avito.tech Москва 2024

Как аппсеки в Авито API собирали

Александр Трифанов ведущий в безопасное будущее инженер

whoami

Александр Трифанов
ведущий в безопасное будущее инженер

- Замкадыш из Новосибирска.
- 7+ лет занимаюсь тестированием на проникновение и разработкой решений для продуктовой безопасности.
- Неравнодушен к реверс-инжинирингу, эксплуатации бинарных уязвимостей и языку Rust.

hostname

Авито

В Авито **больше трех тысяч микросервисов** на Python и Go, фронтенд на JavaScript, базы данных PostgreSQL, MongoDB и Redis, автоматические тесты, обученная на данных система модерации и поисковый движок Sphinx.

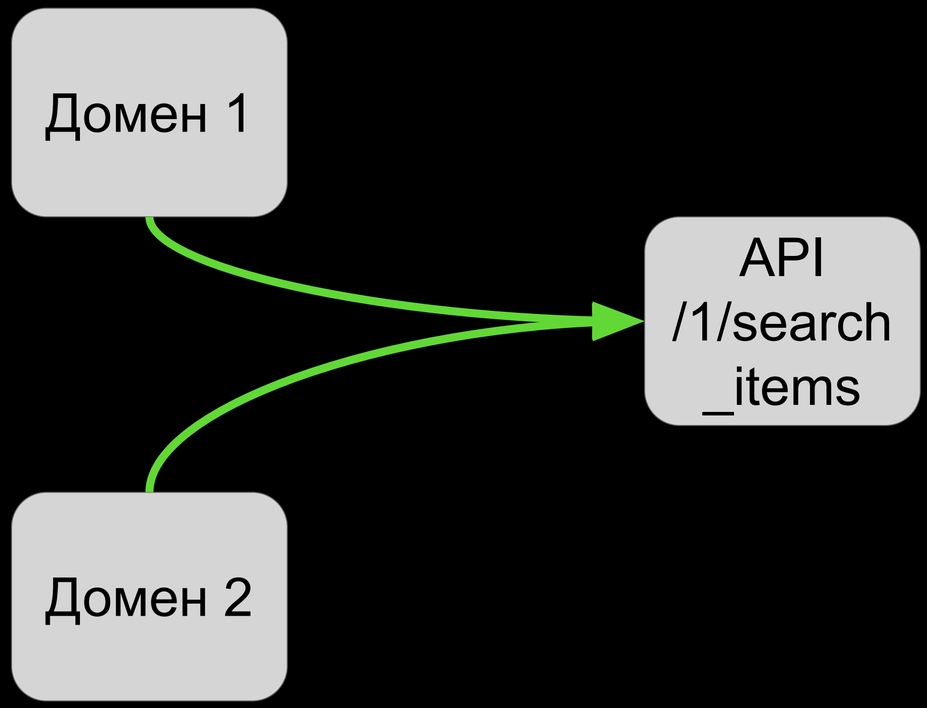
У нас уже **больше 1800 инженеров**. Мы работаем в небольших кросс-функциональных командах, каждая из которых отвечает за конкретную часть сервиса. Нашим продуктом каждый месяц пользуются десятки миллионов человек, поэтому мы умеем работать с большими нагрузками.

О чём пойдёт речь

- Какие проблемы решаем.
- Как устроен роутинг запросов в Авито.
- Подход к сбору данных об API.
- Тонкости определения доступности API через URL.
- Метрики и результаты.
- Место системы сбора API в SSDLC Авито.
- Следующие этапы развития системы.

Проблемы

- **Разные рейтлимиты** на ручку API.



Проблемы

- **Разные рейтлимиты** на ручку API.
- Есть URL — надо **найти владельцев.**
- Давно эта ручка доступна?
- Найти **неиспользуемые** ручки.
- А мы **проверяли** что там?

Типичный репорт в баг баунти

- привет, я нашёл баг в /api/1/some/path

- чья это ручка?
- сколько баг в проде?
- а эта ручка вообще нужна?
- почему не нашли сами до прода?

Проблемы

- **Разные рейтлимиты** на ручку API.
- Есть URL — надо **найти владельцев.**
- Давно эта ручка доступна?
- Найти **неиспользуемые** ручки.
- А мы **проверяли** что там?
- **DAST** не знает куда делать запросы.
- А на ручку включён **WAF**?
- Какие ручки **доступны снаружи**?

- приоритизация
- покрытие инструментами

Проблемы

- **Разные рейтлимиты** на ручку API.
- Есть URL — надо **найти владельцев.**
- Давно эта ручка доступна?
- Найти **неиспользуемые** ручки.
- А мы **проверяли** что там?
- **DAST** не знает куда делать запросы.
- А на ручку включён **WAF**?
- Какие ручки **доступны снаружи**?
- Какие ручки появились за последние сутки?

- мониторинг изменений — что случилось пока аппсек спал?

Проблемы

- **Разные рейтлимиты** на ручку API.
- Есть URL — надо **найти владельцев.**
- Давно эта ручка доступна?
- Найти **неиспользуемые** ручки.
- А мы **проверяли** что там?
- **DAST** не знает куда делать запросы.
- А на ручку включён **WAF**?
- Какие ручки **доступны снаружи**?
- Какие ручки появились за последние сутки?
- Проверка релиза.

Как проверяли релизы раньше?

- Релиз монолита едет на стейджинг.
- Пентестер идёт по списку важных/новых/изменившихся API.
- Список API собирается руками.
- Время нужное для релиза растёт =(

~~Проблемы~~

- ~~**Разные рейтлимиты** на ручку API.~~
- ~~Есть URL — надо **найти владельцев.**~~
- ~~Давно эта ручка доступна?~~
- ~~Найти **неиспользуемые** ручки.~~
- ~~А мы **проверяли** что там?~~
- ~~**DAST** не знает куда делать запросы.~~
- ~~А на ручку включён **WAF**?~~
- ~~Какие ручки **доступны снаружи**?~~
- ~~Какие ручки появились за последние сутки?~~
- ~~Проверка релиза.~~

Возможности

- **Актуальный** роутинг в любой момент времени.
- **Мониторинг** изменений
- Фильтрация по критерию **доступности.**
- **Атрибуция** по покрытию API нашими инструментами.
- Возможность **экспорта** данных в подходящем для DAST и сканера утечек формате.

Как устроена маршрутизация

- nginx
  - envoy
    - api-gateway
      - microservice

Как устроена маршрутизация

- nginx 1
  - envoy 1
    - api-gateway 1
      - microservice 1
      - microservice 2
      - …
      - microservice 3000
    - api-gateway 2
    - …
    - api-gateway 6
  - envoy 2
  - envoy 3
- nginx 2

https://museum-design.ru/nomura-most-difficult-maze/

Идея

- Идеальное решение — управление ассетами API.
- Собираем все правила роутинга запросов.
- Строим граф.
- Пишем аналитику вокруг графа.
- PROFIT!

