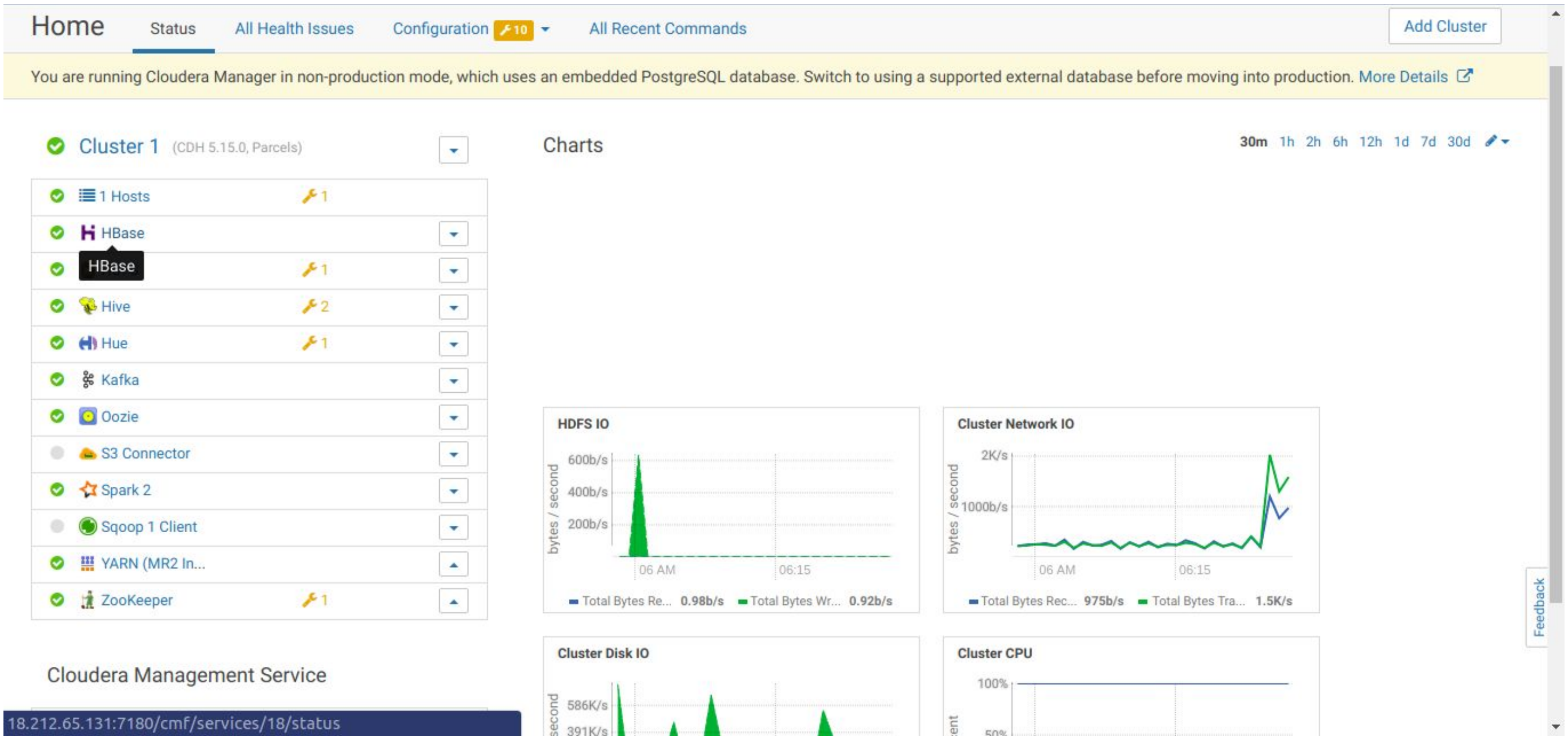
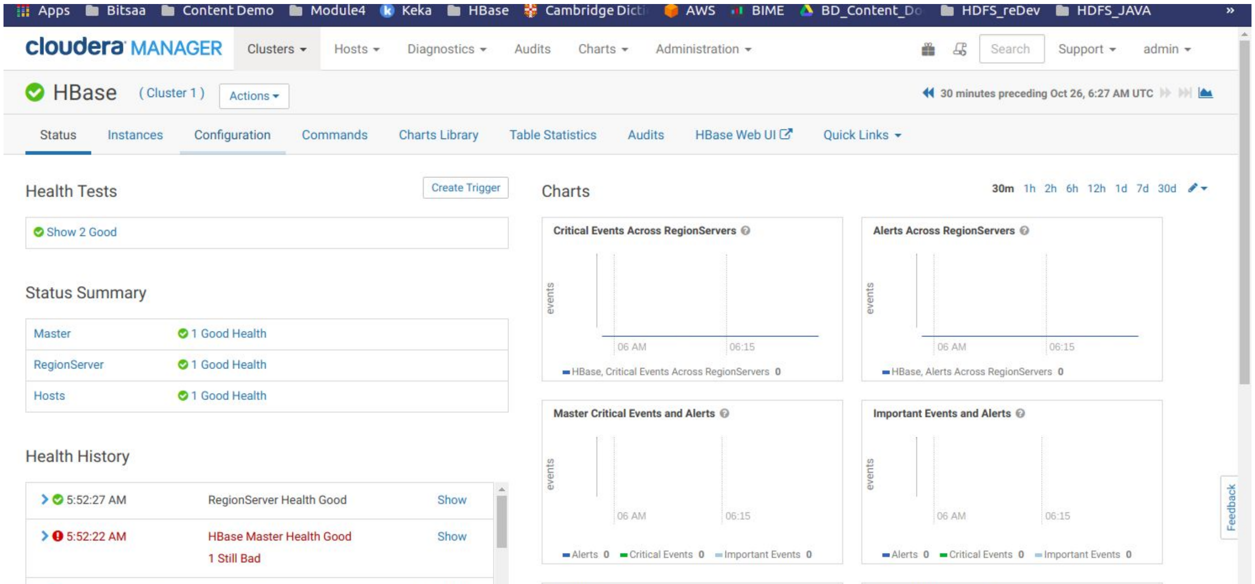
ACL Configuration in HBase

