Резервное копирование мастер-узлов

Регулярное резервное копирование мастер-узлов Nova Container Platform необходимо, чтобы восстановить кластер в критических ситуациях. Рекомендуется организовать хранение резервных копий в инфраструктуре за пределами кластера Kubernetes. Резервное копирование не должно выполняться в пиковые часы нагрузки, поскольку процессы подготовки резервных копий оказывают влияние на дисковую подсистему мастер-узлов.

1. Объекты резервного копирования

Критически важными компонентами платформы, мастер-узлов и среды Kubernetes являются следующие сервисы:

- **Etcd**: основное хранилище данных, содержит всю информацию о ресурсах в Kubernetes.
- **StarVault**: основное хранилище TLS-сертификатов, секретов, учетных записей.

Кроме этого, в Nova Container Platform поддерживается опциональная возможность резервного копирования следующих объектов:

- **Токены StarVault (Unseal Tokens)**: токены для распечатывания (расшифровки) хранилища StarVault.
- **TLS-сертификаты**: сертификаты компонентов Kubernetes Control Plane и Nova Configuration Manager.
- **Ключи шифрования Etcd**: ключи, необходимые для шифрования секретов в Etcd.

2. Совместимость резервных копий

В Nova Container Platform поддерживается полное восстановление резервной копии только для патч-версии платформы (`x.y.Z`), в которой данная резервная копия создавалась.

Восстановление образов хранилищ Etcd и StarVault в несовместимые версии Nova Container Platform может привести к непредсказуемому поведению служебных сервисов платформы.

3. Выбор решения для резервного копирования

В Nova Container Platform поддерживается два основных решения для резервного копирования мастер-узлов:

- Регулярное задание *CronJob* в Kubernetes.
- С помощью дополнительного модуля Nova Data Protection и ПО Velero, входящего в его состав.

В каждом из решений запускается сервис Nova Backup Daemon, который выполняет резервное копирование информации на мастер-узлах. В зависимости от используемого решения в пользовательской инфраструктуре должно быть подготовлено соответствующее хранилище:

- В регулярном задании *CronJob* сервис Nova Backup Daemon использует том, куда выполняется сохранение резервной копии. Рекомендуется использовать в качестве тома подключаемое NFS-хранилище.
- При использовании ПО Velero из модуля Nova Data Protection в пользовательской инфраструктуре должно быть подготовлено и доступно S3-совместимое объектное хранилище. В данном сценарии сервис Nova Backup Daemon сохраняет резервную копию мастер-узлов локально, а затем выполняется резервная копия сервиса с его данными с помощью Velero.

Вы можете использовать наиболее подходящее решение в зависимости от вашей инфраструктуры и доступных ресурсов.

В разделе Резервное копирование и восстановление пользовательских данных вы можете ознакомиться с требованиями для установки модуля Nova Data Protection.

4. Настройка резервного копирования мастер-узлов

В данном разделе описана процедура настройки резервного копирования мастер-узлов платформы Nova Container Platform:

- используя возможности запуска регулярных заданий CronJob в Kubernetes;
- с помощью ПО Velero, входящего в состав модуля Data Protection.

4.1. Настройка резервного копирования с помощью регулярного задания CronJob

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.
- ✓ У вас подготовлено NFS-хранилище для резервных копий.

Порядок действий

1. Подготовьте манифест кастомизации (*Kustomization*) в зависимости от количества мастер-узлов в кластере Kubernetes:

   ▸ **3 мастер-узла**

   ▸ **1 мастер-узел**

2. Установите параметры резервного копирования.

   Укажите переменные окружения сервиса Nova Backup Daemon:

Параметр	Значение по умолчанию	Описание
`INCLUDE_STARVAULT_UNSEAL_TOKENS`	`true`	Добавление в архив резервной копии токенов для распечатывания StarVault.
`INCLUDE_PKI_DATA`	`true`	Добавление в архив резервной копии всех выпущенных TLS-сертификатов для платформы.
`INCLUDE_DATA_ENCRYPTION_CONFIG`	`true`	Добавление в архив резервной копии ключей шифрования Etcd.
`INCLUDE_STARVAULT_DB_BACKUP`	`true`	Добавление в архив резервной копии снимка БД StarVault.
`RETENTION_PERIOD_DAYS`	`7`	Количество дней, в течение которых необходимо хранить резервные копии на внешнем хранилище.

   Информация

   Для каждого мастер-узла создается собственная резервная копия выбранных данных с учетом следующей информации:

   - Резервная копия Etcd создается всегда и только однократно на одном из мастер-узлов платформы.
   - Для получения полного набора токенов для распечатывания StarVault необходимо иметь резервные копии всех мастер-узлов.

- Резервное копирование TLS-сертификатов не включает приватные ключи центров сертификации платформы, поскольку они не являются экспортируемыми и хранятся только в БД StarVault.
- Управление количеством дней хранения резервных копий доступно только для решения резервного копирования с помощью регулярного задания CronJob. При использовании модуля Nova Data Protection политика хранения данных определяется средствами ПО Velero.

Укажите спецификацию тома `backup-volume`, предназначенного для хранения резервных копий.

Например, для NFS-хранилища используйте спецификацию:

```yaml
spec:
  volumes:
  - name: backup-volume
    nfs:
      server: nfs-share.mycompany.local
      path: /nova-cluster-8b4e9344-9dcd-4c64-b98a-4a8a08a53da6
      readOnly: false
```

где `nfs.server` - DNS-имя вашего NFS-сервера или его IP-адрес, `nfs.path` - путь для хранения резервных копий.

Укажите график резервного копирования в формате Cron, например, для выполнения резервных копий каждый день в 4:00:

```yaml
schedule: "0 4 * * *"
```

3. Сохраните полученный манифест и установите его в кластер Kubernetes с помощью Nova Console или *kubectl*.

   Пример вывода:

   ```bash
   kubectl apply -f nova-release-cluster-backup-cronjob.yaml

   kustomization.kustomize.toolkit.fluxcd.io/nova-release-cluster-backup-
   cronjob created
   ```

4. Проверьте статус кастомизации:

   ```bash
   kubectl get ks nova-release-cluster-backup-cronjob -n nova-gitops
   ```

   Пример вывода:

```
kubectl get ks nova-release-cluster-backup-cronjob -n nova-gitops

NAME                                  AGE   READY   STATUS
nova-release-cluster-backup-cronjob   41s   True    Applied revision:
v6.0.1@sha1:6789a4025a1edd244044677ed43d8087018e5a7d
```

5. Получите информацию об установленном регулярном задании Cronjob:

```
kubectl get cronjobs.batch -n nova-cluster-backup
```

Пример вывода:

```
kubectl get cronjobs.batch -n nova-cluster-backup

NAME                    SCHEDULE    SUSPEND   ACTIVE   LAST SCHEDULE   AGE
nova-backup-scheduled   0 4 * * *   False     1        7s              14m
```

Регулярное задание запустится автоматически в указанное в графике время.

В процессе работы задания на мастер-узлах будут запускаться сервисы Nova Backup Daemon, а по завершению работы их статус можно будет отследить в пространстве имен `nova-cluster-backup`. Статус может быть *Completed* в успешном случае, и *Error* в случае ошибки.

Для просмотра статуса резервного копирования выполните команду:

```
kubectl get pods -n nova-cluster-backup
```

Пример вывода:

```
kubectl get pods -n nova-cluster-backup

NAME                                    READY   STATUS      RESTARTS   AGE
nova-backup-scheduled-28629407-0-v7qbv   0/1     Completed   0          24s
nova-backup-scheduled-28629407-1-t2wbt   0/1     Completed   0          24s
nova-backup-scheduled-28629407-2-h6hq9   0/1     Completed   0          24s
```

4.1.1. Проверка резервных копий на внешнем хранилище

Вы также можете проверить наличие резервных копий на внешнем хранилище. На примере ниже показана директория внешнего NFS-сервера, куда выполняется резервное копирование мастер-узлов кластера Nova Container Platform.

Пример вывода:

```
ls -la /storage/nova-364f9cbe-b209-4f3a-a4d4-9fe36a81afef/
```

```
drwxr-xr-x. 2 nobody nobody     4096 Jun  7 15:47 .
drwxr-xr-x. 3 nobody nobody       55 Jun  7 14:44 ..
-rw-r--r--. 1 root   root   16209258 Jun  7 15:47 etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-07_124701.db.tar.gz
-rw-------. 1 root   root     219326 Jun  7 15:47 nova-master-1-nova-
internal_kuberesources_2024-06-07_124701.tar.gz
-rw-------. 1 root   root     219247 Jun  7 15:47 nova-master-2-nova-
internal_kuberesources_2024-06-07_124701.tar.gz
-rw-------. 1 root   root     219279 Jun  7 15:47 nova-master-3-nova-
internal_kuberesources_2024-06-07_124700.tar.gz
-rw-------. 1 root   root     314521 Jun  7 15:47 starvault_snapshot_nova-
v6.0.1_2024-06-07_124701.db
```

Для архивов резервных копий применяется следующая схема именования:

- Имя архива резервной копии Etcd имеет формат `etcd_snapshot_nova-<Версия Nova>_k8s-<Версия Kubernetes>_<Время создания копии>.db.tar.gz`.
- Имена архивов резервных копий конфигураций мастер-узлов имеют формат `<Имя узла в Kubernetes>_kuberesources_<Время создания копии>.tar.gz`.
- Имя архива резервной копии StarVault имеет формат `starvault_snapshot_nova-<Версия Nova>_<Время создания копии>1.db`.

