MINIO

Leveraging MinIO for Splunk SmartStore S3 Storage

SEPTEMBER 2019

OVERVIEW

MinIO is a high performance, Amazon S3 compatible, distributed object storage system. By following the methods and design philosophy of hyperscale computing providers, MinIO delivers high performance and scalability to a wide variety of workloads in the private cloud. Because MinIO is purpose-built to serve only objects, a single-layer architecture achieves all of the necessary functionality without compromise. The advantage of this design is an object server that is simultaneously performant and lightweight.

Splunk is a high performance event processing platform for enterprise computing environments that provides critical and timely insight into IT operations, including data from IoT, firewalls, web servers and more. SmartStore feature enables Splunk to offload Petabytes of data to an external Amazon S3 compatible object storage. Disaggregating compute and storage frees Splunk nodes to focus on indexing and search, while the object storage is free to focus on the management, resilience and security of the data.

This whitepaper describes integration and performance testing between Splunk's SmartStore functionality and MinIO object storage. The results show that the combination of MinIO and Splunk SmartStore provide a high performance, on premise S3 store that allows for complete control over an enterprise's event data. With the combination of Splunk's efficiency in search and compression, and MinIO's ability to quickly store and retrieve data, the results of the "worst case scenario" (searching against terabytes of events across 10 servers) is returned in seconds.

AUDIENCE

This paper is intended for IT and Security professionals who have experience in setting up Splunk and basic understanding of MinIO. This paper assumes high level understanding of the technologies described.

DEFINITION OF TERMS

In this document the following terms are used. They are specific to either Splunk, MinIO, or object storage as a whole.

S3 - Simple storage service, a cloud based object storage system from Amazon. MinIO is a drop in replacement for Amazon S3 for Splunk's SmartStore.

Indexer - A Splunk node dedicated to collating events into actionable data.

Indexer cluster - A group of Splunk nodes also referred to as Peer nodes that, working in concert, provide a redundant indexing and searching capability.

Cluster Master - A Splunk node dedicated for the purpose of managing Splunk clusters.

Search Head - A Splunk node used to query multiple indexers at once. In this document, a single search head is used to query all indexers in a cluster simultaneously.

Bucket - In the context of Splunk, a bucket represents a folder comprised of a time defined collection of events. In the context of MinIO, a bucket represents a logical separation of data. When Splunk creates buckets in MinIO, it uses the same naming convention as used to create buckets on the indexer nodes, providing a one to one mapping of data from the filesystem to the backend object store.

Erasure code - a mathematical algorithm to reconstruct missing or corrupted data. This provides the data resiliency in a smaller footprint than normally required from data replication.

ENVIRONMENT

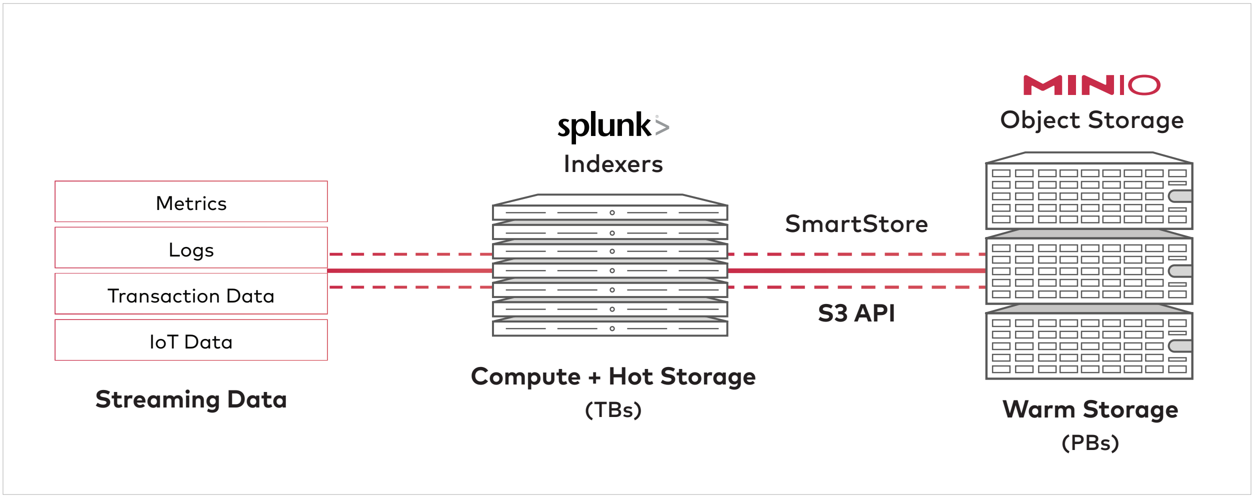
Physical instances were deployed via Packet.net. For Splunk indexers and MinIO nodes, m2-xlarge instances were deployed providing 28 cores per server, 384 GB of RAM and 4TB of usable disk space. As shown later in the document, MinIO only requires moderate amounts of RAM and CPU. However, these instances provided fast storage via NVMe drives with a theoretical maximum of 2.6GB/s bandwidth. Network throughput as measured by iperf is ~ 20Gb/s. Disks were formatted with XFS using default values.

