ООО «Веб-Сервер», ОГРН 1227700436578, ИНН 9704151517
ул. Льва Толстого, д. 14, этаж 1, кв. 2, г. Москва, 119021

Angie Ingress Controller

версия 0.4.0

Руководство по эксплуатации

сент. 12, 2024

Оглавление

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 внесён 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 развертывается 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	3.17	x86_64, arm64
Alt Linux	10	x86_64, arm64
Debian	11	x86_64, arm64

ГЛАВА 2

Настройка

ANIC настраивается путем изменения параметров `ConfigMap` и `Annotation`:

Список 1: Пример ConfigMap

```
$ kubectl apply -f - <<EOF
kind: ConfigMap
apiVersion: v1
metadata:
  name: angie-config
  namespace: angie-ingress
data:
  proxy-connect-timeout: "10s"
   proxy-read-timeout: "10s"
   client-max-body-size: "2m"
EOF
```

2.1 Параметры Ingress Controller

`external-status-`	Задает адрес, который выводится в статусе Ingress ресурса. Имеет приоритет над аргументом командной строки `-external-service`.

2.2 Общие параметры

Параметр	Описание	Умолчание
`proxy-connect-timeout`	Задает значение proxy_connect_timeout и grpc_connect_timeout.	`60s`
`proxy-read-timeout`	Задает значение proxy_read_timeout и grpc_read_timeout	`60s`

продолжается на следующей странице

Таблица 1 – продолжение с предыдущей страницы

Параметр	Описание	Умолчание
`proxy-send-timeout`	Задает значение proxy_send_timeout и grpc_send_timeout	`60s`
`client-max-body-size`	Задает значение client_max_body_size	`1m`
`proxy-buffering`	Включает или отключает буферизацию ответа от проксируемого сервера	`True`
`proxy-buffers`	Задает значение proxy_buffers	Зависит от платформы
`proxy-buffer-size`	Задает значение proxy_buffer_size и grpc_buffer_size	Зависит от платформы
`proxy-max-temp-file-size`	Задает значение proxy_max_temp_file_size	`1024m`
`set-real-ip-from`	Задает значение set_real_ip_from	Нет
`real-ip-header`	Задает значение real_ip_header	`X-Real-IP`
`real-ip-recursive`	Включает или отключает real_ip_recursive	`False`
`default-server-return`	Настраивает ответ в сервере по умолчанию, который перехватывает клиентский запрос, если для запроса не был определен ресурс `Ingress` или `VirtualServer`. Можно установить фиксированный ответ или перенаправление запроса.	Страница с ошибкой HTTP 404
`server-tokens`	Включает или отключает server_tokens	`True`
`worker-processes`	Задает значение worker_processes	`auto`
`worker-rlimit-nofile`	Задает значение worker_rlimit_nofile	Нет
`worker-connections`	Задает значение worker_connections	`1024`
`worker-cpu-affinity`	Задает значение worker_cpu_affinity	Нет
`worker-shutdown-timeout`	Задает значение worker_shutdown_timeout	Нет
`server-names-hash-bucket-siz`	Задает значение server_names_hash_bucket_siz	`256`
`server-names-hash-max-size`	Задает значение server_names_hash_max_size	`1024`
`map-hash-bucket-size`	Задает значение map_hash_bucket_size	`256`
`map-hash-max-size`	Задает значение map_hash_max_size	`2048`
`resolver-addresses`	Задает значение DNS resolver'а	Нет
`resolver-ipv6`	Разрешает или запрещает поиск IPv6-адресов	`True`
`resolver-valid`	Позволяет переопределить срок кэширования DNS-записей	Нет
`resolver-timeout`	Задает значение resolver_timeout	`30s`
`keepalive-timeout`	Задает значение keepalive_timeout	`65s`
`keepalive-requests`	Задает значение keepalive_requests	`100`
`variables-hash-bucket-size`	Задает значение variables_hash_bucket_size	`256`
`variables-hash-max-size`	Задает значение variables_hash_max_size	`1024`

2.3 Параметры ведения журнала

Параметр	Описание	Умолчание
`error-log-level`	Определяет глобальное значение уровня error_log и может принимать одно из следующих значений: *debug*, *info*, *notice*, *warn*, *error*, *crit*, *alert* или *emerg*	`notice`
`access-log-off`	Отключает access_log	`False`
`default-server-access-log-of`	Отключает access_log для сервиса по умолчанию	`False`
`log-format`	Задает общий формат журнала. Для удобства можно использовать несколько строк, разделенных `\n`. В этом случае каждый перевод строки будет заменен на пробел. Все символы ' должны быть экранированы	Нет
`log-format-escaping`	Позволяет задать экранирование символов `json` или `default` в переменных; по умолчанию используется `default`. Значение `none` отключает экранирование	`default`
`stream-log-format`	Задает формат журнала stream для сквозного трафика TCP, UDP и TLS. Для удобства можно использовать несколько строк, разделенных `\n`. В этом случае каждый перевод строки будет заменен на пробел. Все символы ' должны быть экранированы	Нет
`stream-log-format-escaping`	Позволяет задать экранирование символов `json` или `default` в переменных; по умолчанию используется `default`. Значение `none` отключает экранирование	`default`

2.4 Управление URI и заголовками в запросах

`proxy-hide-headers`	Значение одного proxy_hide_header или нескольких
`proxy-pass-headers`	Значение одного proxy_pass_header или нескольких

2.5 Авторизация и SSL/TLS

Параметр	Описание	Умолчание
`redirect-to-https`	Задает правило 301 redirect в зависимости от заголовка *http_x_forwarded_proto*	`False`
`ssl-redirect`	Задает правило 301 redirect для всего входящего HTTP-трафика, чтобы перевести запросы в HTTPS	`True`
`ssl-protocols`	Задает значение ssl_protocols	`TLSv1 TLSv1.1 TLSv1.2`
`ssl-prefer-server-ciphers`	Включает или отключает ssl_prefer_server_ciphers	`False`
`ssl-ciphers`	Задает значение ssl_ciphers	`HIGH:!aNULL:!MD5`
`ssl-dhparam-file`	Указывает файл с параметрами для DHE-шифров	Нет

2.6 Протоколы

Параметр	Описание	Умолчание
`http2`	Включает поддержку протокола HTTP/2	`False`
`proxy-protocol`	Указывает, что все соединения, принимаемые на данном слушающем сокете, должны использовать протокол PROXY	`False`

2.7 Апстримы

Параметр	Описание	Умолчание
`max-fails`	Задает значение max_fails для сервера	`1`
`upstream-zone-size`	Задает имя и размер зоны разделяемой памяти	Нет
`fail-timeout`	Задает значение fail_timeout для сервера	`10s`
`keepalive`	Задействует кэш соединений для группы серверов апстрима	Нет

2.8 Настраиваемые шаблоны

`main-snippets`	Вставляет собственный фрагмент конфигурации в контекст main
`http-snippets`	Вставляет собственный фрагмент конфигурации в контекст http
`location-snippets`	Вставляет собственный фрагмент конфигурации в контекст location
`server-snippets`	Вставляет собственный фрагмент конфигурации в контекст server
`stream-snippets`	Вставляет собственный фрагмент конфигурации в контекст stream
`main-template`	Определяет основной шаблон для основных настроек Angie. По умолчанию шаблон считывается из файла в контейнере
`ingress-template`	Определяет шаблон настроек для ресурса Ingress. По умолчанию шаблон считывается из файла в контейнере
`virtualserver-template`	Определяет шаблон настроек для ресурса *VirtualServer*. По умолчанию шаблон считывается из файла в контейнере

ГЛАВА 3

ConfigMap

Ресурсы ConfigMap позволяют вам настраивать поведение Angie. Например, можно задать количество рабочих процессов или настроить формат журнала доступа.

3.1 Использование ConfigMap

1. Наши инструкции по установке с манифестами развертывают пустой ConfigMap, в то время как манифесты установки по умолчанию указывают ее в аргументах командной строки ANIC. Однако, если вы настроили манифесты, чтобы использовать ConfigMap, обязательно укажите ресурс ConfigMap для использования с помощью аргументов командной строки ANIC.

2. Создайте файл ConfigMap с именем `angie-config.yaml` и установите значения, которые имеют смысл для вашей среды:

```
kind: ConfigMap
apiVersion: v1
metadata:
   name: angie-config
   namespace: angie-ingress
data:
   proxy-connect-timeout: "10s"
   proxy-read-timeout: "10s"
   client-max-body-size: "2m"
```

   См. в разделе *Краткое описание ключей ConfigMap* сведения о доступных ключах ConfigMap (таких как `proxy-connect-timeout` в этом примере).

3. Создайте новый (или обновите существующий) ресурс ConfigMap:

```
kubectl apply -f angie-config.yaml
```

   Конфигурация Angie будет обновлена.

3.2 ConfigMap и аннотации Ingress

Аннотации позволяют настраивать расширенные функции Angie и менять поведение Angie.

ConfigMap применяется глобально, то есть влияет на каждый ресурс Ingress. Напротив, аннотации всегда применяются только к своему ресурсу Ingress. Аннотации позволяют переопределять некоторые ключи ConfigMap. Например, в `angie.software/proxy-connect-timeout` аннотации переопределяют ключ конфигурации `proxy-connect-timeout`.

3.3 ConfigMap и ресурсы VirtualServer, VirtualServerRoute

ConfigMap влияет на все ресурсы VirtualServer и VirtualServerRoute. Однако поля этих ресурсов позволяют переопределять некоторые ключи ConfigMap. Например, поле `connect-timeout` сервера `апстрима` имеет приоритет над ключом ConfigMap `proxy-connect-timeout`.

См. документацию по *ресурсам VirtualServer и VirtualServerRoute*.

3.4 Краткое описание ключей ConfigMap

3.4.1 ANIC (не связанные с конфигурацией Angie)

Ключ ConfigMap	Описание	По умолчанию	Пример
`external-status-address`	Задает адрес, который будет отображаться в статусе ресурсов Ingress. Требуется аргумент командной строки `-report-status`. Имеет приоритет над аргументом `-external-service`.	Н/Д	Отчет о состоянии Ingress

3.4.2 Общая настройка

Ключ ConfigMap	Описание	По умолчанию	Пример
`proxy-connect-timeout`	Задает значение директив proxy_connect_timeout и grpc_connect_timeout.	`60s`	
`proxy-read-timeout`	Задает значение директив proxy_read_timeout и grpc_read_timeout.	`60s`	
`proxy-send-timeout`	Задает значение директив proxy_send_timeout и grpc_send_timeout.	`60s`	
`client-max-body-size`	Задает значение директивы client_max_body_size.	`1m`	
`proxy-buffering`	Включает или отключает буферизацию ответов от проксируемого сервера.	`True`	
`proxy-buffers`	Задает значение директивы proxy_buffers.	Зависит от платформы.	
`proxy-buffer-size`	Задает значение директив proxy_buffer_size и grpc_buffer_size.	Зависит от платформы.	
`proxy-max-temp-file-siz`	Задает значение директивы proxy_max_temp_file_size.	`1024m`	
`set-real-ip-from`	Задает значение директивы set_real_ip_from.	Н/Д	
`real-ip-header`	Задает значение директивы real_ip_header.	`X-Real-IP`	
`real-ip-recursive`	Включает или отключает директиву real_ip_recursive.	`False`	
`default-server-return`	Настраивает директиву return на сервере по умолчанию, которая обрабатывает клиентский запрос, если ни один из узлов ресурсов Ingress или VirtualServer не совпадает. Значение по умолчанию настраивает в Angie возврат страницы с ошибкой 404. Вы можете настроить фиксированный ответ или перенаправление. Например, значение `default-server-return: 302 https://mysite.ru` перенаправит клиент на `https://mysite.ru`.	`404`	
`server-tokens`	Включает или отключает директиву server_tokens.	`True`	
`worker-processes`	Задает значение директивы worker_processes.	`auto`	
`worker-rlimit-nofile`	Задает значение директивы worker_rlimit_nofile.	Н/Д	
`worker-connections`	Задает значение директивы worker_connections.	`1024`	
`worker-cpu-affinity`	Задает значение директивы worker_cpu_affinity.	Н/Д	
`worker-shutdown-timeout`	Задает значение директивы worker_shutdown_timeout.	Н/Д	
`server-names-hash-bucke`	Задает значение директивы server_names_hash_bucket_size.	`256`	
`server-names-hash-max-s`	Задает значение директивы server_names_hash_max_size.	`1024`	
`map-hash-bucket-size`	Задает значение директивы map_hash_bucket_size.	`256`	
`map-hash-max-size`	Задает значение директивы map_hash_max_size.	`2048`	
`resolver-addresses`	Задает значение адресов resolver.	Н/Д	
`resolver-ipv6`	Включает разрешение IPv6 в распознавателе.	`True`	
`resolver-timeout`	Задает значение resolver_timeout для разрешения имен.	`30s`	
`keepalive-timeout`	Задает значение директивы keepalive_timeout.	`65s`	
`keepalive-requests`	Задает значение директивы keepalive_requests.	`100`	
`variables-hash-bucket-s`	Задает значение директивы	`256`	

3.4.3 Ведение журнала

Ключ ConfigMap	Описание	По умолчанию	Пример
`error-log-level`	Задает глобальный уровень журнала ошибок для Angie.	`notice`	
`access-log-off`	Отключает журнал доступа.	`False`	
`default-server-access-l`	Отключает журнал доступа для сервера по умолчанию. Если журнал доступа отключен глобально (`access-log-off: "True"`), то журнал доступа к серверу по умолчанию всегда отключен.	`False`	
`log-format`	Задает настраиваемый формат журнала для HTTP- и HTTPS-трафика. Для удобства можно определить формат журнала в нескольких строках (строки разделяются символом `\n`). В этом случае ANIC заменит каждый символ `\n` символом пробела. Все символы `'` должны быть экранированы.		
`log-format-escaping`	Задает экранирующие символы для переменных формата журнала. Поддерживаемые значения: `json` (экранирование JSON), `default` (экранирование по умолчанию), `none` (отключает экранирование).	`default`	
`stream-log-format`	Задает настраиваемый формат журнала для сквозного трафика TCP, UDP и TLS. Для удобства можно определить формат журнала в нескольких строках (строки разделяются символом `\n`). В этом случае ANIC заменит каждый символ `\n` символом пробела. Все символы `'` должны быть экранированы.		
`stream-log-format-escap`	Задает экранирующие символы для переменных формата журнала потока. Поддерживаемые значения: `json` (экранирование JSON), `default` (экранирование по умолчанию), `none` (отключает экранирование).	`default`	