4.2. Настройка резервного копирования с помощью Velero

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.
- ✓ Вы установили модуль Data Protection в Nova Container Platform.
- ✓ Вы настроили утилиту velero для работы с резервными копиями Velero.
- ✓ Вы подготовили внешнее хранилище Velero `BackupStorageLocation`.

Порядок действий

1. Для установка сервиса Nova Backup Daemon с помощью модуля Data Protection в Nova Container Platform используйте представленный далее манифест кастомизации.

   ▸ **Манифест кастомизации**

2. Проверьте статус кастомизации:

```
kubectl get ks nova-release-cluster-backup-velero -n nova-gitops
```

Пример вывода:

```
kubectl get ks nova-release-cluster-backup-velero -n nova-gitops

NAME                                AGE   READY   STATUS
nova-release-cluster-backup-velero   39s   True    Applied revision:
v6.0.1@sha1:86f53cb7e4dbacb29fa42f2c1c9814fa6aec7a07
```

3. Получите информацию об установленном сервисе Nova Backup Daemon:

```
kubectl get ds nova-backup-daemon -n nova-cluster-backup
```

Пример вывода:

```
kubectl get ds nova-backup-daemon -n nova-cluster-backup

NAME                DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE
NODE SELECTOR                         AGE
nova-backup-daemon   3         3         3       3            3
node-role.kubernetes.io/control-plane=   103s
```

4. Резервную копию можно создать разово или настроить расписание для автоматического создания резервных копий согласно настройкам. Для создания разовой или регулярной резервной копии необходимо сначала создать Место хранения резервной копии и только потом приступить к созданию резервных копий. Далее описаны оба варианта создания резервных копий.

4.2.1. Разовое создание резервной копии

4.2.1.1. Через консоль

Подготовьте и установите манифест резервного копирования мастер-узлов в кластер Kubernetes с помощью Nova Console или *kubectl*.

Пример и расшифровка полей конфигурационного файла приведен ниже:

▸ **YAML-манифест**

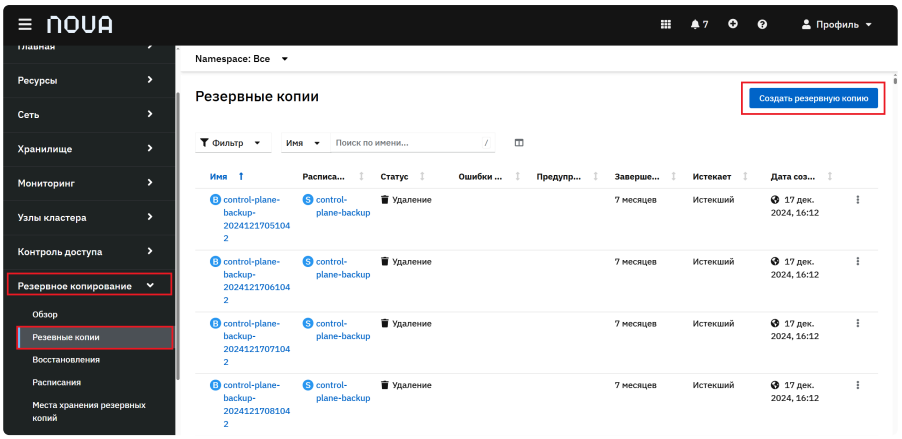
Пример вывода:

```
kubectl create -f backup.yaml

backup.velero.io/one-time-backup created
```

4.2.1.2. Через веб-интерфейс

1. Перейдите в раздел **Резервные копии**.
2. Нажмите кнопку **[Создать резервную копию]**.



3. Далее необходимо задать параметры конфигурации. В верхней части интерфейса выберите один из следующих способов:

▸ Через YAML-манифест

▸ Через заполнение формы

4. Нажмите кнопку **[Создать]**.

4.2.2. Создание резервных копий по расписанию

4.2.2.1. Через консоль

Подготовьте и установите манифест плана резервного копирования мастер-узлов в кластер Kubernetes с помощью Nova Console или *kubectl*:

▸ YAML-манифест

Укажите график резервного копирования в формате Cron, например, для выполнения резервных копий каждый день в 4:00:

Поле для задания времени:

```yaml
schedule: "0 4 * * *"
```

Пример вывода:

```bash
kubectl create -f backup-schedule.yaml

schedule.velero.io/control-plane-backup created
```

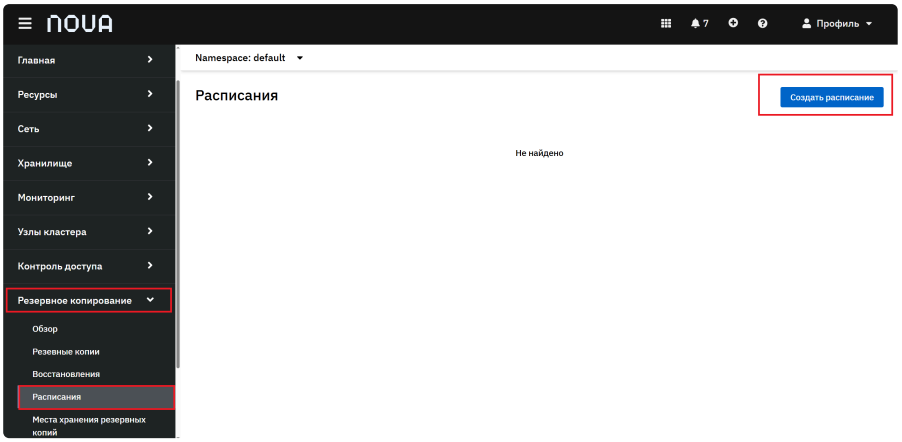
Если требуется выполнить резервное копирование данных внутри пода или PVC, необходимо добавить в ресурс `Schedule` параметры `defaultVolumesToFsBackup` и `orderedResources`.

Пример YAML-манифеста с параметрами для резервного копирования данных внутри пода и характеристика его полей показаны ниже:

▸ YAML-манифест

4.2.2.2. Через веб-интерфейс

1. Перейдите в раздел **Расписания**.
2. Нажмите кнопку **[Создать расписание]**.



3. Далее необходимо задать параметры конфигурации. В верхней части интерфейса выберите один из следующих способов:

   ▸ Через YAML-манифест

   ▸ Через заполнение формы

4. Нажмите на кнопку **[Создать]**.

4.2.3. Проверка после создания резервных копий

1. Проверьте статус плана резервного копирования одним из способов:

   ▸ **Через kubectl**

   ▸ **Через Velero CLI**

   ▸ **Через веб-интерфейс**

2. Дождитесь выполнения резервного копирования и проверьте статус плана резервного копирования:

   ▸ **kubectl**

   ▸ **Velero CLI**

3. Проверьте статус отдельных заданий резервного копирования:

   ▸ **kubectl**

   ▸ **Velero CLI**

4.2.4. Проверка резервных копий на внешнем хранилище

Вы также можете проверить наличие резервных копий в объектном хранилище. На примере ниже показан пример резервных копий в объектном хранилище, куда выполняется резервное копирование мастер-узлов кластера Nova Container Platform.

Вы можете использовать любой совместимый с вашим объектным хранилищем консольный клиент или веб-интерфейс.

Пример вывода:

```bash
aws s3 ls s3://velero-backup-bucket --endpoint-url https://s3.mycompany.local

                           PRE backups/
                           PRE kopia/
```

В директории `backups/` находятся резервные копии спецификаций ресурсов (манифестов) Kubernetes.

Пример вывода:

```bash
aws s3 ls s3://velero-backup-bucket/backups/ --endpoint-url
https://s3.mycompany.local

                           PRE control-plane-backup-20240610121525/
                           PRE control-plane-backup-20240610131525/
                           PRE control-plane-backup-20240610141525/
                           PRE control-plane-backup-20240610151525/
```

В директории `kopia/` находятся резервные копии файлов сервиса Nova Backup Daemon: резервные копии Etcd, StarVault, PKI и др.

