Menu

Конфигурация машины

Обзор

Содержание

Как работает Machine Configuration

Machine Configuration управляет обновлениями файлов, управлением systemd-юнитами и развертыванием публичных SSH-ключей на узлах кластера. Система предоставляет Custom Resource Definition (CRD) MachineConfig для записи конфигурационных файлов на хостах и CRD MachineConfigPool для организации узлов в группы конфигураций.

Каждый MachineConfigPool управляет набором узлов и связанными с ними MachineConfigs. Роли узлов определяют членство в MachineConfigPool — пулы управляют узлами на основе их меток ролей.

Во время установки кластера система автоматически создаёт два MachineConfigPools (master и worker) вместе с двумя пустыми MachineConfigs (00-master и 00-worker). Пул master управляет конфигурацией 00-master, а пул worker — конфигурацией 00-worker.

Вы можете создавать пользовательские MachineConfigPools для рабочих узлов, которым требуются специализированные конфигурации. Мастер-узлы не могут использовать пользовательские пулы.

Пользовательские MachineConfigPools наследуют все конфигурации от пула worker и добавляют свои собственные настройки. Любые изменения в пуле worker автоматически распространяются на пользовательские пулы. Machine Configuration не поддерживает пользовательские пулы, которые не наследуют конфигурации от пула worker.

В кластере есть дефолтный MachineConfiguration CR с именем "cluster" для настройки глобальной политики обновления узлов. Подробнее см. документацию по Node Disruption Policy.

Иногда конфигурация узлов отклоняется от заданного состояния. machine-config-daemon постоянно отслеживает отклонения конфигурации и помечает затронутые узлы как Degraded, пока администратор не решит проблему. Узлы в состоянии Degraded остаются работоспособными, но не могут получать обновления.

Ключевые понятия

Обработка конфигураций
MachineConfigs обрабатываются в алфавитном порядке. Первая конфигурация служит базовой, а последующие накладываются поверх неё. Каждый MachineConfigPool рендерит управляемые им конфигурации в один MachineConfig с именем: `render-<pool-name>-<content-hash>`, который применяется ко всем узлам этого пула.

Стратегия обновления
Machine Configuration обновляет узлы по возрасту, начиная с самых старых. Поле `maxUnavailable` в каждом MachineConfigPool контролирует, сколько узлов обновляется одновременно.

Область управления
Machine Configuration управляет только явно настроенными элементами. Ручные изменения системы остаются без вмешательства Machine Configuration Operator.

Формат конфигурации
Все MachineConfigs используют спецификацию Ignition v3.4.0.

Обнаружение отклонений
Если файлы, управляемые Machine Configuration, изменяются вне системы, machine-config-daemon помечает узел как Degraded, но не перезаписывает изменённые файлы.

Преимущества пулов
MachineConfigPools гарантируют, что новые узлы автоматически получают правильную конфигурацию при присоединении к кластеру.

Поддерживаемые изменения

- Обычные файлы (в записываемых, не корневых каталогах)
- systemd-юниты и их конфигурации
- Публичные SSH-ключи только для пользователя boot

Machine Configuration не создаёт пользователей или группы. Пользователь boot и группа должны быть созданы заранее для настройки SSH-ключей.

Важно: избегайте ручных изменений узлов — они могут вызвать конфликты конфигураций.

Типы конфигураций

Файлы
Создание или изменение содержимого и прав доступа файлов. Файлы можно управлять только если их раздел доступен для записи.

systemd-юниты
Определение новых systemd-сервисов или расширение существующих дополнительной конфигурацией.

Публичные SSH-ключи
Настройка SSH-доступа для пользователя boot. Ключи для других пользователей считаются недействительными.

Процесс обновления узла

При применении MachineConfig Machine Configuration гарантирует, что все затронутые узлы достигнут желаемого состояния. Machine Configuration Operator генерирует новую

рендеренную конфигурацию, а machine-config-daemon выполняет на каждом узле следующие шаги:

1. **Cordon** — пометить узел как недоступный для новых нагрузок
2. **Drain** — завершить текущие нагрузки и перенести их на другие узлы
3. **Apply** — записать новую конфигурацию на диск
4. **Reboot** — перезагрузить узел для активации изменений
5. **Uncordon** — снова сделать узел доступным для нагрузок

