TEST DRIVE PARTICIPANT LAB GUIDE FOR KASTEN EKS BACKUP RECOVERY ON AWS

Co-authored by

Yongkang He he@yongkang.cloud

Desmond Xu laidesmo@amazon.com

To be used with AWS Event Engine at:
https://dashboard.eventengine.run/

Lab Overview

This level 200 lab introduces you to the process of automating the creation of K10 and EKS cluster on AWS and provides guidance on backing up and restoring a sample Cassandra NoSQL database workload. This lab is to be conducted and hosted exclusively on **AWS Event Engine** and we do not recommend customers experimenting on their production AWS accounts. Lab will be conducted jointly by Veeam and AWS SAs as instructors and will last for approximately 90 minutes per session.

Prerequisites

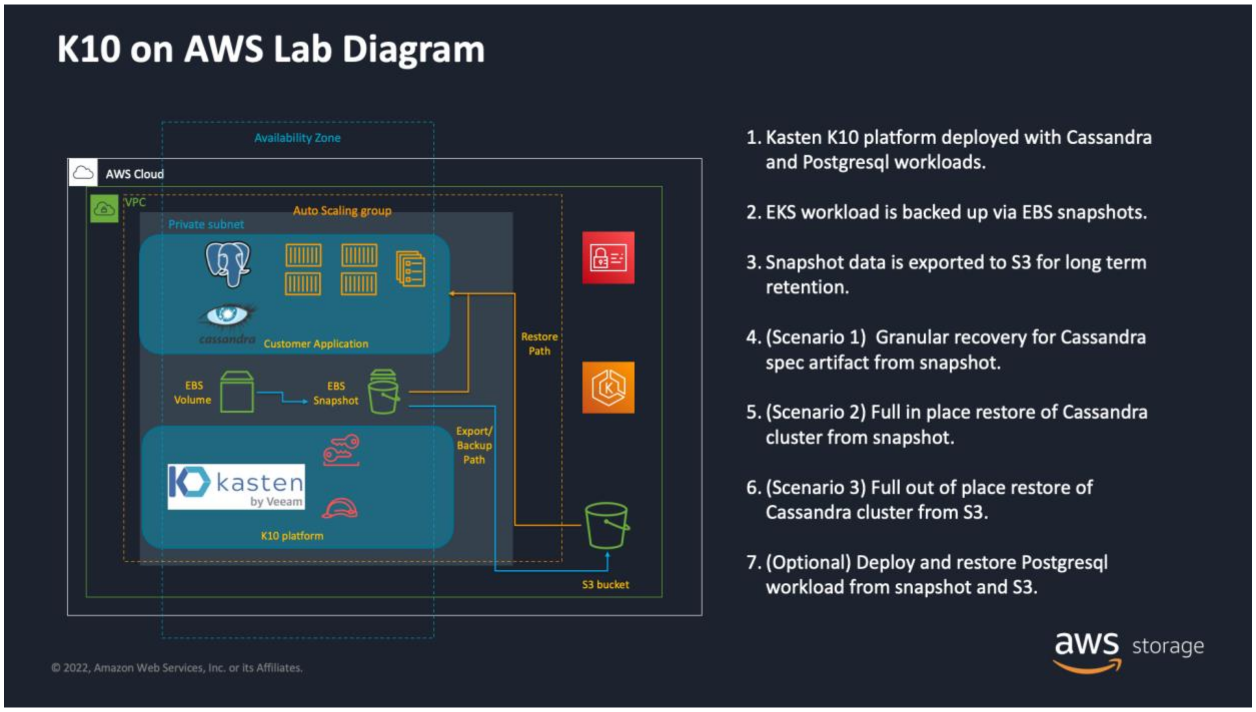
- AWS Event Engine account provisioned by AWS SAs
- Familiarity of backup and recovery
- Basic Knowledge of Kubernetes and AWS CloudShell
- Logout from your other AWS Cloud account first if you logged in, or you can open a New Incognito Window for Chrome and access AWS Event Engine website from there
- Optionally to watch the how-to guide http://ee-eks-k10.yongkang.cloud/
- Optionally to watch how to create an EKS cluster, backup containers, previous workshop http://aws.yongkang.cloud/

NOTE:

Each CloudShell session will timeout after 20 minutes or so of inactivity, and can be re-established by refreshing the window. If you are logged out of CloudShell at any time due to inactivity or by accident, you should run ./ee-destroy.sh first to clean up the environment from the previous session before continuing the labs. Please try to avoid the session expired or quit the CloudShell while you are doing the labs.

To keep your session active, you can simply come back to the CloudShell and click anywhere with your mouse from time to time e.g. every 10 minutes.

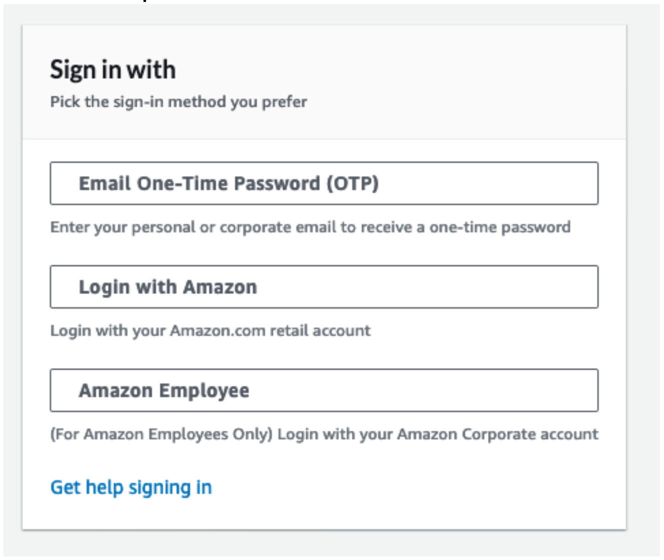
K10 on AWS Lab Diagram



1. Kasten K10 platform deployed with Cassandra and Postgresql workloads.
2. EKS workload is backed up via EBS snapshots.
3. Snapshot data is exported to S3 for long term retention.
4. (Scenario 1) Granular recovery for Cassandra spec artifact from snapshot.
5. (Scenario 2) Full in place restore of Cassandra cluster from snapshot.
6. (Scenario 3) Full out of place restore of Cassandra cluster from S3.
7. (Optional) Deploy and restore Postgresql workload from snapshot and S3.

AWS Account Setup

1. Log in to your provisioned AWS Event Engine here provide by AWS SA with the right event hash.
2. After entering the hash, select Email One-Time OTP as the preferred sign in method. The OTP password will be sent to the preferred email.



Sign in with

Pick the sign-in method you prefer

Enter your personal or corporate email to receive a one-time password

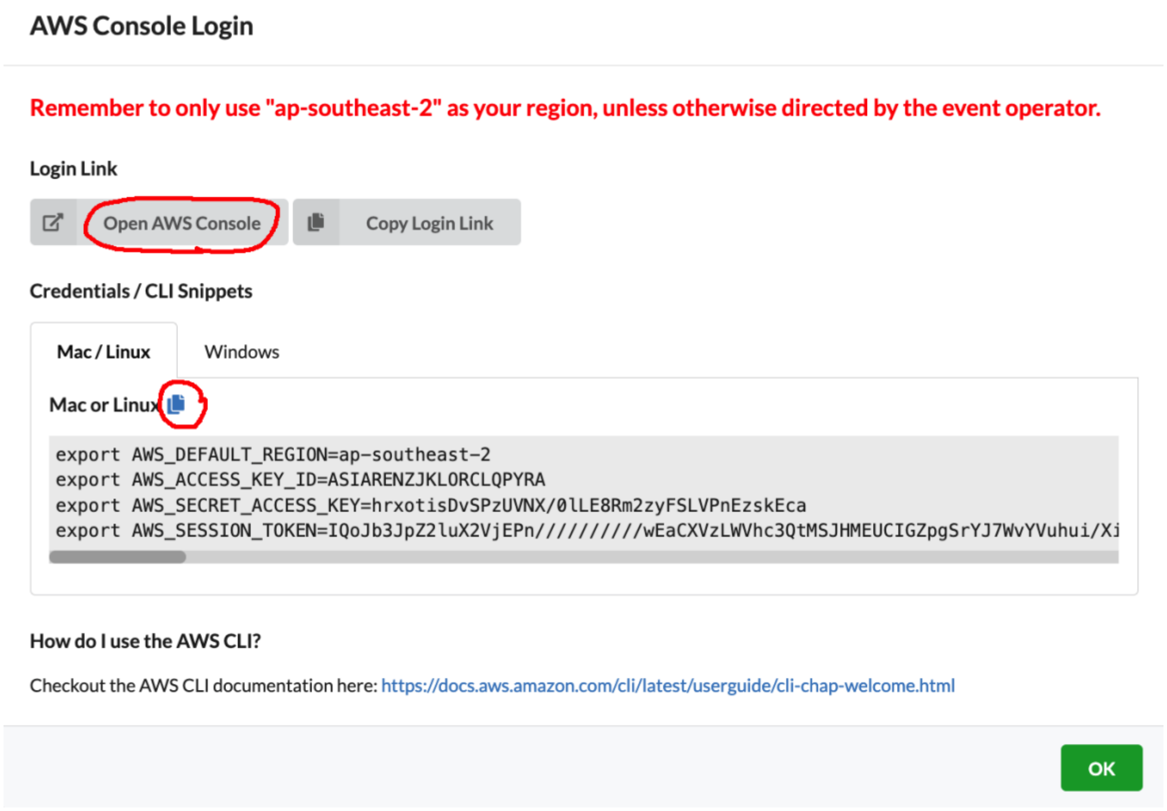
Login with your Amazon.com retail account

(For Amazon Employees Only) Login with your Amazon Corporate account

3. Click “AWS Console” to open “AWS Console Login” page. Click the icon next to “Mac or Linux” to copy all the provided Credentials / CLI Snippets, finally click on “Open AWS Console” to launch AWS Web Console.

NOTE:

Please make sure you’re doing the labs in the region mentioned in the line highlighted in red colour. E.g. ap-southeast-2, us-east-1, eu-west-1 etc..



AWS Console Login

Remember to only use "ap-southeast-2" as your region, unless otherwise directed by the event operator.

Login Link

Credentials / CLI Snippets

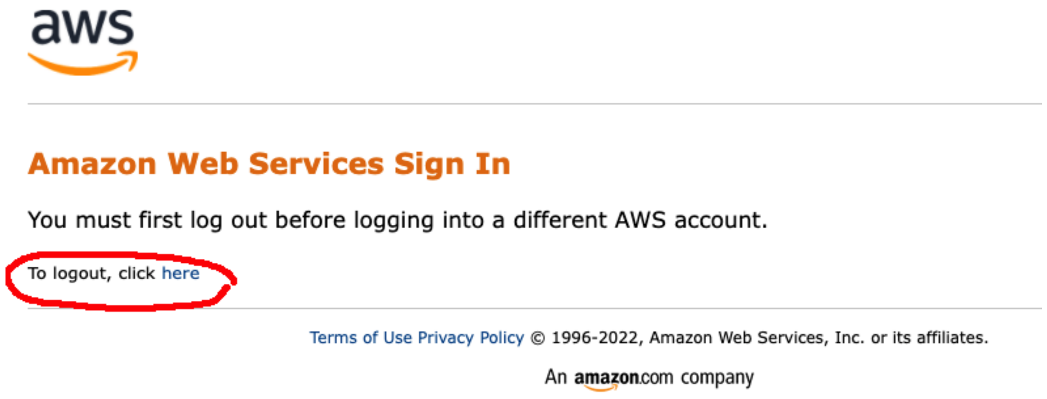
Mac or Linux

```
export AWS_DEFAULT_REGION=ap-southeast-2
export AWS_ACCESS_KEY_ID=ASIARENZJKLORCLQPYRA
export AWS_SECRET_ACCESS_KEY=hrxotisDvSPzUVNX/0lLE8Rm2zyFSLVPnEzskEca
export AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjEPn//////////wEaCXVzLWVhc3QtMSJHMEUCIGZpgSrYJ7WvYVuhui/Xi
```

How do I use the AWS CLI?

Checkout the AWS CLI documentation here: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html

If you did not logout from your other AWS Cloud account, you will see below screen. If you see this, you need to click here to logout first. Alternatively, you can access the event engine from an incognito window.



Amazon Web Services Sign In

You must first log out before logging into a different AWS account.

To logout, click here

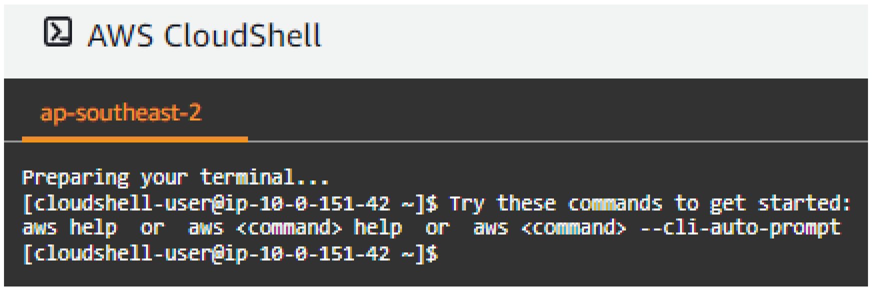
Terms of Use Privacy Policy © 1996-2022, Amazon Web Services, Inc. or its affiliates.

An amazon.com company

4. DO NOT change to other region, ensure that the region selected for your account matches the region mentioned in the last step. E.g. Sydney **ap-southeast-2**, N. Virginia **us-east-1**, Ireland **eu-west-1** etc..

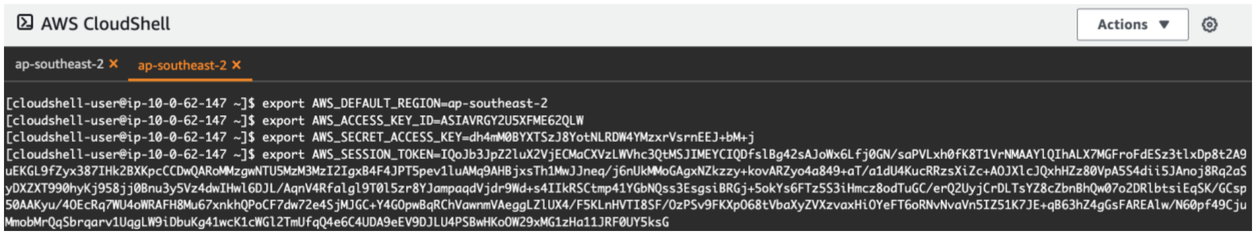
Access AWS CloudShell & environment prerequisites

1. Navigate to AWS CloudShell and wait for terminal to initialize.



```
Preparing your terminal...
[cloudshell-user@ip-10-0-151-42 ~]$ Try these commands to get started:
aws help  or  aws <command> help  or  aws <command> --cli-auto-prompt
[cloudshell-user@ip-10-0-151-42 ~]$
```

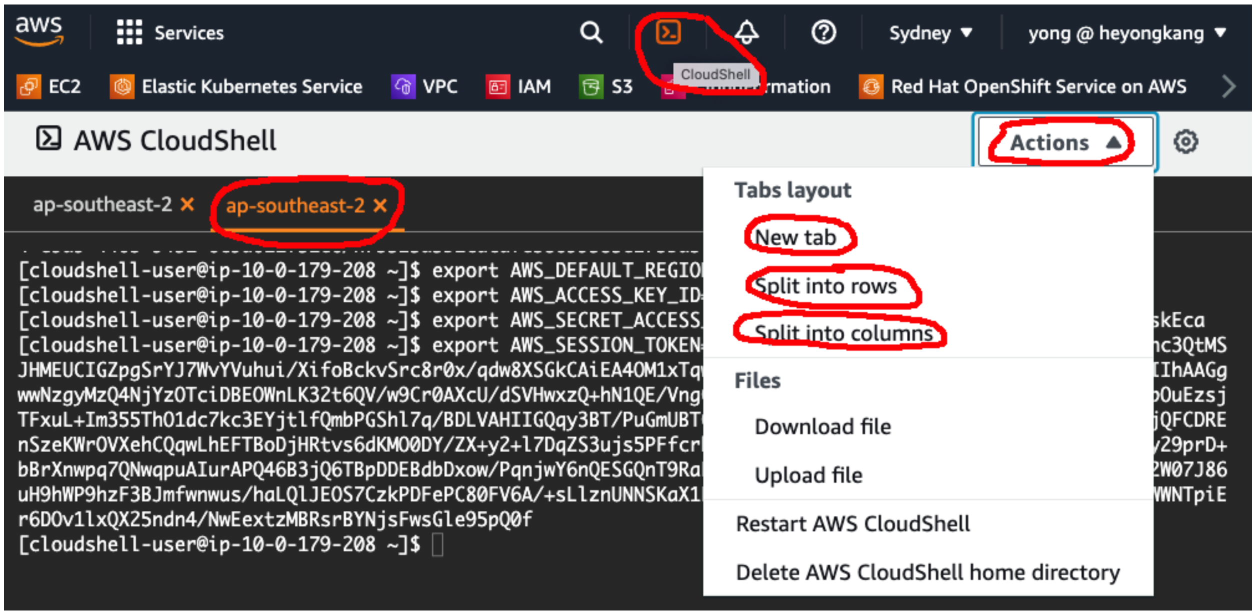
2. Paste the Credentials / CLI Snippets to CloudShell and Press Enter



```
[cloudshell-user@ip-10-0-62-147 ~]$ export AWS_DEFAULT_REGION=ap-southeast-2
[cloudshell-user@ip-10-0-62-147 ~]$ export AWS_ACCESS_KEY_ID=ASIAVRGY2U5XFME62QLW
[cloudshell-user@ip-10-0-62-147 ~]$ export AWS_SECRET_ACCESS_KEY=dh4mM0BYXTSzJ8YotNLRDW4YMzxrVsrnEEJ+bM+j
[cloudshell-user@ip-10-0-62-147 ~]$ export AWS_SESSION_TOKEN=IQoJb3JpZ2luX2VjECMaCXVzLWVhc3QtMSJIMEYCIQDfslBg42sAJoWx6Lfj0GN/saPVLxh0fK8T1VrNMAAYlQIhALX7MGFroFdESz3tlxDp8t2A9
uEKGL9fZyx387IHk2BXKpcCCDwQARoMMzgwNTU5MzM3MzI2IgxB4F4JPT5pev1luAMq9AHBjxsTh1MwJJneq/j6nUkMMoGAgxNZkzzy+kovARZyo4a849+aT/a1dU4KucRRzsXiZc+AOJXlcJQxhHZz80VpA5S4dii5JAnoj8Rq2aS
yDXZXT990hyKj958jj0Bnu3y5Vz4dwIHwl6DJL/AqnV4Rfalgl9T0l5zr8YJampaqdVjdr9Wd+s4IIkRSCtmp41YGbNQss3EsgsiBRGj+5okYs6FTz5S3iHmcz8odTuGC/erQ2UyjCrDLTsYZ8cZbnBhQw07o2DRlbtsiEqSK/GCsp
50AAKyu/4OEcRq7WU4oWRAFH8Mu67xnkhQPoCF7dw72e4SjMJGC+Y4GOpwBqRChVawnmVAeggLZlUX4/F5KLnHVTI8SF/OzPSv9FKXpO68tVbaXyZVXzvaxHiOYeFT6oRNvNvaVn5IZ51K7JE+qB63hZ4gGsFAREAlw/N60pf49Cju
MmobMrQqSbrqarv1UqgLW9iDbuKg41wcK1cWGl2TmUfqQ4e6C4UDA9eEV9DJLU4PSBwHKoOW29xMG1zHa11JRF0UY5ksG
```