Пример вывода:

```bash
aws s3 ls s3://velero-backup-bucket/kopia/nova-cluster-backup/ --endpoint-url
https://s3.mycompany.local

2024-06-10 15:15:43        747
_log_20240610121542_f5ce_1718021742_1718021743_1_6bd1da03b924c1be6ec634227e336f1
9
2024-06-10 15:15:45       1685
_log_20240610121544_c120_1718021744_1718021745_1_dfc7e059b0394a85ca25fe7ecce7ab2
9
2024-06-10 15:16:04       1755
_log_20240610121603_6ea6_1718021763_1718021764_1_64d9d4b3f4b3d8c4eb7dc310be213d1
a
2024-06-10 15:16:28       2640
_log_20240610121626_aa10_1718021786_1718021788_1_0eeb3acc062071d4a84bd08f8ab8262
1
2024-06-10 16:15:32       1919
_log_20240610131531_287a_1718025331_1718025332_1_60750784af3ee83ed569fbe2e82d40e
e
2024-06-10 16:15:39       1941
_log_20240610131537_04d8_1718025337_1718025339_1_564e1ea03aade8ccfb2236efc04f1fb
7
2024-06-10 16:15:50       2737
_log_20240610131548_7f66_1718025348_1718025350_1_5595519176d088bf2512ae9d2251381
6
2024-06-10 16:16:26       3775
_log_20240610131625_6f8c_1718025385_1718025386_1_c60f0d81f4b09128277702e43ddf365
5
2024-06-10 17:15:32       2097
_log_20240610141530_363a_1718028930_1718028932_1_dd21e1a8aa835344b058b311aa58ad7
8
2024-06-10 17:15:39       3334
_log_20240610141537_b56d_1718028937_1718028939_1_efd4f6b0da8be6a04d392e7aa4e8e20
b
2024-06-10 17:15:51       2875
_log_20240610141549_8d12_1718028949_1718028951_1_75f3ceb8dce9a244c56bd08631a3476
1
2024-06-10 17:16:26       1320
_log_20240610141625_ca81_1718028985_1718028986_1_c4975114b6b94de09792c2bbe869406
1
2024-06-10 18:15:32       2261
_log_20240610151531_7b9a_1718032531_1718032532_1_7325bf67c58d582df1363ddaff938cf
a
2024-06-10 18:15:39       2392
_log_20240610151537_074c_1718032537_1718032539_1_4c8b33484e2ee5c7cc21d411b7eeefb
d
2024-06-10 18:15:50       3330
_log_20240610151548_2718_1718032548_1718032550_1_5b2cb4f36c3124cc78acf28574555b6
5
2024-06-10 18:16:26       1825
```

```
_log_20240610151625_aff3_1718032585_1718032586_1_9be5f2ab4206fd4b43b49fdc8dfde30
3
2024-06-10 15:15:42         30 kopia.blobcfg
2024-06-10 18:16:26        620 kopia.maintenance
2024-06-10 15:15:42       1075 kopia.repository
2024-06-10 18:15:31   26726736 p0941d6b0f97eccef7587b5cbff2207f6-
s77d051db729eba1e129
...
```

Защита пользовательских данных с помощью модуля Data Protection

В данном разделе описана защита пользовательских данных с помощью модуля Data Protection.

1. Установка модуля Nova Data Protection

Для установки модуля Data Protection в Nova Container Platform с настройками по умолчанию используйте представленный далее манифест кастомизации.

▸ Манифест кастомизации

1.1. Установка в Kubernetes

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.

Порядок действий

1. Сохраните полученный манифест и установите его в кластер Kubernetes с помощью Nova Console или *kubectl*.

   Пример вывода:

   BASH
   ```
   kubectl apply -f nova-release-cluster-backup-velero.yaml

   kustomization.kustomize.toolkit.fluxcd.io/nova-release-velero-main created
   ```

2. Проверьте статус кастомизации:

   BASH
   ```
   kubectl get ks nova-release-velero-main -n nova-gitops
   ```

   Пример вывода:

   BASH
   ```
   kubectl get ks nova-release-velero-main -n nova-gitops

   NAME                        AGE   READY   STATUS
   ```

```
nova-release-velero-main   55s   True    Applied revision:
v6.0.1@sha1:86f53cb7e4dbacb29fa42f2c1c9814fa6aec7a07
```

3. Проверьте состояние запущенных компонентов Velero, выполнив команду:

```bash
kubectl get pods -n nova-cluster-backup
```

Пример вывода:

```bash
kubectl get pods -n nova-cluster-backup

NAME                            READY   STATUS    RESTARTS   AGE
velero-7877767f4-zkdbh          1/1     Running   0          22s
node-agent-b4wrh                1/1     Running   0          22s
node-agent-hpcj5                1/1     Running   0          22s
node-agent-tmvvp                1/1     Running   0          22s
node-agent-vhh6b                1/1     Running   0          22s
node-agent-w2qbb                1/1     Running   0          22s
node-agent-xk5l4                1/1     Running   0          22s
node-agent-xlh6k                1/1     Running   0          22s
```

На данном этапе установка модуля с настройками по умолчанию завершена, и вы можете перейти к его настройке.

2. Настройка хранилища резервных копий

В модуле Data Protection ПО Velero поставляется с плагином для подключения к объектному хранилищу, совместимому с Amazon Web Services (AWS) S3. Вы также можете использовать любые S3-совместимые хранилища для подключения к Velero.

Для подключения объектного хранилища в Velero вам необходимо настроить в Kubernetes объект *Secret*, в котором должны быть установлены учетные данные. Как правило, это переменные `aws_access_key_id` и `aws_secret_access_key`. Кроме этого, вам потребуется переопределить точку подключения к объектному хранилищу.

Следуйте инструкциям ниже, чтобы настроить хранилище резервных копий.

2.1. Настройка секрета доступа к объектному хранилищу

Секрет доступа к объектному хранилищу должен быть размещен в среде Kubernetes. Для корректной работы Velero должен быть создан секрет по умолчанию, который будет использоваться для доступа к объектному хранилищу в случаях, когда отдельный секрет явно не указан.

При дальнейшей настройке вы можете создать любое дополнительное количество секретов.

Размещение секрета доступа к объектному хранилищу в StarVault не дает явных преимуществ в безопасности платформы. При этом значительно повышается сложность эксплуатации решения по резервному копированию, особенно, когда резервное копирование выполняется в разные бакеты или хранилища, где требуются отдельные учетные записи.

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ У вас подготовлена учетная запись для доступа к объектному хранилищу.

Порядок действий:

1. Создайте на локальной машине файл, например, `cloud-credentials`. В данном файле необходимо указать учетную запись по умолчанию для подключения к объектному хранилищу.

   Пример вывода:

   BASH
   ```
   cat << EOF > ./cloud-credentials
   [default]
   aws_access_key_id=<AWS_ACCESS_KEY_ID>
   aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
   EOF
   ```

2. Создайте секрет по умолчанию:

   BASH
   ```
   kubectl create secret generic cloud-credentials -n nova-cluster-backup --
   from-file cloud=cloud-credentials
   ```

   Пример вывода:

   BASH
   ```
   kubectl create secret generic cloud-credentials -n nova-cluster-backup --
   from-file cloud=cloud-credentials

   secret/cloud-credentials created
   ```

2.2. Настройка ключа шифрования данных

Резервные копии персистентных данных (файлов) зашифровываются с помощью ключа, хранимого в StarVault. Данный ключ уникален для каждой инсталляции Nova Container Platform. При необходимости, вы можете сменить ключ шифрования следуя процедуре ниже.

Рекомендуется сменить ключ шифрования до настройки каких-либо планов резервного копирования. После настройки ключа шифрования вам необходимо выполнить перезапуск Velero с помощью команды:

```
kubectl -n nova-cluster-backup rollout restart deployment/velero
```

BASH

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ У вас есть токен доступа к хранилищу секретов StarVault с привилегиями `root`.
- ✓ У вас подготовлена учетная запись для доступа к объектному хранилищу.

Порядок действий

1. Подключитесь к StarVault следуя процедуре, описанной в разделе Подключение к StarVault.
2. Перейдите в раздел **Secrets**, выберете секрет `nova-secrets`.
3. В секрете `nova-secrets` перейдите в `credentials`, далее в `nova-velero`.
4. Создайте новую версию секрета, используя кнопку “*Create new version*” измените значение ключа `repository_password` на новое.

2.3. Настройка BackupStorageLocation

Для того, чтобы создать резервную копию, необходимо сначала создать место, в котором будет сохранена резервная копия. Для корректной работы необходимо хотя бы 1 место. Чтобы создать место, необходимо выполнить следующие шаги:

1. Создайте место для хранения резервной копии одним из способов:

   Через CLI

   Подготовьте манифест CR *BackupStorageLocation* и установите его в кластер Kubernetes с помощью Nova Console или *kubectl*.