Both the MinIO cluster and the Splunk indexer cluster had 10 nodes each.

The Splunk Search Head and Cluster Master were both provisioned with smaller c2.medium instances as they do not have the same requirements as the clustered nodes, but still have the important 20Gb/s network bandwidth to ensure the search head is not bottlenecked on network.

All nodes are provisioned with Ubuntu 18.04.2 LTS as the OS. Splunk nodes were installed with Splunk Enterprise version 7.3.0. The Splunk index cluster was configured with a replication factor of 3 and a search factor of 2.

Blank	Instance Types	# Nodes	Replication
MinIO Nodes	m2.xlarge: - 28 cores - 384 GB RAM - 4 TB total disk	10	Erasure Code : 2
Splunk Indexers		10	3x Replication 2x Search
Splunk Search Head	c2.medium: -20 GB/s NIC		
Splunk Cluster Master			



CONFIGURING MINIO

MinIO was configured as a systemd service in a ten node cluster, where each node in the MinIO cluster had a single, dedicated disk. The erasure code ratio was set to EC:2, allowing for up to two disks to be lost and still retain data. This was chosen to accommodate having sufficient storage for testing. Based on the performance results listed in this paper, having a higher erasure count would only have had a negligible effect.

```
MINIO_VOLUMES="http://10.88.126.{21...30}:9000/opt/minio"
MINIO_STORAGE_CLASS_STANDARD="EC:2”
# Access Key of the server.
MINIO_ACCESS_KEY=<your access key>
# Secret key of the server.
MINIO_SECRET_KEY=<your secret key>
```

CONFIGURING SMART STORE

SmartStore configuration is simple and requires just a few additional lines to the existing indexes.conf file:

```
[volume:s3]
storageType = remote
path = s3://smartstore/remote_volume
remote.s3.access_key = minio
remote.s3.secret_key = minio123
remote.s3.supports_versioning = false
remote.s3.endpoint = http://miniocluster:9000
```

The path parameter locates the bucket that will hold the data created in this test. On the MinIO service, a bucket named "smartstore" was created for this purpose. It is mandatory to create the bucket configured for SmartStore before applying the new indexes.conf settings.

The remote.s3.access_key and remote.s3.secret_key are configured as the MinIO instance keys in the section above. Since MinIO does not support versioning, the remote.s3.supports_versioning parameter is set to false. The remote.s3.endpoint locates the the MinIO cluster where the smartstore bucket is created.

For distributed environments, it is recommended to have the endpoint configured to the round robin DNS entry for the cluster. For this test, rrDNS was simulated with an /etc/hosts entry on each indexer pointing to a single MinIO node, resulting in a 1:1 mapping for client (Splunk Indexer) and server (MinIO node).

From the Splunk cluster master, add these changes to $SPLUNK_PATH/etc/master-apps/_cluster/local/indexes.conf, for example, /opt/splunk/etc/master-apps/_cluster/local./indexes.conf. After modifying the file, changes can be pushed to all indexer nodes:

```
root@splunk-cluster-master:~# splunk apply cluster-bundle --answer-yes
Created new bundle with checksum=4BD58C40847DC1F5173BD8FDED6903F3
Applying new bundle. The peers may restart depending on the configurations in
applied bundle.
Please run 'splunk show cluster-bundle-status' for checking the status of the
applied bundle.
```