NOTE:

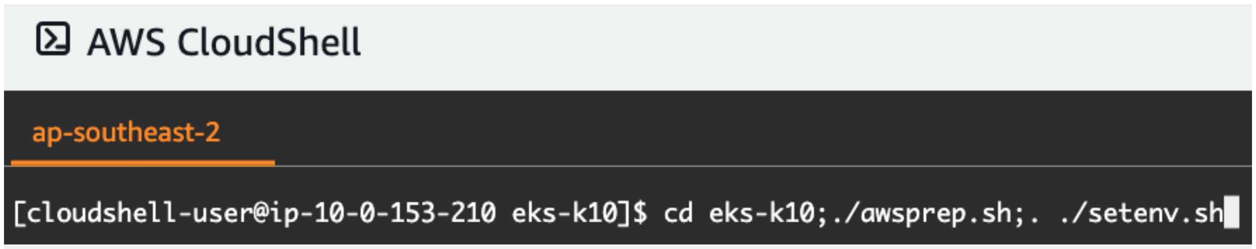
These environment variables are not permanently set in your terminal. It will be missing if you exit from the terminal and then re-login to the terminal or open a new CloudShell tab, Split into rows or columns. So, you need to make sure to re-copy/paste the Credentials / CLI Snippets to the new CloudShell Window or Tab.



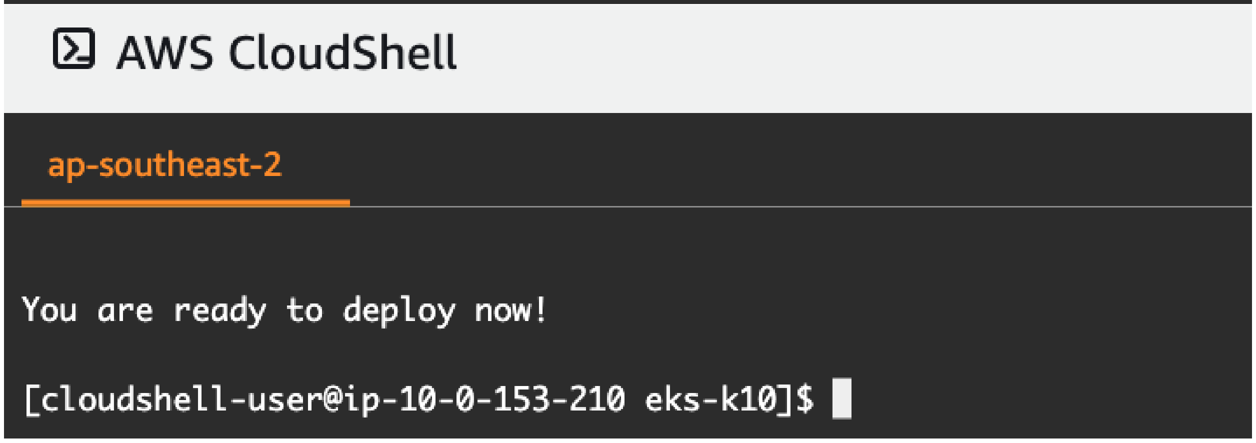
```
[cloudshell-user@ip-10-0-179-208 ~]$ export AWS_DEFAULT_REGIO
[cloudshell-user@ip-10-0-179-208 ~]$ export AWS_ACCESS_KEY_ID
[cloudshell-user@ip-10-0-179-208 ~]$ export AWS_SECRET_ACCESS                  skEca
[cloudshell-user@ip-10-0-179-208 ~]$ export AWS_SESSION_TOKEN                  hc3QtMS
JHMEUCIGZpgSrYJ7WvYVuhui/XifoBckvSrc8r0x/qdw8XSGkCAiEA4OM1xTq                  IIhAAGg
wwNzgyMzQ4NjYzOTciDBEOWnLK32t6QV/w9Cr0AXcU/dSVHwxzQ+hN1QE/Vng                  bOuEzsj
TFxuL+Im355ThO1dc7kc3EYjtlfQmbPGShl7q/BDLVAHIIGQqy3BT/PuGmUBT                  jQFCDRE
nSzeKWrOVXehCQqwLhEFTBoDjHRtvs6dKMO0DY/ZX+y2+l7DqZS3ujs5PFfcr                  y29prD+
bBrXnwpq7QNwqpuAIurAPQ46B3jQ6TBpDDEBdbDxow/PqnjwY6nQESGQnT9Ra                  2W07J86
uH9hWP9hzF3BJmfwnwus/haLQlJEOS7CzkPDFePC80FV6A/+sLlznUNNSKaX1                  WWNTpiE
r6DOv1lxQX25ndn4/NwEextzMBRsrBYNjsFwsGle95pQ0f
[cloudshell-user@ip-10-0-179-208 ~]$
```

3. Clone the github repo below with the following command:
   git clone https://github.com/yongkanghe/eks-k10

4. Change to the folder and install the required tools (eksctl, kubectl, helm) and source the environment variables:
   cd eks-k10;./awsprep.sh;. ./setenv.sh



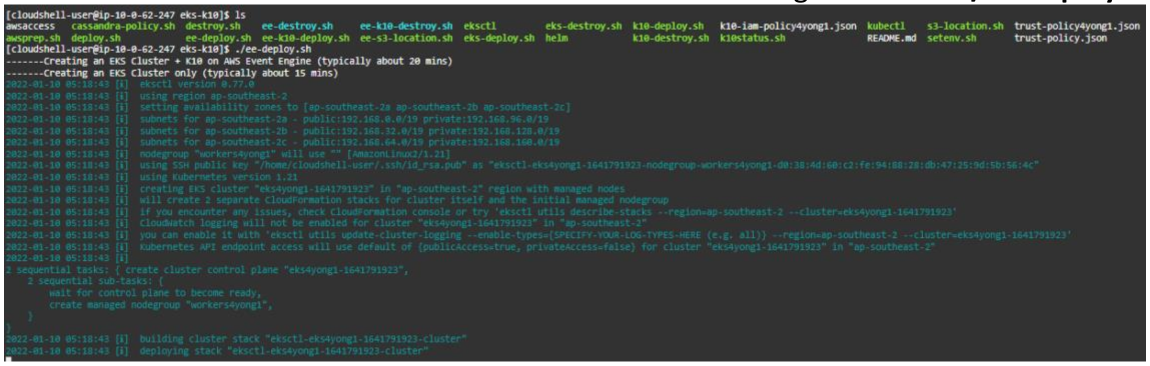
```
AWS CloudShell
ap-southeast-2
[cloudshell-user@ip-10-0-153-210 eks-k10]$ cd eks-k10;./awsprep.sh;. ./setenv.sh
```



```
AWS CloudShell
ap-southeast-2

You are ready to deploy now!

[cloudshell-user@ip-10-0-153-210 eks-k10]$ 
```

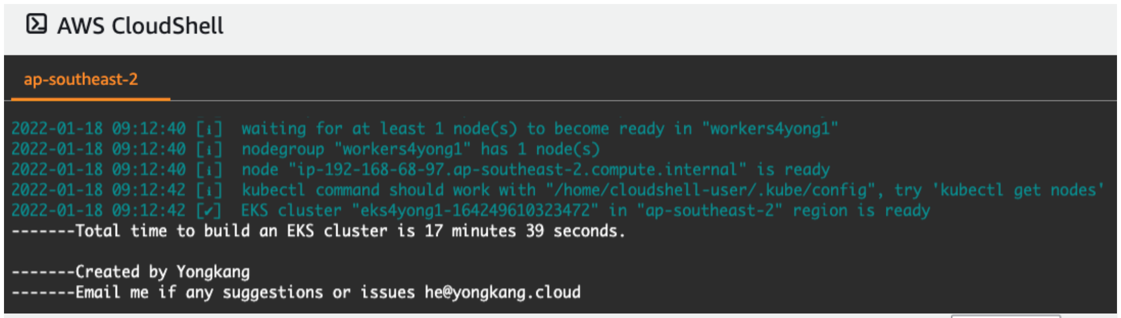
Deploy and build EKS clusters

1. Create the EKS cluster and lab environment with the following command: **./ee-deploy.sh**

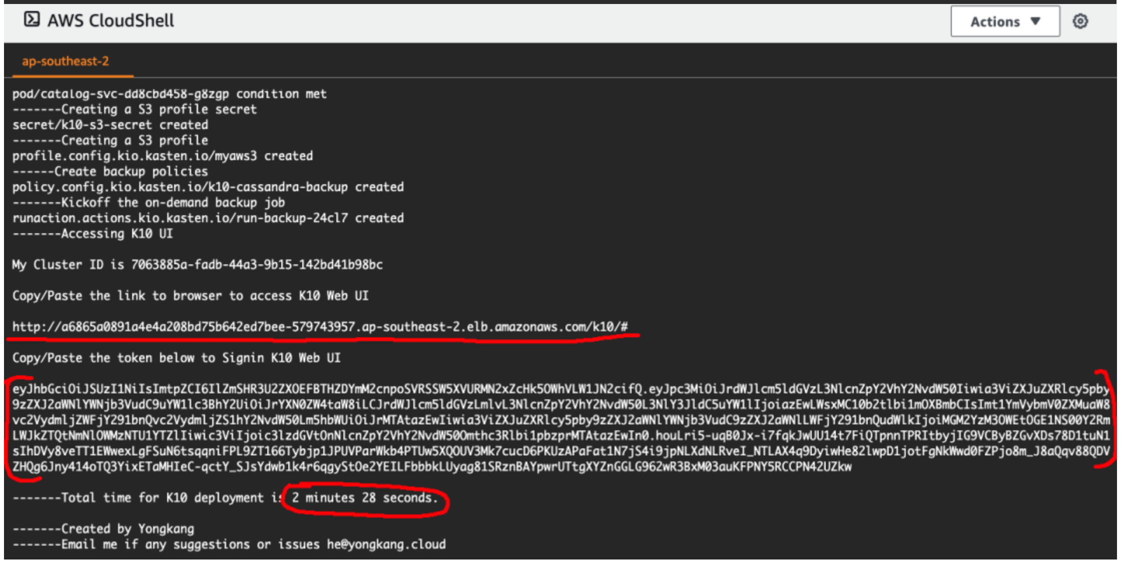


```
[cloudshell-user@ip-10-0-62-247 eks-k10]$ ls
awsaccess   cassandra-policy.sh  destroy.sh    ee-destroy.sh     ee-k10-destroy.sh  eksctl        eks-destroy.sh  k10-deploy.sh   k10-iam-policy4yong1.json  kubectl    s3-location.sh  trust-policy4yong1.json
awsprep.sh  deploy.sh                          ee-deploy.sh  ee-k10-deploy.sh  ee-s3-location.sh  eks-deploy.sh  helm            k10-destroy.sh  k10status.sh               README.md  setenv.sh       trust-policy.json
[cloudshell-user@ip-10-0-62-247 eks-k10]$ ./ee-deploy.sh
-------Creating an EKS Cluster + K10 on AWS Event Engine (typically about 20 mins)
-------Creating an EKS Cluster only (typically about 15 mins)
2022-01-10 05:18:43 [i]  eksctl version 0.77.0
2022-01-10 05:18:43 [i]  using region ap-southeast-2
2022-01-10 05:18:43 [i]  setting availability zones to [ap-southeast-2a ap-southeast-2b ap-southeast-2c]
2022-01-10 05:18:43 [i]  subnets for ap-southeast-2a - public:192.168.0.0/19 private:192.168.96.0/19
2022-01-10 05:18:43 [i]  subnets for ap-southeast-2b - public:192.168.32.0/19 private:192.168.128.0/19
2022-01-10 05:18:43 [i]  subnets for ap-southeast-2c - public:192.168.64.0/19 private:192.168.160.0/19
2022-01-10 05:18:43 [i]  nodegroup "workers4yong1" will use "" [AmazonLinux2/1.21]
2022-01-10 05:18:43 [i]  using SSH public key "/home/cloudshell-user/.ssh/id_rsa.pub" as "eksctl-eks4yong1-1641791923-nodegroup-workers4yong1-d0:38:4d:60:c2:fe:94:88:28:db:47:25:9d:5b:56:4c"
2022-01-10 05:18:43 [i]  using Kubernetes version 1.21
2022-01-10 05:18:43 [i]  creating EKS cluster "eks4yong1-1641791923" in "ap-southeast-2" region with managed nodes
2022-01-10 05:18:43 [i]  will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2022-01-10 05:18:43 [i]  if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=ap-southeast-2 --cluster=eks4yong1-1641791923'
2022-01-10 05:18:43 [i]  CloudWatch logging will not be enabled for cluster "eks4yong1-1641791923" in "ap-southeast-2"
2022-01-10 05:18:43 [i]  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)} --region=ap-southeast-2 --cluster=eks4yong1-1641791923'
2022-01-10 05:18:43 [i]  Kubernetes API endpoint access will use default of {publicAccess=true, privateAccess=false} for cluster "eks4yong1-1641791923" in "ap-southeast-2"
2022-01-10 05:18:43 [i]
2 sequential tasks: { create cluster control plane "eks4yong1-1641791923",
    2 sequential sub-tasks: {
        wait for control plane to become ready,
        create managed nodegroup "workers4yong1",
    }
}
2022-01-10 05:18:43 [i]  building cluster stack "eksctl-eks4yong1-1641791923-cluster"
2022-01-10 05:18:43 [i]  deploying stack "eksctl-eks4yong1-1641791923-cluster"
```

2. This will setup and automate deployment of 1) EKS Cluster 2) Install K10 3) Deploy a Cassandra NoSQL database 4) Create a S3 location profile 5) Create a backup policy and 6) Kick off an on-demand backup job. This step will take approximately 20 minutes to complete. When the job completes, you are expecting to see the screen similar as below with the output of K10 Web Console URL and Token Code.