   ▸ **YAML-манифест**

   Пример вывода:

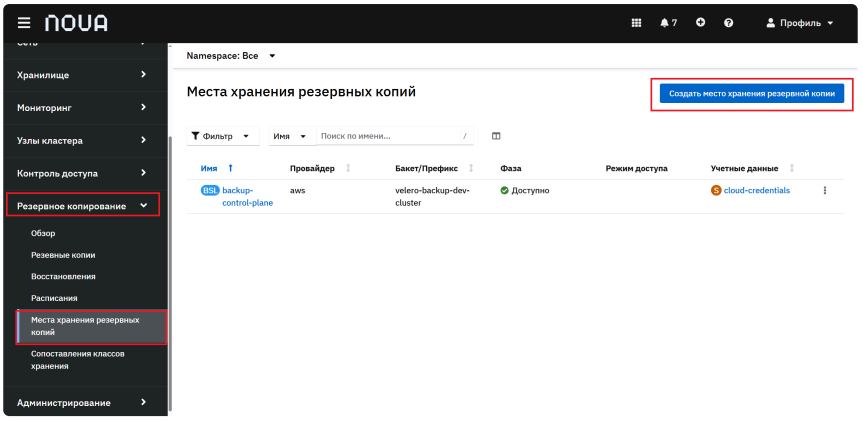
   ```
   kubectl create -f backup-storage-location.yaml

   backupstoragelocation.velero.io/default created
   ```

   BASH

Через веб-интерфейс

1. Перейдите в раздел **Места хранения резервной копии**.
2. Нажмите кнопку **[Создать место хранения резервной копии]**.



3. Далее необходимо задать параметры конфигурации. В верхней части интерфейса выберите один из следующих способов:

   ▸ **Через YAML-манифест**

   ▸ **Через заполнение формы**

4. Нажмите кнопку **[Создать]**.

2. Проверьте статус регистрации объектного хранилища одним из способов:

   ▸ **Через kubectl**

   ▸ **Через Velero CLI**

   ▸ **Через веб-интерфейс**

На данном этапе настройка хранилища резервных копий завершена, и вы можете перейти к настройкам планов резервного копирования.

3. Восстановление резервных копий мастер-узлов

Перед тем, как восстанавливать какой-либо компонент мастер-узла или кластера Kubernetes в Nova Container Platform с помощью резервной копии Velero, необходимо сперва восстановить копию сервиса Nova Backup Daemon, в котором хранится резервная копия мастер-узлов. Кроме этого, поддерживается сценарий восстановления данных из объектного хранилища без доступа к Velero с помощью Kopia.

В разделах ниже описаны обе процедуры восстановления данных.

3.1. Восстановление данных с помощью Velero

При восстановлении данных с помощью Velero в кластер Kubernetes восстанавливается копия сервиса Nova Backup Daemon, в которой находятся резервные копии мастер-узлов. Вы можете перенести данные копии на локальную машину и перейти к восстановлению отдельных компонентов мастер-узлов и среды Kubernetes.

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.
- ✓ Вы настроили утилиту velero для работы с резервными копиями Velero.

Порядок действий

1. Процесс восстановления можно запустить двумя способами:

   Через командную строку

   1. Получите список резервных копий и выберете копию для восстановления:

      ▸ **kubectl**

      ▸ **Velero CLI**

   2. Восстановите выбранную резервную копию в кластер Kubernetes:

      ```
      velero restore create <RESTORE_NAME> \
          --from-backup <BACKUP_NAME> \
          --namespace-mappings nova-cluster-backup:nova-cluster-restore
      ```

      Для восстановления резервной копии используйте следующие данные:

      - `RESTORE_NAME` - имя задания на восстановление резервной копии
      - `BACKUP_NAME` - имя выбранной резервной копии

      ✗ Не изменяйте значение ключа `--namespace-mappings` при создании задания на восстановление резервной копии. Копия восстанавливается в пространство имен `nova-cluster-restore`, сохраняя работоспособность сервиса Nova Backup Daemon в пространстве имен `nova-cluster-backup`.

      Пример вывода:

```
velero restore create restore-control-plane-backup-20240610161525 \
    --from-backup control-plane-backup-20240610161525 \
    --namespace-mappings nova-cluster-backup:nova-cluster-restore

Restore request "restore-control-plane-backup-20240610161525"
submitted successfully.
Run `velero restore describe restore-control-plane-backup-
20240610161525` or `velero restore logs restore-control-plane-backup-
20240610161525` for more details.
```

3. Проверьте статус задания на восстановление резервной копии:

```
velero restore get restore-control-plane-backup-20240610161525
```

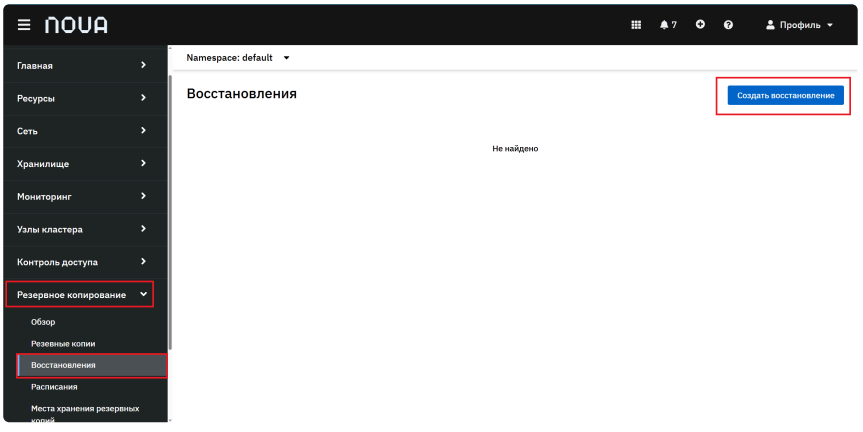
Пример вывода:

```
velero restore get restore-control-plane-backup-20240610161525

NAME                                                BACKUP
STATUS      STARTED                          COMPLETED
ERRORS   WARNINGS   CREATED                          SELECTOR
restore-control-plane-backup-20240610161525   control-plane-backup-
20240610161525   Completed   2024-06-11 15:33:51 +0000 UTC   2024-06-11
15:33:57 +0000 UTC   0        3          2024-06-11 15:33:51 +0000 UTC
<none>
```

Через веб-интерфейс

1. Для создания восстановления перейдите в раздел **Восстановления**.
2. Нажмите на кнопку **[Создать восстановление]**



3. Далее необходимо задать параметры конфигурации. В верхней части интерфейса выберите один из следующих способов:

▸ **Через YAML-манифест**

▸ **Через заполнение формы**

4. Нажмите на кнопку **[Создать]**.

2. Получите список Pod'ов сервиса Nova Backup Daemon в пространстве имен `nova-cluster-restore`:

```
BASH
kubectl get pods -n nova-cluster-restore
```

Пример вывода:

```
BASH
kubectl get pods -n nova-cluster-restore

NAME                        READY   STATUS    RESTARTS   AGE
nova-backup-daemon-2xf9p    1/1     Running   0          78s
nova-backup-daemon-s5zws    1/1     Running   0          78s
nova-backup-daemon-z4lmn    1/1     Running   0          77s
```

3. Скопируйте резервные копии мастер-узлов с каждого из Pod:

```
BASH
PODNAME=<POD_NAME>; \
    for file in $(kubectl exec $PODNAME -n nova-cluster-restore -c backup-
daemon -- ls /opt/backup); \
    do kubectl cp -c backup-daemon nova-cluster-
restore/$PODNAME:/opt/backup/$file $PWD/$file; done
```

В качестве `<POD_NAME>` укажите имя Pod сервиса Nova Backup Daemon.

Пример вывода:

```
BASH
PODNAME=nova-backup-daemon-2xf9p; \
    for file in $(kubectl exec $PODNAME -n nova-cluster-restore -c backup-
daemon -- ls /opt/backup); \
    do kubectl cp -c backup-daemon nova-cluster-
restore/$PODNAME:/opt/backup/$file $PWD/$file; done
```

Файлы резервной копии будут сохранены в текущую директорию на локальной машине пользователя.

4. Проверьте список полученных файлов:

Пример вывода:

```
BASH
ls -la

drwxr-xr-x. 2 root root     4096 Jun 11 19:09 .
drwxr-xr-x. 7 root root     4096 Jun 11 18:39 ..
-rw-r--r--. 1 root root 68989925 Jun 11 19:09 etcd_snapshot_nova-v6.0.1_k8s-
```

```
v1.27.11_2024-06-11_050616.db.tar.gz
-rw-r--r--. 1 root root 33970260 Jun 11 19:09 nova-master-1-nova-
internal_kuberesources_2024-06-11_050616.tar.gz
-rw-r--r--. 1 root root 33971757 Jun 11 19:08 nova-master-2-nova-
internal_kuberesources_2024-06-11_050535.tar.gz
-rw-r--r--. 1 root root 33973476 Jun 11 19:08 nova-master-3-nova-
internal_kuberesources_2024-06-11_050556.tar.gz
-rw-r--r--. 1 root root   375225 Jun 11 19:09 starvault_snapshot_nova-
v6.0.1_2024-06-11_050616.db
```

5. После восстановления резервной копии сервиса Nova Backup Daemon удалите задание на восстановление:

```
velero restore delete <RESTORE_NAME>
```

BASH

В качестве `<RESTORE_NAME>` укажите имя задания на восстановление резервной копии, которое необходимо удалить.