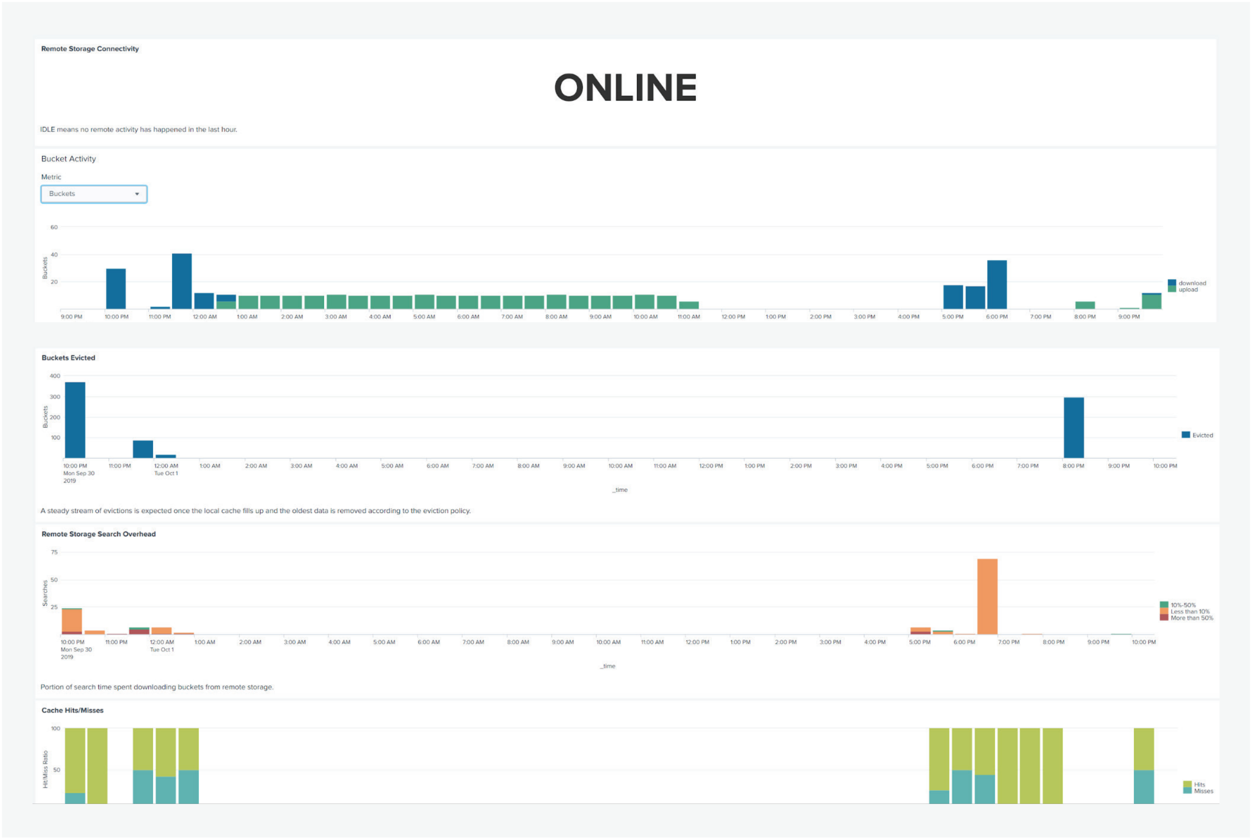
After configuring MinIO in Splunk, Splunk internal buckets (such as _audit and _telemetry) are automatically created. This can be checked via the MinIO client (mc):

```
mc ls miniocluster/smartstore/remote_volume/
[2019-10-01 11:01:43 PDT]      0B _audit/
[2019-10-01 11:01:43 PDT]      0B _internal/
[2019-10-01 11:01:43 PDT]      0B _introspection/
[2019-10-01 11:01:43 PDT]      0B _telemetry/
```

After the environment has reached the threshold of rolling hot buckets, index buckets will be rolled to MinIO as well:

```
mc ls miniocluster/smartstore/remote_volume/
[2019-10-01 14:06:13 PDT]      0B _audit/
[2019-10-01 14:06:13 PDT]      0B _internal/
[2019-10-01 14:06:13 PDT]      0B _introspection/
[2019-10-01 14:06:13 PDT]      0B _telemetry/
[2019-10-01 14:06:13 PDT]      0B minio_index/
```

Additionally, activity can be observed from the Splunk Monitoring console under Indexing -> SmartStore -> Activity and Cache Performance



IDLE means no remote activity has happened in the last hour.

A steady stream of evictions is expected once the local cache fills up and the oldest data is removed according to the eviction policy.