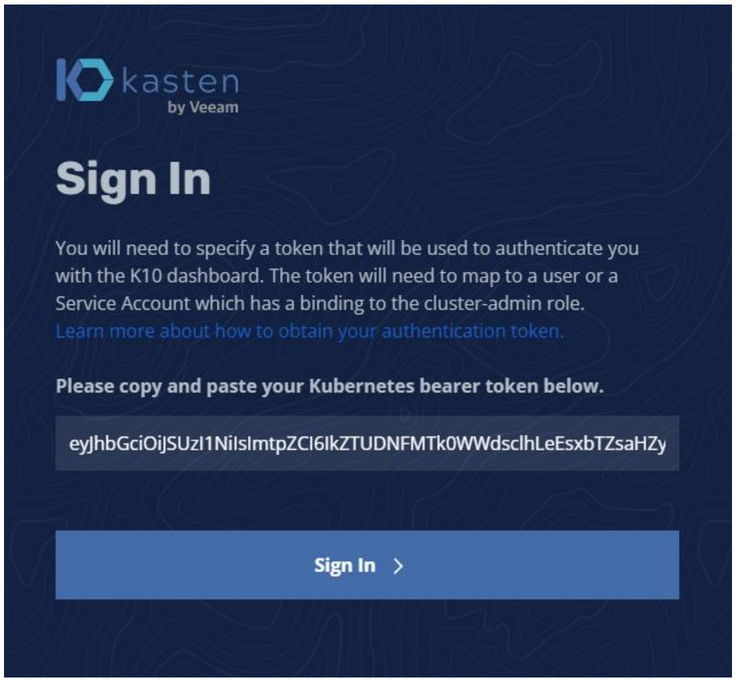


```
2022-01-18 09:12:40 [ℹ]  waiting for at least 1 node(s) to become ready in "workers4yong1"
2022-01-18 09:12:40 [ℹ]  nodegroup "workers4yong1" has 1 node(s)
2022-01-18 09:12:40 [ℹ]  node "ip-192-168-68-97.ap-southeast-2.compute.internal" is ready
2022-01-18 09:12:42 [ℹ]  kubectl command should work with "/home/cloudshell-user/.kube/config", try 'kubectl get nodes'
2022-01-18 09:12:42 [✔]  EKS cluster "eks4yong1-1642496103234 72" in "ap-southeast-2" region is ready
-------Total time to build an EKS cluster is 17 minutes 39 seconds.

-------Created by Yongkang
-------Email me if any suggestions or issues he@yongkang.cloud
```



```
pod/catalog-svc-dd8cbd458-g8zgp condition met
-------Creating a S3 profile secret
secret/k10-s3-secret created
-------Creating a S3 profile
profile.config.kio.kasten.io/myaws3 created
------Create backup policies
policy.config.kio.kasten.io/k10-cassandra-backup created
-------Kickoff the on-demand backup job
runaction.actions.kio.kasten.io/run-backup-24cl7 created
-------Accessing K10 UI

My Cluster ID is 7063885a-fadb-44a3-9b15-142bd41b98bc

Copy/Paste the link to browser to access K10 Web UI

http://a6865a0891a4e4a208bd75b642ed7bee-579743957.ap-southeast-2.elb.amazonaws.com/k10/#

Copy/Paste the token below to Signin K10 Web UI

eyJhbGciOiJSUzI1NiIsImtpZCI6IlZmSHR3U2ZXOEFBTHZDYmM2cnpoSVRSSW5XVURMN2xZcHk5OWhVLW1JN2cifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby
9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrYXN0ZW4taW8iLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoiazEwLWsxMC10b2tlbi1mOXBmbCIsImt1YmVybmV0ZXMuaW8
vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50Lm5hbWUiOiJrMTAtazEwIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiMGM2YzM3OWEtOGE1NS00Y2Rm
LWJkZTQtNmNlOWMzNTU1YTZlIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50Omthc3Rlbi1pbzprMTAtazEwIn0.houLri5-uqB0Jx-i7fqkJwUU14t7FiQTpnnTPRItbyjIG9VCByBZGvXDs78D1tuN1
sIhDVy8veTT1EWwexLgFSuN6tsqqniFPL9ZT166Tybjp1JPUVParWkb4PTUw5XQOUV3Mk7cucD6PKUzAPaFat1N7jS4i9jpNLXdNLRveI_NTLAX4q9DyiwHe82lwpD1jotFgNkWwd0FZPjo8m_J8aQqv88QDV
ZHQg6Jny414oTQ3YixETaMHIeC-qctY_SJsYdwb1k4r6qgyStOe2YEILFbbbkLUyag81SRznBAYpwrUTtgXYZnGGLG962wR3BxM03auKFPNY5RCCPN42UZkw

-------Total time for K10 deployment is 2 minutes 28 seconds.

-------Created by Yongkang
-------Email me if any suggestions or issues he@yongkang.cloud
```

3. Verify if the cluster is created with the command:
   kubectl get nodes
4. Once done, copy and paste the output link to any web browser to access K10 Web UI. Copy and paste the output token to sign in.



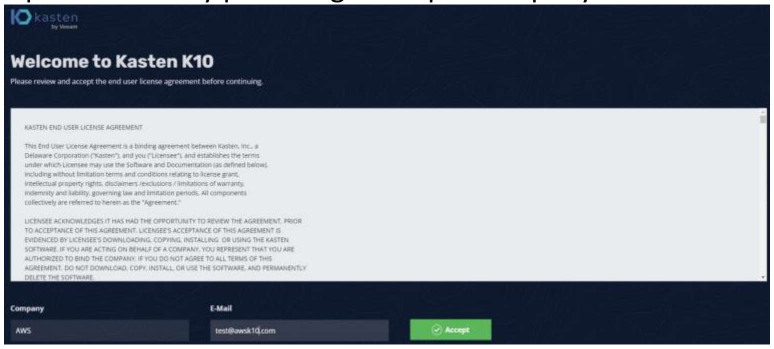
Sign In

You will need to specify a token that will be used to authenticate you with the K10 dashboard. The token will need to map to a user or a Service Account which has a binding to the cluster-admin role.
Learn more about how to obtain your authentication token.

Please copy and paste your Kubernetes bearer token below.

eyJhbGciOiJSUzI1NiIsImtpZCI6IkZTUDNFMTk0WWdsclhLeEsxbTZsaHZy

5. Review and accept the EULA by providing a sample company and E-Mail and click Accept.



Welcome to Kasten K10

Please review and accept the end user license agreement before continuing.

KASTEN END USER LICENSE AGREEMENT

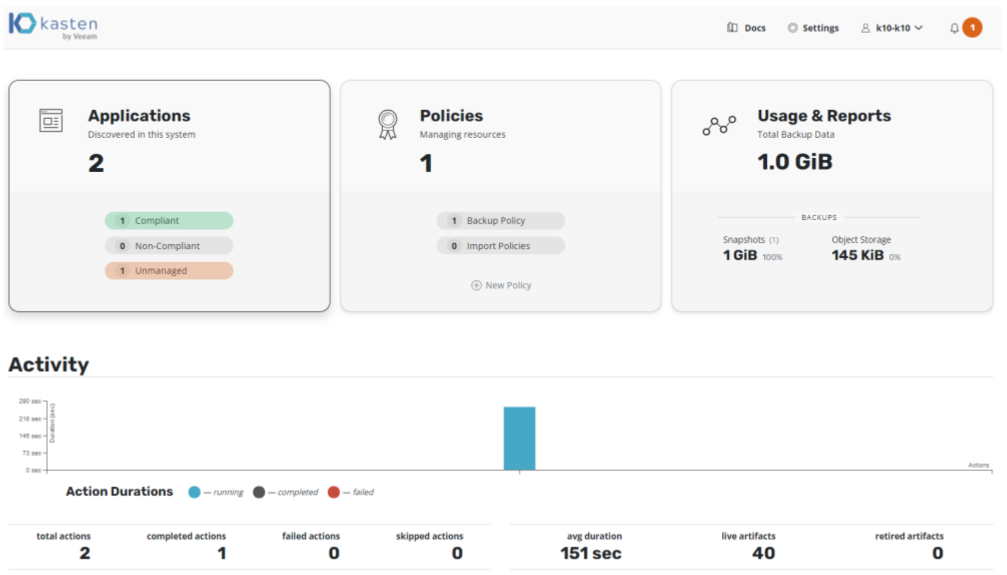
This End User License Agreement is a binding agreement between Kasten, Inc., a Delaware Corporation ("Kasten"), and you ("Licensee"), and establishes the terms under which Licensee may use the Software and Documentation (as defined below), including without limitation terms and conditions relating to license grant, intellectual property rights, disclaimers /exclusions / limitations of warranty, indemnity and liability, governing law and limitation periods. All components collectively are referred to herein as the "Agreement."

LICENSEE ACKNOWLEDGES IT HAS HAD THE OPPORTUNITY TO REVIEW THE AGREEMENT, PRIOR TO ACCEPTANCE OF THIS AGREEMENT. LICENSEE'S ACCEPTANCE OF THIS AGREEMENT IS EVIDENCED BY LICENSEE'S DOWNLOADING, COPYING, INSTALLING OR USING THE KASTEN SOFTWARE. IF YOU ARE ACTING ON BEHALF OF A COMPANY, YOU REPRESENT THAT YOU ARE AUTHORIZED TO BIND THE COMPANY. IF YOU DO NOT AGREE TO ALL TERMS OF THIS AGREEMENT, DO NOT DOWNLOAD, COPY, INSTALL, OR USE THE SOFTWARE, AND PERMANENTLY DELETE THE SOFTWARE.

Company: AWS

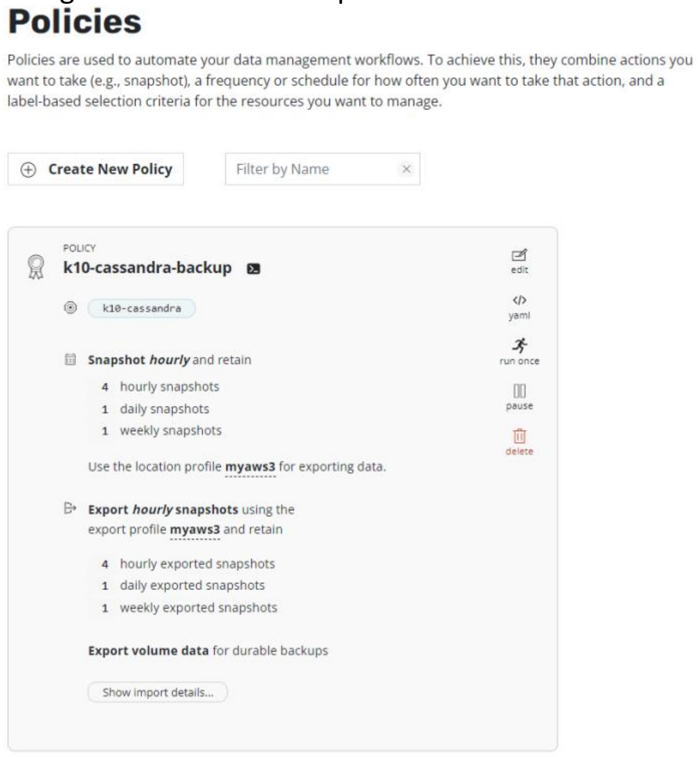
E-Mail: test@awsk10.com

6. Explore the K10 UI dashboard. Under Applications, there will be default and k10-cassandra workload which is compliant. Verify this by typing this command in CloudShell: **kubectl get ns**



total actions	completed actions	failed actions	skipped actions	avg duration	live artifacts	retired artifacts
2	1	0	0	151 sec	40	0

7. In Policies, a backup policy called k10-cassandra-backup with hourly, daily and weekly snapshots and exports to **myaws3** location profile is created. Exported snapshots data are written to S3 for long term durable backups.



Policies

Policies are used to automate your data management workflows. To achieve this, they combine actions you want to take (e.g., snapshot), a frequency or schedule for how often you want to take that action, and a label-based selection criteria for the resources you want to manage.

POLICY
k10-cassandra-backup

k10-cassandra

Snapshot ***hourly*** and retain
4 hourly snapshots
1 daily snapshots
1 weekly snapshots

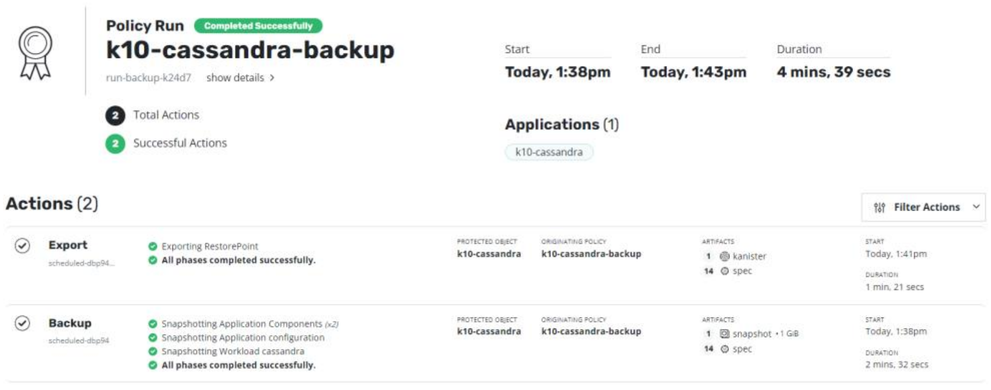
Use the location profile **myaws3** for exporting data.

Export *hourly* snapshots using the export profile **myaws3** and retain
4 hourly exported snapshots
1 daily exported snapshots
1 weekly exported snapshots

Export volume data for durable backups

Show import details...

8. From the top right corner, Click Settings, Under Locations -> Location Profiles, you will be able to see the S3 bucket where data will be exported from snapshots and the region it is residing in.
9. In actions, K10 will report Backup (snapshots) and Exports (write to S3) has been successfully completed for k10-cassandra database with the k-10cassandra-backup policy.



Policy Run Completed Successfully
k10-cassandra-backup
run-backup-k24d7 show details >

Start: Today, 1:38pm
End: Today, 1:43pm
Duration: 4 mins, 39 secs

2 Total Actions
2 Successful Actions

Applications (1)
k10-cassandra

Actions (2)

Filter Actions

Action	Status	Protected Object	Originating Policy	Artifacts	Start / Duration
Export scheduled-dbp94...	Exporting RestorePoint; All phases completed successfully.	k10-cassandra	k10-cassandra-backup	1 kanister; 14 spec	Today, 1:41pm; 1 min, 21 secs
Backup scheduled-dbp94	Snapshotting Application Components (x2); Snapshotting Application configuration; Snapshotting Workload cassandra; All phases completed successfully.	k10-cassandra	k10-cassandra-backup	1 snapshot • 1 GiB; 14 spec	Today, 1:38pm; 2 mins, 32 secs

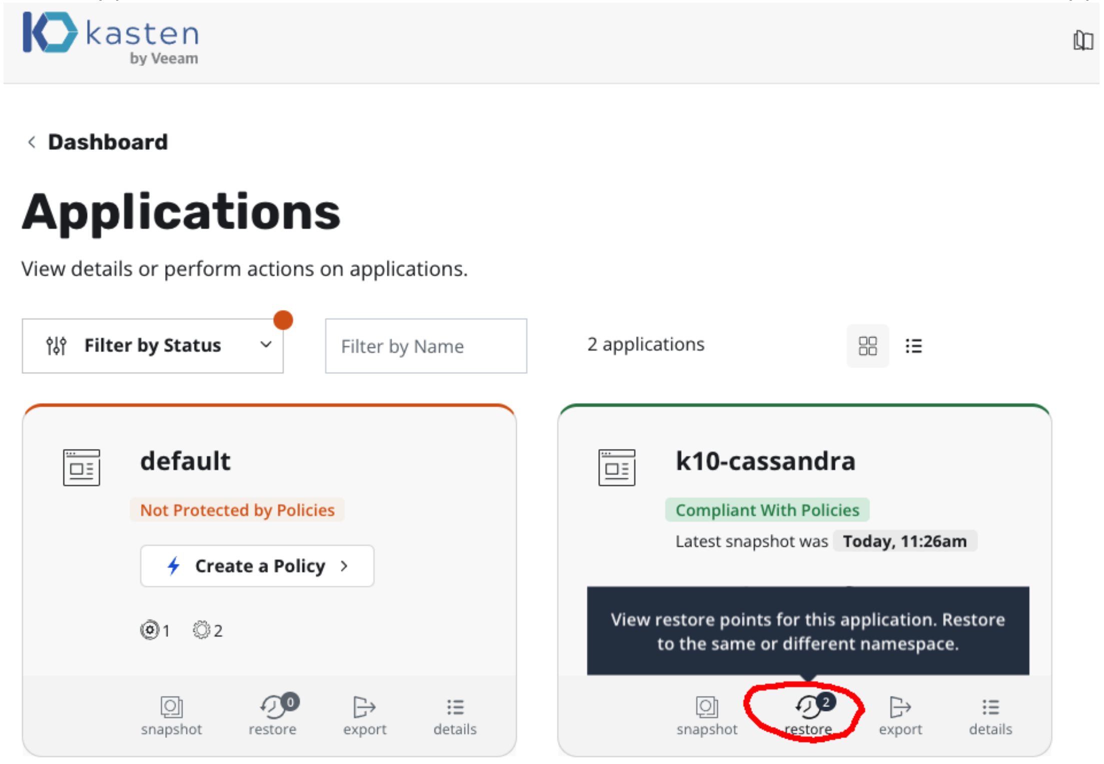
Awesome Kasten K10 Restore Capabilities

- Granular recovery of individual spec artifacts
- Full Restore in-place
- DR restore from S3 bucket
- Restore to a new namespace

Granular recovery of individual spec artifacts

Someone accidentally deleted a secrets, configmaps or any other Kubernetes configurations, the customers just want to restore the individual spec artifacts instead of restoring everything. Why? It is more efficient and can bring up my apps back online as quickly as possible. In this case, let’s do a granular recovery of the individual spec artifacts.

1. Click Applications from K10 Web Console, then click “restore” in k10-cassandra app



Applications

View details or perform actions on applications.