1. Open **Cloudera Manager** and Click on **HBase**



You are running Cloudera Manager in non-production mode, which uses an embedded PostgreSQL database. Switch to using a supported external database before moving into production. More Details

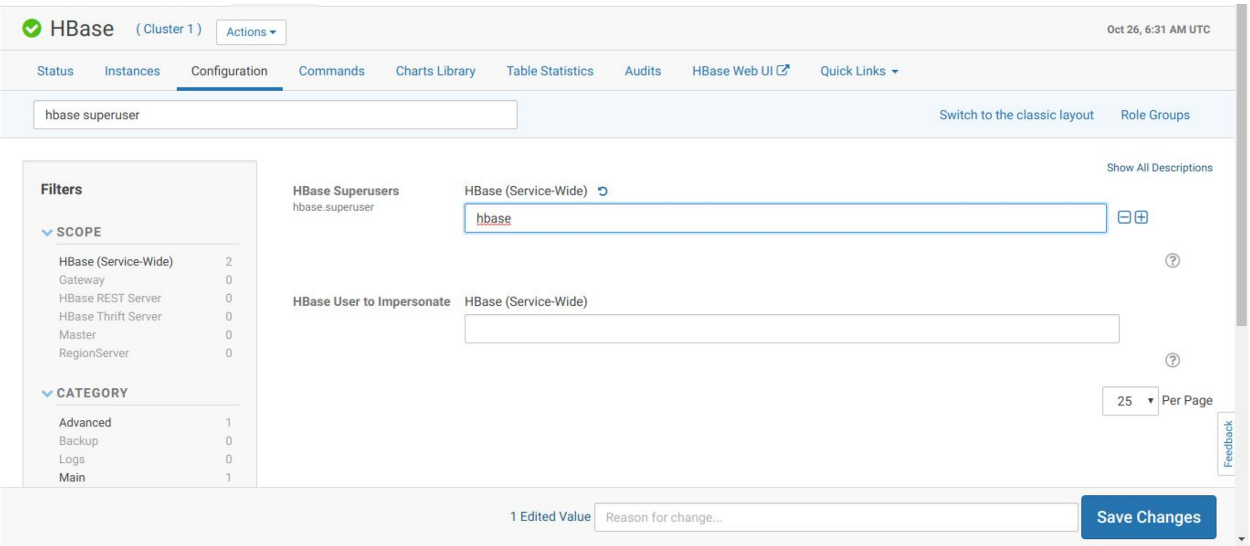
2. Click on **Configurations** and from the search bar, Search for following properties-



Master	1 Good Health
RegionServer	1 Good Health
Hosts	1 Good Health

5:52:27 AM	RegionServer Health Good	Show
5:52:22 AM	HBase Master Health Good 1 Still Bad	Show

a. Search **hbase.superuser** and put the value **hbase,** click on **Save Changes**



hbase superuser

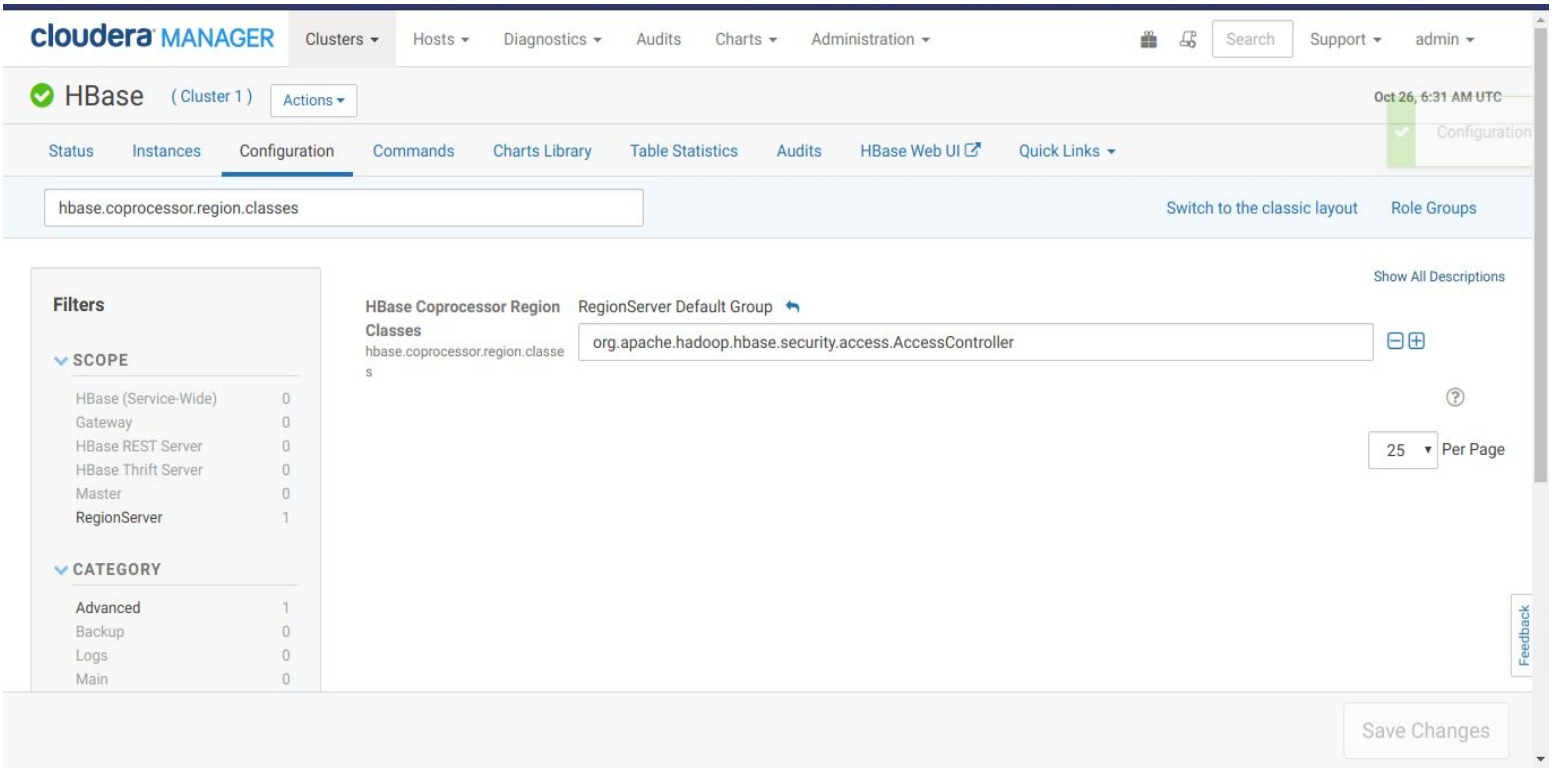
HBase (Service-Wide)	2
Gateway	0
HBase REST Server	0
HBase Thrift Server	0
Master	0
RegionServer	0
Advanced	1
Backup	0
Logs	0
Main	1

