

Реализация модели доступа на основе ролей в Nova на основе групп LDAP

Данный сценарий описывает реализацию политики доступа на основе ролей в Nova Container Platform для доступа к ресурсам kubernetes в воображаемом процессе разработки приложения. Для этого проектируется и реализуется ряд уровней доступа, соответствующих различных ролям в команде разработки.

В Nova уже имеются средства для настройки аутентификации и авторизации в режиме единого окна. Задача реализации матрицы доступа сводится к подключению провайдера аутентификации и конфигурации соответствующих ролей внутри kubernetes и StarVault. В примере используется LDAP в качестве провайдера аутентификации.

Порядок действий:

1. Спроектируйте модель доступа на основе ролей (RBAC) .
2. Создайте группы в LDAP-каталоге.
3. Сконфигурируйте метод аутентификации через Active Directory в StarVault и спроектированные роли.
4. Сконфигурируйте RBAC в kubernetes.

1. Проектирование модели доступа на основе ролей

В примере предполагается наличие ролей QA, DevOps, разработчиков, технических менеджеров, служебных учетных записей конвейера разработки и администраторов кластера.

Атрибутирование ролей пользователям осуществляется с помощью групп Active Directory или другого LDAP-каталога.

Ниже представлена таблица соответствия ролей воображаемой модели доступа ролям Kubernetes.

Роль в модели	Описание	Соответствие кластерным ролям Kubernetes в каждом пространстве имён	Соответствие кластерным ролям в масштабе кластера	Группа в LDAP-каталоге
QA	Получать список и содержимое объектов, журналы контейнеров, но не секреты.	view	view	auth-demo-qa-users + auth-demo-project-users
DevOps	Изменять и удалять объекты, кроме ресурсных квот.	admin	view	auth-demo-devops-users + auth-demo-project-users
Разработчики	Изменять объекты, кроме секретов.	edit	view	auth-demo-dev-users + auth-demo-project-users
Технические менеджеры	Дополнительно к правам DevOps: изменять ресурсные квоты, создавать и удалять.	admin + tech-man (дополнительная роль)	view	auth-demo-owner-users + auth-demo-project-users
Служебные учетные записи	Дополнительно к правам технических менеджеров: создавать CRD.	admin + tech-man (дополнительная роль)	view + ci-job (дополнительная роль)	отсутствует, создаются служебные учетные записи kubernetes + auth-demo-project-users
Администраторы кластера	Полный доступ, распределение квот, сетевых политик и политик доступа.		cluster-admin	auth-demo-sre

Особенности использования встроенных ролей на разных уровнях пространства имен и кластера:

1. роль `view` на кластерном уровне используется, чтобы избежать ошибок непредвиденных ошибок доступа, мешающих осуществлять навигацию внутри веб-

- консоли или получать базовые списки с помощью утилиты `kubectl`;
2. использования кластерных ролей в пространстве имен требует создания объекта `RoleBinding`, а на уровне кластера - `ClusterRoleBinding`;
 3. в данном примере для простоты конфигурации используются только кластерные роли `ClusterRole`, подробнее о средствах реализации доступа на основе ролей можно ознакомиться в разделе [Использование RBAC для разграничения доступа в Kubernetes](#).

2. Создание групп в LDAP-каталоге

Создайте группы в LDAP-каталоге и внесите в них соответствующих пользователей. Ниже представлен шаблон таблицы для создания групп, заполненный данными для примера сценария.

Группа	Группа в LDAP	Члены группы
Администраторы кластера	auth-demo-sre	ivan-petrov
Члены вымышленной проектной группы	auth-demo-project-users	auth-demo-qa-user, auth-demo-devops-user, auth-demo-dev-user, auth-demo-owner-user
QA-инженеры проекты	auth-demo-qa-users	auth-demo-qa-user
DevOps-инженеры проекта	auth-demo-devops-users	auth-demo-devops-user
Технический менеджер проекта	auth-demo-owner-users	auth-demo-owner-user

3. Конфигурация метода аутентификации

Выполните действия, согласно [инструкции](#).