3.4.4 Манипулирование URI и заголовками запроса

Ключ ConfigMap	Описание	По умолчанию	Пример
`proxy-hide-headers`	Задает значение одной директивы proxy_hide_header или нескольких.	Н/Д	`"angie.software/proxy-hide-headers": "header-a, header-b"`
`proxy-pass-headers`	Задает значение одной директивы proxy_pass_header или нескольких.	Н/Д	`"angie.software/proxy-pass-headers": "header-a, header-b"`

3.4.5 Аутентификация, SSL, TLS

Ключ ConfigMap	Описание	По умолчанию	Пример
`redirect-to-https`	Задает правило перенаправления 301 на основе значения заголовка `http_x_forwarded_proto` в серверном блоке, требуя, чтобы входящий трафик шел по протоколу HTTPS. Полезно при терминировании SSL в системе балансировки нагрузки перед ANIC.	`False`	
`ssl-redirect`	Задает безусловное правило перенаправления 301 для всего входящего HTTP-трафика, требуя, чтобы входящий трафик шел по протоколу HTTPS.	`True`	
`hsts`	Включает режим HTTP Strict Transport Security (HSTS): заголовок HSTS добавляется к ответам от проксируемых серверов. Директива `preload` будет включена в заголовок.	`False`	
`hsts-max-age`	Задает значение директивы `max-age` заголовка HSTS.	2592000 (1 месяц)	
`hsts-include-subdomains`	Добавляет директиву `includeSubDomains` в заголовок HSTS.	`False`	
`hsts-behind-proxy`	Включает HSTS на основе значения заголовка запроса `http_x_forwarded_proto`. Следует использовать только в том случае, если в балансировщике нагрузки (прокси-сервере) перед ANIC настроено терминирование TLS. **Примечание** Чтобы управлять перенаправлением с HTTP на HTTPS, настройте аннотацию `angie.software/redirect-to-https`.	`False`	
`ssl-protocols`	Задает значение директивы ssl_protocols.	`TLSv1` `TLSv1.1` `TLSv1.2`	
`ssl-prefer-server-ciphe`	Включает или отключает директиву ssl_prefer_server_ciphers.	`False`	
`ssl-ciphers`	Задает значение директивы ssl_ciphers.	`HIGH:!` `aNULL:!` `MD5`	
`ssl-dhparam-file`	Задает содержимое файла dhparam. Контроллер создаст файл и установит значение директивы ssl_dhparam с указанием пути к файлу.	Н/Д	

3.4.6 Прослушиватели

Ключ ConfigMap	Описание	По умолчанию	Пример
`http2`	Включает HTTP/2 на серверах с включенным SSL.	`False`	
`proxy-protocol`	Включает прокси-протокол для входящих соединений.	`False`	

3.4.7 Бэкэнд-сервисы (апстримы)

Ключ ConfigMap	Описание	По умолчанию	Пример
`lb-method`	Задает метод балансировки нагрузки. Чтобы использовать циклический метод, укажите `"round_robin"`.	`"random two least_conn"`	
`max-fails`	Задает значение параметра max_fails директивы `server`.	`1`	
`upstream-zone-size`	Задает размер зоны разделяемой памяти для апстримов.		
`fail-timeout`	Задает значение параметра fail_timeout директивы `server`.	`10s`	
`keepalive`	Задает значение директивы keepalive. Обратите внимание: если значение больше 0, в сгенерированную конфигурацию добавляется `proxy_set_header Connection "";`.	`0`	

3.4.8 Фрагменты и пользовательские шаблоны

Ключ ConfigMap	Описание	По умолчанию	Пример
`main-snippets`	Задает пользовательский фрагмент в основном контексте.	Н/Д	
`http-snippets`	Задает пользовательский фрагмент в контексте http.	Н/Д	
`location-snippets`	Задает пользовательский фрагмент в контексте location.	Н/Д	
`server-snippets`	Задает пользовательский фрагмент в контексте server.	Н/Д	
`stream-snippets`	Задает пользовательский фрагмент в контексте stream.	Н/Д	
`main-template`	Задает основной шаблон конфигурации Angie.	По умолчанию шаблон считывается из файла в контейнере.	
`ingress-template`	Задает шаблон конфигурации Angie для ресурса Ingress.	По умолчанию шаблон считывается из файла в контейнере.	
`virtualserver-template`	Задает шаблон конфигурации Angie для ресурса VirtualServer.	По умолчанию шаблон считывается из файла в контейнере.	

ГЛАВА 4

GlobalConfiguration

Ресурс GlobalConfiguration позволяет вам определить глобальные параметры конфигурации ANIC. Он реализован как пользовательский ресурс.

Ресурс поддерживает настройку прослушивателей для балансировки нагрузки TCP и UDP. Прослушиватели требуются *ресурсам TransportServer*.

4.1 Предварительные требования

При установке ANIC с манифестами необходимо указать ссылку на ресурс GlobalConfiguration в аргументе командной строки -global-configuration. Для ANIC требуется только один ресурс GlobalConfiguration.

4.2 Спецификация GlobalConfiguration

Ресурс GlobalConfiguration определяет глобальные параметры конфигурации ANIC. Ниже приведен пример:

```
apiVersion: k8s.angie.software/v1alpha1
kind: GlobalConfiguration
metadata:
  name: angie-configuration
  namespace: angie-ingress
spec:
  listeners:
  - name: dns-udp
    port: 5353
    protocol: UDP
  - name: dns-tcp
    port: 5353
    protocol: TCP
```

Поле	Описание	Тип	Обязательно
`listeners`	Список прослушивателей.	*[]listener*	Нет

4.2.1 Прослушиватель

Прослушиватель определяет комбинацию протокола и порта, которые Angie будет использовать при приеме трафика для *TransportServer*:

```
name: dns-tcp
port: 5353
protocol: TCP
```

Поле	Описание	Тип	Обязательно
`name`	Имя прослушивателя. Это должна быть допустимая метка DNS, как определено в RFC 1035. Например, допустимы значения `hello` и `listener-123`. Имя должно быть уникальным среди всех прослушивателей. Имя `tls-passthrough` зарезервировано для встроенного прослушивателя TLS Passthrough и не может быть использовано.	`string`	Да
`port`	Порт прослушивателя. Порт должен находиться в диапазоне `1..65535` со следующими исключениями: `80`, `443`, порт [статуса](/angie-ingress-controller/logging-and-monitoring/status-page). Комбинация порта и протокола должна быть уникальна среди всех прослушивателей.	`int`	Да
`protocol`	Протокол прослушивателя. Поддерживаемые значения: `TCP` и `UDP`.	`string`	Да

4.3 Использование GlobalConfiguration

Вы можете использовать обычные команды `kubectl` для работы с ресурсом GlobalConfiguration.

Например, следующая команда создает ресурс GlobalConfiguration, определенный в `global-configuration.yaml` с именем `angie-configuration`:

```
$ kubectl apply -f global-configuration.yaml

    globalconfiguration.k8s.angie.software/angie-configuration created
```

Предполагая, что пространство имен ресурса называется `angie-ingress`, вы можете получить ресурс, запустив:

```
$ kubectl get globalconfiguration angie-configuration -n angie-ingress

    NAME                  AGE
    angie-configuration   13s
```

В kubectl get и подобных командах также можно использовать короткое имя `gc` вместо `globalconfiguration`.

4.3.1 Валидация

Для ресурса GlobalConfiguration доступны два типа валидации:

- *Структурная валидация* с помощью `kubectl` и сервера Kubernetes API.
- *Всесторонняя валидация* с помощью ANIC.

Структурная валидация

Пользовательское определение ресурса для GlobalConfiguration включает структурную схему OpenAPI, которая описывает тип каждого поля ресурса.

Если вы попытаетесь создать (или обновить) ресурс с нарушением структурной схемы (например, используете строковое значение для поля порта прослушивателя), `kubectl` и сервер Kubernetes API отклонят такой ресурс:

- Пример проверки `kubectl`:

```
$ kubectl apply -f global-configuration.yaml

    error: error validating "global-configuration.yaml": error validating
    data: ValidationError(GlobalConfiguration.spec.listeners[0].port):
    invalid type for
    software.angie.k8s.v1alpha1.GlobalConfiguration.spec.listeners.port:
    got "string", expected "integer"; if you choose to ignore these
    errors, turn validation off with --validate=false
```

- Пример проверки сервера API Kubernetes:

```
$ kubectl apply -f global-configuration.yaml --validate=false

    The GlobalConfiguration "angie-configuration" is invalid: []: Invalid
    value: map[string]interface {}{ ... }: validation failure list:
    spec.listeners.port in body must be of type integer: "string"
```

Если ресурс не отклонен (то есть не нарушает структурную схему), ANIC проверит его дополнительно.

Всесторонняя валидация

ANIC проверяет поля ресурса GlobalConfiguration. Если ресурс недопустим, ANIC не будет его использовать. Рассмотрим следующие два случая:

1. Если при запуске пода ANIC ресурс GlobalConfiguration недопустим, ANIC не сможет запуститься и завершит работу с ошибкой.
2. Если ресурс GlobalConfiguration становится недействительным, когда ANIC запущен, то ANIC проигнорирует новую версию. Он сообщит об ошибке и продолжит использовать предыдущую версию. Когда ресурс снова станет действительным, ANIC начнет его использовать.

Примечание

Если ресурс GlobalConfiguration был удален во время работы ANIC, тот продолжит использовать предыдущую версию ресурса.

Вы можете проверить, успешно ли ANIC применил конфигурацию для GlobalConfiguration. Для нашего ресурса GlobalConfiguration `angie-configuration` мы можем запустить:

```
$ kubectl describe gc angie-configuration -n angie-ingress

    . . .
    Events:
      Type     Reason    Age   From                      Message


      Normal   Updated   11s   angie-ingress-controller  GlobalConfiguration
      angie-ingress/angie-configuration was updated
```

Обратите внимание, что раздел «События» (Events) включает событие Normal с причиной Updated, которое информирует нас о том, что конфигурация была успешно применена.

Если вы создадите недопустимый ресурс, ANIC отклонит его и выдаст событие Rejected. Например, если вы создадите ресурс GlobalConfiguration *angie-configuration* с несколькими прослушивателями, для которых задан один и тот же протокол UDP и порт 53, вы получите:

```
   $ kubectl describe gc angie-configuration -n angie-ingress

       . . .
       Events:
        Type     Reason    Age   From                      Message


       Normal    Updated   55s   angie-ingress-controller  GlobalConfiguration
       angie-ingress/angie-configuration was updated

       Warning   Rejected  6s    angie-ingress-controller  GlobalConfiguration
       angie-ingress/angie-configuration is invalid and was rejected:
       spec.listeners: Duplicate value: "Duplicated port/protocol combination
       53/UDP"


Обратите внимание, что раздел "События" (Events) включает предупреждающее
событие с указанием причины отклонения.
```

ГЛАВА 5

Policy

Ресурс Policy позволяет настраивать такие функции, как контроль доступа и ограничение скорости; их можно добавить к вашим ресурсам VirtualServer и VirtualServerRoute.

Он реализован как пользовательский ресурс.

Это справочная документация по ресурсу Policy.

5.1 Предварительные требования

Политики работают совместно с ресурсами VirtualServer и VirtualServerRoute, которые необходимо создавать отдельно.

5.2 Спецификация Policy

Ниже приведен пример политики, которая разрешает доступ клиентам из подсети `10.0.0.0/8` и запрещает доступ любым другим:

```
apiVersion: k8s.angie.software/v1
kind: Policy
metadata:
  name: allow-localhost
spec:
  accessControl:
    allow:
    - 10.0.0.0/8
```

Поле	Описание	Тип	Обязательно
`AccessControl`	Политика контроля доступа, основанная на IP-адресе клиента.	*AccessContr*	Нет
`ingressClassName`	Указывает, какой экземпляр ANIC должен обрабатывать ресурс Policy.	`string`	Нет
`rateLimit`	Политика ограничения скорости управляет скоростью обработки запросов по определенному ключу.	ref:*rateLimit*	Нет
`basicAuth`	Политика базовой аутентификации настраивает в Angie аутентификацию клиентских запросов с использованием базовой аутентификации HTTP по учетным данным.	*BasicAuth*	Нет
`ingressMTLS`	Политика IngressMTLS настраивает проверку сертификата клиента.	*IngressMTLS*	Нет
`egressMTLS`	Политика EgressMTLS настраивает аутентификацию и проверку сертификата апстрима.	*EgressMTLS*	Нет

Примечание

Политика должна включать в себя ровно одно значение.

5.2.1 AccessControl

Политика контроля доступа настраивает в Angie отклонение или принятие запросов от клиентов с указанными IP-адресами и подсетями.

Например, следующая политика разрешает доступ клиентам из подсети `10.0.0.0/8` и запрещает доступ любым другим:

```
accessControl:
  allow:
  - 10.0.0.0/8
```

Напротив, приведенная ниже политика делает обратное: запрещает доступ клиентам с `10.0.0.0/8` и разрешает доступ любым другим клиентам:

```
accessControl:
  deny:
  - 10.0.0.0/8
```

Примечание

Функция реализована с использованием модуля Angie http_access. Политика контроля доступа ANIC поддерживает либо разрешающие, либо запрещающие правила, но не оба вида сразу (в отличие от модуля).

