



Руководство по установке
версия 0.7.1

ООО «Веб-Сервер»

июн. 02, 2025

Оглавление

1	Аннотация	1
1.1	Общие сведения	1
1.2	Системные требования	2
2	Установка	3
2.1	Установка с помощью Helm	3
2.1.1	Получение доступа к образу ANIC	4
2.1.2	Скачивание образа ANIC	4
2.1.3	Подключение лицензии	4
2.1.4	Получение и настройка Helm-чарта	4
2.1.5	Обновление CRD	5
2.1.6	Удаление CRD	5
2.1.7	Конфигурация	5
2.2	Установка с помощью манифестов	13
2.2.1	Предварительные требования	13
2.2.2	Настройка RBAC	13
2.2.3	Создание ресурсов	17
2.2.4	Развертывание ANIC	53
2.3	Подключение лицензии	56
2.4	Установка и настройка для Яндекс.Облака	57
2.4.1	Настройка локальной рабочей среды для работы с Kubernetes и Яндекс.Облаком	57
2.4.2	Получение доступа к образу ANIC	57
2.4.3	Получение и настройка Helm-чартов	57
2.4.4	Запуск менеджера сертификатов	58
2.4.5	Развертывание ANIC и тестового приложения в Kubernetes	59
2.4.6	Проверка работы	60
2.4.7	Удаление развернутого приложения	60
2.4.8	Проксирование TCP-трафика через ANIC в Kubernetes	60
2.5	Миграция с Ingress-NGINX Controller на ANIC	64
2.5.1	Перенос конфигурации	64
2.5.2	Соответствия для аннотаций	64
2.5.3	Глобальная конфигурация с помощью ConfigMap	65
2.5.4	Примеры	66
3	Аргументы командной строки	74
3.1	-angie-configmaps <строка>	74
3.2	-angie-debug	74
3.3	-angie-reload-timeout <значение>	75
3.4	-angie-status	75
3.5	-angie-status-allow-cidrs <строка>	75

3.6	-angie-status-port <int>	75
3.7	-angie-status-prometheus <bool>	75
3.8	-angie-status-prometheus-allow-cidrs	75
3.9	-angie-status-prometheus-path <строка>	75
3.10	-angie-status-prometheus-port <int>	76
3.11	-default-server-tls-secret <строка>	76
3.12	-disable-ipv6	76
3.13	-enable-cert-manager	76
3.14	-enable-custom-resources	76
3.15	-enable-external-dns	76
3.16	-enable-jwt	77
3.17	-enable-leader-election	77
3.18	-enable-oidc	77
3.19	-enable-prometheus-metrics	77
3.20	-enable-service-insight	77
3.21	-enable-snippets	77
3.22	-enable-tls-passthrough	77
3.23	-external-service <строка>	78
3.24	-global-configuration <строка>	78
3.25	-health-status	78
3.26	-health-status-uri <строка>	78
3.27	-ingress-class <строка>	78
3.28	-ingresslink <строка>	78
3.29	-ingress-template-path <строка>	79
3.30	-leader-election-lock-name <строка>	79
3.31	-main-template-path <строка>	79
3.32	-prometheus-metrics-listen-port <int>	79
3.33	-prometheus-tls-secret <строка>	79
3.34	-proxy <строка>	79
3.35	-ready-status	80
3.36	-ready-status-port	80
3.37	-report-ingress-status	80
3.38	-service-insight-listen-port <int>	80
3.39	-service-insight-tls-secret <строка>	80
3.40	-tls-passthrough-port <int>	80
3.41	-transportserver-template-path <строка>	81
3.42	-v <значение>	81
3.43	-version	81
3.44	-virtualserver-template-path <строка>	81
3.45	-vmodule <значение>	81
3.46	-watch-namespace <строка>	81
3.47	-watch-namespace-label <строка>	81
3.48	-watch-secret-namespace <строка>	81
3.49	-wildcard-tls-secret <строка>	82
4	Версии Angie Ingress Controller (ANIC)	83
4.1	2025	83
4.1.1	ANIC 0.7.1	83
4.1.2	ANIC 0.7.0	83
4.2	2024	84
4.2.1	ANIC 0.6.0	84
4.2.2	ANIC 0.5.0	84
4.2.3	ANIC 0.4.0	85
4.2.4	ANIC 0.3.0	86
4.3	2023	86
4.3.1	ANIC 0.2.0	86
5	Права на интеллектуальную собственность	88

ГЛАВА 1

Аннотация

Angie Ingress Controller (ANIC) — приложение, которое запускается в кластере и управляет балансировщиком нагрузки.

ANIC использует в своей работе [Angie PRO](#) — эффективный, мощный и масштабируемый веб-сервер, который позволяет балансировать нагрузку между серверами как по протоколам TCP/UDP, так и по HTTP.

Примечание

Angie PRO внесен в Единый реестр российских программ для электронных вычислительных машин и баз данных (запись № 17604).

1.1 Общие сведения

Angie Ingress Controller (ANIC) - это решение для управления трафиком контейнеризированных приложений в Kubernetes.

ANIC разворачивается и работает в кластере, управляя функциями Ingress с возможностью настройки правил обработки трафика. Продукт базируется на [Angie PRO](#), что позволяет строить безопасные масштабируемые высокопроизводительные окружения, используя российское решение с профессиональными сервисами миграции и технической поддержки на русском языке.

ANIC использует широкий набор функций Ingress:

- *Балансировка нагрузки TCP, UDP, TLS, HTTP, gRPC*: Гибкое распределение трафика и его плавного переноса при обновлениях приложений
- *Терминирование сессий TLS*: Подтверждения подлинности сервисов и защиты онлайн-транзакций
- *Настройки гибкого логирования*: Управление современными динамическими приложениями
- *Расширенная маршрутизация трафика*: Разделение трафика и расширенная маршрутизация на основе содержимого
- *Ограничение поступающего трафика*: По различным критериям для защиты приложений от DDoS

- *Модификация ответов на запросы:* На уровне балансировщика HTTP

1.2 Системные требования

Список поддерживаемых ОС и архитектур:

ОС	Версии	Архитектуры
Alpine Linux	3.21	x86_64, arm64
Alt Linux	10	x86_64, arm64
Debian	11	x86_64, arm64

ГЛАВА 2

Установка

В этом разделе описаны доступные способы установки ANIC, а также миграция с Ingress-NGINX Controller на ANIC.

Установка с помощью Helm

Получение лицензии

Установка и настройка для Яндекс.Облака

Миграция с Ingress-NGINX Controller на ANIC

2.1 Установка с помощью Helm

Angie Ingress Controller (ANIC) разворачивается в кластере Kubernetes с помощью Helm.

Программные требования:

- Kubernetes 1.22+
- Helm 3.0+

Поддерживаемые дистрибутивы:

Название	Версии	Архитектуры
Alpine Linux	3.21	x86_64, ARM64
Debian	11 "Bullseye"	x86_64, ARM64
ALT Linux	10	x86_64, ARM64

2.1.1 Получение доступа к образу ANIC

1. Авторизуйтесь в Docker Registry, чтобы получить доступ к образу ANIC:

```
$ docker login -u=<login> -p=<password> anic.docker.angie.software
```

После успешной авторизации информация о доступе будет сохранена в файле `~/.docker/config.json`.

2. Создайте секрет для Kubernetes, используя файл Docker-конфигурации:

```
$ kubectl create secret generic regcred \
  --from-file=.dockerconfigjson=$HOME/.docker/config.json \
  --type=kubernetes.io/dockerconfigjson
```

Секрет `regcred` будет использоваться для доступа к приватным образам в Kubernetes.

2.1.2 Скачивание образа ANIC

Скачайте образ ANIC и перенесите его в свой личный репозиторий, для этого:

1. Загрузите образ ANIC:

```
$ docker pull anic.docker.angie.software/anic:latest
```

2. Переименуйте образ:

```
$ docker tag anic.docker.angie.software/anic:latest <ваш_репозиторий>/<образ_
  →anic>
```

3. Отправьте переименованный образ в свой репозиторий:

```
$ docker push <ваш_репозиторий>/<образ_anic>
```

4. В файле `values.yaml` обновите поле `controller.image.repository` соответственно.

2.1.3 Подключение лицензии

Подключите лицензию для ANIC, следуя [инструкции](#).

2.1.4 Получение и настройка Helm-чарта

1. Склонировать Helm-чарт из репозитория Angie Software.

Доступ к репозиторию можно получить, обратившись на info@wbsrv.ru.

```
$ git clone https://git.angie.software/web-server/anic-helm-charts.git
```

2. Отредактируйте файл `values.yaml`, указав имя секрета, созданного для доступа к образам:

```
imagePullSecretName: "regcred"
```

3. Соберите и установите Helm-чарт:

```
$ helm install <имя_релиза> <путь_до_чарта>
```

Helm сгенерирует манифесты Kubernetes на основе шаблонов (`templates/`) и значений (`values.yaml`), а затем установит их в кластере.

Примечание

По умолчанию для ANIC требуется несколько CRD (определений пользовательских ресурсов), установленных в кластере. Helm устанавливает их автоматически. Если не установить CRD, поды ANIC не будут готовы (Ready). Если вы не используете пользовательские ресурсы, для которых требуются CRD (что соответствует параметру `controller.enableCustomResources`, установленному как `false`), установку CRD можно пропустить, указав в команде `helm install` параметр `--skip-crds`.

2.1.5 Обновление CRD

Чтобы обновить CRD, скачайте исходные файлы Helm-чарта, а затем выполните команду:

```
$ kubectl apply -f crds/
```

Примечание

Может появиться следующее предупреждение, но его можно проигнорировать:

```
Warning: kubectl apply should be used on resource created by either
kubectl create --save-config or kubectl apply
```

2.1.6 Удаление CRD

Чтобы удалить CRD, скачайте исходные файлы Helm-чарта, а затем выполните команду:

```
$ kubectl delete -f crds/
```

Примечание

Эта команда удалит все соответствующие CRD в вашем кластере во всех пространствах имен. Убедитесь, что в кластере нет пользовательских ресурсов, которые вы хотите сохранить, и не запущены другие экземпляры ANIC.

2.1.7 Конфигурация

В таблице ниже перечислены настраиваемые параметры Helm-чарта Ingress Controller и их значения по умолчанию.

Параметр	Описание	По умолчанию
<code>controller.name</code>	Имя daemonset или deployment ANIC.	Создается автоматически
<code>controller.kind</code>	Тип установки ANIC - deployment или daemonset.	deployment

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
controller.annotations	Позволяет устанавливать аннотации для deployment или daemonset.	{}
controller.angiePro	Развертывает ANIC для Angie PRO.	false
controller.reloadTimeout	Время ожидания в миллисекундах, в течение которого ANIC будет ожидать успешной перезагрузки Angie после изменения или при начальном запуске.	60000
controller.hostNetwork	Позволяет подам ANIC использовать сетевое пространство имен хоста.	false
controller.dnsPolicy	Политика DNS для подов ANIC.	ClusterFirst
controller.debug	Включает отладку для Angie. Требуется задать значение <code>error-log-level: debug</code> в ConfigMap через <code>controller.config.entries</code> .	false
controller.logLevel	Уровень ведения журнала ANIC.	1
controller.image.digest	Дайджест образа ANIC.	Нет
controller.image.repository	Репозиторий образов ANIC.	myregistry.host.ru/angie-ingress
controller.image.tag	Тег образа ANIC.	0.1.2
controller.image.pullPolicy	Политика скачивания образа ANIC.	IfNotPresent
controller.lifecycle	Жизненный цикл подов ANIC.	{}
controller.customConfigMap	Имя пользовательской ConfigMap, используемой ANIC. Если имя задано, то конфигурация по умолчанию игнорируется.	""
controller.config.name	Имя ConfigMap, используемой ANIC.	Создается автоматически
controller.config.annotations	Аннотации к ConfigMap в ANIC.	{}
controller.config.entries	Записи в ConfigMap для настройки конфигурации Angie.	{}
controller.customPorts	Список пользовательских портов, которые должны быть доступны в поде ANIC. Следует обычному синтаксису <code>yaml</code> Kubernetes для контейнерных портов.	[]
controller.defaultTLS.cert	Сертификат TLS в кодировке Base64 для сервера HTTPS по умолчанию.	Нет

Примечание

Рекомендуется указать свой собственный сертификат. Альтернативное решение: полный пропуск секрета сервера по умолчанию приведет к тому, что Angie будет по умолчанию отклонять TLS-подключения к серверу.