- Для каждой роли сопоставьте группу пользователей LDAP согласно разделу [Настройка групп пользователей](#).
- Для каждой созданной группы сопоставьте oidc assignment согласно разделу [Настройка назначений](#).
- Единоажды разрешите доступ в приложения Nova по созданному сопоставлению согласно разделу* [Привязка назначений к приложениям](#)

4. Создание сервисной учетной записи

В целях демонстрации в данном примере используется вымышленная сервисная учетная запись. Чтобы ее использовать, создайте следующий объект в kubernetes.

YAML | ▾

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ci-service-account
  namespace: auth-demo-namespace
```

О получении реквизитов доступа этой учетной записи можно подробнее ознакомиться в статье базы знаний "[Создание учетной записи для не интерактивного доступа](#)".

5. Конфигурация модели RBAC в kubernetes

В данном примере используется имя auth-demo-namespace . Перед началом работы замените имена auth-demo-namespace , а также наименования групп на соответствующие вашему сценарию

1. Выдайте права администраторам кластера. Для этого создайте следующий объект в kubernetes:

- ClusterRoleBinding для администраторов кластера.

YAML | ▾

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: cluster-admin-binding
roleRef:
  name: cluster-admin
  kind: ClusterRole
subjects:
- kind: Group
  name: auth-demo-sre
  apiGroup: rbac.authorization.k8s.io
```

В дополнение ко встроенным кластерным ролям в данной модели используются специфичные для сценария роли. Поэтому в начале процесса конфигурации kubernetes необходимо создать эти роли. Для этого создайте следующие объекты в kubernetes:

- ClusterRole для технических менеджеров.

YAML | ▾

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: tech-man
rules:
- apiGroups: [""]
```

```
resources: ["resourcequotas", "resourcequotas/status"]
verbs: ["create", "get", "list", "watch", "update", "delete"]
```

- ClusterRole для служебных учетных записей.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: ci-job
rules:
  - apiGroups: ["apiextensions.k8s.io"]
    resources: ["customresourcedefinitions"]
    verbs: ["create", "get", "list", "watch", "update"]
```

2. Сопоставьте созданные роли группам в конкретных пространствах имен:

- RoleBinding для QA в пространстве имен auth-demo-namespace .

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: qa-view-binding
  namespace: auth-demo-namespace
roleRef:
  name: view
  kind: ClusterRole
subjects:
  - kind: Group
    name: auth-demo-qa-users
    apiGroup: rbac.authorization.k8s.io
```

- RoleBinding для DevOps в пространстве имен auth-demo-namespace .

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: devops-admin-binding
  namespace: auth-demo-namespace
roleRef:
  name: admin
  kind: ClusterRole
subjects:
  - kind: Group
    name: auth-demo-devops-users
    apiGroup: rbac.authorization.k8s.io
```

- RoleBinding для разработчиков в пространстве имен auth-demo-namespace .

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: dev-edit-binding
  namespace: auth-demo-namespace
roleRef:
  name: edit
  kind: ClusterRole
subjects:
- kind: Group
  name: auth-demo-dev-users
  apiGroup: rbac.authorization.k8s.io
```

- RoleBinding для технических менеджеров в пространстве имен auth-demo-namespace .

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: tech-man-admin-binding
  namespace: auth-demo-namespace
roleRef:
  name: admin
  kind: ClusterRole
subjects:
- kind: Group
  name: auth-demo-owner-users
  apiGroup: rbac.authorization.k8s.io
```

- RoleBinding для служебной учетной записи в пространстве имен auth-demo-namespace .

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: ci-admin-binding
  namespace: auth-demo-namespace
roleRef:
  name: admin
  kind: ClusterRole
subjects:
- kind: ServiceAccount
  name: ci-service-account
  namespace: auth-demo-namespace
```

- RoleBinding для технических менеджеров в пространстве имен auth-demo-namespace .

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: tech-man-quota-binding
  namespace: auth-demo-namespace
roleRef:
  name: tech-man
  kind: ClusterRole
subjects:
- kind: Group
  name: auth-demo-owner-users
  apiGroup: rbac.authorization.k8s.io
```

3. Выдайте права на уровне кластера:

- ClusterRoleBinding для QA на уровне кластера.

