |                   |                   |            (ms)             |            (ms)             |            (ms)             |            (ms)             |
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
| 192.168.201.84 | 19                | 25                | 34                | 95                          | 1                           | 0                           | 70                          |
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
| 192.168.201.85 | 6                 | 27                | 29                | 114                         | 1                           | 0                           | 90                          |
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
| 192.168.201.86 | 100               | 100               | 255               | 21                          | 3                           | 1                           | 30                          |
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
| 192.168.201.88 | 19                | 26                | 28                | 136                         | 1                           | 0                           | 140                         |
+----------------+-------------------+-------------------+-------------------+-----------------------------+-----------------------------+-----------------------------+-----------------------------+
```

swarmctl -m hpstats

- **swarmctl -m hpstats**
- **used to show health processor statistics including the current ongoing cycle**
- **typically used with -a AND -x to output all nodes' information automatically to a file where you will see much more output**

```
root@c-csn1:~/tmp>swarmctl -m hpstats -o
```

nodeIPAddress	HP Exam queue count	HP Replication queue count	HP State	HP ongoing cycle: Cycle number	HP ongoing cycle: Streams examined	HP last cycle: Streams examined	HP ongoing cycle, whole reps: Trims requested	HP last cycle, whole reps: Trims requested
192.168.201.84	0	0	idle (3); running (1)	158	39,882	41,997	0	0
192.168.201.85	0	0	idle (3); running (1)	113	43,201	44,635	0	0
192.168.201.86	0	0	initializing	0	0	0	0	0
192.168.201.88	0	0	running (2); idle (2)	153	28,467	37,776	0	0

swarmctl -m networktest

- **swarmctl -m networktest**
- **this starts a network connectivity test and can take quite some time so use with caution**
- **typically used with -a output all nodes' information automatically to a file where you will see much more output.**
- **-x is not required to output to a file with this option as that is the only method of output**

```
root@c-csn1:~/tmp>swarmctl -m networktest -a
Note: running a networktest may take a long time.
networktest output for c-csn1.enfield.com is in 2020_0323_1019-networktest.csv in this directory
```

```
root@c-csn1:~/tmp>cat 2020_0323_1019-networktest.csv
nodeIPAddress,nodeId,1:nodeIp,1:tcp:reps,1:tcp:responseTime,1:udp:reps,1:udp:responseTime,2:nodeIp,2:tcp:reps,2:tcp:responseTime,2:udp:reps,2:udp:responseTime,3:nodeIp,3:tcp:reps,3:tcp:responseTime,3:udp:r
eps,3:udp:responseTime,4:nodeIp,4:tcp:reps,4:tcp:responseTime,4:udp:reps,4:udp:responseTime
192.168.201.84,c038e3e20bd1e422,192.168.201.84,100,2.67790150642395,100,0.05864691734313965,192.168.201.85,100,3.777550220489502,95,0.10914874076843262,192.168.201.86,100,2.612891435623169,100,0.0660774707
7941895,192.168.201.88,100,2.535768508911133,100,0.06617021560668945
192.168.201.85,b9248f679cadd114,192.168.201.84,100,2.7409722805023193,100,0.08721709251403809,192.168.201.85,100,3.7966408729553223,100,0.05715632438659668,192.168.201.86,100,2.7524921894073486,100,0.06425
38070678711,192.168.201.88,100,2.624887466430664,100,0.06181812286376953
192.168.201.86,bc4c1f4dbf1a8805,192.168.201.84,100,2.8358359336853027,100,0.06846165657043457,192.168.201.85,100,4.392239809036255,99,0.08651709955657959,192.168.201.86,100,2.818005084991455,100,0.0447640041
900634766,192.168.201.88,100,2.6616790294647217,100,0.05435681343078613
192.168.201.88,2ac34b8eb78dad05,192.168.201.84,100,2.9755702018737793,100,0.0733039379119873,192.168.201.85,100,4.481632471084595,99,0.10448074340820312,192.168.201.86,100,2.62528920173645,100,0.0723197460
1745605,192.168.201.88,100,2.668856382369995,100,0.06692433357238777
```

swarmctl -m meminfo

- **swarm -m meminfo**
- **shows memory information including index and overlay index**
- **typically used with -a AND -x to output all nodes' information automatically to a file**

```
root@c-csn1:~/tmp>swarmctl -m meminfo -a
+----------------+----------+----------+----------+--------+------+----------+----+--------+--------+--------+----------+
| nodeIPAddress  | accounte | accounte | accounte | indexS | inde | ioBuffer | ov | overla | overla | overla | overlay: |
|                | dMemoryH | dMemoryI | dMemoryU | lotsAv | xUti |  Memory  | er | y:slot | y:slot | y:stat | totalMem |
|                | ighwater |   nUse   | tilizati | ailabl | liza |          | la | sTotal | sUsed  |   us   |          |
|                |          |          |    on    |   e    | tion |          | y: |        |        |        |          |
|                |          |          |          |        |      |          | en |        |        |        |          |
|                |          |          |          |        |      |          | ab |        |        |        |          |
|                |          |          |          |        |      |          | le |        |        |        |          |
|                |          |          |          |        |      |          | d  |        |        |        |          |
+================+==========+==========+==========+========+======+==========+====+========+========+========+==========+
| 192.168.201.8  | 1041890  | 1041890  | 0        | 53.60  | 0    | 1.42GB   | 1  | 53.69  | 15,479 | author | 2.17GB   |
| 4              |          |          |          | mil    |      |          |    | mil    |        | itativ |          |
|                |          |          |          |        |      |          |    |        |        | e      |          |
+----------------+----------+----------+----------+--------+------+----------+----+--------+--------+--------+----------+
| 192.168.201.8  | 1045386  | 1043247  | 0        | 53.59  | 0    | 1.42GB   | 1  | 53.69  | 15,830 | author | 2.17GB   |
| 5              |          |          |          | mil    |      |          |    | mil    |        | itativ |          |
|                |          |          |          |        |      |          |    |        |        | e      |          |
+----------------+----------+----------+----------+--------+------+----------+----+--------+--------+--------+----------+
| 192.168.201.8  | 1043890  | 1043890  | 0        | 53.67  | 0    | 1.42GB   | 1  | 53.69  | 15,784 | author | 2.17GB   |
| 6              |          |          |          | mil    |      |          |    | mil    |        | itativ |          |
|                |          |          |          |        |      |          |    |        |        | e      |          |
+----------------+----------+----------+----------+--------+------+----------+----+--------+--------+--------+----------+
| 192.168.201.8  | 1043243  | 1043243  | 0        | 53.60  | 0    | 1.42GB   | 1  | 53.69  | 13,701 | author | 2.17GB   |
| 8              |          |          |          | mil    |      |          |    | mil    |        | itativ |          |
|                |          |          |          |        |      |          |    |        |        | e      |          |
+----------------+----------+----------+----------+--------+------+----------+----+--------+--------+--------+----------+
```

swarmctl -m features

- **swarm -m features**
- **numbers of interesting types of requests like md5, if-match, integrity seal, rename, and versioning**
- **typically used with -a AND -x to output all nodes' information automatically to a file where you will see much more output**

```
root@c-csn1:~/tmp>swarmctl -m features -a
+----------------+-------------------------------+-------------------------------+-------------------------------------+-------------------------------+-------------------------------------+
| nodeIPAddress  | Feature: Number of MD5 requests | Feature: Number of if-match requests |  Feature: Number of integrity seal requests |  Feature: Number of rename requests |  Feature: Number of versioning requests |
+================+===============================+===============================+=====================================+===============================+=====================================+
| 192.168.201.84 | 12266                         | 5                             | 0                                   | 0                             | 0                                   |
+----------------+-------------------------------+-------------------------------+-------------------------------------+-------------------------------+-------------------------------------+
| 192.168.201.85 | 18921                         | 2                             | 0                                   | 0                             | 0                                   |
+----------------+-------------------------------+-------------------------------+-------------------------------------+-------------------------------+-------------------------------------+
| 192.168.201.86 | 1987                          | 2                             | 0                                   | 0                             | 0                                   |
+----------------+-------------------------------+-------------------------------+-------------------------------------+-------------------------------+-------------------------------------+
| 192.168.201.88 | 11125                         | 3                             | 0                                   | 0                             | 0                                   |
+----------------+-------------------------------+-------------------------------+-------------------------------------+-------------------------------+-------------------------------------+
```

swarmctl -O SCSP Response Counts

- **swarmctl -O shows all of the SCSP response codes that a node has processed**
- **typically used with -a AND -x to output all nodes' information automatically to a file**
- **you can see below that since I have been using .85 to send these example commands to, that 200 OK is much higher on that node than on other nodes**