Пример вывода:

```
velero restore delete restore-control-plane-backup-20240610161525

Are you sure you want to continue (Y/N)? y
Request to delete restore "restore-control-plane-backup-20240610161525"
submitted successfully.
The restore will be fully deleted after all associated data (restore files
in object storage) are removed.
```

BASH

Также удалите восстановленный сервис Nova Backup Daemon:

Пример вывода:

```
kubectl delete ds -n nova-cluster-restore nova-backup-daemon
```

BASH

3.2. Восстановление данных с помощью Kopia

Для восстановления данных из объектного хранилища напрямую без участия Velero вам необходимо будет подключиться к репозиторию резервных копий *Kopia* с помощью утилиты *Kopia CLI*. Данный репозиторий инициализируется с помощью Velero автоматически в процессе настройки резервного копирования. Вы сможете получить файлы, которые были подготовлены сервисом Nova Backup Daemon. Имея данные файлы, вы сможете перейти к восстановлению отдельных компонентов мастер-узлов и среды Kubernetes.

Необходимые условия

- ✓ Вы установили утилиту Kopia CLI.
- ✓ У вас есть ключ шифрования резервных копий.

✓ У вас есть доступ к бакету в объектном хранилище, где сохранены резервные копии.

Порядок действий

1. Подключитесь к хранилищу резервных копий с помощью Kopia CLI:

```bash
kopia repository connect s3 \
    --access-key=<AWS_ACCESS_KEY_ID> \
    --secret-access-key=<AWS_SECRET_ACCESS_KEY> \
    --bucket=<BUCKET_NAME> \
    --prefix=kopia/nova-cluster-backup/ \
    --endpoint=<S3_ENDPOINT_URL>

Enter password to open repository:
<ENCRYPTION_PASSWORD>
```

   Для подключения используйте следующие данные:

   - `AWS_ACCESS_KEY_ID` и `AWS_SECRET_ACCESS_KEY` - учетные данные для подключения к объектному хранилищу
   - `BUCKET_NAME` - имя бакета для хранения резервных копий.
   - `S3_ENDPOINT_URL` - точка подключения к объектному хранилищу без указания протокола.

     Пример вывода:

```bash
kopia repository connect s3 \
    --access-key="ASIAIOSFODNN7EXAMPLE" \
    --secret-access-key="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY" \
    --bucket=velero-backup-bucket \
    --prefix=kopia/nova-cluster-backup/ \
    --endpoint=s3.mycompany.local

Enter password to open repository: ********

Connected to repository.
```

2. Получите список идентификаторов снапшотов в хранилище резервных копий:

```bash
kopia snapshot list --all -l
```

   Пример вывода:

```bash
kopia snapshot list –all –l

default@default:/host_pods/1eeb2ac9-36de-4a9c-a1d1-
a0ac4618f6fd/volumes/kubernetes.io~empty-dir/backup-volume 2024-06-11
12:15:40 MSK k702d957ec030e814fbc61bcd63d2d316 34 MB drwxrwxrwx files:1
dirs:1 (latest-3,hourly-3) pins:velero-pin 2024-06-11 13:15:40 MSK
```

```
kf576cc276650d10920e96b48d162ea5a 34 MB drwxrwxrwx files:1 dirs:1 (latest-
2,hourly-2) pins:velero-pin 2024-06-11 14:15:40 MSK
k2d67a575200c57344846b221c36e9bd6 34 MB drwxrwxrwx files:1 dirs:1 (latest-
1,hourly-1,daily-1,weekly-1,monthly-1,annual-1) pins:velero-pin

default@default:/host_pods/4f4aaa17-da1a-4c30-a587-
535604a991b1/volumes/kubernetes.io~empty-dir/backup-volume 2024-06-11
12:15:57 MSK kfcafeadd427239980e47f4eceeed244a 94.2 MB drwxrwxrwx files:3
dirs:1 (latest-3,hourly-3) pins:velero-pin 2024-06-11 13:15:58 MSK
k31d18913f2a147722f292609b987ad15 93.3 MB drwxrwxrwx files:3 dirs:1 (latest-
2,hourly-2) pins:velero-pin 2024-06-11 14:15:58 MSK
k570cf06adf55dbf2e87b2959f148c3b2 92.3 MB drwxrwxrwx files:3 dirs:1 (latest-
1,hourly-1,daily-1,weekly-1,monthly-1,annual-1) pins:velero-pin

default@default:/host_pods/fd076d09-d8f6-480e-addf-
03f3fa33f633/volumes/kubernetes.io~empty-dir/backup-volume 2024-06-11
12:15:47 MSK kf3872b5f18ef1fd536fe1d75cb2fa160 34 MB drwxrwxrwx files:1
dirs:1 (latest-3,hourly-3) pins:velero-pin 2024-06-11 13:15:47 MSK
k7113f38a1b63f03caf4ca1cb179550fb 34 MB drwxrwxrwx files:1 dirs:1 (latest-
2,hourly-2) pins:velero-pin 2024-06-11 14:15:47 MSK
kc950a4ee5808557fe2c3d01df22ff4e3 34 MB drwxrwxrwx files:1 dirs:1 (latest-
1,hourly-1,daily-1,weekly-1,monthly-1,annual-1) pins:velero-pin +
```

В выводе команды отображен список резервных копий томов `backup-volume` для трех разных Pod сервиса Nova Backup Daemon, имеющих UID:

- `1eeb2ac9-36de-4a9c-a1d1-a0ac4618f6fd`
- `4f4aaa17-da1a-4c30-a587-535604a991b1`
- `fd076d09-d8f6-480e-addf-03f3fa33f633`

Для того, чтобы проверить соответствие данных резервных копий узлам в Kubernetes, вы можете выполнить команду:

BASH

```
kubectl get pod -l app.kubernetes.io/component=nova-backup-daemon -n
nova-cluster-backup \
    -o custom-
columns=PodName:.metadata.name,UID:.metadata.uid,NODE:.spec.nodeName
```

Пример вывода:

BASH

```
kubectl get pod -l app.kubernetes.io/component=nova-backup-daemon -n
nova-cluster-backup \
    -o custom-
columns=PodName:.metadata.name,UID:.metadata.uid,NODE:.spec.nodeName

PodName                    UID                                   NODE
nova-backup-daemon-2xf9p   1eeb2ac9-36de-4a9c-a1d1-a0ac4618f6fd   nova-
master-2.mycompany.local
nova-backup-daemon-s5zws   fd076d09-d8f6-480e-addf-03f3fa33f633   nova-
```

```
master-3.mycompany.local
nova-backup-daemon-z4lmn   4f4aaa17-da1a-4c30-a587-535604a991b1   nova-
master-1.mycompany.local
```

3. Восстановите необходимую резервную копию:

```bash
kopia snapshot restore <SNAPSHOT_ID>
```

В качестве `SNAPSHOT_ID` укажите тот идентификатор снапшота, временная метка которого соответствует требуемой точке восстановления.

Пример вывода:

```bash
kopia snapshot restore k570cf06adf55dbf2e87b2959f148c3b2

Restoring to local filesystem (/root/k570cf06adf55dbf2e87b2959f148c3b2) with
parallelism=8...
Processed 4 (92.3 MB) of 3 (92.3 MB) 69.9 MB/s (100.0%) remaining 0s.
Processed 4 (92.3 MB) of 3 (92.3 MB) 69.9 MB/s (100.0%) remaining 0s.
Restored 3 files, 1 directories and 0 symbolic links (92.3 MB).
```

4. Проверьте восстановленные данные:

Пример вывода:

```bash
ls -la /root/k570cf06adf55dbf2e87b2959f148c3b2/
итого 90160
drwxrwxrwx.  2 root root     4096 Jun 11 14:15 .
dr-xr-x---. 89 root root     4096 Jun 11 14:50 ..
-rw-r--r--.  1 root root 58044113 Jun 11 14:15 etcd_snapshot_nova-
v6.0.1_k8s-v1.27.11_2024-06-11_111549.db.tar.gz
-rw-------.  1 root root 33970429 Jun 11 14:15 nova-master-1-nova-
internal_kuberesources_2024-06-11_111549.tar.gz
-rw-------.  1 root root   292009 Jun 11 14:15 starvault_snapshot_nova-
v6.0.1_2024-06-11_111549.db
```

Для архивов резервных копий применяется следующая схема именования:

- Имя архива резервной копии Etcd имеет формат `etcd_snapshot_nova-<Версия Nova>_k8s-<Версия Kubernetes>_<Время создания копии>.db.tar.gz`.
- Имена архивов резервных копий конфигураций мастер-узлов имеют формат `<Имя узла в Kubernetes>_kuberesources_<Время создания копии>.tar.gz`.
- Имя архива резервной копии StarVault имеет формат `starvault_snapshot_nova-<Версия Nova>_<Время создания копии>1.db`.

