SUSE Enterprise Storage 5.5
Object Storage Metadata Sync Module Configuration
Elasticsearch, Fluentbit, Kibana (EFK Stack)

Setup of EFK Stack
Install SLES 15 SP1

Install Elasticsearch
 - Download the rpm version (7.0.1) from
https://www.elastic.co/downloads/elasticsearch

```
zypper in elasticsearch-7.0.1-x86_64.rpm
```

Test the service is alive and responding curl http://localhost:9200/

Response
```
# curl http://localhost:9200
{
 "name" : "efk-server",
 "cluster_name" : "elasticsearch",
 "cluster_uuid" : "btjzcngDTuuNSyVNuPtfbA",
 "version" : {
   "number" : "7.0.1",
   "build_flavor" : "default",
   "build_type" : "rpm",
   "build_hash" : "e4efcb5",
   "build_date" : "2019-04-29T12:56:03.145736Z",
   "build_snapshot" : false,
   "lucene_version" : "8.0.0",
   "minimum_wire_compatibility_version" : "6.7.0",
   "minimum_index_compatibility_version" : "6.0.0-beta1"
 },
 "tagline" : "You Know, for Search"
}
```

Fluentbit Installation:

```
# vi td-agent-bit.repo

# cat td-agent-bit.repo
[td-agent-bit]
name = TD Agent Bit
baseurl = http://packages.fluentbit.io/centos/7
gpgcheck=1
gpgkey=http://packages.fluentbit.io/fluentbit.key
enabled=1

# zypper ref td-agent-bit
Retrieving repository 'TD Agent Bit' metadata
---------------------------------------------------------------------------------
-------------------------[|]
Warning: File 'repomd.xml' from repository 'TD Agent Bit' is unsigned.

   Note: Signing data enables the recipient to verify that no modifications
occurred after the data
   were signed. Accepting data with no, wrong or unknown signature can lead to a
corrupted system
   and in extreme cases even to a system compromise.
   Note: File 'repomd.xml' is the repositories master index file. It ensures the
integrity of the
   whole repo.

   Warning: We can't verify that no one meddled with this file, so it might not be
trustworthy
   anymore! You should not continue unless you know it's safe.

File 'repomd.xml' from repository 'TD Agent Bit' is unsigned, continue? [yes/no]
(no): yes
Retrieving repository 'TD Agent Bit'
metadata ...........................................................................
..............................[done]
Building repository 'TD Agent Bit'
cache ..............................................................................
................................[done]
Specified repositories have been refreshed.


# zypper in td-agent-bit
Refreshing service 'Basesystem_Module_x86_64'.
Refreshing service 'SUSE_Linux_Enterprise_Server_x86_64'.
Refreshing service 'Server_Applications_Module_x86_64'.
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
 td-agent-bit

The following package has no support information from its vendor:
 td-agent-bit

1 new package to install.
Overall download size: 5.4 MiB. Already cached: 0 B. After the operation,
additional 20.7 MiB will be used.
Continue? [y/n/v/...? shows all options] (y): y
Retrieving package td-agent-bit-1.1.0-1.x86_64
(1/1),   5.4 MiB ( 20.7 MiB unpacked)
Retrieving: td-agent-bit-1.1.0-
1.x86_64.rpm .......................................................................
........................[done (2.8 MiB/s)]
td-agent-bit-1.1.0-1.x86_64.rpm:
   Header V4 RSA/SHA256 Signature, key ID 6ea0722a: NOKEY

Looking for gpg key ID 6EA0722A in cache /var/cache/zypp/pubkeys.
Looking for gpg key ID 6EA0722A in repository TD Agent Bit.
 gpgkey=http://packages.fluentbit.io/fluentbit.key
Retrieving:
fluentbit.key ......................................................................
........................................................[done]

New repository or package signing key received:

 Repository:       TD Agent Bit
 Key Name:         Eduardo Silva <eduardo@treasure-data.com>
 Key Fingerprint:  F209D876 2A60CD49 E680633B 4FF8368B 6EA0722A
 Key Created:      Tue 05 Jul 2016 09:58:07 PM MDT
 Key Expires:      (does not expire)
 Subkey:          AE8602F007A243C2 2015-08-31 [does not expire]
 Rpm Name:        gpg-pubkey-6ea0722a-577c81cf


Do you want to reject the key, or trust always? [r/a/?] (r): a
Checking for file
conflicts: ...................................................................
..................................................[done]
(1/1) Installing: td-agent-bit-1.1.0-
1.x86_64 .....................................................................
...............................[done]
```

