

# Обзор

На текущий момент в Nova Container Platform можно установить дополнительный модуль OpenSearch и компоненты для агрегации логов Logging operator. Logging operator добавляется в кластер при его установке.

## 1. OpenSearch

OpenSearch — это легко масштабируемая система поисковых и аналитических инструментов с открытым исходным кодом. Она была ответвлена от кодовой базы Elasticsearch 7.10.2. Из OpenSearch убрали компоненты, распространяемые не под лицензией Apache 2.0. В систему включены: движок хранения и поиска, веб-интерфейс, среда визуализации данных OpenSearch Dashboards, а также дополнения, которые ранее поставлялись в продукте Open Distro for Elasticsearch.

## 2. Logging operator

Logging operator управляет сборщиками и форвардерами логов в инфраструктуре логирования, а также правилами маршрутизации, которые определяют адресатов различных логов. Вы можете фильтровать и обрабатывать входящие логи, используя настраиваемый ресурс `_flow` форвардера логов для их маршрутизации к соответствующему `output`. `Outputs` - это места назначения, куда вы хотите отправлять логи, например, OpenSearch или сервер Syslog. Вы также можете определить места назначения (`outputs`) и потоки логов (`flows`) для всего кластера, например, использовать `output`, на который могут ссылаться пользователи пространства имен, но не могут его изменять.

## 3. Содержание раздела

- Custom Resource Definitions
- Opensearch
  - Планирование установки и системные требования
  - Установка модуля OpenSearch
  - Запрет на удаление индексов
  - Настройка уведомлений в Opensearch
- Logging Operator

- [Установка Logging Operator](#)
  - [Примеры использования Logging Operator](#)
- 

2025 orionsoft. Все права защищены.

# Управление узлами платформы

## 1. Общие сведения о процессах управления узлами

В процессе эксплуатации кластеров Nova Container Platform вы можете столкнуться с различными задачами по управлению узлами кластера, например:

- Расширение количества узлов
- Изменение объема выделенных ресурсов
- Переустановка компонентов платформы на узле кластера
- Перевод узла в режим обслуживания

Текущая конфигурация узлов кластера определена в манифесте `infrastructures.config.nova-platform.io/cluster`, ознакомиться с которым вы можете выполнив команду:

```
kubectl describe infrastructures.config.nova-platform.io cluster
```

BASH | □

### ► Пример



Изменение манифеста `infrastructures.config.nova-platform.io/cluster` вручную с помощью `kubectl`, `curl` и подобных утилит не поддерживается. Взаимодействие с объектом осуществляется только с помощью утилиты `nova-ctl`.

## 2. Горизонтальное масштабирование кластера

Горизонтальное масштабирование заключается в добавлении новых узлов в кластер Kubernetes и выполняется с помощью утилиты `nova-ctl`.

### 2.1. Масштабирование кластера, установленного универсальным методом (UPI)

*Необходимые условия*

- ✓ Вы подготовили виртуальный или физический узел для добавления в кластер согласно разделу [Подготовка к установке платформы Nova Container Platform](#).

- ✓ На вашем локальном компьютере установлена утилита `nova-ctl`.
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера ( `cluster-admin` ).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl` .
- ✓ У вас есть токен доступа к хранилищу секретов StarVault с привилегиями `root` .



При подготовке нового узла используйте такие же имя пользователя и открытую часть ключа SSH, как и на остальных узлах кластера, которые использовались на [подготовительном этапе](#).

## Процедура

1. Запустите процесс масштабирования узлов кластера с помощью команды:

```
BASH | nova-ctl scale --ssh-user <имя_пользователя> --ssh-key <закрытый ключ SSH>
```



В качестве аргументов `--ssh-key` и `--ssh-user` укажите информацию, использованную [на этапе конфигурации ключевой пары SSH](#).

2. Далее для временного редактирования будет открыт файл конфигурации кластера в текстовом редакторе `vi`.
3. Добавьте новый узел в блок конфигурации `ClusterNodes` и сохраните изменения.

## Пример

```
BASH | nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem
spec:
...
clusterNodes:
...
worker: # Роль узла в кластере Kubernetes.
- hostGroup: "worker" # Существующий узел в кластере Kubernetes.
  networkSpec:
    hostname: "worker01.mycompany.local"
    ip: "172.31.101.26"
    state: "present"
...
- hostGroup: "worker" # Добавляемый узел в кластер Kubernetes.
  networkSpec:
    hostname: "worker02.mycompany.local"
    ip: "172.31.101.27"
    state: "present"
```

4. На запрос `Enter Vault root token:` введите токен доступа к хранилищу секретов StarVault с привилегиями `root` , после чего начнется процесс добавления узла в

кластер.