Проверка статуса MachineConfigPool

Проверьте статус пула командой:

```
kubectl get machineconfigpool
```

Пример вывода:

```
NAME     CONFIG                   UPDATED  UPDATING  DEGRADED  MACHINECOUNT
READYMACHINECOUNT  UPDATEDMACHINECOUNT  DEGRADEDMACHINECOUNT  AGE
master   rendered-master-06c9c4    True     False     False     3             3
3                  0                    4h42m
worker   rendered-worker-f4b64     False    True      False     3             2
2                  0                    4h42m
```

Описание полей:

- **NAME**: идентификатор пула
- **CONFIG**: последняя применённая конфигурация для всех узлов пула
- **UPDATED**: `True`, если все узлы имеют текущую конфигурацию; `False` во время обновления
- **UPDATING**: `True`, если хотя бы один узел обновляется; `False`, если все актуальны

- **DEGRADED**: `True`, если конфигурация не может быть применена хотя бы к одному узлу
- **MACHINECOUNT**: общее число узлов в пуле
- **READYMACHINECOUNT**: узлы с текущей конфигурацией в здоровом и доступном состоянии
- **UPDATEDMACHINECOUNT**: узлы, применившие текущую конфигурацию
- **DEGRADEDMACHINECOUNT**: узлы, помеченные как degraded или неустранимые

В этом примере все три мастер-узла актуальны, а пул worker обновляется — два узла завершили обновление, один в процессе.

Получите подробную информацию о пуле:

```
kubectl describe machineconfigpool worker
```

Просмотрите все MachineConfigs:

```
kubectl get machineconfig
```

Пример вывода:

```
NAME                    IGNITIONVERSION  AGE
00-master               3.4.0            3h2m
00-worker               3.4.0            3h2m
rendered-master-ccb     3.4.0            1h12m
rendered-worker-bad     3.4.0            1h20m
```

Изучите конкретную конфигурацию:

```
kubectl describe machineconfig 00-master
```

Проверьте статус отдельного узла:

```
kubectl get node -o custom-
columns=NODE:.metadata.name,DESIRED:.metadata.annotations."machineconfiguration\.alauda\.io/c
```

Пример вывода:

```
NODE             DESIRED                                    CURRENT
STATE
192.168.132.216   rendered-master-98db9ca4f4b4cd             rendered-master-
98db9ca4f4b4cd             Degraded
192.168.135.83    rendered-worker-05f27341ba49cf86dc4b      rendered-master-
e08d9cab50e383             Working
192.168.134.99    rendered-worker-05f27341ba49cf86dc4b      rendered-worker-
05f27341ba49cf86dc4b      Done
```

Описание состояний узла:

- **NODE**: идентификатор узла
- **DESIRED**: целевая конфигурация узла
- **CURRENT**: текущая применённая конфигурация
- **STATE**: статус конфигурации
  - `Done` : узел здоров, желаемая и текущая конфигурации совпадают
  - `Working` : узел обновляется (текущая ≠ желаемой)
  - `Degraded` : обнаружено отклонение конфигурации или сбой применения — проверьте логи для выяснения причины

Управление конфигурацией узлов с помощью MachineConfig

Вы можете использовать задачи, описанные в этом разделе, для создания объектов `MachineConfig`, которые изменяют файлы, systemd-юниты и публичные SSH-ключи на узлах, а также для восстановления узлов, у которых произошёл дрейф конфигурации.

`MachineConfig` поддерживает спецификацию Ignition версии 3.4. Все объекты `MachineConfig` должны создаваться в соответствии с этой версией.

В некоторых случаях конфигурация на узле может не полностью соответствовать конфигурации, применённой через `MachineConfig`. Такое состояние называется дрейфом конфигурации. Демон конфигурации машины периодически проверяет, произошёл ли дрейф конфигурации на узле. Если дрейф обнаружен, узел помечается как `Degraded` и остаётся в этом состоянии до тех пор, пока администратор не восстановит ожидаемую конфигурацию.

В следующих примерах показано, как использовать объекты `MachineConfig` для управления конфигурацией узлов.

Содержание

Настройка службы времени Chrony

Чтобы настроить службу синхронизации времени Chrony (`chronyd`) и указать NTP-серверы и связанные параметры, вы можете обновить файл `chrony.conf` на целевых узлах с помощью объекта `MachineConfig` .