Kibana Installation

```
# zypper in kibana-7.0.0-x86_64.rpm
Refreshing service 'Basesystem_Module_x86_64'.
Refreshing service 'SUSE_Linux_Enterprise_Server_x86_64'.
Refreshing service 'Server_Applications_Module_x86_64'.
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
 kibana

The following package has no support information from its vendor:
 kibana

1 new package to install.
Overall download size: 165.0 MiB. Already cached: 0 B. After the operation,
additional 397.2 MiB will be used.
Continue? [y/n/v/...? shows all options] (y): y
Retrieving package kibana-7.0.0-1.x86_64
(1/1), 165.0 MiB (397.2 MiB unpacked)
kibana-7.0.0-x86_64.rpm:
   Header V4 RSA/SHA512 Signature, key ID d88e42b4: NOKEY

Looking for gpg key ID D88E42B4 in cache /var/cache/zypp/pubkeys.
Repository Plain RPM files cache does not define additional 'gpgkey=' URLs.
kibana-7.0.0-1.x86_64 (Plain RPM files cache): Signature verification failed [4-
Signatures public key is not available]
Abort, retry, ignore? [a/r/i] (a): i
Checking for file
conflicts: ...................................................................
..................................................[done]
(1/1) Installing: kibana-7.0.0-
1.x86_64 .....................................................................
.....................................[done]
Additional rpm output:
warning: /var/cache/zypper/RPMS/kibana-7.0.0-x86_64.rpm: Header V4 RSA/SHA512
Signature, key ID d88e42b4: NOKEY
```

Ceph radosgw Configuration

Configure Deepsea to deploy custom RGW roles
First we will walk through the process of teaching Deepsea to deploy a custom RGW role. For this case it will be for a separate zone which will be used for sending its metadata to elasticsearch. This will also allow us to have two RGW gateways running on the same node with different ports.

Since we are using Deepsea it automatically sets up a default system user with default zone and zonegroup that its tied to. We will let this stay intact and not modify it, but will create our own sets of system users, realm, zones, and zonegroup so that we can set things up for elasticsearch. The future will see much improvement in this area around Deepsea and ceph manager.

Modify Ceph configuration files
In this step we will create the two new radosgw services that can be referenced on the node we will be running it.
One will be for the elasticsearch called zone us-west-2 and the other will be zone us-west-1.

Change your directory to /srv/salt/ceph/configuration/files/

```
$ cd /srv/salt/ceph/configuration/files/
```

Copy the rgw.conf file to a new file for your elasticsearch role

```
$ cp rgw.conf rgw-elasticsearch.conf
$ cp rgw.conf rgw-us-west-1.conf
```

Open the file rgw-us-west-1.conf and change the values for the port and assigned zone. The results should be like the output below. Save the file and quit. The “rgw zone” parameter that has been added will be the zone that we create at a later step.

```
$ cat rgw-us-west-1.conf

[client.{{ client }}]
rgw zone = "us-west-1"
rgw frontends = "civetweb port=8000"
rgw dns name = {{ fqdn }}
rgw enable usage log = true
```

Open the file rgw-elasticsearch.conf and change the values for the port and assigned zone. The results should be like the output below. Save the file and quit. The “rgw zone” parameter that has been added will be the zone that we create at a later step that is used for sending metadata to elasticsearch.

```
$ cat rgw-elasticsearch.conf

[client.{{ client }}]
rgw zone = "us-west-2"
rgw frontends = "civetweb port=8002"
rgw dns name = {{ fqdn }}
rgw enable usage log = true
```

Change your directory to /srv/salt/ceph/rgw/files/

```
$ cd /srv/salt/ceph/rgw/files/
```

Copy the rgw.j2 file to your new role file as rgw-elasticsearch.j2

```
$ cp rgw.j2 rgw-elasticsearch.j2
$ cp rgw.j2 rgw-us-west-1.j2
```

Nothing needs to change in these files. Its all ready.

global.yml changes
open the file /srv/pillar/ceph/stack/global.yml and add the following. Your email, name, or uid might be different. Once your finished then save the file and quit.

```
rgw_configurations:
 rgw:
   users:
     - { uid: "admin", name: "Admin", email: "admin@susedojo.com", system: True }
 # when using only RGW& not ganesha ssl will have all the users of rgw already,
 # but to be consistent we define atleast one user
 rgw-elasticsearch:
   users:
     - { uid: "zone.user", name: "Zone Admin", email: "zoneadmin@susedojo.com",
system: True }
 rgw-us-west-1:
   users:
     - { uid: "zone.user", name: "Zone Admin", email: "zoneadmin@susedojo.com",
system: True }
```

Update policy.cfg
Now we can add the new role to the configuration proposal.