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
controller.defaultTLS.key	<p>Ключ TLS в кодировке Base64 для сервера HTTPS по умолчанию.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Примечание</p> <p>Рекомендуется указать свой собственный ключ. Альтернативное решение: полный пропуск секрета сервера по умолчанию приведет к тому, что Angie будет по умолчанию отклонять TLS-подключения к серверу.</p> </div>	Нет
controller.defaultTLS.secret	<p>Секрет с сертификатом TLS и ключом для сервера HTTPS по умолчанию. Значение должно соответствовать следующему формату: <пространство имен>/<имя>. Используется в качестве альтернативы указанию сертификата и ключа с помощью параметров controller.defaultTLS.cert и controller.defaultTLS.key.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>Примечание</p> <p>Альтернативное решение: полный пропуск секрета сервера по умолчанию приведет к тому, что Angie будет по умолчанию отклонять TLS-подключения к серверу.</p> </div>	Нет
controller.wildcardTLS.cert	Сертификат TLS в кодировке Base64 для каждого узла Ingress или VirtualServer, у которого включен TLS, но не указан секрет. Если параметр не задан, Angie прервет любую попытку установить TLS-соединение для таких узлов Ingress или VirtualServer.	Нет
controller.wildcardTLS.key	Ключ TLS в кодировке Base64 для каждого узла Ingress или VirtualServer, у которого включен TLS, но не указан секрет. Если параметр не задан, Angie прервет любую попытку установить TLS-соединение для таких узлов Ingress или VirtualServer.	Нет
controller.wildcardTLS.secret	Секрет с сертификатом TLS и ключом для каждого узла Ingress или VirtualServer, у которого включен TLS, но не указан секрет. Значение должно соответствовать следующему формату: <пространство имен>/<имя>. Используется в качестве альтернативы указанию сертификата и ключа с помощью параметров controller.wildcardTLS.cert и controller.wildcardTLS.key.	Нет
controller.nodeSelector	Селектор узлов для назначения подов ANIC.	{}
controller.terminationGrace	Период плавного завершения работы пода ANIC.	30
controller.tolerations	Допуски подов ANIC (tolerations).	[]
controller.affinity	Привязка подов ANIC (affinity).	{}
controller.topologySpreadCo	Ограничения на распространение топологии подов ANIC.	{}
controller.env	Дополнительные переменные окружения, которые должны быть установлены на подах ANIC.	[]
controller.volumes	Тома подов ANIC.	[]
controller.volumeMounts	Точки подключения томов подов ANIC.	[]
controller.initContainers	Значение initContainers для подов ANIC.	[]

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
<code>controller.extraContainers</code>	Дополнительные контейнеры (например, сайдкар) для подов Ingress Controller.	[]
<code>controller.resources</code>	Ресурсы подов ANIC.	requests: cpu=100m,memory=128M
<code>controller.replicaCount</code>	Количество реплик deployment ANIC.	1
<code>controller.ingressClass</code>	Класс ANIC. Должен быть развернут ресурс IngressClass с именем, тождественным этому классу. В противном случае ANIC не запустится. ANIC обрабатывает только те ресурсы, которые принадлежат его классу, т. е. их ресурс поля "ingressClassName" совпадает с классом. ANIC обрабатывает все ресурсы VirtualServer, VirtualServerRoute и TransportServer, которые не имеют поля "ingressClassName", во всех версиях Kubernetes.	angie
<code>controller.setAsDefaultIngress</code>	Новым Ingress без указанного поля "ingressClassName" будет присвоен класс, указанный в <code>controller.ingressClass</code> .	false
<code>controller.watchNamespace</code>	Разделенный запятыми список пространств имен, за ресурсами которых должен следить ANIC. По умолчанию ANIC отслеживает все пространства имен. Взаимоисключающие с <code>controller.watchNamespaceLabel</code> . Обратите внимание, что при настройке нескольких пространств имен с использованием опции Helm cli <code>--set</code> строка должна быть заключена в двойные кавычки, а запятые экранированы с помощью обратной косой черты - например, <code>--set controller.watchNamespace="default\,anic"</code> .	""
<code>controller.watchNamespaceLabel</code>	Настраивает в ANIC просмотр только пространств имен с меткой <code>foo=bar</code> . По умолчанию ANIC отслеживает все пространства имен. Взаимоисключающая с <code>controller.watchNamespace</code> настройка.	""
<code>controller.watchSecretNamespaces</code>	Разделенный запятыми список пространств имен, за которыми Ingress Controller должен следить в поисках ресурсов типа Secret. Если этот параметр не настроен, ANIC отслеживает одни и те же пространства имен в поисках всех ресурсов. См. также <code>controller.watchNamespace</code> и <code>controller.watchNamespaceLabel</code> . Обратите внимание, что при настройке нескольких пространств имен с использованием опции Helm cli <code>--set</code> строка должна быть заключена в двойные кавычки, а запятые экранированы с помощью обратной косой черты - например, <code>--set controller.watchSecretNamespaces="default\,angie-ingress"</code> .	""
<code>controller.enableCustomResources</code>	Включает пользовательские ресурсы.	true
<code>controller.enableTLSPassthrough</code>	Включает передачу данных по протоколу TLS на порту 443. Требуется <code>controller.enableCustomResources</code> .	false
<code>controller.enableCertManager</code>	Включает автоматическое управление сертификатами x509 для ресурсов виртуального сервера с помощью cert-manager (cert-manager.io). Требуется <code>controller.enableCustomResources</code> .	false
<code>controller.enableExternalDNS</code>	Включает интеграцию с ExternalDNS для настройки общедоступных записей DNS у ресурсов VirtualServer с использованием ExternalDNS. Требуется <code>controller.enableCustomResources</code> .	false
<code>controller.globalConfiguration.create</code>	Создает пользовательский ресурс GlobalConfiguration. Требуется <code>controller.enableCustomResources</code> .	false
<code>controller.globalConfiguration.spec</code>	Спецификация GlobalConfiguration для определения параметров глобальной конфигурации ANIC.	{}

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
<code>controller.enableSnippets</code>	Включает пользовательские фрагменты конфигурации Angie в ресурсах Ingress, VirtualServer, VirtualServerRoute и TransportServer.	false
<code>controller.healthStatus</code>	Добавляет местоположение <code>"/angie-health"</code> на сервер по умолчанию. Местоположение отвечает кодом статуса 200 на любой запрос. Это полезно для внешней проверки работоспособности ANIC.	false
<code>controller.healthStatusURI</code>	Задаёт URI местоположения состояния работоспособности на сервере по умолчанию. Требуется <code>controller.HealthStatus</code> .	<code>"/angie-health"</code>
<code>controller.angieStatus.enable</code>	Включает в Angie API.	true
<code>controller.angieStatus.port</code>	Задаёт порт, на котором доступен Angie API.	8080
<code>controller.angieStatus.allowCidrs</code>	Добавляет блоки IP или CIDR в список разрешенных для Angie API. Несколько IP или CIDR разделяются запятыми.	127.0.0.1,::1
<code>controller.priorityClassName</code>	Класс приоритета подов ANIC.	Нет
<code>controller.service.create</code>	Создаёт сервис для предоставления доступа к подам ANIC.	true
<code>controller.service.type</code>	Тип сервиса, который необходимо создать для ANIC.	LoadBalancer
<code>controller.service.externalTrafficPolicy</code>	Внешняя политика трафика сервиса. Значение Local сохраняет исходный IP-адрес клиента.	Local
<code>controller.service.annotations</code>	Аннотации сервиса ANIC.	{}
<code>controller.service.extraLabels</code>	Экстра-метки сервиса.	{}
<code>controller.service.loadBalancerIP</code>	Статический IP-адрес для балансировщика нагрузки. Для <code>controller.service.type</code> должно быть установлено значение LoadBalancer. Поставщик облачных услуг должен поддерживать эту функцию.	""
<code>controller.service.externalIPs</code>	Список внешних IP-адресов для сервиса ANIC.	[]
<code>controller.service.loadBalancerSourceRanges</code>	Диапазоны IP-адресов (CIDR), которым разрешен доступ к балансировщику нагрузки. Для <code>controller.service.type</code> должно быть установлено значение LoadBalancer. Поставщик облачных услуг должен поддерживать эту функцию.	[]
<code>controller.service.name</code>	Имя сервиса.	Создается автоматически
<code>controller.service.customPorts</code>	Список пользовательских портов, которые будут доступны через сервис ANIC. Следует обычному синтаксису yaml Kubernetes для портов сервиса.	[]

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
controller.service.httpPort.enable	Включает HTTP-порт для сервиса ANIC.	true
controller.service.httpPort.port	HTTP-порт сервиса ANIC.	80
controller.service.httpPort.nodePort	Пользовательский NodePort для HTTP-порта. Для controller.service.type должно быть установлено значение NodePort.	""
controller.service.httpPort.targetPort	Целевое значение HTTP-порта сервиса ANIC.	80
controller.service.httpsPort.enable	Включает порт HTTPS для сервиса ANIC.	true
controller.service.httpsPort.port	HTTPS-порт сервиса ANIC.	443
controller.service.httpsPort.nodePort	Пользовательский NodePort для HTTPS-порта. Для controller.service.type должно быть установлено значение NodePort.	""
controller.service.httpsPort.targetPort	Целевой порт HTTPS-порта сервиса ANIC.	443
controller.serviceAccount.annotations	Аннотации учетной записи сервиса ANIC.	{}
controller.serviceAccount.name	Имя учетной записи сервиса подов ANIC. Используется для RBAC.	Создается автоматически
controller.serviceAccount.imagePullSecretName	Имя секретного файла, содержащего учетные данные реестра Docker. Секрет должен находиться в том же пространстве имен, что и релиз Helm.	""
controller.serviceMonitor.name	Имя serviceMonitor.	Создается автоматически
controller.serviceMonitor.create	Создает пользовательский ресурс ServiceMonitor.	false
controller.serviceMonitor.labels	Метки объектов Kubernetes для применения к объекту serviceMonitor.	""

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
<code>controller.serviceMonitor.selectorMatchLabels</code>	Набор меток, позволяющих выбирать конечные точки для <code>serviceMonitor</code> .	" "
<code>controller.serviceMonitor.endpoints</code>	Список конечных точек, разрешенных в составе этого <code>serviceMonitor</code> .	" "
<code>controller.reportIngressStatus.enable</code>	Добавляет в поле адреса в статусе ресурсов Ingress внешний адрес Ingress Controller. Нужно также указать источник внешнего адреса через внешнюю службу через <code>controller.reportIngressStatus.externalService</code> , либо через <code>controller.reportIngressStatus.ingressLink</code> , либо через запись <code>external-status-address</code> в ConfigMap через <code>controller.config.entries</code> .	true
<div style="border: 1px solid #ccc; padding: 5px; background-color: #e6f2ff;"> <p>Примечание</p> <p>Значение <code>controller.config.entries.external-status-address</code> имеет приоритет над остальными.</p> </div>		
<code>controller.reportIngressStatus.externalService</code>	Указывает имя сервиса с типом LoadBalancer, через который Ingress Controller будет доступен извне. Внешний адрес сервиса используется для отчетов о состоянии ресурсов Ingress, VirtualServer и VirtualServerRoute. Значение <code>controller.reportIngressStatus.enable</code> должно быть задано как true. Значение по умолчанию создается автоматически и включается, когда <code>controller.service.create</code> имеет значение true, а <code>controller.service.type</code> - значение LoadBalancer.	Создается автоматически
<code>controller.reportIngressStatus.ingressLink</code>	Указывает имя ресурса IngressLink, который предоставляет доступ к подам ANIC через систему BIG-IP. IP-адрес системы BIG-IP используется для отчетов о состоянии ресурсов Ingress, VirtualServer и VirtualServerRoute. Значение <code>controller.reportIngressStatus.enable</code> должно быть задано как true.	" "
<code>controller.reportIngressStatus.enableLeaderElection</code>	Включает выбор лидера, чтобы избежать ситуации, когда несколько реплик контроллера сообщают о состоянии ресурсов Ingress. Значение <code>controller.reportIngressStatus.enable</code> должно быть задано как true.	true
<code>controller.reportIngressStatus.leaderElectionLock</code>	Указывает имя ConfigMap в том же пространстве имен, что и контроллер, которое используется для блокировки выбора лидера. Значение <code>controller.reportIngressStatus.enableLeaderElection</code> должно быть задано как true.	Создается автоматически
<code>controller.reportIngressStatus.annotations</code>	Аннотации к конфигурационной карте выборов лидера.	{}
<code>pod.annotations</code>	Аннотации пода ANIC.	{}
<code>pod.extraLabels</code>	Дополнительные экстра-метки для пода ANIC.	{}

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
<code>controller.readyStatus.enable</code>	Включает конечную точку готовности <code>"/angie-ready"</code> . Конечная точка возвращает код успешного завершения, если Angie загрузил всю конфигурацию после запуска. Этим также настраивается проверка готовности для подов ANIC, которая использует конечную точку готовности.	<code>true</code>
<code>controller.readyStatus.port</code>	HTTP-порт для конечной точки готовности.	8081
<code>controller.readyStatus.initialDelaySeco</code>	Число секунд с запуска пода ANIC до инициирования проверки готовности.	0
<code>controller.enableLatencyMet</code>	Включает сбор метрик задержки для алстримов. Требуется <code>prometheus.create</code> .	<code>false</code>
<code>controller.minReadySeconds</code>	Задаёт минимальное количество секунд, в течение которых вновь созданный под должен прийти в готовое состояние без сбоя какого-либо из контейнеров, чтобы считаться доступным; документацию см. здесь .	0
<code>controller.autoscaling.enabled</code>	Включает <code>HorizontalPodAutoscaling</code> .	<code>false</code>
<code>controller.autoscaling.annotations</code>	Аннотации <code>HorizontalPodAutoscaler</code> для ANIC.	<code>{}</code>
<code>controller.autoscaling.minReplicas</code>	Минимальное число реплик для НРА.	1
<code>controller.autoscaling.maxReplicas</code>	Максимальное число реплик для НРА.	3
<code>controller.autoscaling.targetCPUUtiliza</code>	Целевой процент загрузки ЦП.	50
<code>controller.autoscaling.targetMemoryUtil</code>	Целевой процент использования памяти.	50
<code>controller.podDisruptionBud</code>	Включает <code>PodDisruptionBudget</code> .	<code>false</code>
<code>controller.podDisruptionBud</code>	Аннотации к бюджету сбоев пода ANIC.	<code>{}</code>
<code>controller.podDisruptionBud</code>	Количество подов ANIC, которые должны быть доступны. Взаимоисключающая с <code>maxUnavailable</code> настройка.	0
<code>controller.podDisruptionBud</code>	Количество подов ANIC, которые могут быть недоступны. Взаимоисключающая с <code>minAvailable</code> настройка.	0
<code>controller.strategy</code>	Задаёт стратегию замены старых подов новыми. Документация по стратегии обновления развёртывания и стратегии обновления <code>daemonset</code>	<code>{}</code>
<code>controller.disableIPV6</code>	В явной форме отключает прослушиватели IPV6 для узлов, которые не поддерживают стек IPV6.	<code>false</code>

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	По умолчанию
controller.readOnlyRootFile	Настраивает корневую файловую систему как доступную только для чтения и добавляет тома для временных данных.	false
rbac.create	Настраивает RBAC.	true
prometheus.create	Публикует метрики Angie в формате Prometheus.	true
prometheus.port	Настраивает порт для получения метрик.	9113
prometheus.scheme	Настраивает схему HTTP, используемую для подключений к конечной точке Prometheus.	http
prometheus.secret	Пространство имен или имя TLS-секрета Kubernetes. Если секрет указан, он используется для защиты конечной точки Prometheus с помощью TLS-соединений.	" "

2.2 Установка с помощью манифестов

2.2.1 Предварительные требования

Необходим доступ к Docker-образу в нашем репозитории:

```
anic.docker.angie.software/
```

Для текущей версии доступны следующие образы:

```
anic.docker.angie.software/anic:0.7.1-alpine
anic.docker.angie.software/anic:0.7.1-debian
anic.docker.angie.software/anic:0.7.1-altlinux
```

За доступом обращайтесь на info@wbsrv.ru.