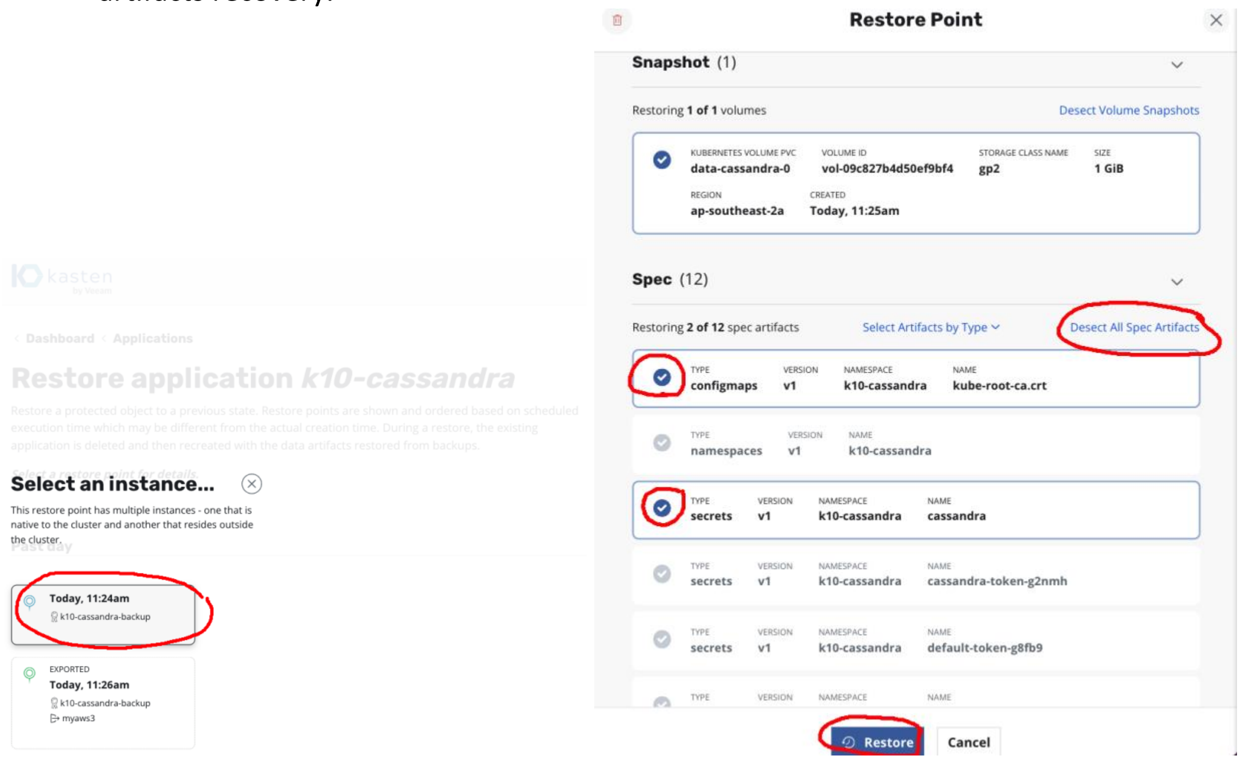
2 applications

default — Not Protected by Policies — 1, 2

k10-cassandra — Compliant With Policies — Latest snapshot was Today, 11:26am

View restore points for this application. Restore to the same or different namespace.

2. Select the first restore point, leave the rest of settings as default, scroll down to the section of **Spec**, Click "Deselect All Spec Artifacts", then only select the individual spec artifacts (e.g. configmaps, secrets) before clicking "Restore" to kickoff the individual spec artifacts recovery.



Dashboard < Applications

Restore application *k10-cassandra*

Restore a protected object to a previous state. Restore points are shown and ordered based on scheduled execution time which may be different from the actual creation time. During a restore, the existing application is deleted and then recreated with the data artifacts restored from backups.

Select a restore point for details

Select an instance...

This restore point has multiple instances - one that is native to the cluster and another that resides outside the cluster.

Past day

Today, 11:24am
k10-cassandra-backup

EXPORTED
Today, 11:26am
k10-cassandra-backup
myaws3

Restore Point

Snapshot (1)

Restoring **1 of 1** volumes — Desect Volume Snapshots

KUBERNETES VOLUME PVC	VOLUME ID	STORAGE CLASS NAME	SIZE
data-cassandra-0	vol-09c827b4d50ef9bf4	gp2	1 GiB

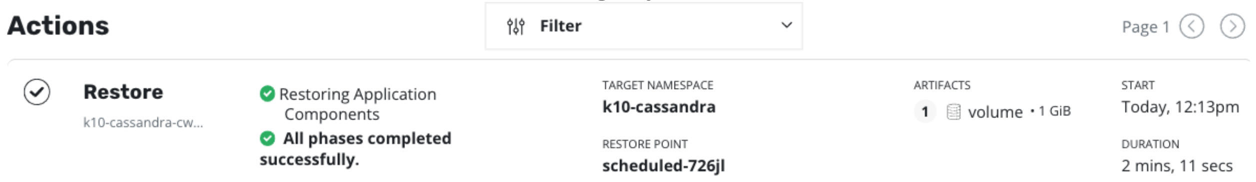
REGION	CREATED
ap-southeast-2a	Today, 11:25am

Spec (12)

Restoring **2 of 12** spec artifacts — Select Artifacts by Type — Desect All Spec Artifacts

	TYPE	VERSION	NAMESPACE	NAME
✓	configmaps	v1	k10-cassandra	kube-root-ca.crt
	namespaces	v1		k10-cassandra
✓	secrets	v1	k10-cassandra	cassandra
	secrets	v1	k10-cassandra	cassandra-token-g2nmh
	secrets	v1	k10-cassandra	default-token-g8fb9

3. Monitor the restore process under Actions in the dashboard or type the following command in AWS CloudShell: **kubectl get po -n k10-cassandra -w**



Actions — Filter — Page 1

Restore
k10-cassandra-cw...

Restoring Application Components
All phases completed successfully.

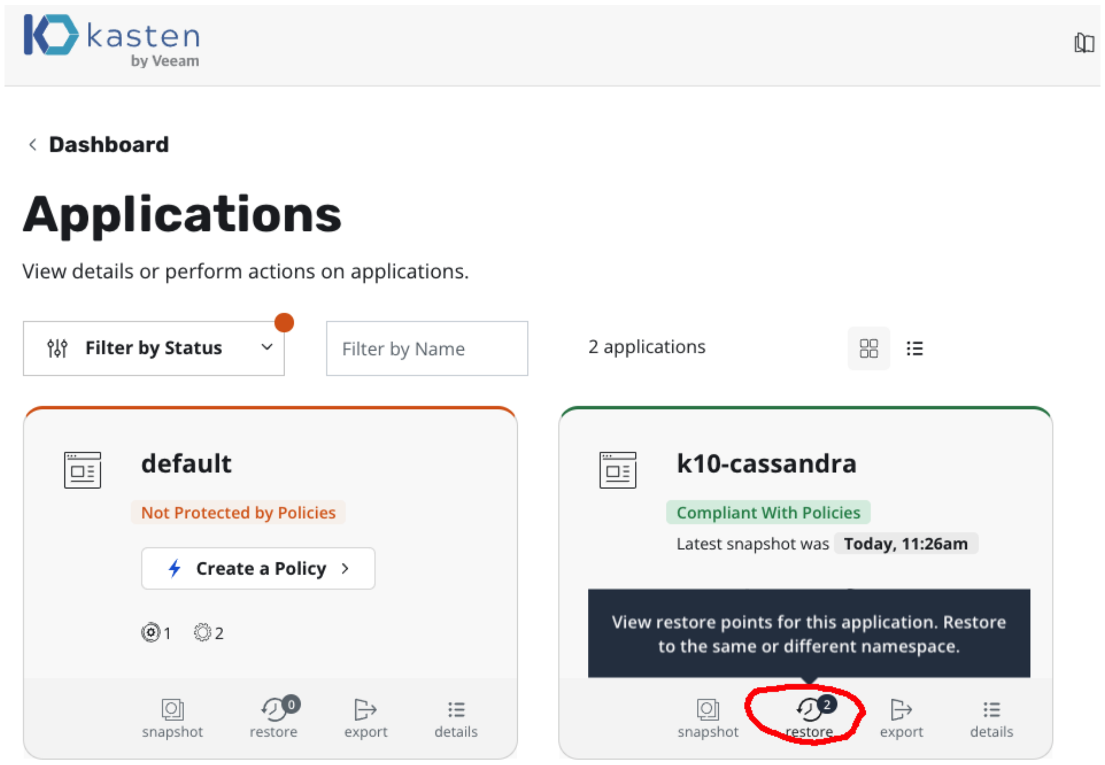
TARGET NAMESPACE: **k10-cassandra**
RESTORE POINT: **scheduled-726jl**
ARTIFACTS: 1 volume • 1 GiB
START: Today, 12:13pm
DURATION: 2 mins, 11 secs

4. k10-cassandra namespace should be restored and be in ready state in ~ 2 minutes.

Full Restore in-place

I know my containers are broken in my original namespace, I need my applications are back running normal as quickly as I can. Let's do a full restore in place.

1. Click Applications from K10 Web Console, then click "restore" in **k10-cassandra** app



Dashboard

Applications

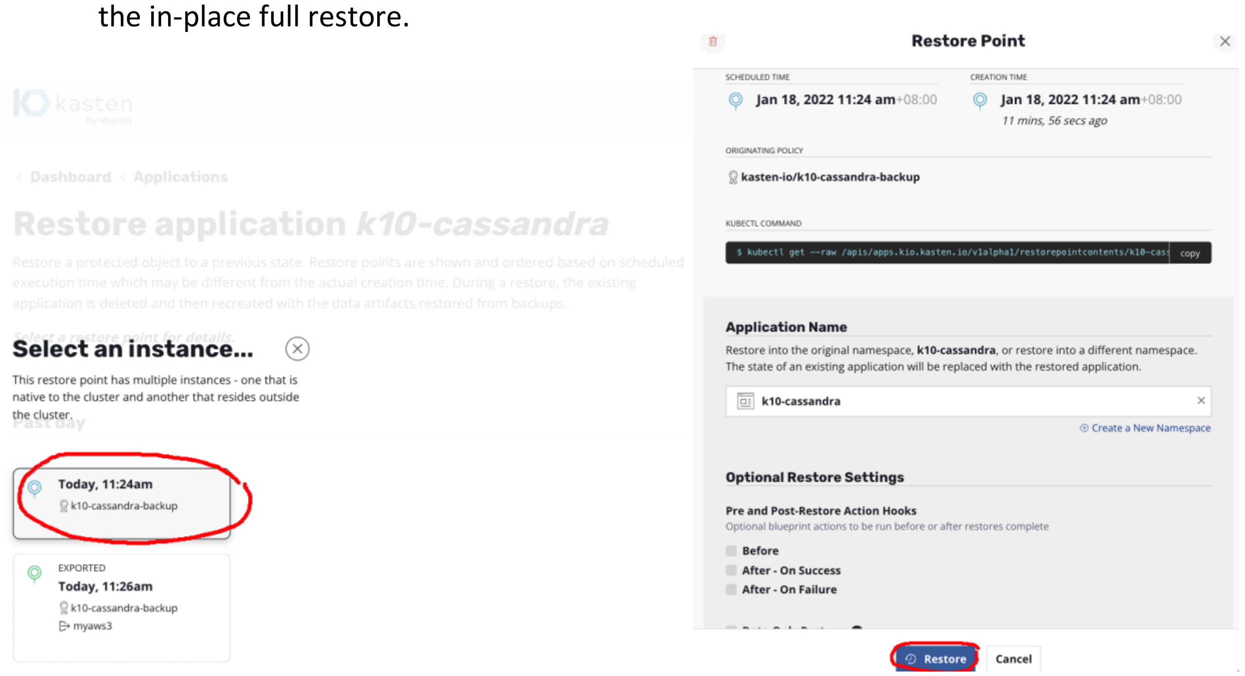
View details or perform actions on applications.

default
Not Protected by Policies

k10-cassandra
Compliant With Policies
Latest snapshot was Today, 11:26am

View restore points for this application. Restore to the same or different namespace.

2. Select the first restore point, leave the rest of settings as default, click retore to kickoff the in-place full restore.



Dashboard < Applications

Restore application *k10-cassandra*

Restore a protected object to a previous state. Restore points are shown and ordered based on scheduled execution time which may be different from the actual creation time. During a restore, the existing application is deleted and then recreated with the data artifacts restored from backups.

Select an instance...

This restore point has multiple instances - one that is native to the cluster and another that resides outside the cluster.

Today, 11:24am
k10-cassandra-backup

EXPORTED
Today, 11:26am
k10-cassandra-backup
myaws3

Restore Point

SCHEDULED TIME
Jan 18, 2022 11:24 am+08:00

CREATION TIME
Jan 18, 2022 11:24 am+08:00
11 mins, 56 secs ago

ORIGINATING POLICY
kasten-io/k10-cassandra-backup

KUBECTL COMMAND

```
$ kubectl get --raw /apis/apps.kio.kasten.io/v1alpha1/restorepointcontents/k10-cas
```

Application Name

Restore into the original namespace, **k10-cassandra**, or restore into a different namespace. The state of an existing application will be replaced with the restored application.

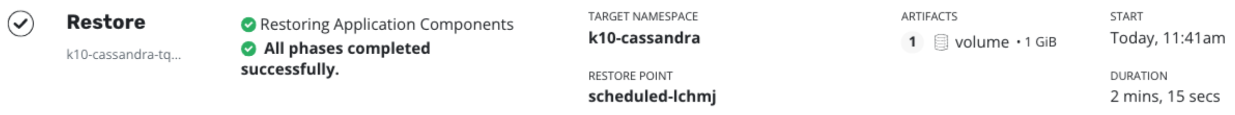
k10-cassandra

Optional Restore Settings

Pre and Post-Restore Action Hooks
Optional blueprint actions to be run before or after restores complete

- Before
- After - On Success
- After - On Failure

3. Monitor the restore process under Actions in the dashboard or type the following command in AWS CloudShell: **kubectl get po -n k10-cassandra -w**



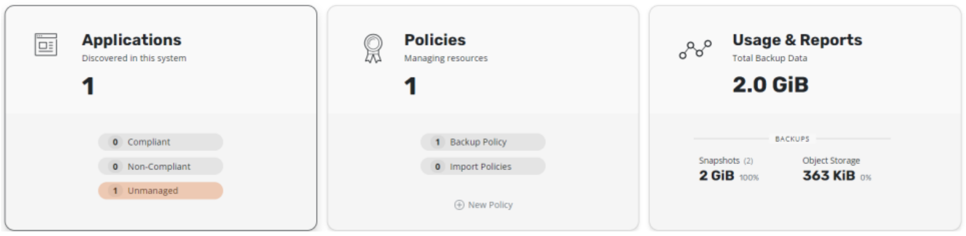
Restore		TARGET NAMESPACE	ARTIFACTS	START
k10-cassandra-tq...	Restoring Application Components **All phases completed successfully.**	k10-cassandra	1 volume • 1 GiB	Today, 11:41am
		RESTORE POINT scheduled-lchmj		DURATION 2 mins, 15 secs

4. k10-cassandra namespace should be terminated first and then restored and be in ready state in ~ 2 minutes.

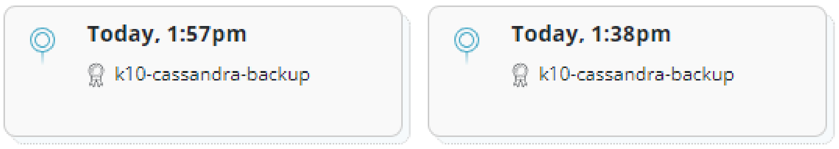
DR Restore from S3 bucket

The lab will now simulate accidental deletion of the Cassandra DB and we will attempt recovery with K10 solution.

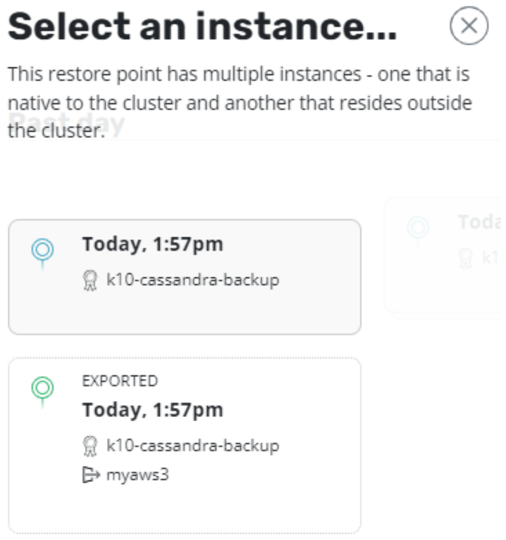
1. Return to AWS CloudShell and run the command to delete k10-cassandra database: **kubectl delete ns k10-cassandra --force**
2. Double check deletion by refreshing K10 Web UI and running this command: **kubectl get ns**



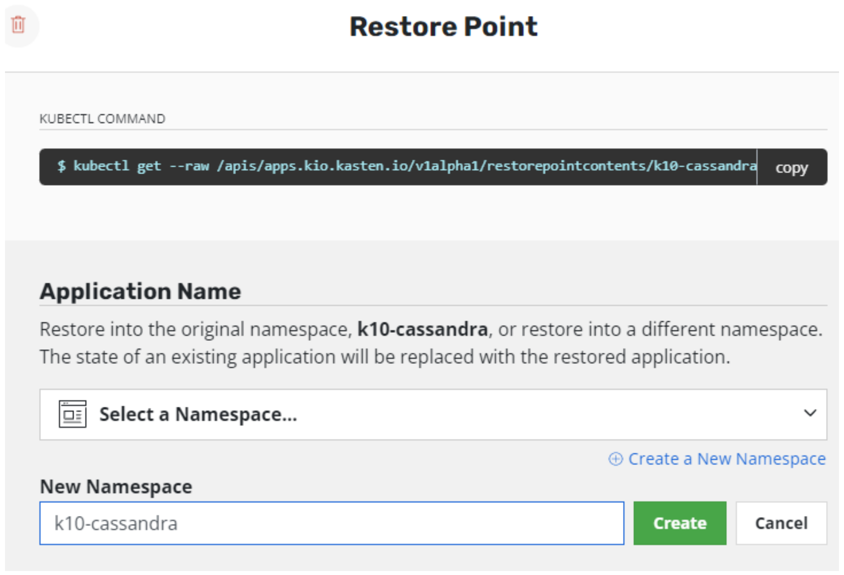
3. Now that we have confirmed deletion, navigate to K10 Web UI > Applications. In Filter by Status, select Removed. Select the k10-cassandra application and click restore and you will be able to see multiple recovery points. Select the latest one.



4. Once the recovery point is selected, you will see 2 instance options restore points. One will be native to cluster as snapshots and another resides outside of cluster as exported data to S3. Since my original namespace deleted including the snapshots on container storage, we will now recover from S3 bucket. Select EXPORTED restore point.