Резервная копия только одного из мастер-узлов включает копии Etcd и StarVault.

5. Повторите действия по восстановлению для оставшихся Pod’ов сервиса Nova Backup Daemon.
6. Отключитесь от хранилища резервных копий:

```bash
kopia repository disconnect
```

В настоящее время в Velero используется общий статический ключ шифрования для всех создаваемых репозиториев резервного копирования. Это означает, что любой, кто имеет доступ к вашему хранилищу резервных копий, может расшифровать ваши резервные копии. Обязательно ограничьте доступ к хранилищу резервных копий соответствующим образом.

Восстановление данных на мастер-узлах

Данный раздел содержит статьи полезные для восстановления данных Nova Container Platform.

1. Восстановление Etcd

В данном разделе представлены процедуры восстановления работоспособности кластера Etcd.

Повреждение данных Etcd может произойти по многочисленным причинам, например:

- При выходе из строя узлов виртуализации, повреждении серверного оборудования или его компонентов, программных сбоев или потери сетевой связанности.
- При отказе программных и аппаратных систем хранения данных и ошибках записи данных на файловую систему.
- При “жестком” отключении мастер-узлов.
- При удалении критически важных данных Etcd.

Как правило, при повреждении базы данных сервис Etcd на узле переходит из состояния `active` в состояние `activating/failed`, а в системе мониторинга регистрируются события `etcdMembersDown`. При сохранении кворума кластера Etcd сервер Kubernetes API продолжает работать, поскольку выполняет на своей стороне балансировку запросов. При этом после повреждения данных мастер-узел не сможет самостоятельно подключиться к прежнему кластеру, и может потребоваться выполнить его восстановление.

После оценки проблем с кластером Etcd воспользуйтесь наиболее подходящей процедурой восстановления или обратитесь в техническую поддержку.

1.1. Общие практики

Etcd не всегда требует наличия резервной копии (снапшота) для восстановления одного участника кластера.

Существует два основных сценария решения проблем в зависимости от того, что произошло с участником кластера Etcd:

- *Перезапуск неисправного участника кластера*: Если участник кластера перестал отвечать на запросы, сервис Etcd находится в состоянии `activating/failed` или `active`, но при этом не отвечает, то необходимо начать решение проблемы с

перезапуска сервиса Etcd на узле. После перезапуска узел должен автоматически добавиться в существующих кластер и синхронизировать свое состояние с остальными участниками.

- *Восстановление неисправного участника кластера*: Если проблема не решается перезапуском сервиса Etcd, а данные Etcd повреждены, вы можете полностью восстановить узел. Для этого вам необходимо удалить неисправного участника из кластера Etcd, удалить поврежденные данные и передобавить участника в кластер. После подключения восстановленный участник кластера получит актуальную копию данных и синхронизирует свое состояние с остальными участниками.

Восстановление полного кластера Etcd из резервной копии необходимо в том случае, когда 2 и более участников кластера неисправны. В данной ситуации также перестает отвечать сервер Kubernetes API, поэтому важно иметь возможность получить резервные копии Etcd независимо от работоспособности Kubernetes API. Данный сценарий требует полной остановки сервисов Etcd на каждом мастер-узле, восстановления резервной копии на каждый мастер-узел и повторный запуск кластера Etcd.

При полном восстановлении кластера Etcd выполняется откат всех данных Kubernetes на время последней успешной резервной копии. При этом все клиенты Kubernetes (kubelet, CNI, CSI и др.) могут испытывать проблемы и конфликты с локальными данными при подключении к восстановленному кластеру. Это означает, что после успешного восстановления и запуска кластера Etcd могут потребоваться ручные действия для дальнейшего восстановления работоспособности Kubernetes.

1.2. Восстановление одного неисправного узла Etcd

В данном разделе описывается процесс восстановления отдельного неисправного участника кластера Etcd. Процедура подразумевает, что кластер Kubernetes остается работоспособным, однако один из участников кластера Etcd перестает отвечать на запросы.

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.
- ✓ У вас есть доступ по SSH на мастер-узлы кластера.

Порядок действий

1. Подключитесь по SSH на один из исправных мастер-узлов.
2. Установите переменные окружения для подключения к Etcd:

```bash
source <(cat /etc/etcd.env | grep -v "^#\|^$" | sed 's/^/export /')
```

3. Определите неисправный узел Etcd.

```
/opt/etcd/etcdctl endpoint health --cluster --write-out=table
```

Пример

```
{"level":"warn","ts":"2024-06-
14T12:22:37.426365+0300","logger":"client","caller":"v3@v3.5.10/retry_interc
eptor.go:62","msg":"retrying of unary invoker failed","target":"etcd-
endpoints://0xc000325340/172.31.100.105:2379","attempt":0,"error":"rpc
error: code = DeadlineExceeded desc = latest balancer error: last connection
error: connection error: desc = \"transport: Error while dialing: dial tcp
172.31.100.105:2379: connect: connection refused\""}

+----------------------------+--------+-------------+--------------------
------+
|          ENDPOINT          | HEALTH |     TOOK    |           ERROR
|
+----------------------------+--------+-------------+--------------------
------+
| https://172.31.100.97:2379  |   true |   7.856832ms |
|
| https://172.31.100.106:2379 |   true |   8.094994ms |
|
| https://172.31.100.105:2379 |  false | 5.000964147s | context deadline
exceeded |
+----------------------------+--------+-------------+--------------------
------+
Error: unhealthy cluster
```

В примере выше неисправным является узел, имеющий адрес подключения `https://172.31.100.105:2379`.

4. Получите дополнительные сведения о неисправном узле Etcd:

```
/opt/etcd/etcdctl member list --write-out=table
```

Пример

```
/opt/etcd/etcdctl member list –write-out=table

+------------------+---------+-------+---------------------------+--------
--------------------+------------+
|        ID        | STATUS  | NAME  |         PEER ADDRS          |
CLIENT ADDRS         | IS LEARNER |
+------------------+---------+-------+---------------------------+--------
--------------------+------------+
| 51fc1c73bfe7f25f | started | etcd3 | https://172.31.100.106:2380 |
```

```
https://172.31.100.106:2379 |      false |
| 56be76cb75b1242d | started | etcd1 | https://172.31.100.97:2380  |
https://172.31.100.97:2379  |      false |
| e42bdaa6ace5c691 | started | etcd2 | https://172.31.100.105:2380 |
https://172.31.100.105:2379 |      false |
+------------------+---------+-------+-----------------------------+--------
--------------------+------------+
```

Сохраните информацию о неисправном узле:

- `ID` - идентификатор узла,
- `NAME` - имя узла,
- `PEER ADDR` - адрес подключения узла.

5. Удалите неисправный узел из кластера Etcd:

```
BASH
/opt/etcd/etcdctl member remove <ID>
```

Пример

```
BASH
/opt/etcd/etcdctl member remove e42bdaa6ace5c691

Member e42bdaa6ace5c691 removed from cluster  dae05cd826ec589
```

6. Добавьте неисправный узел повторно в кластер:

```
BASH
/opt/etcd/etcdctl member add <NAME> --peer-urls=<PEER ADDR>
```

Пример

```
BASH
/opt/etcd/etcdctl member add etcd2 --peer-urls=https://172.31.100.105:2380

Member 4cafbd8e176044a1 added to cluster  dae05cd826ec589

ETCD_NAME="etcd2"
ETCD_INITIAL_CLUSTER="etcd2=https://172.31.100.105:2380,etcd3=https://172.31
.100.106:2380,etcd1=https://172.31.100.97:2380"
ETCD_INITIAL_ADVERTISE_PEER_URLS="https://172.31.100.105:2380"
ETCD_INITIAL_CLUSTER_STATE="existing"
```

7. Проверьте список участников кластера Etcd:

```
BASH
/opt/etcd/etcdctl member list --write-out=table
```

Пример

```
BASH
/opt/etcd/etcdctl member list --write-out=table

+------------------+-----------+-------+---------------------------+------
----------------------+------------+
|        ID        |  STATUS   | NAME  |         PEER ADDRS          |
CLIENT ADDRS         | IS LEARNER |
+------------------+-----------+-------+---------------------------+------
----------------------+------------+
| 4cafbd8e176044a1 | unstarted |       | https://172.31.100.105:2380 |
|      false |
| 51fc1c73bfe7f25f |   started | etcd3 | https://172.31.100.106:2380 |
https://172.31.100.106:2379 |      false |
| 56be76cb75b1242d |   started | etcd1 |  https://172.31.100.97:2380 |
https://172.31.100.97:2379 |      false |
+------------------+-----------+-------+---------------------------+------
----------------------+------------+
```

Вновь добавленный узел находится в состоянии `unstarted`.