2.2.2 Настройка RBAC

1. Создайте пространство имен и сервисный аккаунт для ANIC:

```
$ kubectl apply -f - <<EOF
apiVersion: v1
kind: Namespace
metadata:
  name: angie-ingress
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: angie-ingress
  namespace: angie-ingress
EOF
```

2. Создайте ClusterRole и ClusterRoleBinding:

Пример

```
$ kubectl apply -f - <<EOF
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: angie-ingress
rules:
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - services
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - secrets
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - configmaps
  verbs:
  - get
  - list
  - watch
  - update
  - create
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - update
- apiGroups:
  - ""
  resources:
  - namespaces
  verbs:
  - get
  - list
```

```

- watch
- apiGroups:
  - ""
  resources:
  - events
  verbs:
  - create
  - patch
  - list
- apiGroups:
  - coordination.k8s.io
  resources:
  - leases
  verbs:
  - get
  - list
  - watch
  - update
  - create
- apiGroups:
  - networking.k8s.io
  resources:
  - ingresses
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - networking.k8s.io
  resources:
  - ingresses/status
  verbs:
  - update
- apiGroups:
  - k8s.angie.software
  resources:
  - virtualservers
  - virtualserverroutes
  - globalconfigurations
  - transportservers
  - policies
  verbs:
  - list
  - watch
  - get
- apiGroups:
  - k8s.angie.software
  resources:
  - virtualservers/status
  - virtualserverroutes/status
  - policies/status
  - transportservers/status
  - dnsendpoints/status
  verbs:
  - update
- apiGroups:
  - networking.k8s.io

```

```

resources:
- ingressclasses
verbs:
- get
- apiGroups:
  - cis.f5.com
resources:
- ingresslinks
verbs:
- list
- watch
- get
- apiGroups:
  - cert-manager.io
resources:
- certificates
verbs:
- list
- watch
- get
- update
- create
- delete
- apiGroups:
  - externaldns.angie.software
resources:
- dnsendpoints
verbs:
- list
- watch
- get
- update
- create
- delete
- apiGroups:
  - externaldns.angie.software
resources:
- dnsendpoints/status
verbs:
- update
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: angie-ingress
subjects:
- kind: ServiceAccount
  name: angie-ingress
  namespace: angie-ingress
roleRef:
  kind: ClusterRole
  name: angie-ingress
  apiGroup: rbac.authorization.k8s.io
EOF

```

2.2.3 Создание ресурсов

- Добавьте TLS-сертификат в настройки:

```
$ kubectl apply -f - <<EOF
apiVersion: v1
kind: Secret
metadata:
  name: default-server-secret
  namespace: angie-ingress
type: kubernetes.io/tls
data:
  tls.crt: Place TLS Certificate here in base64 format
  tls.key: Place TLS Key here in base64 format
EOF
```

- Добавьте ConfigMap с настройками для Angie PRO:

```
$ kubectl apply -f - <<EOF
kind: ConfigMap
apiVersion: v1
metadata:
  name: angie-config
  namespace: angie-ingress
data:
EOF
```

- Создайте IngressClass:

```
$ kubectl apply -f - <<EOF
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: angie
spec:
  controller: angie/ingress-controller
EOF
```

- Создайте пользовательские ресурсы VirtualServer, VirtualServerRoute, TransportServer и Policy:

Пример Virtual Server

```
$ kubectl apply -f - <<EOF
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.11.3
  creationTimestamp: null
  name: virtualservers.k8s.angie.software
spec:
  group: k8s.angie.software
  names:
    kind: VirtualServer
    listKind: VirtualServerList
    plural: virtualservers
    shortNames:
```

```

- vs
  singular: virtualserver
  scope: Namespaced
  versions:
  - additionalPrinterColumns:
    - description: Current state of the VirtualServer. If the resource has a
    ↪valid status, it means it has been validated and accepted by ANIC.
      jsonPath: .status.state
      name: State
      type: string
    - jsonPath: .spec.host
      name: Host
      type: string
    - jsonPath: .status.externalEndpoints[*].ip
      name: IP
      type: string
    - jsonPath: .status.externalEndpoints[*].hostname
      name: ExternalHostname
      priority: 1
      type: string
    - jsonPath: .status.externalEndpoints[*].ports
      name: Ports
      type: string
    - jsonPath: .metadata.creationTimestamp
      name: Age
      type: date
  name: v1
  schema:
    openAPIV3Schema:
      description: VirtualServer defines the VirtualServer resource.
      type: object
      properties:
        apiVersion:
          description: 'APIVersion defines the versioned schema of this
          ↪representation of an object. Servers should convert recognized schemas to the
          ↪latest internal value, and may reject unrecognized values. More info: https://
          ↪git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
          ↪#resources'
          type: string
        kind:
          description: 'Kind is a string value representing the REST
          ↪resource this object represents. Servers may infer this from the endpoint the
          ↪client submits requests to. Cannot be updated. In CamelCase. More info: https://
          ↪git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
          ↪#types-kinds'
          type: string
        metadata:
          type: object
        spec:
          description: VirtualServerSpec is the spec of the VirtualServer
          ↪resource.
          type: object
          properties:
            dns:
              type: string
            externalDNS:
              description: ExternalDNS defines externaldns sub-resource of a

```

```

→virtual server.
    type: object
    properties:
      enable:
        type: boolean
      labels:
        description: Labels stores labels defined for the Endpoint
        type: object
        additionalProperties:
          type: string
      providerSpecific:
        description: ProviderSpecific stores provider specific
→config
    type: array
    items:
      description: ProviderSpecificProperty defines specific
→property for using with ExternalDNS sub-resource.
      type: object
      properties:
        name:
          description: Name of the property
          type: string
        value:
          description: Value of the property
          type: string
      recordTTL:
        description: TTL for the record
        type: integer
        format: int64
      recordType:
        type: string
    host:
      type: string
    http-snippets:
      type: string
    ingressClassName:
      type: string
    policies:
      type: array
      items:
        description: PolicyReference references a policy by name and
→an optional namespace.
      type: object
      properties:
        name:
          type: string
        namespace:
          type: string
    routes:
      type: array
      items:
        description: Route defines a route.
        type: object
        properties:
          action:
            description: Action defines an action.
            type: object

```

```

        properties:
          pass:
            type: string
          proxy:
            description: ActionProxy defines a proxy in an
↳Action.
            type: object
            properties:
              requestHeaders:
                description: ProxyRequestHeaders defines the
↳request headers manipulation in an ActionProxy.
                type: object
                properties:
                  pass:
                    type: boolean
                  set:
                    type: array
                    items:
                      description: Header defines an HTTP Header.
                      type: object
                      properties:
                        name:
                          type: string
                        value:
                          type: string
                  responseHeaders:
                description: ProxyResponseHeaders defines the
↳response headers manipulation in an ActionProxy.
                type: object
                properties:
                  add:
                    type: array
                    items:
                      description: AddHeader defines an HTTP
↳Header with an optional Always field to use with the add_header directive.
                      type: object
                      properties:
                        always:
                          type: boolean
                        name:
                          type: string
                        value:
                          type: string
                  hide:
                    type: array
                    items:
                      type: string
                  ignore:
                    type: array
                    items:
                      type: string
                pass:
                    type: array
                    items:
                      type: string
              rewritePath:
                type: string

```

```

        upstream:
          type: string
      redirect:
        description: ActionRedirect defines a redirect in an
→Action.
        type: object
        properties:
          code:
            type: integer
          url:
            type: string
      return:
        description: ActionReturn defines a return in an
→Action.
        type: object
        properties:
          body:
            type: string
          code:
            type: integer
          type:
            type: string
      dos:
        type: string
      errorPages:
        type: array
        items:
          description: ErrorPage defines an ErrorPage in a Route.
          type: object
          properties:
            codes:
              type: array
              items:
                type: integer
            redirect:
          description: ErrorPageRedirect defines a redirect
→for an ErrorPage.
          type: object
          properties:
            code:
              type: integer
            url:
              type: string
          return:
          description: ErrorPageReturn defines a return for
→an ErrorPage.
          type: object
          properties:
            body:
              type: string
            code:
              type: integer
            headers:
              type: array
              items:
                description: Header defines an HTTP Header.
                type: object

```

```

        properties:
            name:
                type: string
            value:
                type: string
        type:
            type: string
location-snippets:
    type: string
matches:
    type: array
    items:
        description: Match defines a match.
        type: object
        properties:
            action:
                description: Action defines an action.
                type: object
                properties:
                    pass:
                        type: string
                    proxy:
                        description: ActionProxy defines a proxy in an
→Action.
                        type: object
                        properties:
                            requestHeaders:
                                description: ProxyRequestHeaders defines
→the request headers manipulation in an ActionProxy.
                                type: object
                                properties:
                                    pass:
                                        type: boolean
                                    set:
                                        type: array
                                        items:
                                            description: Header defines an HTTP
→Header.
                                            type: object
                                            properties:
                                                name:
                                                    type: string
                                                value:
                                                    type: string
                                            responseHeaders:
                                                description: ProxyResponseHeaders defines
→the response headers manipulation in an ActionProxy.
                                                type: object
                                                properties:
                                                    add:
                                                        type: array
                                                        items:
                                                            description: AddHeader defines an
→HTTP Header with an optional Always field to use with the add_header directive.
                                                            type: object
                                                            properties:
                                                                always:

```

```

        type: boolean
        name:
          type: string
          value:
            type: string
      hide:
        type: array
        items:
          type: string
      ignore:
        type: array
        items:
          type: string
      pass:
        type: array
        items:
          type: string
      rewritePath:
        type: string
      upstream:
        type: string
    redirect:
      description: ActionRedirect defines a redirect
    in an Action.
      type: object
      properties:
        code:
          type: integer
        url:
          type: string
      return:
      description: ActionReturn defines a return in
    an Action.
      type: object
      properties:
        body:
          type: string
        code:
          type: integer
        type:
          type: string
      conditions:
        type: array
        items:
          description: Condition defines a condition in a
    MatchRule.
      type: object
      properties:
        argument:
          type: string
        cookie:
          type: string
        header:
          type: string
        value:
          type: string
        variable:

```

```

        type: string
splits:
  type: array
  items:
    description: Split defines a split.
    type: object
    properties:
      action:
        description: Action defines an action.
        type: object
        properties:
          pass:
            type: string
          proxy:
            description: ActionProxy defines a proxy
→in an Action.

        type: object
        properties:
          requestHeaders:
            description: ProxyRequestHeaders
→defines the request headers manipulation in an ActionProxy.
            type: object
            properties:
              pass:
                type: boolean
              set:
                type: array
                items:
                  description: Header defines an
→HTTP Header.

            type: object
            properties:
              name:
                type: string
              value:
                type: string
            responseHeaders:
              description: ProxyResponseHeaders
→defines the response headers manipulation in an ActionProxy.
            type: object
            properties:
              add:
                type: array
                items:
                  description: AddHeader defines
→an HTTP Header with an optional Always field to use with the add_header
→directive.

            type: object
            properties:
              always:
                type: boolean
              name:
                type: string
              value:
                type: string
            hide:
              type: array

```

```

        items:
            type: string
        ignore:
            type: array
        items:
            type: string
        pass:
            type: array
        items:
            type: string
        rewritePath:
            type: string
        upstream:
            type: string
    redirect:
        description: ActionRedirect defines a
→redirect in an Action.
        type: object
        properties:
            code:
                type: integer
            url:
                type: string
    return:
        description: ActionReturn defines a
→return in an Action.
        type: object
        properties:
            body:
                type: string
            code:
                type: integer
            type:
                type: string
        weight:
            type: integer
    path:
        type: string
    policies:
        type: array
        items:
            description: PolicyReference references a policy by
→name and an optional namespace.
            type: object
            properties:
                name:
                    type: string
                namespace:
                    type: string
    route:
        type: string
    splits:
        type: array
        items:
            description: Split defines a split.
            type: object
            properties:

```

```

    action:
      description: Action defines an action.
      type: object
      properties:
        pass:
          type: string
        proxy:
          description: ActionProxy defines a proxy in an Action.
          type: object
          properties:
            requestHeaders:
              description: ProxyRequestHeaders defines the request headers manipulation in an ActionProxy.
              type: object
              properties:
                pass:
                  type: boolean
                set:
                  type: array
                  items:
                    description: Header defines an HTTP Header.
                    type: object
                    properties:
                      name:
                        type: string
                      value:
                        type: string
            responseHeaders:
              description: ProxyResponseHeaders defines the response headers manipulation in an ActionProxy.
              type: object
              properties:
                add:
                  type: array
                  items:
                    description: AddHeader defines an HTTP Header with an optional Always field to use with the add_header directive.
                    type: object
                    properties:
                      always:
                        type: boolean
                      name:
                        type: string
                      value:
                        type: string
                hide:
                  type: array
                  items:
                    type: string
                ignore:
                  type: array
                  items:
                    type: string
                pass:
                  type: array

```

```

        items:
          type: string
        rewritePath:
          type: string
        upstream:
          type: string
      redirect:
        description: ActionRedirect defines a redirect in
→ in an Action.
        type: object
        properties:
          code:
            type: integer
          url:
            type: string
      return:
        description: ActionReturn defines a return in
→ an Action.
        type: object
        properties:
          body:
            type: string
          code:
            type: integer
          type:
            type: string
        weight:
          type: integer
      server-snippets:
        type: string
      tls:
        description: TLS defines TLS configuration for a VirtualServer.
        type: object
        properties:
          cert-manager:
            description: CertManager defines a cert manager config for
→ a TLS.
            type: object
            properties:
              cluster-issuer:
                type: string
              common-name:
                type: string
              duration:
                type: string
              issuer:
                type: string
              issuer-group:
                type: string
              issuer-kind:
                type: string
              renew-before:
                type: string
              usages:
                type: string
          redirect:
            description: TLSRedirect defines a redirect for a TLS.