Open the file /srv/pillar/ceph/proposals/policy.cfg and add your new roles you just created in the previous step so it looks similar to the below output. Your cluster nodes will be different in your environment so change ses-node4 to match your setup.

```
# RGW
role-rgw/cluster/ses-node4*.sls
role-rgw-elasticsearch/cluster/ses-node4*.sls
role-rgw-us-west-1/cluster/ses-node4*.sls
```

Execute Deepsea stages [1-4]
Each stage needs to be rerun because of the creation of a new role.

```
$ deepsea stage run ceph.stage.1
$ deepsea stage run ceph.stage.2
$ deepsea stage run ceph.stage.3
$ deepsea stage run ceph.stage.4
```

After running stage 4 you will notice that two of the three radosgw daemons won’t start. This is fine since we have not created the zones for them yet. The output looks like below.

```
Ended stage: ceph.stage.4 succeeded=16/17 failed=1/17 time=96.0s

Failures summary:

ceph.rgw (/srv/salt/ceph/rgw):
 ses-node4.susedojo.com:
   start rgw-us-west-1: Service ceph-radosgw@rgw-us-west-1.ses-node4 is already
enabled, and is dead
   start rgw-elasticsearch: Service ceph-radosgw@rgw-elasticsearch.ses-node4 is
already enabled, and is dead
```

The default radosgw setup will have a single admin user with a zone and zonegroup called default.

System Key and RGW Host Parameters
Since there are multiple zones we will use the zone.user system key for cross radosgw communications. First we will create the access-key and secret that will later be used in the creation of the zone.user, but for now we can’t since order here does matter.

The variables we will be using for the CLI commands related to radosgw-admin are the following.

```
SYSTEM_ACCESS_KEY=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 20 | head -n 1)
SYSTEM_SECRET_KEY=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 40 | head -n 1)
RGW_HOST=Rados Gateway host name (in this example its ses-node4.susedojo.com)
```

These variables can be set directly in the shell and called through the commands given in the next steps.

Create a Realm

```
$ radosgw-admin realm create --rgw-realm=gold --default
```

Create zonegoup
The zonegroup that we create here will contain both the data zone and the elasticsearch zone, which will share the same data set. Change the RGW_HOST to match your environment. The port 8000 is what we setup for the radosgw daemon that is for the master zone we will create next. There is a port 80 which is the radosgw daemon that deepsea had setup for the “default” zone.

```
$ radosgw-admin zonegroup create --rgw-zonegroup=boise
–-endpoints=http://$RGW_HOST:8000 --master --default

{
   "id": "0c3bc24d-11c7-4214-98d9-0b1a764c65e5",
   "name": "boise",
   "api_name": "boise",
   "is_master": "true",
   "endpoints": [
       "http://ses-node4.susedojo.com:8000"
   ],
   "hostnames": [],
   "hostnames_s3website": [],
   "master_zone": "",
   "zones": [],
   "placement_targets": [],
   "default_placement": "",
   "realm_id": "73098396-10ff-4ff8-8a39-3da6415db903"
}
```

Create a Zone
We now create the normal zone where we will store the object data.
For the SYSTEM_ACCESS_KEY, SYSTEM_SECRET_KEY, and RGW_HOST parameters refer back to the section “System Key and RGW Host Parameters”

```
$ radosgw-admin zone create --rgw-zonegroup=boise --rgw-zone=us-west-1 \
  --endpoints=http://$RGW_HOST:8000 --access-key=$SYSTEM_ACCESS_KEY \
  --secret=$SYSTEM_SECRET_KEY --default --master
 {
   "id": "887e1765-7e0f-4936-ad32-1da30928dfce",
   "name": "us-west-1",
   "domain_root": "us-west-1.rgw.meta:root",
   "control_pool": "us-west-1.rgw.control",
   "gc_pool": "us-west-1.rgw.log:gc",
   "lc_pool": "us-west-1.rgw.log:lc",
   "log_pool": "us-west-1.rgw.log",
   "intent_log_pool": "us-west-1.rgw.log:intent",
   "usage_log_pool": "us-west-1.rgw.log:usage",
   "reshard_pool": "us-west-1.rgw.log:reshard",
   "user_keys_pool": "us-west-1.rgw.meta:users.keys",
   "user_email_pool": "us-west-1.rgw.meta:users.email",
   "user_swift_pool": "us-west-1.rgw.meta:users.swift",
   "user_uid_pool": "us-west-1.rgw.meta:users.uid",
   "system_key": {
       "access_key": "9AQ7TW70A38JF0M1MCY9",
       "secret_key": "MupKyw7vCKmQHLaK9U1SwshoR2yHBcPEGNosXKlK"
   },
   "placement_pools": [
       {
           "key": "default-placement",
           "val": {
               "index_pool": "us-west-1.rgw.buckets.index",
               "data_pool": "us-west-1.rgw.buckets.data",
               "data_extra_pool": "us-west-1.rgw.buckets.non-ec",
               "index_type": 0,
               "compression": ""
           }
       }
   ],
   "metadata_heap": "",
   "tier_config": [],
   "realm_id": "73098396-10ff-4ff8-8a39-3da6415db903"
 }
```