5. Дождитесь сообщения об успешном выполнении операции.

*Пример*

```
BASH | □
■ Validating cluster nodes... done
■ Validating license... done
■ Cleaning up... done
■ Preparing cluster nodes... done

🚀 Cluster is successfully scaled.
```

6. Проверьте состояние узлов кластера Kubernetes после успешного масштабирования согласно руководству.

## 2.2. Масштабирование кластера, установленного в среде zVirt (IPI)

*Необходимые условия*

- ✓ На вашем локальном компьютере установлена утилита `nova-ctl`.
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера (`cluster-admin`).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl`.
- ✓ У вас есть токен доступа к хранилищу секретов StarVault с привилегиями `root`.



Если вы решили использовать новый шаблон для виртуальной машины, то при его подготовке используйте такие же имя пользователя и открытую часть ключа SSH, как и на остальных узлах кластера, которые использовались на подготовительном этапе.

*Процедура*

1. Запустите процесс масштабирования узлов кластера с помощью команды:

```
BASH | □
nova-ctl scale --ssh-user <имя_пользователя> --ssh-key <закрытый ключ SSH>
```



В качестве аргументов `--ssh-key` и `--ssh-user` укажите информацию, использованную на этапе конфигурации ключевой пары SSH.

2. Далее для временного редактирования будет открыт файл конфигурации кластера в текстовом редакторе `vi`.

3. Добавьте новый узел в блок конфигурации `ClusterNodes` и сохраните изменения.

## Пример

```
BASH | □
nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem
spec:
...
clusterNodes:
...
worker: # Роль узла в кластере Kubernetes.
- hostGroup: "worker" # Существующий узел в кластере Kubernetes.
  networkSpec:
    hostname: "worker01.mycompany.local"
    ip: "10.251.11.120"
    gateway: "10.251.11.254"
    netmask: "255.255.255.0"
    dns:
      - "10.251.1.2"
    state: "present"
...
- hostGroup: "example-worker" # Добавляемый узел в кластер
Kubernetes.
  networkSpec:
    hostname: "worker02.mycompany.local"
    ip: "10.251.11.119"
    gateway: "10.251.11.254"
    netmask: "255.255.255.0"
    dns:
      - "10.251.1.2"
    state: "present"
```

4. При необходимости вы можете добавить в блок `hostGroup` описание новой группы узлов, в которой можно указать идентификатор нового шаблона виртуальной машины.

## Пример

```
BASH | □
nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem
spec:
...
infrastructureProvider:
...
zvirt:
...
hostGroup:
...
- name: "example-worker"
  templateId: "729df7ab-900b-4df7-ac40-01b70629dd4e"
  vnicProfileId: "9a58ffa8-98d8-472c-9198-5f54a9f8aaac"
  cpuCores: 1
  cpuSockets: 8
  cpuThreads: 1
```

```
memory: "8GiB"
maximumMemory: "16GiB"
```

5. На запрос Enter Vault root token: введите токен доступа к хранилищу секретов StarVault с привилегиями root , после чего начнется процесс добавления узла в кластер.
6. В процессе масштабирования будет запрошено подтверждение на создание виртуальных машин и сетевых интерфейсов в среде виртуализации zVirt.

*Пример*

```
BASH | □
nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem

■ Validating license... done ☕ 3/8: Preparing infrastructure... The following
actions will be preformed:
* create ovirt_vm worker01-nova-internal
* create ovirt_vm worker02-nova-internal
* create ovirt_nic worker01-nova-internal
* create ovirt_nic worker02-nova-internal
* create ovirt_vm_start worker01-nova-internal
* create ovirt_vm_start worker02-nova-internal Are you sure you want to
apply these changes? (yes/no) [no] yes
```

7. Дождитесь сообщения об успешном выполнении операции.

*Пример*

```
BASH | □
nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem

■ Validating license... done
■ Preparing infrastructure... done
■ Validating cluster nodes... done
■ Preparing cluster nodes... done
■ Scaling Kubernetes cluster... done

🚀 Cluster is successfully scaled.
```

8. Проверьте состояние узлов кластера Kubernetes после успешного масштабирования согласно руководству.

### 3. Вертикальное масштабирование узлов кластера

Вертикальное масштабирование заключается в добавлении физических или виртуальных ресурсов к текущим узлам кластера Kubernetes. В зависимости от метода установки платформы вертикальное масштабирование может выполняться как с помощью утилиты [nova-ctl](#), так и вручную.

## 3.1. Масштабирование узлов кластера, установленного универсальным методом [UPI]

Перед проведением работ по изменению ресурсов узла кластера Kubernetes необходимо вывести данный узел в режим обслуживания.