```

```

        type: object
        properties:
          basedOn:
            type: string
          code:
            type: integer
          enable:
            type: boolean
        secret:
          type: string
    upstreams:
      type: array
      items:
        description: Upstream defines an upstream.
        type: object
        properties:
          buffer-size:
            type: string
          buffering:
            type: boolean
          buffers:
            description: UpstreamBuffers defines Buffer_
    → Configuration for an Upstream.
            type: object
            properties:
              number:
                type: integer
              size:
                type: string
            client-max-body-size:
              type: string
            connect-timeout:
              type: string
            fail-timeout:
              type: string
            healthCheck:
              description: HealthCheck defines the parameters for_
    → active Upstream HealthChecks.
            type: object
            properties:
              connect-timeout:
                type: string
              enable:
                type: boolean
              fails:
                type: integer
              grpcService:
                type: string
              grpcStatus:
                type: integer
              headers:
                type: array
                items:
                  description: Header defines an HTTP Header.
                  type: object
                  properties:
                    name:

```

```

        type: string
        value:
          type: string
      interval:
        type: string
      jitter:
        type: string
      keepalive-time:
        type: string
      mandatory:
        type: boolean
      passes:
        type: integer
      path:
        type: string
      persistent:
        type: boolean
      port:
        type: integer
      read-timeout:
        type: string
      send-timeout:
        type: string
      statusMatch:
        type: string
      tls:
        description: UpstreamTLS defines a TLS configuration
    }
  }
}
→for an Upstream.

      type: object
      properties:
        enable:
          type: boolean
    keepalive:
      type: integer
    lb-method:
      type: string
    max-conns:
      type: integer
    max-fails:
      type: integer
    name:
      type: string
    next-upstream:
      type: string
    next-upstream-timeout:
      type: string
    next-upstream-tries:
      type: integer
    ntlm:
      type: boolean
    port:
      type: integer
    queue:
      description: UpstreamQueue defines Queue Configuration
    }
  }
}
→for an Upstream.

      type: object
      properties:

```

```

        size:
            type: integer
        timeout:
            type: string
    read-timeout:
        type: string
    send-timeout:
        type: string
    service:
        type: string
    sessionCookie:
        description: SessionCookie defines the parameters for
↳session persistence.
        type: object
        properties:
            domain:
                type: string
            enable:
                type: boolean
            expires:
                type: string
            httpOnly:
                type: boolean
            name:
                type: string
            path:
                type: string
            secure:
                type: boolean
    slow-start:
        type: string
    subselector:
        type: object
        additionalProperties:
            type: string
    tls:
        description: UpstreamTLS defines a TLS configuration for
↳an Upstream.
        type: object
        properties:
            enable:
                type: boolean
        type:
            type: string
        use-cluster-ip:
            type: boolean
    status:
        description: VirtualServerStatus defines the status for the
↳VirtualServer resource.
        type: object
        properties:
            externalEndpoints:
                type: array
                items:
                    description: ExternalEndpoint defines the IP/ Hostname and
↳ports used to connect to this resource.
                    type: object

```

```

        properties:
          hostname:
            type: string
          ip:
            type: string
          ports:
            type: string
        message:
          type: string
        reason:
          type: string
        state:
          type: string
    served: true
    storage: true
    subresources:
      status: {}
EOF

```

Пример VirtualServerRoute

```

$ kubectl apply -f - <<EOF
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.11.3
  creationTimestamp: null
  name: virtualserverroutes.k8s.angie.software
spec:
  group: k8s.angie.software
  names:
    kind: VirtualServerRoute
    listKind: VirtualServerRouteList
    plural: virtualserverroutes
    shortNames:
      - vsr
    singular: virtualserverroute
  scope: Namespaced
  versions:
    - additionalPrinterColumns:
      - description: Current state of the VirtualServerRoute. If the resource
↳has a valid status, it means it has been validated and accepted by ANIC.
        jsonPath: .status.state
        name: State
        type: string
      - jsonPath: .spec.host
        name: Host
        type: string
      - jsonPath: .status.externalEndpoints[*].ip
        name: IP
        type: string
      - jsonPath: .status.externalEndpoints[*].hostname
        name: ExternalHostname
        priority: 1
        type: string

```

```

- jsonPath: .status.externalEndpoints[*].ports
  name: Ports
  type: string
- jsonPath: .metadata.creationTimestamp
  name: Age
  type: date
name: v1
schema:
  openAPIV3Schema:
    description: VirtualServerRoute defines the VirtualServerRoute
→resource.
  type: object
  properties:
    apiVersion:
      description: 'APIVersion defines the versioned schema of this
→representation of an object. Servers should convert recognized schemas to the
→latest internal value, and may reject unrecognized values. More info: https://
→git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
→#resources'
      type: string
    kind:
      description: 'Kind is a string value representing the REST
→resource this object represents. Servers may infer this from the endpoint the
→client submits requests to. Cannot be updated. In CamelCase. More info: https://
→git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
→#types-kinds'
      type: string
    metadata:
      type: object
    spec:
      description: VirtualServerRouteSpec is the spec of the
→VirtualServerRoute resource.
      type: object
      properties:
        host:
          type: string
        ingressClassName:
          type: string
        subroutes:
          type: array
          items:
            description: Route defines a route.
            type: object
            properties:
              action:
                description: Action defines an action.
                type: object
                properties:
                  pass:
                    type: string
                  proxy:
                    description: ActionProxy defines a proxy in an
→Action.
                    type: object
                    properties:
                      requestHeaders:
                        description: ProxyRequestHeaders defines the

```

```

↳request headers manipulation in an ActionProxy.
    type: object
    properties:
      pass:
        type: boolean
      set:
        type: array
        items:
          description: Header defines an HTTP Header.
          type: object
          properties:
            name:
              type: string
            value:
              type: string
      responseHeaders:
        description: ProxyResponseHeaders defines the
↳response headers manipulation in an ActionProxy.
    type: object
    properties:
      add:
        type: array
        items:
          description: AddHeader defines an HTTP
↳Header with an optional Always field to use with the add_header directive.
    type: object
    properties:
      always:
        type: boolean
      name:
        type: string
      value:
        type: string
      hide:
        type: array
        items:
          type: string
      ignore:
        type: array
        items:
          type: string
      pass:
        type: array
        items:
          type: string
      rewritePath:
        type: string
      upstream:
        type: string
      redirect:
        description: ActionRedirect defines a redirect in an
↳Action.
    type: object
    properties:
      code:
        type: integer
      url:

```

```

        type: string
    return:
        description: ActionReturn defines a return in an
→Action.

        type: object
        properties:
            body:
                type: string
            code:
                type: integer
            type:
                type: string
    dos:
        type: string
    errorPages:
        type: array
        items:
            description: ErrorPage defines an ErrorPage in a Route.
            type: object
            properties:
                codes:
                    type: array
                    items:
                        type: integer
            redirect:
            description: ErrorPageRedirect defines a redirect
→for an ErrorPage.

            type: object
            properties:
                code:
                    type: integer
                url:
                    type: string
            return:
            description: ErrorPageReturn defines a return for
→an ErrorPage.

            type: object
            properties:
                body:
                    type: string
                code:
                    type: integer
                headers:
                    type: array
                    items:
                        description: Header defines an HTTP Header.
                        type: object
                        properties:
                            name:
                                type: string
                            value:
                                type: string
                type:
                    type: string
    location-snippets:
        type: string
    matches:

```

```

    type: array
    items:
      description: Match defines a match.
      type: object
      properties:
        action:
          description: Action defines an action.
          type: object
          properties:
            pass:
              type: string
            proxy:
              description: ActionProxy defines a proxy in an
→Action.
              type: object
              properties:
                requestHeaders:
          description: ProxyRequestHeaders defines
→the request headers manipulation in an ActionProxy.
          type: object
          properties:
            pass:
              type: boolean
            set:
              type: array
              items:
                description: Header defines an HTTP
→Header.
                type: object
                properties:
                  name:
                    type: string
                  value:
                    type: string
                responseHeaders:
          description: ProxyResponseHeaders defines
→the response headers manipulation in an ActionProxy.
          type: object
          properties:
            add:
              type: array
              items:
                description: AddHeader defines an
→HTTP Header with an optional Always field to use with the add_header directive.
                type: object
                properties:
                  always:
                    type: boolean
                  name:
                    type: string
                  value:
                    type: string
                hide:
              type: array
              items:
                type: string
            ignore:

```

```

        type: array
        items:
            type: string
    pass:
        type: array
        items:
            type: string
    rewritePath:
        type: string
    upstream:
        type: string
    redirect:
        description: ActionRedirect defines a redirect
→ in an Action.

    type: object
    properties:
        code:
            type: integer
        url:
            type: string
    return:
        description: ActionReturn defines a return in
→ an Action.

    type: object
    properties:
        body:
            type: string
        code:
            type: integer
        type:
            type: string
    conditions:
        type: array
        items:
            description: Condition defines a condition in a
→ MatchRule.

    type: object
    properties:
        argument:
            type: string
        cookie:
            type: string
        header:
            type: string
        value:
            type: string
        variable:
            type: string
    splits:
        type: array
        items:
            description: Split defines a split.
            type: object
            properties:
                action:
                    description: Action defines an action.
                    type: object

```

```

        properties:
          pass:
            type: string
          proxy:
            description: ActionProxy defines a proxy
→in an Action.

            type: object
            properties:
              requestHeaders:
                description: ProxyRequestHeaders
→defines the request headers manipulation in an ActionProxy.
                type: object
                properties:
                  pass:
                    type: boolean
                  set:
                    type: array
                    items:
                      description: Header defines an
→HTTP Header.

                      type: object
                      properties:
                        name:
                          type: string
                        value:
                          type: string
                      responseHeaders:
                        description: ProxyResponseHeaders
→defines the response headers manipulation in an ActionProxy.
                        type: object
                        properties:
                          add:
                            type: array
                            items:
                              description: AddHeader defines
→an HTTP Header with an optional Always field to use with the add_header
→directive.

                              type: object
                              properties:
                                always:
                                  type: boolean
                                name:
                                  type: string
                                value:
                                  type: string
                              hide:
                                type: array
                                items:
                                  type: string
                              ignore:
                                type: array
                                items:
                                  type: string
                              pass:
                                type: array
                                items:
                                  type: string

```

```

        rewritePath:
            type: string
        upstream:
            type: string
    redirect:
        description: ActionRedirect defines a
→redirect in an Action.
        type: object
        properties:
            code:
                type: integer
            url:
                type: string
    return:
        description: ActionReturn defines a
→return in an Action.
        type: object
        properties:
            body:
                type: string
            code:
                type: integer
            type:
                type: string
        weight:
            type: integer
    path:
        type: string
    policies:
        type: array
        items:
            description: PolicyReference references a policy by
→name and an optional namespace.
            type: object
            properties:
                name:
                    type: string
                namespace:
                    type: string
    route:
        type: string
    splits:
        type: array
        items:
            description: Split defines a split.
            type: object
            properties:
                action:
                    description: Action defines an action.
                    type: object
                    properties:
                        pass:
                            type: string
                        proxy:
                            description: ActionProxy defines a proxy in an
→Action.
                            type: object
                    type: object

```

```

        properties:
          requestHeaders:
            description: ProxyRequestHeaders defines
↳the request headers manipulation in an ActionProxy.
            type: object
            properties:
              pass:
                type: boolean
              set:
                type: array
                items:
                  description: Header defines an HTTP
↳Header.
                  type: object
                  properties:
                    name:
                      type: string
                    value:
                      type: string
              responseHeaders:
            description: ProxyResponseHeaders defines
↳the response headers manipulation in an ActionProxy.
            type: object
            properties:
              add:
                type: array
                items:
                  description: AddHeader defines an
↳HTTP Header with an optional Always field to use with the add_header directive.
                  type: object
                  properties:
                    always:
                      type: boolean
                    name:
                      type: string
                    value:
                      type: string
              hide:
                type: array
                items:
                  type: string
              ignore:
                type: array
                items:
                  type: string
              pass:
                type: array
                items:
                  type: string
              rewritePath:
                type: string
              upstream:
                type: string
            redirect:
            description: ActionRedirect defines a redirect
↳in an Action.
            type: object

```

```

        properties:
          code:
            type: integer
          url:
            type: string
        return:
          description: ActionReturn defines a return in
→an Action.
          type: object
          properties:
            body:
              type: string
            code:
              type: integer
            type:
              type: string
          weight:
            type: integer
    upstreams:
      type: array
      items:
        description: Upstream defines an upstream.
        type: object
        properties:
          buffer-size:
            type: string
          buffering:
            type: boolean
          buffers:
            description: UpstreamBuffers defines Buffer
→Configuration for an Upstream.
            type: object
            properties:
              number:
                type: integer
              size:
                type: string
            client-max-body-size:
              type: string
            connect-timeout:
              type: string
            fail-timeout:
              type: string
            healthCheck:
            description: HealthCheck defines the parameters for
→active Upstream HealthChecks.
              type: object
              properties:
                connect-timeout:
                  type: string
                enable:
                  type: boolean
                fails:
                  type: integer
                grpcService:
                  type: string
                grpcStatus:

```

```

        type: integer
headers:
  type: array
  items:
    description: Header defines an HTTP Header.
    type: object
    properties:
      name:
        type: string
      value:
        type: string
interval:
  type: string
jitter:
  type: string
keepalive-time:
  type: string
mandatory:
  type: boolean
passes:
  type: integer
path:
  type: string
persistent:
  type: boolean
port:
  type: integer
read-timeout:
  type: string
send-timeout:
  type: string
statusMatch:
  type: string
tls:
  description: UpstreamTLS defines a TLS configuration.
  type: object
  properties:
    enable:
      type: boolean
keepalive:
  type: integer
lb-method:
  type: string
max-conns:
  type: integer
max-fails:
  type: integer
name:
  type: string
next-upstream:
  type: string
next-upstream-timeout:
  type: string
next-upstream-tries:
  type: integer
ntlm:

```

→for an Upstream.

```

        type: boolean
    port:
        type: integer
    queue:
        description: UpstreamQueue defines Queue Configuration
→for an Upstream.
        type: object
        properties:
            size:
                type: integer
            timeout:
                type: string
    read-timeout:
        type: string
    send-timeout:
        type: string
    service:
        type: string
    sessionCookie:
        description: SessionCookie defines the parameters for
→session persistence.
        type: object
        properties:
            domain:
                type: string
            enable:
                type: boolean
            expires:
                type: string
            httpOnly:
                type: boolean
            name:
                type: string
            path:
                type: string
            secure:
                type: boolean
    slow-start:
        type: string
    subselector:
        type: object
        additionalProperties:
            type: string
    tls:
        description: UpstreamTLS defines a TLS configuration for
→an Upstream.
        type: object
        properties:
            enable:
                type: boolean
    type:
        type: string
    use-cluster-ip:
        type: boolean
    status:
        description: VirtualServerRouteStatus defines the status for the
→VirtualServerRoute resource.