```
root@c-csn1:~/tmp>swarmctl -a -O
+----------------+-------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| nodeIPAddress  |  200  | 201 | 202 | 206 | 301 | 304 | 400 | 401 | 404 | 410 | 412 | 4xx | 500 | 503 | 507 | 5xx |
+================+=======+=====+=====+=====+=====+=====+=====+=====+=====+=====+=====+=====+=====+=====+=====+
| 192.168.201.84 | 258   | 0   | 0   | 0   | 334 | 0   | 0   | 0   | 3   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
+----------------+-------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 192.168.201.85 | 2,924 | 6   | 0   | 0   | 12  | 0   | 0   | 1   | 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
+----------------+-------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 192.168.201.86 | 102   | 2   | 0   | 0   | 397 | 0   | 0   | 2   | 5   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
+----------------+-------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 192.168.201.88 | 296   | 0   | 0   | 0   | 344 | 0   | 0   | 0   | 3   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
+----------------+-------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

swarmctl -p specify user_ password for admin access

- **swarmctl -p [user:password]**
- **the commands run previously in this deck worked ONLY because we try to use 2 different default passwords if the -p option is not specified**
- **if the admin password for the cluster is "ourpwdofchoicehere" or "caringo", then swarmctl doesn't require you to include -p [user:password] for commands that make changes or otherwise need admin rights**
- **if you have a non-default password, which is the better strategy, then simply add -p [user:password] with your commands for authentication.**

swarmctl -P Persistent Settings Stream (PSS)

- **swarmctl -P export the Persistent Settings Stream to standard out**
- **requires admin credentials (-p user:pass) - see note on -p option.**
- **use -x to export to a file**
- **doesn't support -a - if you need the PSS from multiple nodes, use -d [ip]**

```
root@c-csn1:~/tmp>swarmctl -P
Successfully retrieved cluster persistent settings stream from 03fd85d03e861697e81337eaf2b1af85
recovery.volMaintenanceInterval=10800
scsp.autoRecursiveDelete=1
log.forceAudit=0
policy.eCEncoding=unspecified anchored
health.iterTasks=2
cip.queryRetryMultiplier=1.00000
metrics.target=
ec.segmentConsolidationFrequency=10
power.cap=100
policy.eCMinStreamSize=1Mb anchored
```

swarmctl -q Quick summary

- **swarmctl -q get a quick summary of the cluster status**
- **what you might see from the cluster status page on port 90**

```
root@c-csn1:~/tmp>swarmctl -q
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
|    Type    |     name      |   status    | ava |  usedSpace   |   maxSpace   | streamCount  |  swVer   | errC | volE |
|            |               |             | ilP |              |              |              |          | ount | rrs  |
|            |               |             | erc |              |              |              |          |      |      |
|            |               |             | ent |              |              |              |          |      |      |
+============+===============+=============+=====+==============+==============+==============+==========+======+======+
| Cluster    | c-csn1.enfiel |     ok      | 62% |      28.89GB |     121.43GB |      120,035 |  11.0.3  |    0 |    0 |
|            | d.com         |             |     |              |              |              |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
| Subcluster | default       |     ok      | 62% |      28.89GB |     121.43GB |      120,035 |  11.0.3  |    0 |    0 |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 75% |       9.91GB |      40.48GB |       44,364 |  11.0.3  |    0 |    0 |
|            | 5             |             |     |              |              |              |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 76% |       9.42GB |      40.48GB |       37,915 |  11.0.3  |    0 |    0 |
|            | 4             |             |     |              |              |              |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
| Chassis    | 192.168.201.8 |   retired   |  0% |       0.00MB |       0.00MB |            0 |  11.0.3  |    0 |    0 |
|            | 6             |             |     |              |              |              |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 76% |       9.56GB |      40.48GB |       37,756 |  11.0.3  |    0 |    0 |
|            | 8             |             |     |              |              |              |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+--------------+----------+------+------+
```

swarmctl -Q dmesg_ hwinfo_ healthreport

- **swarmctl -Q**
- **allows options for exporting dmesg, hwinfo, and the health report from a single node**
- **this replaces other scripts like hwinfo-dmesg-grab.sh and collect_health_reports.sh**
- **without a modifier, assumes "dmesg" by default**

```
root@c-csn1:~/tmp>swarmctl -Q
Node 192.168.201.85 dmesg
[Fri Mar 20 20:33:21 2020] Linux version 4.19.56 (root@ad06e4fd7169) (gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1)) #1 SMP Wed Nov 13 20:02:43 UTC 2019
[Fri Mar 20 20:33:21 2020] Command line: initrd=fsimage/000c292422d5 ramdisk_size=200000 root=/dev/ram0 castor_cfg=http://192.168.201.3:8088/config?mac=000c292422d5 castor_net=balance-alb: BOOT_IMAGE=kerne
l/000c292422d5
[Fri Mar 20 20:33:21 2020] Disabled fast string operations
[Fri Mar 20 20:33:21 2020] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[Fri Mar 20 20:33:21 2020] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[Fri Mar 20 20:33:21 2020] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[Fri Mar 20 20:33:21 2020] x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
[Fri Mar 20 20:33:21 2020] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using 'standard' format.
[Fri Mar 20 20:33:21 2020] BIOS-provided physical RAM map:
[Fri Mar 20 20:33:21 2020] BIOS-e820: [mem 0x0000000000000000-0x000000000008dfff] usable
[Fri Mar 20 20:33:21 2020] BIOS-e820: [mem 0x000000000008e000-0x000000000009ffff] reserved
```

swarmctl -Q dmesg

- **swarmctl -Q dmesg**
- **export dmesg output**
- **most commonly used with -x to export output to a file**
- **can use with -a to grab the dmesg from ALL nodes in the cluster**

```
root@c-csn1:~/tmp>swarmctl -Q dmesg -x
Node 192.168.201.85 dmesg
dmesg for 192.168.201.85 written to 2020_0323_1449-dmesg-192.168.201.85.txt
root@c-csn1:~/tmp>swarmctl -Q dmesg -x -a
Node 192.168.201.84 dmesg
dmesg for 192.168.201.84 written to 2020_0323_1450-dmesg-192.168.201.84.txt
Node 192.168.201.85 dmesg
dmesg for 192.168.201.85 written to 2020_0323_1450-dmesg-192.168.201.85.txt
Node 192.168.201.86 dmesg
dmesg for 192.168.201.86 written to 2020_0323_1450-dmesg-192.168.201.86.txt
Node 192.168.201.88 dmesg
dmesg for 192.168.201.88 written to 2020_0323_1450-dmesg-192.168.201.88.txt
```

swarmctl -Q hwinfo

- **swarmctl -Q hwinfo**
- **results in hwinfo output - can take some time to run**
- **typically run with -x -a to get all nodes' output to a file**

```
root@c-csn1:~/tmp>swarmctl -Q hwinfo -x -a
Node 192.168.201.84 hwinfo
hwinfo for 192.168.201.84 written to 2020_0323_1453-hwinfo-192.168.201.84.txt
Node 192.168.201.85 hwinfo
hwinfo for 192.168.201.85 written to 2020_0323_1453-hwinfo-192.168.201.85.txt
Node 192.168.201.86 hwinfo
hwinfo for 192.168.201.86 written to 2020_0323_1453-hwinfo-192.168.201.86.txt
Node 192.168.201.88 hwinfo
hwinfo for 192.168.201.88 written to 2020_0323_1454-hwinfo-192.168.201.88.txt
```