### Необходимые условия

- ✓ На вашем локальном компьютере установлена утилита `nova-ctl`.
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера ( `cluster-admin` ).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl` .

### Процедура

1. Выведите список всех узлов кластера и определите имя узла, которому необходимо изменить количество выделенных ресурсов:

```
kubectl get nodes
```

BASH | ↗

### Пример

```
kubectl get nodes
```

BASH | ↗

NAME	STATUS	ROLES	AGE	VERSION
master.mycompany.local	Ready	control-plane	5d17h	v1.26.8
worker.mycompany.local	Ready	ingress,worker	5d17h	v1.26.8
infra.mycompany.local	Ready	infra	5d17h	v1.26.8

2. Переведите выбранный узел в режим обслуживания согласно процедуре, описанной в разделе [Перевод узла в режим обслуживания](#).
3. После успешного перевода узла в режим обслуживания вы можете выключить узел для проведения работ по масштабированию ресурсов.
4. После завершения работ по масштабированию включите узел и дождитесь его загрузки.
5. Проверьте, что узел стал доступен для размещения рабочих нагрузок, следуя процедуре [возврата узла в эксплуатацию](#).

## 3.2. Масштабирование кластера, установленного в среде zVirt [IPI]

### Необходимые условия

- ✓ На вашем локальном компьютере установлена утилита [nova-ctl](#).
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера ( cluster-admin ).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите nova-ctl .

## Процедура

1. Запустите процесс масштабирования узлов кластера с помощью команды:

```
BASH | nova-ctl scale --ssh-user <имя_пользователя> --ssh-key <закрытый ключ SSH>
```



В качестве аргументов `--ssh-key` и `--ssh-user` укажите информацию, использованную [на этапе конфигурации ключевой пары SSH](#).

2. Далее для временного редактирования будет открыт файл конфигурации кластера в текстовом редакторе `vi`.
3. Измените количество выделенных ресурсов для групп узлов определенных в блоке `host-Group` и сохраните изменения.

## Пример

```
BASH | nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem
```

```
spec:  
...  
infrastructureProvider:  
...  
zvirt:  
...  
hostGroup:  
...  
- name: example-worker  
  templateId: 729df7ab-900b-4df7-ac40-01b70629dd4e  
  vnicProfileId: 9a58ffa8-98d8-472c-9198-5f54a9f8aaac  
  cpuCores: 1  
  cpuSockets: 12  
  cpuThreads: 1  
  memory: 12GiB  
  maximumMemory: 16GiB
```

 В среде виртуализации zVirt ресурсы для виртуальных машин будут добавлены с помощью горячего подключения (hot plug) при следующих условиях:

- Увеличение объема ОЗУ (memory) на значение не превышающее установленный лимит (maximumMemory).
- Увеличение количества виртуальных процессоров (cpuSockets) без изменениях их характеристик (cpuCores, cpuThreads).

Уменьшение количества ресурсов, изменение характеристик виртуальных процессоров (cpuCores, cpuThreads), а также изменение лимита ОЗУ (maximumMemory) требует последующего перезапуска виртуальных машин вручную. Для выполнения корректного перезапуска узлов кластера действуйте согласно процедуре вертикального масштабирования кластера, установленного универсальным методом (UPI).

4. В процессе масштабирования будет запрошено подтверждение на изменение виртуальных машин в среде виртуализации zVirt.

*Пример*

```
BASH | nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem  
■ Validating license... done ☕ 3/8: Preparing infrastructure... The following actions will be preformed:  
* update ovirt_vm worker03-nova-internal  
* update ovirt_vm worker04-nova-internal  
* update ovirt_vm worker05-nova-internal  
* update ovirt_vm ingress01-nova-internal  
* update ovirt_vm ingress02-nova-internal Are you sure you want to apply these changes? (yes/no) [no] yes
```

5. Дождитесь сообщения об успешном выполнении операции.

*Пример*

```
BASH | nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem  
■ Validating license... done  
■ Preparing infrastructure... done  
■ Validating cluster nodes... done  
■ Preparing cluster nodes... done  
■ Scaling Kubernetes cluster... done  
🚀 Cluster is successfully scaled.
```

6. Перейдите в веб-интерфейс среды виртуализации zVirt и убедитесь, что ресурсы виртуальных машин были изменены.

7. Запланируйте и выполните процедуру перезапуска узла в среде zVirt. Перезапуск узла рекомендуется выполнять вместе с переводом узла в режим обслуживания.

## 4. Удаление узлов кластера

Удаление узлов из кластера может быть выполнено в автоматическом режиме с помощью утилиты [nova-ctl](#).

### Необходимые условия

- ✓ На вашем локальном компьютере установлена утилита [nova-ctl](#).
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера ( `cluster-admin` ).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl`.
- ✓ У вас есть токен доступа к хранилищу секретов StarVault с привилегиями `root`.