```

```

        type: object
        properties:
          externalEndpoints:
            type: array
            items:
              description: ExternalEndpoint defines the IP/ Hostname and
↳ ports used to connect to this resource.
              type: object
              properties:
                hostname:
                  type: string
                ip:
                  type: string
                ports:
                  type: string
              message:
                type: string
              reason:
                type: string
              referencedBy:
                type: string
              state:
                type: string
        served: true
        storage: true
        subresources:
          status: {}
EOF

```

Пример TransportServer

```

$ kubectl apply -f - <<EOF
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.11.3
  creationTimestamp: null
  name: transportservers.k8s.angie.software
spec:
  group: k8s.angie.software
  names:
    kind: TransportServer
    listKind: TransportServerList
    plural: transportservers
    shortNames:
      - ts
    singular: transportserver
  scope: Namespaced
  versions:
    - additionalPrinterColumns:
      - description: Current state of the TransportServer. If the resource has
↳ a valid status, it means it has been validated and accepted by ANIC.
        jsonPath: .status.state
        name: State
        type: string

```

```

- jsonPath: .status.reason
  name: Reason
  type: string
- jsonPath: .metadata.creationTimestamp
  name: Age
  type: date
name: v1alpha1
schema:
  openAPIV3Schema:
    description: TransportServer defines the TransportServer resource.
    type: object
    properties:
      apiVersion:
        description: 'APIVersion defines the versioned schema of this
↳representation of an object. Servers should convert recognized schemas to the
↳latest internal value, and may reject unrecognized values. More info: https://
↳git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
↳#resources'
        type: string
      kind:
        description: 'Kind is a string value representing the REST
↳resource this object represents. Servers may infer this from the endpoint the
↳client submits requests to. Cannot be updated. In CamelCase. More info: https://
↳git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
↳#types-kinds'
        type: string
      metadata:
        type: object
      spec:
        description: TransportServerSpec is the spec of the
↳TransportServer resource.
        type: object
        properties:
          action:
            description: Action defines an action.
            type: object
            properties:
              pass:
                type: string
          host:
            type: string
          ingressClassName:
            type: string
          listener:
            description: TransportServerListener defines a listener for a
↳TransportServer.
            type: object
            properties:
              name:
                type: string
              protocol:
                type: string
          serverSnippets:
            type: string
          sessionParameters:
            description: SessionParameters defines session parameters.
            type: object

```

```

        properties:
            timeout:
                type: string
        streamSnippets:
            type: string
        tls:
            description: TLS defines TLS configuration for a
↳TransportServer.
            type: object
            properties:
                secret:
                    type: string
            upstreamParameters:
            description: UpstreamParameters defines parameters for an
↳upstream.
            type: object
            properties:
                connectTimeout:
                    type: string
                nextUpstream:
                    type: boolean
                nextUpstreamTimeout:
                    type: string
                nextUpstreamTries:
                    type: integer
                udpRequests:
                    type: integer
                udpResponses:
                    type: integer
            upstreams:
            type: array
            items:
            description: Upstream defines an upstream.
            type: object
            properties:
                failTimeout:
                    type: string
                healthCheck:
            description: HealthCheck defines the parameters for
↳active Upstream HealthChecks.
            type: object
            properties:
                enable:
                    type: boolean
                fails:
                    type: integer
                interval:
                    type: string
                jitter:
                    type: string
                match:
            description: Match defines the parameters of a
↳custom health check.
            type: object
            properties:
                expect:
                    type: string

```

```

        send:
          type: string
        passes:
          type: integer
        port:
          type: integer
        timeout:
          type: string
        loadBalancingMethod:
          type: string
        maxConns:
          type: integer
        maxFails:
          type: integer
        name:
          type: string
        port:
          type: integer
        service:
          type: string
      status:
        description: TransportServerStatus defines the status for the
↳TransportServer resource.
        type: object
        properties:
          message:
            type: string
          reason:
            type: string
          state:
            type: string
        served: true
        storage: true
        subresources:
          status: {}
EOF

```

Пример Policy

```

$ kubectl apply -f - <<EOF
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.11.3
  creationTimestamp: null
  name: policies.k8s.angie.software
spec:
  group: k8s.angie.software
  names:
    kind: Policy
    listKind: PolicyList
    plural: policies
    shortNames:
      - pol
    singular: policy

```

```

scope: Namespaced
versions:
  - additionalPrinterColumns:
    - description: Current state of the Policy. If the resource has a valid
↳status, it means it has been validated and accepted by ANIC.
      jsonPath: .status.state
      name: State
      type: string
    - jsonPath: .metadata.creationTimestamp
      name: Age
      type: date
  name: v1
  schema:
    openAPIV3Schema:
      description: Policy defines a Policy for VirtualServer and
↳VirtualServerRoute resources.
      type: object
      properties:
        apiVersion:
          description: 'APIVersion defines the versioned schema of this
↳representation of an object. Servers should convert recognized schemas to the
↳latest internal value, and may reject unrecognized values. More info: https://
↳git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
↳#resources'
          type: string
        kind:
          description: 'Kind is a string value representing the REST
↳resource this object represents. Servers may infer this from the endpoint the
↳client submits requests to. Cannot be updated. In CamelCase. More info: https://
↳/git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
↳#types-kinds'
          type: string
        metadata:
          type: object
        spec:
          description: PolicySpec is the spec of the Policy resource. The
↳spec includes multiple fields, where each field represents a different policy.
↳Only one policy (field) is allowed.
          type: object
          properties:
            accessControl:
              description: AccessControl defines an access policy based on
↳the source IP of a request.
              type: object
              properties:
                allow:
                  type: array
                  items:
                    type: string
                deny:
                  type: array
                  items:
                    type: string
            basicAuth:
              description: 'BasicAuth holds HTTP Basic authentication
↳configuration policy status: preview'
          type: object

```

```

properties:
  realm:
    type: string
  secret:
    type: string
egressMTLS:
  description: EgressMTLS defines an Egress MTLS policy.
  type: object
  properties:
    ciphers:
      type: string
    protocols:
      type: string
    serverName:
      type: boolean
    sessionReuse:
      type: boolean
    sslName:
      type: string
    tlsSecret:
      type: string
    trustedCertSecret:
      type: string
    verifyDepth:
      type: integer
    verifyServer:
      type: boolean
ingressClassName:
  type: string
ingressMTLS:
  description: IngressMTLS defines an Ingress MTLS policy.
  type: object
  properties:
    clientCertSecret:
      type: string
    crlFileName:
      type: string
    verifyClient:
      type: string
    verifyDepth:
      type: integer
jwt:
  description: JWT holds JWT authentication configuration.
  realm: string
  secret: string
  token: string
oidc:
  description: OIDC defines an Open ID Connect policy.
  type: object
  properties:
    clientID:
      type: string
    clientSecret:
      type: string
    authEndpoint:
      type: string
    jwksURI:

```

```

        type: string
    tokenEndpoint:
        type: string
    scope:
        type: string
    accessTokenEnable:
        type: boolean
    rateLimit:
        description: RateLimit defines a rate limit policy.
        type: object
        properties:
            burst:
                type: integer
            delay:
                type: integer
            dryRun:
                type: boolean
            key:
                type: string
            logLevel:
                type: string
            noDelay:
                type: boolean
            rate:
                type: string
            rejectCode:
                type: integer
            zoneSize:
                type: string
    status:
        description: PolicyStatus is the status of the policy resource
        type: object
        properties:
            message:
                type: string
            reason:
                type: string
            state:
                type: string
    served: true
    storage: true
    subresources:
        status: {}
- name: v1alpha1
  schema:
    openAPIV3Schema:
        description: Policy defines a Policy for VirtualServer and
        ↳VirtualServerRoute resources.
        type: object
        properties:
            apiVersion:
                description: 'APIVersion defines the versioned schema of this
                ↳representation of an object. Servers should convert recognized schemas to the
                ↳latest internal value, and may reject unrecognized values. More info: https://
                ↳git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
                ↳#resources'
                type: string

```

```

    kind:
      description: 'Kind is a string value representing the REST
→resource this object represents. Servers may infer this from the endpoint the
→client submits requests to. Cannot be updated. In CamelCase. More info: https://
→/git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
→#types-kinds'
      type: string
      metadata:
        type: object
      spec:
        description: PolicySpec is the spec of the Policy resource. The
→spec includes multiple fields, where each field represents a different policy.
→Only one policy (field) is allowed.
        type: object
        properties:
          accessControl:
            description: AccessControl defines an access policy based on
→the source IP of a request.
            type: object
            properties:
              allow:
                type: array
                items:
                  type: string
              deny:
                type: array
                items:
                  type: string
          egressMTLS:
            description: EgressMTLS defines an Egress MTLs policy.
            type: object
            properties:
              ciphers:
                type: string
              protocols:
                type: string
              serverName:
                type: boolean
              sessionReuse:
                type: boolean
              sslName:
                type: string
              tlsSecret:
                type: string
              trustedCertSecret:
                type: string
              verifyDepth:
                type: integer
              verifyServer:
                type: boolean
          ingressMTLS:
            description: IngressMTLS defines an Ingress MTLs policy.
            type: object
            properties:
              clientCertSecret:
                type: string
              verifyClient:

```

```

        type: string
    verifyDepth:
        type: integer
    jwt:
        description: JWT holds JWT authentication configuration.
        realm: string
        secret: string
        token: string
    rateLimit:
        description: RateLimit defines a rate limit policy.
        type: object
        properties:
            burst:
                type: integer
            delay:
                type: integer
            dryRun:
                type: boolean
            key:
                type: string
            logLevel:
                type: string
            noDelay:
                type: boolean
            rate:
                type: string
            rejectCode:
                type: integer
            zoneSize:
                type: string
    served: true
    storage: false
EOF

```

- Если нужно использовать балансировщик нагрузки для TCP- и UDP-соединений, добавьте GlobalConfiguration:

Пример

```

$ kubectl apply -f - <<EOF
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.11.3
  creationTimestamp: null
  name: globalconfigurations.k8s.angie.software
spec:
  group: k8s.angie.software
  names:
    kind: GlobalConfiguration
    listKind: GlobalConfigurationList
    plural: globalconfigurations
    shortNames:
      - gc
    singular: globalconfiguration
EOF

```

```

scope: Namespaced
versions:
- name: v1alpha1
  schema:
    openAPIV3Schema:
      description: GlobalConfiguration defines the GlobalConfiguration.
→resource.
      type: object
      properties:
        apiVersion:
          description: 'APIVersion defines the versioned schema of this
→representation of an object. Servers should convert recognized schemas to the
→latest internal value, and may reject unrecognized values. More info: https://
→git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
→#resources'
          type: string
        kind:
          description: 'Kind is a string value representing the REST
→resource this object represents. Servers may infer this from the endpoint the
→client submits requests to. Cannot be updated. In CamelCase. More info: https://
→git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md
→#types-kinds'
          type: string
        metadata:
          type: object
        spec:
          description: GlobalConfigurationSpec is the spec of the
→GlobalConfiguration resource.
          type: object
          properties:
            listeners:
              type: array
              items:
                description: Listener defines a listener.
                type: object
                properties:
                  name:
                    type: string
                  port:
                    type: integer
                  protocol:
                    type: string

      served: true
      storage: true
EOF

```

2.2.4 Развертывание ANIC

8. Поддерживаются два варианта использования ANIC:

- **Deployment:** используйте этот тип развертывания, если планируете динамически изменять количество реплик ANIC.

Пример Deployment

```
$ kubectl apply -f - <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: angie-ingress
  namespace: angie-ingress
spec:
  replicas: 1
  selector:
    matchLabels:
      app: angie-ingress
  template:
    metadata:
      labels:
        app: angie-ingress
        app.kubernetes.io/name: angie-ingress
    #annotations:
    #prometheus.io/scrape: "true"
    #prometheus.io/port: "9113"
    #prometheus.io/scheme: http
    spec:
      serviceAccountName: angie-ingress
      automountServiceAccountToken: true
      securityContext:
        seccompProfile:
          type: RuntimeDefault
#      fsGroup: 101 #angie
      sysctls:
        - name: "net.ipv4.ip_unprivileged_port_start"
          value: "0"
#      volumes:
#        - name: angie-etc
#          emptyDir: {}
#        - name: angie-cache
#          emptyDir: {}
#        - name: angie-lib
#          emptyDir: {}
#        - name: angie-log
#          emptyDir: {}
    containers:
      - image: docker.angie.software/angie-ingress:latest
        imagePullPolicy: IfNotPresent
        name: angie-ingress
        ports:
          - name: http
            containerPort: 80
          - name: https
            containerPort: 443
```

```

- name: readiness-port
  containerPort: 8081
- name: prometheus
  containerPort: 9113
readinessProbe:
  httpGet:
    path: /angie-ready
    port: readiness-port
    periodSeconds: 1
resources:
  requests:
    cpu: "100m"
    memory: "128Mi"
#limits
# cpu: "1"
# memory: "1Gi"
securityContext:
  allowPrivilegeEscalation: false
  runAsUser: 101 #angie
  runAsNonRoot: true
  capabilities:
    drop:
      - ALL
#
# volumeMounts:
# - mountPath: /etc/angie
#   name: angie-etc
# - mountPath: /var/cache/angie
#   name: angie-cache
# - mountPath: /var/lib/angie
#   name: angie-lib
# - mountPath: /var/log/angie
#   name: angie-log
env:
- name: POD_NAMESPACE
  valueFrom:
    fieldRef:
      fieldPath: metadata.namespace
- name: POD_NAME
  valueFrom:
    fieldRef:
      fieldPath: metadata.name
args:
  - -angie-configmaps=$(POD_NAMESPACE)/angie-config
  #- -default-server-tls-secret=$(POD_NAMESPACE)/default-server-secret
  #- -include-year
  #- -enable-cert-manager
  #- -enable-external-dns
  #- -v=3 # Enables extensive logging. Useful for troubleshooting.
  #- -report-ingress-status
  #- -external-service=angie-ingress
  #- -enable-prometheus-metrics
  #- -global-configuration=$(POD_NAMESPACE)/angie-configuration
EOF

```

- **DaemonSet**: используйте этот тип, если планируете разворачивать ANIC на каждом узле кластера или подмножестве узлов.