8. Подключитесь на неисправный узел по SSH и проверьте статус сервиса Etcd:

```
BASH
systemctl status etcd
```

Пример

```
BASH
systemctl status etcd

● etcd.service - etcd
Loaded: loaded (/etc/systemd/system/etcd.service; enabled; vendor preset:
disabled)
Active: activating (auto-restart) (Result: exit-code) since Thu 2024-06-13
19:01:42 MSK; 8s ago
Process: 243655 ExecStart=/opt/etcd/etcd (code=exited, status=2)
Main PID: 243655 (code=exited, status=2)
```

9. Проверьте журнал событий сервиса Etcd на неисправном узле:

```
BASH
journalctl -u etcd
```

Пример

```
BASH
journalctl -u etcd

Jun 13 19:03:14 nova-master-3.mycompany.local etcd[244118]: \
{"`level`":"`info`","`ts`":"`2024-06-
13T19:03:14.605421+0300`","`caller`":"`embed/etcd.go:309`","`msg`":"`startin
g an etcd server`","`etcd-version`":"`3.5.10`","`git-sha`"> Jun 13 19:03:14
nova-master-3.mycompany.local etcd[244118]: \
```

```
{"`level`":"`warn`","`ts`":"`2024-06-
13T19:03:14.605518+0300`","`caller`":"`fileutil/fileutil.go:53`","`msg`":"`c
heck file permission`","`error`":"`directory "/var/lib> Jun 13 19:03:14
nova-master-3.mycompany.local etcd[244118]: \
{`"level”:"`panic`","`ts`":"`2024-06-
13T19:03:14.607293+0300`","`caller`":"`backend/backend.go:189`","`msg`":"`fa
iled to open database`","`path`":“/var/lib/etcd/membe> Jun 13 19:03:14 nova-
master-3.mycompany.local etcd[244118]: panic: failed to open database Jun 13
19:03:14 nova-master-3.mycompany.local etcd[244118]: goroutine 110
[running]: Jun 13 19:03:14 nova-master-3.mycompany.local etcd[244118]:
go.uber.org/zap/zapcore.(*CheckedEntry).Write(0xc000108840, \{0xc000532580,
0x2, 0x2}) Jun 13 19:03:14 nova-master-3.mycompany.local etcd[244118]:
go.uber.org/zap@v1.17.0/zapcore/entry.go:234 +0x49b
```

10. Остановите сервис Etcd на неисправном узле:

```
BASH
systemctl stop etcd
```

11. Удалите директорию с данными Etcd на неисправном узле:

```
BASH
rm -rf /var/lib/etcd
```

12. Измените параметр состояния кластера для запуска сервиса Etcd:

```
BASH
sed -i
's/ETCD_INITIAL_CLUSTER_STATE=new/ETCD_INITIAL_CLUSTER_STATE=existing/g'
/etc/etcd.env
```

13. Запустите сервис Etcd на неисправном узле:

```
BASH
systemctl start etcd
```

14. Проверьте состояние сервиса Etcd:

```
BASH
systemctl status etcd
```

Пример

```
BASH
● etcd.service - etcd
Loaded: loaded (/etc/systemd/system/etcd.service; enabled; vendor preset:
disabled)
Active: active (running) since Fri 2024-06-14 12:37:55 MSK; 30s ago
Main PID: 96408 (etcd)
    Tasks: 9 (limit: 102296)
Memory: 221.6M
CGroup: /system.slice/etcd.service
        └─96408 /opt/etcd/etcd
```

```
Jun 14 12:37:55 nova-master-2.mycompany.local etcd[96408]:
{"level":"info","ts":"2024-06-
14T12:37:55.065848+0300","caller":"etcdmain/main.go:44","msg":"notifying
init daemon"}
Jun 14 12:37:55 nova-master-2.mycompany.local etcd[96408]:
{"level":"info","ts":"2024-06-
14T12:37:55.065967+0300","caller":"etcdmain/main.go:50","msg":"successfully
notified init daemon"}
Jun 14 12:37:55 nova-master-2.mycompany.local systemd[1]: Started etcd.
```

Статус сервиса Etcd должен быть `active`.

15. Проверьте список участников кластера Etcd:

```
BASH
/opt/etcd/etcdctl member list --write-out=table

+-----------------+---------+-------+---------------------------+--------
--------------------+------------+
|        ID        | STATUS  | NAME  |         PEER ADDRS          |
CLIENT ADDRS         | IS LEARNER |
+-----------------+---------+-------+---------------------------+--------
--------------------+------------+
| 4cafbd8e176044a1 | started | etcd2 | https://172.31.100.105:2380 |
https://172.31.100.105:2379 |      false |
| 51fc1c73bfe7f25f | started | etcd3 | https://172.31.100.106:2380 |
https://172.31.100.106:2379 |      false |
| 56be76cb75b1242d | started | etcd1 |  https://172.31.100.97:2380 |
https://172.31.100.97:2379 |      false |
+-----------------+---------+-------+---------------------------+--------
--------------------+------------+
```

Вновь добавленный узел находится в состоянии `started`.

16. Проверьте статус участников кластера Etcd:

```
BASH
/opt/etcd/etcdctl endpoint status --cluster --write-out=table
```

Пример

```
BASH
/opt/etcd/etcdctl endpoint status --cluster --write-out=table

+---------------------------+-----------------+---------+---------+------
-----+------------+-----------+-----------+-------------------+--------+
|          ENDPOINT           |        ID        | VERSION | DB SIZE | IS
LEADER | IS LEARNER | RAFT TERM | RAFT INDEX | RAFT APPLIED INDEX | ERRORS |
+---------------------------+-----------------+---------+---------+------
-----+------------+-----------+-----------+-------------------+--------+
| https://172.31.100.105:2379 | 4cafbd8e176044a1 |  3.5.10 |  151 MB |
false |      false |         5 |     376222 |            376222 |        |
```

```
| https://172.31.100.106:2379 | 51fc1c73bfe7f25f |  3.5.10 |  151 MB |
false |      false |         5 |     376222 |             376222 |        |
|  https://172.31.100.97:2379 | 56be76cb75b1242d |  3.5.10 |  151 MB |
true |      false |         5 |     376222 |             376222 |        |
+-----------------------------+------------------+---------+---------+------
-----+-----------+-----------+-----------+--------------------+--------+
```

1.3. Восстановление кластера узлов Etcd

В данном разделе описывается процесс восстановления полного кластера Etcd. Процедура подразумевает, что кластер Kubernetes не работоспособен, а сервер Kubernetes API не отвечает на запросы. Вам потребуется наличие актуальной резервной копии для восстановление кластера в прежнее состояние.

Восстановление кластера Etcd в исходное состояние с помощью резервной копии является крайним случаем при восстановлении работоспособности Kubernetes. Если сервер Kubernetes API отвечает на запросы, значит Etcd доступен и полное восстановление не требуется. Достаточно восстановить неисправного участника кластера.

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.
- ✓ У вас есть доступ по SSH на мастер-узлы кластера.
- ✓ У вас есть актуальная резервная копия мастер-узлов.
- ✓ У вас есть ключ шифрования резервных копий, если резервное копирование выполнялось с помощью модуля Data Protection.

Если вы используете модуль Data Protection, то восстановите актуальную резервную копию Etcd из последней доступной резервной копии с помощью *Kopia* следуя процедуре Восстановление резервных копий мастер-узлов). Если резервные копии хранятся на NFS-хранилище, то вам будет необходимо скопировать их на каждый мастер-узел.

При восстановлении кластера Etcd вам потребуется открыть 3 SSH-сессии (по одной на каждый мастер-узел).

Информация

В качестве примера в руководстве используются следующие имена и IP-адреса мастер-узлов:

```
NAME                           InternalIP
nova-master-1.mycompany.local   172.31.100.97
nova-master-2.mycompany.local   172.31.100.105
nova-master-3.mycompany.local   172.31.100.106
```

Резервная копия Etcd располагается на каждом мастер-узле в директории `/tmp`.

Порядок действий

1. Подключитесь по SSH к каждому мастер-узлу кластера Kubernetes.
2. Проверьте статус сервисов Etcd на каждом мастер-узле и остановите сервис, если он запущен:

   ```
   systemctl status etcd
   systemctl stop etcd
   ```

   Пример

   ```
   systemctl status etcd
   systemctl stop etcd

   ● etcd.service - etcd
   Loaded: loaded (/etc/systemd/system/etcd.service; enabled; vendor preset:
   disabled)
   Active: activating (auto-restart) (Result: exit-code) since Mon 2024-06-17
   16:47:47 MSK; 2s ago
   Process: 134875 ExecStart=/opt/etcd/etcd (code=exited, status=2)
   Main PID: 134875 (code=exited, status=2)
   ```

3. Убедитесь, что параметр состояния кластера `ETCD_INITIAL_CLUSTER_STATE` имеет значение `new` на каждом мастер-узле:

   ```
   cat /etc/etcd.env  | grep ETCD_INITIAL_CLUSTER_STATE

   ETCD_INITIAL_CLUSTER_STATE=new
   ```

   Если значение отличается, измените его на `new` в файле `/etc/etcd.env`.