swarmctl -Q healthreport

- **swarmctl -Q get the health report in json format from a single node**
- **use with -x to export to json format**
- **use with -a also to export all nodes output to json files**

```
root@c-csn1:~/tmp>swarmctl -Q healthreport -a -x
Node 192.168.201.84 healthreport
healthreport for 192.168.201.84 written to 2020_0323_1454-healthreport-192.168.201.84.json
Node 192.168.201.85 healthreport
healthreport for 192.168.201.85 written to 2020_0323_1454-healthreport-192.168.201.85.json
Node 192.168.201.86 healthreport
healthreport for 192.168.201.86 written to 2020_0323_1454-healthreport-192.168.201.86.json
Node 192.168.201.88 healthreport
healthreport for 192.168.201.88 written to 2020_0323_1454-healthreport-192.168.201.88.json
```

swarmctl -R, -S Cluster Restart and Shutdown

- swarmctl -R [chassis] and swarmctl -S [chassis]
- -R restarts the whole cluster, -S shuts down the whole cluster
- dialog box for "are you sure?"
- requires admin access, so use -p admin:password if not using default admin passwords
- can add "chassis", like "`swarmctl -S chassis -d [chassis-ip]`" to shut down a single chassis
- can add -n to operate per chassis against only IP addresses in a NODES.csv file located in the script directory:
  example, if ./NODES.csv has 2 IP addresses, "`./swarmctl -R -d [chassis-ip] -n`" would only reboot the 2 IPs in NODES.csv

```
root@c-csn1:~/tmp>swarmctl -R
Are you sure you want to restart this cluster? Storage will be offline for an extended period until restart has completed. [y/N]: y
API reports restart command success.
```

swarmctl -t Disk related statistics

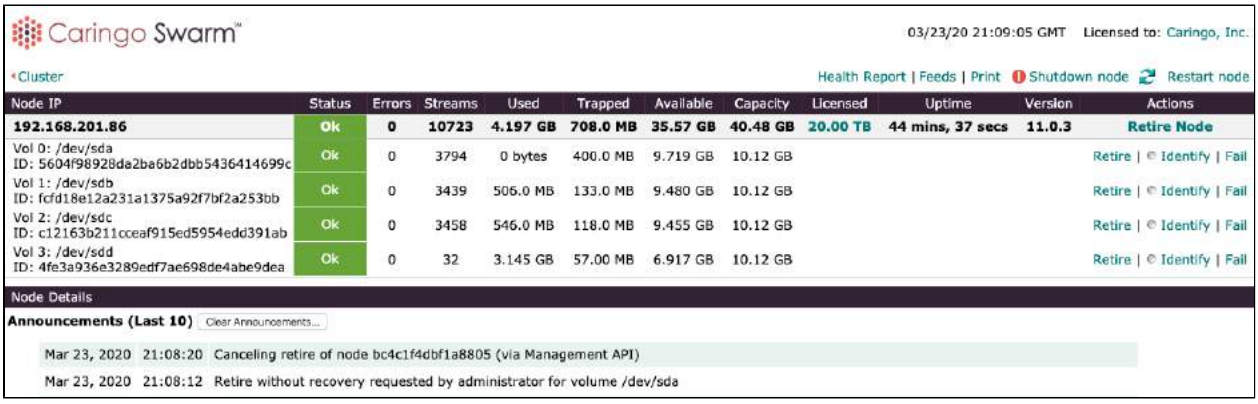
- **swarmctl -t gives specific details on each disk on a node**
- **use -a -x to get more disk statistics for all nodes in the cluster**
- **great for seeing changes over time and tracking potentially bad disks- cron job**

```
root@c-csn1:~/tmp>swarmctl -t
+-----------------+----------+-----------------------------+-------------+----------+----------+----------+----------+----------+-----------------------------+-----------------+-----------+
| nodeIPAddress   |   name   |            uuid             | trappedBytes | trapRatio | usedSpace | availSpace | maxSpace | streamCount | status |     largestStreamUuid     | largestStreamSize | journalBid |
+=================+==========+=============================+=============+==========+==========+==========+==========+==========+=============================+=================+===========+
| 192.168.201.85  | /dev/sda | 3810719596 1f9bac73d88062ac | 938         | 0.865    | 3.15GB   | 6029     | 10.12GB  | 1,038    | ok     | b191f88b765f52ee04293d544 | 1048            | 0         |
|                 |          | e250f5                      |             |          |          |          |          |          |        | 02c1101                   |                 |           |
+-----------------+----------+-----------------------------+-------------+----------+----------+----------+----------+----------+-----------------------------+-----------------+-----------+
| 192.168.201.85  | /dev/sdb | c3470745206b23d505c459f1c9  | 1537        | 0.805    | 2.22GB   | 6362     | 10.12GB  | 21,058   | ok     |                           | 102             | 1         |
|                 |          | 9a67f6                      |             |          |          |          |          |          |        |                           |                 |           |
+-----------------+----------+-----------------------------+-------------+----------+----------+----------+----------+----------+-----------------------------+-----------------+-----------+
| 192.168.201.85  | /dev/sdc | aa995b97265dd862386bef9d5d  | 1564        | 0.804    | 2.16GB   | 6399     | 10.12GB  | 6,935    | ok     |                           | 102             | 0         |
|                 |          | d8d005                      |             |          |          |          |          |          |        |                           |                 |           |
+-----------------+----------+-----------------------------+-------------+----------+----------+----------+----------+----------+-----------------------------+-----------------+-----------+
| 192.168.201.85  | /dev/sdd | a18e8592836fd39d1da925de6a  | 1513        | 0.805    | 2.38GB   | 6230     | 10.12GB  | 15,348   | ok     |                           | 76              | 1         |
|                 |          | 4f1e4c                      |             |          |          |          |          |          |        |                           |                 |           |
+-----------------+----------+-----------------------------+-------------+----------+----------+----------+----------+----------+-----------------------------+-----------------+-----------+
```

swarmctl -u Unretire currently retiring drives

- **swarmctl -u stops retiring drives that are currently retiring**
- **does not resurrect drives that have already retired**
- **use with -a to stop retiring on all nodes in the cluster**

```
root@c-csn1:~/tmp>swarmctl -u -d 192.168.201.86
Attempting to cancel retire of node 192.168.201.86
Management API reports success in canceling retire of node 192.168.201.86
```



swarmctl -U User management

- swarmctl -U shows you, and allows you to change, the list of admin users
- run without options shows you the list of admin users
- requires -p [user:password] if you are not using default admin credentials
- use -V [password] to add the password to a new user, specified like: swarmctl -U [new admin user] -V [password for new admin user]
- change password for existing user like: swarmctl -U [current admin username] -V [new password for admin user]
- be advised when changing the admin password that other systems might have the admin password set and could potentially break if changed until they are updated

```
root@c-csn1:~/tmp>swarmctl -U
API reports ['admin'] as admin users on c-csn1.enfield.com
root@c-csn1:~/tmp>swarmctl -U bob -V password
bob:password
API reports success=True setting/changing password for bob
root@c-csn1:~/tmp>swarmctl -U
API reports ['admin', 'bob'] as admin users on c-csn1.enfield.com
root@c-csn1:~/tmp>swarmctl -U bob -V newpassword
bob:newpassword
API reports success=True setting/changing password for bob
```

swarmctl -V Variables

- **swarmctl -V [option] adds a variable option**
- **is not used standalone- this option is used to add information to other parameters**
- **examples include -C, -D, -U, -z**