HBase Superusers
hbase.superuser
HBase (Service-Wide)
hbase

HBase User to Impersonate
HBase (Service-Wide)

1 Edited Value

B. search an **hbbase.coprocessor.region.classes** and put value as **org.apache.hadoop.hbase.security.access.AccessController,** click on **Save Changes**



hbase.coprocessor.region.classes

Filters

SCOPE

HBase (Service-Wide)	0
Gateway	0
HBase REST Server	0
HBase Thrift Server	0
Master	0
RegionServer	1

CATEGORY

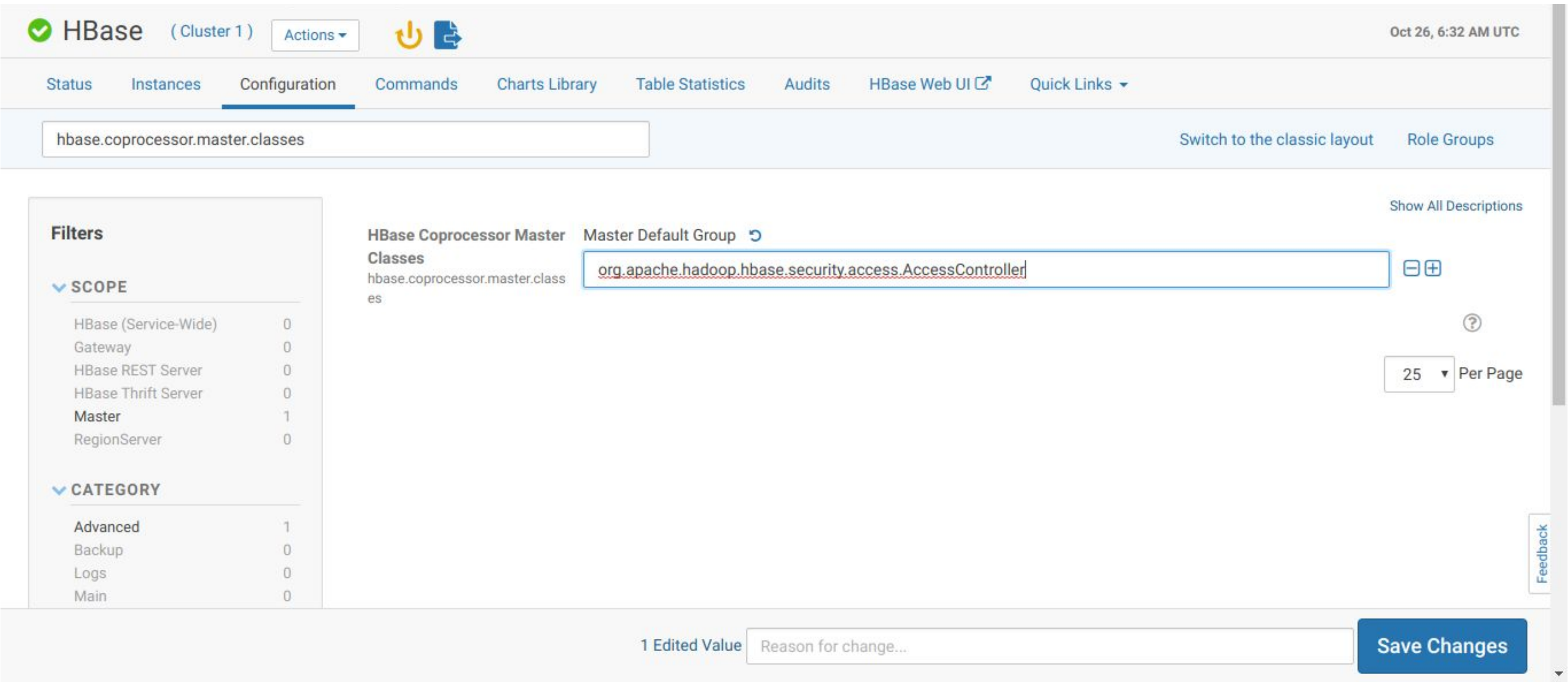
Advanced	1
Backup	0
Logs	0
Main	0

HBase Coprocessor Region Classes
hbase.coprocessor.region.classes

RegionServer Default Group

org.apache.hadoop.hbase.security.access.AccessController

C. search **hbase.coprocessor.master.classes** and put the value as **org.apache.hadoop.hbase.security.access.AccessController** then **Save Changes**