Поле	Описание	Тип	Обязательно
`allow`	Разрешает доступ для указанных сетей или адресов. Например, `192.168.1.1` или `10.1.1.0/16`.	`[]strin`	Нет
`deny`	Запрещает доступ для указанных сетей или адресов. Например, `192.168.1.1` или `10.1.1.0/16`.	`[]strin`	Нет

AccessControl должен включать либо `allow`, либо `deny`.

Поведение слияния AccessControl

Ресурс VirtualServer или VirtualServerRoute может ссылаться на несколько политик контроля доступа. Например, здесь мы ссылаемся на две политики, в каждой из которых настроен список разрешений:

```
policies:
  - name: allow-policy-one
  - name: allow-policy-two
```

Когда вы ссылаетесь на несколько политик контроля доступа, ANIC объединит их содержимое в один список разрешений или запретов.

Ссылки как на разрешающие, так и на запрещающие политики, как показано в примере ниже, не поддерживаются. Если указаны ссылки как на разрешающие, так и на запрещающие списки, ANIC использует только политики разрешающих списков.

```
policies:
  - name: deny-policy
  - name: allow-policy-one
  - name: allow-policy-two
```

RateLimit

Политика ограничения скорости настраивает в Angie ограничение скорости обработки запросов.

Например, следующая политика ограничит все последующие запросы, поступающие с одного IP-адреса, при превышении скорости в 10 запросов в секунду:

```
rateLimit:
  rate: 10r/s
  zoneSize: 10M
  key: ${binary_remote_addr}
```

Примечание

Функция реализована с использованием модуля Angie http_limit_req_module.

Поле	Описание	Тип	Обязательно
`rate`	Допустимая скорость запросов. Скорость указывается в запросах в секунду (r/s) или запросах в минуту (r/m).	`string`	Да
`key`	Ключ, к которому применяется ограничение скорости. Может содержать текст, переменные или их комбинацию. Переменные должны заключены в `${}`. Например: `${binary_remote_addr}`. Допустимые переменные: `$binary_remote_addr`, `$request_uri`, `$url`, `$http_`, `$args`, `$arg_`, `$cookie_`.	`string`	Да
`zoneSize`	Размер зоны разделяемой памяти. Допускаются только положительные значения. Допустимые суффиксы - `k` или `m`; если суффикс не задан, предполагается `k`.	`string`	Да
`delay`	Указывает предел, при достижении которого избыточные запросы становятся отложенными. Если этот параметр не задан, задерживаются все избыточные запросы.	`int`	Нет
`noDelay`	Отключает задержку избыточных запросов при ограничении количества запросов. Имеет приоритет над `delay`, если заданы оба параметра.	`bool`	Нет
`burst`	Избыточные запросы задерживаются до тех пор, пока их количество не превысит размер `burst`, после чего запрос завершается с ошибкой.	`int`	Нет
`dryRun`	Включает режим сухого прогона. В этом режиме ограничение скорости фактически не применяется, но количество избыточных запросов учитывается, как обычно, в зоне разделяемой памяти.	`bool`	Нет
`logLevel`	Устанавливает желаемый уровень ведения журнала для случаев, когда сервер отказывается обрабатывать запросы из-за превышения скорости или задерживает обработку запросов. Допустимые значения: `info`, `notice`, `warn` или `error`. Значение по умолчанию - `error`.	`string`	Нет
`rejectCode`	Задает код состояния, возвращаемый в ответ на отклоненные запросы. Значение должно попадать в диапазон `400..599`. Значение по умолчанию - `503`.	`int`	Нет

Для каждой политики, на которую ссылается VirtualServer или его VirtualServerRoute, ANIC сгенерирует единую зону ограничения скорости, определенную директивой http_limit_req. Если два ресурса VirtualServer ссылаются на одну и ту же политику, ANIC сгенерирует две разные зоны ограничения скорости, по одной на каждый VirtualServer.

Поведение слияния RateLimit

Ресурс VirtualServer или VirtualServerRoute может ссылаться на несколько политик ограничения скорости. Например, здесь мы ссылаемся на две политики:

```
policies:
  - name: rate-limit-policy-one
  - name: rate-limit-policy-two
```

Когда вы ссылаетесь на несколько политик ограничения скорости, ANIC настроит в Angie использование всех указанных ограничений скорости. Если определено несколько политик, каждая дополнительная политика наследует параметры `dryRun`, `LogLevel` и `rejectCode` из первой политики, на которую идет ссылка (`rate-limit-policy-one` в примере выше).

5.2.2 BasicAuth

Настраивает в Angie аутентификацию клиентских запросов при помощи базовой схемы аутентификации HTTP.

Например, следующая политика будет отклонять все запросы, которые не содержат действительную комбинацию имени пользователя и пароля в заголовке HTTP `Authentication`

```
basicAuth:
  secret: htpasswd-secret
  realm: "My API"
```

Примечание

Функция реализована с использованием модуля Angie http_auth_basic.

Поле	Описание	Тип	Обязательно
`secret`	Имя секрета Kubernetes, в котором хранится конфигурация Htpasswd. Он должен находиться в том же пространстве имен, что и ресурс Policy. Секрет должен иметь тип `angie.software/htpasswd`, а конфигурация должна храниться в секрете по ключу `htpasswd`; в противном случае секрет будет отклонен как недействительный.	`string`	Да
`realm`	Область для базовой аутентификации.	`string`	Нет

Поведение слияния BasicAuth

Ресурс VirtualServer или VirtualServerRoute может ссылаться на несколько политик базовой аутентификации. При этом будет применяться только одна из них. Все последующие ссылки будут проигнорированы. Например, здесь мы ссылаемся на две политики:

```
policies:
  - name: basic-auth-policy-one
  - name: basic-auth-policy-two
```

В этом примере ANIC будет использовать конфигурацию из первой ссылки на политику «basic-auth-policy-one» и игнорирует «basic-auth-policy-two».

5.2.3 IngressMTLS

Политика IngressMTLS настраивает проверку сертификата клиента.

Например, следующая политика будет проверять сертификат клиента, используя сертификат Центра Сертификации, указанный в `ingress-mtls-secret`:

```
ingressMTLS:
  clientCertSecret: ingress-mtls-secret
  verifyClient: "on"
  verifyDepth: 1
```

Ниже приведен пример `ingress-mtls-secret` типа `angie.software/ca`

```
kind: Secret
metadata:
  name: ingress-mtls-secret
apiVersion: v1
type: angie.software/ca
data:
  ca.crt: <base64encoded-certificate>
```

У ресурса VirtualServer, который ссылается на политику IngressMTLS, должно быть следующие настройки:

- включено завершение TLS.
- ссылка на политику в спецификации VirtualServer. Не разрешается ссылаться на политику IngressMTLS в маршруте или вложенном маршруте VirtualServerRoute.

Если эти условия нарушены, Angie будет отправлять клиентам код состояния 500.

Вы можете передавать сведения о сертификате клиента, включая сам сертификат, серверам апстрима. Например:

```
action:
  proxy:
    upstream: webapp
    requestHeaders:
      set:
        - name: client-cert-subj-dn
          value: ${ssl_client_s_dn} # subject DN
        - name: client-cert
          value: ${ssl_client_escaped_cert} # клиентский сертификат в формате PEM␣
↪(urlencoded)
```

Мы используем параметр `requestHeaders` в Action.Proxy для задания значений двух заголовков, которые Angie будет передавать серверам апстрима. См. список встроенных переменных,

поддерживаемых модулем `http_ssl`, которые вы можете использовать для передачи сведений о сертификате клиента.

Примечание

Функция реализована с использованием модуля Angie http_ssl.

Использование списка отзыва сертификатов

Политика IngressMTLS поддерживает настройку списка CRL для политики. Это можно сделать одним из двух способов.

Примечание

Одновременно можно использовать только один из этих параметров конфигурации.

1. Добавление в тип секрета `angie.software/ca`поля `ca.crl`, которое содержит список отзыва сертификатов в кодировке base64. Пример YAML:

```
kind: Secret
metadata:
  name: ingress-mtls-secret
apiVersion: v1
type: angie.software/ca
data:
  ca.crt: <base64encoded-certificate>
  ca.crl: <base64encoded-crl>
```

2. Добавление поля `crlFileName` с именем CRL-файла в спецификацию политики IngressMTLS.

   Примечание

   Этот параметр конфигурации следует использовать только при наличии CRL-файла размером более 1 МБ; в противном случае рекомендуется использовать для управления CRL тип секрета `angie.software/ca`.

   Пример YAML:

```
apiVersion: k8s.angie.software/v1
kind: Policy
metadata:
  name: ingress-mtls-policy
spec:
ingressMTLS:
    clientCertSecret: ingress-mtls-secret
    crlFileName: webapp.crl
    verifyClient: "on"
    verifyDepth: 1
```

Предупреждение

При настройке CRL с помощью поля `ingressMTLS.crlFileName` следует учитывать дополнительный контекст:

1. ANIC ожидает, что CRL, в данном случае `webapp.crl`, будет находиться в каталоге `/etc/angie/secrets`. Для развертывания ANIC необходимо будет добавить точку подключения тома. Добавьте свой CRL в каталог `/etc/angie/secrets`.
2. При обновлении содержимого списка CRL (например, был отозван новый сертификат) Angie необходимо перезагрузить, чтобы отразились последние изменения. В зависимости от вашей среды для этого может потребоваться обновить имя списка CRL и применить это обновление к политике `ingress-mtls.yaml`, чтобы Angie получил последнюю версию CRL.

Обратитесь к документации Kubernetes по томам, чтобы найти наилучшую реализацию для вашей среды.

Поле	Описание	Тип	Обязательно
`clientCertSecret`	Имя секрета Kubernetes, в котором хранится сертификат центра сертификации. Он должен находиться в том же пространстве имен, что и ресурс Policy. Секрет должен иметь тип `angie.software/ca`, а конфигурация должна храниться в секрете по ключу `ca.crt`; в противном случае секрет будет отклонен как недействительный.	`string`	Да
`verifyClient`	Верификация для клиента. Допустимые значения: `"on"`, `"off"`, `"optional"`, `"optional_no_ca"`. Значение по умолчанию - `"on"`.	`string`	Нет
`verifyDepth`	Устанавливает глубину проверки в цепочке клиентских сертификатов. Значение по умолчанию равно 1.	`int`	Нет
`crlFileName`	Имя файла списка отзыва сертификатов. ANIC будет искать этот файл в каталоге */etc/angie/secrets*	`string`	Нет

Поведение слияния IngressMTLS

Ресурс VirtualServer может ссылаться только на одну политику IngressMTLS. Все последующие ссылки будут проигнорированы. Например, здесь мы ссылаемся на две политики:

```
policies:
  - name: ingress-mtls-policy-one
  - name: ingress-mtls-policy-two
```

В этом примере ANIC будет использовать конфигурацию из первой ссылки на политику `ingress-mtls-policy-one` и игнорирует `ingress-mtls-policy-two`.

5.2.4 EgressMTLS

EgressMTLS настраивает аутентификацию и проверку сертификатов для апстримов.

Например, следующая политика будет использовать `egress-mtls-secret` для аутентификации в приложении апстрима и `egress-trusted-ca-secret` для проверки сертификата приложения:

```
egressMTLS:
  tlsSecret: egress-mtls-secret
  trustedCertSecret: egress-trusted-ca-secret
  verifyServer: on
  verifyDepth: 2
```

Примечание

Функция реализована с использованием модуля Angie http_proxy.

Поле	Описание	Тип	Обязательно
`tlsSecret`	Имя секрета файла Kubernetes, в котором хранятся сертификат и ключ TLS. Он должен находиться в том же пространстве имен, что и ресурс Policy. Секрет должен иметь тип `kubernetes.io/tls`, сертификат - храниться в секрете под ключом `tls.crt`, а ключ - как `tls.key`; в противном случае секрет будет отклонен как недействительный.	`string`	Нет
`trustedCertSecret`	Имя секрета Kubernetes, в котором хранится сертификат центра сертификации. Он должен находиться в том же пространстве имен, что и ресурс Policy. Секрет должен иметь тип `angie.software/ca`, а конфигурация должна храниться в секрете по ключу `ca.crt`; в противном случае секрет будет отклонен как недействительный.	`string`	Нет
`verifyServer`	Включает проверку сертификата HTTPS-сервера апстрима.	`bool`	Нет
`verifyDepth`	Устанавливает глубину проверки в цепочке сертификатов проксируемого HTTPS-сервера. Значение по умолчанию равно `1`.	`int`	Нет
`sessionReuse`	Позволяет повторно использовать SSL-сеансы к апстримам. Значение по умолчанию равно `true`.	`bool`	Нет
`serverName`	Позволяет передавать имя сервера через расширение `SNI`.	`bool`	Нет
`sslName`	Позволяет переопределить имя сервера, используемое для проверки сертификата HTTPS-сервера апстрима.	`string`	Нет
`ciphers`	Указывает разрешенные шифры для запросов к HTTPS-серверу апстрима. Значение по умолчанию - `DEFAULT`.	`string`	Нет
`protocols`	Задает протоколы для запросов к HTTPS-серверу апстрима. Значение по умолчанию - `TLSv1, TLSv1.1, TLSv1.2`.	`string`	Нет

Поведение слияния EgressMTLS

Ресурс VirtualServer или VirtualServerRoute может ссылаться на несколько политик EgressMTLS. При этом будет применяться только одна из них. Все последующие ссылки будут проигнорированы. Например, здесь мы ссылаемся на две политики:

```
policies:
  - name: egress-mtls-policy-one
  - name: egress-mtls-policy-two
```

В этом примере ANIC будет использовать конфигурацию из первой ссылки на политику `egress-mtls-policy-one` и игнорирует `egress-mtls-policy-two`.