### Процедура

1. Запустите процесс удаления узлов кластера с помощью команды:

```
BASH | nova-ctl scale --ssh-user <имя_пользователя> --ssh-key <закрытый ключ SSH>
```



В качестве аргументов `--ssh-key` и `--ssh-user` укажите информацию, использованную [на этапе конфигурации ключевой пары SSH](#).

2. Далее для временного редактирования будет открыт файл конфигурации кластера в текстовом редакторе `vi`.
3. Отметьте узлы выбранные для удаления, изменив значение ключа `state` с `present` на `absent` и сохраните изменения.

### Пример

```
BASH | nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem  
spec:  
...  
    clusterNodes:  
    ...  
        worker: # Роль узла в кластере Kubernetes.  
        - hostGroup: "worker" # Существующий узел в кластере Kubernetes.  
        networkSpec:  
            hostname: "worker01.mycompany.local"  
            ip: "10.251.11.119"  
            gateway: "10.251.11.254"  
            netmask: "255.255.255.0"  
            dns:  
            - "10.251.1.2"  
            state: "present"
```

```
...
- hostGroup: "worker" # Удаляемый узел из кластера Kubernetes.
  networkSpec:
    hostname: "worker02.mycompany.local"
    ip: "10.251.11.119"
    gateway: "10.251.11.254"
    netmask: "255.255.255.0"
    dns:
      - "10.251.1.2"
    state: "absent"
```

4. В процессе будет запрошено подтверждение на удаление виртуальной машины.

*Пример*

```
nova-ctl scale --ssh-user nova-installer --ssh-key id_rsa.pem
```

BASH | □

```
☕ 8/8: Finishing installation... The following actions will be preformed:
* delete ovirt_nic worker02-nova-internal
* delete ovirt_vm worker02-nova-internal
* delete ovirt_vm_start worker02-nova-internal Are you sure you want to
apply these changes? (yes/no) [no] yes
```



Данный этап выполняется только для кластеров Kubernetes, развернутых в инфраструктуре, подготавливаемой узлом `nova-ctl` для управления платформой (IPI). Виртуальная машина будет полностью удалена из платформы виртуализации.

5. Дождитесь сообщения об успешном выполнении операции.

*Пример*

```
■ Validating license... done
■ Preparing infrastructure... done
■ Validating cluster nodes... done
■ Preparing cluster nodes... done
■ Scaling Kubernetes cluster... done

🚀 Cluster is successfully scaled.
```

BASH | □

6. Проверьте состояние узлов кластера Kubernetes после успешного масштабирования согласно руководству.

## 5. Повторное добавление узлов в кластер

В случае необходимости повторного добавления узла в кластер Kubernetes, например, в ситуациях, когда на узле требуется полная переустановка компонентов платформы, вам необходимо сначала удалить узел из кластера, следя процедуре удаления узлов, а затем

выполнить повторное добавление узла, используя процедуру горизонтального масштабирования кластера. На данном узле будет выполнен полный сброс и переустановка всех компонентов Nova Container Platform.

## 6. Перевод узла в режим обслуживания

В процессе эксплуатации платформы Nova Container Platform может возникнуть необходимость перевода узла в режим обслуживания. Находясь в режиме обслуживания, узел исключается из очереди Kubernetes Scheduler и более не может размещать какие-либо вновь запланированные рабочие нагрузки. Любые существующие рабочие нагрузки будут эвакуированы и запущены на других подходящих узлах кластера Kubernetes. Исключение составляют нагрузки, запущенные контроллером DaemonSet, перезапуск которых на других узлах невозможен.

Перевод узла кластера Kubernetes в режим обслуживания может быть выполнен с помощью утилиты [kubectl](#) согласно процедуре, описанной далее.



Объекты Pod, которые не управляются контроллерами ReplicaSet, DaemonSet, StatefulSet и Job, не могут быть эвакуированы с узла и будут удалены при переводе узла в режим обслуживания. Убедитесь, что в кластере Kubernetes есть достаточное количество узлов, отвечающих тем же критериям, что и переводимый в режим обслуживания узел. В противном случае эвакуированные объекты Pod будут находиться в состоянии `Pending` до тех пор, пока узел не будет возвращен в работу.

### 6.1. Перевод узла кластера в режим обслуживания

Для перевода узла в режим эксплуатации следуйте процедуре ниже.

#### Необходимые условия

- ✓ На вашем локальном компьютере установлена утилита [nova-ctl](#).
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера (`cluster-admin`).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl`.