Warning (!)

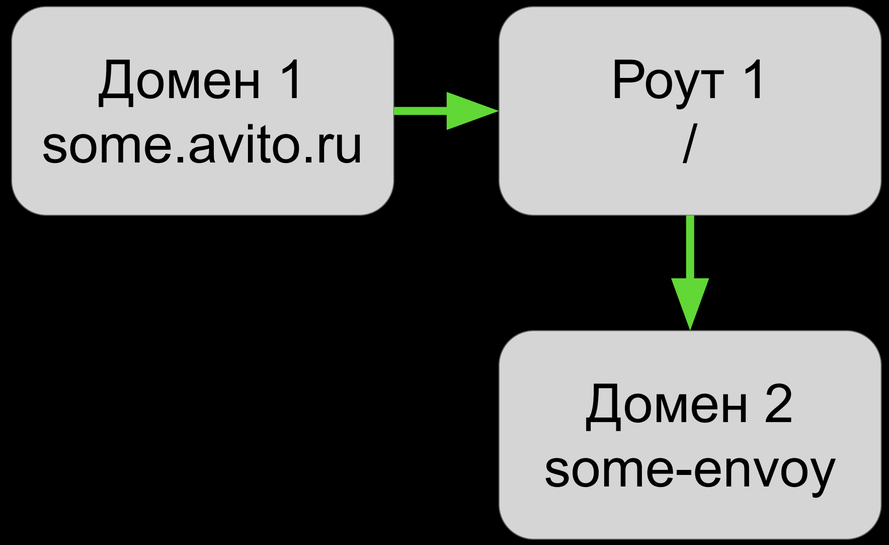
- **Маршрут** — цепочка правил маршрутизации, через которые идёт запрос.
- **API** — конечная точка следования запроса.
- **Роут** — отдельное правило обработки запроса (например location в nginx).
- **Сервис** — микросервис, как правило содержит API, иногда роуты.
- **Домен** — узел маршрутизации из nginx или envoy, как правило содержит роуты.

Разница между доменами, сервисами и роутами в правилах обработки запроса

В начале был nginx

- Спасибо Open source - gixy :)
- Вжух, и у нас есть первоначальный роутинг — да это было легко!

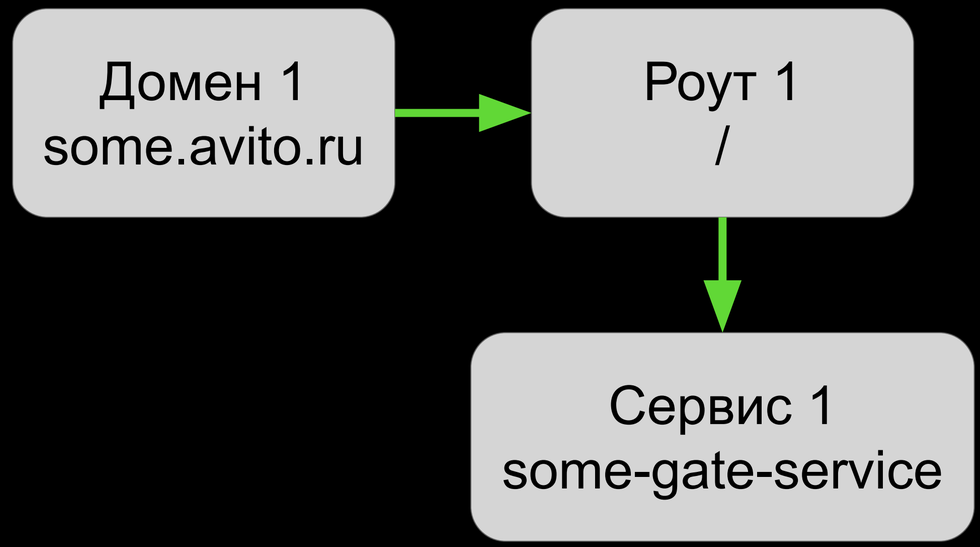
```
server {
    server_name some.avito.ru;
    location / {
       proxy_pass  http://some-envoy;
    }
}
```



Заходим в k8s и envoy

- весь конфиг доступен как json

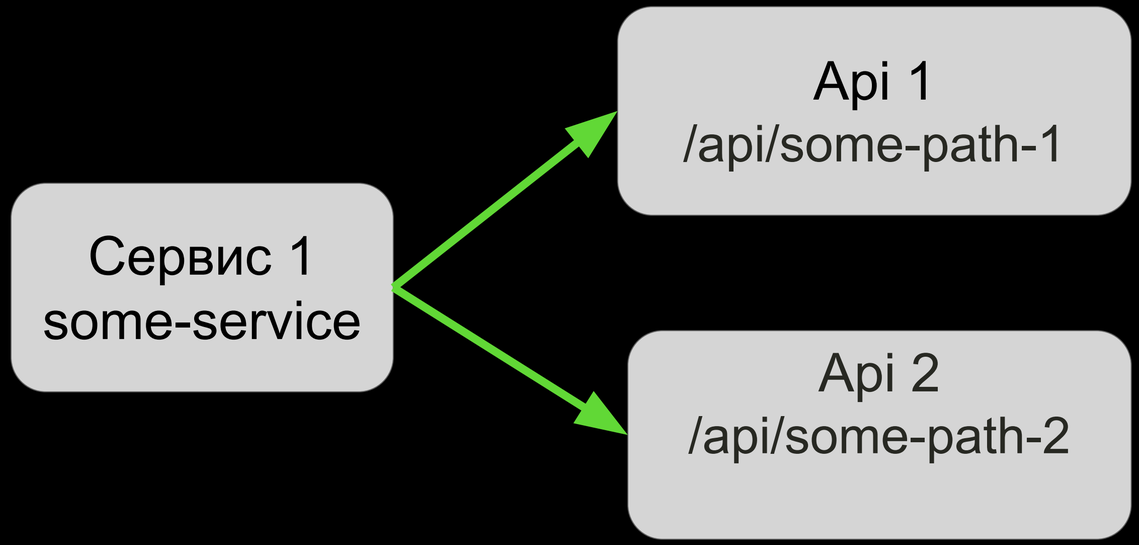
```
{
  "name": "some.avito.ru",
  "domains": [
   "some.avito.ru"
  ],
  "routes": [ {
"match": {
   "prefix": "/",
   "route": { "weighted_clusters": {
      "clusters": [
         {
           "name": "some-gate-service",
         }  ], },
     "prefix_rewrite": "/",
   }, },]}
```



Прикоснёмся к микросервисам

- API-gateway: конфиги - yaml файлы.
- Сами микросервисы: реализация контрактов gRPC (yaml|toml подобное описание).
- API для гейтвея в микросервисах - есть кодогенерация swagger-файлов.

```
- name: 'some-service'
  locations:
    - '~ /api/some-path-1'
    - '~ /api/some-path-2'
```