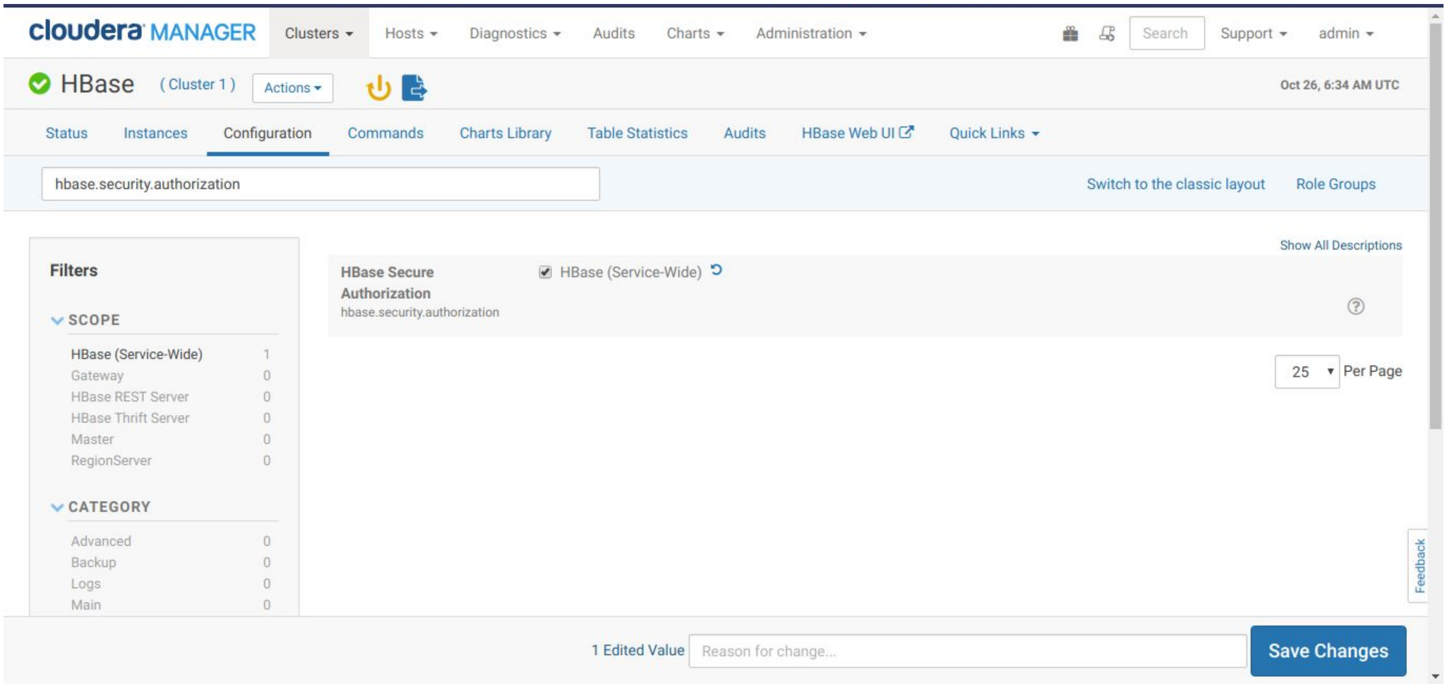
hbase.coprocessor.master.classes

HBase Coprocessor Master Classes
hbase.coprocessor.master.classes

Master Default Group: org.apache.hadoop.hbase.security.access.AccessController

1 Edited Value | Reason for change...

D. Search **Hbase.security.authorization**, tick the checkbox and **Save Changes**



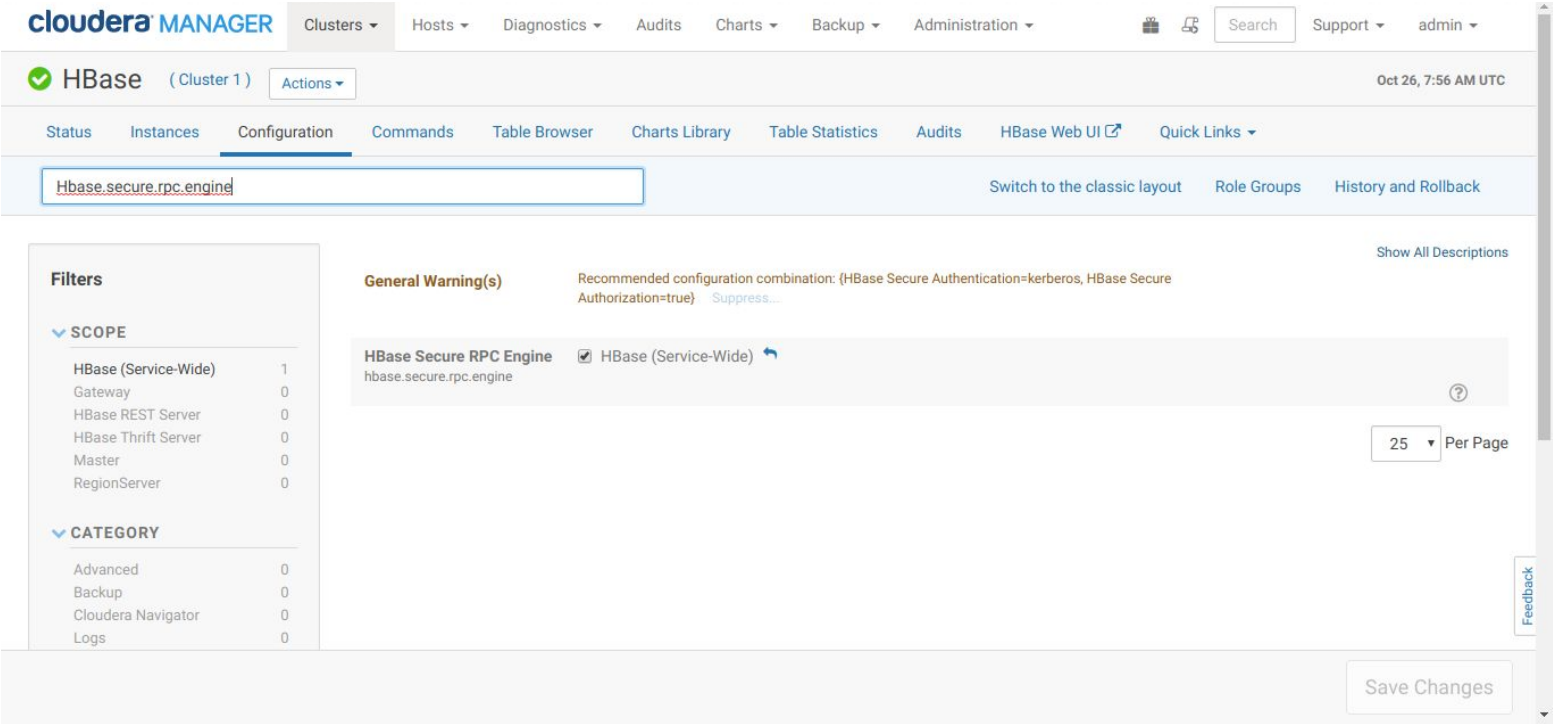
hbase.security.authorization

HBase Secure Authorization
hbase.security.authorization

☑ HBase (Service-Wide)

1 Edited Value | Reason for change...

E. search **hbase.secure.rpc.engine** tick the checkbox then **Save Changes.**
Note: if this property doesn’t appear on your system, Ignore it



Hbase.secure.rpc.engine

Filters

SCOPE

HBase (Service-Wide)	1
Gateway	0
HBase REST Server	0
HBase Thrift Server	0
Master	0
RegionServer	0

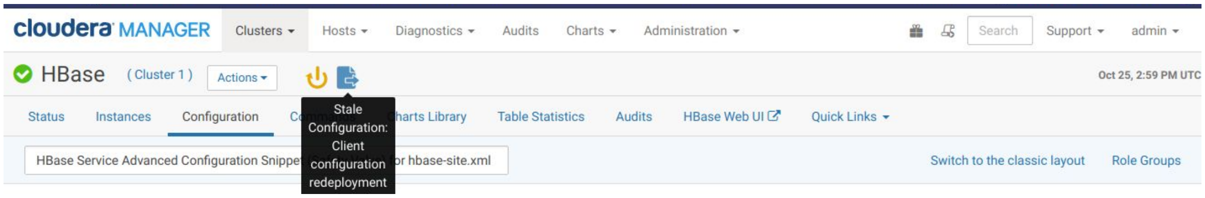
CATEGORY

Advanced	0
Backup	0
Cloudera Navigator	0
Logs	0

General Warning(s) Recommended configuration combination: {HBase Secure Authentication=kerberos, HBase Secure Authorization=true}

HBase Secure RPC Engine
hbase.secure.rpc.engine
☑ HBase (Service-Wide)

3. Click in **Stale Configuration** > **Restart Now** > **Finish**