Portion of search time spent downloading buckets from remote storage.

LOADING DATA

Loading data was achieved using the gogen event generating utility to load data into Splunk. Initial load was 100GB per day per node for the previous ten days:

```
root@splunk-cluster-master:~# ./gogen -c examples/weblog/weblog_da-
ta.yml  -o file -f /opt/splunk/generated_data/minio_index/min-
io_events.log -g 12 gen -c 3551 -b="-10d"
```

This generated a total of 10TB of data across all nodes for a total of ~ 3 billion events per day per node. The generated data is of weblog format. Sample events have the format:

```
27.35.11.11 - - [18/Sep/2019:23:59:59 +0000] "GET /prod-
uct.screen?product_id=HolyGouda&JSESSIONID=SD3SL1FF7ADFF8 HTTP/1.1"
200 2243 "http://shop.buttercupgames.com/cart.do?ac-
tion=view&itemId=HolyGouda" "Mozilla/5.0 (iPhone; U; CPU iPhone OS
5_0_1 like Mac OS X; en_US) AppleWebKit (KHTML, like Gecko) Mobile
[FBAN/FBForIPhone;FBAV/4.0.2;FBBV/4020.0;FBDV/iPhone3,1;FBM-
D/iPhone;FBSN/iPhone OS;FBSV/5.0.1;FBSS/2; FBCR/AT&T;FBID/phone;F-
BLC/en_US;FBSF/2.0]"
```

The MinIO command line client `mc` has an inbuilt function called 'trace' that allows viewing of all S3 requests that come to the server:

```
root@splunk-cluster-master:~# mc admin trace miniocluster
...
21:27:21.309 [200 OK] s3.PutObject miniocluster:9000/smartstore/re-
mote_volume/minio_index/d-
b/21/70/482~FF906DA0-5A09-4A5F-ABDF-3F64BFF58C64/receipt.json
20.39ms      ↑ 1.8 KiB  ↓ 261 B

21:27:22.726 [200 OK] s3.NewMultipartUpload miniocluster:9000/smart-
store/remote_volume/minio_index/d-
b/86/cd/486~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6
AB-4579-857E-790ECB03D107/rawdata/journal.gz?uploads      1.031492s
↑ 69 B  ↓ 637 B

21:27:24.822 [200 OK] s3.PutObjectPart miniocluster:9000/smartstore/re-
mote_volume/minio_index/d-
b/86/cd/486~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6
AB-4579-857E-790ECB03D107/rawdata/journal.gz?partNumber=3&uploadId=aa0d
8aeb-1caf-46b0-8843-d1c65a773270      573.302ms    ↑ 79 MiB  ↓ 261 B
21:27:24.822 [200 OK] s3.PutObjectPart miniocluster:9000/smartstore/re-
mote_volume/minio_index/d-
b/86/cd/486~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6
AB-4579-857E-790ECB03D107/rawdata/journal.gz?partNumber=2&uploadId=aa0d
8aeb-1caf-46b0-8843-d1c65a773270      861.608ms    ↑ 128 MiB  ↓ 261 B

21:27:24.822 [200 OK] s3.PutObjectPart miniocluster:9000/smartstore/re-
mote_volume/minio_index/d-
b/86/cd/486~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6
AB-4579-857E-790ECB03D107/rawdata/journal.gz?partNumber=1&uploadId=aa0d
8aeb-1caf-46b0-8843-d1c65a773270     1.287407s    ↑ 128 MiB  ↓ 261 B
...
```

To see only API requests, run:

```
mc admin trace --json miniocluster  | jq '{path, "API": .api}'
```

```
root@splunk-indexer01:~# mc admin trace miniocluster/ --json  | jq '{path,
"API": .api}'
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/journal.gz",
  "API": "s3.NewMultiPartUpload"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/journal.gz",
  "API": "s3.PutObjectPart"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/journal.gz",
  "API": "s3.PutObjectPart"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/journal.gz",
  "API": "s3.PutObjectPart"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/journal.gz",
  "API": "s3.CompleteMultiPartUpload"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/slicesv2.dat",
  "API": "s3.PutObjectPart"
}
{
  "path": "/smartstore/remote_volume/minio_index/d-
b/62/89/507~7753C048-D6AB-4579-857E-790ECB03D107/guidSplunk-7753C048-D6AB-4
579-857E-790ECB03D107/rawdata/slicesmin.dat",
  "API": "s3.PutObjectPart"
}
```