5.2.5 Применение политик

Политики можно применять как к ресурсам VirtualServer, так и к VirtualServerRoute. Например:

```
- VirtualServer:
    apiVersion: k8s.angie.software/v1
    kind: VirtualServer
    metadata:
      name: cafe
      namespace: cafe
    spec:
      host: cafe.example.com
      tls:
        secret: cafe-secret
      policies: # spec policies
        - name: policy1
    upstreams:
      - name: coffee
        service: coffee-svc
        port: 80
    routes:
      - path: /tea
        policies: # route policies
        - name: policy2
          namespace: cafe
          route: tea/tea
      - path: /coffee
        policies: # route policies
          - name: policy3
            namespace: cafe
        action:
          pass: coffee
```

В случае VirtualServer политику можно применить:

- для всех маршрутов (политики спецификации)
- к определенному маршруту (политики маршрутов)

  Политики маршрутов имеют приоритет над политиками спецификации *того же типа*. Если в примере выше тип политик `policy-1` и `policy-3` - `AccessControl`, то для запросов к `cafe.example.com/coffee` Angie применит `policy-3`.

  Переопределение обеспечивается Angie: политики спецификации реализуются в контексте конфигурации `server`, а политики маршрутов реализуются в контексте `location`. В результате приоритет в рамках одного типа имеют политики маршрутов.

- Ресурс VirtualServerRoute, на который ссылается указанный выше VirtualServer:

```
apiVersion: k8s.angie.software/v1
kind: VirtualServerRoute
metadata:
  name: tea
  namespace: tea
spec:
  host: cafe.example.com
  upstreams:
    - name: tea
      service: tea-svc
      port: 80
  subroutes: # subroute policies
    - path: /tea
      policies:
      - name: policy4
        namespace: tea
      action:
        pass: tea
```

В VirtualServerRoute можно применить политику к вложенному маршруту (политики вложенных маршрутов).

Политики вложенных маршрутов имеют приоритет над политиками спецификации того же типа. В приведенном выше примере, если тип политик `policy-1` (в VirtualServer) и `policy-4` - `AccessControl`, то для запросов к `cafe.example.com/tea` Angie будет применять `policy-4`. Как и в случае с VirtualServer, переопределение обеспечивается средствами Angie.

Политики вложенных маршрутов всегда имеют приоритет над политиками маршрутов независимо от типа. Например, политика `policy-2` в маршруте VirtualServer будет проигнорирована на вложенном маршруте `/tea`, поскольку у того есть свои собственные политики (в нашем случае это только `policy4`). Если бы у вложенного маршрута не было политик, то была бы применена `policy-2`. Это переопределение выполняет ANIC - контекст `location` для вложенного маршрута будет содержать либо политики маршрута, либо политики вложенного маршрута, но не то и другое вместе.

5.2.6 Недопустимые политики

Angie будет рассматривать политику как недействительную, если выполняется одно из следующих условий:

- Политика не проходит *всестороннюю валидацию*.
- Политика отсутствует в кластере.
- Политика не соответствует требованиям, предъявляемым к ее конкретному типу. Например, `политика ingressMTLS` требует, чтобы в VirtualServer было включено завершение TLS.

В случае недопустимой политики Angie возвращает код состояния 500 для клиентских запросов со следующими правилами:

- Если на политику ссылается `маршрут` VirtualServer или `вложенный маршрут` VirtualServerRoute, Angie будет возвращать код состояния 500 для запросов к URI такого маршрута.
- Если ссылка на политику задана в `спецификации` VirtualServer, Angie будет возвращать код состояния 500 для запросов ко всем URI этого VirtualServer.

Если политика недействительна, VirtualServer или VirtualServerRoute будет иметь *статус* с *предупреждением* о состоянии и сообщением, объясняющим, почему политика не была признана недействительной.

5.2.7 Валидация

Для ресурса Policy доступны два типа валидации:

- *Структурная валидация* с помощью `kubectl` и сервера Kubernetes API.
- *Всесторонняя валидация* с помощью ANIC.

Структурная валидация

Пользовательское определение ресурса для Policy включает структурную схему OpenAPI, которая описывает тип каждого поля ресурса.

Если вы попытаетесь создать (или обновить) ресурс, который нарушает структурную схему (например, использует строковое значение вместо массива строк в поле `allow`), то `kubectl` и сервер Kubernetes API отклонят ресурс.

- Пример проверки `kubectl`:

```
kubectl apply -f access-control-policy-allow.yaml

error: error validating "access-control-policy-allow.yaml": error validating␣
↪data: ValidationError(Policy.spec.accessControl.allow): invalid type for␣
↪software.angie.k8s.v1.Policy.spec.accessControl.allow: got "string", expected
↪"array"; if you choose to ignore these errors, turn validation off with --
↪validate=false
```

- Пример проверки сервера Kubernetes API:

```
kubectl apply -f access-control-policy-allow.yaml --validate=false

The Policy "webapp-policy" is invalid: spec.accessControl.allow: Invalid value:
↪"string": spec.accessControl.allow in body must be of type array: "string"
```

Если ресурс прошел структурную валидацию, выполняется всесторонняя валидация ANIC.

Всесторонняя валидация

ANIC проверяет поля ресурса Policy. Если ресурс недопустим, ANIC отклонит его. Ресурс останется в кластере, но ANIC будет игнорировать его.

Можно использовать `kubectl`, чтобы проверить, успешно ли ANIC применил конфигурацию Policy. Для политики `mypolicy` мы можем запустить:

```
kubectl describe pol mypolicy

. . .
Events:
  Type    Reason          Age   From                      Message
-----
Normal  AddedOrUpdated  11s   angie-ingress-controller  Policy default/mypolicy was␣
↪added or updated
```

Обратите внимание, что раздел «События» (Events) включает событие Normal с причиной AddedOrUpdated, которое информирует нас о том, что конфигурация была успешно применена.

Если вы создадите недопустимый ресурс, ANIC отклонит его и выдаст событие Rejected. Например, если вы создадите политику `mypolicy` с недопустимым IP-адресом `10.0.0.` в поле `allow`, то вы получите:

```
kubectl describe policy mypolicy
. . .
Events:
  Type     Reason    Age   From                      Message
-----
Warning  Rejected  7s    angie-ingress-controller  Policy default/mypolicy is invalid␣
↪and was rejected: spec.accessControl.allow[0]: Invalid value: "10.0.0.": must be a␣
↪CIDR or IP
```

Обратите внимание, что раздел «События» (Events) включает предупреждающее событие с указанием причины отклонения.

Кроме того, эта информация также доступна в поле `status` ресурса Policy. Обратите внимание на раздел «Статус» (Status) политики:

```
kubectl describe pol mypolicy

. . .
Status:
  Message:  Policy default/mypolicy is invalid and was rejected: spec.accessControl.
↪allow[0]: Invalid value: "10.0.0.": must be a CIDR or IP
  Reason:   Rejected
  State:    Invalid
```

Примечание

Если вы сделаете существующий ресурс недействительным, ANIC отклонит его.

ГЛАВА 6

TransportServer

Ресурс TransportServer позволяет настраивать балансировку нагрузки по протоколам TCP, UDP и TLS Passthrough. Он реализован как пользовательский ресурс.

Это справочная документация по ресурсу TransportServer.

6.1 Предварительные требования

- Для TCP и UDP ресурс TransportServer должен использоваться совместно с ресурсом `GlobalConfiguration`, который должен быть создан отдельно.
- Для TLS Passthrough обязательно включите параметр командной строки `-enable-tls-passthrough` в ANIC.

6.2 Спецификация TransportServer

Ресурс TransportServer определяет конфигурацию балансировки нагрузки для трафика TCP, UDP или TLS Passthrough. Ниже приведено несколько примеров:

- Балансировка нагрузки TCP:

```
apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
name: dns-tcp
spec:
listener:
    name: dns-tcp
    protocol: TCP
tls:
    secret: cafe-secret
upstreams:
- name: dns-app
  service: dns-service
  port: 5353
```

```
  action:
  pass: dns-app
```

- Балансировка нагрузки UDP:

```
apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
name: dns-udp
spec:
listener:
    name: dns-udp
    protocol: UDP
upstreams:
- name: dns-app
  service: dns-service
  port: 5353
  upstreamParameters:
  udpRequests: 1
  udpResponses: 1
  action:
  pass: dns-app
```

- Балансировка нагрузки TLS Passthrough:

```
apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
name: secure-app
spec:
listener:
    name: tls-passthrough
    protocol: TLS_PASSTHROUGH
host: app.example.com
upstreams:
- name: secure-app
  service: secure-app
  port: 8443
  action:
  pass: secure-app
```

Поле	Описание	Тип	Обязательно
`listener`	Прослушиватель, через который Angie будет принимать входящие соединения и датаграммы.	*Listener*	Да
`host`	Хост (доменное имя) сервера. Это должен быть допустимый поддомен, как определено в RFC 1123, например `my-app` или `hello.example.com`. Домены с подстановочными знаками, такие как `*.example.com`, не допускаются. Требуется для балансировки нагрузки TLS Passthrough.	*строка*	Нет
tls	Конфигурация завершения TLS. Не поддерживается для балансировки нагрузки TLS Passthrough.	*TLS*	Нет
`upstreams`	Список апстримов.	*[]upstream*	Да
`upstreamParameters`	Параметры апстрима.	*UpstreamPa*	Нет
`action`	Действие, выполняемое для клиентского соединения или датаграммы.	*Action*	Да
`ingressClassName`	Указывает, какой экземпляр ANIC должен обрабатывать ресурс TransportServer.	`string`	Нет
`streamSnippets`	Задает пользовательский фрагмент в контексте `stream`.	`string`	Нет
`serverSnippets`	Задает пользовательский фрагмент в контексте `server`.	`string`	Нет

6.2.1 Listener

Ссылается на прослушиватель, через который Angie будет принимать входящий трафик к TransportServer. Для TCP и UDP прослушиватель должен быть определен в ресурсе `GlobalConfiguration`. При ссылке на прослушиватель должны совпадать как имя, так и протокол. Для TLS Passthrough используйте встроенный прослушиватель с именем `tls-passthrough` и протоколом `TLS_PASSTHROUGH`.

Пример:

```
listener:
  name: dns-udp
   protocol: UDP
```

Поле	Описание	Тип	Обязательно
`name`	Имя прослушивателя.	`string`	Да
`protocol`	Протокол прослушивателя.	`string`	Да

6.2.2 TLS

Поле tls определяет конфигурацию TLS для TransportServer. Обратите внимание, что текущая реализация поддерживает завершение TLS на нескольких портах, где каждому приложению принадлежит выделенный порт. При этом ANIC завершает TLS-соединения на каждом порту, где каждое приложение использует свой собственный сертификат или ключ, и направляет соединения соответствующему приложению (сервису) на основе этого входящего порта (т. е. любое TLS-соединение независимо от настроек SNI на порту будет перенаправлено в приложение, соответствующее этому порту). Пример конфигурации показан ниже:

```
secret: cafe-secret
```

Поле	Описание	Тип	Обязательно
`secret`	Имя секрета с сертификатом TLS и ключом. Секрет должен принадлежать тому же пространству имен, что и транспортный сервер. Секрет должен иметь тип `kubernetes.io/tls` и содержать ключи с именами `tls.crt` и `tls.key`, содержащие сертификат и закрытый ключ, как описано здесь.	`string`	Нет

6.2.3 Upstream

Определяет конечное место назначения для TransportServer. Например:

```
name: secure-app
service: secure-app
port: 8443
maxFails: 3
maxConns: 100
failTimeout: 30s
loadBalancingMethod: least_conn
```

Поле	Описание	Тип	Обязательно
`name`	Имя апстрима. Это должна быть допустимая метка DNS, как определено в RFC 1035. Например, допустимы значения `hello` и `upstream-123`. Имя должно быть уникальным среди всех апстримов ресурса.	`string`	Да
`service`	Название сервиса. Сервис должен принадлежать к тому же пространству имен, что и ресурс. Если сервиса не существует, Angie предположит, что у него нет конечных точек, и будет закрывать клиентские соединения и игнорировать датаграммы.	`string`	Да
`port`	Порт службы. Если у сервиса этот порт не задан, Angie предположит, что у него нет конечных точек, и будет закрывать клиентские соединения и игнорировать датаграммы. Значение должно находиться в диапазоне `1..65535`.	`int`	Да
`maxFails`	Задает число неудачных попыток установить связь с сервером, которые должны произойти в течение времени, заданного параметром failTimeout, чтобы сервер считался недоступным. Значение по умолчанию: `1`.	`int`	Нет
`maxConns`	Задает максимальное число подключений к проксируемому серверу. Значение по умолчанию равно нулю, что означает отсутствие ограничений. Значение по умолчанию равно `0`.	`int`	Нет
`failTimeout`	Задает время, в течение которого должно произойти указанное количество неудачных попыток установить связь с сервером, чтобы считать сервер недоступным, и период времени, в течение которого сервер будет считаться недоступным. Значение по умолчанию равно `10` секундам.	`string`	Нет
`loadBalancingMethod`	Метод балансировки нагрузки между серверами апстрима. По умолчанию соединения распределяются между серверами по методу взвешенной циклической балансировки. Доступные методы и подробности смотрите в разделе Апстрим.	`string`	Нет

6.2.4 UpstreamParameters

Различные параметры апстрима:

```
upstreamParameters:
  udpRequests: 1
  udpResponses: 1
  connectTimeout: 60s
  nextUpstream: true
  nextUpstreamTimeout: 50s
  nextUpstreamTries: 1
```

Поле	Описание	Тип	Обязательно
`udpRequests`	Количество датаграмм, после получения которых следующая датаграмма от того же клиента запускает новый сеанс. См. директиву proxy_requests. Значение по умолчанию равно 0.	`int`	Нет
`udpResponses`	Количество датаграмм, ожидаемых от проксируемого сервера в ответ на клиентскую датаграмму. См. директиву proxy_responses. По умолчанию количество датаграмм не ограничено.	`int`	Нет
`connectTimeout`	Тайм-аут установки соединения с проксируемым сервером. См. директиву proxy_connect_timeout. Значение по умолчанию - 60 секунд.	`string`	Нет
`nextUpstream`	Если соединение с проксируемым сервером установить не удается, определяет, будет ли клиентское соединение передано на следующий сервер. См. директиву proxy_next_upstream. Значение по умолчанию равно `true`.	bool	Нет
`nextUpstreamTries`	Количество попыток до передачи соединения к следующему серверу. См. директиву proxy_next_upstream_tries. Значение по умолчанию равно 0.	`int`	Нет
`nextUpstreamTimeout`	Время, отведенное для передачи соединения к следующему серверу. См. директиву proxy_next_upstream_timeout. Значение по умолчанию - 0.	`string`	Нет

6.2.5 SessionParameters

Различные параметры для TCP-соединений и UDP-сеансов.

```
sessionParameters:
  timeout: 50s
```

Поле	Описание	Тип	Обязательно
`timeout`	Тайм-аут между двумя последовательными операциями чтения или записи в соединениях с клиентом или проксируемым сервером. См. директиву proxy_timeout. Значение по умолчанию равно `10m`.	`string`	Нет

6.2.6 Action

Действие, которое необходимо выполнить для клиентского соединения или датаграммы.