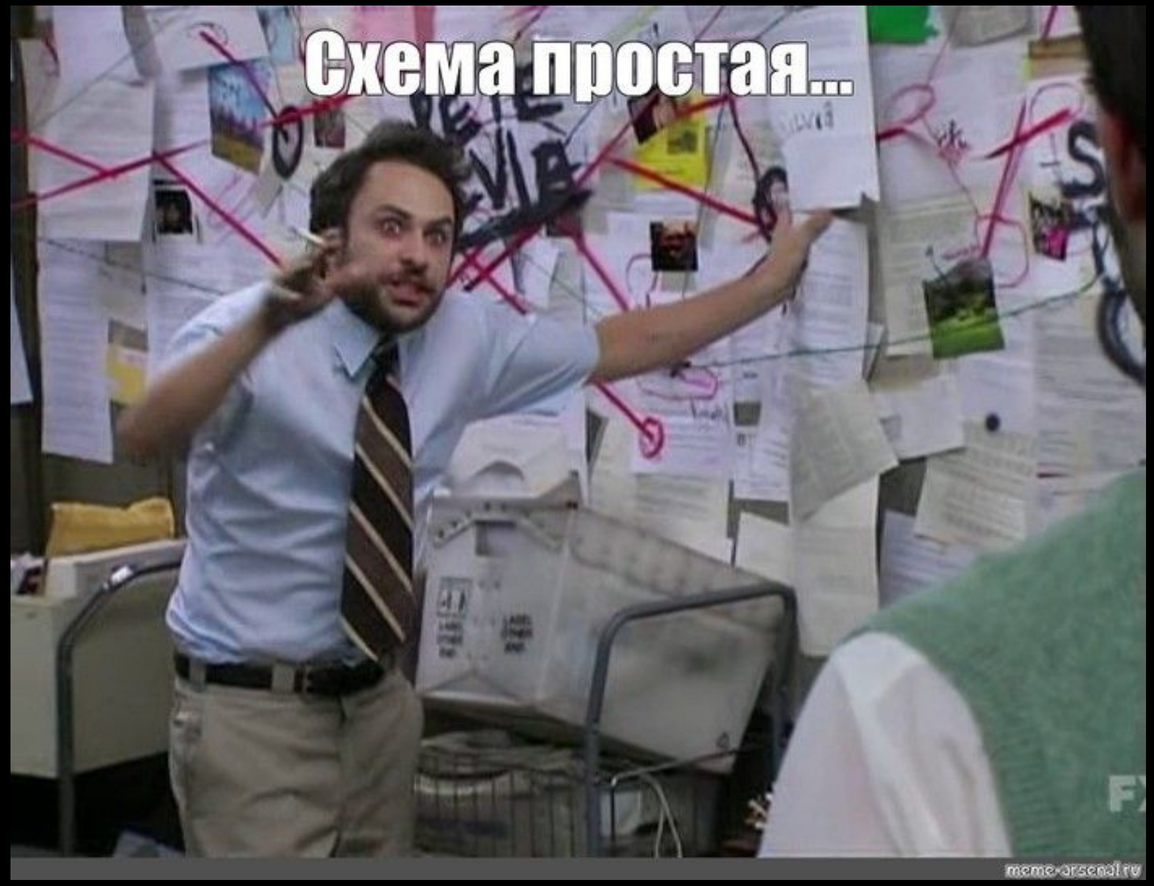
Чудесный мир Open API

- Ведь 100% есть библиотеки, которые парсят swagger файлы, да?
- Однажды увидел странное — большущий swagger файл на несколько килобайт после парсинга отдавал всего 3-4 ручки.
- В файле их 100+
- Достаём отладчик и …

https://www.meme-arsenal.com

Чудесный мир Open API

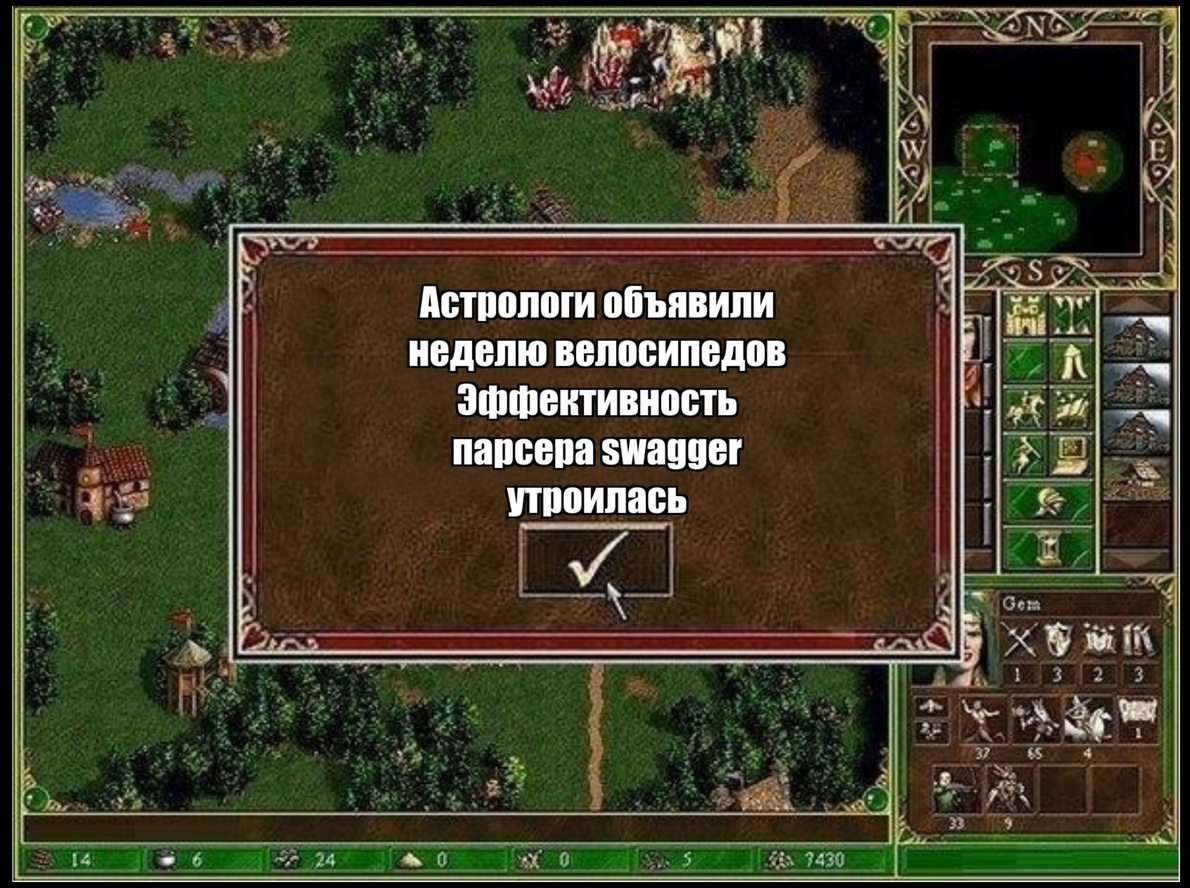
- В swagger-файлах есть ссылки — и это реально URI.
- В URI символ разделитель - это «/».
- В имени ручки API тоже может быть этот символ.
- Поэтому при генерации swagger ссылки на ручку/поле в ручке «/» превращается в «~1».
- А «~» в «~0»
- Из просмотренных мной парсеров swagger для питона ни один из них в такое не умел.
- Равно как в них не было возможности распарсить рекурсивную структуру данных.