swarmctl -w Recovery Reports

- **swarmctl -w gets the recovery reports for disks that have been retired or are retiring**
- **useful with -x -a to export all recovery reports to file for further analysis**

```
root@c-csn1:~/tmp>swarmctl -w -a -x
recoveries output for c-csn1.enfield.com is in 2020_0323_1544-recoveries.csv in this directory
root@c-csn1:~/tmp>head 2020_0323_1544-recoveries.csv
node,localVolumeID,localVolumeName,recoveryType,remoteVolumeID,remoteVolumeIP,remoteVolumeName,state,timeEnded,timeStarted
192.168.201.84,78765d2b5d28fb613d55e688eaf878bf,/dev/sdd,ECR,63319b6940a91895ff3ff748dc67fa71,unknown,unknown,completed,2020-03-23T19:38:59.770148Z,2020-03-20T20:24:04.873824Z
192.168.201.84,78765d2b5d28fb613d55e688eaf878bf,/dev/sdd,FVR,63319b6940a91895ff3ff748dc67fa71,unknown,unknown,completed,2020-03-23T19:38:59.770021Z,2020-03-20T20:24:04.875415Z
192.168.201.84,8886bbaf6685815fbf0e92d2f236dc62,/dev/sda,ECR,63319b6940a91895ff3ff748dc67fa71,unknown,unknown,completed,2020-03-23T19:54:04.193961Z,2020-03-20T20:24:04.910437Z
192.168.201.84,8886bbaf6685815fbf0e92d2f236dc62,/dev/sda,FVR,63319b6940a91895ff3ff748dc67fa71,unknown,unknown,completed,2020-03-23T19:54:04.193776Z,2020-03-20T20:24:04.912178Z
192.168.201.84,78765d2b5d28fb613d55e688eaf878bf,/dev/sdd,ECR,bea15642b51bfc1e9540f0e5ab5ed6be,unknown,unknown,completed,2020-03-23T19:38:59.770204Z,2020-03-20T20:35:53.697240Z
192.168.201.84,78765d2b5d28fb613d55e688eaf878bf,/dev/sdd,FVR,bea15642b51bfc1e9540f0e5ab5ed6be,unknown,unknown,completed,2020-03-23T19:38:59.770092Z,2020-03-20T20:35:53.698169Z
192.168.201.84,8886bbaf6685815fbf0e92d2f236dc62,/dev/sda,ECR,bea15642b51bfc1e9540f0e5ab5ed6be,unknown,unknown,completed,2020-03-23T19:54:04.194020Z,2020-03-20T20:35:53.758803Z
192.168.201.84,8886bbaf6685815fbf0e92d2f236dc62,/dev/sda,FVR,bea15642b51bfc1e9540f0e5ab5ed6be,unknown,unknown,completed,2020-03-23T19:54:04.193901Z,2020-03-20T20:35:53.759541Z
192.168.201.84,78765d2b5d28fb613d55e688eaf878bf,/dev/sdd,ECR,9af0fdab6bb9a7525a34c100f2816301,unknown,unknown,completed,2020-03-23T19:38:59.770239Z,2020-03-20T20:44:53.509530Z
```

swarmctl -x export to csv format

- swarmctl -x -[other option] allows you to export the output to csv (or json in some cases) format for further analysis
- useful with -a to export all nodes' information to file for further analysis
- works with multiple other options (it is not a standalone option)
- if used with -a, a zipped bundle will be made of the output and you will be prompted with the option to delete the individual files.
- some flags [-Q, -m, -z] allow multiple options for a single node, in which case the output may be bundled in a zip

```
[root@c-csn1 tmp]# swarmctl -m commstats hpstats -x
commstats output for c-csn1.enfield.com is in 2020_0407_0930-commstats.csv in this directory
hpstats output for c-csn1.enfield.com is in 2020_0407_0930-hpstats.csv in this directory
Compressed files left in 2020_0407_0930-c-csn1.enfield.com-commstats+hpstats.zip.
These files have been included in the zip file and can be deleted:
['2020_0407_0930-commstats.csv', '2020_0407_0930-hpstats.csv']
Delete these files? [y/N]: n
[root@c-csn1 tmp]#
```

swarmctl -X Don’t persist sessions

- **swarmctl -X**
- **if you have hundreds of nodes, running cluster-wide operations might cause your client to run out of file descriptors**
- **run this to prevent session persistence while running other swarmctl options**
- **no need to use except if requested by Caringo Support**

swarmctl -z Component Log Level

- swarmctl -z [component] -V [level]
- can run with multiple components in the same call: swarmctl -z component1 component2 [-V [level]]
- this option allows you to change the log level of a particular component instead of changing the log level of every component
- useful when troubleshooting very specific issues, especially in large environments where debug level produces too much data to sift through
- run with no options to see the default levels
- use like "-z -V 0" to reset all components to their default log level
- use with -a to affect all nodes

```
root@c-csn1:~/tmp>swarmctl -z
+----------------+------------------+-----------+
|      Node      |    Component     | Log Level |
+================+==================+===========+
| 192.168.201.85 | ADMIN            | 20        |
+----------------+------------------+-----------+
| 192.168.201.85 | ASYNC FILE       | 20        |
+----------------+------------------+-----------+
| 192.168.201.85 | BUFFERS          | 20        |
+----------------+------------------+-----------+
| 192.168.201.85 | CACHE            | 20        |
+----------------+------------------+-----------+
| 192.168.201.85 | COLLECTION       | 20        |
+----------------+------------------+-----------+
| 192.168.201.85 | CONFIG           | 20        |
+----------------+------------------+-----------+
```

```
root@c-csn1:~/tmp>swarmctl -z LICENSE
+----------------+-----------+-----------+
|      Node      | Component | Log Level |
+================+===========+===========+
| 192.168.201.85 | LICENSE   | 20        |
+----------------+-----------+-----------+
root@c-csn1:~/tmp>swarmctl -z LICENSE -V 10
+----------------+-----------+--------------------+---------------+
|      Node      | Component | Previous Log Level | New Log Level |
+================+===========+====================+===============+
| 192.168.201.85 | LICENSE   | 20                 | 10            |
+----------------+-----------+--------------------+---------------+
root@c-csn1:~/tmp>swarmctl -z -V 0
+----------------+------------------+--------------------+---------------+
|      Node      |    Component     | Previous Log Level | New Log Level |
+================+==================+====================+===============+
| 192.168.201.85 | ADMIN            | 20                 | 20            |
+----------------+------------------+--------------------+---------------+
| 192.168.201.85 | ASYNC FILE       | 20                 | 20            |
+----------------+------------------+--------------------+---------------+
| 192.168.201.85 | BUFFERS          | 20                 | 20            |
+----------------+------------------+--------------------+---------------+
| 192.168.201.85 | CACHE            | 20                 | 20            |
+----------------+------------------+--------------------+---------------+
```

swarmctl --debug debug mgmt api calls

- swarmctl -[other flags] --debug {[api args http returns]} [api args http returns]
- this flag is used to show what's going on under the covers during swarmctl runs
- the "api" variable shows all of the mgmt api calls (this is default if no variable specified)
- the "args" variable just shows with args were called
- the "http" flavariableg shows all of the http traffic and headers
- the "returns" variable is verbose

```
[root@c-csn1 tmp]# swarmctl --debug -q
APICommand: logOperations=True
changeAuth: from None:None to admin:caringo
apiGet: "http://192.168.201.84:91/api/storage/clusters" -> 200
apiGet: "http://192.168.201.84:91/api/storage/clusters/c-csn1.enfield.com/summary" -> 200
apiGet: "http://192.168.201.84:91/api/storage/nodes" -> 200
apiGet: "http://192.168.201.84:91/api/storage/nodes/c038e3e20bd1e422" -> 200
apiGet: "http://192.168.201.85:91/api/storage/nodes/b9248f679cadd114" -> 200
apiGet: "http://192.168.201.88:91/api/storage/nodes/2ac34b8eb78dad05" -> 200
apiGet: "http://192.168.201.84:91/api/storage/nodes/_self" -> 200
validateUser: "http://192.168.201.84:91/api/validateUser" user: admin
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
|    Type    |     name      |   status    | ava |  usedSpace   |   maxSpace   | streamCount |  swVer   | errC | volE |
|            |               |             | ilP |              |              |             |          | ount | rrs  |
|            |               |             | erc |              |              |             |          |      |      |
|            |               |             | ent |              |              |             |          |      |      |
+============+===============+=============+=====+==============+==============+=============+==========+======+======+
| Cluster    | c-csn1.enfiel |     ok      | 63% |      28.88GB |     121.43GB |     120,033 |  11.0.3  |    0 |    0 |
|            | d.com         |             |     |              |              |             |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
| Subcluster | default       |     ok      | 63% |      28.88GB |     121.43GB |     120,033 |  11.0.3  |    0 |    0 |
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 75% |      10.11GB |      40.48GB |      38,217 |  11.0.3  |    0 |    0 |
|            | 5             |             |     |              |              |             |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 76% |       9.40GB |      40.48GB |      39,238 |  11.0.3  |    0 |    0 |
|            | 4             |             |     |              |              |             |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
| Chassis    | 192.168.201.8 |     ok      | 76% |       9.37GB |      40.48GB |      42,578 |  11.0.3  |    0 |    0 |
|            | 8             |             |     |              |              |             |          |      |      |
+------------+---------------+-------------+-----+--------------+--------------+-------------+----------+------+------+
```