В приведенном ниже примере клиентские подключения и датаграммы передаются на апстрим в `dns-app`:

```
action:
  pass: dns-app
```

Поле	Описание	Тип	Обязательно
`pass`	Передает соединения и датаграммы апстриму. Апстрим с таким именем должен быть определен в ресурсе.	`string`	Да

6.3 Использование TransportServer

Для работы с ресурсами TransportServer можно использовать обычные команды `kubectl`, аналогично ресурсам Ingress.

Например, следующая команда создает ресурс TransportServer, определенный в `transport-server-passthrough.yaml`, с именем `secure-app`:

```
kubectl apply -f transport-server-passthrough.yaml

transportserver.k8s.angie.software/secure-app created
```

Вы можете получить ресурс, выполнив:

```
kubectl get transportserver secure-app

NAME         AGE
secure-app   46sm
```

В kubectl get и подобных командах также можно использовать короткое имя `ts` вместо `transportserver`.

6.3.1 Использование фрагментов

Фрагменты позволяют вставлять элементы конфигурации Angie в различные контексты конфигурации Angie. В приведенном ниже примере мы используем фрагменты для настройки контроля доступа на TransportServer:

```
apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
  name: cafe
spec:
  host: cafe.example.com
  serverSnippets: |
    deny  192.168.1.1;
    allow 192.168.1.0/24;
  upstreams:
  - name: tea
```

```
    service: tea-svc
    port: 80
```

Фрагменты также можно указать для потока. В приведенном ниже примере мы используем фрагменты для ограничения количества подключений :

```
apiVersion: k8s.angie.software/v1alpha1
kind: TransportServer
metadata:
  name: cafe
spec:
  host: cafe.example.com
  streamSnippets: limit_conn_zone $binary_remote_addr zone=addr:10m;
  serverSnippets: limit_conn addr 1;
  upstreams:
  - name: tea
    service: tea-svc
    port: 80
```

Фрагменты предназначены для продвинутых пользователей Angie, которым требуется больше контроля над генерируемой конфигурацией Angie.

Однако из-за недостатков, описанных ниже, фрагменты по умолчанию отключены. Чтобы использовать фрагменты, задайте аргумент командной строки `enable-snippets`.

Недостатки использования фрагментов:

- *Сложность*. Чтобы использовать фрагменты, требуется:
  - Понимать примитивы конфигурации Angie и реализовать правильную конфигурацию Angie.
  - Понимать, как ANIC генерирует конфигурацию Angie, чтобы фрагмент не мешал другим функциям конфигурации.
- *Сниженная надежность*. Неправильный фрагмент делает конфигурацию Angie недействительной, что приведет к ошибке при перезагрузке. Это помешает применить какие-либо обновления конфигурации, включая обновления для другого ресурса TransportServer, пока фрагмент не будет исправлен.
- *Последствия для безопасности*. Фрагменты предоставляют доступ к примитивам конфигурации Angie, и эти примитивы не проверяются самим ANIC.

Примечание

Пока конфигурация Angie содержит недопустимый фрагмент, Angie будет продолжать работать с последней допустимой конфигурацией.

Примечание

Чтобы настроить фрагменты в контексте `stream`, используйте ключ `stream-snippets` ConfigMap.

6.3.2 Валидация

Для ресурса TransportServer доступны два типа валидации:

- *Структурная валидация* с помощью `kubectl` и сервера Kubernetes API.
- *Всесторонняя валидация* с помощью ANIC.

Структурная валидация

Пользовательское определение ресурса для TransportServer включает структурную схему OpenAPI, которая описывает тип каждого поля ресурса.

Если вы попытаетесь создать (или обновить) ресурс с нарушением структурной схемы (например, используете строковое значение для поля порта апстрима), сервер `kubectl` и Kubernetes API отклонят такой ресурс:

- Пример проверки `kubectl`:

```
kubectl apply -f transport-server-passthrough.yaml

  error: error validating "transport-server-passthrough.yaml": error validating␣
↪data: ValidationError(TransportServer.spec.upstreams[0].port): invalid type␣
↪for software.angie.k8s.v1alpha1.TransportServer.spec.upstreams.port: got
↪"string", expected "integer"; if you choose to ignore these errors, turn␣
↪validation off with --validate=false
```

- Пример проверки сервера Kubernetes API:

```
kubectl apply -f transport-server-passthrough.yaml --validate=false

  The TransportServer "secure-app" is invalid: []: Invalid value:␣
↪map[string]interface {}{ ... }: validation failure list:
  spec.upstreams.port in body must be of type integer: "string"
```

Если ресурс не отклонен (то есть не нарушает структурную схему), ANIC проверит его дополнительно.

Всесторонняя валидация

ANIC проверяет поля ресурса TransportServer. Если ресурс недействителен, ANIC отклонит его: ресурс продолжит существовать в кластере, но ANIC будет его игнорировать.

Вы можете проверить, успешно ли ANIC применил конфигурацию TransportServer. Для примера TransportServer `secure-app` мы можем запустить:

```
kubectl describe ts secure-app

. . .
Events:
  Type    Reason          Age   From                      Message
-----
Normal  AddedOrUpdated  3s    angie-ingress-controller  Configuration for default/
↪secure-app was added or updated
```

Обратите внимание, что раздел «События» (Events) включает событие Normal с причиной AddedOrUpdated, которое информирует нас о том, что конфигурация была успешно применена.

Если вы создадите недопустимый ресурс, ANIC отклонит его и выдаст событие Rejected. Например, если вы создадите TransportServer `secure-app` с действием pass, которое ссылается на несуществующий апстрим, вы получите:

```
kubectl describe ts secure-app

. . .
Events:
  Type     Reason    Age   From                      Message
-----
Warning  Rejected  2s    angie-ingress-controller  TransportServer default/secure-app␣
↪is invalid and was rejected: spec.action.pass: Not found: "some-app"
```

Обратите внимание, что раздел событий включает событие Warning с причиной Rejected.

Примечание

Если вы внесете ошибку в уже существующий ресурс, контроллер входа отклонит его и удалит соответствующую конфигурацию из Angie.

6.3.3 Настройка с помощью ConfigMap

Ключи `ConfigMap` (за исключением `stream-snippets`, `stream-log-format`, `resolver-addresses`, `resolver-ipv6`, `resolver-valid` и `resolver-timeout`) не влияют на ресурсы TransportServer.

ГЛАВА 7

VirtualServer, VirtualServerRoute

Ресурсы VirtualServer и VirtualServerRoute, представленные в версии 1.5, реализуют сценарии использования, не поддерживаемые ресурсом Ingress, такие как разделение трафика и продвинутая маршрутизация на основе содержимого. Они реализованы как пользовательские ресурсы .

Это справочная документация по обоим ресурсам.

7.1 Спецификация VirtualServer

Ресурс VirtualServer определяет конфигурацию балансировки нагрузки для доменного имени, например `example.com`. Ниже приведен пример такой конфигурации:

```
apiVersion: k8s.angie.software/v1
kind: VirtualServer
metadata:
  name: cafe
spec:
  host: cafe.example.com
  tls:
    secret: cafe-secret
  gunzip: on
  upstreams:
  - name: tea
    service: tea-svc
    port: 80
  - name: coffee
    service: coffee-svc
    port: 80
  routes:
  - path: /tea
    action:
      pass: tea
  - path: /coffee
    action:
      pass: coffee
  - path: ~ ^/decaf/.\*\\.jpg$
```

```
    action:
      pass: coffee
  - path: = /green/tea
    action:
      pass: tea
```

Поле	Описание	Тип	Обязательно
`host`	Хост (доменное имя) сервера. Это должен быть допустимый поддомен, как определено в RFC 1123, например `my-app` или `hello.example.com`. При использовании домена с подстановочным знаком, например `*.example.com`, домен должен быть заключен в двойные кавычки. Значение `host` должно быть уникальным среди всех ресурсов Ingress и VirtualServer.	`string`	Да
`tls`	Конфигурация завершения TLS.	*tls*	Нет
`gunzip`	Включает или отключает распаковку архивированных ответов для клиентов. Допустимые пары значений: “on” и “off”, “true” и “false” или “yes” и “no”. Если значение `gunzip` не установлено, то по умолчанию оно равно `off`.	`boolean`	Нет
`ExternalDNS`	Конфигурация ExternalDNS для VirtualServer.	*ExternalDNS*	Нет
`dos`	Ссылка на DosProtectedResource; установка этого параметра включает защиту VirtualServer от DOS-атак.	`string`	Нет
`policies`	Список политик.	*[]policy*	Нет
`upstreams`	Список апстримов.	*[]upstream*	Нет
`routes`	Список маршрутов.	*[]route*	Нет
`ingressClassName`	Указывает, какой экземпляр ANIC должен обрабатывать ресурс VirtualServer.	`string`	Нет
`internalRoute`	Указывает, является ли ресурс VirtualServer внутренним маршрутом.	`boolean`	Нет
`http-snippets`	Задает пользовательский фрагмент в контексте http.	`string`	Нет
`server-snippets`	Задает пользовательский фрагмент в контексте . Имеет приоритет над ключом ConfigMap `server-snippets`.	`string`	Нет

7.1.1 VirtualServer.TLS

Поле tls определяет конфигурацию TLS для ресурса VirtualServer. Например:

```
secret: cafe-secret
redirect:
  enable: true
```

Поле	Описание	Тип	Обязательно
`secret`	Имя секрета с сертификатом TLS и ключом. Секрет должен принадлежать тому же пространству имен, что и VirtualServer. Секрет должен иметь тип `kubernetes.io/tls` и содержать ключи с именами `tls.crt` и `tls.key`, содержащие сертификат и закрытый ключ, как описано здесь. Если секрет не существует или недействителен, Angie прервет любую попытку установить TLS-соединение с хостом VirtualServer. Если секрет не указан, но настроен секрет TLS с подстановочным знаком, Angie будет использовать секрет со знаком для завершения TLS.	`string`	Нет
`redirect`	Конфигурация перенаправления TLS для VirtualServer.	*tls.redirect*	Нет
`cert-manager`	Конфигурация TLS cert-manager для VirtualServer.	*tls.cert-manager*	Нет

7.1.2 VirtualServer.TLS.Redirect

Поле перенаправления настраивает перенаправление TLS для VirtualServer:

```
enable: true
code: 301
basedOn: scheme
```

Поле	Описание	Тип	Обязательно
`enable`	Включает перенаправление TLS для VirtualServer. Значение по умолчанию равно `false`.	`boolean`	Нет
`код`	Код состояния перенаправления. Допустимые значения: 301, 302, 307, 308. Значение по умолчанию - 301.	`int`	Нет
`basedOn`	Атрибут запроса, который Angie будет оценивать для отправки перенаправления. Допустимыми значениями являются `scheme` (схема запроса) или `x-forwarded-proto` (заголовок `X-Forwarded-Proto` запроса). Значение по умолчанию - `scheme`.	`string`	Нет

7.1.3 VirtualServer.TLS.CertManager

Поле cert-manager настраивает автоматическое управление сертификатами x509 для ресурсов VirtualServer с помощью cert-manager (cert-manager.io). Ознакомьтесь с документацией по конфигурации cert-manager для получения дополнительной информации о развертывании и настройке эмитентов (Issuer). Пример:

```
cert-manager:
  cluster-issuer: "my-issuer-name"
```

Поле	Описание	Тип	Обязательно
`issuer`	Имя эмитента. Эмитент - это ресурс cert-manager, который описывает центр сертификации, способный подписывать сертификаты. Он должен находиться в том же пространстве имен, что и ресурс VirtualServer. Обратите внимание, что требуется задать *issuer* или *cluster-issuer*, но эти параметры взаимоисключающие - должен быть задан один и только один.	`string`	Нет
`cluster-issuer`	Имя ClusterIssuer. ClusterIssuer - это ресурс cert-manager, который описывает центр сертификации, способный подписывать сертификаты. Не имеет значения, в каком пространстве имен находится ваш VirtualServer, поскольку ClusterIssuer - это ресурсы, не относящиеся к пространствам имен. Обратите внимание, что требуется задать *issuer* или *cluster-issuer*, но эти параметры взаимоисключающие - должен быть задан один и только один.	`string`	Нет
`issuer-kind`	Тип внешнего ресурса-эмитента, например AWSPCAIssuer. Это необходимо только для сторонних эмитентов. Его нельзя задавать, если также задан *cluster-issuer*.	`string`	Нет
`issuer-group`	Группа API внешнего контроллера-эмитента, например awspca.cert-manager.io. Это необходимо только для сторонних эмитентов. Его нельзя задавать, если также задан *cluster-issuer*.	`string`	Нет
`common-name`	Это поле позволяет настроить spec.commonName для создаваемого сертификата. Эта конфигурация добавляет CN к сертификату x509.	`string`	Нет
`duration`	Это поле позволяет настроить поле spec.duration для генерируемого сертификата. Оно должно быть задано с использованием формата time.Duration в Go, который не допускает суффикса d (дни). Указывайте такие значения, используя вместо них суффиксы s, m и h.	`string`	Нет
`renew-before`	Эта аннотация позволяет настроить поле spec.renewBefore для генерируемого сертификата. Оно должно быть задано с использованием формата time.Duration в Go, который не допускает суффикса d (дни). Указывайте такие значения, используя вместо них суффиксы s, m и h.	`string`	Нет
`usages`	Позволяет настроить поле spec.usages для генерируемого сертификата. Задайте строку со значениями, разделенными запятыми, т. е. `соглашение о ключе, цифровая подпись, серверная аутентификация`. Исчерпывающий список поддерживаемых способов использования ключей можно найти в документации API cert-manager.	`string`	Нет