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: auth-demo-view-cluster-binding
roleRef:
  name: view
  kind: ClusterRole
subjects:
- kind: Group
  name: auth-demo-project-users
  apiGroup: rbac.authorization.k8s.io
```

- ClusterRoleBinding для служебных учетных записей на уровне кластера.

YAML | □

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: ci-job-cluster-binding
roleRef:
  name: ci-job
  kind: ClusterRole
subjects:
- kind: ServiceAccount
  name: ci-service-account
  namespace: auth-demo-namespace
```

Использование RBAC для разграничения доступа в Kubernetes

1. Об RBAC в Kubernetes

Объекты RBAC (Role-based access control) в Kubernetes определяют, разрешена ли пользователю определенная операция в контексте всего кластера или в контексте пространства имен (*Namespace*).

Администраторы кластера Kubernetes могут использовать кластерные роли (*ClusterRoles*) и их привязки (*ClusterRoleBindings*) к пользовательским объектам, чтобы контролировать тот или иной доступ к ресурсам Kubernetes, пространствам имен и другим сущностям в контексте всего кластера.

Регулярные пользователи кластера могут использовать локальные роли (*Roles*) и их локальные привязки (*RoleBindings*), чтобы контролировать доступ к собственным пространствам имен (*Namespaces*).

Авторизация в Kubernetes управляет с помощью следующих объектов:

Объект авторизации	Описание
Правила (<i>Rules</i>)	Перечень разрешенных методов работы с объектами Kubernetes.
Роли (<i>Roles</i>)	Набор правил, определяющий разрешенные действия с объектами Kubernetes.
Привязки (<i>Bindings</i>)	Ассоциация между пользователями или группами с какой-либо ролью.

В Kubernetes предусмотрено два уровня ролей RBAC и их привязок:

Уровень RBAC	Описание
Кластерный RBAC	Кластерные роли и привязки, которые могут применяться на уровне всего кластера.

Уровень RBAC	Описание
Локальный RBAC	Локальные роли и привязки, которые могут применяться на уровне пространства имен. При этом, в привязке может указываться также и кластерная роль, описывающая какие-либо действия в Kubernetes.



Для удобства администрирования используйте кластерные роли (*ClusterRoles*) в локальных привязках (*RoleBindings*) и создавайте локальные роли (*Roles*) только при необходимости.

Данная двухуровневая иерархия позволяет переиспользовать одни и те же кластерные роли (*ClusterRoles*) в пределах пространств имен, а также сохраняет возможность установки дополнительных локальных ролей.

В результате какого-либо действия пользователя в Kubernetes предварительно оцениваются правила в ролях (*Roles*), назначенных ему с помощью привязок (*Bindings*):

- Выполняется проверка разрешений по кластерным ролям (*ClusterRoles*)
- Выполняется проверка разрешений по локальным ролям (*Roles*)
- Запрещается все, что явно не разрешено.

1.1. Кластерные роли по умолчанию

Nova Container Platform включает базовый набор кластерных ролей (*ClusterRoles*), которые вы можете использовать для назначения пользователям и группам в контекстах кластера и пространств имен.



Не рекомендуется изменять базовые кластерные роли (*ClusterRoles*). Изменение базовых кластерных ролей или системных ролей может привести к некорректной работе Kubernetes. При необходимости вы можете продублировать базовую кластерную роль и внести в нее изменения.

Кластерная роль	Описание
cluster-admin	Роль, определяющая права супер-пользователя. Данный пользователь может выполнить любое действие с любым объектом в кластере, если роль привязана с помощью <i>ClusterRoleBinding</i> . Если роль привязана с помощью <i>RoleBinding</i> , то пользователь сможет управлять всеми ресурсами пространства имен, в том числе квотами.

Кластерная роль	Описание
admin	Роль администратора пространства имен. Пользователь может управлять всеми ресурсами пространства имен кроме квот.
edit	Роль пользователя в пространстве имен, позволяющая выполнять операции с большинством объектов в пространстве имен за исключением ролей и их привязок.
view	Роль пользователя, который не может производить какие-либо изменения в Kubernetes, но может просматривать большинство объектов кроме ролей, их привязок, некоторых CR и секретов.

2. Просмотр кластерных ролей и их привязок

Вы можете использовать утилиту `kubectl` для просмотра и оценки кластерных ролей и их привязок.