#### Процедура

1. Получите список всех узлов кластера и определите имя узла, который необходимо вывести в режим обслуживания:

```
kubectl get nodes
```

BASH | ↗

*Пример*

```
kubectl get nodes
```

BASH | ↗

```
NAME STATUS ROLES AGE VERSION
master.mycompany.local Ready control-plane
5d17h v1.26.8
worker.mycompany.local Ready ingress,worker 5d17h v1.26.8
infra.mycompany.local Ready infra 5d17h v1.26.8
```

2. Переведите выбранный узел в состояние `SchedulingDisabled`, выполнив команду:

```
kubectl cordon <имя узла>
```

BASH | ↗

*Пример*

```
$ kubectl cordon worker.mycompany.local
```

BASH | ↗

```
node/worker.mycompany.local cordoned
```

3. Выполните эвакуацию всех рабочих нагрузок с выбранного узла, выполнив команду:

```
kubectl drain <имя узла> --ignore-daemonsets
```

BASH | ↗

*Пример*

```
$ kubectl drain worker.mycompany.local --ignore-daemonsets
```

BASH | ↗

```
node/worker.mycompany.local already cordoned
Warning: ignoring DaemonSet-managed Pods: kube-system/kube-proxy-rdmzh,
nova-csi-drivers/nova-oauth-csi-provider-6hnbs, nova-csi-drivers/nova-
secrets-store-csi-driver-qcmnh, nova-ingress-public/nova-ingress-public-
controller-g8g7h, nova-monitoring/nova-cadvisor-f9pfm, nova-monitoring/nova-
prometheus-node-exporter-ppfn6, nova-neuvector/neuvector-enforcer-pod-nxwch,
nova-secrets-webhook/nova-oauth-secrets-webhook-slssp
evicting pod kube-system/coredns-5b7664d478-w269k
evicting pod kube-system/coredns-5b7664d478-7vbbb
pod/coredns-5b7664d478-w269k evicted
pod/coredns-5b7664d478-7vbbb evicted
node/worker.mycompany.local drained
```



Для удаления с узла объектов Pod, запущенных без использования контроллеров ReplicaSet, DaemonSet, StatefulSet и Job, используйте `kubectl drain` с опцией `--force`.

4. После успешного выполнения команды `kubectl drain` вы можете выключить узел для проведения запланированных работ.

## 6.2. Возврат узла кластера в режим эксплуатации

Для возврата узла в режим эксплуатации следуйте процедуре ниже.

### Необходимые условия

- ✓ На вашем локальном компьютере установлена утилита [nova-ctl](#).
- ✓ У вас есть доступ к Kubernetes API с привилегиями администратора кластера ( `cluster-admin` ).
- ✓ У вас есть закрытый ключ SSH на вашем локальном компьютере, который нужно предоставить утилите `nova-ctl`.

### Процедура

1. После завершения работ включите ранее выключенный узел и дождитесь его загрузки.
2. Проверьте, что узел стал доступен для размещения рабочих нагрузок (статус узла `Ready`) выполнив команду:

```
kubectl get nodes
```

BASH | ↗

### Пример

```
kubectl get nodes
  NAME           STATUS    ROLES      AGE   VERSION
  master.mycompany.local   Ready    control-plane   5d18h   v1.26.8
  worker.mycompany.local   Ready,SchedulingDisabled   ingress,worker   5d18h   v1.26.8
  infra.mycompany.local   Ready    infra       5d18h   v1.26.8
```

BASH | ↗

3. Разрешите запуск рабочих нагрузок на выбранном узле, выполнив команду:

```
kubectl uncordon <имя узла>
```

BASH | ↗

### Пример

```
$ kubectl uncordon worker.mycompany.local
node/worker.mycompany.local uncordoned
```

BASH | ↗

4. Узел готов к эксплуатации в кластере Kubernetes.

## 7. Выключение платформы Nova Container Platform

В данном разделе описана процедура корректного выключения платформы Nova Container Platform.

### 7.1. Подготовительные действия для выключение Nova Container Platform

- Выполните [полное резервное копирование мастер-узлов](#) перед выключением кластера Kubernetes.



Убедитесь, что резервная копия надежна сохранена на внешнем хранилище. В случае каких-либо проблем с Nova Container Platform данная резервная копия поможет вам восстановить кластер в прежнее состояние.

- Если вы планируете отключение кластера Kubernetes на довольно продолжительное время, то проверьте срок действия TLS-сертификатов.
  - Для проверки срока действия сертификатов используйте раздел документации [Проверка срока действия сертификатов платформы](#).
  - При необходимости обновления сертификатов воспользуйтесь разделом документации [Обновление сертификатов платформы](#).
- (Опционально) Выполните резервное копирование данных пользовательских приложений предусмотренным вами методом.