7.1.4 VirtualServer.ExternalDNS

Поле ExternalDNS настраивает динамическое управление записями DNS для ресурсов VirtualServer с использованием ExternalDNS . Ознакомьтесь с документацией по конфигурации ExternalDNS для получения дополнительной информации о развертывании и настройке ExternalDNS и поставщиков. Пример:

```
enable: true
```

Поле	Описание	Тип	Обязательно
`enable`	Включает интеграцию ExternalDNS для ресурса VirtualServer. Значение по умолчанию равно `false`.	`string`	Нет
`labels`	Настраивает метки, применяемые к ресурсам конечной точки, которые будут использоваться ExternalDNS.	`map[string]`	Нет
`providerSpecific`	Настраивает свойства, относящиеся к конкретному поставщику, которые содержат имя и значение конфигурации, специфичной для отдельных поставщиков DNS.	*[]ProviderSp*	Нет
`recordTTL`	TTL для записи DNS. По умолчанию это значение равно 0. См. документацию ExternalDNS TTL для определения значений по умолчанию для конкретного поставщика	`int64`	Нет
`recordType`	Тип создаваемой записи, например «A», «AAAA», «CNAME». Если значение не задано, оно автоматически вычисляется на основе внешних конечных точек.	`string`	Нет

7.1.5 VirtualServer.ExternalDNS.ProviderSpecific

Поле providerSpecific блока ExternalDNS позволяет указать свойства, специфичные для поставщика, которые представляют собой список пар «ключ-значение» для конфигураций, специфичных для отдельных поставщиков DNS. Пример:

```
- name: my-name
  value: my-value
- name: my-name2
  value: my-value2
```

Поле	Описание	Тип	Обязательно
`name`	Имя в паре «ключ-значение».	`string`	Да
`value`	Значение в паре «ключ-значение».	`string`	Да

7.1.6 VirtualServer.Policy

Ссылается на ресурс Policy по имени и необязательному пространству имен. Например:

```
name: access-control
```

Поле	Описание	Тип	Требуется
`name`	Имя политики. Если политика не существует или недействительна, Angie выдаст сообщение об ошибке с кодом состояния `500`.	`strir`	Да
`namesp`	Пространство имен политики. Если не указано, используется пространство имен ресурса VirtualServer.	`strir`	Нет

7.1.7 VirtualServer.Route

Маршрут определяет правила для сопоставления клиентских запросов с такими действиями, как передача запроса апстриму. Например:

```
path: /tea
action:
  pass: tea
```

Поле	Описание	Тип	Обязательно
`path`	Путь маршрута. Angie сопоставит его с URI запроса. Возможные значения: префикс (`/`, `/path`), точное совпадение (`=/exact/match`), регулярное выражение без учета регистра (`~*^/Bar.*\.jpg`) или регулярное выражение с учетом регистра (`~^/foo.*\.jpg`). В случае префикса (должен начинаться с `/`) или точного совпадения (должно начинаться с `=`) путь не должен содержать никаких пробельных символов, `{` , `}` или `;`. В случае регулярных выражений все двойные кавычки `"` должны быть экранированы, при этом совпадение не может заканчиваться неэкранированной обратной косой чертой \. Путь должен быть уникальным среди путей всех маршрутов VirtualServer. Дополнительные сведения см. в описании директивы location.	`string`	Да
`policies`	Список политик. Эти политики имеют приоритет над политиками того же типа, определенными в `спецификации` VirtualServer. Более подробную информацию смотрите в *Применение политик*.	*[]policy*	Нет
`action`	Действие по умолчанию, выполняемое для запроса.	*Action*	Нет
`dos`	Ссылка на DosProtectedResource; установка этого параметра включает защиту маршрута VirtualServer от DOS-атак.	`string`	Нет
`splits`	Конфигурация разделения трафика по умолчанию. Должно быть не менее 2 разделений.	*Split*	Нет
`matches`	Правила сопоставления для продвинутой маршрутизации на основе содержимого. Требуется задать `action` или `splits` по умолчанию. Несопоставленные запросы будут обрабатываться `action` или `splits` по умолчанию.	*matches*	Нет
`route`	Имя ресурса VirtualServerRoute, который определяет этот маршрут. Если VirtualServerRoute не принадлежит к тому же пространству имен, что и VirtualServer, необходимо включить пространство имен. Например: `tea-namespace/tea`.	`string`	Нет
`errorPages`	Настраиваемые ответы на коды ошибок. Angie будет использовать эти ответы вместо того, чтобы возвращать ответы об ошибках с серверов апстрима или ответы по умолчанию, сгенерированные Angie. Настраиваемый ответ может быть перенаправлением или сохраненным ответом. Например, это может быть перенаправление на другой URL-адрес, если вышестоящий сервер ответил кодом состояния 404.	*[]errorPage*	Нет
`location-snippets`	Задает пользовательский фрагмент в контексте местоположения. Имеет приоритет над ключом ConfigMap `location-snippets`.	`string`	Нет

Примечание

Маршрут должен включать в себя ровно одно из следующих действий: `action`, `splits` или `route`.

7.2 Спецификация VirtualServerRoute

Ресурс VirtualServerRoute определяет маршрут для VirtualServer. Он может состоять из одного вложенного маршрута или нескольких. VirtualServerRoute является альтернативой объединяемым типам Ingress.

В приведенном ниже примере виртуальный сервер `cafe` из пространства имен `cafe-ns` определяет маршрут с путем `/coffee`, который далее определяется через VirtualServerRoute `coffee` из пространства имен `coffee-ns`.

VirtualServer:

```
apiVersion: k8s.angie.software/v1
kind: VirtualServer
metadata:
  name: cafe
  namespace: cafe-ns
spec:
  host: cafe.example.com
  upstreams:
  - name: tea
    service: tea-svc
    port: 80
  routes:
  - path: /tea
    action:
      pass: tea
  - path: /coffee
    route: coffee-ns/coffee
```

VirtualServerRoute:

```
apiVersion: k8s.angie.software/v1
kind: VirtualServerRoute
metadata:
  name: coffee
  namespace: coffee-ns
spec:
  host: cafe.example.com
  upstreams:
  - name: latte
    service: latte-svc
    port: 80
  - name: espresso
    service: espresso-svc
    port: 80
  subroutes:
  - path: /coffee/latte
    action:
      pass: latte
  - path: /coffee/espresso
```

```
    action:
      pass: espresso
```

Обратите внимание, что каждый вложенный маршрут должен иметь путь `path`, начинающийся с того же префикса (здесь «/coffee»), что и в маршруте VirtualServer. Кроме того, `host` в VirtualServerRoute должен совпадать с `host` у VirtualServer.

Поле	Описание	Тип	Обязательно
`host`	Хост (доменное имя) сервера. Это должен быть допустимый поддомен, как определено в RFC 1123, например `my-app` или `hello.example.com`. При использовании домена с подстановочным знаком, например `*.example.com`, домен должен быть заключен в двойные кавычки. Значение должно совпадать с `host` у VirtualServer, который ссылается на этот ресурс.	`string`	Да
`upstreams`	Список апстримов.	*[]upstream*	Нет
`subroutes`	Список вложенных маршрутов.	*[]subroute*	Нет
`ingressClassName`	Указывает, какой экземпляр ANIC должен обрабатывать ресурс VirtualServerRoute. Значение должно совпадать с `ingressClassName` у VirtualServer, который ссылается на этот ресурс.	`string`	Нет

7.2.1 VirtualServerRoute.Subroute

Определяет правила сопоставления клиентских запросов и действий, например передача запроса апстриму. Например:

```
path: /coffee
action:
  pass: coffee
```

Поле	Описание	Тип	Обязательно
`path`	Путь вложенного маршрута. Angie сопоставит его с URI запроса. Возможные значения: префикс (`/`, `/path`), точное совпадение (`=/exact/match`), регулярное выражение без учета регистра (`~*^/Bar.*\.jpg`) или регулярное выражение с учетом регистра (`~^/foo.*\.jpg`). В случае префикса путь должен начинаться с того же пути, что и путь маршрута VirtualServer, который ссылается на этот ресурс. В случае точного совпадения или регулярного выражения путь должен совпадать с путем маршрута VirtualServer, который ссылается на этот ресурс. В случае префикса или точного совпадения путь не должен содержать никаких пробельных символов, `{` , `}` или `;`. В случае регулярных выражений все двойные кавычки `"` должны быть экранированы, при этом совпадение не может заканчиваться неэкранированной обратной косой чертой . Путь должен быть уникальным среди путей всех вложенных маршрутов VirtualServerRoute.	`string`	Да
`policies`	Список политик. Эти политики имеют приоритет над *всеми* политиками, определенными в маршруте VirtualServer, который ссылается на этот ресурс. Они также имеют приоритет над политиками того же типа, определенными в `спецификации` VirtualServer. Более подробную информацию смотрите в разделе *Применение политик*.	*[]policy*	Нет
`action`	Действие по умолчанию, выполняемое для запроса.	*Action*	Нет
`dos`	Ссылка на DosProtectedResource; установка этого параметра включает защиту вложенного маршрута VirtualServerRoute от DOS-атак.	`string`	Нет
`splits`	Конфигурация разделения трафика по умолчанию. Должно быть не менее 2 разделений.	*[]split*	Нет
`matches`	Правила сопоставления для продвинутой маршрутизации на основе содержимого. Требуется задать `action` или `splits` по умолчанию. Несопоставленные запросы будут обрабатываться `action` или `splits` по умолчанию.	*matches*	Нет
`errorPages`	Настраиваемые ответы на коды ошибок. Angie будет использовать эти ответы вместо того, чтобы возвращать ответы об ошибках с серверов апстрима или ответы по умолчанию, сгенерированные Angie. Настраиваемый ответ может быть перенаправлением или сохраненным ответом. Например, это может быть перенаправление на другой URL-адрес, если вышестоящий сервер ответил кодом состояния 404.	*[]errorPage*	Нет
`location-snippets`	Задает пользовательский фрагмент в контексте местоположения. Переопределяет значение `location-snippets` VirtualServer (если задано) или ключ ConfigMap `location-snippets`.	`string`	Нет

Примечание

Вложенный маршрут должен включать в себя ровно одно из следующих действий: `action` или `splits`.

7.3 Общие части VirtualServer и VirtualServerRoute

7.3.1 Upstream

Апстрим определяет конечное место назначения для конфигурации маршрутизации. Например:

```
name: tea
service: tea-svc
subselector:
  version: canary
port: 80
lb-method: round_robin
fail-timeout: 10s
max-fails: 1
max-conns: 32
keepalive: 32
connect-timeout: 30s
read-timeout: 30s
send-timeout: 30s
next-upstream: "error timeout non_idempotent"
next-upstream-timeout: 5s
next-upstream-tries: 10
client-max-body-size: 2m
tls:
  enable: true
```

Примечание

Протокол WebSocket поддерживается без какой-либо дополнительной настройки.

Поле	Описание	Тип	Обязательно
`name`	Имя апстрима. Это должна быть допустимая метка DNS, как определено в RFC 1035. Например, допустимы значения `hello` и `upstream-123`. Имя должно быть уникальным среди всех апстримов ресурса.	`string`	Да
`service`	Название сервиса. Сервис должен принадлежать к тому же пространству имен, что и ресурс. Если сервиса не существует, Angie предположит, что у него нет конечных точек, и будет возвращать ответ 502 для запросов к этому апстриму.	`string`	Да
`subselector`	Выбирает поды внутри сервиса, используя ключи меток и значения. По умолчанию выбраны все поды сервиса. **Примечание** Ожидается, что указанные метки будут присутствовать в подах при их создании. Если метки подов изменяются, ANIC не увидит это изменение до тех пор, пока не будет изменено количество подов.	`map[string]`	Нет
`use-cluster-ip`	Позволяет использовать IP-адрес кластера и порт сервиса вместо использования IP-адреса и порта подов по умолчанию. Когда это поле включено, поля, которые настраивают поведение Angie, относящееся к нескольким серверам апстрима (например, `lb-method` и `next-upstream`), не будут иметь никакого эффекта, поскольку ANIC настроит Angie только с одним сервером апстрима, который будет соответствовать IP-адресу кластера сервиса.	`boolean`	Нет
`port`	Порт службы. Если у сервиса не определен этот порт, Angie предположит, что у него нет конечных точек, и будет возвращать ответ 502 для запросов к этому апстриму. Значение должно находиться в диапазоне `1..65535`.	`uint16`	Да
`lb-method`	Метод балансировки нагрузки. Чтобы использовать циклический метод, укажите `round_robin`. Значение по умолчанию указано в ключе `lb-method` ConfigMap.	`string`	Нет
`fail-timeout`	Время, в течение которого должно произойти указанное количество неудачных попыток установить связь с сервером апстрима, чтобы он считался недоступным. См. параметр fail_timeout директивы сервера. Значение по умолчанию задано в ключе ConfigMap `fail-timeout`.	`string`	Нет
`max-fails`	Количество неудачных попыток установить связь с сервером апстрима, которые должны произойти в течение времени, заданного в `fail-timeout`, чтобы считать сервер недоступным. См. параметр max_fails директивы сервера. Значение по умолчанию задано в ключе ConfigMap `max-fails`.	`int`	Нет
`max-conns`	Максимальное количество одновременных активных подключений к серверу апстрима. См. параметр max_conns директивы сервера. По	`int`	Нет

7.3.2 Upstream.Buffers

Настраивает буферы, используемые для чтения ответа от сервера апстрима в рамках одного соединения.

```
number: 4
size: 8K
```

См. директиву proxy_buffers для получения дополнительной информации.