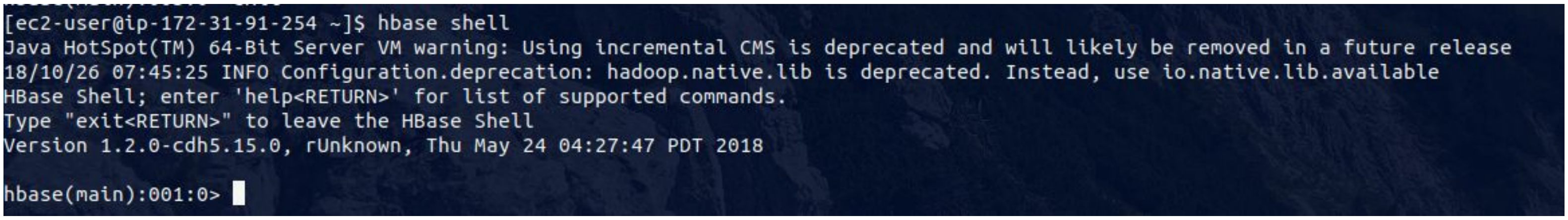
Stale Configuration: Client configuration redeployment

HBase Service Advanced Configuration Snippe ... r hbase-site.xml

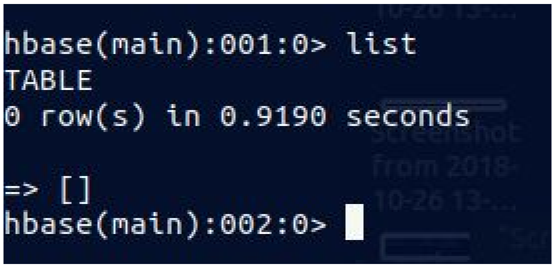
4. Restart your Cloudera Manager and Cluster services, if Needed.

Now your Security setup is ready Now, let’s see how to use the **grant**, **revoke** and **user_persmission** command in HBase. The first we set up, we actually secifired the superUser scope. We gave **‘hbase’** user the superuser rights, Note that the **superUser** above all scopes.

Open hbase shell from as **ec2-user**, and enter **list** command, as you can see below you can’t see any table(if there was any before), that is because now you have the security parameters, hence by default, the the **ec2-user** doesn’t have any kind of rights.



```
[ec2-user@ip-172-31-91-254 ~]$ hbase shell
Java HotSpot(TM) 64-Bit Server VM warning: Using incremental CMS is deprecated and will likely be removed in a future release
18/10/26 07:45:25 INFO Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.15.0, rUnknown, Thu May 24 04:27:47 PDT 2018

hbase(main):001:0>
```