Select an instance...
This restore point has multiple instances - one that is native to the cluster and another that resides outside the cluster.
Today, 1:57pm
k10-cassandra-backup
EXPORTED
Today, 1:57pm
k10-cassandra-backup
myaws3

5. Once the restore point is selected, we can choose to restore to an existing namespace or a new namespace. Click “Create A New Namespace” and type **k10-cassandra**, then Click “Create”, followed by Clicking “Restore” and confirm restore. Leave everything else as default. We will restore it to the namespace you specified.


Restore Point

KUBECTL COMMAND

```
$ kubectl get --raw /apis/apps.kio.kasten.io/v1alpha1/restorepointcontents/k10-cassandra
```

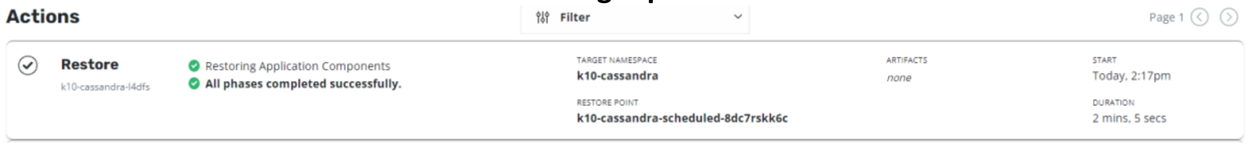
Application Name

Restore into the original namespace, **k10-cassandra**, or restore into a different namespace. The state of an existing application will be replaced with the restored application.

Select a Namespace...

New Namespace: k10-cassandra

6. Monitor the restore process under Actions in the dashboard or type the following command in AWS CloudShell: **kubectl get po -n k10-cassandra -w**


Actions

Restore — k10-cassandra-l4dfs

Restoring Application Components
All phases completed successfully.

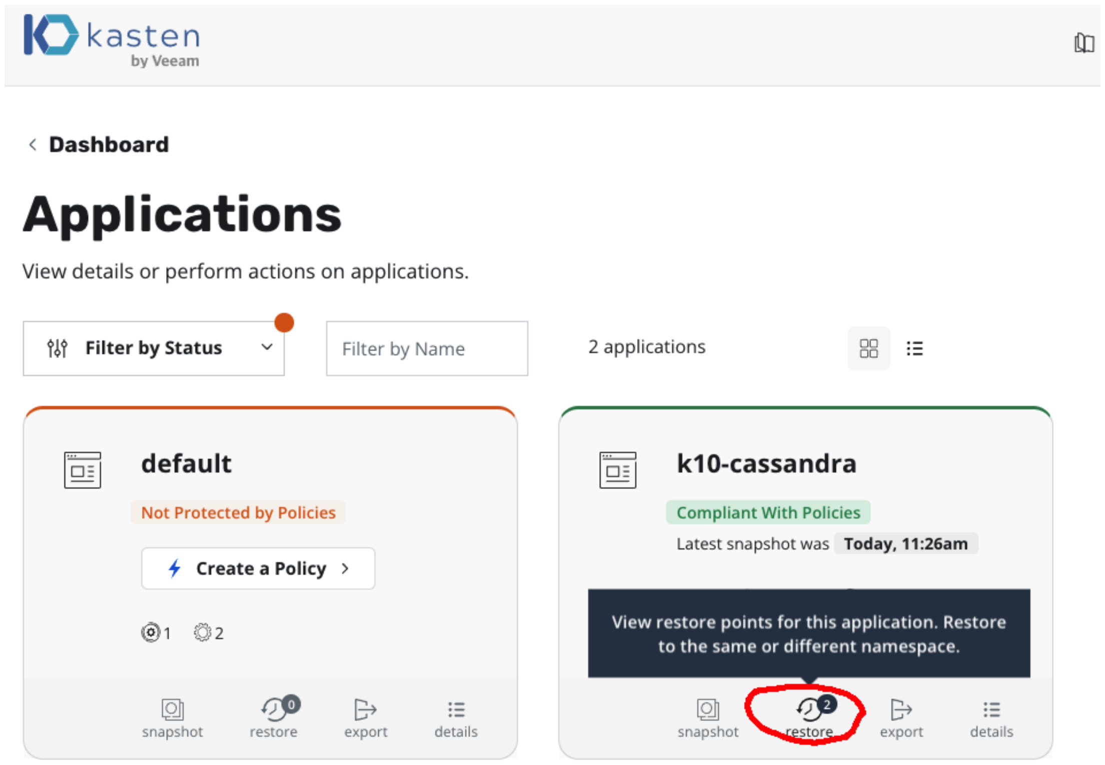
TARGET NAMESPACE: k10-cassandra
ARTIFACTS: none
START: Today, 2:17pm
RESTORE POINT: k10-cassandra-scheduled-8dc7rskk6c
DURATION: 2 mins, 5 secs

7. k10-cassandra namespace should be restored and be in ready state in approximately 3 minutes.

Restore to a new namespace

In other situations, the customers just want to clone the apps from your existing running containers. In this case, let’s do a full restore to a new namespace.

1. Click Applications from K10 Web Console, then click “restore” in k10-cassandra app



Applications

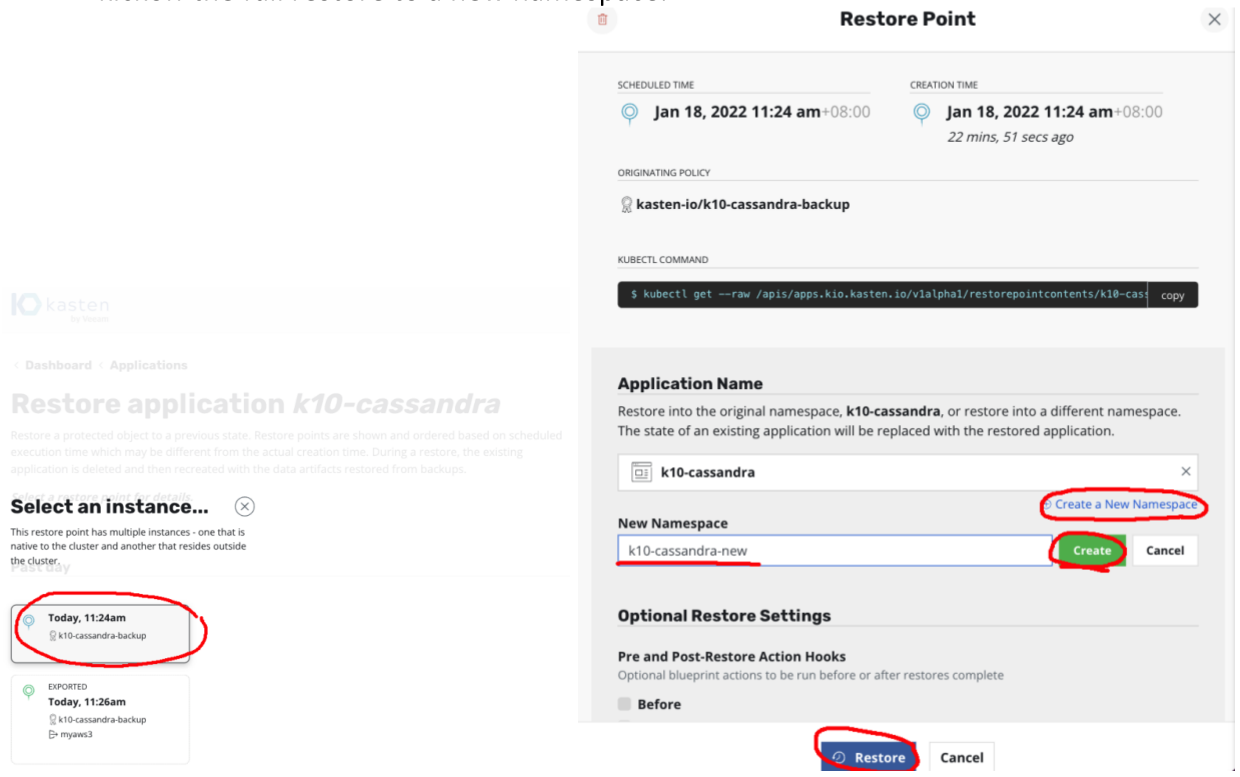
View details or perform actions on applications.

default
Not Protected by Policies

k10-cassandra
Compliant With Policies
Latest snapshot was Today, 11:26am

View restore points for this application. Restore to the same or different namespace.

2. Select the first restore point, leave the rest of settings as default, click "Create a New Namespace" to create a new namespace **k10-cassandra-new** before clicking "Restore" to kickoff the full restore to a new namespace.



Restore application *k10-cassandra*

Restore a protected object to a previous state. Restore points are shown and ordered based on scheduled execution time which may be different from the actual creation time. During a restore, the existing application is deleted and then recreated with the data artifacts restored from backups.

Select a restore point for details.

Select an instance...

This restore point has multiple instances - one that is native to the cluster and another that resides outside the cluster.

Past day

Today, 11:24am
k10-cassandra-backup

EXPORTED
Today, 11:26am
k10-cassandra-backup
myaws3

Restore Point

SCHEDULED TIME
Jan 18, 2022 11:24 am+08:00

CREATION TIME
Jan 18, 2022 11:24 am+08:00
22 mins, 51 secs ago

ORIGINATING POLICY
kasten-io/k10-cassandra-backup

KUBECTL COMMAND

```
$ kubectl get --raw /apis/apps.kio.kasten.io/v1alpha1/restorepointcontents/k10-cas
```

Application Name

Restore into the original namespace, **k10-cassandra**, or restore into a different namespace. The state of an existing application will be replaced with the restored application.

k10-cassandra

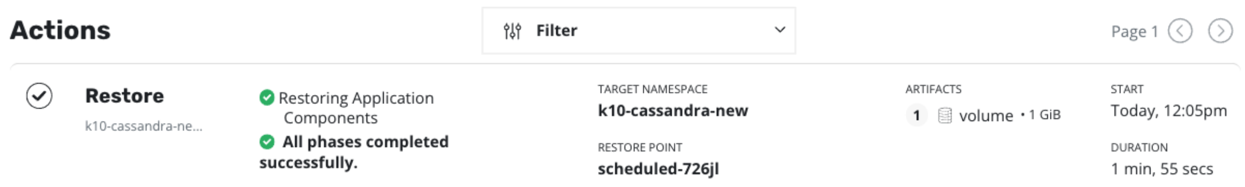
New Namespace
k10-cassandra-new

Optional Restore Settings

Pre and Post-Restore Action Hooks
Optional blueprint actions to be run before or after restores complete

Before

3. Monitor the restore process under Actions in the dashboard or type the following command in AWS CloudShell: **kubectl get po -n k10-cassandra-new -w**


Actions

Filter

Page 1

Restore
k10-cassandra-ne...

Restoring Application Components
All phases completed successfully.

TARGET NAMESPACE
k10-cassandra-new

RESTORE POINT
scheduled-726jl

ARTIFACTS
1 volume • 1 GiB

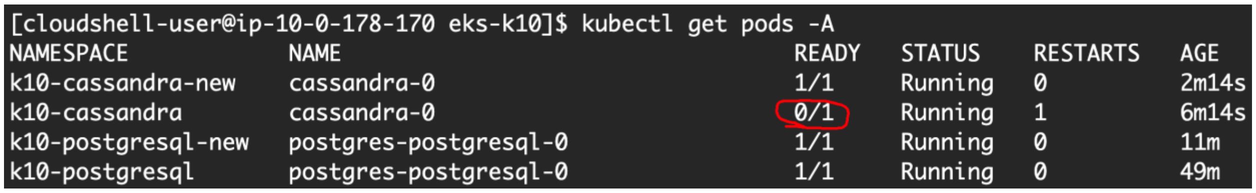
START
Today, 12:05pm

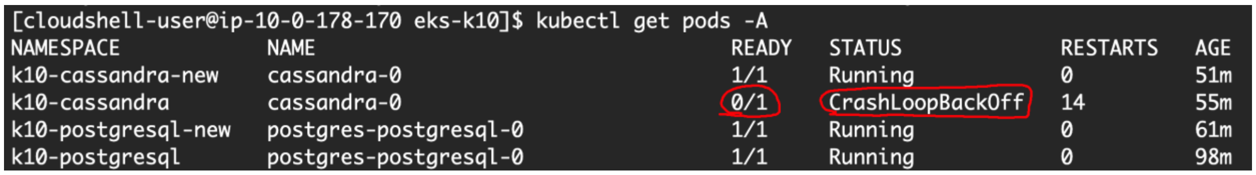
DURATION
1 min, 55 secs

4. k10-cassandra-new namespace should be restored and be in ready state in ~ 2 minutes.

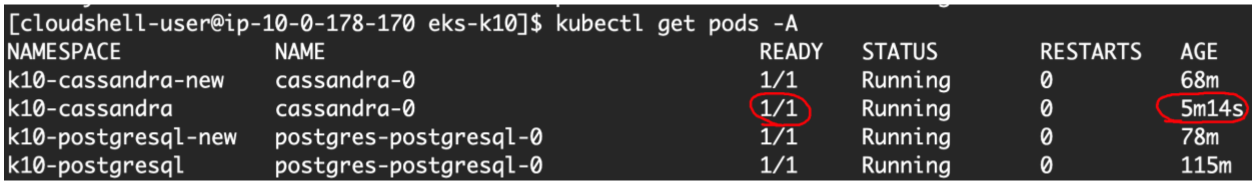
Known Issues for Cassandra

When restoring to a new namespace for Cassandra without deleting the original namespace, you're expecting to see below behaviours. The original namespace will become Running but not Ready or failed with Status "CrashLoopBackOff".


```
[cloudshell-user@ip-10-0-178-170 eks-k10]$ kubectl get pods -A
NAMESPACE           NAME                                  READY   STATUS    RESTARTS   AGE
k10-cassandra-new   cassandra-0                           1/1     Running   0          2m14s
k10-cassandra       cassandra-0                           0/1     Running   1          6m14s
k10-postgresql-new  postgres-postgresql-0                 1/1     Running   0          11m
k10-postgresql      postgres-postgresql-0                 1/1     Running   0          49m
```


```
[cloudshell-user@ip-10-0-178-170 eks-k10]$ kubectl get pods -A
NAMESPACE           NAME                                  READY   STATUS             RESTARTS   AGE
k10-cassandra-new   cassandra-0                           1/1     Running            0          51m
k10-cassandra       cassandra-0                           0/1     CrashLoopBackOff   14         55m
k10-postgresql-new  postgres-postgresql-0                 1/1     Running            0          61m
k10-postgresql      postgres-postgresql-0                 1/1     Running            0          98m
```

This is NOT an issue from Kasten. As you can see from below screenshots, for Postgresql, there is no such issues. To clear the Cassandra errors, you can simply practice another full restore in-place by restoring k10-cassandra. Below is the screenshot after the full restore in-place.


```
[cloudshell-user@ip-10-0-178-170 eks-k10]$ kubectl get pods -A
NAMESPACE           NAME                                  READY   STATUS    RESTARTS   AGE
k10-cassandra-new   cassandra-0                           1/1     Running   0          68m
k10-cassandra       cassandra-0                           1/1     Running   0          5m14s
k10-postgresql-new  postgres-postgresql-0                 1/1     Running   0          78m
k10-postgresql      postgres-postgresql-0                 1/1     Running   0          115m
```

Additional Labs (step by step manually)

If you are interested to learn step by step how to complete below tasks manually, here is the instructions.