Finally, for detailed information, use the -v switch for verbose mode

```
root@splunk-indexer01:~# mc admin trace -v miniocluster
10.88.126.47 [REQUEST s3.GetObject] 22:58:18.809
10.88.126.47 POST /smartstore/remote_volume/minio_index/d-
b/01/4a/2~356932B6-38BC-4A6D-877C-A767DB61560A/guidSplunk-356932B6-38BC-4A6
D-877C-A767DB61560A/1568660462-1568324333-12826708760433608263.tsidx
10.88.126.47 Host: miniocluster:9000
10.88.126.47 Authorization: AWS4-HMAC-SHA256 Credential=min-
io/20191008//s3/aws4_request, SignedHeaders=host;range;x-amz-con-
tent-sha256;x-amz-date, Signa-
ture=c8fb025e3958d72e00b5872dbb08f9196be1c11c71196a25cc8454accc02ba96
10.88.126.47 Content-Length: 0
10.88.126.47 Range: bytes=134217728-268435455
10.88.126.47 Te: trailers, chunked
10.88.126.47 X-Amz-Content-Sha256: UNSIGNED-PAYLOAD
10.88.126.47 X-Amz-Date: 20191008T225818Z
10.88.126.47 <BODY>
10.88.126.47 [RESPONSE] [22:58:19.380] [ Duration 571.316ms  ↑ 75 B  ↓ 128
MiB ]
10.88.126.47 206 Partial Content
```

In addition to the regular events, one special event per minute was inserted with the search term "manticore":

```
[28/Sep/2019:23:59:21 +0000] "GET /product.screen?product_id=Holy-
Gouda&JSESSIONID=SD3SL1FF7ADFF8 HTTP/1.1" 404 1661 "http://shop.but-
tercupgames.com/cart.do?action=view&itemId=manticore" "Mozilla/5.0
(iPad; U; CPU OS 3_2 like Mac OS X; en-us)
```

This enables targeted searches that will scan multiple buckets.

When loading event via gogen, the system load on the indexers is minimal (load average of 3.66, 1.46, 0.58) as evidenced below:

```
top - 20:04:48 up 5 days, 2:44, 2 users, load average:3.66, 1.46, 0.58
Tasks: 663 total, 2 running, 343 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5.8 us, 2.8 sy, 0.0 ni, 91.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 39491139+total, 10764680 free, 2950712 used, 38119600+buff/cache
KiB Swap : 1996796 total,  1742332 free,  254464 used. 38657564+avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  34544 root      20   0 1638960 200768 39876 S 237.0  0.1   4571:01 splunkd
 996885 root      20   0 4041096  27248 10112 S 197.0  0.0   3:33.97 gogen
 997929 root      20   0  197496  19396  2052 R  65.0  0.0   0:01.97 splunk-optimize
 996992 root      20   0       0      0     0 I   1.0  0.0   0:00.13 kworker/u113:3
  34562 root      20   0 1648644  31184 18184 S   0.7  0.0  18:32.50 mongod
```

In addition, `mc` has the ability to query the cluster for resource usage:

```
root@splunk-indexer01:~# mc admin info server miniocluster
●  10.88.126.17:9000
    Uptime: 3 days
   Version: 2019-09-25T18:25:51Z
   Storage: Used 4.5 TiB, Free 30 TiB
    Drives: 1/1 OK

    CPU         min        avg       max
    current     0.15%      0.17%     0.19%
    historic    0.02%      1.55%     1286.78%

    MEM         usage
    current     1.6 GiB
    historic    961 GiB


●  10.88.126.23:9000
    Uptime: 4 days
   Version: 2019-09-26T19:42:35Z
   Storage: Used 4.5 TiB, Free 30 TiB
    Drives: 1/1 OK

    CPU          min        avg       max
    current      0.09%      0.12%     0.16%
    historic     0.01%      1.81%     1542.97%

    MEM         usage
    current     2.3 GiB
    historic    857 GiB
```

From this snippet, it is apparent that various servers are utilizing very low amounts of RAM and CPU.

In all cases, the systems were never under more than nominal strain for resources.

