

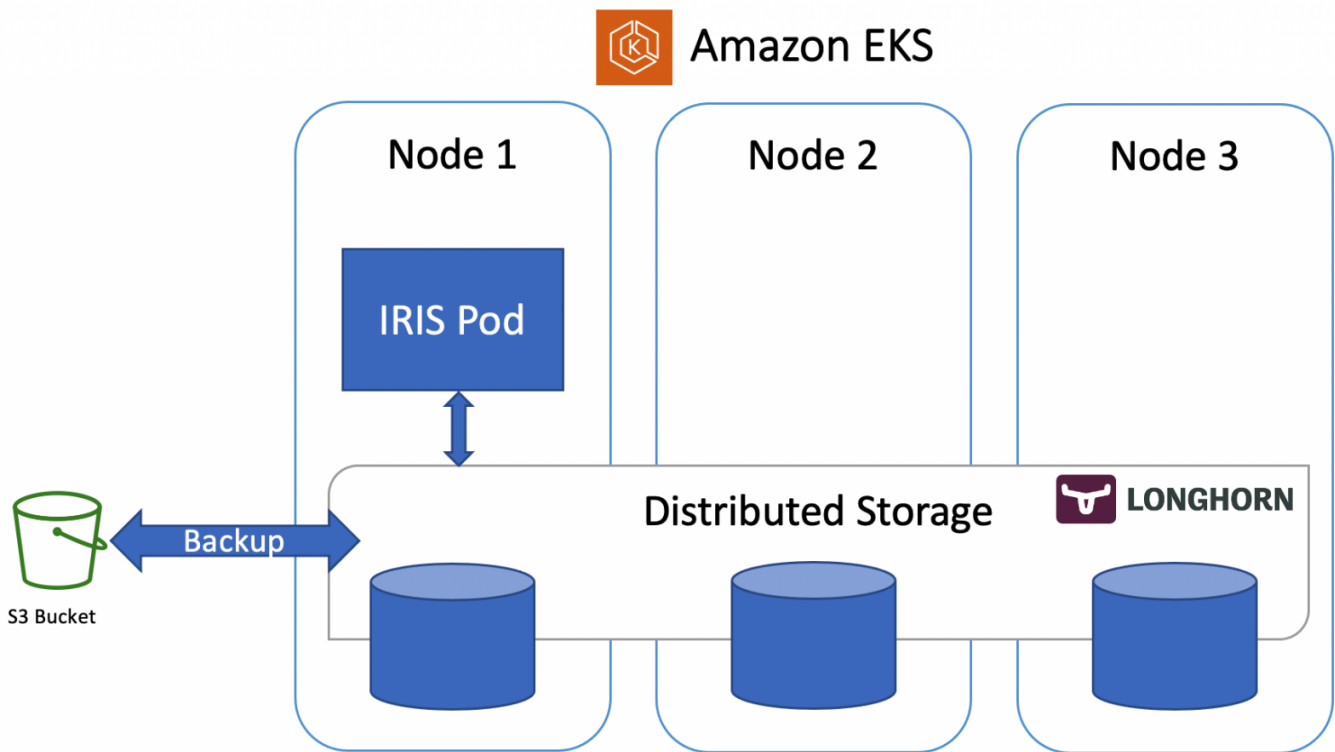
文章

[Michael Lei](#) · 五月 3, 2022 阅读大约需 6 分钟

[Open Exchange](#)

Amazon EKS, IRIS 高可用与备份

所有源代码均在: <https://github.com/antonum/ha-iris-k8s>



在上一篇[文章](#)

中，我们讨论了如何在k8s集群上建立具有高可用性的IRIS，基于分布式存储，而不是传统的镜像。作为一个例子，那篇文章使用了Azure AKS集群。在这一篇中，我们将继续探讨k8s上的高可用配置。这一次，基于Amazon EKS（AWS管理的Kubernetes服务），并将包括一个基于Kubernetes 快照进行数据库备份和恢复的选项。