- Deploy a Postgresql database
- Create a backup policy
- Create a S3 location profile
- Run the backup of Postgresql
- Do an in-place full restore

1. Deploy a Postgresql database from CloudShell

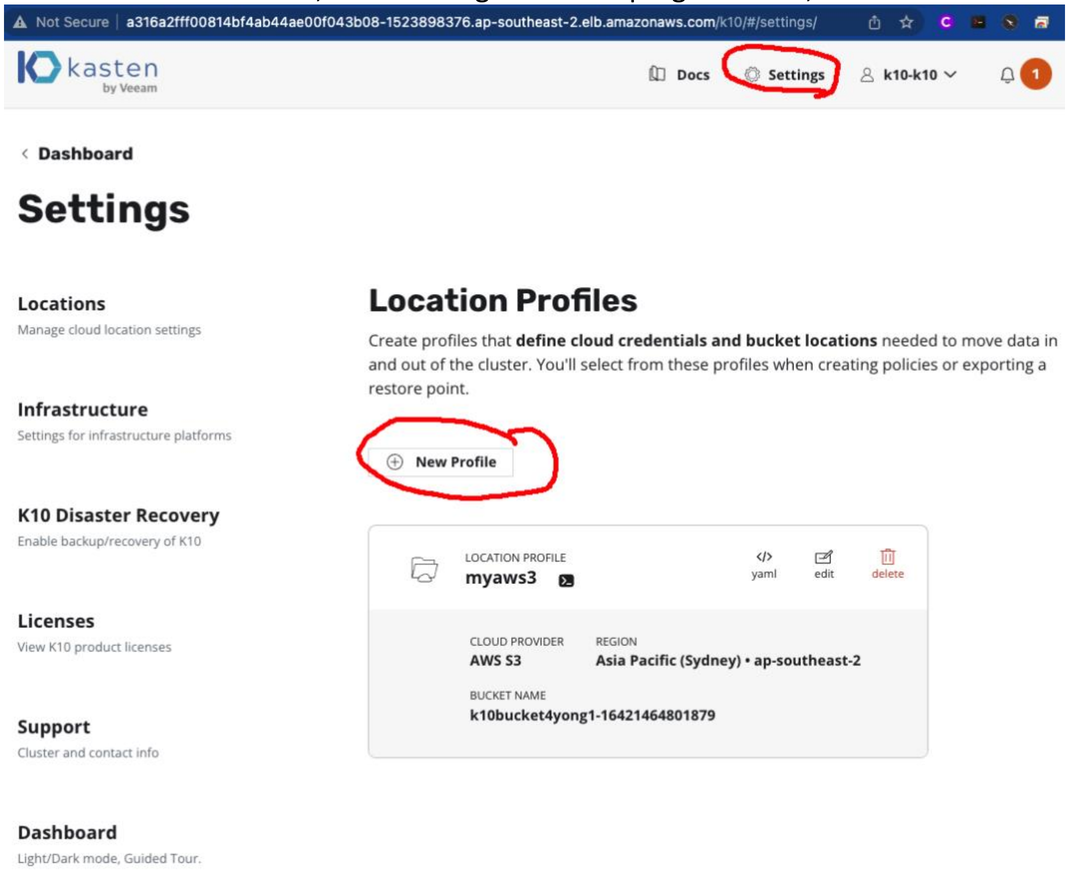
```
kubectl create namespace k10-postgresql
```

```
helm repo add bitnami https://charts.bitnami.com/bitnami
helm repo update
helm install --namespace k10-postgresql postgres bitnami/postgresql --set persistence.size=1Gi
```

Monitor the pod status via the command below, press Ctrl-C once it is running and ready.
kubectl get po -n k10-postgresql -w

```
[cloudshell-user@ip-10-0-180-78 eks-k10]$ kubectl get po -n k10-postgresql -w
NAME                    READY   STATUS    RESTARTS   AGE
postgres-postgresql-0   1/1     Running   0          51s
```

2. Create a S3 location profile from Kasten Web UI
   From Kasten Dashboard, Click Settings on the top right corner, then Click "New Profile"


Not Secure
a316a2fff00814bf4ab44ae00f043b08-1523898376.ap-southeast-2.elb.amazonaws.com/k10/#/settings/
kasten by Veeam
Docs
Settings
k10-k10
1
Dashboard
Settings
Locations
Manage cloud location settings
Infrastructure
Settings for infrastructure platforms
K10 Disaster Recovery
Enable backup/recovery of K10
Licenses
View K10 product licenses
Support
Cluster and contact info
Dashboard
Light/Dark mode, Guided Tour.
Location Profiles
New Profile
yaml
edit
delete

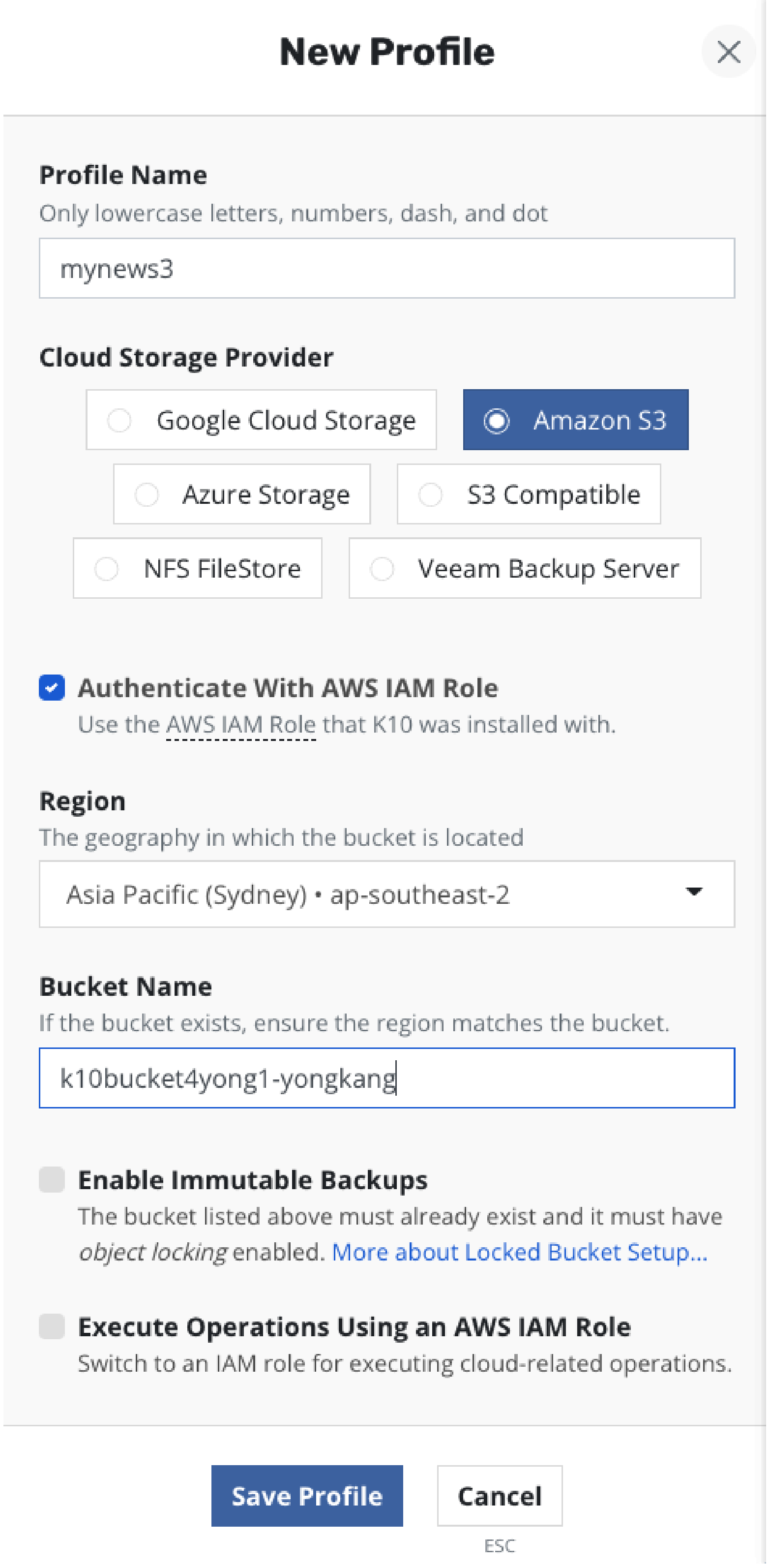
Create profiles that **define cloud credentials and bucket locations** needed to move data in and out of the cluster. You'll select from these profiles when creating policies or exporting a restore point.

LOCATION PROFILE: myaws3
CLOUD PROVIDER: AWS S3
REGION: Asia Pacific (Sydney) • ap-southeast-2
BUCKET NAME: k10bucket4yong1-16421464801879

Under New Profile window, fill out all the details including Profile Name e.g. **mynews3**, make sure to choose the same region since we only have the access to that particular region in this lab and provide a globally unique Bucket Name. In this Event Engine lab platform, we will tick the box "Authenticate With AWS IAM Role" since we are using IAM role instead of the AWS Access key and Secret. In most of real situations, you might need the AWS Access Key and Secret to create the Location Profile.

NOTE:

Change the Bucket Name from k10bucket4yong1-yongkang to a globally unique name or append timestamp to the end. E.g. ***k10bucket4yong1-20220222104805***



New Profile

Profile Name
Only lowercase letters, numbers, dash, and dot
mynews3

Cloud Storage Provider
Google Cloud Storage [✓ Amazon S3] Azure Storage S3 Compatible NFS FileStore Veeam Backup Server

[✓] Authenticate With AWS IAM Role
Use the AWS IAM Role that K10 was installed with.

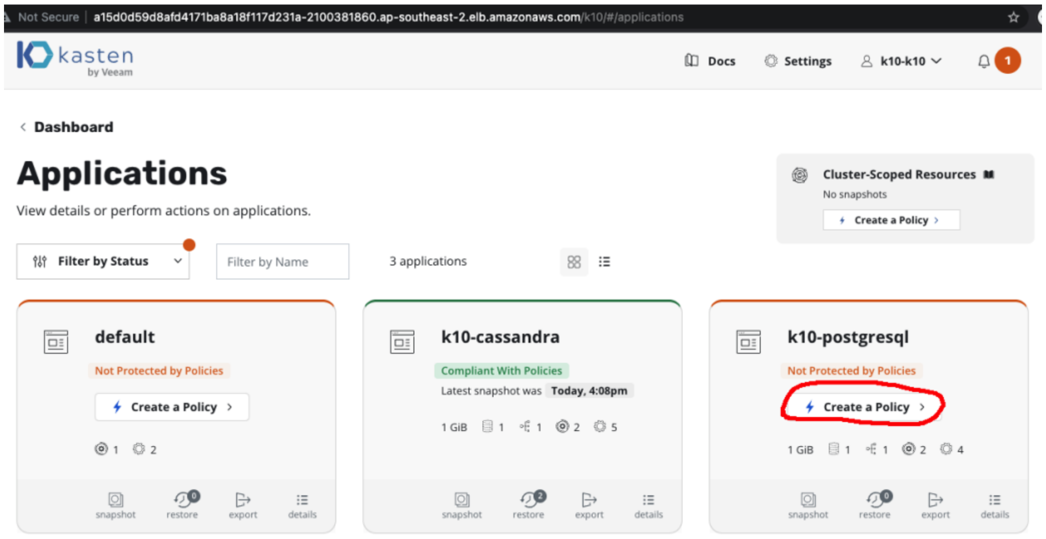
Region
The geography in which the bucket is located
Asia Pacific (Sydney) • ap-southeast-2

Bucket Name
If the bucket exists, ensure the region matches the bucket.
k10bucket4yong1-yongkang

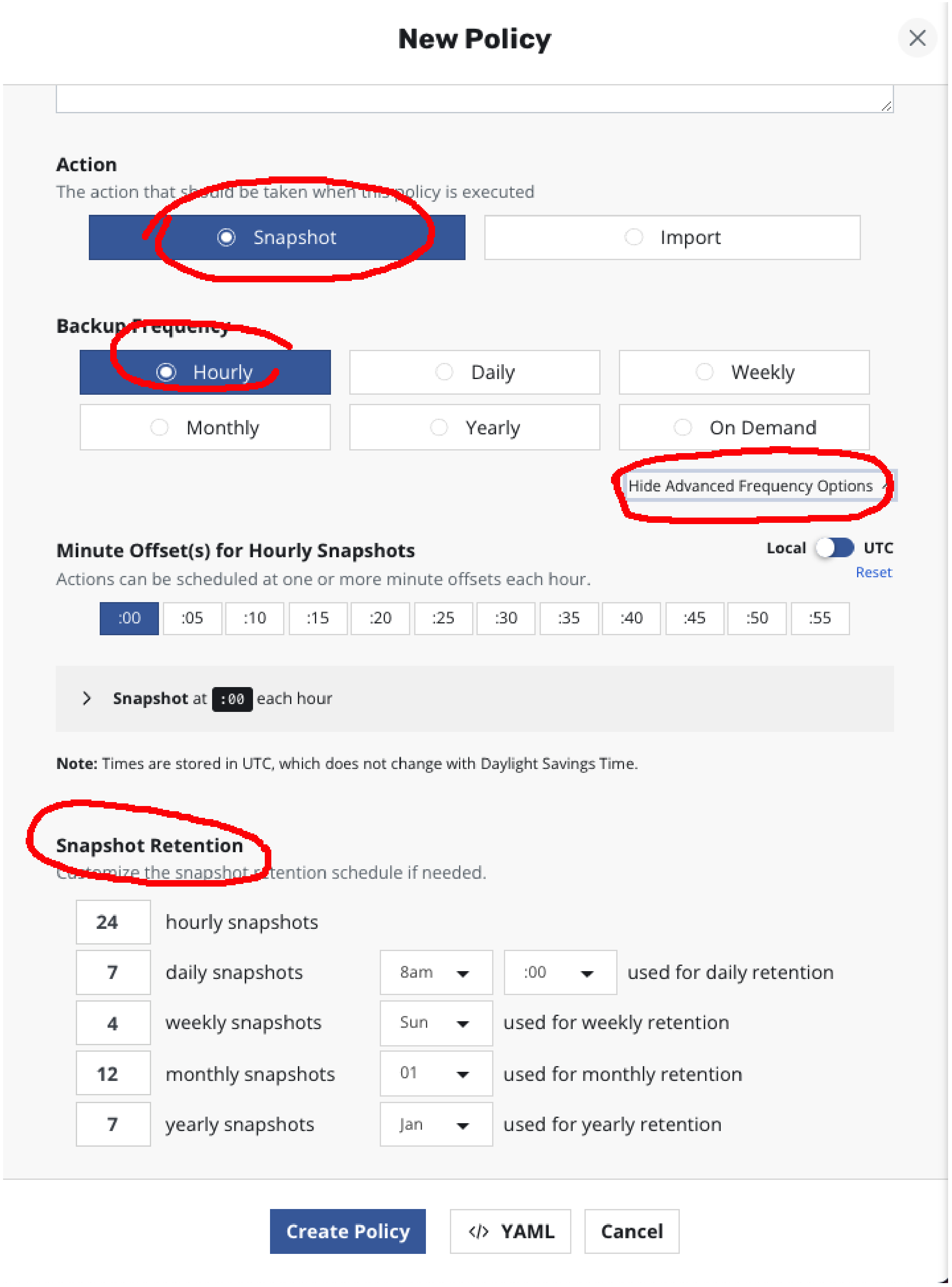
[ ] Enable Immutable Backups
The bucket listed above must already exist and it must have *object locking* enabled. More about Locked Bucket Setup...

[ ] Execute Operations Using an AWS IAM Role
Switch to an IAM role for executing cloud-related operations.

3. Create a backup policy from Kasten Web UI
   From Kasten Dashboard, Click Applications, Under 10-postgresql, Click "Create a Policy"



Review the details in the New Policy window, you can select different Backup Frequency (Hourly, Daily, Weekly, Monthly, Yearly, On Demand). If you click "Show Advanced Frequency Options", you will notice Kasten allow you to run the backups every 5 minutes from the UI. You can customize the Snapshot Retention.


New Policy

Action

The action that should be taken when this policy is executed

[✓ Snapshot] Import

Backup Frequency

[✓ Hourly] Daily Weekly Monthly Yearly On Demand

Hide Advanced Frequency Options

Minute Offset(s) for Hourly Snapshots

Actions can be scheduled at one or more minute offsets each hour.

Local / UTC Reset

[✓ :00] :05 :10 :15 :20 :25 :30 :35 :40 :45 :50 :55

Snapshot at :00 each hour

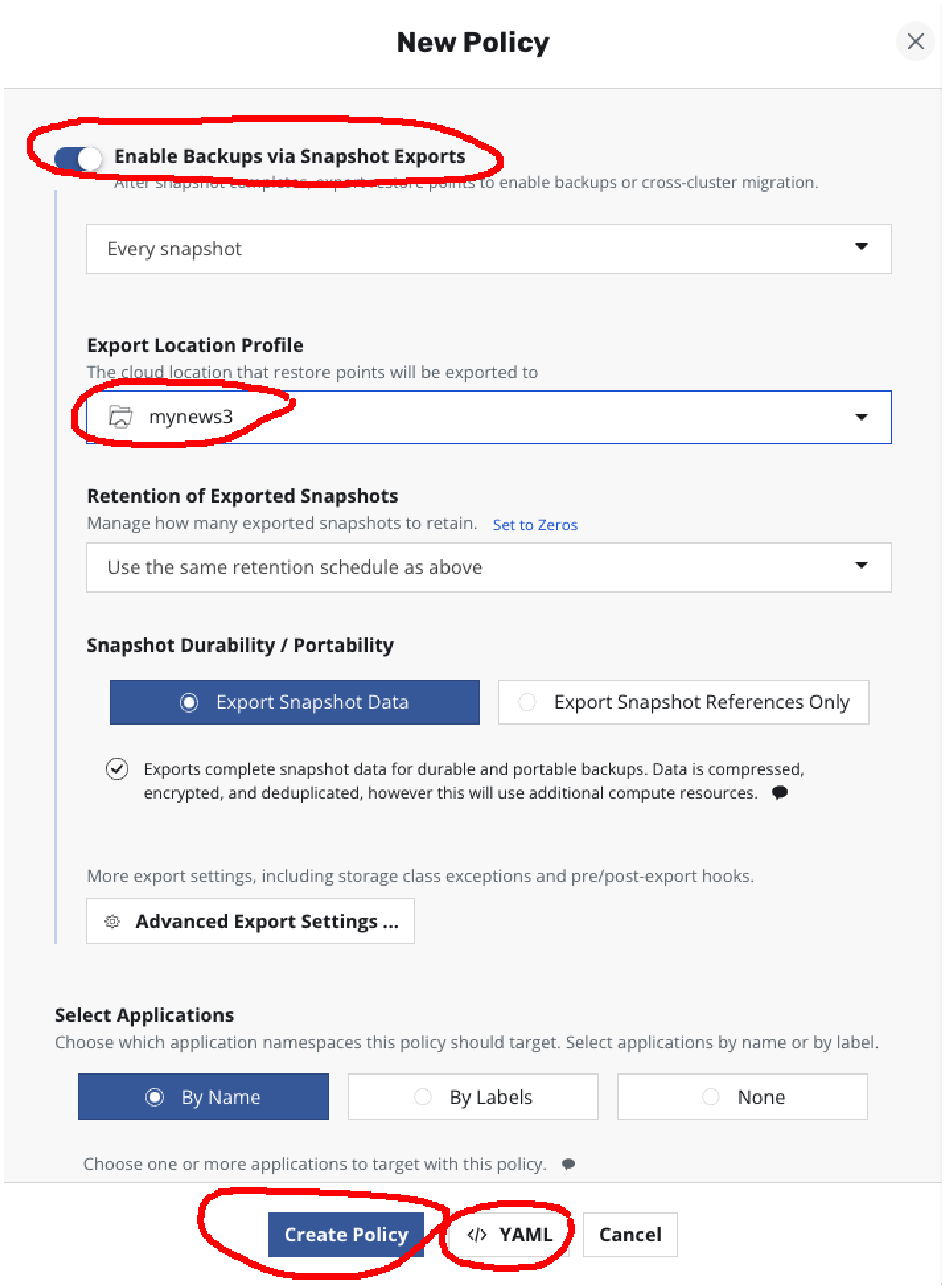
Note: Times are stored in UTC, which does not change with Daylight Savings Time.