### 7.2. Выключение узлов кластера



Вы можете выключить узлы кластера Kubernetes на срок не более 1 года от даты первоначальной установки платформы или перевыпуска сертификатов в кластере. При соблюдении данных сроков кластер Kubernetes запустится корректно и продолжит функционировать.

#### Предварительные условия:

- ✓ Вы выполнили [полное резервное копирование мастер-узлов](#)
- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.

#### Процедура

- Установите на каждом узле кластера Kubernetes запрет на запуск новых нагрузок:

```
BASH | □  
for node in $(kubectl get nodes -o jsonpath='{.items[*].metadata.name}'); do  
echo ${node} ; kubectl cordon ${node} ; done
```

*Пример*

```
BASH | □  
for node in $(kubectl get nodes -o jsonpath='{.items[*].metadata.name}'); do  
echo ${node} ; kubectl cordon ${node} ; done  
  
nova-infra-1.mycompany.local  
node/nova-infra-1.mycompany.local cordoned  
nova-master-1.mycompany.local  
node/nova-master-1.mycompany.local cordoned  
nova-worker-1.mycompany.local  
node/nova-worker-1.mycompany.local cordoned
```

2. Остановите существующие нагрузки и освободите от них рабочие узлы кластера

Kubernetes:

```
BASH | □  
for node in $(kubectl get nodes -l node-role.kubernetes.io/worker -o  
jsonpath='{.items[*].metadata.name}'); do echo ${node} ; kubectl drain  
${node} --delete-emptydir-data --ignore-daemonsets=true --timeout=15s ; done
```

*Пример*

```
BASH | □  
for node in $(kubectl get nodes -l node-role.kubernetes.io/worker -o  
jsonpath='{.items[*].metadata.name}'); do echo ${node} ; kubectl drain  
${node} --delete-emptydir-data --ignore-daemonsets=true --timeout=15s ; done  
  
nova-worker-1.mycompany.local  
node/nova-worker-1.mycompany.local already cordoned  
Warning: ignoring DaemonSet-managed Pods: kube-system/kube-proxy-4c5t6,  
nova-csi-drivers/nova-oauth-csi-provider-s4zk5, nova-csi-drivers/nova-  
secrets-store-csi-driver-t65px, nova-ingress-public/nova-ingress-public-  
controller-lzjhk, nova-monitoring/nova-cadvisor-s2pc4, nova-monitoring/nova-  
prometheus-node-exporter-nqhmq, nova-secrets-webhook/nova-oauth-secrets-  
webhook-dzvilk  
node/nova-worker-1.mycompany.local drained
```

3. Остановите существующие нагрузки и освободите от них инфраструктурные узлы кластера Kubernetes:

```
BASH | □  
for node in $(kubectl get nodes -l node-role.kubernetes.io/infra -o  
jsonpath='{.items[*].metadata.name}'); do echo ${node} ; kubectl drain  
${node} --delete-emptydir-data --ignore-daemonsets=true --timeout=15s ; done
```

*Пример*

BASH | □

```

for node in $(kubectl get nodes -l node-role.kubernetes.io/infra -o jsonpath='{.items[*].metadata.name}'); do echo ${node}; kubectl drain ${node} --delete-emptydir-data --ignore-daemonsets=true --timeout=15s; done

nova-infra-1.mycompany.local
node/nova-infra-1.mycompany.local already cordoned
Warning: ignoring DaemonSet-managed Pods: kube-system/kube-proxy-qkdc8,
nova-csi-drivers/nova-oauth-csi-provider-4rp6k, nova-csi-drivers/nova-
secrets-store-csi-driver-2l7mj, nova-ingress-internal/nova-ingress-internal-
controller-97mkb, nova-monitoring/nova-cadvisor-pdncn, nova-monitoring/nova-
prometheus-node-exporter-4626w, nova-secrets-webhook/nova-oauth-secrets-
webhook-sc8ms
evicting pod nova-monitoring/prometheus-main-0
evicting pod kube-system/nova-dns-6fb4489cd7-qn8mw
evicting pod nova-gitops/image-automation-controller-5ffdb68d9d-t4flv
evicting pod kube-system/coredns-6ccbfbd9d-6fk7l
...

```

#### 4. Выключите узлы кластер Kubernetes:

```

for node in $(kubectl get nodes -o jsonpath='{.items[*].metadata.name}'); do
  kubectl debug node/${node} --image=hub.nova-
  platform.io/registry/nova/support-tools:v1.0.0 -n kube-system -- chroot
  /host shutdown -h 1; done

```

#### Пример

```

for node in $(kubectl get nodes -o jsonpath='{.items[*].metadata.name}'); do
  kubectl debug node/${node} --image=hub.nova-
  platform.io/registry/nova/support-tools:v1.0.0 -n kube-system -- chroot
  /host shutdown -h 1; done

Creating debugging pod node-debugger-nova-infra-1.mycompany.local-n6vjk with
container debugger on node nova-infra-1.mycompany.local.
Creating debugging pod node-debugger-nova-master-1.mycompany.local-78gfx
with container debugger on node nova-master-1.mycompany.local.
Creating debugging pod node-debugger-nova-worker-1.mycompany.local-xd6wc
with container debugger on node nova-worker-1.mycompany.local.