swarmctl --feeds Feeds tables

- **swarmctl --feeds shows the feeds tables for indexer and replication feeds**
- **this is the same information as seen in the feeds definition page**
- **does *not* show you feed statistics**

```
root@c-csn1:~/tmp>swarmctl --feeds -x
No replicationfeeds defined on c-csn1.enfield.com
1 searchfeed defined on c-csn1.enfield.com
searchfeed: IndexerFeed-5.6.12-to-c-csn1-indexer2-3
{   'destination': {   'fullMetadata': True,
                       'host': '192.168.201.203',
                       'indexAlias': 'c-csn1.enfield.com0',
                       'insertBatchSize': 100,
                       'insertBatchTimeout': 1,
                       'port': 9200},
    'id': 0,
    'isDefault': True,
    'lastchanged': '2020-03-10T16:03:57.000+00:00',
    'latency': 30.0,
    'name': 'IndexerFeed-5.6.12-to-c-csn1-indexer2-3',
    'nodeletes': False,
    'noversioned': False,
    'paused': False,
    'type': 'Search'}
No s3backupfeeds defined on c-csn1.enfield.com
```

swarmctl --feed-control

- **Apply to the whole feed across all nodes as applicable**
- **--feed-control [action] allows you to perform these operations: pause, resume, restart, setdefault**
- **--feed-type [type] specifies one of three types of feeds: searchfeeds, replicationfeeds, s3backupfeeds**
- **--feed-number [value] specifies the particular feed**
- **All of the above are required per operation**

```
[root@c-csn1 tmp]# swarmctl --feed-type searchfeeds --feed-number 2 --feed-control pause
Attempting to pause searchfeed #2
Management API reports success with pause of searchfeed #2
```

```
[root@c-csn1 tmp]# swarmctl --feed-type searchfeeds --feed-number 2 --feed-control resume
Attempting to resume searchfeed #2
Management API reports success with resume of searchfeed #2
```

swarmctl --feed-control and --node-feed-restart

- **--feed-control [action] allows you to perform these operations: pause, resume, restart, setdefault**
- **--feed-type [type] specifies one of three types of feeds: searchfeeds, replicationfeeds, s3backupfeeds**
- **--feed-number [value] specifies the particular feed**
- **--node-feed-restart - restarts the feed ONLY for a particular node**
- **--feed-type and --feed-number are both required for both --feed-action and --node-feed-restart**

```
[root@c-csn1 tmp]# swarmctl --feed-type searchfeeds --feed-number 2 --feed-control pause
Attempting to pause searchfeed #2
Management API reports success with pause of searchfeed #2
```

```
[root@c-csn1 tmp]# swarmctl --feed-type searchfeeds --feed-number 2 --feed-control resume
Attempting to resume searchfeed #2
Management API reports success with resume of searchfeed #2
```

swarmctl --license License information

- **swarmctl --license shows the currently deployed license**
- **this is the same information as seen on the license page or in the license itself**

```
root@c-csn1:~/tmp>swarmctl --license
{   'clusterDescription': 'Ace Lab Cluster',
    'cn': 'Caringo, Inc.',
    'co': 'USA',
    'expirationDate': None,
    'featureClusterMaxObjects': 0,
    'featureClusterMaxTB': 20.0,
    'featureContentIndexing': True,
    'featureErasureCoding': True,
    'featureHardwareCheck': False,
    'featureHealthReportRequired': False,
    'featureKeys': [],
    'featureMinimumMinReps': 1,
    'featurePlatformId': '',
    'featureVolumeLifetime': 'unlimited',
    'l': 'Austin',
    'licenseFormat': '1.1',
```

swarmctl --policies Policies

- **swarmctl --policies shows you the policies as currently evaluated by the cluster**
- **shows the policies for Replicas, ECEncoding, ECMinStream, and SizeVersioning**

```
root@c-csn1:~/tmp>swarmctl --policies -V Replicas
4 policies found on c-csn1.enfield.com
Policy: Versioning
{   'evaluatedValue': 'disabled',
    'headerName': 'Policy-Versioning',
    'id': 1,
    'name': 'Versioning',
    'settingDefValue': 'disallowed',
    'settingMibName': 'policyVersioning',
    'settingName': 'policy.versioning',
    'settingValue': 'disallowed',
    'validValues': ['disallowed', 'suspended', 'allowed']}
Policy: ECMinStreamSize
{   'evaluatedValue': '1000000',
    'headerName': 'Policy-ECMinStreamSize',
    'id': 2,
    'name': 'ECMinStreamSize',
    'settingDefValue': '1Mb anchored',
    'settingMibName': 'policyECMinStreamSize',
    'settingName': 'policy.eCMinStreamSize',
    'settingValue': '1Mb anchored',
    'validValues': ['20Mb', '1Gb', '20Mb anchored']}
Policy: ECEncoding
{   'evaluatedValue': 'unspecified',
    'headerName': 'Policy-ECEncoding',
    'id': 3,
    'name': 'ECEncoding',
    'settingDefValue': 'unspecified anchored',
    'settingMibName': 'policyECEncoding',
    'settingName': 'policy.eCEncoding',
    'settingValue': 'unspecified anchored',
    'validValues': ['unspecified', '5:2', 'disabled', '5:2 anchored']}
Policy: Replicas
{   'evaluatedValue': 'min:2 max:16 default:2',
    'headerName': 'Policy-Replicas',
    'id': 4,
    'name': 'Replicas',
    'settingDefValue': 'min:2 max:16 default:2 anchored',
    'settingMibName': 'policyReplicas',
    'settingName': 'policy.replicas',
    'settingValue': 'min:2 max:16 default:2 anchored',
    'validValues': [   'min:1 max:15 default:2',
                       'min:2 max:16 default:3',
                       'min:2 max:10 default:2 anchored']}
```

swarmrestart

- swarmrestart [options]
- swarmrestart is a binary script replacement for the -G option in snmp-castor-tool.sh
- used to rolling restart a cluster
- common options include
  - -p "user:password"
  - -n - reboots only those nodes in the local NODES.csv file (like -n in snmp-castor-tool.sh)
  - -m [minutes] - number of minutes to wait for a booted node to mount the disk. 45 minutes by default
  - -d [ip address] - any IP in the cluster from which the script will read all storage IPs
  - -v [version] - specify the version of storage nodes to restart. Default will restart any. Useful if you have already upgraded some nodes and want only to rolling reboot the nodes that haven't been upgraded yet.
  - -x [filename] - a list of IPs that should NOT be rebooted
  - -w [boot wait] - wait this long after rebooting a node until trying to contact it. Faster booting nodes, you can set this lower than the 5 minutes default.
- this script will generate a log file while running to show you exactly what's going on at each timestamp

```
root@c-csn1:~/tmp>swarmrestart -d 192.168.201.85 -w 2 -p admin:caringo -n -u 0
Preparing for rolling restart of cluster: c-csn1.enfield.com
Collecting cluster summary via 192.168.201.85
Ignoring server at 192.168.201.86 with status retired
3 Servers ready for restart:
192.168.201.84[11.0.3] 4 vols, VMware, Inc. VMware Virtual sn:VMware-56 4d c8 .. da 63, up 1 hour 27 minutes/ok
192.168.201.85[11.0.3] 4 vols, VMware, Inc. VMware Virtual sn:VMware-56 4d 5a .. 22 d5, up 1 hour 27 minutes/ok
192.168.201.88[11.0.3] 4 vols, VMware, Inc. VMware Virtual sn:VMware-56 4d 84 .. 56 4c, up 1 hour 27 minutes/ok

Are you sure you want to restart 3 servers in cluster c-csn1.enfield.com? [y/N]: y
Attempting restart of 192.168.201.84... restart command reports success.
192.168.201.84: waiting
```

Wrap-Up

- **There are plenty of options.**
- **Spend time in the lab *before* you need them, to know how they work and what might be useful for your environment.**
- **Please contact support with any issues or requests.**

How to use indexer-enumerator.sh

If you want to enumerate an entire cluster and you have an Search (Indexer) Feed already configured, you may use the `indexer-enumerator.sh` script from the support tools bundle to do so.