1. Сначала создайте временный файл с нужной конфигурацией Chrony:

```
chrony.conf
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
makestep 1.0 3
rtcsync
logdir /var/log/chrony
```

2. Затем закодируйте содержимое файла в base64:

```
base64 -w0 chrony.conf
```

3. Создайте объект `MachineConfig` с именем `99-worker-chrony` . В поле `.spec.config.storage.files[0].contents.source` вставьте строку в формате `data:text/plain;base64,<encoded-content>` с base64-кодированным содержимым:

```
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfig
metadata:
  name: 99-worker-chrony
  labels:
    machineconfiguration.alauda.io/role: worker
spec:
  config:
    ignition:
      version: 3.4.0
    storage:
      files:
        - path: /etc/chrony.conf
          mode: 0644
          contents:
            source:
'data:text/plain;base64,c2VydmVyIDAuY2VudG9zLnBvb2wubnRwLm9yZyBpYnVyc3QKc2VydmVyIDEuY2VudG
```

Эта конфигурация создаёт объект `MachineConfig`, который применяет кастомный файл `chrony.conf` к узлам, связанным с пулом конфигурации машин `worker`. Файл будет записан в `/etc/chrony.conf` на каждом узле с правами доступа `0644`.

Отключение службы времени Chrony

Чтобы отключить службу синхронизации времени Chrony на узлах с определённой ролью, можно создать объект `MachineConfig`, который переопределит определение systemd-юнита и отключит службу.

Пример конфигурации:

```yaml
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfig
metadata:
  name: 99-worker-disable-chrony
  labels:
    machineconfiguration.alauda.io/role: worker
spec:
  config:
    ignition:
      version: 3.4.0
    systemd:
      units:
        - name: chronyd.service
          enabled: false
          contents: |
            [Unit]
            Description=NTP client/server
            Documentation=man:chronyd(8) man:chrony.conf(5)
            After=ntpdate.service sntp.service ntpd.service
            Conflicts=ntpd.service systemd-timesyncd.service
            ConditionCapability=CAP_SYS_TIME

            [Service]
            Type=forking
            PIDFile=/run/chrony/chronyd.pid
            EnvironmentFile=-/etc/sysconfig/chronyd
            ExecStart=/usr/sbin/chronyd $OPTIONS
            ExecStartPost=/usr/libexec/chrony-helper update-daemon
            PrivateTmp=yes
            ProtectHome=yes
            ProtectSystem=full

            [Install]
            WantedBy=multi-user.target
```

Эта конфигурация отправляет кастомную версию файла юнита `chronyd.service` на узлы в пуле конфигурации машин `worker`. Служба явно отключена. После применения конфигурации и перезагрузки узлов служба Chrony больше не будет запускаться автоматически.

Настройка публичного SSH-ключа для пользователя `boot`

Система конфигурации машин позволяет настроить публичный SSH-ключ для пользователя `boot` на управляемых узлах. Конфигурация для других учётных записей пользователей не поддерживается. Обратите внимание, что конфигурация машин не создаёт пользователей или группы автоматически — вы должны убедиться, что пользователь и группа `boot` существуют на узле до применения конфигурации.

Пример конфигурации:

```
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfig
metadata:
  name: 99-worker-ssh
  labels:
    machineconfiguration.alauda.io/role: worker
spec:
  config:
    ignition:
      version: 3.4.0
    passwd:
      users:
        - user: boot
          sshAuthorizedKeys:
            - ssh-rsa <ssh-public-key>
```

Этот `MachineConfig` установит указанный SSH-ключ в файл `/home/boot/.ssh/authorized_keys` на узлах в пуле конфигурации машин `worker`.

Восстановление после дрейфа конфигурации

Если конфигурация узла расходится с назначенным `MachineConfig`, он будет помечен как `Degraded`. В этом состоянии узел продолжает работать, но не может получать

дальнейшие обновления конфигурации до устранения проблемы.

Существует два способа восстановить узел из этого состояния:

1. **Ручное восстановление конфигурации**
   Вы можете вручную привести файлы и права доступа на узле в точное соответствие с теми, что указаны в назначенном `MachineConfig`. Система обнаружит исправление и снимет статус degraded.

2. **Принудительное повторное применение конфигурации**
   Создайте пустой файл `/run/machine-config-daemon-force` на затронутом узле. Демон конфигурации машины обнаружит этот триггер, повторно применит текущий `MachineConfig`, удалит файл-триггер и перезагрузит узел. После перезагрузки узел перейдёт из состояния `Degraded` обратно в `Done`.