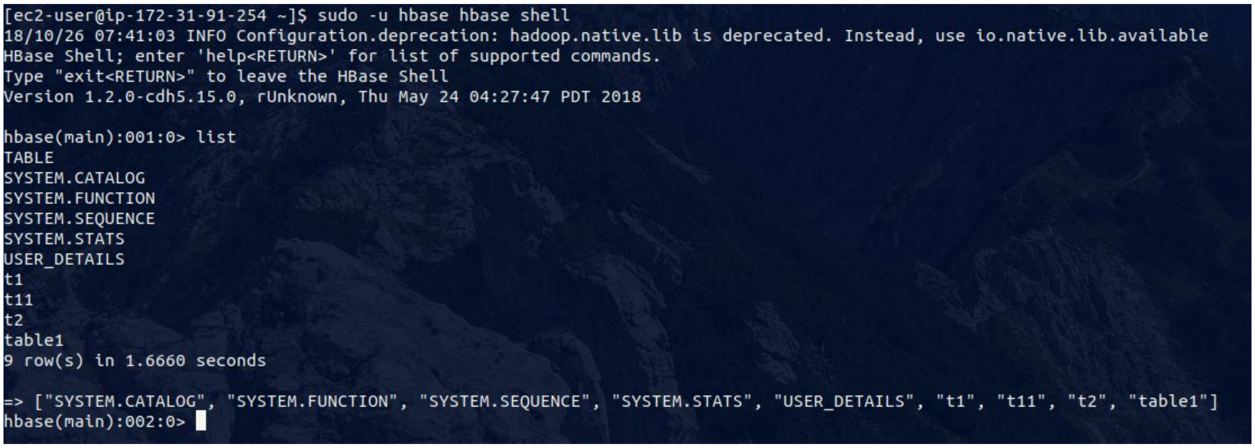
```
hbase(main):001:0> list
TABLE
0 row(s) in 0.9190 seconds

=> []
hbase(main):002:0>
```

Now let’s grant some permission to ec2-user for that we need to open hbase shell as the superuser, which is **hbase.** For that exit from this shell.

To open hbase shell as **hbase** superuser, enter the following command

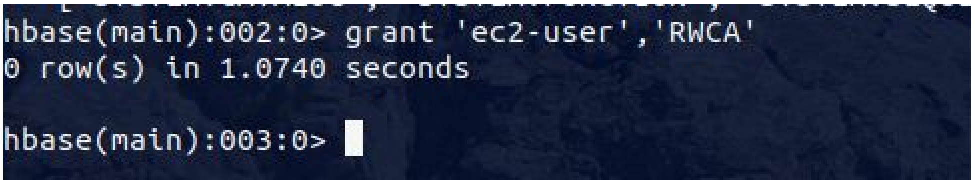
```
sudo -u hbase hbase shell
```

and now if you do a list, you can see all your tables, because the superuser by default has all the permissions.



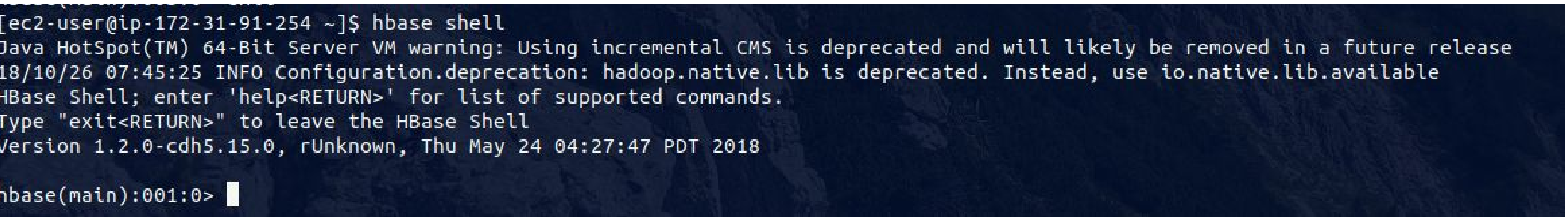
```
[ec2-user@ip-172-31-91-254 ~]$ sudo -u hbase hbase shell
18/10/26 07:41:03 INFO Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.15.0, rUnknown, Thu May 24 04:27:47 PDT 2018

hbase(main):001:0> list
TABLE
SYSTEM.CATALOG
SYSTEM.FUNCTION
SYSTEM.SEQUENCE
SYSTEM.STATS
USER_DETAILS
t1
t11
t2
table1
9 row(s) in 1.6660 seconds

=> ["SYSTEM.CATALOG", "SYSTEM.FUNCTION", "SYSTEM.SEQUENCE", "SYSTEM.STATS", "USER_DETAILS", "t1", "t11", "t2", "table1"]
hbase(main):002:0>
```

Now let's grant read, write, create, and admin(**RWCA**) permission to the **ec2-user**



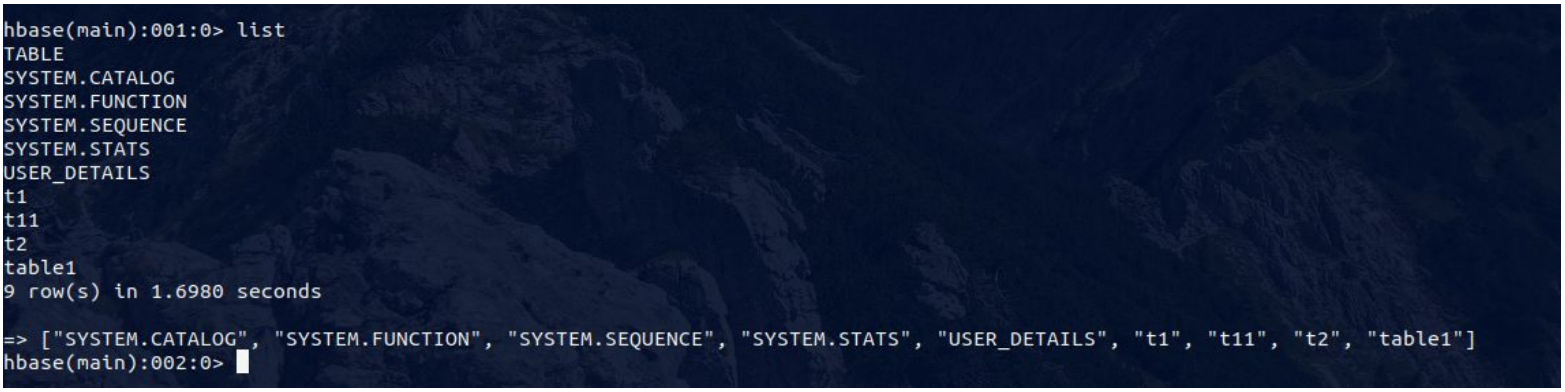
```
hbase(main):002:0> grant 'ec2-user','RWCA'
0 row(s) in 1.0740 seconds

hbase(main):003:0>
```