https://www.meme-arsenal.com

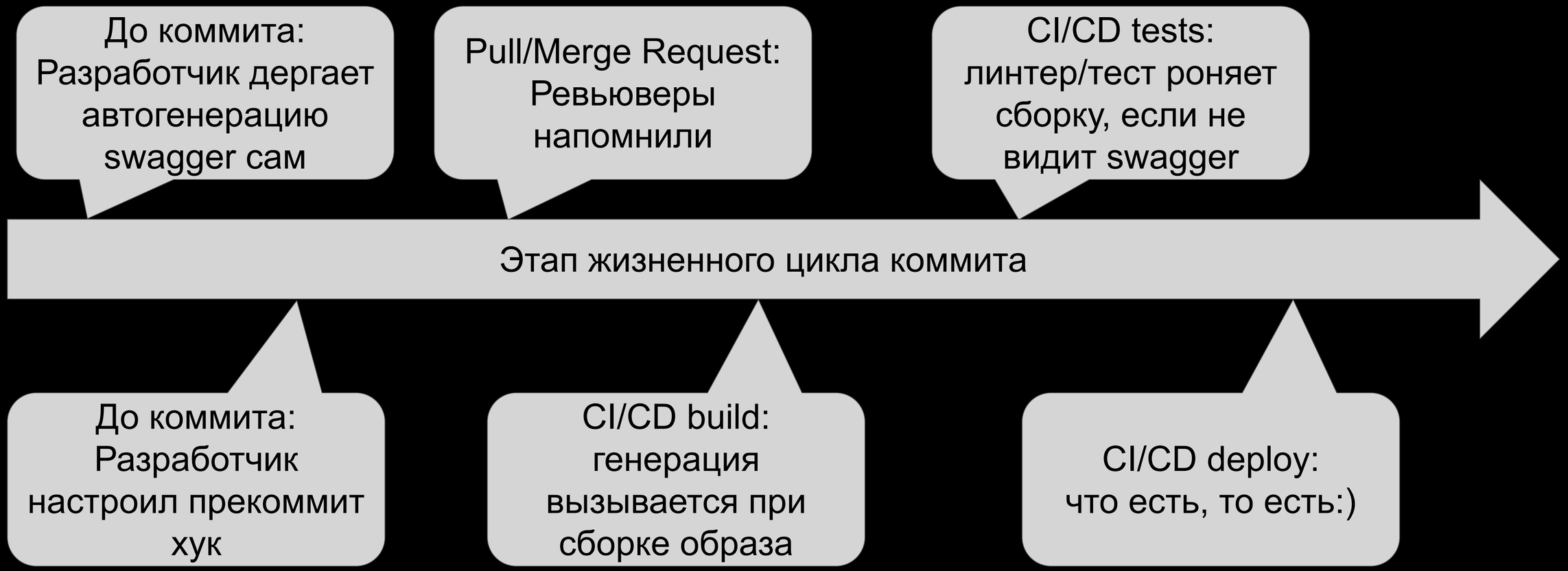
Чудесный мир Open API

- Написал свой парсер - количество ручек увеличилось втрое.
- Но вы ж не думали, что это победа? :)



https://www.meme-arsenal.com

Чудесный мир Open API: автогенерация



Чудесный мир Open API: как достать?

https://memepedia.ru

- Идём в registry.
- Забираем docker-образ.
- Достаём swagger.
- …
- PROFIT!

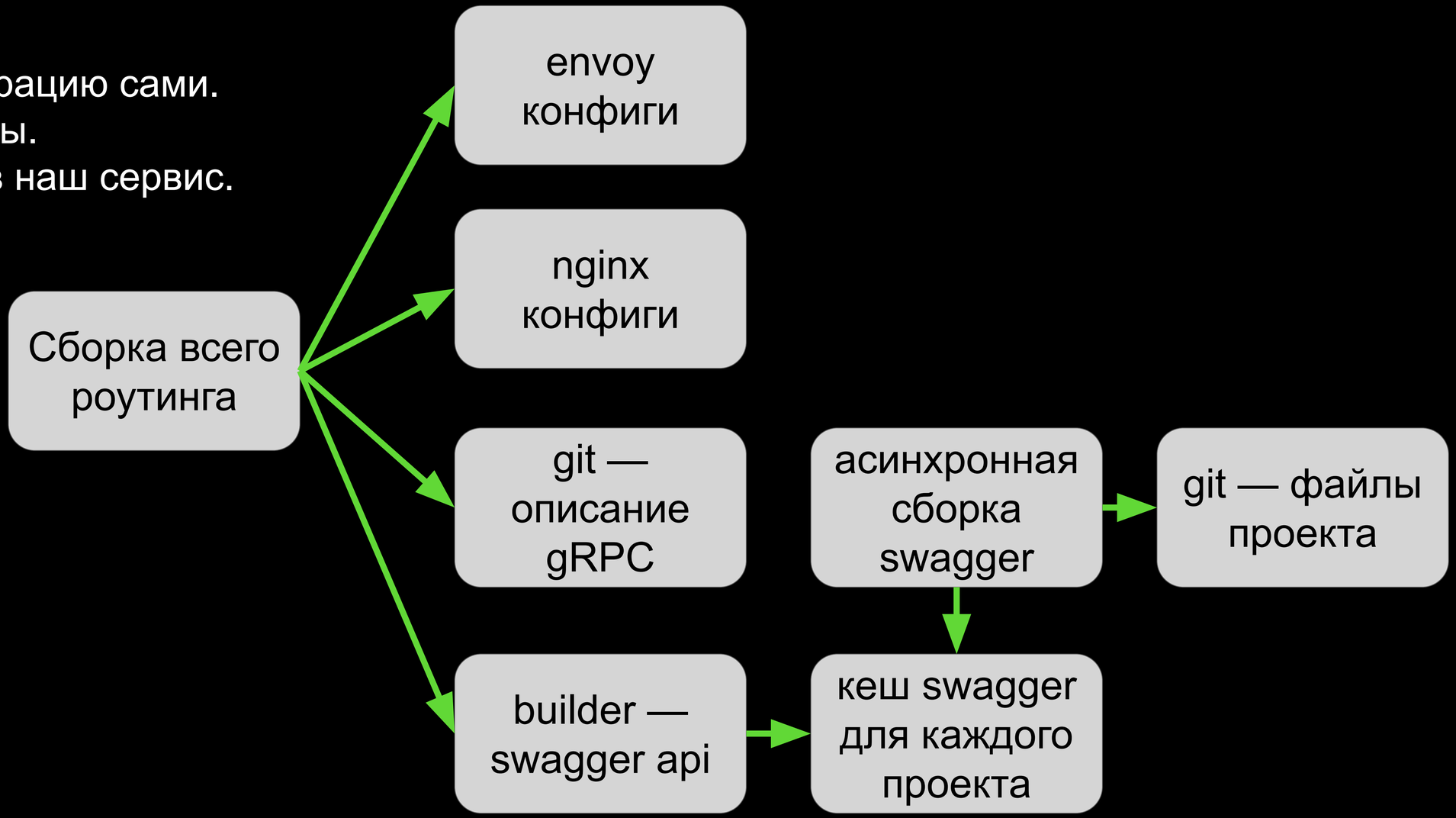
Чудесный мир Open API: как достать?