Пример DaemonSet

```
$ kubectl apply -f - <<EOF
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: angie-ingress
  namespace: angie-ingress
spec:
  selector:
    matchLabels:
      app: angie-ingress
  template:
    metadata:
      labels:
        app: angie-ingress
        app.kubernetes.io/name: angie-ingress
    spec:
      serviceAccountName: angie-ingress
      automountServiceAccountToken: true
      securityContext:
        seccompProfile:
          type: RuntimeDefault
      sysctls:
        - name: "net.ipv4.ip_unprivileged_port_start"
          value: "0"
      containers:
        - image: docker.angie.software/angie-ingress:latest
          imagePullPolicy: IfNotPresent
          name: angie-ingress
          ports:
            - name: http
              containerPort: 80
              hostPort: 80
            - name: https
              containerPort: 443
              hostPort: 443
            - name: readiness-port
              containerPort: 8081
            - name: prometheus
              containerPort: 9113
          readinessProbe:
            httpGet:
              path: /angie-ready
              port: readiness-port
            periodSeconds: 1
          resources:
            requests:
              cpu: "100m"
              memory: "128Mi"
          env:
            - name: POD_NAMESPACE
              valueFrom:
                fieldRef:
                  fieldPath: metadata.namespace
            - name: POD_NAME
              valueFrom:
                fieldRef:
```

```

        fieldPath: metadata.name
    args:
      - --angie-configmaps=$(POD_NAMESPACE)/angie-config
      #- --default-server-tls-secret=$(POD_NAMESPACE)/default-server-secret
      #- --include-year
      #- -v=3 # Enables extensive logging. Useful for troubleshooting.
      #- --report-ingress-status
      #- --external-service=angie-ingress
      #- --enable-prometheus-metrics
      #- --global-configuration=$(POD_NAMESPACE)/angie-configuration
EOF

```

2.3 Подключение лицензии

Перед *установкой ANIC* необходимо подключить лицензию с помощью файла `licence.pem`. Файл `licence.pem` содержит лицензионный ключ и подтверждает право на использование ANIC. Получить его можно, обратившись в техническую поддержку или скачав в личном кабинете.

Чтобы подключить лицензию, выполните следующие шаги:

1. Создайте секрет с лицензионным ключом:

```

apiVersion: v1
kind: Secret
metadata:
  name: angie-pro-secret
  namespace: default
data:
  angiePro.license: <licence.pem>

```

Здесь `<licence.pem>` - лицензионный ключ, закодированный в Base64, из файла `licence.pem`.

Пример конвертации в Linux:

```
$ cat licence.pem | base64 -w0
```

Пример вывода:

```

LS0tLS1CRUdJTiBMSUNFT1NFLS0tLSOKZDI5eWEyVn1.....
→1FTkQgQ0VSVE1GSUNBVEUtLS0tLQo=

```

2. Добавьте секрет в файл `values.yaml`.

Для этого найдите в `values.yaml` параметр `angieProLicense` и добавьте для него значение `<пространство имен/имя секрета>`, например:

```
angieProLicense: "default/angie-pro-secret"
```

Далее продолжите установку ANIC *по инструкции*.

2.4 Установка и настройка для Яндекс.Облака

В этом разделе описана установка и настройка ANIC в качестве Ingress для Яндекс.Облака.

2.4.1 Настройка локальной рабочей среды для работы с Kubernetes и Яндекс.Облаком

Пример для Ubuntu 24/04.

1. Скачайте и установите CLI (YC) для управления ресурсами в Яндекс.Облаке:

```
$ curl -sSL https://storage.yandexcloud.net/yandexcloud-yc/install.sh | bash
$ yc init
```

Запустится процесс настройки CLI, где потребуется ввести токен для доступа в Яндекс.Облако. Токен можно получить по ссылке https://oauth.yandex.ru/verification_code.

2. Добавьте параметры подключения к Kubernetes-кластеру (например, otus) в файл `~/.kube/config` и подключитесь к кластеру:

```
$ yc managed-kubernetes cluster get-credentials otus --external
```

3. Проверьте конфигурацию кластера, ноды и доступные сервисы:

```
$ kubectl config view
$ kubectl get nodes
$ kubectl get svc
```

2.4.2 Получение доступа к образу ANIC

1. Создайте Docker-секрет.

Выполните авторизацию в Docker Registry, чтобы получить доступ к образу:

```
$ docker login -u=<login> -p=<password> anic.docker.angie.software
```

После успешной авторизации информация о доступе будет сохранена в файле `~/.docker/config.json`.

2. Создайте секрет для Kubernetes, используя файл Docker-конфигурации:

```
$ kubectl create secret generic regcred \
  --from-file=.dockerconfigjson=$HOME/.docker/config.json \
  --type=kubernetes.io/dockerconfigjson
```

Секрет `regcred` будет использоваться для доступа к приватным образам в Kubernetes.

2.4.3 Получение и настройка Helm-чартов

1. Склонируйте Helm-чарты из репозитория:

```
$ git clone https://git.angie.software/web-server/anic-helm-charts.git
$ cd anic-helm-charts/anic/
```

2. Отредактируйте `values.yaml`. Укажите имя созданного секрета для доступа к образам и включите поддержку менеджера сертификатов:

```
...
imagePullSecretName: "regcred"
...
enableCertManager: true
...
```

3. Соберите и установите чарт:

```
$ helm install anic .
```

4. Обновите чарт:

```
$ helm upgrade anic .
```

2.4.4 Запуск менеджера сертификатов

1. Установите `cert-manager` для управления сертификатами. Работа тестировалась на примере версии 1.12.1:

```
$ kubectl apply -f https://github.com/cert-manager/cert-manager/releases/
→download/v1.12.1/cert-manager.yaml
```

2. Убедитесь, что три пода `cert-manager` находятся в состоянии `running` с готовностью 1/1:

```
$ kubectl get pods -n cert-manager --watch
```

3. Создайте объект `ClusterIssuer` для выпуска сертификатов:

http01-clusterissuer.yaml

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: http01-clusterissuer
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: test@test.ru
    privateKeySecretRef:
      name: http01-clusterissuer-secret
    solvers:
      - http01:
          ingress:
            class: angie
```

В этом примере `cert-manager` настроен для автоматической выдачи сертификатов от Let's Encrypt с помощью HTTP-01 проверки.

4. Примените манифест для объекта `ClusterIssuer`:

```
$ kubectl apply -f http01-clusterissuer.yaml
```

После выполнения всех шагов, у вас будет настроен доступ к образу ANIC, чарт развернут в Kubernetes, а `cert-manager` будет управлять сертификатами.

2.4.5 Развертывание ANIC и тестового приложения в Kubernetes

1. Настройте домен и поиск внешнего IP-адреса. Создайте Ingress-ресурс, сервис и деплоймент для тестового приложения.

Замените доменное имя `otus-kube.mtdlb.ru` на ваш действующий домен. Внешний IP-адрес кластера можно найти в Яндекс.Облаке в разделе **Managed Service for Kubernetes** → Кластеры → `otus` → Сеть. Этот IP-адрес будет использоваться для привязки домена.

app-angie.yaml

```

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    cert-manager.io/cluster-issuer: "http01-clusterissuer"
    acme.cert-manager.io/http01-edit-in-place: "true"
spec:
  ingressClassName: angie
  tls:
  - hosts:
    - otus-kube.mtdlb.ru
    secretName: domain-name-secret
  rules:
  - host: otus-kube.mtdlb.ru
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: app
            port:
              number: 80
---
apiVersion: v1
kind: Service
metadata:
  name: app
spec:
  selector:
    app: app
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
  labels:
    app: app
spec:
  replicas: 1
  selector:

```

```

matchLabels:
  app: app
template:
  metadata:
    labels:
      app: app
  spec:
    containers:
      - name: app
        image: angie:latest
        ports:
          - containerPort: 80

```

2. Запустите приложение:

```
$ kubectl apply -f app-angie.yaml
```

3. Проверьте статус SSL-сертификата и его секрет в Kubernetes:

```
$ kubectl describe certificate domain-name-secret
$ kubectl describe secret domain-name-secret
```

2.4.6 Проверка работы

Откройте в браузере домен, который вы указали в конфигурации (например, <https://otus-kube.mtdlb.ru>).

Если возникают проблемы, можно проверить логи подов:

```
$ kubectl logs -l app=app
```

Также можно посмотреть конфигурацию ANIC (подставьте в `anic-64ff957589-jlg7s` имя пода с ANIC в вашем кластере):

```
$ kubectl exec anic-64ff957589-jlg7s -- angie -T
```

2.4.7 Удаление развернутого приложения

Если нужно удалить все созданные ресурсы, выполните:

```
$ kubectl delete -f app-angie.yaml
```

2.4.8 Проксирование TCP-трафика через ANIC в Kubernetes

Ниже приведен пример настройки проксирования TCP-трафика для доступа к MySQL.

Предварительно необходимо включить `globalConfiguration` в `values.yaml` чарта, за счет него можно определить `listener`.

1. Установите чарт с выключенным параметром `globalConfiguration`:

```

globalConfiguration:
  ## Creates the GlobalConfiguration custom resource. Requires controller.
  →enableCustomResources.
  create: false

```

2. Выполните команду:

```
$ helm upgrade anic .
```

3. Теперь установите чарт с включенным параметром `globalConfiguration` и добавьте TCP-прослушиватель:

```
globalConfiguration:
  ## Creates the GlobalConfiguration custom resource. Requires controller.
  →enableCustomResources.
  create: true

  ## The spec of the GlobalConfiguration for defining the global configuration
  →parameters of the Ingress Controller.
  spec:
    listeners:
      # - name: dns-udp
      #   port: 5353
      #   protocol: UDP
      - name: mysql-tcp
        port: 13306
        protocol: TCP
```

4. Обновите чарт:

```
$ helm upgrade anic .
```

Теперь ANIC будет слушать TCP-порт 13306 для подключения к MySQL.

5. Создайте секрет с паролем root:

mysql-seceret.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: mysql-secret
type: kubernetes.io/basic-auth
stringData:
  password: kljJ218q23cujvdshi
```

6. Создайте постоянное хранилище (volume) для MySQL:

mysql-storage.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mysql-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
```

```

hostPath:
  path: "/mnt/data"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi

```

7. Разверните сам MySQL:

mysql-deployment.yaml

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - image: mysql:8.0
          name: mysql
          env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mysql-secret
                  key: password
          ports:
            - containerPort: 3306
              name: mysql
          volumeMounts:
            - name: mysql-persistent-storage
              mountPath: /var/lib/mysql
      volumes:
        - name: mysql-persistent-storage
          persistentVolumeClaim:
            claimName: mysql-pv-claim
---
apiVersion: v1
kind: Service

```

```

metadata:
  name: mysql
spec:
  ports:
  - port: 3306
  selector:
    app: mysql

```

8. Создайте сервис для MySQL:

mysql-service.yaml

```

apiVersion: v1
kind: Service
metadata:
  name: mysql-ext
spec:
  type: LoadBalancer
  ports:
  - port: 13306
    name: mysql
    targetPort: 13306
  # Kubernetes-метки селектора, использованные в шаблоне подов при создании
  ↪ объекта Deployment.
  selector:
    app: mysql-tcp

```

9. Настройте ANIC для MySQL:

anic-mysql.yaml

```

apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
  name: mysql-tcp
spec:
  listener:
    name: mysql-tcp
    protocol: TCP
  upstreams:
  - name: mysql
    service: mysql
    port: 3306
  action:
    pass: mysql

```

10. Примените настройки, чтобы развернуть сервис:

```

$ kubectl apply -f mysql-secret.yaml
$ kubectl apply -f mysql-storage.yaml
$ kubectl apply -f mysql-deployment.yaml
$ kubectl apply -f mysql-service.yaml
$ kubectl apply -f anic-mysql.yaml

```

Теперь трафик на порт 13306 будет перенаправляться в MySQL.

11. Проверьте статус:

```
$ kubectl get svc           # Список сервисов
$ kubectl describe gc      # Описание GlobalConfiguration
$ kubectl describe ts mysql-tcp # Описание TCP-listener
$ kubectl describe service anic # Информация об ANIC
```

2.5 Миграция с Ingress-NGINX Controller на ANIC

Для миграции ANIC должен быть *установлен* на том же хосте, где уже работает Ingress-NGINX Controller.

2.5.1 Перенос конфигурации

1. Настройте ConfigMap для ANIC, добавив общие параметры для всех Ingress. Эти параметры будут использоваться по умолчанию для всех Ingress, если конкретный манифест их не переопределяет.
2. В манифестах для сервисов измените аннотации Ingress-NGINX Controller на соответствующие аннотации ANIC, где это необходимо.
Для аннотаций, которые не поддерживаются в ANIC, используйте другие варианты настройки конфигурации (см. примеры ниже).
3. По очереди перенесите манифесты для сервисов, проверяя, что каждый манифест работает корректно.

Примечание

Не рекомендуется изменять поле `spec` в ресурсе Ingress. Реализации Ingress-NGINX Controller и ANIC отличаются, что может привести к проблемам совместимости.

2.5.2 Соответствия для аннотаций

В таблице ниже приведены аннотации Ingress-NGINX Controller с эквивалентными аннотациями ANIC.

Ingress-NGINX Controller	ANIC
nginx.ingress.kubernetes.io/ configuration-snippet	angie.software/location-snippets
nginx.ingress.kubernetes.io/load-balance	angie.software/lb-method
nginx.ingress.kubernetes.io/ proxy-buffering	angie.software/proxy-buffering
nginx.ingress.kubernetes.io/ proxy-buffers-number	angie.software/proxy-buffers
nginx.ingress.kubernetes.io/ proxy-buffer-size	angie.software/proxy-buffer-size
nginx.ingress.kubernetes.io/ proxy-connect-timeout	angie.software/proxy-connect-timeout
nginx.ingress.kubernetes.io/ proxy-read-timeout	angie.software/proxy-read-timeout
nginx.ingress.kubernetes.io/ proxy-send-timeout	angie.software/proxy-send-timeout
nginx.ingress.kubernetes.io/ rewrite-target	angie.software/rewrites
nginx.ingress.kubernetes.io/ server-snippet	angie.software/server-snippets
nginx.ingress.kubernetes.io/ssl-redirect	ingress.kubernetes.io/ssl-redirect

Полный список аннотаций ANIC см. в разделе Расширенная конфигурация с помощью аннотаций.