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.

Процедура

1. Для просмотра кластерных ролей и их правил выполните команду:

```
BASH | kubectl describe clusterrole.rbac
```

Пример

```
BASH | $ kubectl describe clusterrole.rbac

Name:           admin
Labels:         kubernetes.io/bootstrapping=rbac-defaults
Annotations:   rbac.authorization.kubernetes.io/autoupdate: true
PolicyRule:
  Resources           Non-Resource URLs
  Resource Names    Verbs
  -----             -----
  leases.coordination.k8s.io      []          []
  [create delete deletecollection get list patch update watch]
```

```
rolebindings.rbac.authorization.k8s.io          []
[create delete deletecollection get list patch update watch]
  roles.rbac.authorization.k8s.io          []
[create delete deletecollection get list patch update watch]
  configmaps          []
[create delete deletecollection patch update get list watch]
  events          []
[create delete deletecollection patch update get list watch]
  persistentvolumeclaims          []
[create delete deletecollection patch update get list watch]
  pods          []
[create delete deletecollection patch update get list watch]
  replicationcontrollers/scale          []
[create delete deletecollection patch update get list watch]
  replicationcontrollers          []
[create delete deletecollection patch update get list watch]
  services          []
[create delete deletecollection patch update get list watch]
  challenges.acme.cert-manager.io          []
[create delete deletecollection patch update get list watch]
  orders.acme.cert-manager.io          []
[create delete deletecollection patch update get list watch]
  daemonsets.apps          []
[create delete deletecollection patch update get list watch]
  deployments.apps/scale          []
[create delete deletecollection patch update get list watch]
  deployments.apps          []
[create delete deletecollection patch update get list watch]
  replicasetss.apps/scale          []
[create delete deletecollection patch update get list watch]
  replicasetss.apps          []
[create delete deletecollection patch update get list watch]
  statefulsetss.apps/scale          []
...
...
```

2. Для просмотра кластерных привязок с пользователями или группами, выполните команду:

```
kubectl describe clusterrolebinding.rbac
```

BASH | ↗

Пример

```
$ kubectl describe clusterrolebinding.rbac
```

BASH | ↗

```
Name:      cluster-admin
Labels:    kubernetes.io/bootstrapping=rbac-defaults
Annotations: rbac.authorization.kubernetes.io/autoupdate: true
Role:
  Kind:  ClusterRole
```

```
Name: cluster-admin
Subjects:
  Kind   Name           Namespace
  ----  ---             -----
  Group  system:masters

Name:      cluster-reconciler
Labels:    app.kubernetes.io/instance=nova-gitops
          app.kubernetes.io/part-of=flux
Annotations: <none>
Role:
  Kind: ClusterRole
  Name: cluster-admin
Subjects:
  Kind       Name           Namespace
  ----       ---             -----
  ServiceAccount  kustomize-controller  nova-gitops
  ServiceAccount  helm-controller     nova-gitops

Name:      crd-controller
Labels:    app.kubernetes.io/instance=nova-gitops
          app.kubernetes.io/part-of=flux
Annotations: <none>
Role:
  Kind: ClusterRole
  Name: crd-controller
Subjects:
  Kind       Name           Namespace
  ----       ---             -----
  ServiceAccount  kustomize-controller  nova-gitops
  ServiceAccount  helm-controller     nova-gitops
  ServiceAccount  source-controller   nova-gitops
  ServiceAccount  notification-controller  nova-gitops
  ServiceAccount  image-reflector-controller  nova-gitops
  ServiceAccount  image-automation-controller  nova-gitops

Name:      hubble-generate-certs
Labels:    app.kubernetes.io/managed-by=Nova
          kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations: <none>
Role:
  Kind: ClusterRole
  Name: hubble-generate-certs
Subjects:
  Kind       Name           Namespace
  ----       ---             -----
  ServiceAccount  hubble-generate-certs  kube-system
```

```
Name:          hubble-ui
Labels:        app.kubernetes.io/managed-by=Nova
               kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations:   <none>
Role:
  Kind: ClusterRole
  Name: hubble-ui
Subjects:
  Kind      Name      Namespace
  ----      ----      -----
  ServiceAccount  hubble-ui  kube-system
```

```
Name:          kubeadm:get-nodes
Labels:        <none>
Annotations:  <none>
Role:
  Kind: ClusterRole
  Name: kubeadm:get-nodes
Subjects:
  Kind    Name           Namespace
  ----    ----           -----
  Group   system:bootstrappers:kubeadm:default-node-token
...
...
```

3. Просмотр локальных ролей и их привязок

Вы можете использовать утилиту `kubectl` для просмотра и оценки локальных ролей и их привязок.