https://memepedia.ru

https://memepedia.ru

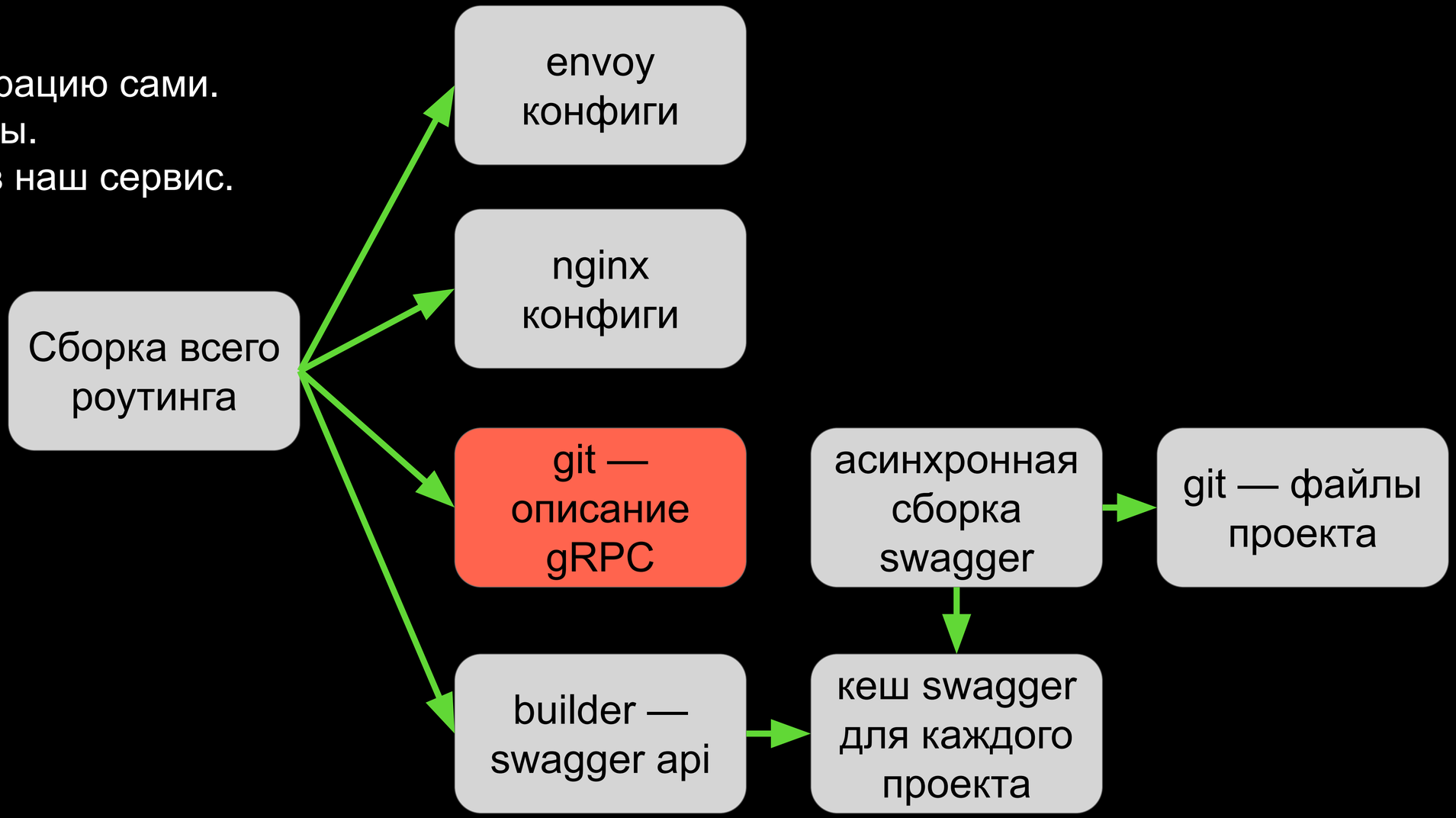
Чудесный мир Open API: как достать?

- Забираем код из git.
- Запускаем кодогенерацию сами.
- Кешируем результаты.
- По запросу отдаём в наш сервис.