2.5.3 Глобальная конфигурация с помощью ConfigMap

В таблице ниже приведено ключи ConfigMap в Ingress-NGINX Controller и их эквиваленты в ANIC.

Ingress-NGINX Controller	ANIC
disable-access-log	access-log-off
hsts	hsts
hsts-include-subdomains	hsts-include-subdomains
hsts-max-age	hsts-max-age
http-snippet	http-snippets
keep-alive	keepalive-timeout
keep-alive-requests	keepalive-requests
load-balance	lb-method
location-snippet	location-snippets
log-format-escape-json	log-format-escaping: "json"
log-format-stream	stream-log-format
log-format-upstream	log-format
main-snippet	main-snippets
max-worker-connections	worker-connections
max-worker-open-files	worker-rlimit-nofile
proxy-body-size	client-max-body-size
proxy-buffering	proxy-buffering
proxy-buffers-number	proxy-buffers: number size
proxy-buffer-size	proxy-buffers: number size
proxy-connect-timeout	proxy-connect-timeout
proxy-read-timeout	proxy-read-timeout
proxy-send-timeout	proxy-send-timeout
server-name-hash-bucket-size	server-names-hash-bucket-size
proxy-headers-hash-max-size	server-names-hash-max-size

продолжается на следующей странице

Таблица 2 – продолжение с предыдущей страницы

Ingress-NGINX Controller	ANIC
server-snippet	server-snippets
server-tokens	server-tokens
upstream-keepalive-connections	keepalive
use-http2	http2
use-proxy-protocol	proxy-protocol
variables-hash-bucket-size	variables-hash-bucket-size
worker-cpu-affinity	worker-cpu-affinity
worker-processes	worker-processes
worker-shutdown-timeout	worker-shutdown-timeout

2.5.4 Примеры

Настройка терминации SSL и маршрутизации на основе HTTP-путей

Ingress-NGINX Controller:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: nginx-test
spec:
  tls:
    - hosts:
      - foo.bar.com
      secretName: tls-secret
  rules:
    - host: foo.bar.com
      http:
        paths:
          - path: /login
            backend:
              serviceName: login-svc
              servicePort: 80
          - path: /billing
            serviceName: billing-svc
            servicePort: 80
```

ANIC:

```
apiVersion: networking.k8s.io/v1
kind: VirtualServer
metadata:
  name: anic-test
spec:
  host: foo.bar.com
  tls:
    secret: tls-secret
  upstreams:
    - name: login
      service: login-svc
      port: 80
    - name: billing
      service: billing-svc
      port: 80
```

```
routes:
- path: /login
  action:
    pass: login
- path: /billing
  action:
    pass: billing
```

Настройка балансировки нагрузки TCP/UDP и TLS Passthrough

ANIC предоставляет доступ к TCP- и UDP-сервисам с помощью ресурсов TransportServer и GlobalConfiguration. Ingress-NGINX Controller использует для этой цели ConfigMap.

Канареечные развертывания

Канареечные и сине-зеленые развертывания (canary deployment и blue-green deployment) позволяют внедрять изменения в код в продакшн-среде, не прерывая работу пользователей. ANIC выполняет развертывания на уровне передачи данных: чтобы перейти с Ingress-NGINX Controller на ANIC, необходимо сопоставить аннотации Ingress-NGINX Controller с ресурсами VirtualServer и VirtualServerRoute в ANIC.

Ingress-NGINX Controller обрабатывает канареечные аннотации в следующем порядке:

1. `nginx.ingress.kubernetes.io/canary-by-header`
2. `nginx.ingress.kubernetes.io/canary-by-cookie`
3. `nginx.ingress.kubernetes.io/canary-by-weight`

Примечание

Чтобы канареечные аннотации интерпретировались в ANIC аналогично, они должны располагаться в манифесте VirtualServer или VirtualServerRoute в том же порядке.

`nginx.ingress.kubernetes.io/canary-by-header`

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/canary: "true"
nginx.ingress.kubernetes.io/canary-by-header: "httpHeader"
```

ANIC:

```
matches:
- conditions:
- header: httpHeader
  value: never
  action:
    pass: echo
- header: httpHeader
  value: always
  action:
    pass: echo-canary
  action:
    pass: echo
```

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/canary: "true"
nginx.ingress.kubernetes.io/canary-by-header: "httpHeader"
nginx.ingress.kubernetes.io/canary-by-header-value: "my-value"
```

ANIC:

```
matches:
- conditions:
  - header: httpHeader
    value: my-value
  action:
    pass: echo-canary
action:
  pass: echo
```

nginx.ingress.kubernetes.io/canary-by-cookie

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/canary: "true"
nginx.ingress.kubernetes.io/canary-by-cookie: "cookieName"
```

ANIC:

```
matches:
- conditions:
  - cookie: cookieName
    value: never
  action:
    pass: echo
  - cookie: cookieName
    value: always
  action:
    pass: echo-canary
action:
  pass: echo
```

Управление трафиком

Аннотации Ingress-NGINX Controller, для которых пока нет соответствующих полей в ресурсах ANIC, необходимо обрабатывать с помощью фрагментов (сниппетов).

i Примечание

Для работы в чарте необходимо установить `controller.enableSnippets` в `true`.

В примерах ниже аннотации Ingress-NGINX Controller сопоставлены с ресурсами `VirtualServer` и `VirtualServerRoute` в ANIC.

custom-http-errors

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/custom-http-errors: "code"
nginx.ingress.kubernetes.io/default-backend: "default-svc"
```

ANIC:

```
errorPages:
- codes: [code]
  redirect:
    code: 301
    url: default-svc
```

limit-connections

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/limit-connections: "number"
```

ANIC:

```
http-snippets: |
  limit_conn_zone $binary_remote_addr zone=zone_name:size;
routes:
- path: /path
  location-snippets: |
    limit_conn zone_name number;
```

limit-rate

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/limit-rate: "number"
nginx.ingress.kubernetes.io/limit-rate-after: "number"
```

ANIC:

```
location-snippets: |
  limit_rate number;

  limit_rate_after number;
```

limit-rpm

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/limit-rpm: "number"
nginx.ingress.kubernetes.io/limit-burst-multiplier: "multiplier"
```

ANIC:

```
rateLimit:
  rate: numberr/m
```

```
burst: number * multiplier
key: ${binary_remote_addr}
zoneSize: size
```

limit-rps

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/limit-rps: "number"
nginx.ingress.kubernetes.io/limit-burst-multiplier: "multiplier"
```

ANIC:

```
rateLimit:
  rate: numberr/s

  burst: number * multiplier
  key: ${binary_remote_addr}
  zoneSize: size
```

limit-whitelist

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/limit-whitelist: "CIDR"
```

ANIC:

```
http-snippets: |
server-snippets: |
```

rewrite-target

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/rewrite-target: "URI"
```

ANIC:

```
rewritePath: "URI"
```

Манипуляция заголовками

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/enable-cors: "true"
nginx.ingress.kubernetes.io/cors-allow-credentials: "true"

nginx.ingress.kubernetes.io/cors-allow-headers: "X-Forwarded-For"

nginx.ingress.kubernetes.io/cors-allow-methods: "PUT, GET, POST, OPTIONS"

nginx.ingress.kubernetes.io/cors-allow-origin: "*"
```

```
nginx.ingress.kubernetes.io/cors-max-age: "seconds"
```

ANIC:

Установка для Ingress возможна с помощью аннотации `angie.software/location-snippets`. Для работы в чарте необходимо установить `controller.enableSnippets` в `true`.

```
Ingress:
annotations:
  angie.software/location-snippets |
    add_header 'Access-Control-Allow-Origin' '*' always;
    add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS' always;
    add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-With,If-
↵Modified-Since,Cache-Control,Content-Type,Range' always;
    add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range'
↵always;
```

```
VirtualServer
responseHeaders:
  add:
    - name: Access-Control-Allow-Credentials
      value: "true"
    - name: Access-Control-Allow-Headers
      value: "X-Forwarded-For"
    - name: Access-Control-Allow-Methods
      value: "PUT, GET, POST, OPTIONS"
    - name: Access-Control-Allow-Origin
      value: "*"
    - name: Access-Control-Max-Age
      value: "seconds"
```

Проксирование и балансировка нагрузки

В таблице ниже приведены аннотации Ingress-NGINX Controller и параметры в поле `upstream` для ресурсов `VirtualServer` и `VirtualServerRoute`.

Ingress-NGINX Controller	ANIC
<code>nginx.ingress.kubernetes.io/load-balance</code>	<code>lb-method</code>
<code>nginx.ingress.kubernetes.io/proxy-buffering</code>	<code>buffering</code>
<code>nnginx.ingress.kubernetes.io/proxy-buffers-number</code>	<code>buffers</code>
<code>nginx.ingress.kubernetes.io/proxy-next-upstream</code>	<code>next-upstream</code>
<code>nginx.ingress.kubernetes.io/proxy-next-upstream-timeout</code>	<code>next-upstream-timeout</code>
<code>nginx.ingress.kubernetes.io/proxy-read-timeout</code>	<code>read-timeout</code>
<code>nginx.ingress.kubernetes.io/proxy-send-timeout</code>	<code>send-timeout</code>
<code>nginx.ingress.kubernetes.io/service-upstream</code>	<code>use-cluster-ip</code>

Аутентификация mTLS

ANIC может обрабатывать аутентификацию mTLS на уровне входного трафика, требуя предоставления действительных сертификатов для внешних соединений. Настройка реализуется с помощью ресурсов Policy, соответствующих аннотациям Ingress-NGINX Controller.

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/auth-tls-secret: secretName
nginx.ingress.kubernetes.io/auth-tls-verify-client: "on"
nginx.ingress.kubernetes.io/auth-tls-verify-depth: "1"
```

ANIC:

```
ingressMTLS:
  clientCertSecret: secretName
  verifyClient: "on"

  verifyDepth: 1
```

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/proxy-ssl-secret: "secretName"
nginx.ingress.kubernetes.io/proxy-ssl-verify: "on|off"
nginx.ingress.kubernetes.io/proxy-ssl-verify-depth: "1"
nginx.ingress.kubernetes.io/proxy-ssl-protocols: "TLSv1.2"
nginx.ingress.kubernetes.io/proxy-ssl-ciphers: "DEFAULT"
nginx.ingress.kubernetes.io/proxy-ssl-name: "server-name"
nginx.ingress.kubernetes.io/proxy-ssl-server-name: "on|off"
```

ANIC:

```
egressMTLS:
  tlsSecret: secretName

  verifyServer: true|false

  verifyDepth: 1

  protocols: TLSv1.2

  ciphers: DEFAULT

  sslName: server-name

  serverName: true|false
```

Сохранение сессий

Для сохранения сессий в ANIC используются ресурсы Policy. В Ingress-NGINX Controller используются соответствующие аннотации.

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/affinity: "cookie"
nginx.ingress.kubernetes.io/session-cookie-name: "cookieName"
nginx.ingress.kubernetes.io/session-cookie-expires: "x"
nginx.ingress.kubernetes.io/session-cookie-path: "/route"
nginx.ingress.kubernetes.io/session-cookie-secure: "true"
```

ANIC:

```
sessionCookie:
  enable: true

  name: cookieName

  expires: xh

  path: /route

  secure: true
```

Аутентификация через внешний сервис

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/auth-url
```

ANIC:

Используется собственная реализация (применима только для VirtualServer): настройка OIDC.

Привязка sticky-сессий через cookie

Ingress-NGINX Controller:

```
nginx.ingress.kubernetes.io/affinity: "cookie"
nginx.ingress.kubernetes.io/session-cookie-name: "cookie_name"
nginx.ingress.kubernetes.io/session-cookie-expires: "seconds"
nginx.ingress.kubernetes.io/session-cookie-path: "/route"
```

ANIC:

```
angie.software/sticky-cookie-services: "serviceName=example-svc cookie_name_
↪expires=time path=/route"
```

ГЛАВА 3

Аргументы командной строки

ANIC поддерживает ряд аргументов командной строки. Способ указания этих аргументов зависит от того, как вы устанавливаете ANIC:

- Если вы используете *манифесты Kubernetes* (Deployment или DaemonSet) для установки ANIC, измените эти манифесты соответствующим образом, чтобы задать аргументы командной строки. См. документацию по установке с манифестами.
- Если вы используете *Helm* для установки ANIC, измените параметры диаграммы Helm, соответствующие аргументам командной строки. См. документацию по *установке с помощью Helm*.

Ниже в алфавитном порядке перечислены доступные аргументы командной строки:

3.1 -angie-configmaps <строка>

Ресурс ConfigMap для настройки конфигурации Angie. Если ConfigMap задан, но ANIC не может получить его из API Kubernetes, то ANIC не запустится.

Формат: <пространство имен>/<имя>

3.2 -angie-debug

Включает отладку для Angie. Использует бинарник angie-debug. Требуется 'error-log-level: debug' в ConfigMap.

3.3 -angie-reload-timeout <значение>

Время ожидания в миллисекундах, в течение которого ANIC будет ожидать успешной перезагрузки Angie после изменения конфигурации или при начальном запуске.

Значение по умолчанию - 60000.

3.4 -angie-status

Включает Angie stub_status.

По умолчанию true.

3.5 -angie-status-allow-cidrs <строка>

Добавляет блоки IP/CIDR в список разрешений для Angie stub_status.

Несколько IP или CIDR разделяются запятыми. (По умолчанию 127.0.0.1,:::1)

3.6 -angie-status-port <int>

Задаёт порт, на котором доступен Angie stub_status.

Формат: [1024 - 65535] (по умолчанию 8080)

3.7 -angie-status-prometheus <bool>

Включает или отключает выдачу статистики Angie в формате Prometheus.

Формат: false или true (по умолчанию true)

3.8 -angie-status-prometheus-allow-cidrs

Добавляет блоки IP/CIDR в список разрешений для статистики Angie в формате Prometheus.

Несколько IP или CIDR разделяются запятыми. (По умолчанию 127.0.0.1,:::1)

3.9 -angie-status-prometheus-path <строка>

Позволяет менять путь для публикации статистики Angie в формате Prometheus.

По умолчанию используется /p8s.

3.10 `-angie-status-prometheus-port` <int>

Задаёт порт, на котором доступна статистика Angie в формате Prometheus.