Необходимые условия

- ✓ У вас есть доступ к кластеру с учетной записью, имеющей роль `cluster-admin` в Kubernetes или доступ к кластеру с учетной записью `admin` для просмотра ролей и привязок в пространстве имен.
- ✓ Вы установили утилиту `kubectl` для работы с Kubernetes.

Процедура

1. Для просмотра локальных ролей и их правил в пространстве имен, например `nova-cert-management`, выполните команду:

```
kubectl describe roles.rbac -n nova-cert-management
```

BASH | ↗

Пример

BASH | □

```
$ kubectl describe roles.rbac -n nova-cert-management
```

```
Name: nova-cert-manager-startupapicheck:create-cert
Labels: app=startupapicheck
        app.kubernetes.io/component=startupapicheck
        app.kubernetes.io/managed-by=Nova
        app.kubernetes.io/name=startupapicheck
        app.kubernetes.io/version=v1.12.3
        kustomize.toolkit.fluxcd.io/name=nova-release-cert-manager-
main
        kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations: <none>
PolicyRule:
  Resources           Non-Resource URLs  Resource Names  Verbs
  -----
  certificates.cert-manager.io  []          []            [create]
```

```
Name: nova-cert-manager-webhook:dynamic-serving
Labels: app=webhook
        app.kubernetes.io/component=webhook
        app.kubernetes.io/managed-by=Nova
        app.kubernetes.io/name=webhook
        app.kubernetes.io/version=v1.12.3
        kustomize.toolkit.fluxcd.io/name=nova-release-cert-manager-
main
        kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations: <none>
PolicyRule:
  Resources   Non-Resource URLs  Resource Names      Verbs
  -----
  secrets     []                  []                [create]
  secrets     []                  [nova-cert-manager-webhook-ca]  [get list
watch update]
```

2. Для просмотра локальных привязок с пользователями или группами в пространстве имен, например `nova-monitoring`, выполните команду:

BASH | □

```
kubectl describe rolebinding.rbac -n nova-monitoring
```

Пример

BASH | □

```
$ kubectl describe rolebinding.rbac -n nova-monitoring

Name: nova-grafana
Labels: app.kubernetes.io/managed-by=Nova
        app.kubernetes.io/name=nova-release-grafana
        app.kubernetes.io/version=10.0.3
        kustomize.toolkit.fluxcd.io/name=nova-release-grafana-main
```

```
kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations: <none>
Role:
  Kind:  Role
  Name:  nova-grafana
Subjects:
  Kind        Name        Namespace
  ----        ---        -----
  ServiceAccount  nova-grafana  nova-monitoring

Name:          nova-metrics-server-auth-reader
Labels:        app.kubernetes.io/managed-by=Nova
               app.kubernetes.io/name=nova-release-metrics-server
               app.kubernetes.io/version=0.6.4
               kustomize.toolkit.fluxcd.io/name=nova-release-metrics-
server-main
               kustomize.toolkit.fluxcd.io/namespace=nova-gitops
Annotations: <none>
Role:
  Kind:  Role
  Name:  extension-apiserver-authentication-reader
Subjects:
  Kind        Name        Namespace
  ----        ---        -----
  ServiceAccount  nova-metrics-server  nova-monitoring
```

4. Добавление ролей пользователям и группам

Вы можете использовать утилиту `kubectl` для управления ролями и их привязками или веб-интерфейс *Nova Console*.

► [Web UI](#)

► [CLI](#)

5. Создание ролей

Вы можете использовать утилиту `kubectl` для управления ролями и их привязками или веб-интерфейс *Nova Console*.

► [Web UI](#)

► CLI