Snapshot Retention

Customize the snapshot retention schedule if needed.

24	hourly snapshots			
7	daily snapshots	8am	:00	used for daily retention
4	weekly snapshots	Sun	used for weekly retention	
12	monthly snapshots	01	used for monthly retention	
7	yearly snapshots	Jan	used for yearly retention	

As a best practice, we should always enable “Enable Backups via Snapshot Exports” to export the snapshots from your cluster to a S3 bucket which can be in a different region.



New Policy

Enable Backups via Snapshot Exports
After snapshot completes, export restore points to enable backups or cross-cluster migration.

Every snapshot

Export Location Profile
The cloud location that restore points will be exported to

mynews3

Retention of Exported Snapshots
Manage how many exported snapshots to retain. Set to Zeros

Use the same retention schedule as above

Snapshot Durability / Portability

[✓ Export Snapshot Data] Export Snapshot References Only

Exports complete snapshot data for durable and portable backups. Data is compressed, encrypted, and deduplicated, however this will use additional compute resources.

More export settings, including storage class exceptions and pre/post-export hooks.

Advanced Export Settings ...

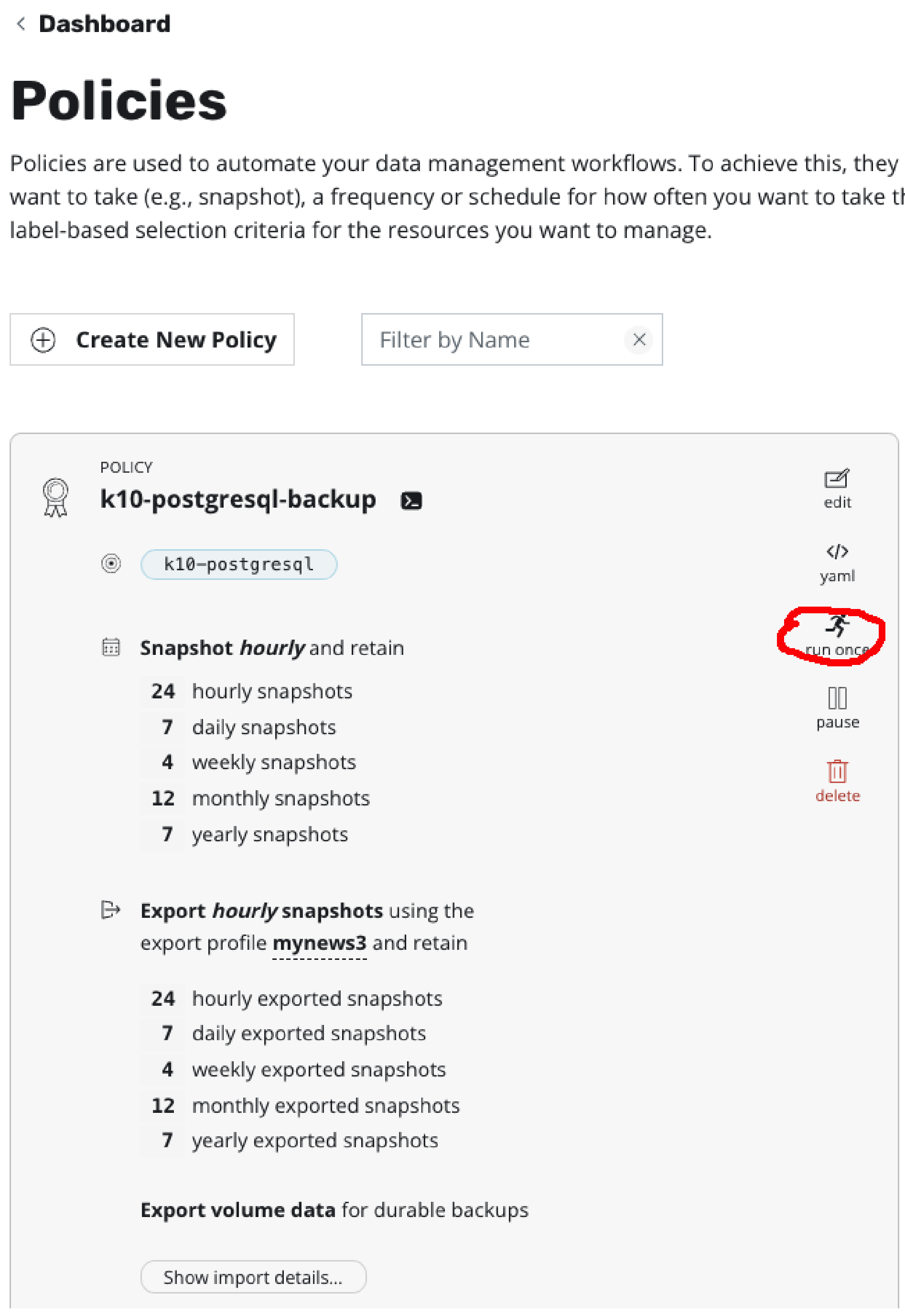
Select Applications
Choose which application namespaces this policy should target. Select applications by name or by label.

[✓ By Name] By Labels None

Choose one or more applications to target with this policy.

You can leave the rest of settings as it is. Before Click "Create Policy", Click "</> YAML" to review the yaml file which can be generated from the web console. You may also explore the rest of settings before creating the policy.

Now, the backup policy created, instead of waiting for next scheduled job, you can click "run once" to kick off the backup job immediately.


< Dashboard

Policies

Policies are used to automate your data management workflows. To achieve this, they c
want to take (e.g., snapshot), a frequency or schedule for how often you want to take th
label-based selection criteria for the resources you want to manage.

POLICY
k10-postgresql-backup

k10-postgresql

Snapshot *hourly* and retain

24	hourly snapshots
7	daily snapshots
4	weekly snapshots
12	monthly snapshots
7	yearly snapshots

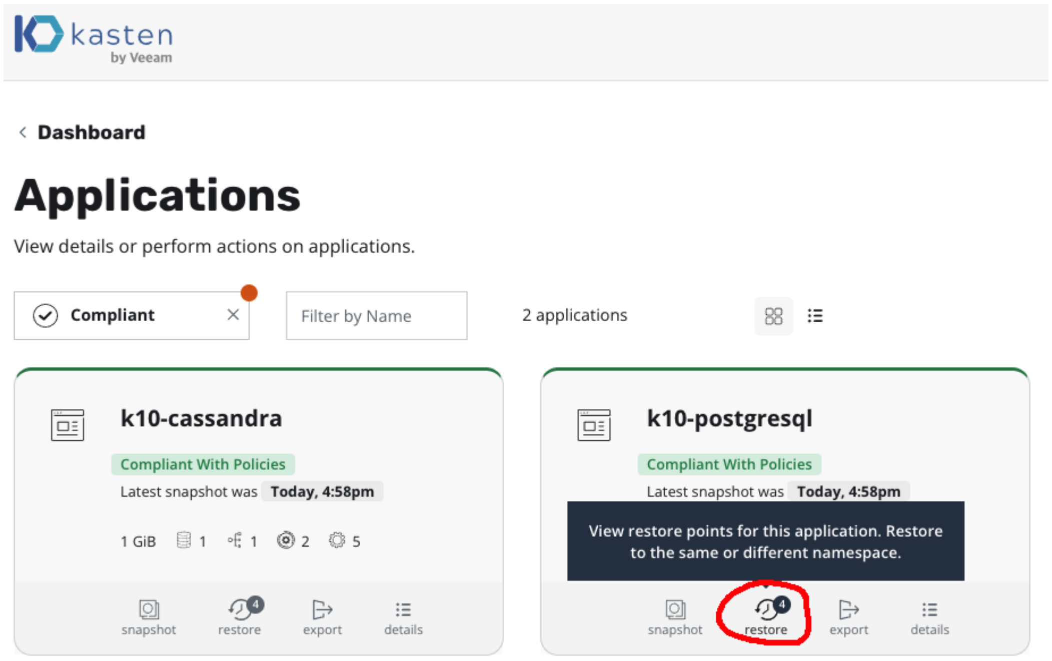
Export *hourly* snapshots using the export profile **mynews3** and retain

24	hourly exported snapshots
7	daily exported snapshots
4	weekly exported snapshots
12	monthly exported snapshots
7	yearly exported snapshots

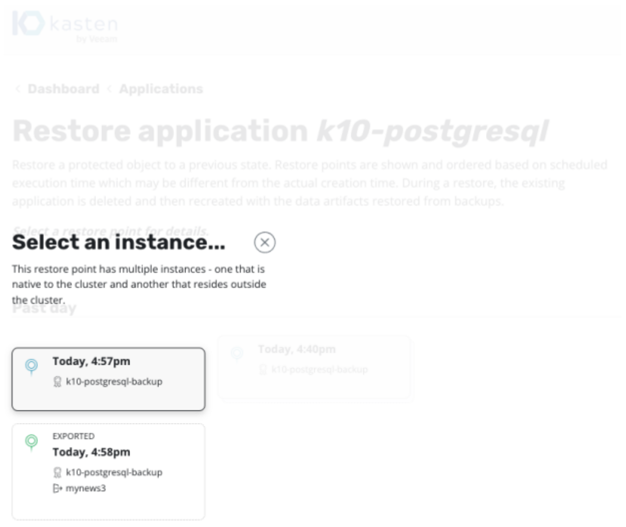
Export volume data for durable backups

Then, you can go back to the Dashboard to monitor the job progress.

4. In-place full restore
   Do a full restore in-place and confirm if the database is up running, from Dashboard->Applications, Click "restore"

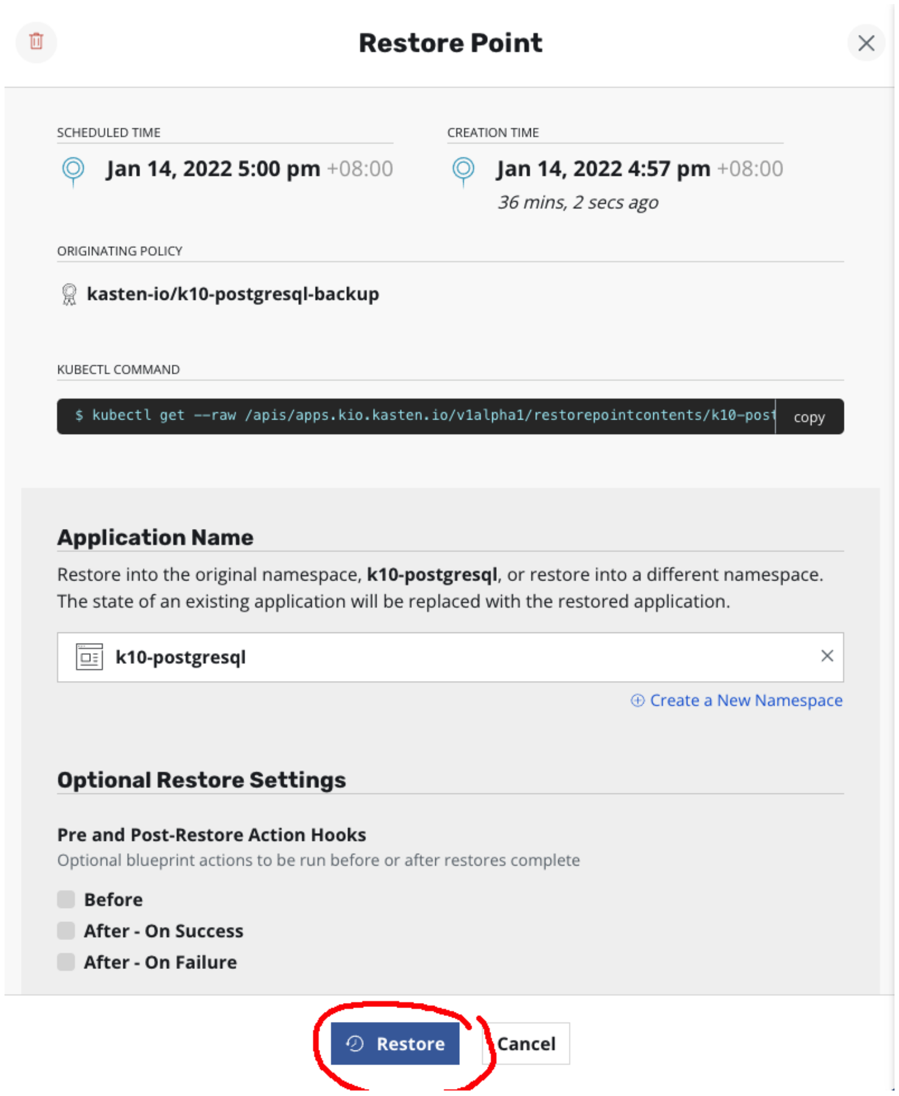


Select any one of the restore points



kasten by Veeam
Dashboard
Applications
Restore application k10-postgresql
Restore a protected object to a previous state. Restore points are shown and ordered based on scheduled execution time which may be different from the actual creation time. During a restore, the existing application is deleted and then recreated with the data artifacts restored from backups.
Select a restore point for details.
Select an instance...
This restore point has multiple instances - one that is native to the cluster and another that resides outside the cluster.
Past day
Today, 4:57pm
k10-postgresql-backup
Today, 4:40pm
k10-postgresql-backup
EXPORTED
Today, 4:58pm
k10-postgresql-backup
mynews3

Leave the rest of settings as default, Click Restore to restore in place.



SCHEDULED TIME
Jan 14, 2022 5:00 pm +08:00

CREATION TIME
Jan 14, 2022 4:57 pm +08:00
36 mins, 2 secs ago

ORIGINATING POLICY
kasten-io/k10-postgresql-backup

KUBECTL COMMAND

```
$ kubectl get --raw /apis/apps.kio.kasten.io/v1alpha1/restorepointcontents/k10-post
```

Application Name

Restore into the original namespace, **k10-postgresql**, or restore into a different namespace. The state of an existing application will be replaced with the restored application.

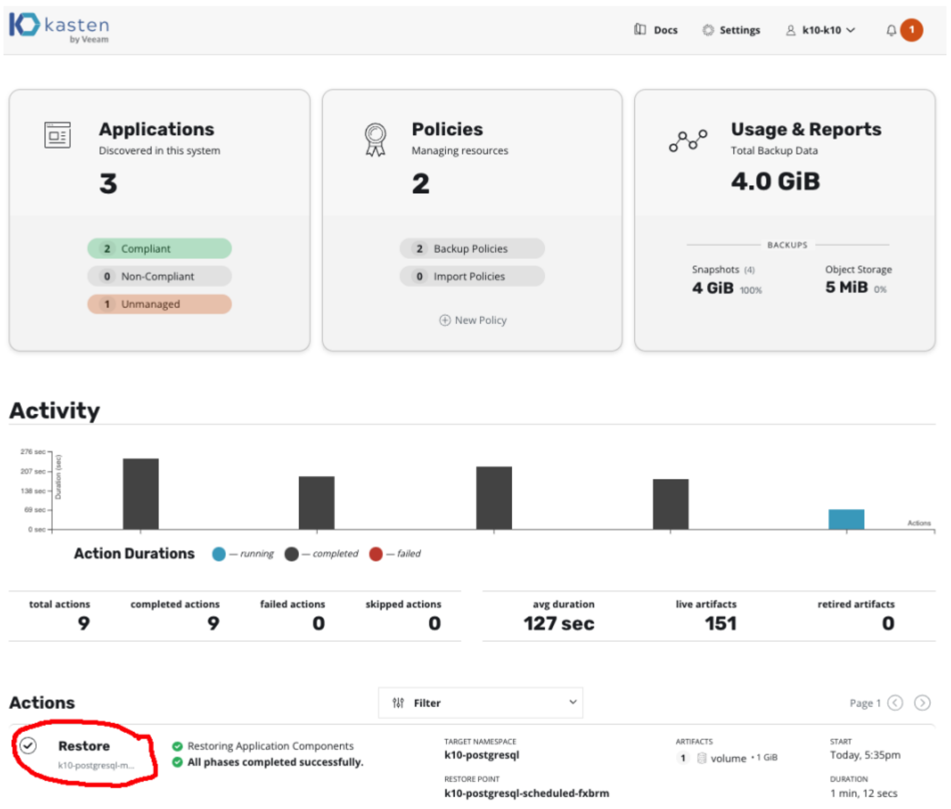
k10-postgresql

Optional Restore Settings

Pre and Post-Restore Action Hooks
Optional blueprint actions to be run before or after restores complete

- Before
- After - On Success
- After - On Failure

Monitor the restore progress from the Dashboard



Docs Settings k10-k10

Applications
Discovered in this system
3
2 Compliant
0 Non-Compliant
1 Unmanaged

Policies
Managing resources
2
2 Backup Policies
0 Import Policies
New Policy

Usage & Reports
Total Backup Data
4.0 GiB
BACKUPS
Snapshots (4) **4 GiB** 100%
Object Storage **5 MiB** 0%

Activity

total actions	completed actions	failed actions	skipped actions	avg duration	live artifacts	retired artifacts
9	9	0	0	127 sec	151	0