```



Если кластер Kubernetes установлен в закрытом сетевом окружении с использованием Nova Universe, то необходимо использовать образ `nova/support-tools:v1.0.0` из хранилища образов Nova Universe.

#### 5. Выключите при необходимости связанные с Nova Container Platform внешние ресурсы (внешние хранилища, службы каталогов, хранилища образов и т.п.).

Для включения кластера Nova Container Platform следуйте процедуре перезапуска узлов платформы.

## 8. Перезапуск платформы Nova Container Platform

В данном разделе описана процедура корректного перезапуска (включения) платформы Nova Container Platform.

### 8.1. Предварительные условия для перезапуска Nova Container Platform

- ✓ Вы корректно выключили кластер Nova Container Platform согласно процедуре выключения.
- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Nova Container Platform.
- ✓ Вы установили утилиту `kubectl` для работы с Nova Container Platform.

#### Процедура

1. Включите связанные с Nova Container Platform внешние ресурсы (внешние хранилища, службы каталогов, хранилища образов и т.п.).
2. Включите узлы кластера Nova Container Platform и дождитесь загрузки ОС. Подождите не менее 5 минут перед тем как проверять статус мастер-узлов платформы.
3. Проверьте, узлы запущены и готовы к работе:

```
kubectl get nodes
```

BASH | ↗

Узлы запущены и готовы к работе, если имеют статус Ready, как показано ниже:

NAME	AGE	VERSION	STATUS	ROLES
nova-master-1.mycompany.local	22h	v1.27.11	Ready, SchedulingDisabled	control-plane
nova-infra-1.mycompany.local	22h	v1.27.11	Ready, SchedulingDisabled	infra
nova-worker-1.mycompany.local	22h	v1.27.11	Ready, SchedulingDisabled	ingress,worker

BASH | ↗

4. Разрешите планирование (запуск) рабочих нагрузок на мастер-узлах:

```
kubectl uncordon -l node-role.kubernetes.io/control-plane
```

BASH | ↗

### *Пример*

```
kubectl uncordon -l node-role.kubernetes.io/control-plane  
node/nova-master-1.mycompany.local uncordoned
```

BASH | □

5. Дождитесь запуска рабочих нагрузок на мастер-узлах.

6. Разрешите запуск (планирование) рабочих нагрузок на инфраструктурных узлах:

```
kubectl uncordon -l node-role.kubernetes.io/infra
```

BASH | □

### *Пример*

```
kubectl uncordon -l node-role.kubernetes.io/infra  
node/nova-infra-1.mycompany.local uncordoned
```

BASH | □

7. Дождитесь запуска рабочих нагрузок на инфраструктурных узлах.

8. Разрешите запуск рабочих нагрузок на рабочих узлах:

```
kubectl uncordon -l node-role.kubernetes.io/worker
```

BASH | □

### *Пример*

```
kubectl uncordon -l node-role.kubernetes.io/worker  
node/nova-worker-1.mycompany.local uncordoned
```

BASH | □

9. Проверьте, что в кластере Kubernetes все необходимые сервисы запустились корректно:

```
kubectl get pods -A -o wide | grep -v "Run\|Comp"
```

BASH | □

Пустой вывод данной команды означает, что в кластере Kubernetes отсутствуют какие-либо сервисы, находящиеся в статусах, отличных от *Running* или *Completed*.

## 9. Распределение рабочей нагрузки в процессе работы

Базовая функциональность Kubernetes не предусматривает перераспределение рабочей нагрузки в процессе работы. В Nova Container Platform для поддержки оптимального распределения и использования ресурсов задействуется компонент descheduler.

Descheduler на основе заданных критериев периодически оценивает нагрузку и вытесняет поды.

Descheduler запускается в Nova Container Platform с политикой, заданной в **ConfigMap** `nova-descheduler` в namespace `nova-descheduler`.

Descheduler каждые 15 минут вытесняет поды, которые удовлетворяют включенной в **ConfigMap** политике.



Время, после которого будут вытеснены поды зависит от настройки `minPodLifetimeSeconds`.