For a smaller query, it might be easier to use the Content UI portal (if it's installed on a Content Gateway). This script is for enumerating potentially large data sets where the UI would be less helpful.

Tips

- You can run the script with "`bash -x`" to get examples of the curl syntax that you can adapt for your own custom indexer calls.
- You can search by domain, bucket, prefix, size, date written, and type of object.
- When you have the match you want, you can remove the `-orc` options and from there output the object match results to file.

> ⚠ Be careful to run this script from a directory/partition with plenty of disk space if you are returning millions of objects.
>
> For full enumerations of larger data sets, you may want to add the `-s` option to echo the enumerator loop count. Each call to the indexer has a maximum of 10k returned values, so knowing how many iterations of that 10k figure the script has returned is valuable for larger enumerations.

Instructions

This is an extended example of how you can use this script to investigate what is in your cluster.

> ⓘ The environmental variable SCSP_HOST is set to a storage node IP to avoid having to put `-a [storage-node-ip]` on every example below.

Listing domains

Run `indexer-enumerator.sh -D` to find out what domains exist in your cluster.

```
[root@c-csn1 tmp]# indexer-enumerator.sh -D
A complete domain listing can be found here: ./OUTPUTDIR-2020_0722-124732/domains.txt
```

Because a domain listing should be short, I use the `-or` options to output the results to stdout:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -D -or

Here are the domains:
test1.c-csn1.enfield.com
caringodrive.c-csn1.enfield.com
filefly-c-csn1.enfield.com
c-csn1-test1.enfield.com
c-csn1-admindomain
m-csn4.enfield.com
nfstest1.enfield.com
filefly-s3-target.c-csn1.enfield.com
es-backups.enfield.com
c-csn1.enfield.com
bob.is.great.com
s3-compatible
c-csn1-cfs1.enfield.com
c-csn1-s3-target.enfield.com
```

Counting objects and space usage

Now I know the domains but not what's in them. Next, to find out how many objects are in each domain and how much space each takes, I combine the `-c` option with the `-d ALL` option:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -d ALL -c

Enumerating all domains in the cluster:
A complete domain listing can be found here: ./OUTPUTDIR-2020_0722-124949/domains.txt
test1.c-csn1.enfield.com/ has 3147 unique matching objects of stype: all, withreps, uses 458.44MB
caringodrive.c-csn1.enfield.com/ has 20 unique matching objects of stype: all, withreps, uses 156
filefly-c-csn1.enfield.com/ has 1597 unique matching objects of stype: all, withreps, uses 9.32GB
c-csn1-test1.enfield.com/ has 1114 unique matching objects of stype: all, withreps, uses 971.29MB
c-csn1-admindomain/ has 38 unique matching objects of stype: all, withreps, uses 382.00bytes disk
m-csn4.enfield.com/ has 8 unique matching objects of stype: all, withreps, uses 184.14MB disk spa
nfstest1.enfield.com/ has 19 unique matching objects of stype: all, withreps, uses 13.59MB disk s
filefly-s3-target.c-csn1.enfield.com/ has 8217 unique matching objects of stype: all, withreps, u
es-backups.enfield.com/ has 41360 unique matching objects of stype: all, withreps, uses 3.69GB di
c-csn1.enfield.com/ has 129 unique matching objects of stype: all, withreps, uses 2.12GB disk spa
bob.is.great.com/ has 11 unique matching objects of stype: all, withreps, uses 10.81MB disk space
s3-compatible/ has 5 unique matching objects of stype: all, withreps, uses 5.86MB disk space
c-csn1-cfs1.enfield.com/ has 9853 unique matching objects of stype: all, withreps, uses 259.00MB
c-csn1-s3-target.enfield.com/ has 76 unique matching objects of stype: all, withreps, uses 428.23


Only streams counts are listed.  To get the streams themselves, remove the -c flag.
All domains: 65594 unique matching objects of stype: all, withreps, uses 18.21GB disk space
```

This gives me a good idea of what's in my cluster.

Counting untenanted objects

What it does not show me are the *untenanted* objects (those not in any domain). Older clusters may not have *any* domains and so all of the objects would be untenanted. Newer clusters will have most or all objects tenanted and use `enforceTenancy=true` in the cluster configuration to ensure that all objects are in a domain.

We can see if we have any untenanted objects by using the `-t` option. I will again use the `-c` option just to get a count of the number of objects.

```
[root@c-csn1 tmp]# indexer-enumerator.sh -t -c

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

Untenanted streams enumerated: 9 unique objects, withreps, uses 101.44KB disk space
```

By this, I learn that I have only 9 untenanted objects in this particular cluster.

Counting buckets

Going back to the all domains output, I see the `c-csn1-test1.enfield.com` domain looks interesting to me because the domain name doesn't give me a good idea what's in it (in the way that the `filefly-c-csn1.enfield.com` and `es-backups.enfield.com` do).

So, let's drill down into that domain by using the `-d c-csn1-test1.enfield.com` option.

How many buckets live in here?

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -B -c

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 20 unique objects of stype: bucket, withreps, uses 0 disk space.
```

There appear to be 20 buckets here, and they seem to use no disk space. That's because I asked for *only* bucket objects, which don't take up data. To see how much data resides inside a particular bucket, I would need to do a query on that bucket. Also, there might be unnamed objects that live in this domain (that is, are named by UUID and do not live in a bucket).

Let's see what buckets exist in this domain (not just count them, as we did above):

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -B

Starting to enumerate the requested streams in domain: c-csn1-test1.enfield.com

c-csn1-test1.enfield.com/.TOKEN
c-csn1-test1.enfield.com/Bucket15917374547579_0
c-csn1-test1.enfield.com/Bucket15917374547579_1
c-csn1-test1.enfield.com/Bucket15917374547579_2
c-csn1-test1.enfield.com/Bucket15917374547579_3
c-csn1-test1.enfield.com/Bucket15917374547579_4
c-csn1-test1.enfield.com/Bucket15917374547579_5
c-csn1-test1.enfield.com/Bucket15917374547579_6
c-csn1-test1.enfield.com/Bucket15917374547579_7
c-csn1-test1.enfield.com/Bucket15917374547579_8
c-csn1-test1.enfield.com/Bucket15917374547579_9
c-csn1-test1.enfield.com/Bucket15917383799242_0
c-csn1-test1.enfield.com/Bucket15917383799242_1
c-csn1-test1.enfield.com/Bucket15917383799242_2
c-csn1-test1.enfield.com/Bucket15917383799242_3
c-csn1-test1.enfield.com/Bucket15917383799242_4
c-csn1-test1.enfield.com/pants
c-csn1-test1.enfield.com/10kbuckettest
c-csn1-test1.enfield.com/superpants
c-csn1-test1.enfield.com/20200622

c-csn1-test1.enfield.com/ has 20 unique objects of stype: bucket, withreps, uses 0 disk space.
```

Searching objects

I see that I have a bucket named "pants". Let's see how many objects live in my `pants` bucket.

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -b pants -c

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/pants has 3 unique objects of stype: all, withreps, uses 11.83KB disk sp
```

As there are only three, I will output them to stdout (keeping the `-or` flags and removing the `-c` flag):

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -b pants

Starting to enumerate the requested streams in domain: c-csn1-test1.enfield.com

c-csn1-test1.enfield.com/pants/vimium-options.json
c-csn1-test1.enfield.com/pants/vimium-options-2020-mbp.json
c-csn1-test1.enfield.com/pants/plugins.txt

c-csn1-test1.enfield.com/pants has 3 unique objects of stype: all, withreps, uses 11.83KB disk sp
```

I keep using the `-c` option because I could potentially make a query that returns millions if not billions of results. Certainly I don't want to do that right now. From the above, I see that I have 3 files in that bucket.