安装

开始干活. 首先需要有一个AWS账户, 安装 [AWS CLI](#), [kubectl](#) 和 [eksctl](#) 工具. 要创建新的集群, 请运行以下命令:

```
eksctl create cluster \
--name my-cluster \
--node-type m5.2xlarge \
--nodes 3 \
--node-volume-size 500 \
--region us-east-1
```

这个命令需要大约15分钟，部署EKS集群并使其成为你的kubectl工具的默认集群。你可以通过运行以下代码来验证你的部署：

```
kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
ip-192-168-19-7.ca-central-1.compute.internal	Ready	<none>	18d	v1.18.9-eks-d1db3c
ip-192-168-37-96.ca-central-1.compute.internal	Ready	<none>	18d	v1.18.9-eks-d1db3c
ip-192-168-76-18.ca-central-1.compute.internal	Ready	<none>	18d	v1.18.9-eks-d1db3c

下一步是安装Longhorn分布式存储引擎。

```
kubectl create namespace longhorn-system
```

```
kubectl apply -f https://raw.githubusercontent.com/longhorn/longhorn/v1.1.0/deploy/iscsi/longhorn-iscsi-installation.yaml --namespace longhorn-system
```

```
kubectl apply -f https://raw.githubusercontent.com/longhorn/longhorn/master/deploy/longhorn.yaml --namespace longhorn-system
```

最后安装 IRIS :

```
kubectl apply -f https://github.com/antonum/ha-iris-k8s/raw/main/tldr.yaml
```

现在，你就有了一个功能齐全的EKS集群，并安装了Longhorn分布式存储和IRIS。

你可以回到上一篇[文章](#)，尝试对集群和IRIS部署进行各种破坏，看看系统如何自我修复。请看[模拟故障Simulate the Failure](#)部分。

Bonus #1 IRIS on ARM

IRIS EKS和Longhorn都支持ARM架构，所以我们可以使用AWS Graviton 2实例部署相同的配置，基于ARM架构。

你只需要将EKS节点的实例类型改为 "m6g "系列，将IRIS镜像改为基于ARM的。

```
eksctl create cluster \
--name my-cluster-arm \
--node-type m6g.2xlarge \
--nodes 3 \
--node-volume-size 500 \
--region us-east-1
```

tldr.yaml

```
containers:
#- image: store/intersystems/iris-community:2020.4.0.524.0
- image: store/intersystems/irishealth-community-arm64:2020.4.0.524.0
name: iris
```

或只是用:

```
kubectl apply -f https://github.com/antonum/ha-iris-k8s/raw/main/tldr-iris4h-arm.yaml
```

这就行了! 你就有了一个在ARM平台上运行的IRIS Kubernetes集群。

Bonus #2 - 备份与恢复

生产级架构的一个经常被忽视的部分是创建数据库的备份，并在需要时快速恢复和克隆这些备份的能力。

在Kubernetes中，常见的方式是使用持久化卷快照。

首先--你需要安装所有需要的k8s组件:

```
#Install CSI Snapshotter and CRDs
```

```
kubectl apply -n kube-system -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -n kube-system -f https://github.com/kubernetes-csi/external-snapshotter/raw/master/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -n kube-system -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
kubectl apply -n kube-system -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
kubectl apply -n kube-system -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
```

下一步 - 为Longhorn配置S3 bucket凭证 (请查看 [详细指导](#)):