4. Установите переменные окружения для подключения к Etcd на первом мастер-узле:

   ```
   source <(cat /etc/etcd.env | grep -v "^#\|^$" | sed 's/^/export /')
   ```

5. Удалите директорию с данными Etcd на каждом мастер-узле:

   ```
   rm -rf /var/lib/etcd
   ```

6. Распакуйте резервную копию Etcd на каждом мастер-узле:

```
cd /tmp/ && tar -zxvf /tmp/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-14_081538.db.tar.gz
```

BASH

> Достаточно использовать один снапшот Etcd и скопировать его на каждый мастер-узел.

7. Выполните восстановление снапшота Etcd на каждом мастер-узле:

```
ETCDCTL_API=3 /opt/etcd/etcdutl snapshot restore \
    --data-dir=/var/lib/etcd \
    --name=<MEMBER_NAME> \
    --initial-cluster=etcd1=https://<MEMBER_1_IP_ADDRESS>:2380,etcd2=https://<MEMBER_2_IP_ADDRESS>:2380,etcd3=https://<MEMBER_3_IP_ADDRESS>:2380 \
    --initial-cluster-token=k8s_etcd \
    --initial-advertise-peer-urls=https://<MEMBER_1_IP_ADDRESS>:2380 \
    /tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-14_081538.db
```

BASH

где `MEMBER_X_IP_ADDRESS` - IP-адрес мастер-узла.

Пример

Пример команды для первого мастер-узла будет выглядеть следующим образом:

```
ETCDCTL_API=3 /opt/etcd/etcdutl snapshot restore \
    --data-dir=/var/lib/etcd \
    --name=etcd1 \
    --initial-cluster=etcd1=https://172.31.100.97:2380,etcd2=https://172.31.100.105:2380,etcd3=https://172.31.100.106:2380 \
    --initial-cluster-token=k8s_etcd \
    --initial-advertise-peer-urls=https://172.31.100.97:2380 \
    /tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-14_081538.db

    2024-06-19T20:02:10+03:00   info    snapshot/v3_snapshot.go:260 restoring snapshot  {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal", "data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
    2024-06-19T20:02:11+03:00   info    membership/store.go:141 Trimming membership information from the backend...
    2024-06-19T20:02:11+03:00   info    membership/cluster.go:421   added member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-peer-id": "4c357ea1a12149d0", "added-peer-peer-urls":
```

BASH

```
["https://172.31.100.97:2380"]}
    2024-06-19T20:02:11+03:00   info    membership/cluster.go:421   added
member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-
peer-id": "e42bdaa6ace5c691", "added-peer-peer-urls":
["https://172.31.100.105:2380"]}
    2024-06-19T20:02:11+03:00   info    membership/cluster.go:421   added
member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-
peer-id": "e7b44abd88b7fad0", "added-peer-peer-urls":
["https://172.31.100.106:2380"]}
    2024-06-19T20:02:11+03:00   info    snapshot/v3_snapshot.go:287 restored
snapshot   {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal",
"data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
```

Пример команды для второго мастер-узла будет выглядеть следующим образом:

BASH

```
ETCDCTL_API=3 /opt/etcd/etcdutl snapshot restore \
    --data-dir=/var/lib/etcd \
    --name=etcd2 \
    --initial-
cluster=etcd1=https://172.31.100.97:2380,etcd2=https://172.31.100.105:2380,e
tcd3=https://172.31.100.106:2380 \
    --initial-cluster-token=k8s_etcd \
    --initial-advertise-peer-urls=https://172.31.100.105:2380 \
    /tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-
14_081538.db

    2024-06-19T20:02:48+03:00   info    snapshot/v3_snapshot.go:260
restoring snapshot  {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal",
"data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
    2024-06-19T20:02:48+03:00   info    membership/store.go:141 Trimming
membership information from the backend...
    2024-06-19T20:02:49+03:00   info    membership/cluster.go:421   added
member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-
peer-id": "4c357ea1a12149d0", "added-peer-peer-urls":
["https://172.31.100.97:2380"]}
    2024-06-19T20:02:49+03:00   info    membership/cluster.go:421   added
member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-
peer-id": "e42bdaa6ace5c691", "added-peer-peer-urls":
["https://172.31.100.105:2380"]}
    2024-06-19T20:02:49+03:00   info    membership/cluster.go:421   added
member    {"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-
peer-id": "e7b44abd88b7fad0", "added-peer-peer-urls":
["https://172.31.100.106:2380"]}
    2024-06-19T20:02:49+03:00   info    snapshot/v3_snapshot.go:287 restored
snapshot   {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal",
"data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
```

Пример команды для третьего мастер-узла будет выглядеть следующим образом:

```
BASH
ETCDCTL_API=3 /opt/etcd/etcdutl snapshot restore \
    --data-dir=/var/lib/etcd \
    --name=etcd3 \
    --initial-
cluster=etcd1=https://172.31.100.97:2380,etcd2=https://172.31.100.105:2380,e
tcd3=https://172.31.100.106:2380 \
    --initial-cluster-token=k8s_etcd \
    --initial-advertise-peer-urls=https://172.31.100.106:2380 \
    /tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-v1.27.11_2024-06-
14_081538.db

2024-06-19T20:04:01+03:00   info    snapshot/v3_snapshot.go:260 restoring
snapshot  {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal",
"data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
2024-06-19T20:04:02+03:00   info    membership/store.go:141 Trimming
membership information from the backend...
2024-06-19T20:04:02+03:00   info    membership/cluster.go:421   added member
{"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-peer-id":
"4c357ea1a12149d0", "added-peer-peer-urls": ["https://172.31.100.97:2380"]}
2024-06-19T20:04:02+03:00   info    membership/cluster.go:421   added member
{"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-peer-id":
"e42bdaa6ace5c691", "added-peer-peer-urls": ["https://172.31.100.105:2380"]}
2024-06-19T20:04:02+03:00   info    membership/cluster.go:421   added member
{"cluster-id": "dae05cd826ec589", "local-member-id": "0", "added-peer-id":
"e7b44abd88b7fad0", "added-peer-peer-urls": ["https://172.31.100.106:2380"]}
2024-06-19T20:04:02+03:00   info    snapshot/v3_snapshot.go:287 restored
snapshot   {"path": "/tmp/opt/backup/etcd_snapshot_nova-v6.0.1_k8s-
v1.27.11_2024-06-14_081538.db", "wal-dir": "/var/lib/etcd/member/wal",
"data-dir": "/var/lib/etcd", "snap-dir": "/var/lib/etcd/member/snap"}
```

8. Запустите сервис Etcd на каждом мастер-узле подряд и проверьте его статус:

```
BASH
systemctl start etcd
systemctl status etcd
```

Пример

```
BASH
systemctl start etcd
systemctl status etcd

● etcd.service - etcd
Loaded: loaded (/etc/systemd/system/etcd.service; enabled; vendor preset:
disabled)
Active: active (running) since Wed 2024-06-19 20:05:17 MSK; 59s ago
Main PID: 134154 (etcd)
    Tasks: 10 (limit: 102296)
```

```
Memory: 56.8M
CGroup: /system.slice/etcd.service
        └─134154 /opt/etcd/etcd
```

Статус сервиса Etcd должен быть `active`.

9. Проверьте статус участников кластера Etcd:

```
BASH
/opt/etcd/etcdctl endpoint status --cluster --write-out=table
```

Пример

```
BASH
/opt/etcd/etcdctl endpoint status --cluster --write-out=table

+----------------------------+------------------+---------+---------+------
-----+------------+-----------+------------+--------------------+--------+
|          ENDPOINT          |        ID        | VERSION | DB SIZE | IS
LEADER | IS LEARNER | RAFT TERM | RAFT INDEX | RAFT APPLIED INDEX | ERRORS |
+----------------------------+------------------+---------+---------+------
-----+------------+-----------+------------+--------------------+--------+
|  https://172.31.100.97:2379 | 4c357ea1a12149d0 |  3.5.10 |  151 MB |
true |      false |         2 |       6489 |               6489 |        |
| https://172.31.100.105:2379 | e42bdaa6ace5c691 |  3.5.10 |  151 MB |
false |      false |         2 |       6489 |               6489 |        |
| https://172.31.100.106:2379 | e7b44abd88b7fad0 |  3.5.10 |  151 MB |
false |      false |         2 |       6489 |               6489 |        |
+----------------------------+------------------+---------+---------+------
-----+------------+-----------+------------+--------------------+--------+
```

На данном этапе восстановление кластера Etcd завершено. Далее необходимо выполнить проверку работоспособности среды Kubernetes.