Because that domain should be an FQDN that resolves to the Content Gateway (or Swarm cluster, if not using Gateway), I can just curl info any of these files to see more information:

```
[root@c-csn1 tmp]# curl -IL c-csn1-test1.enfield.com/pants/plugins.txt -ucaringoadmin:caringo
HTTP/1.1 200 OK
Date: Wed, 22 Jul 2020 18:28:05 GMT
Gateway-Request-Id: ED6BE75CEE440295
Server: CAStor Cluster/11.2.0
Via: 1.1 c-csn1-test1.enfield.com (Cloud Gateway SCSP/6.4.0)
Gateway-Protocol: scsp
Castor-System-CID: 15a648db93dc29a6819bb256643915fc
Castor-System-Cluster: c-csn1.enfield.com
Castor-System-Created: Fri, 19 Jun 2020 21:31:53 GMT
Castor-System-Name: plugins.txt
Castor-System-Version: 1592602313.352
Content-Type: application/x-www-form-urlencoded
Last-Modified: Fri, 19 Jun 2020 21:31:53 GMT
X-Last-Modified-By-Meta: acepelon@
X-Owner-Meta: acepelon
ETag: "f877345eb91e9b72ad44d2a4480af33c"
Castor-System-Path: /c-csn1-test1.enfield.com/pants/plugins.txt
Castor-System-Domain: c-csn1-test1.enfield.com
Volume: 53a22d293eea60eb4bfaacc9933f12d6
Content-MD5: Li8xabfpx+wi+MMZFE3Uqg==
Content-Length: 616
```

I can see that I wrote this object recently, and there aren't many objects in this bucket. I am going to poke around more.

Searching unnamed objects

So, if that bucket doesn't contain a majority of the objects in my domain, what bucket does? Or, perhaps unnamed objects are the majority of my objects. We can search only for unnamed objects like so using the `-u` option:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -u -c

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 79 unique objects of stype: unnamed, withreps, uses 80.33KB disk sp
```

We might be tempted to use the -t option for "untenanted" objects, because untenanted objects are always unnamed, but these objects ARE tenanted (meaning, they live in a domain) but are also unnamed. Therefore, using -d [domain] -t will error.

Ok, we have 79 unnamed objects that live in c-csn1-test1.enfield.com. I want to get a few examples of these to show you what unnamed objects in a domain look like, but I don't want to output all 79 to stdout. I will use the `-u -1 -M 5` options to say "only send a single request for results (-1) and only return 5 items (-5) in that single request, and only return unnamed (-u) objects:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -u -1 -M 5

Starting to enumerate the requested streams in domain: c-csn1-test1.enfield.com

c-csn1-test1.enfield.com/7f7c9ecde7f265ac7dd4ba81e4388540
c-csn1-test1.enfield.com/f5b214774783d8bcc91ceae67c50a080
c-csn1-test1.enfield.com/e0896cec233e382c17840ae1c7d92054
c-csn1-test1.enfield.com/6fe0dbcdbf8bc538250f655dd152b5fd
c-csn1-test1.enfield.com/3122faaa7d02f9f7438702bf6bedb6ff


c-csn1-test1.enfield.com/ has 79 unique objects of stype: unnamed, withreps, uses 80.33KB disk sp
```

I can now curl any of these objects if I wanted to see their headers:

```
[root@c-csn1 tmp]# curl -IL c-csn1-test1.enfield.com/3122faaa7d02f9f7438702bf6bedb6ff -ucaringoad
HTTP/1.1 200 OK
Date: Wed, 22 Jul 2020 18:42:48 GMT
Gateway-Request-Id: FE7629C67527A767
Server: CAStor Cluster/11.2.0
Via: 1.1 c-csn1-test1.enfield.com (Cloud Gateway SCSP/6.4.0)
Gateway-Protocol: scsp
Castor-System-CID: 21876415934a554d1072804cfc776e10
Castor-System-Cluster: c-csn1.enfield.com
Castor-System-Created: Tue, 23 Jun 2020 15:07:45 GMT
Content-Type: application/x-www-form-urlencoded
Last-Modified: Tue, 23 Jun 2020 15:07:45 GMT
X-Last-Modified-By-Meta:
X-Owner-Meta:
x-bob-meta-apples: dunkin
ETag: "3122faaa7d02f9f7438702bf6bedb6ff"
Castor-System-Domain: c-csn1-test1.enfield.com
Volume: fa52b18e98d6164c5c0b700bba9652bb
Content-MD5: Ep4TEA3HwH8cOehCM1zZIQ==
Content-Length: 412
```

Searching metadata

Notice I have a metadata header called "x-bob-meta-apples" with value of "dunkin". That's interesting to me. I wonder if I have that metadata elsewhere in this domain:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 43 unique objects of stype: all, withreps, uses 3.42MB disk space.
```

The -m and -v options together show me that indeed I do have 43 matching objects. I wonder if I have any other objects that match the header but not necessarily that value. For this test, I simply remove the `-v dunkin` part of the command:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples  -c

Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 78 unique objects of stype: all, withreps, uses 3.46MB disk space.
```

Since I have more results here, I know that I have that header with a different value.

Searching across multiple domains

One of the more powerful things about the indexer-enumerator.sh is that I can search across all domains, not just one domain. Let's see how many objects matching that metadata header I have across my whole cluster. For this query, I change the domain name to "ALL" and I am just going to get a count match by using the -c option again:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d ALL -m x-bob-meta-apples -c

Enumerating all domains in the cluster:

Here are the domains:
test1.c-csn1.enfield.com
caringodrive.c-csn1.enfield.com
filefly-c-csn1.enfield.com
c-csn1-test1.enfield.com
c-csn1-admindomain
m-csn4.enfield.com
nfstest1.enfield.com
filefly-s3-target.c-csn1.enfield.com
es-backups.enfield.com
c-csn1.enfield.com
bob.is.great.com
s3-compatible
c-csn1-cfs1.enfield.com
c-csn1-s3-target.enfield.com

test1.c-csn1.enfield.com/ has 7 unique matching objects of stype: all, withreps, uses 7.32KB disk
caringodrive.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 di
filefly-c-csn1.enfield.com/ has 42 unique matching objects of stype: all, withreps, uses 41.67KB
c-csn1-test1.enfield.com/ has 78 unique matching objects of stype: all, withreps, uses 3.46MB dis
c-csn1-admindomain/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
m-csn4.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
nfstest1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
filefly-s3-target.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses
es-backups.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
bob.is.great.com/ has 6 unique matching objects of stype: all, withreps, uses 10.81MB disk space
s3-compatible/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-cfs1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-s3-target.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk


Only streams counts are listed.  To get the streams themselves, remove the -c flag.
All domains: 133 unique matching objects of stype: all, withreps, uses 14.32MB disk space
```

That shows me 4 different domains (although it doesn't show me untenanted objects that may match) have objects with that metadata. I can then narrow the search down to match that particular header value "dunkin":

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d ALL -m x-bob-meta-apples -v dunkin -c

Enumerating all domains in the cluster:

Here are the domains:
test1.c-csn1.enfield.com
caringodrive.c-csn1.enfield.com
filefly-c-csn1.enfield.com
c-csn1-test1.enfield.com
c-csn1-admindomain
m-csn4.enfield.com
nfstest1.enfield.com
filefly-s3-target.c-csn1.enfield.com
es-backups.enfield.com
c-csn1.enfield.com
bob.is.great.com
s3-compatible
c-csn1-cfs1.enfield.com
c-csn1-s3-target.enfield.com

test1.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk spac
caringodrive.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 di
filefly-c-csn1.enfield.com/ has 14 unique matching objects of stype: all, withreps, uses 14.64KB
c-csn1-test1.enfield.com/ has 43 unique matching objects of stype: all, withreps, uses 3.42MB dis
c-csn1-admindomain/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
m-csn4.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
nfstest1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
filefly-s3-target.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses
es-backups.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
bob.is.great.com/ has 3 unique matching objects of stype: all, withreps, uses 5.40MB disk space
s3-compatible/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-cfs1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-s3-target.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk


Only streams counts are listed.  To get the streams themselves, remove the -c flag.
All domains: 60 unique matching objects of stype: all, withreps, uses 8.84MB disk space
```