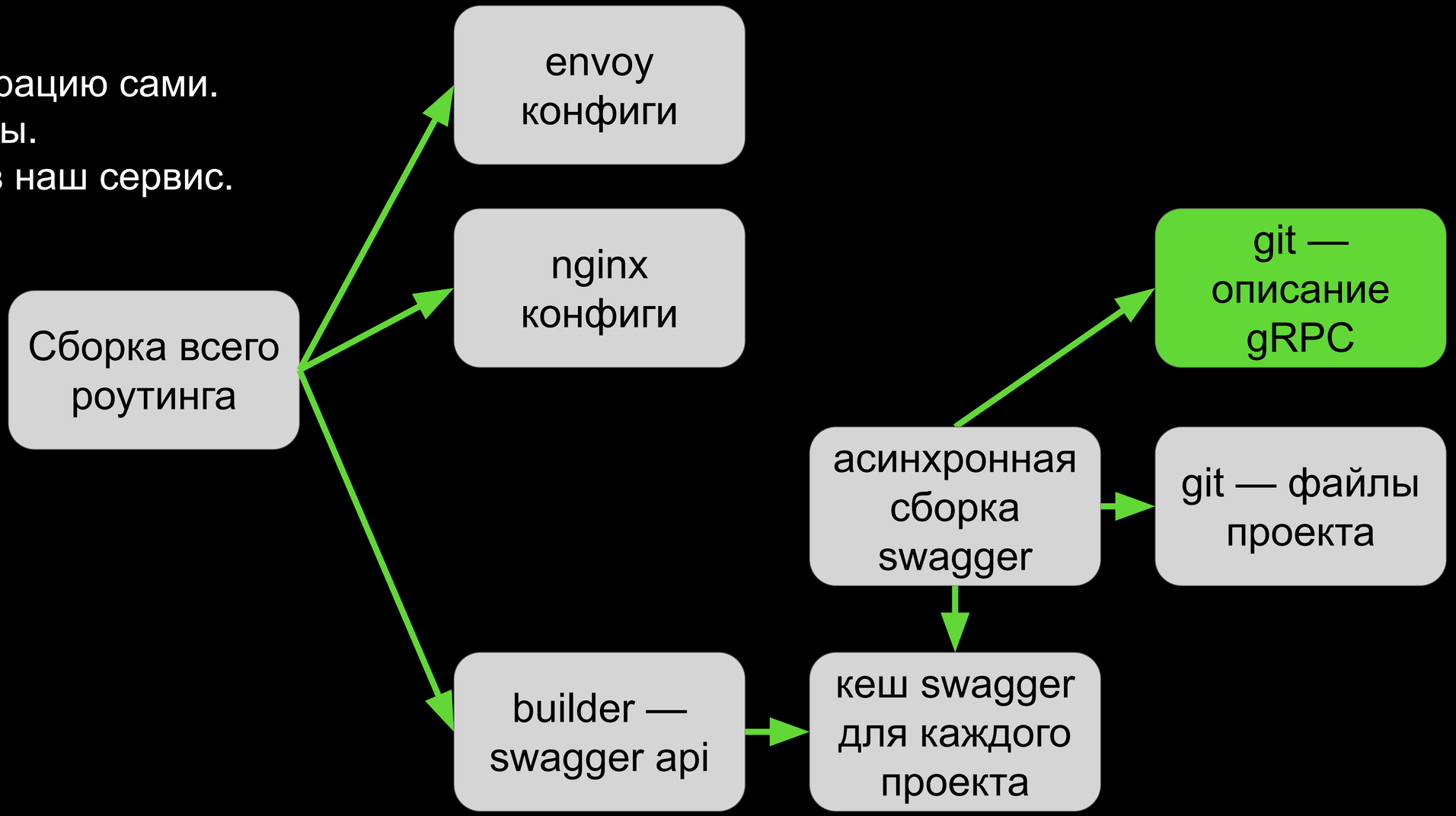
Чудесный мир Open API: как достать?

- Забираем код из git.
- Запускаем кодогенерацию сами.
- Кешируем результаты.
- По запросу отдаём в наш сервис.



Чудесный мир Open API: как достать?

- Забираем код из git.
- Запускаем кодогенерацию сами.
- Кешируем результаты.
- По запросу отдаём в наш сервис.



Собираем роутинг воедино: The Core

- Собрали списки доменов, роутов, сервисов и ручек.
- Теперь строим граф кто куда ведёт.

Собираем роутинг воедино: The Core

- Собрали списки доменов, роутов, сервисов и ручек.
- Теперь строим граф кто куда ведёт.
- Понимаем, что надо обрабатывать кейсы, когда:
  - Домен 1 ведет в Домен 2.
  - Домен 2 ведет в Роут 3.

Собираем роутинг воедино: The Core

- Собрали списки доменов, роутов, сервисов и ручек.
- Теперь строим граф кто куда ведёт.
- Понимаем, что надо обрабатывать кейсы, когда:
  - Домен 1 ведет в Домен 2.
  - Домен 2 ведет в Роут 3.
  - Роут 3 модифицирует путь регуляркой, отрезает от него префикс, меняет параметры местами и шлёт в Домен 4.
  - ...
  - Роут 100500 ведет в микросервис 100501 в API-ручку 100502.

https://www.meme-arsenal.com

Собираем роутинг воедино: The Core

Вершины графа — это домены, роуты, сервисы и ручки API.

Рёбра графа — это наличие правила маршрутизации между ними.

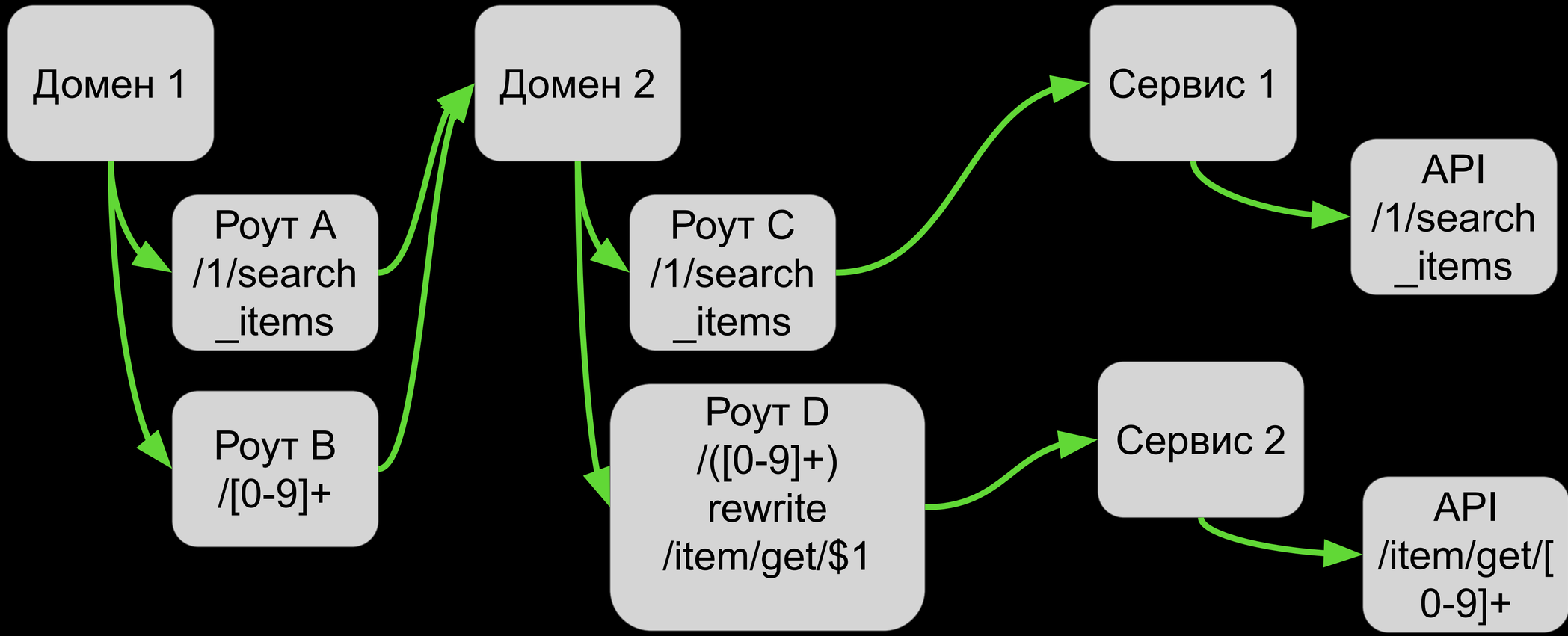
Собираем роутинг воедино: The Core

Вершины графа — это домены, роуты, сервисы и ручки API.