Формат: [1024 - 65535] (по умолчанию 8083)

3.11 `-default-server-tls-secret` <строка>

Секрет с сертификатом TLS и ключом для TLS-терминации на сервере по умолчанию.

- Если значение не задано, используются сертификат и ключ в файле `/etc/angie/secrets/default`.
- Если `/etc/angie/secrets/default` не существует, ANIC настроит в Angie отклонение TLS-подключений к серверу по умолчанию.
- Если секрет установлен, но ANIC не может получить его из API Kubernetes, или же не установлен, и ANIC не удастся прочитать файл `/etc/angie/secrets/default`, то ANIC не запустится.

Формат: <пространство имен>/<имя>

3.12 `-disable-ipv6`

Явно отключает прослушиватели IPV6 для узлов, которые не поддерживают стек IPV6.

По умолчанию `false`.

3.13 `-enable-cert-manager`

Включает автоматическое управление сертификатами x509 для ресурсов VirtualServer с помощью cert-manager (cert-manager.io).

Требует `-enable-custom-resources`.

3.14 `-enable-custom-resources`

Включает пользовательские ресурсы.

По умолчанию `true`.

3.15 `-enable-external-dns`

Включает интеграцию с ExternalDNS для настройки общедоступных записей DNS у ресурсов VirtualServer с использованием ExternalDNS.

Требует наличия `-enable-custom-resources`.

3.16 -enable-jwt

Включает функцию аутентификации JWT в ресурсах Policy.

По умолчанию `false`.

3.17 -enable-leader-election

Позволяет выбирать лидера, чтобы избежать ситуации, когда несколько реплик контроллера сообщают о статусе ресурсов Ingress, VirtualServer и VirtualServerRoute; сообщать о статусе будет только одна реплика. По умолчанию `true`.

См. флаг `-report-ingress-status`.

3.18 -enable-oidc

Включает функцию аутентификации по OpenID Connect в ресурсах Policy.

По умолчанию `false`.

3.19 -enable-prometheus-metrics

Позволяет публиковать метрики Angie в формате Prometheus.

3.20 -enable-service-insight

Публикует конечную точку Service Insight для ANIC.

3.21 -enable-snippets

Включает пользовательские фрагменты конфигурации Angie в ресурсах Ingress, VirtualServer, VirtualServerRoute и TransportServer.

По умолчанию `false`.

3.22 -enable-tls-passthrough

Включает сквозную передачу данных по протоколу TLS на порту 443.

Требует наличия `-enable-custom-resources`.

3.23 `-external-service` <строка>

Указывает имя сервиса с типом LoadBalancer, через который поды ANIC делаются доступными извне. Внешний адрес сервиса используется для отчетов о состоянии ресурсов Ingress, VirtualServer и VirtualServerRoute.

Только для ресурсов Ingress: требует наличия `-report-ingress-status`.

3.24 `-global-configuration` <строка>

Ресурс GlobalConfiguration для глобальной настройки ANIC.

Формат: <пространство имен>/<имя>

Требует наличия `-enable-custom-resources`.

3.25 `-health-status`

Добавляет местоположение `"/angie-health"` к серверу по умолчанию. Местоположение отвечает кодом статуса 200 на любой запрос.

Это полезно для внешней проверки работоспособности ANIC.

3.26 `-health-status-uri` <строка>

Задаёт URI местоположения проверки работоспособности на сервере по умолчанию. Требует наличия `-health-status`.

По умолчанию `/angie-health`.

3.27 `-ingress-class` <строка>

Класс ANIC.

Должен быть развернут соответствующий ресурс IngressClass с именем, равным классу. В противном случае ANIC не запустится. ANIC обрабатывает только те ресурсы, которые принадлежат его классу, т. е. имеют ресурс поля `ingressClassName`, равный классу.

ANIC обрабатывает все ресурсы, у которых нет поля `ingressClassName`.

По умолчанию `angie`.

3.28 `-ingresslink` <строка>

Указывает имя ресурса IngressLink, через который предоставляется доступ к подам ANIC через систему BIG-IP. IP-адрес системы BIG-IP используется для отчетов о состоянии ресурсов Ingress, VirtualServer и VirtualServerRoute.

Только для ресурсов Ingress: требует наличия `-report-ingress-status`.

3.29 `-ingress-template-path` <строка>

Путь к шаблону конфигурации Ingress Angie для ресурса Ingress. По умолчанию для Angie используется `angie.ingress.tpl`.

3.30 `-leader-election-lock-name` <строка>

Указывает в том же пространстве имен, где находится контроллер, имя ConfigMap, используемое для блокировки при выборе лидера.

Требуется наличие `-enable-leader-election`.

3.31 `-main-template-path` <строка>

Путь к основному шаблону конфигурации Angie.

- По умолчанию для Angie используется `angie.ingress.tpl`.

3.32 `-prometheus-metrics-listen-port` <int>

Задаёт порт, на котором публикуются метрики Prometheus.

Формат: [1024 - 65535] (по умолчанию 9113)

3.33 `-prometheus-tls-secret` <строка>

Секрет с сертификатом TLS и ключом для TLS-терминации конечной точки метрик Prometheus.

- Если аргумент не задан, конечная точка Prometheus не будет использовать TLS-соединение.
- Если аргумент задан, но ANIC не может получить секрет из API Kubernetes, то ANIC не запустится.

3.34 `-proxy` <строка>

Задаёт использование прокси-сервера для подключения к API Kubernetes, запускаемого командой `"kubectl proxy"`. **Только в целях тестирования.**

ANIC не запускает Angie и не записывает на диск никакие сгенерированные файлы конфигурации Angie.

3.35 `-ready-status`

Включает конечную точку готовности `/angie-ready`. Конечная точка возвращает код успеха, когда Angie загрузил всю конфигурацию после запуска.

По умолчанию `true`.

3.36 `-ready-status-port`

HTTP-порт для конечной точки готовности.

Формат: [1024 - 65535] (по умолчанию 8081)

3.37 `-report-ingress-status`

Обновляет поле адреса в статусе ресурсов Ingress.

Требуется флаг `-external-service` или `-ingresslink`, либо ключ `external-status-address` в ConfigMap.

3.38 `-service-insight-listen-port <int>`

Задаёт порт, на котором публикуется Service Insight.

Формат: [1024 - 65535] (по умолчанию 9114)

3.39 `-service-insight-tls-secret <строка>`

Секрет с сертификатом TLS и ключом для TLS-терминации конечной точки Service Insight.

- Если аргумент не задан, конечная точка Service Insight не будет использовать TLS-соединение.
- Если аргумент задан, но ANIC не может получить секрет из API Kubernetes, то ANIC не запустится.

Формат: `<пространство имен>/<имя>`

3.40 `-tls-passthrough-port <int>`

Задаёт порт для сквозной передачи данных по протоколу TLS. Формат: [1024 - 65535] (по умолчанию 443)

Требуется включить `-enable-custom-resources`.

3.41 `-transportserver-template-path` <строка>

Путь к шаблону конфигурации TransportServer Angie для ресурса TransportServer.

- По умолчанию для Angie используется `angie.transportserver.tpl`.

3.42 `-v` <значение>

Уровень детализации записи логов. Значение по умолчанию — 1, при этом значении записывается минимальное количество логов. Значение 3 полезно для устранения неполадок.

3.43 `-version`

Выводит версию, хэш git-коммита и дату сборки, затем завершает работу.

3.44 `-virtualserver-template-path` <строка>

Путь к шаблону конфигурации VirtualServer Angie для ресурса VirtualServer.

- По умолчанию для Angie используется `angie.ingress.tpl`.

3.45 `-vmodule` <значение>

Разделенный запятыми список параметров `pattern=N` для ведения журнала с фильтрацией файлов.

3.46 `-watch-namespace` <строка>

Разделенный запятыми список пространств имен, за ресурсами которых должен следить ANIC. По умолчанию ANIC отслеживает все пространства имен. Нельзя использовать вместе с `"watch-namespace-label"`.

3.47 `-watch-namespace-label` <строка>

Настраивает в ANIC просмотр только пространств имен с меткой `foo=bar`. По умолчанию ANIC отслеживает все пространства имен. Нельзя использовать вместе с `"watch-namespace"`.

3.48 `-watch-secret-namespace` <строка>

Разделенный запятыми список пространств имен, за которыми ANIC должен следить на предмет наличия секретов. Если этот параметр не настроен, ANIC отслеживает одни и те же пространства имен для всех ресурсов. См. также `"watch-namespace"` и `"watch-namespace-label"`.

3.49 -wildcard-tls-secret <строка>

Секрет с сертификатом TLS и ключом для TLS-терминации каждого узла Ingress или VirtualServer, для которого включена TLS-терминация, но секрет не указан.

- Если аргумент не задан, для таких узлов Ingress и VirtualServer Angie прервет любую попытку установить TLS-соединение.
- Если аргумент задан, но ANIC не может получить секрет из API Kubernetes, то ANIC не запустится.

Формат: <пространство имен>/<имя>

ГЛАВА 4

Версии Angie Ingress Controller (ANIC)

4.1 2025

4.1.1 ANIC 0.7.1

Дата выпуска: 02.06.2025

Добавления

В справку добавлен раздел Общие примеры.

Исправления

Исправление уязвимостей.

4.1.2 ANIC 0.7.0

Дата выпуска: 05.02.2025.

Добавления

Установка ANIC *по лицензии*.

4.2 2024

4.2.1 ANIC 0.6.0

Дата выпуска: 26.12.2024.

Добавления

Новые функции и настройки:

- Добавлены дополнительные поля в метриках Prometheus для ANIC:
 - `controller_pod`
 - `controller_namespace`
 - `controller_class`
- Добавлена возможность настраивать и осуществлять авторизацию клиента на основании результатов подзапроса.
- Версия Angie PRO обновлена до 1.8.1.

4.2.2 ANIC 0.5.0

Дата выпуска: 30.09.2024.

Добавления

Новые функции и настройки:

- Добавлена возможность настраивать OIDC-авторизацию.
- Появилась возможность настройки JWT-авторизацию.
- Добавлена аннотация `angie.software/configmap` с параметром `namespace/config-name`, которая позволяет определить расширенный ConfigMap для заданного Ingress-ресурса.
- Директива `staticLocations` позволяет задавать расположение для раздачи статических файлов.
- Параметр `angie-status-prometheus-path` позволяет менять путь для статистики Angie в формате Prometheus.
- Добавлены параметры настройки SSL для ресурса VirtualServer:
 - `ssl_session_timeout`
 - `ssl_session_cache`
 - `ssl_session_tickets`
 - `ssl_stapling`
 - `ssl_stapling_verify`
- Для директивы `ssl_prefer_server_ciphers` теперь можно задавать значение `off`.
- Появилась возможность добавлять директиву `s_map` в конфигурацию Angie PRO, см. примеры настройки.
- Появилась поддержка директивы `activeHealthProbes` в Angie PRO.
- Версия Angie PRO обновлена до 1.7.0.

Для облегчения процесса миграции сделаны следующие улучшения:

- Параметры типа `boolean` теперь могут принимать значения `true` или `false`, `t` или `f`, `on` или `off` и `1` или `0`.
- Параметр `proxy-buffers` директивы `Upstream.buffers` теперь может принимать только значение количества буферов `number`, без указания размера `size`. Если значение `size` не указано, то по умолчанию будет задано `8K`.

Исправления

- Исправлена ошибка при указании значения `HTTPS` для `backend-protocol`.
- Исправлено отображение `IP` в статусе `k8s`.
- Параметр `include-year` больше нельзя изменять, его значение теперь всегда `true`.

4.2.3 ANIC 0.4.0

Дата выпуска: 04.06.2024.

Добавления

Новые функции и настройки:

- Добавлены алиасы для следующих аннотаций:
 - `angie.software/force-ssl-redirect` — перенаправляет HTTP-запросы на HTTPS.
 - `angie.software/proxy-body-size` — устанавливает максимальный размер для тела запроса, которое может обработать проксируемый сервер.
 - `angie.software/proxy-buffer-size` — определяет размер буфера для чтения заголовков ответов от проксируемого сервера.
 - `angie.software/proxy-buffering` — включает или отключает буферизацию ответов от проксируемого сервера.
 - `angie.software/proxy-buffers` — определяет количество буферов, используемых для хранения ответа от проксируемого сервера.
 - `angie.software/proxy-max-temp-file-size` — задает максимальный размер временного файла, используемого для буферизации больших ответов.
- Директива `server_tokens` теперь может принимать строковые значения.
- Версия Angie PRO обновлена до 1.5.2.

Исправления

- Исправлены требования к `resolver-addresses`.

4.2.4 ANIC 0.3.0

Дата выпуска: 02.03.2024.

Добавления

Новые функции и настройки:

- Добавлена аннотация `angie.software/force-ssl-redirect`, с помощью которой можно переводить небезопасные HTTP-запросы на защищенные HTTPS, что позволит исправить возможные ошибки при использовании SSL.

Примечание

В связи с особенностями работы аннотации `force-ssl-redirect` рекомендуем использовать альтернативную настройку — `backend-protocol` со значением `HTTPS`.

- В Helm-чарты добавлены CRD (Common Resource Definitions), такие как `Virtual Server`, `Virtual Server Route`, `TransportServer`, `Policies`. Теперь пользователи Helm могут использовать эти определения в своих проектах, что значительно расширяет возможности настройки веб-сервера по сравнению со стандартным ресурсом Ingress.

Исправления

- Теперь ANIC запускается от имени пользователя `angie`.

4.3 2023

4.3.1 ANIC 0.2.0

Дата выпуска: 23.11.2023.

Добавления

- Добавлена поддержка консоли `Console Light` для мониторинга активности в реальном времени.
- Теперь можно отключать протокол `ipv6` с помощью `-disable-ipv6`.
- Добавлена точка подключения Prometheus `/ps8` для мониторинга статуса.
- Добавлена поддержка `sticky cookie` и `sticky route`.
- В параметры конфига добавлена переменная `$proxy_upstream_name` для использования в формате логов.

Исправления

- Исправлена ошибка с отсутствием прав доступа к `angie-syslog.sock`.

ГЛАВА 5

Права на интеллектуальную собственность

Документация на программный продукт Angie Ingress Controller (ANIC) является интеллектуальной собственностью ООО «Веб-Сервер».

Copyright © 2025, ООО «Веб-Сервер». Все права защищены.