73 fewer objects. Let's try a different header value:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d ALL -m x-bob-meta-apples -v donuts -c

Enumerating all domains in the cluster:

Here are the domains:
test1.c-csn1.enfield.com
caringodrive.c-csn1.enfield.com
filefly-c-csn1.enfield.com
c-csn1-test1.enfield.com
c-csn1-admindomain
m-csn4.enfield.com
nfstest1.enfield.com
filefly-s3-target.c-csn1.enfield.com
es-backups.enfield.com
c-csn1.enfield.com
bob.is.great.com
s3-compatible
c-csn1-cfs1.enfield.com
c-csn1-s3-target.enfield.com

test1.c-csn1.enfield.com/ has 7 unique matching objects of stype: all, withreps, uses 7.32KB disk
caringodrive.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 di
filefly-c-csn1.enfield.com/ has 28 unique matching objects of stype: all, withreps, uses 27.02KB
c-csn1-test1.enfield.com/ has 35 unique matching objects of stype: all, withreps, uses 36.62KB di
c-csn1-admindomain/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
m-csn4.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
nfstest1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
filefly-s3-target.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses
es-backups.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
bob.is.great.com/ has 3 unique matching objects of stype: all, withreps, uses 5.40MB disk space
s3-compatible/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-cfs1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-s3-target.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk


Only streams counts are listed.  To get the streams themselves, remove the -c flag.
All domains: 73 unique matching objects of stype: all, withreps, uses 5.47MB disk space
```

Ah! This shows me that all of the objects matching that header have a value of either “dunkin” or “donuts”.

Searching by age

What if I was only interested in objects written long ago? Maybe I want to find all objects written x days ago so that I can delete them…

Let’s get a single object from the matching output above and then do a curl INFO.

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Starting to enumerate the requested streams in domain: c-csn1-test1.enfield.com

c-csn1-test1.enfield.com/e0896cec233e382c17840ae1c7d92054


c-csn1-test1.enfield.com/ has 43 unique objects of stype: all, withreps, uses 3.42MB disk space.
```

```
[root@c-csn1 tmp]# curl -IL c-csn1-test1.enfield.com/e0896cec233e382c17840ae1c7d92054 -ucaringoad
HTTP/1.1 200 OK
Date: Wed, 22 Jul 2020 18:57:27 GMT
Gateway-Request-Id: 58E8B3631B9490E0
Server: CAStor Cluster/11.2.0
Via: 1.1 c-csn1-test1.enfield.com (Cloud Gateway SCSP/6.4.0)
Gateway-Protocol: scsp
Castor-System-CID: 21876415934a554d1072804cfc776e10
Castor-System-Cluster: c-csn1.enfield.com
Castor-System-Created: Tue, 23 Jun 2020 15:07:45 GMT
Content-Type: application/x-www-form-urlencoded
Last-Modified: Tue, 23 Jun 2020 15:07:45 GMT
X-Last-Modified-By-Meta:
X-Owner-Meta:
x-bob-meta-apples: dunkin
ETag: "e0896cec233e382c17840ae1c7d92054"
Castor-System-Domain: c-csn1-test1.enfield.com
Volume: fa52b18e98d6164c5c0b700bba9652bb
Content-MD5: 6AspDUv0/7hEBsMFALI5Ig==
Content-Length: 858
```

I can see that it was written on June 23 of this year. Were ALL of the objects written this year matching that header written this year? We can check by using the `-G 1` and `-g 1` options:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only enumerating streams written since 1 year(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 43 unique objects of stype: all, withreps, uses 3.42MB disk space.

[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only enumerating streams written at least 1 year(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 0 unique objects of stype: all, withreps, uses 0 disk space.
```

Yes, they were all written this year. Since the object example we had was written on June 23 (today is July 22), I can do some further narrowing down based on my example. June 23 was 29 days ago from when I am running these examples:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only enumerating streams written at least 29 day(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 14 unique objects of stype: all, withreps, uses 14.64KB disk space.
```

14 objects matched that, and our test object in particular you can see matches as expected:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v
c-csn1-test1.enfield.com/e0896cec233e382c17840ae1c7d92054
```

But only 14 of 43 objects were written at least 29 days ago. Were any written more than 30 days ago?

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only enumerating streams written at least 30 day(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 0 unique objects of stype: all, withreps, uses 0 disk space.
```

Nope.

I can further winnow my results as desired.

Let's back out a little bit and add another option.

Search by size

How about if we were looking for small files with that same metadata. Let's try to match objects about the same size as our example above - which was 858 bytes. To that end, I will add the `-l 859` (l for "littler") option to our query.

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Only streams smaller than 859 bytes are listed.
Only enumerating streams written at least 29 day(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.

c-csn1-test1.enfield.com/ has 12 unique objects of stype: all, withreps, uses 10.11KB disk space.
```

Ok, 12 of the 14 objects were smaller than 858 bytes. Nice to know. I want to verify that my example object is in that result set as a sanity check. The UUID started with e, so let's use yet another option- the prefix match. I will add `-p e` to match any object starting with "p" in its name/UUID. I will remove the `-c` option so that I am actually seeing the match:

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d c-csn1-test1.enfield.com -m x-bob-meta-apples -v

Starting to enumerate the requested streams in domain: c-csn1-test1.enfield.com

c-csn1-test1.enfield.com/e0896cec233e382c17840ae1c7d92054

Only streams smaller than 859 bytes are listed.
Only streams with names (or UUIDs) starting with "e" are listed.
Only enumerating streams written at least 29 day(s) ago

c-csn1-test1.enfield.com/ has 1 unique objects of stype: all, withreps, uses 1.67KB disk space.
[root@c-csn1 tmp]#
```

Sure enough, there's our object!

Now, how about if I want to match all objects *larger* than that object across all domains, matching that same header, written more than 29 days ago. I will use the capital L option and change the domain to "ALL":

```
[root@c-csn1 tmp]# indexer-enumerator.sh -ro -d ALL -m x-bob-meta-apples -v dunkin -f 29 -L 859

Enumerating all domains in the cluster:

Here are the domains:
test1.c-csn1.enfield.com
caringodrive.c-csn1.enfield.com
filefly-c-csn1.enfield.com
c-csn1-test1.enfield.com
c-csn1-admindomain
m-csn4.enfield.com
nfstest1.enfield.com
filefly-s3-target.c-csn1.enfield.com
es-backups.enfield.com
c-csn1.enfield.com
bob.is.great.com
s3-compatible
c-csn1-cfs1.enfield.com
c-csn1-s3-target.enfield.com

test1.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk spac
caringodrive.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 di
filefly-c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk sp
c-csn1-test1.enfield.com/ has 2 unique matching objects of stype: all, withreps, uses 4.53KB disk
c-csn1-admindomain/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
m-csn4.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
nfstest1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
filefly-s3-target.c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses
es-backups.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
bob.is.great.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
s3-compatible/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-cfs1.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk space
c-csn1-s3-target.enfield.com/ has 0 unique matching objects of stype: all, withreps, uses 0 disk


Only streams larger than 859 bytes are listed.
Only enumerating streams written at least 29 day(s) ago
Only streams counts are listed.  To get the streams themselves, remove the -c flag.
All domains: 2 unique matching objects of stype: all, withreps, uses 4.53KB disk space
```

I can see only 2 objects match. I can remove the -c option and get those results if I wanted.

Hopefully the above gives you a good understanding of how the indexer-enumerator.sh script works and the power of its flexibility.

Related articles

- How to use indexer-enumerator.sh
- Where is my metering data?
- Install head tool for elasticsearch5 in a bucket
- Using Monit to monitor Elasticsearch and Content Gateway

caringo.com	6801 N. Capital of Texas Hwy Bldg 2 Ste. 200 Austin, Texas 78731 USA +1 512.782.4490
connect.caringo.com	Log on to download Caringo software and product documentation and to view training videos and online help.
support.caringo.com	Log on to enter support tickets and to search the knowledge base, which includes the latest documentation, FAQs, technical notes, product advisories, and troubleshooting.