Поле	Описание	Тип	Требуется
`number`	Задает количество буферов. Значение по умолчанию задано в ключе ConfigMap `proxy-buffers`.	`int`	Да
`size`	Задает размер буфера. Значение по умолчанию задано в ключе ConfigMap `proxy-buffers`.	`string`	Да

7.3.3 Upstream.TLS

Поле	Описание	Тип	Обязательно
`enable`	Включает HTTPS для запросов к серверам апстрима. Значение по умолчанию равно `False`, что означает, что будет использоваться HTTP. **Примечание** По умолчанию Angie не будет проверять сертификат вышестоящего сервера. Чтобы включить проверку, настройте политику EgressMTLS.	`boolean`	Нет

7.3.4 Upstream.SessionCookie

Поле SessionCookie настраивает сохранение сеансов, что позволяет передавать запросы от одного и того же клиента на один и тот же сервер апстрима. Информация о назначенном сервере апстрима передается в сеансовом cookie, сгенерированном Angie.

В приведенном ниже примере мы настраиваем сохранение сеанса с помощью cookie сеанса для апстрима и задаем все доступные параметры:

```
name: tea
service: tea-svc
port: 80
sessionCookie:
  enable: true
  name: srv_id
  path: /
  expires: 1h
  domain: .example.com
  httpOnly: false
```

```
  secure: true
  samesite: strict
```

См. директиву sticky для получения дополнительной информации. Сеансовый cookie соответствует методу `sticky cookie`.

Поле	Описание	Тип	Обязательно
`enable`	Включает сохранение сеанса с помощью сеансового cookie для сервера апстрима. Значение по умолчанию равно `false`.	`boolean`	Нет
`name`	Имя cookie.	`string`	Да
`path`	Путь, для которого установлен cookie.	`string`	Нет
`expires`	Время, в течение которого браузер должен сохранять cookie. Может быть установлено специальное значение `max`; тогда срок действия cookie истечет `31 декабря 2037 года в 23:55:55 по Гринвичу`.	`string`	Нет
`domain`	Домен, для которого установлен cookie.	`string`	Нет
`httpOnly`	Добавляет атрибут `HttpOnly` к cookie.	`boolean`	Нет
`secure`	Добавляет атрибут `Secure` к cookie.	`boolean`	Нет
`samesite`	Добавляет атрибут `SameSite` к cookie. Допустимые значения: `strict`, `lax`, `none`	`string`	Нет

7.3.5 Upstream.SessionRoute

Поле sessionRoute настраивает сохранение маршрутов, что позволяет передавать запросы от одного и того же клиента на один и тот же сервер апстрима. Информация о назначенном сервере апстрима поддерживается в режиме route Angie.

В приведенном ниже примере мы формируем директиву Angie `sticky route $cookie_route $arg_route;`:

```
sessionRoute:
  enable: true
  variables:
    \- "\$cookie_route"
    \- "\$arg_route"
```

См. описание режима `route` директивы sticky для получения дополнительной информации.

Параметры:

Поле	Описание	Тип	Обязательно
`enable`	Включает сохранение сеанса в режиме `route` для сервера апстрима. Значение по умолчанию равно `false`.	`boolean`	Нет
`variable`	Список переменных, подставляемых в директиву `sticky route` в порядке следования.	`[]string`	Нет

7.3.6 Header

Определяет HTTP-заголовок:

```
name: Host
value: example.com
```

Поле	Описание	Тип	Требуется
`name`	Имя заголовка.	`string`	Да
`value`	Значение заголовка.	`string`	Нет

7.3.7 Action

Определяет действие, которое необходимо выполнить для запроса.

В приведенном ниже примере клиентские запросы передаются на апстрим `coffee`:

```
path: /coffee
action:
 pass: coffee
```

Поле	Описание	Тип	Обязательно
`pass`	Передает запросы серверу апстрима. Апстрим с таким именем должен быть определен в ресурсе.	`string`	Нет
`redirect`	Перенаправляет запросы на указанный URL-адрес.	*action.redire*	Нет
`return`	Возвращает предварительно сконфигурированный ответ.	*action.return*	Нет
`proxy`	Передает запросы апстриму, добавляет возможность изменять запрос и ответ (например, переписывать URI или изменять заголовки).	*action.proxy*	Нет

Примечание

Действие должно включать в себя ровно одно из следующих значений: `pass`, `redirect`, `return` или `proxy`.

7.3.8 Action.Redirect

Определяет перенаправление, возвращаемое для запроса.

В приведенном ниже примере клиентские запросы направляются на URL-адреса `http://myhost.ru`:

```
redirect:
  url: http://myhost.ru
  code: 301
```

Поле	Описание	Тип	Требуется
`url`	URL-адрес, на который будет перенаправлен запрос. Поддерживаемые переменные Angie: `$scheme` , `$http_x_forwarded_proto` , `$request_uri` , `$host`. Переменные должны быть заключены в фигурные скобки. Например: `${host}${request_uri}`.	`stri`	Да
`код`	Код состояния перенаправления. Допустимые значения: `301`, `302`, `307`, `308`. Значение по умолчанию - `301`.	`int`	Нет

7.3.9 Action.Return

Определяет предварительно сконфигурированный ответ на запрос.

В приведенном ниже примере Angie будет отвечать предварительно настроенным ответом на каждый запрос:

```
return:
  code: 200
  type: text/plain
  body: "Hello World\n"
```

Поле	Описание	Тип	Требуется
`код`	Код состояния ответа. Допустимые значения: `2XX`, `4XX` или `5XX`. Значение по умолчанию равно 200.	`int`	Нет
`тип`	MIME-тип ответа. Значение по умолчанию - `text/plain`.	`stri`	Нет
`body`	Основная часть ответа. Поддерживает переменные Angie*. Переменные должны быть заключены в фигурные скобки. Например: `Зап рос равен ${request_uri}\n`.	`stri`	Да

Примечание

Поддерживаемые переменные Angie: `$request_uri`, `$request_method`, `$request_body`, `$scheme`, `$http_`, `$args`, `$arg_`, `$cookie_`, `$host`, `$request_time`, `$request_length`, `$angie_version`, `$pid`, `$connection`, `$remote_addr`, `$remote_port`, `$time_iso8601`, `$time_local`, `$server_addr`, `$server_port`, `$server_name`, `$server_protocol`, `$connections_active`, `$connections_reading`, `$connections_writing` и `$connections_waiting`.

7.3.10 Action.Proxy

Передает запросы апстриму с возможностью изменять запрос и ответ (например, переписывать URI или изменять заголовки).

В приведенном ниже примере URI запроса переписывается на `/`, а заголовки запроса и ответа изменяются:

```
proxy:
  upstream: coffee
  requestHeaders:
```

```
    pass: true
    set:
    - name: My-Header
      value: Value
    - name: Client-Cert
      value: ${ssl_client_escaped_cert}
  responseHeaders:
    add:
    - name: My-Header
      value: Value
    - name: IC-Angie-Version
      value: ${angie_version}
      always: true
    hide:
    - x-internal-version
    ignore:
    - Expires
    - Set-Cookie
    pass:
    - Server
  rewritePath: /
```

Поле	Описание	Тип	Обязательно
`upstream`	Имя апстрима, куда будут проксироваться запросы. Апстрим с таким именем должен быть определен в ресурсе.	`string`	Да
`requestHeaders`	Изменения заголовков запросов.	*Action.Proxy*	Нет
`responseHeaders`	Изменения в заголовках ответов.	*Action.Proxy*	Нет
`rewritePath`	Переписанный URI. Если путь маршрута является регулярным выражением, т. е. начинается с `~`, то *rewritePath* может включать группы захвата `$1-9`. Например, *$1* - первая группа, и так далее.	`string`	Нет

7.3.11 Action.Proxy.RequestHeaders

Поле requestHeaders изменяет заголовки запроса к проксируемому серверу апстрима.

Поле	Описание	Тип	Обязательно
`pass`	Передает исходные заголовки запроса на проксируемый сервер апстрима. Дополнительные сведения см. в описании директивы proxy_pass_request_header. Значение по умолчанию - true.	`bool`	Нет
`set`	Позволяет переопределять или добавлять поля для представления заголовков запросов, передаваемых на проксируемые серверы апстрима. Дополнительные сведения см. в описании директивы proxy_set_header.	*[]header*	Нет

7.3.12 Action.Proxy.RequestHeaders.Set.Header

Определяет HTTP-заголовок:

```
name: My-Header
value: My-Value
```

Можно переопределить значение заголовка `Host` по умолчанию, которое ANIC устанавливает равным $host:

```
name: Host
value: example.com
```

Поле	Описание	Тип	Требуется
`name`	Имя заголовка.	`stri`	Да
`valu`	Значение заголовка. Поддерживает переменные Angie*. Переменные должны быть заключены в фигурные скобки. Например: `${scheme}`.	`stri`	Нет

Примечание

Поддерживаемые переменные Angie: `$request_uri`, `$request_method`, `$request_body`, `$scheme`, `$http_`, `$args`, `$arg_`, `$cookie_`, `$host`, `$request_time`, `$request_length`, `$angie_version`, `$pid`, `$connection`, `$remote_addr`, `$remote_port`, `$time_iso8601`, `$time_local`, `$server_addr`, `$server_port`, `$server_name`, `$server_protocol`, `$connections_active`, `$connections_reading`, `$connections_writing`, `$connections_waiting`, `$ssl_cipher`, `$ssl_ciphers`, `$ssl_client_cert`, `$ssl_client_escaped_cert`, `$ssl_client_fingerprint`, `$ssl_client_i_dn`, `$ssl_client_i_dn_legacy`, `$ssl_client_raw_cert`, `$ssl_client_s_dn`, `$ssl_client_s_dn_legacy`, `$ssl_client_serial`, `$ssl_client_v_end`, `$ssl_client_v_remain`, `$ssl_client_v_start`, `$ssl_client_verify`, `$ssl_curves`, `$ssl_early_data`, `$ssl_protocol`, `$ssl_server_name`, `$ssl_session_id`, `$ssl_session_reused`.

7.3.13 Action.Proxy.ResponseHeaders

Поле responseHeaders изменяет заголовки ответа клиенту.

Поле	Описание	Тип	Обязательно
`hide`	Заголовки, которые не будут переданы в ответе клиенту с проксируемого сервера апстрима. Дополнительные сведения см. в описании директивы proxy_hide_header.	`[]string`	Нет
`pass`	Позволяет передавать скрытые поля заголовка клиенту с проксируемого сервера апстрима. Дополнительные сведения см. в описании директивы proxy_pass_header.	`[]string`	Нет
`ignore`	Отключает обработку определенных заголовков** при передаче клиенту ответа с проксируемого сервера апстрима. Дополнительные сведения см. в описании директивы proxy_ignore_headers.	`[]string`	Нет
`add`	Добавляет заголовки к ответу для клиента.	*[]addHeader*	Нет

Примечание

Скрытые заголовки по умолчанию: `Date`, `Server`, `X-Pad` и `X-Accel-...`.

Примечание

Следующие поля могут быть проигнорированы: `X-Accel-Redirect`, `X-Accel-Expires`, `X-Accel-Limit-Rate`, `X-Accel-Buffering`, `X-Accel-Charset`, `Expires`, `Cache-Control`, `Set-Cookie` и `Vary`.

7.3.14 AddHeader

Определяет HTTP-заголовок с необязательным полем `always`:

```
name: My-Header
value: My-Value
always: true
```

Поле	Описание	Тип	Обязательно
`name`	Имя заголовка.	`string`	Да
`value`	Значение заголовка. Поддерживает переменные Angie*. Переменные должны быть заключены в фигурные скобки. Например: `${scheme}`.	`string`	Нет
`always`	Если установлено значение true, добавляет заголовок независимо от кода состояния ответа**. Значение по умолчанию - false. Дополнительные сведения см. в описании директивы add_header.	`bool`	Нет

Примечание

Поддерживаемые переменные Angie: `$request_uri`, `$request_method`, `$request_body`, `$scheme`, `$http_`, `$args`, `$arg_`, `$cookie_`, `$host`, `$request_time`, `$request_length`, `$angie_version`, `$pid`, `$connection`, `$remote_addr`, `$remote_port`, `$time_iso8601`, `$time_local`, `$server_addr`, `$server_port`, `$server_name`, `$server_protocol`, `$connections_active`, `$connections_reading`, `$connections_writing`, `$connections_waiting`, `$ssl_cipher`, `$ssl_ciphers`, `$ssl_client_cert`, `$ssl_client_escaped_cert`, `$ssl_client_fingerprint`, `$ssl_client_i_dn`, `$ssl_client_i_dn_legacy`, `$ssl_client_raw_cert`, `$ssl_client_s_dn`, `$ssl_client_s_dn_legacy`, `$ssl_client_serial`, `$ssl_client_v_end`, `$ssl_client_v_remain`, `$ssl_client_v_start`, `$ssl_client_verify`, `$ssl_curves`, `$ssl_early_data`, `$ssl_protocol`, `$ssl_server_name`, `$ssl_session_id`, `$ssl_session_reused`.

Примечание

Если значение `always` - false, заголовок ответа добавляется только в том случае, если код состояния ответа - это 200, 201, 204, 206, 301, 302, 303, 304, 307 или 308.

7.3.15 Split

Определяет вес действия в составе конфигурации разделений.

В приведенном ниже примере Angie передает 80% запросов вышестоящему `coffee-v1`, а оставшиеся 20% - `coffee-v2`:

```
splits:
  - weight: 80
    action:
      pass: coffee-v1
  - weight: 20
    action:
      pass: coffee-v2
```

Поле	Описание	Тип	Обязательно
`weight`	Вес действия. Значение должно попадать в диапазон `1..99`. Сумма весов всех разделений должна быть равна `100`.	`int`	Да
`action`	Действие, которое необходимо выполнить для запроса.	:ref`:action`	Да

7.3.16 Match

Определяет сопоставление между условиями и действием или разделениями.