Now exit and open hbase shell as **ec2-user**



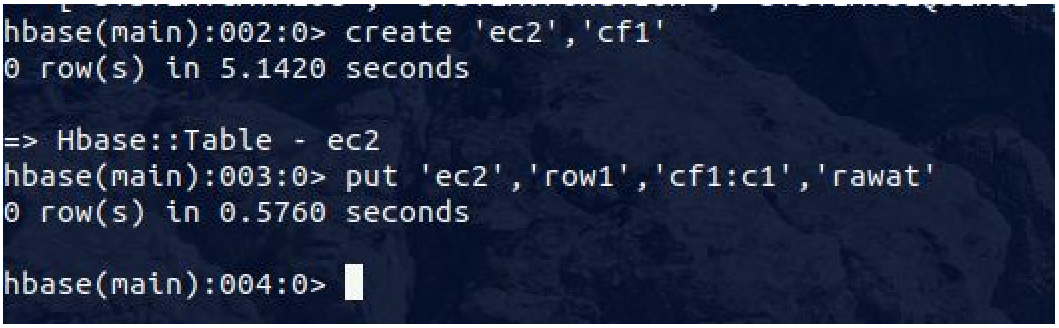
```
[ec2-user@ip-172-31-91-254 ~]$ hbase shell
Java HotSpot(TM) 64-Bit Server VM warning: Using incremental CMS is deprecated and will likely be removed in a future release
18/10/26 07:45:25 INFO Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.15.0, rUnknown, Thu May 24 04:27:47 PDT 2018

hbase(main):001:0>
```

Now **list**,all your table,



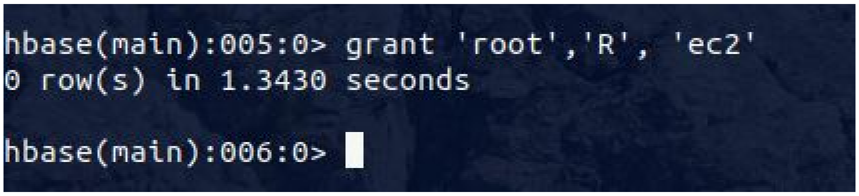
```
hbase(main):001:0> list
TABLE
SYSTEM.CATALOG
SYSTEM.FUNCTION
SYSTEM.SEQUENCE
SYSTEM.STATS
USER_DETAILS
t1
t11
t2
table1
9 row(s) in 1.6980 seconds

=> ["SYSTEM.CATALOG", "SYSTEM.FUNCTION", "SYSTEM.SEQUENCE", "SYSTEM.STATS", "USER_DETAILS", "t1", "t11", "t2", "table1"]
hbase(main):002:0>
```

Let’s create a table named ‘ec2’, Put a value in it



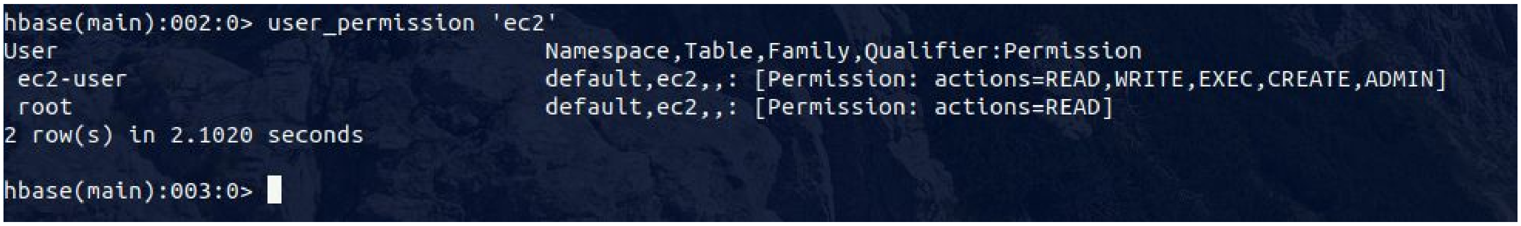
```
hbase(main):002:0> create 'ec2','cf1'
0 row(s) in 5.1420 seconds

=> Hbase::Table - ec2
hbase(main):003:0> put 'ec2','row1','cf1:c1','rawat'
0 row(s) in 0.5760 seconds

hbase(main):004:0> 
```

Let’s grant the **root** user only the **read** permission for the ‘ec2’ table



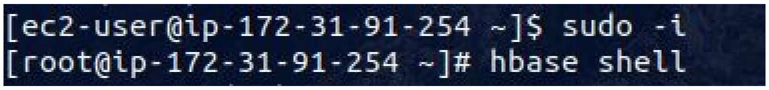
```
hbase(main):005:0> grant 'root','R', 'ec2'
0 row(s) in 1.3430 seconds

hbase(main):006:0> 
```

You can see all the permission granted for a table using user_permission,



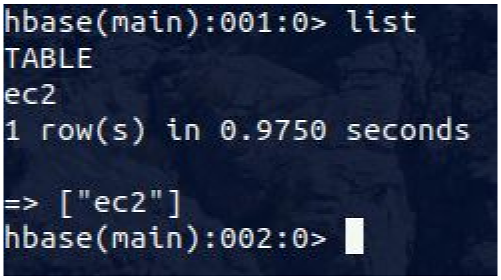
```
hbase(main):002:0> user_permission 'ec2'
User                                   Namespace,Table,Family,Qualifier:Permission
 ec2-user                              default,ec2,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN]
 root                                  default,ec2,,: [Permission: actions=READ]
2 row(s) in 2.1020 seconds

hbase(main):003:0> 
```