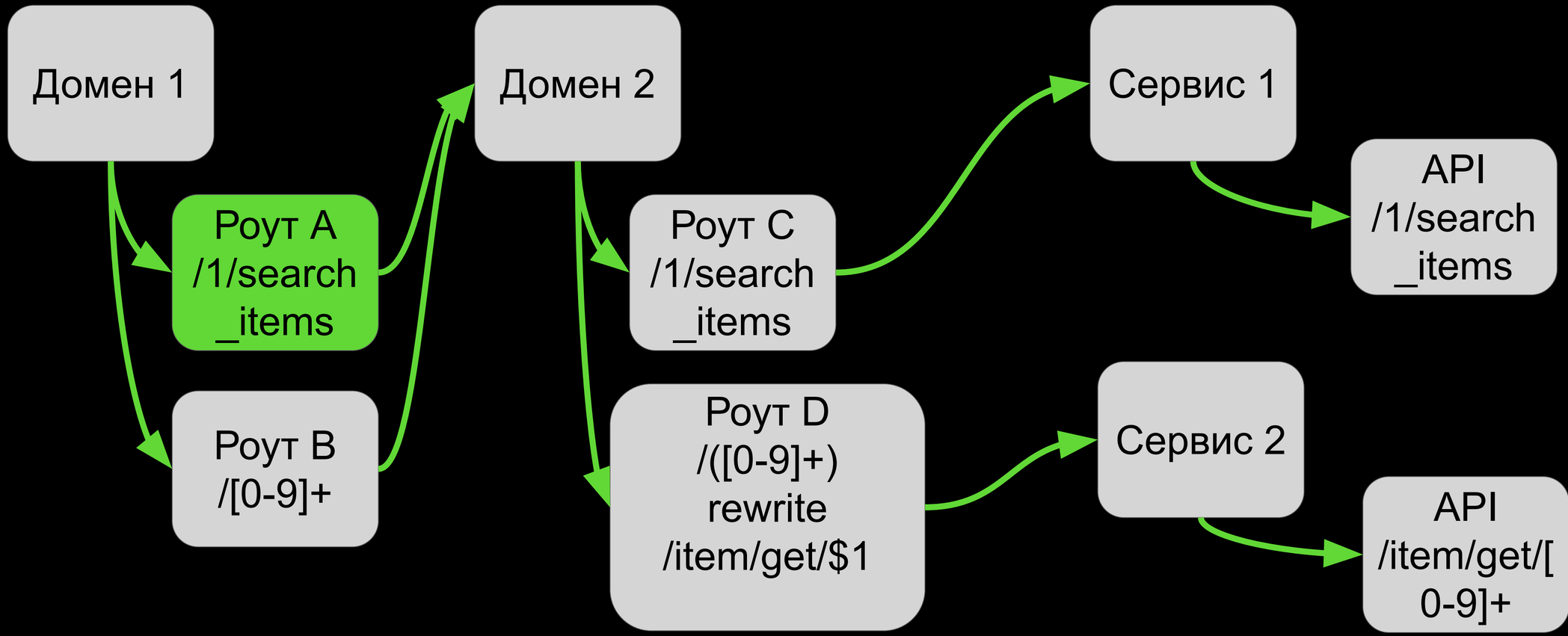
Рёбра графа — это наличие правила маршрутизации между ними.

Наличие правила маршрутизации между Роутом 1 и Сервисом 1.
!=
Роут 1 ведет в Сервис 1 и, соответственно, в ручки API1 и API2