В приведенном ниже примере Angie направляет запросы с путем «/coffee» в разные апстримы на основе значения cookie `user`:

- `user=john` -> `coffee-future`
- `user=bob` -> `coffee-deprecated`
- Если cookie не установлен или не равен ни `john`, ни `bob`, Angie перенаправляет запрос в `coffee-stable`

```
path: /coffee
matches:
- conditions:
  - cookie: user
    value: john
    action:
    pass: coffee-future
- conditions:
  - cookie: user
    value: bob
    action:
      pass: coffee-deprecated
  action:
    pass: coffee-stable
```

В следующем примере Angie направляет запросы на основе значения встроенной переменной $request_method , которая представляет HTTP-метод запроса:

- все запросы POST -> `coffee-post`
- все прочие запросы -> `coffee`

```
path: /coffee
matches:
  - conditions:
    - variable: $request\_method
        value: POST
      action:
        pass: coffee-post
    action:
      pass: coffee
```

Поле	Описание	Тип	Обязательно
`conditions`	Список условий. Должен включать по крайней мере одно условие.	*[]condition*	Да
`action`	Действие, которое необходимо выполнить для запроса.	*Action*	Нет
`splits`	Конфигурация разбиений для разделения трафика. Должно быть указано не менее двух разделений.	*[]split*	Нет

Примечание

Сопоставление должно использовать ровно одно из следующих значений: `action` или `splits`.

7.3.17 Condition

Определяет условие в сопоставлении.

Поле	Описание	Тип	Обязательно
`header`	Имя заголовка. Должно состоять из буквенно-цифровых символов или `-`.	`string`	Нет
`cookie`	Имя cookie. Должно состоять из буквенно-цифровых символов или `_`.	`string`	Нет
`argument`	Имя аргумента. Должно состоять из буквенно-цифровых символов или `_`.	`string`	Нет
`variable`	Имя переменной Angie. Должно начинаться с `$`. См. список поддерживаемых переменных после таблицы.	`string`	Нет
`value`	Значение, которому должно соответствовать условие. Как определить значение, показано ниже в таблице.	`string`	Да

Примечание

Условие должно включать ровно одно из следующих значений: `header`, `cookie`, `argument` или `variable`.

Поддерживаемые переменные Angie: `$args`, `$http2`, `$https`, `$remote_addr`, `$remote_port`, `$query_string`, `$request`, `$request_body`, `$request_uri`, `$request_method`, `$scheme`.

Значение поддерживает два вида сопоставления:

- *Сравнение строк без учета регистра*. Например:
  - `john` - сопоставление без учета регистра, которое выполняется успешно для таких строк, как `john`, `John`, `JOHN`.

  - `!john` - отрицание соответствия без учета регистра для john, которое выполняется успешно для таких строк, как `bob`, `anything`, `''` (пустая строка).

- *Сопоставление с регулярным выражением.* Обратите внимание, что Angie поддерживает регулярные выражения, совместимые с языком программирования Perl (PCRE). Например:
  - `~^yes` - регулярное выражение с учетом регистра, которое соответствует любой строке, начинающейся с `yes`. Например: `yes`, `yes123`.
  - `!~^yes` - отрицание предыдущего регулярного выражения, которое успешно выполняется для строк типа `YES`, `Yes123`, `noyes`. (Механизм отрицания не является частью синтаксиса PCRE).
  - `~*no$` – регулярное выражение без учета регистра, которое соответствует любой строке, заканчивающейся на `no`. Например: `no`, `123no`, `123NO`.

Примечание

Значение не должно содержать неэкранированных двойных кавычек (`"`) и не должно заканчиваться неэкранированной обратной косой чертой (\). Например, следующие значения недопустимы: `some"value`, `somevalue\`.

7.3.18 ErrorPage

Определяет настраиваемый ответ для маршрута на случай, когда сервер апстрима отвечает кодом состояния ошибки (или его генерирует Angie). В качестве ответа может быть задано перенаправление или сохраненный ответ. Дополнительные сведения см. в описании директивы error_page .

```
path: /coffee
errorPages:
  - codes: [502, 503]
    redirect:
      code: 301
      url: https://angie.software
  - codes: [404]
    return:
      code: 200
      body: "Original resource not found, but success!"
```

Поле	Описание	Тип	Обязательно
`codes`	Список кодов состояния ошибки.	`[]int`	Да
`redirect`	Действие перенаправления для заданных кодов состояния.	*errorPage.Re*	Нет
`return`	Сохраненное ответное действие для заданных кодов состояния.	*errorPage.Re*	Нет

Примечание

Страница с ошибкой должна содержать ровно одно из следующих значений: `return` или `redirect`.

7.3.19 ErrorPage.Redirect

Определяет перенаправление для errorPage.

В приведенном ниже примере Angie отвечает перенаправлением, когда ответ от сервера апстрима имеет код состояния 404.

```
codes: [404]
redirect:
  code: 301
  url: ${scheme}://cafe.example.com/error.html
```

Поле	Описание	Тип	Требуется
`код`	Код состояния перенаправления. Допустимые значения: `301`, `302`, `307`, `308`. Значение по умолчанию - `301`.	`int`	Нет
`url`	URL-адрес, на который будет перенаправлен запрос. Поддерживаемые переменные Angie: `$scheme` и `$http_x_forwarded_proto`. Переменные должны быть заключены в фигурные скобки. Например: `${scheme}`.	`stri`	Да

7.3.20 ErrorPage.Return

Определяет сохраненный ответ для errorPage.

В приведенном ниже примере Angie выдает сохраненный ответ, когда ответ от сервера апстрима имеет код состояния 401 или 403.

```
codes: [401, 403]
return:
  code: 200
  type: application/json
  body: |
    {\"msg\": \"You don't have permission to do this\"}
  headers:
    - name: x-debug-original-statuses
      value: ${upstream\_status}
```

Поле	Описание	Тип	Обязательно
`code`	Код состояния ответа. По умолчанию используется код состояния исходного ответа.	`int`	Нет
`type`	MIME-тип ответа. Значение по умолчанию - `text/html`.	`string`	Нет
`body`	Тело ответа. Поддерживаемая переменная Angie: `$upstream_status`. Переменные должны быть заключены в фигурные скобки. Например: `${upstream_status}`.	`string`	Да
`headers`	Настраиваемые заголовки ответа.	*errorPage.Re*	Нет *der*

7.3.21 ErrorPage.Return.Header

Определяет HTTP-заголовок для сохраненного ответа у errorPage:

```
name: x-debug-original-statuses
value: ${upstream_status}
```

Поле	Описание	Тип	Требуется
`name`	Имя заголовка.	`stri`	Да
`valu`	Значение заголовка. Поддерживаемая переменная Angie: `$upstream_status`. Переменные должны быть заключены в фигурные скобки. Например: `${upstream_status}`.	`stri`	Нет

7.4 Использование VirtualServer и VirtualServerRoute

Для работы с ресурсами VirtualServer и VirtualServerRoute можно использовать обычные команды `kubectl`, аналогично ресурсам Ingress.

Например, следующая команда создает ресурс VirtualServer, определенный в `cafe-virtual-server.yaml` с именем `cafe`:

```
kubectl apply -f cafe-virtual-server.yaml

virtualserver.k8s.angie.software "cafe" created
```

Вы можете получить ресурс, выполнив:

```
kubectl get virtualserver cafe

NAME   STATE   HOST                IP            PORTS      AGE
cafe   Valid   cafe.example.com    12.13.23.123  [80,443]   3m
```

В `kubectl get` и подобных командах также можно использовать короткое имя `vs` вместо `virtualserver`.

Работать с ресурсами VirtualServerRoute можно аналогично. В командах kubectl используйте `virtualserverroute` или короткое имя `vsr`.

7.4.1 Использование фрагментов

Фрагменты позволяют вставлять элементы конфигурации Angie в различные контексты конфигурации Angie. В приведенном ниже примере мы используем фрагменты кода для настройки нескольких функций Angie на VirtualServer:

```
apiVersion: k8s.angie.software/v1
kind: VirtualServer
metadata:
  name: cafe
  namespace: cafe
spec:
  http-snippets: |
    limit_req_zone $binary_remote_addr zone=mylimit:10m rate=1r/s;
    proxy_cache_path /tmp keys_zone=one:10m;
```

```
host: cafe.example.com
tls:
  secret: cafe-secret
server-snippets: |
  limit_req zone=mylimit burst=20;
upstreams:
  - name: tea
    service: tea-svc
    port: 80
  - name: coffee
    service: coffee-svc
    port: 80
routes:
  - path: /tea
    location-snippets: |
      proxy\_cache one;
      proxy\_cache\_valid 200 10m;
    action:
      pass: tea
  - path: /coffee
    action:
      pass: coffee
```

Фрагменты предназначены для продвинутых пользователей Angie, которым требуется больше контроля над генерируемой конфигурацией Angie.

Однако из-за недостатков, описанных ниже, фрагменты по умолчанию отключены. Чтобы использовать фрагменты, задайте аргумент командной строки `enable-snippets`.

Недостатки использования фрагментов:

- *Сложность*. Чтобы использовать фрагменты, требуется:
  - Понимать примитивы конфигурации Angie и реализовать правильную конфигурацию Angie.
  - Понимать, как ANIC генерирует конфигурацию Angie, чтобы фрагмент не мешал другим функциям конфигурации.
- *Сниженная надежность*. Неправильный фрагмент делает конфигурацию Angie недействительной, что приведет к ошибке при перезагрузке. Это помешает применить какие-либо обновления конфигурации, включая обновления для других ресурсов VirtualServer и VirtualServerRoute, пока фрагмент не будет исправлен.
- *Последствия для безопасности*. Фрагменты предоставляют доступ к примитивам конфигурации Angie, и эти примитивы не проверяются самим ANIC. Например, через фрагмент можно настроить в Angie произвольную отправку сертификатов TLS и ключей, используемых для завершения TLS у ресурсов Ingress и VirtualServer.

Чтобы помочь отлавливать ошибки при использовании фрагментов, ANIC сообщает об ошибках перезагрузки конфигурации в журналах, а также в полях событий и состояния ресурсов VirtualServer и VirtualServerRoute.

Примечание

Пока конфигурация Angie содержит недопустимый фрагмент, Angie будет продолжать работать с последней допустимой конфигурацией.

7.4.2 Валидация

Для ресурсов VirtualServer и VirtualServerRoute доступны два типа валидации:

- *Структурная валидация* с помощью `kubectl` и сервера Kubernetes API.
- *Всесторонняя валидация* с помощью ANIC.

Структурная валидация

Пользовательские определения ресурсов для VirtualServer и VirtualServerRoute включают структурную схему OpenAPI, которая описывает тип каждого поля этих ресурсов.

Если вы попытаетесь создать (или обновить) ресурс с нарушением структурной схемы (например, используете строковое значение для поля порта апстрима), `kubectl` и сервер Kubernetes API отклонят такой ресурс:

- Пример проверки `kubectl`:

```
kubectl apply -f cafe-virtual-server.yaml

  error: error validating "cafe-virtual-server.yaml": error validating
  data: ValidationError(VirtualServer.spec.upstreams[0].port): invalid
  type for software.angie.k8s.v1.VirtualServer.spec.upstreams.port: got
  "string", expected "integer"; if you choose to ignore these errors,
  turn validation off with --validate=false
```

- Пример проверки сервера Kubernetes API:

```
kubectl apply -f cafe-virtual-server.yaml --validate=false

  The VirtualServer "cafe" is invalid: []: Invalid value:
  map[string]interface {}{ ... }: validation failure list:
  spec.upstreams.port in body must be of type integer: "string"
```

Если ресурс не отклонен (т. е. не нарушает структурную схему), ANIC проверит его дополнительно.

Всесторонняя валидация

ANIC проверяет поля ресурсов VirtualServer и VirtualServerRoute. Если ресурс недействителен, ANIC отклонит его: ресурс продолжит существовать в кластере, но ANIC будет его игнорировать.

Вы можете проверить, успешно ли ANIC применил конфигурацию для VirtualServer. Для нашего примера VirtualServer `cafe` мы можем запустить:

```
kubectl describe vs cafe

. . .
Events:
  Type    Reason          Age   From                      Message
  -----
  Normal  AddedOrUpdated  16s   angie-ingress-controller  Configuration for default/
↪cafe was added or updated
```

Обратите внимание, что раздел «События» (Events) включает событие Normal с причиной AddedOrUpdated, которое информирует нас о том, что конфигурация была успешно применена.

Если вы создадите недопустимый ресурс, ANIC отклонит его и выдаст событие Rejected. Например, если вы создадите VirtualServer `cafe` с двумя серверами апстрима с одинаковым именем `tea`, вы получите:

```
kubectl describe vs cafe

. . .
Events:
  Type     Reason    Age   From                      Message
  -----
  Warning  Rejected  12s   angie-ingress-controller  VirtualServer default/cafe is␣
↪invalid and was rejected: spec.upstreams[1].name: Duplicate value: "tea"
```

Обратите внимание, что раздел «События» (Events) включает предупреждающее событие с указанием причины отклонения.

Кроме того, эта информация также доступна в поле `status` ресурса VirtualServer. Обратите внимание на раздел `Status` VirtualServer:

```
kubectl describe vs cafe

. . .
Status:
  External Endpoints:
    Ip:        12.13.23.123
    Ports:     [80,443]
  Message:  VirtualServer default/cafe is invalid and was rejected: spec.upstreams[1].
↪name: Duplicate value: "tea"
  Reason:   Rejected
  State:    Invalid
```

ANIC проверяет ресурсы VirtualServerRoute аналогичным образом.

Примечание

Если вы внесете ошибку в существующий ресурс, ANIC отклонит его и удалит соответствующую конфигурацию из Angie.

7.5 Настройка с помощью ConfigMap

Вы можете дополнительно настроить конфигурацию Angie для ресурсов VirtualServer и VirtualServerRoutes, используя `ConfigMap`. Поддерживается большинство ключей ConfigMap, за следующими исключениями:

- `proxy-hide-headers`
- `proxy-pass-headers`
- `hsts`
- `hsts-max-age`
- `hsts-include-subdomains`
- `hsts-behind-proxy`
- `redirect-to-https`
- `ssl-redirect`