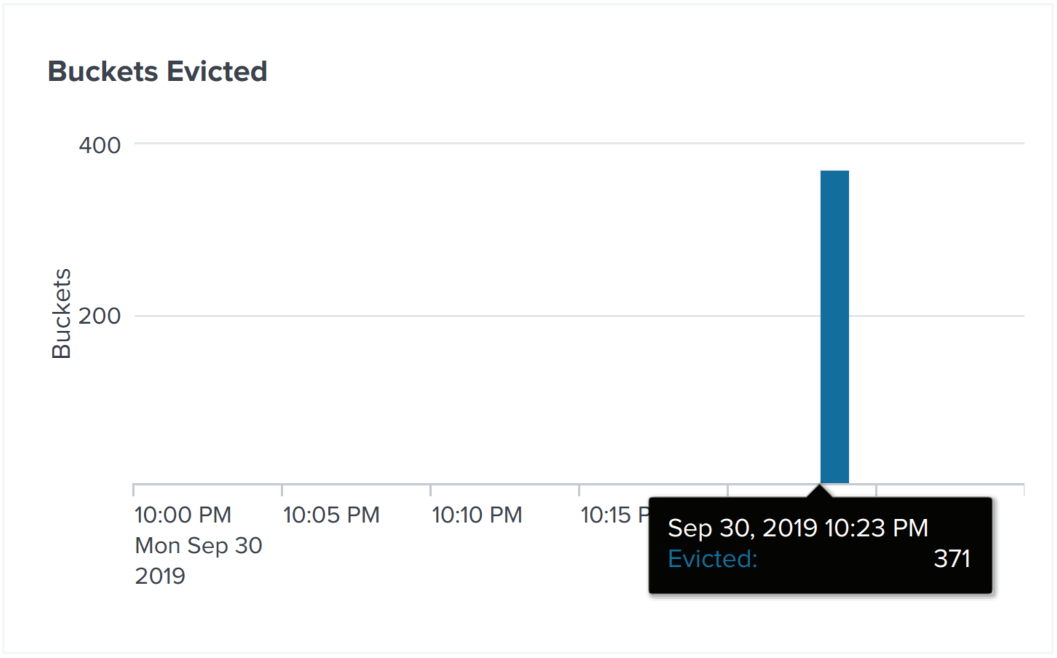
SEARCH METRICS

Using the targeted search term from before, queries were run against several days of back dated data.

Before the search is run, the Splunk indexer cache was cleared. This represents a "worst case scenario", in which SmartStore is forced to download all buckets before being able to return a search result.

```
curl -ku admin:changeme "https://localhost:8089/services/admin/-
cacheman/_evict" -d path=/opt/splunk -d mb=99999999999
```

Confirmation of buckets eviction is performed via the Splunk Monitoring Console:



From the search head, the following query is issued:

```
time splunk search 'index=minio_index manticore' —maxout 0  |wc —l
```

Directly after cache is flushed, the following is seen:

```
root@splunk-search-head:~# time splunk search ‘index=minio_index manticore’
—maxout 0 |wc —l 172800

real   0m39.925s
user   0m8.821s
sys    0m2.538s
```

By monitoring the output of `mc` admin trace, a flurry of GET requests were seen, indicating that SmartStore is actively downloading buckets and objects to search. Since the search term occurs every minute across a time span of ten days data was loaded, it ensures a large amount of buckets must be downloaded. With the combination of Splunk's efficiency in search and compression, and MinIO's ability to quickly store and retrieve data, the results of the "worst case scenario" (searching against terabytes of events across 10 servers) is returned in seconds.

In a real world scenario, Splunk queries are typically targeted to a limited number of events to prevent situations such as detailed above. To simulate this behavior, a unique record, identified by the term “unicorn” was created. After clearing the cache, Splunk needs only to find and download the bucket which holds the event:

```
root@splunk-search-head:~# time splunk search ‘index=minio_index unicorn’
-maxout 0 |wc -l 1

real    0m13.052s
user    0m0.292s
sys     0m0.079s
```

After the event has been returned to cache, the search is nearly instantaneous:

```
root@splunk-search-head:~# time splunk search ‘index=minio_index unicorn’
-maxout 0 |wc -l 1

real    0m0.926s
user    0m0.288s
sys     0m0.084s
```

CONCLUSION

MinIO and Splunk SmartStore are exceptionally well suited for each other in deployment scenarios that emphasize performance at scale. MinIO’s inherent simplicity and scaling properties allow for Petabyte plus deployments to be managed via Splunk - providing significant cost savings versus AWS while retaining full control and security of the data assets.