Let’s verify it from the root user, From this shell, and open hbase shell from **root** user



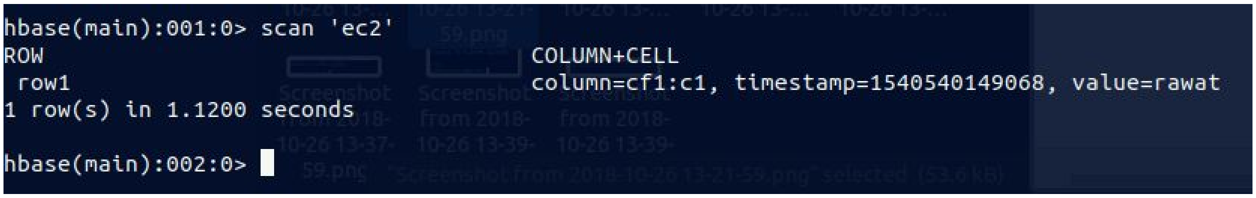
```
[ec2-user@ip-172-31-91-254 ~]$ sudo -i
[root@ip-172-31-91-254 ~]# hbase shell
```

Do a **list**, you can and see the ‘ec2’ table



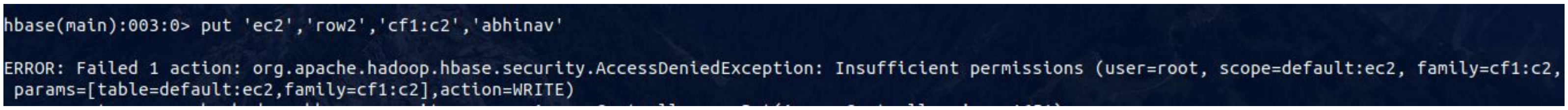
```
hbase(main):001:0> list
TABLE
ec2
1 row(s) in 0.9750 seconds

=> ["ec2"]
hbase(main):002:0>
```

Let’s read the ‘ec2’ table



```
hbase(main):001:0> scan 'ec2'
ROW                                COLUMN+CELL
 row1                              column=cf1:c1, timestamp=1540540149068, value=rawat
1 row(s) in 1.1200 seconds

hbase(main):002:0>
```

Now, let’s write a value in it,

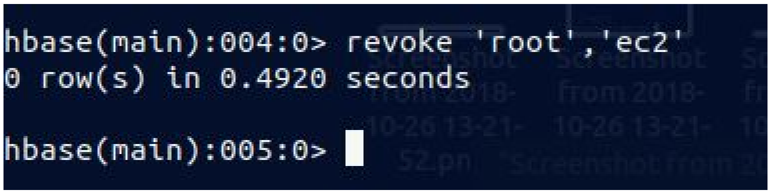


```
hbase(main):003:0> put 'ec2','row2','cf1:c2','abhinav'

ERROR: Failed 1 action: org.apache.hadoop.hbase.security.AccessDeniedException: Insufficient permissions (user=root, scope=default:ec2, family=cf1:c2,
 params=[table=default:ec2,family=cf1:c2],action=WRITE)
```

Because as ec2-user we only granted read permission for this ‘ec2’ table, the root can on;y read it and can’t write any value.

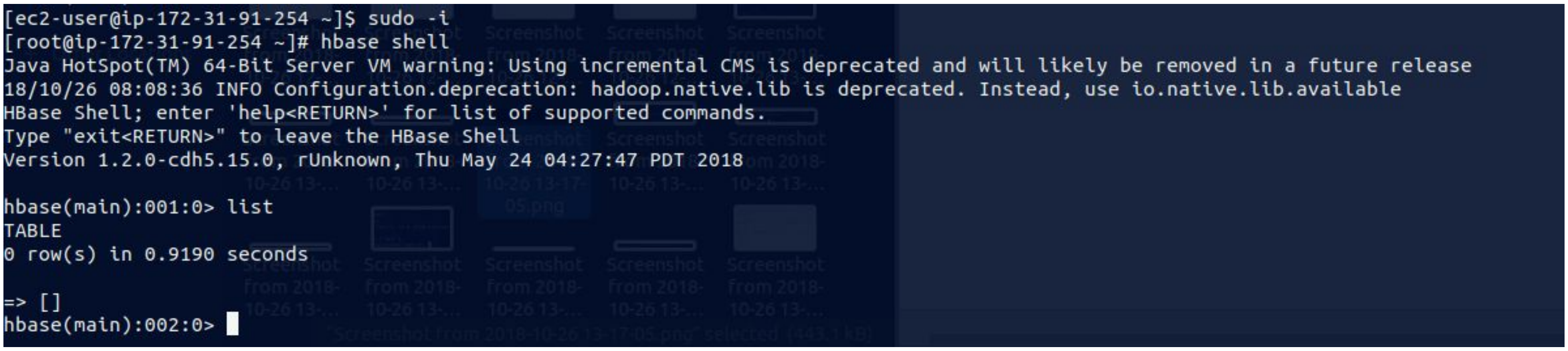
To revoke the permissions of the root user, go back and again open hbase shell from **ec2-user.** Following command is used to revoke all the permission of the root user from the ‘ec2’ table

```
revoke 'root','ec2'
```



```
hbase(main):004:0> revoke 'root','ec2'
0 row(s) in 0.4920 seconds

hbase(main):005:0>
```

As you can see below, the **read** permission of the **root** user has been revoked.



```
[ec2-user@ip-172-31-91-254 ~]$ sudo -i
[root@ip-172-31-91-254 ~]# hbase shell
Java HotSpot(TM) 64-Bit Server VM warning: Using incremental CMS is deprecated and will likely be removed in a future release
18/10/26 08:08:36 INFO Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.15.0, rUnknown, Thu May 24 04:27:47 PDT 2018

hbase(main):001:0> list
TABLE
0 row(s) in 0.9190 seconds

=> []
hbase(main):002:0>
```