Политики прерывания узлов

Содержание

Понимание прерывания узлов в конфигурации машины

По умолчанию, когда вы вносите определённые изменения в раздел `systemd` units объекта `MachineConfig`, оператор Machine Configuration выполнит дренаж и перезагрузку узлов, связанных с этим `MachineConfig`. Однако изменения в обычных файлах, как правило, **не** вызывают перезагрузку, что может привести к тому, что конфигурация не вступит в силу, как ожидалось. Чтобы решить эту проблему, вы можете определить политики прерывания узлов, чтобы указать, какие типы изменений должны вызывать перезагрузку узлов или другие действия прерывания. Эти политики настраиваются в объекте `MachineConfiguration`, расположенном в пространстве имён `cpaas-system`. См. пример ниже для настройки политики прерывания узлов.

После определения оператор Machine Configuration проверяет политику на наличие ошибок, таких как неверный формат. Затем он записывает политику в поле `status.nodeDisruptionPolicyStatus` объекта `MachineConfiguration`. Эти пользовательские политики переопределяют настройки прерывания по умолчанию в кластере.

По умолчанию с кластером устанавливается пользовательский ресурс `MachineConfiguration` с именем `cluster`. Вы можете настроить поведение прерывания узлов на этом ресурсе.

Что можно контролировать с помощью политик прерывания узлов

Политики прерывания узлов позволяют определить, что происходит при изменениях в следующих областях конфигурации:

- **Файлы**: Вы можете определить поведение при изменениях файлов (за исключением изменений в корневом каталоге). По умолчанию изменения файлов не вызывают прерывания. Вы можете изменить это поведение с помощью поля `spec.defaultNodeDisruptionPolicySpecAction.files`.
- **Systemd Units**: Вы можете создавать или изменять службы `systemd`, включая их включение, отключение или изменение состояния. По умолчанию изменения в `systemd` units вызывают дренаж узла и перезагрузку.
- **SSH Public Keys**: Вы можете добавлять или обновлять SSH-ключи для пользователя `boot`. Эти изменения применяются немедленно по умолчанию и не вызывают перезагрузку или дренаж.

Каждое изменение оценивается в соответствии с политикой прерывания узлов, которая может вызвать одно или несколько из следующих действий:

- `Reboot`: Дренаж и перезагрузка узла.
- `None`: Прерывание не вызывается; изменение применяется без уведомления.
- `Drain`: Дренаж узла без перезагрузки.
- `Restart`: Перезапуск указанной службы `systemd`.
- `DaemonReload`: Перезагрузка всех конфигураций `systemd` units.

Пример: Политика прерывания узлов по умолчанию

Ниже приведена конфигурация по умолчанию для ресурса `MachineConfiguration` с именем `cluster` после установки:

```yaml
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfiguration
metadata:
  name: cluster
spec:
  defaultNodeDisruptionPolicySpecAction:
    files:
    - type: None
    units:
    - type: Reboot
  nodeDisruptionPolicy:
    sshkey:
      actions:
      - type: None
```

Эта конфигурация означает:

- Изменения файлов не вызывают никаких действий.
- Изменения в systemd units вызывают дренаж узла и перезагрузку.
- Изменения SSH-ключей применяются без прерывания.

Пример: Настройка поведения для файлов

Вы можете изменить действие по умолчанию для изменений файлов, чтобы они вызывали перезагрузку:

```yaml
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfiguration
metadata:
  name: cluster
spec:
  defaultNodeDisruptionPolicySpecAction:
    files:
    - type: Reboot
    units:
    - type: Reboot
  nodeDisruptionPolicy:
    sshkey:
      actions:
      - type: None
```

С этой конфигурацией любое изменение управляемого файла приведёт к дренажу и перезагрузке затронутого узла оператором Machine Configuration.

Пример: Применение политики к конкретному файлу

Вы также можете определить действия прерывания для конкретного пути файла. В следующем примере изменения файла `/usr/local/bin/myapp.sh` не вызовут перезагрузку узла, а вместо этого приведут к перезагрузке конфигурации systemd и перезапуску связанной службы:

```yaml
apiVersion: machineconfiguration.alauda.io/v1alpha1
kind: MachineConfiguration
metadata:
  name: cluster
spec:
  defaultNodeDisruptionPolicySpecAction:
    files:
    - type: Reboot
    units:
    - type: Reboot
  nodeDisruptionPolicy:
    files:
    - path: /usr/local/bin/myapp.sh
      actions:
      - type: DaemonReload
      - type: Restart
        restart:
          serviceName: myapp.service
    sshkey:
      actions:
      - type: None
```

В этом случае при обновлении `/usr/local/bin/myapp.sh` оператор Machine Configuration перезагрузит все `systemd` units и перезапустит `myapp.service` — без дренажа или перезагрузки узла.