```
#Longhorn backup target s3 bucket and credentials longhorn would use to access that bucket
#See https://longhorn.io/docs/1.1.0/snapshots-and-backups/backup-and-restore/setup-backup-target/ for manual setup instructions
longhorn_s3_bucket=longhorn-backup-123xx #bucket name should be globally unique, unless you want to reuse existing backups and credentials
longhorn_s3_region=us-east-1
longhorn_aws_key=AKIAVHCUNTEXAMPLE
longhorn_aws_secret=g2q2+5DVXk5p3AHIB5m/Tk6U6dXrEXAMPLE
```

下面的命令将从上一步骤中获取环境变量，并使用它们来配置Longhorn备份。

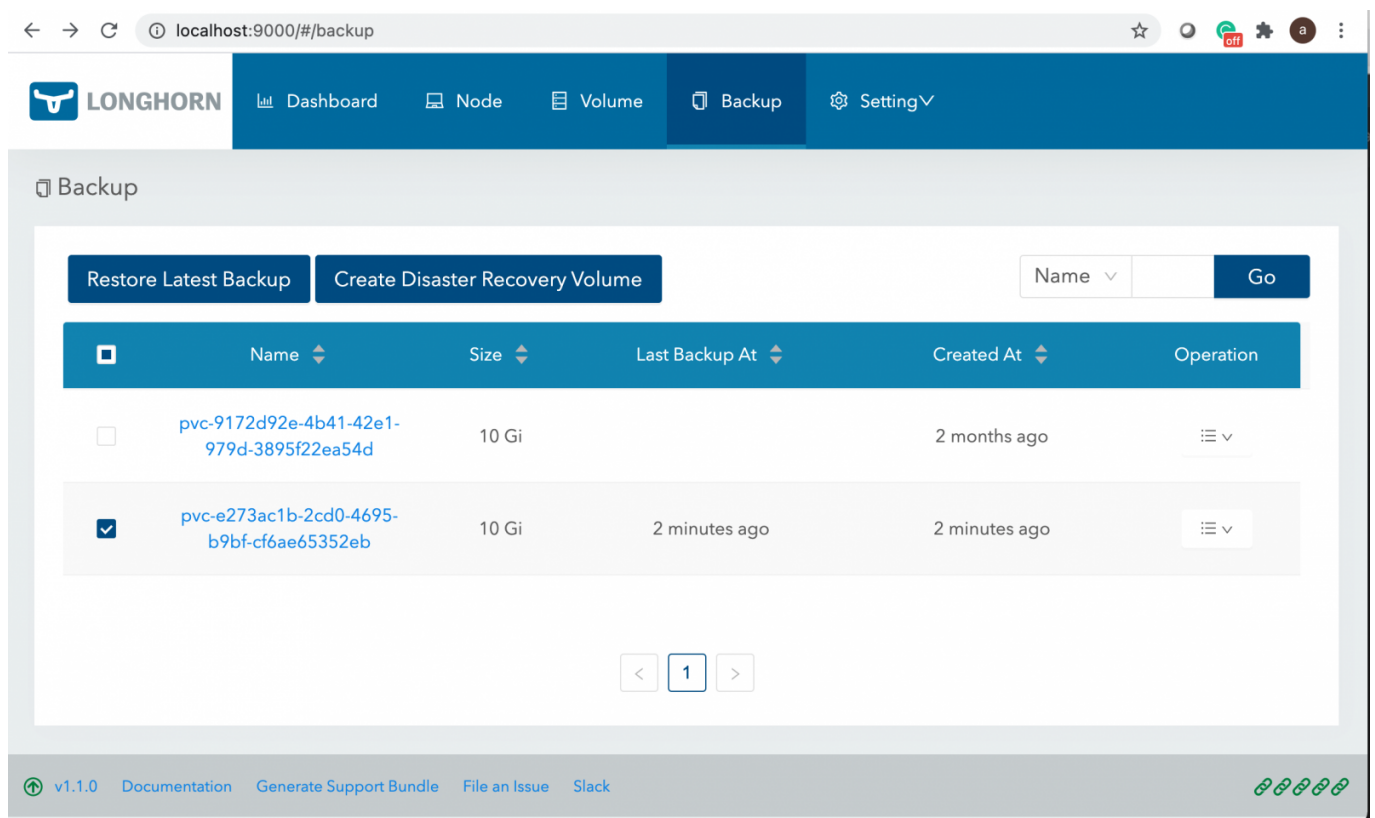
```
#configure Longhorn backup target and credentials
```