## 9.1. Настройки политики Descheduler

### 9.1.1. DefaultEvictor

`DefaultEvictor` обеспечивает вытеснение подов таким образом, чтобы результирующий под поместился хотя бы на одном доступном узле, если указана опция `nodeFit: true`.

```
- name: "DefaultEvictor"
  args:
    nodeFit: true
```

[YAML](#) | [JSON](#)

### 9.1.2. RemoveDuplicates

Политика `RemoveDuplicates` следит за тем, чтобы на узле был запущен только один под с одним контроллером. Если подов с контроллером два на одном узле, descheduler удаляет дублирующий под.

```
- name: ProfileName
  pluginConfig:
    - name: "RemoveDuplicates"
      plugins:
        balance:
          enabled:
            - "RemoveDuplicates"
```

[YAML](#) | [JSON](#)

### 9.1.3. RemoveFailedPods

Политика `RemoveFailedPods` вытесняет поды, которые завершились с ошибкой и в течении установленного времени `minPodLifetimeSeconds` не восстановились, за исключением тех, которые принадлежат типам, перечисленным в `excludeOwnerKinds`.

```
- name: "RemoveFailedPods"
  args:
    excludeOwnerKinds:
```

[YAML](#) | [JSON](#)

```
- "Job"  
minPodLifetimeSeconds: 900
```

## 9.1.4. PodLifeTime

Политика PodLifeTime гарантирует, что поды в состояниях `Pending` и/или `PodInitializing` старше 30 минут будут вытеснены с узлов.

```
- name: "PodLifeTime"  
args:  
  maxPodLifeTimeSeconds: 1800  
  states:  
    - "Pending"  
    - "PodInitializing"
```

[YAML](#) | [JSON](#)

## 9.1.5. RemovePodsHavingTooManyRestarts

Политика RemovePodsHavingTooManyRestarts гарантирует, что поды, имеющие больше 50 перезапусков контейнеров (включая init-контейнеры), будут удалены с узлов.

```
- name: "RemovePodsHavingTooManyRestarts"  
args:  
  podRestartThreshold: 50  
  includingInitContainers: true
```

[YAML](#) | [JSON](#)

## 9.1.6. RemovePodsViolatingNodeTaints

Политика RemovePodsViolatingNodeTaints гарантирует, что поды, нарушающие определённые условия (`taint`) на узлах, будут удалены. Например, под имеющий `toleration` и запущенный на узле с соответствующим `taint` будет вытеснен с узла, если `taint` на узле будет изменен или удален.

Настроенные `taint` можно посмотреть в настройке политики.

```
- name: "RemovePodsViolatingNodeTaints"  
args:  
  excludedTaints:  
    - node.kubernetes.io/memory-pressure  
    - node.kubernetes.io/disk-pressure  
    - node.kubernetes.io/pid-pressure  
    - node.kubernetes.io/unschedulable  
    - node.kubernetes.io/network-unavailable
```

[YAML](#) | [JSON](#)

## 9.1.7. RemovePodsViolatingNodeAffinity

Политика RemovePodsViolatingNodeAffinity обеспечивает соблюдение подами `affinity rules`. Если поды не соответствуют `affinity rules`, то поды будут вытеснены.

с узла.

Настроенные affinity rules :

- requiredDuringSchedulingIgnoredDuringExecution
- preferredDuringSchedulingIgnoredDuringExecution

[YAML](#) | [JSON](#)

```
- name: "RemovePodsViolatingNodeAffinity"
  args:
    nodeAffinityType:
      - "requiredDuringSchedulingIgnoredDuringExecution"
      - "preferredDuringSchedulingIgnoredDuringExecution"
```

## 9.1.8. RemovePodsViolatingInterPodAntiAffinity

Политика RemovePodsViolatingInterPodAntiAffinity следит за тем, чтобы все поды нарушающие правила anti-affinity были удалены.

[YAML](#) | [JSON](#)

```
- name: "RemovePodsViolatingInterPodAntiAffinity"
```

## 9.1.9. LowNodeUtilization

Политика LowNodeUtilization находит нагруженные узлы и перераспределяет их поды на ненагруженные узлы. Узлы в кластере отслеживаются по CPU/памяти/подам (в процентах).

После превышения порога указанного в политике — перераспределяет поды.



Настройка учитывает не реальное потребление ресурсов на узле, а реквесты у подов.

[YAML](#) | [JSON](#)

```
- name: ProfileName
  pluginConfig:
    - name: "LowNodeUtilization"
      args:
        thresholds:
          "cpu" : 40
          "memory": 40
          "pods": 20
        targetThresholds:
          "cpu" : 80
          "memory": 80
          "pods": 60
      plugins:
        balance:
          enabled:
            - "LowNodeUtilization"
```

# 9.2. Пример настроенной политики Descheduler

## ► policy.yaml

При обновлении платформы старая политика сохраняется в ConfigMap `nova-descheduler-deprecated`.