Remove default zone and zonegroup
Since in SUSE Enterprise Storage 5.5 you setup your radosgw through deepsea it automatically starts up the radosgw daemon and then creates the default zone and zonegroup. So these must be removed in order to move forward with multiple zones.

```
$ radosgw-admin zonegroup remove --rgw-zonegroup=default --rgw-zone=default
$ radosgw-admin period update --commit
$ radosgw-admin zone delete --rgw-zone=default
$ radosgw-admin period update --commit
$ radosgw-admin zonegroup delete --rgw-zonegroup=default
$ radosgw-admin period update --commit
```

Listing them now reveals only the zone and zonegroup you have defined.

```
$ radosgw-admin zonegroup list
$ radosgw-admin zone list
```

Create system user
The system user will be used to access the radosgw REST API and zone synchronization. This user is in addition to the admin user that deepsea already creates and ties to the default zone. The creation will auto generate the access and secret key used in future steps.

```
$ radosgw-admin user create --uid=zone.user --display-name="Zone User" --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY --system
$ radosgw-admin user list
[
   "zone.user"
]
```

Update the period
This commits the federation changes we’ve proposed so that the zones will start talking to each other.

```
$ radosgw-admin period update --commit

{
   "id": "8f2fc412-e4f9-4f00-bfab-6c60b09caba0",
   "epoch": 1,
   "predecessor_uuid": "32d0e270-0220-4ac7-86bf-a8114d34b825",
   "sync_status": [],
   "period_map": {
       "id": "8f2fc412-e4f9-4f00-bfab-6c60b09caba0",
       "zonegroups": [
…
"realm_id": "73098396-10ff-4ff8-8a39-3da6415db903",
   "realm_name": "gold",
   "realm_epoch": 2
}
```

Restart the RadosGW
Login to the node which will be running the three RGW services
This will start the radosgw daemon and tie it to zone us-west-1 with port 8000

```
$ systemctl restart ceph-radosgw@rgw-us-west-1.ses-node4.service

$ ss -4nlp | grep 8000
tcp    LISTEN     0      128       *:8000                  *:*
users:(("radosgw",pid=87906,fd=46))
```

Configure second zone in the same cluster, used for metadata indexing
This is the zone and radosgw daemon that will be dedicated for sending metadata to elasticsearch.
For the SYSTEM_ACCESS_KEY, SYSTEM_SECRET_KEY, and RGW_HOST parameters refer back to the section “System Key and RGW Host Parameters”

```
$ radosgw-admin zone create --rgw-zonegroup=boise --rgw-zone=us-west-2 \
  --access-key=$SYSTEM_ACCESS_KEY --secret=$SYSTEM_SECRET_KEY \
  --endpoints=http://$RGW_HOST:8002
```

Elasticsearch zone modify configuration
Here we will specify that the zone is an elasticsearch type and its associated endpint for the Elasticsearch API. In this case our efk-server we setup earlier.

```
$ radosgw-admin zone modify --rgw-zone=us-west-2 --tier-type=elasticsearch --tier-config=endpoint=http://efk-server.susedojo.com:9200,num_shards=10,num_replicas=1
```

Update the period
This commits the federation changes we’ve proposed so that the zones will start talking to each other.

```
$ radosgw-admin period update --commit
```

Restart the second RadosGW
Login to the node which will be running the three RGW services
This will start the radosgw daemon and tie it to zone us-west-2 with port 8002

```
$ systemctl restart ceph-radosgw@rgw-elasticsearch.ses-node4.service

$ ss -4nlp | grep 8002
tcp    LISTEN     0      128       *:8002                  *:*
users:(("radosgw",pid=88698,fd=86))
```

Status of services

```
$ ps auxf | grep radosgw
ceph       87906  0.3  0.8 5175324 64728 ?       Ssl  16:46   0:05 /usr/bin/radosgw
-f --cluster ceph --name client.rgw-us-west-1.ses-node4 --setuser ceph --
setgroup ceph
ceph       88698  2.2  0.8 5181472 64516 ?       Ssl  17:08   0:02 /usr/bin/radosgw
-f --cluster ceph --name client.rgw-elasticsearch.ses-node4 --setuser cep
```

```
h --setgroup ceph
```

You now have radosgw daemons for each zone running.

Create a user for bucket access

```
$ radosgw-admin user create --uid=cseader --display-name=”Cameron Seader”
```