Actions Filter Page 1

Restore k10-postgresql-m...
Restoring Application Components
All phases completed successfully.
TARGET NAMESPACE **k10-postgresql**
RESTORE POINT **k10-postgresql-scheduled-fxbrm**
ARTIFACTS 1 volume • 1 GiB
START Today, 5:35pm
DURATION 1 min, 12 secs

Explore automation, try re-deploy and clean up

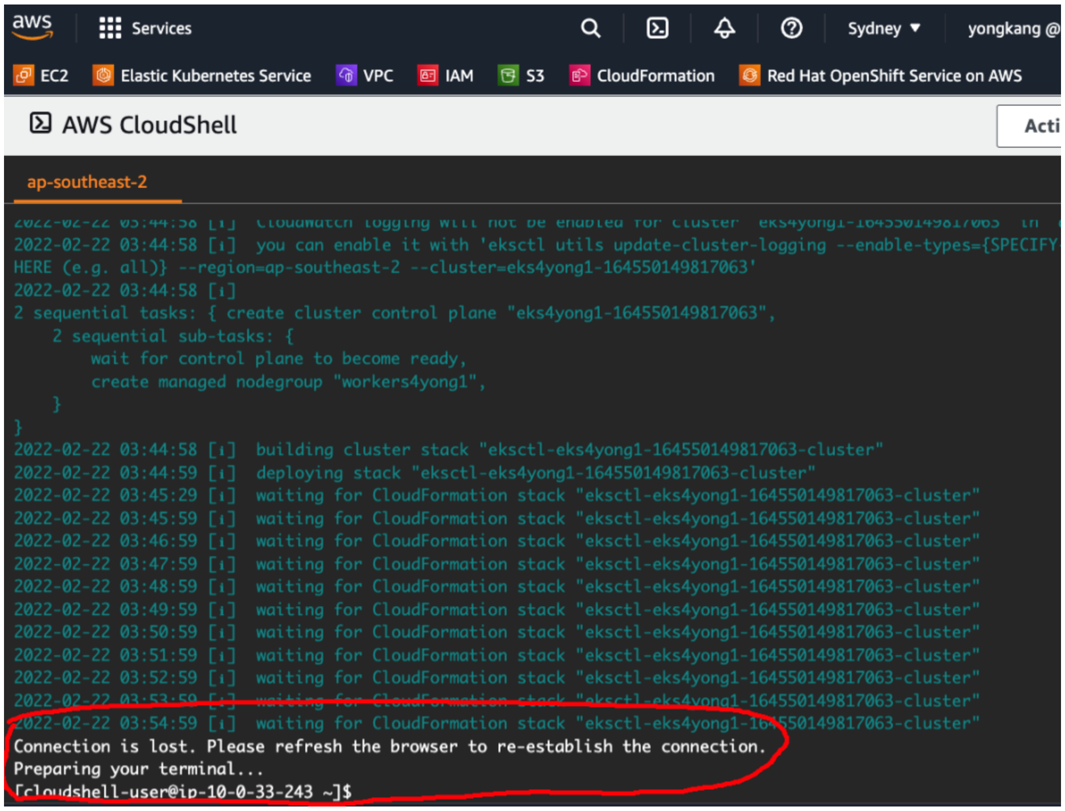
1. Remove K10 and Cassandra database using the command from CloudShell
   ./ee-k10-destroy.sh

2. Re-deploy K10, Cassandra Database, Backup Policy, run below command
   ./ee-k10-deploy.sh

3. Once done with the lab, return to AWS CloudShell and key in the command to clean up and delete lab resources. If you want to re-do it from the scratch, you may run the command to remove eks-k10 folder as well after ./ee-destroy.sh.
   ./ee-destroy.sh #Clean up the labs
   cd ~;rm -rf ~/eks-k10 #Remove the cloned repository

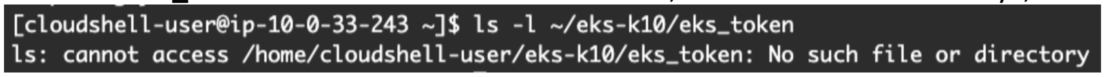
FAQ

1. My CloudShell session disconnected, what should I do?
   If your CloudShell session has no activity for 20 minutes, it will be disconnected, you may end up half way and you don't know what's the status of the labs. If you see something similar as below screenshot, please follow the instructions here to verify if the lab built successfully or not.



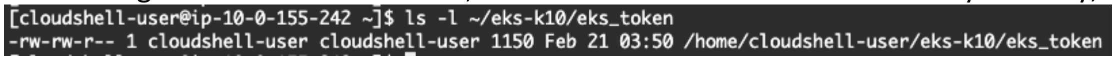
```
2022-02-22 03:44:58 [ℹ]  cloudwatch logging will not be enabled for cluster "eks4yong1-164550149817063" in
2022-02-22 03:44:58 [ℹ]  you can enable it with 'eksctl utils update-cluster-logging --enable-types={SPECIFY
HERE (e.g. all)} --region=ap-southeast-2 --cluster=eks4yong1-164550149817063'
2022-02-22 03:44:58 [ℹ]
2 sequential tasks: { create cluster control plane "eks4yong1-164550149817063",
    2 sequential sub-tasks: {
        wait for control plane to become ready,
        create managed nodegroup "workers4yong1",
    }
}
2022-02-22 03:44:58 [ℹ]  building cluster stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:44:59 [ℹ]  deploying stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:45:29 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:45:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:46:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:47:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:48:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:49:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:50:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:51:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:52:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:53:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
2022-02-22 03:54:59 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164550149817063-cluster"
Connection is lost. Please refresh the browser to re-establish the connection.
Preparing your terminal...
[cloudshell-user@ip-10-0-33-243 ~]$
```

a) Go to CloudShell terminal, re-connect or re-launch CloudShell
b) Copy/Paste the Credentials / CLI Snippets into your terminal and Press ENTER
c) Check if the **eks_token** file exists by running **ls -l ~/eks-k10/eks_token**
d) If the file **eks_token** exists and showing created recently, then everything is good, follow the lab guide to continue the labs.
e) If the file **eks_token** does not exist similar as below, "No such file or directory",



```
[cloudshell-user@ip-10-0-33-243 ~]$ ls -l ~/eks-k10/eks_token
ls: cannot access /home/cloudshell-user/eks-k10/eks_token: No such file or directory
```

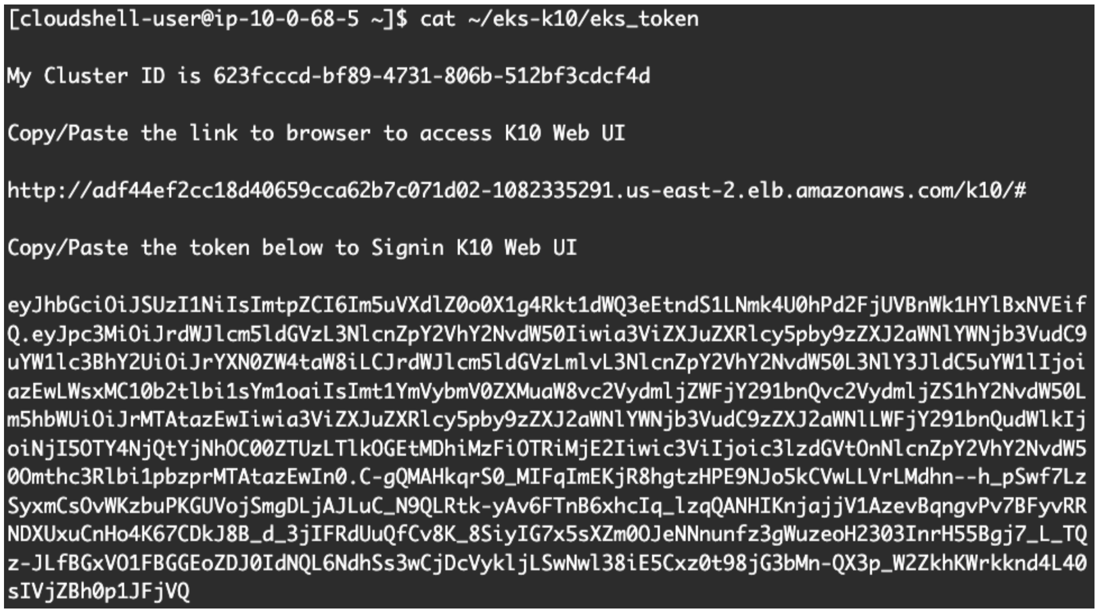
Or showing the date is old as below, the file created on Feb 21 which was yesterday,



```
[cloudshell-user@ip-10-0-155-242 ~]$ ls -l ~/eks-k10/eks_token
-rw-rw-r-- 1 cloudshell-user cloudshell-user 1150 Feb 21 03:50 /home/cloudshell-user/eks-k10/eks_token
```

Most likely Kasten K10 deployment was not completed yet. If that's the case, you should run **./ee-k10-deploy.sh** to deploy Kasten K10 and sample database etc..

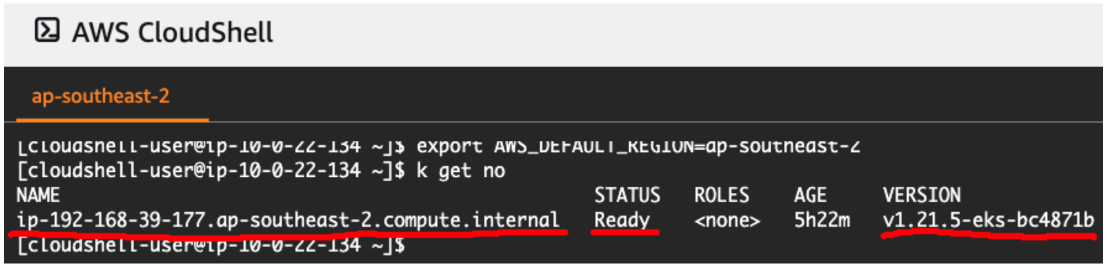
2. I lost Kasten K10 Web Console URL and Token, where to find it?
   a) Go to CloudShell terminal
   b) Copy/Paste the Credentials / CLI Snippets into your terminal and Press ENTER
   c) Run **cat ~/eks-k10/eks_token** to show the URL and token code
   You will see the output similar as below.



```
[cloudshell-user@ip-10-0-68-5 ~]$ cat ~/eks-k10/eks_token

My Cluster ID is 623fcccd-bf89-4731-806b-512bf3cdcf4d

Copy/Paste the link to browser to access K10 Web UI

http://adf44ef2cc18d40659cca62b7c071d02-1082335291.us-east-2.elb.amazonaws.com/k10/#

Copy/Paste the token below to Signin K10 Web UI

eyJhbGciOiJSUzI1NiIsImtpZCI6Im5uVXdlZ0o0X1g4Rkt1dWQ3eEtndS1LNmk4U0hPd2FjUVBnWk1HYlBxNVEif
Q.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9
uYW1lc3BhY2UiOiJrYXN0ZW4taW8iLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlY3JldC5uYW1lIjoi
azEwLWsxMC10b2tlbi1sYm1oaiIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50L
m5hbWUiOiJrMTAtazEwIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIj
oiNjI5OTY4NjQtYjNhOC00ZTUzLTlkOGEtMDhiMzFiOTRiMjE2Iiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW5
0Omthc3Rlbi1pbzprMTAtazEwIn0.C-gQMAHkqrS0_MIFqImEKjR8hgtzHPE9NJo5kCVwLLVrLMdhn--h_pSwf7Lz
SyxmCsOvWKzbuPKGUVojSmgDLjAJLuC_N9QLRtk-yAv6FTnB6xhcIq_lzqQANHIKnjajjV1AzevBqngvPv7BFyvRR
NDXUxuCnHo4K67CDkJ8B_d_3jIFRdUuQfCv8K_8SiyIG7x5sXZm0OJeNNnunfz3gWuzeoH2303InrH55Bgj7_L_TQ
z-JLfBGxVO1FBGGEoZDJ0IdNQL6NdhSs3wCjDcVykljLSwNwl38iE5Cxz0t98jG3bMn-QX3p_W2ZkhKWrkknd4L40
sIVjZBh0p1JFjVQ
```

3. How do I know if my EKS cluster is up running correctly?
   d) Go to CloudShell terminal
   e) Copy/Paste the Credentials / CLI Snippets into your terminal and Press ENTER
   f) Run **kubectl get nodes** to confirm if the worker nodes are up running

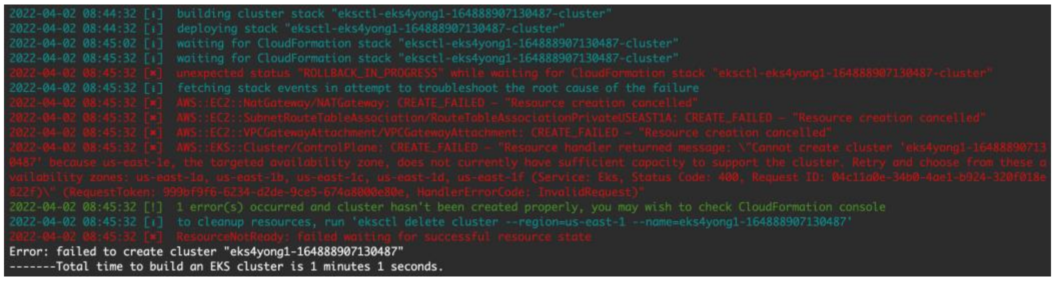
   If you see the output similar as below, that confirms the EKS cluster is Ready. If it is not Ready or not seeing any node at all, you need to follow the next FAQ #4 to clean up the environment first before redeploying it.



```
[cloudshell-user@ip-10-0-22-134 ~]$ export AWS_DEFAULT_REGION=ap-southeast-2
[cloudshell-user@ip-10-0-22-134 ~]$ k get no
NAME                                             STATUS   ROLES    AGE     VERSION
ip-192-168-39-177.ap-southeast-2.compute.internal   Ready    <none>   5h22m   v1.21.5-eks-bc4871b
[cloudshell-user@ip-10-0-22-134 ~]$
```

4. **./ee-deploy.sh** failed with errors
   g) Go to CloudShell terminal
   h) Copy/Paste the Credentials / CLI Snippets into your terminal and Press ENTER
   i) Run **./ee-destroy.sh** to clean up the environment first
   j) Then re-deploy by running **./ee-deploy.sh**

5. **./ee-deploy.sh** failed with errors “does not currently have sufficient capacity”, follow the steps here to re-deploy it.



```
2022-04-02 08:44:32 [ℹ]  building cluster stack "eksctl-eks4yong1-164888907130487-cluster"
2022-04-02 08:44:32 [ℹ]  deploying stack "eksctl-eks4yong1-164888907130487-cluster"
2022-04-02 08:45:02 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164888907130487-cluster"
2022-04-02 08:45:32 [ℹ]  waiting for CloudFormation stack "eksctl-eks4yong1-164888907130487-cluster"
2022-04-02 08:45:32 [✖]  unexpected status "ROLLBACK_IN_PROGRESS" while waiting for CloudFormation stack "eksctl-eks4yong1-164888907130487-cluster"
2022-04-02 08:45:32 [ℹ]  fetching stack events in attempt to troubleshoot the root cause of the failure
2022-04-02 08:45:32 [✖]  AWS::EC2::NatGateway/NATGateway: CREATE_FAILED – "Resource creation cancelled"
2022-04-02 08:45:32 [✖]  AWS::EC2::SubnetRouteTableAssociation/RouteTableAssociationPrivateUSEAST1A: CREATE_FAILED – "Resource creation cancelled"
2022-04-02 08:45:32 [✖]  AWS::EC2::VPCGatewayAttachment/VPCGatewayAttachment: CREATE_FAILED – "Resource creation cancelled"
2022-04-02 08:45:32 [✖]  AWS::EKS::Cluster/ControlPlane: CREATE_FAILED – "Resource handler returned message: \"Cannot create cluster 'eks4yong1-16488890713
0487' because us-east-1e, the targeted availability zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these a
vailability zones: us-east-1a, us-east-1b, us-east-1c, us-east-1d, us-east-1f (Service: Eks, Status Code: 400, Request ID: 04c11a0e-34b0-4ae1-b924-320f018e
822f)\" (RequestToken: 999bf9f6-6234-d2de-9ce5-674a8000e80e, HandlerErrorCode: InvalidRequest)"
2022-04-02 08:45:32 [!]  1 error(s) occurred and cluster hasn't been created properly, you may wish to check CloudFormation console
2022-04-02 08:45:32 [ℹ]  to cleanup resources, run 'eksctl delete cluster --region=us-east-1 --name=eks4yong1-164888907130487'
2022-04-02 08:45:32 [✖]  ResourceNotReady: failed waiting for successful resource state
Error: failed to create cluster "eks4yong1-164888907130487"
-------Total time to build an EKS cluster is 1 minutes 1 seconds.
```

a) Run **./ee-destroy.sh** to clean up the environment
b) Re-deploy by running **./ee-deploy.sh**

6. All backup, export or restore job related failures or issues
   a) Go to CloudShell terminal
   b) Copy/Paste the Credentials / CLI Snippets into your terminal and Press ENTER
   c) Run **./ee-k10-destroy.sh** to clean up K10 deployment first
   d) Then re-deploy by running **./ee-k10-deploy.sh**

7. How to request AWS Event Engine lab access? This is only for those don't have an AWS account or prefer not to charge to your own AWS account.
   a) 24 hours before your scheduled lab time
   b) Send an email request to he@yongkang.cloud with request date, time information and number of labs required. E.g. 4th Feb 2022, 12pm to 4pm, 2 labs

8. I already have an AWS account. I need to access the labs anytime and I don't mind to be charged. Which lab guide I should follow?
   Here is the lab guide for those are not using AWS Event Engine Lab Platform.
   http://aws-lg.yongkang.cloud