```
cat <<EOF | kubectl apply -f -
apiVersion: longhorn.io/v1beta1
kind: Setting
metadata:
  name: backup-target
  namespace: longhorn-system
value: "s3://$longhorn_s3_bucket@$longhorn_s3_region/" # backup target here
---
apiVersion: v1
kind: Secret
metadata:
  name: "aws-secret"
  namespace: "longhorn-system"
  labels:
data:
  # echo -n '<secret>' | base64
  AWS_ACCESS_KEY_ID: $(echo -n $longhorn_aws_key | base64)
```

```
AWS_SECRET_ACCESS_KEY: $(echo -n $longhorn_aws_secret | base64)
---
apiVersion: longhorn.io/v1beta1
kind: Setting
metadata:
  name: backup-target-credential-secret
  namespace: longhorn-system
value: "aws-secret" # backup secret name here
EOF
```

它可能看起来很多，但它基本上是告诉Longhorn使用一个特定的S3 bucket，用指定的凭证来存储备份的内容。

搞定! 如果你现在进入Longhorn的用户界面，你就可以创建备份，恢复等等。



关于如何连接到Longhorn用户界面的快速练习:

```
kubectl get pods -n longhorn-system
# note the full pod name for 'longhorn-ui-...' pod
kubectl port-forward longhorn-ui-df95bdf85-469sz 9000:8000 -n longhorn-system
```

这将把到Longhorn UI的流量转发到你的本地 <http://localhost:9000>

程序化的 备份/恢复

通过Longhorn用户界面进行备份和恢复可能是足够好的第一步--但我们将更进一步，使用k8s Snapshot APIs，以编程方式进行备份和恢复。

首先 - 快照本身. iris-volume-snapshot.yaml

```
apiVersion: snapshot.storage.k8s.io/v1beta1
```

```
kind: VolumeSnapshot
metadata:
  name: iris-longhorn-snapshot
spec:
  volumeSnapshotClassName: longhorn
  source:
    persistentVolumeClaimName: iris-pvc
```

这个卷快照指的是源卷 "iris-pvc"，我们确实在IRIS部署中使用。因此，只要应用这个就可以立即开始备份过程。

在快照之前/之后执行IRIS Write Daemon Freeze/Thaw是一个好主意。

```
#Freeze Write Daemon
echo "Freezing IRIS Write Daemon"
kubectl exec -it -n $namespace $pod_name -- iris session iris -U%SYS "##Class(Backup.General).ExternalFreeze()"
status=$?
if [[ $status -eq 5 ]]; then
  echo "IRIS WD IS FROZEN, Performing backup"
  kubectl apply -f backup/iris-volume-snapshot.yaml -n $namespace
elif [[ $status -eq 3 ]]; then
  echo "IRIS WD FREEZE FAILED"
fi
#Thaw Write Daemon
kubectl exec -it -n $namespace $pod_name -- iris session iris -U%SYS "##Class(Backup.General).ExternalThaw()"
```

恢复过程是非常直接的。它基本上是创建一个新的PVC，并指定快照为源。

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: iris-pvc-restored
spec:
  storageClassName: longhorn
  dataSource:
    name: iris-longhorn-snapshot
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
```

然后你就根据这个PVC创建一个新的部署。检查github repo中的这个[测试脚本](#)，它将依次进行。

创建全新的IRIS部署

- 向IRIS添加一些数据
- 冻结写Daemon，拍摄快照，解冻写Daemon
- 在快照的基础上，创建一个IRIS部署的克隆
- 验证所有的数据是否还在那里

验证所有的数据是否还在那里，现在你将有两个相同的IRIS部署，一个是另一个的克隆备份。

Amazon EKS, IRIS 高可用与备份

Published on InterSystems Developer Community (<https://community.intersystems.com>)

Enjoy!

[#AWS](#) [#云](#) [#备份](#) [#容器化](#) [#开发运维](#) [#部署](#) [#高可用性](#) [#InterSystems IRIS](#) [#InterSystems IRIS for Health](#) [#Open Exchange](#)
[在 InterSystems Open Exchange 上检查相关应用程序](#)

源

URL:<https://cn.community.intersystems.com/post/amazon-eks-iris-%E9%AB%98%E5%8F%AF%E7%94%A8%E4%B8%8E%E5%A4%87%E4%BB%BD>