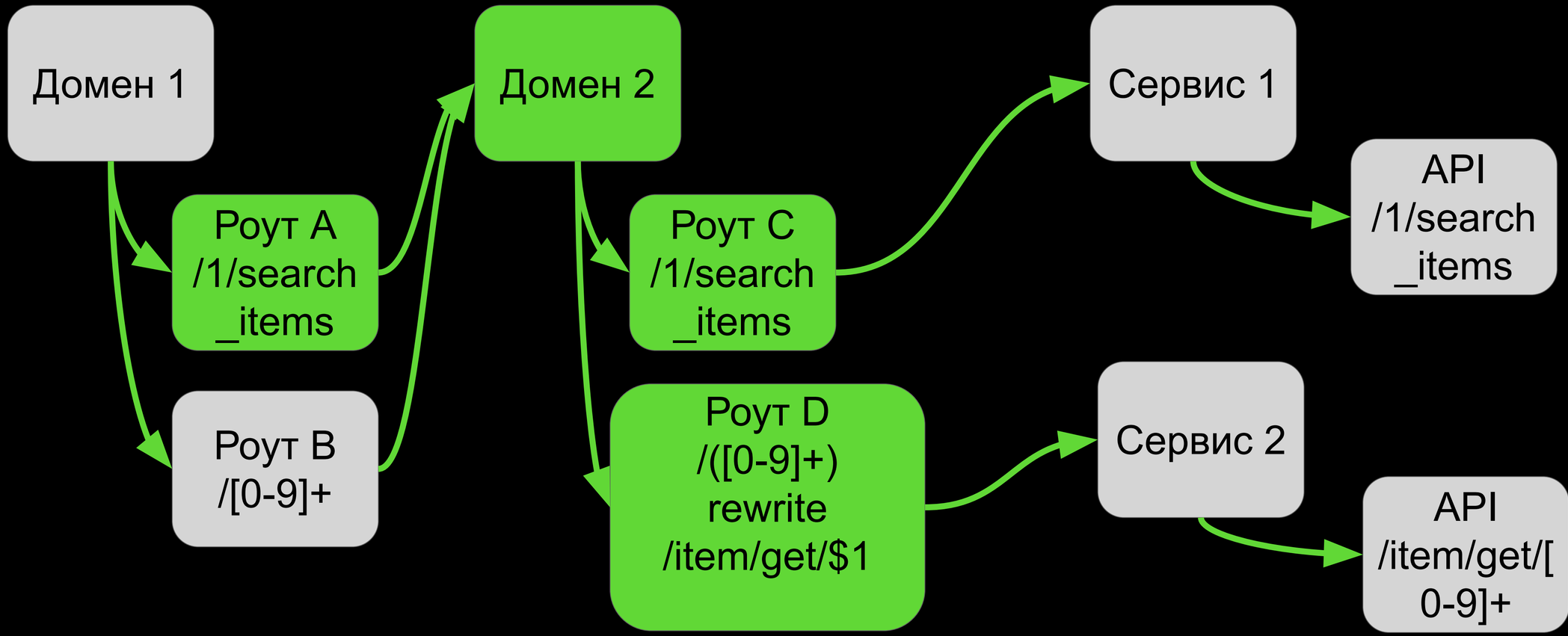
Небольшой пример



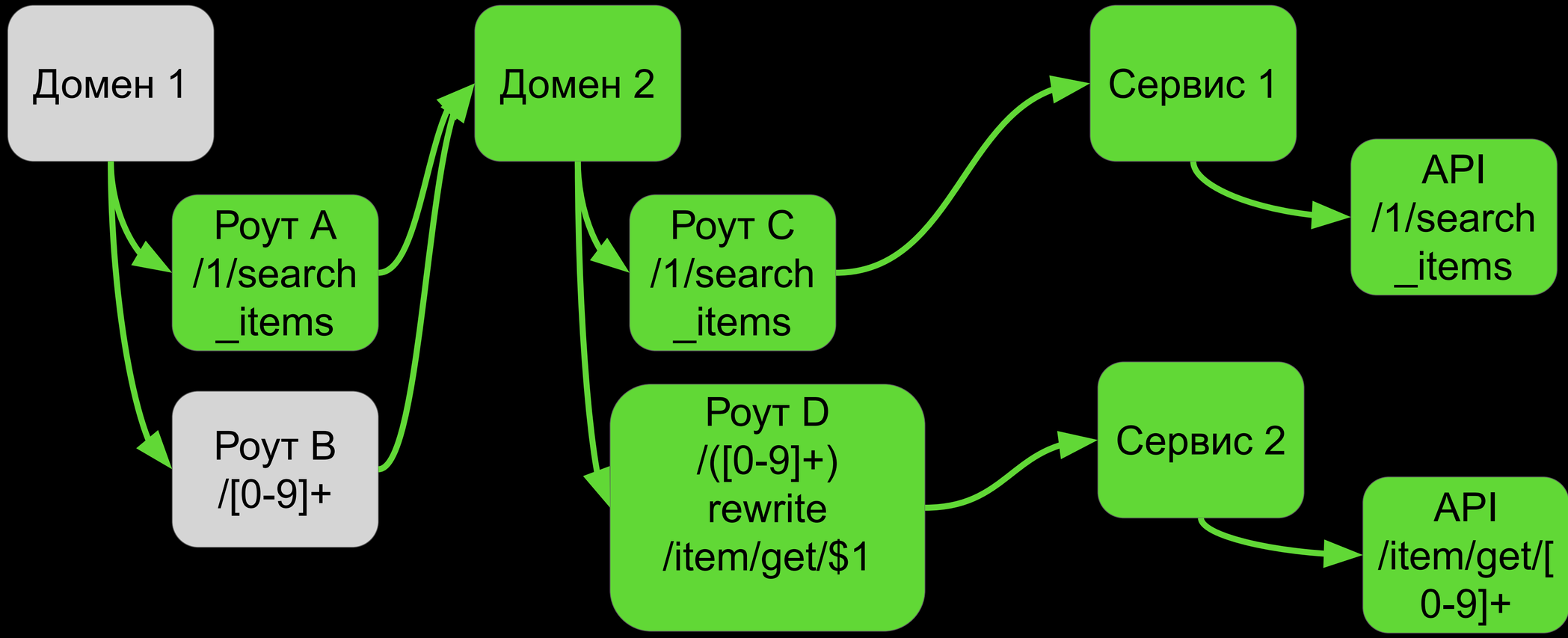
Небольшой пример



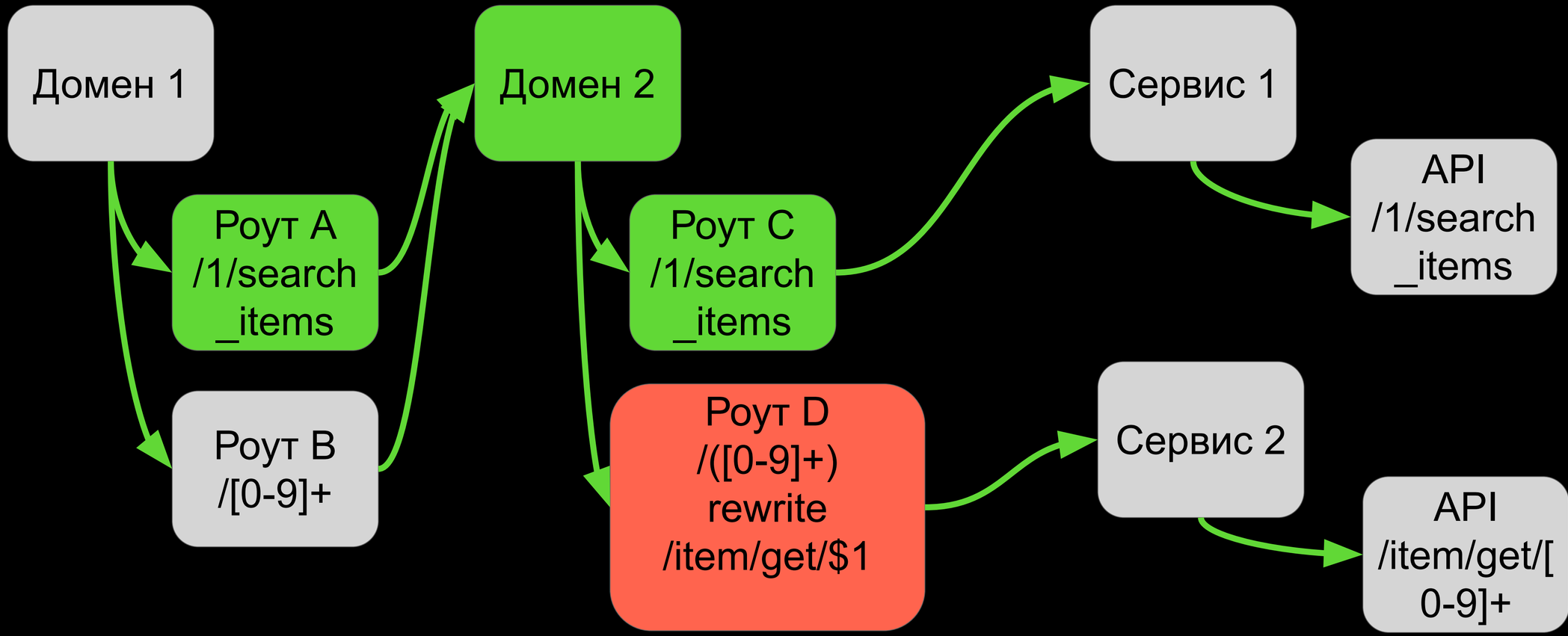
Небольшой пример



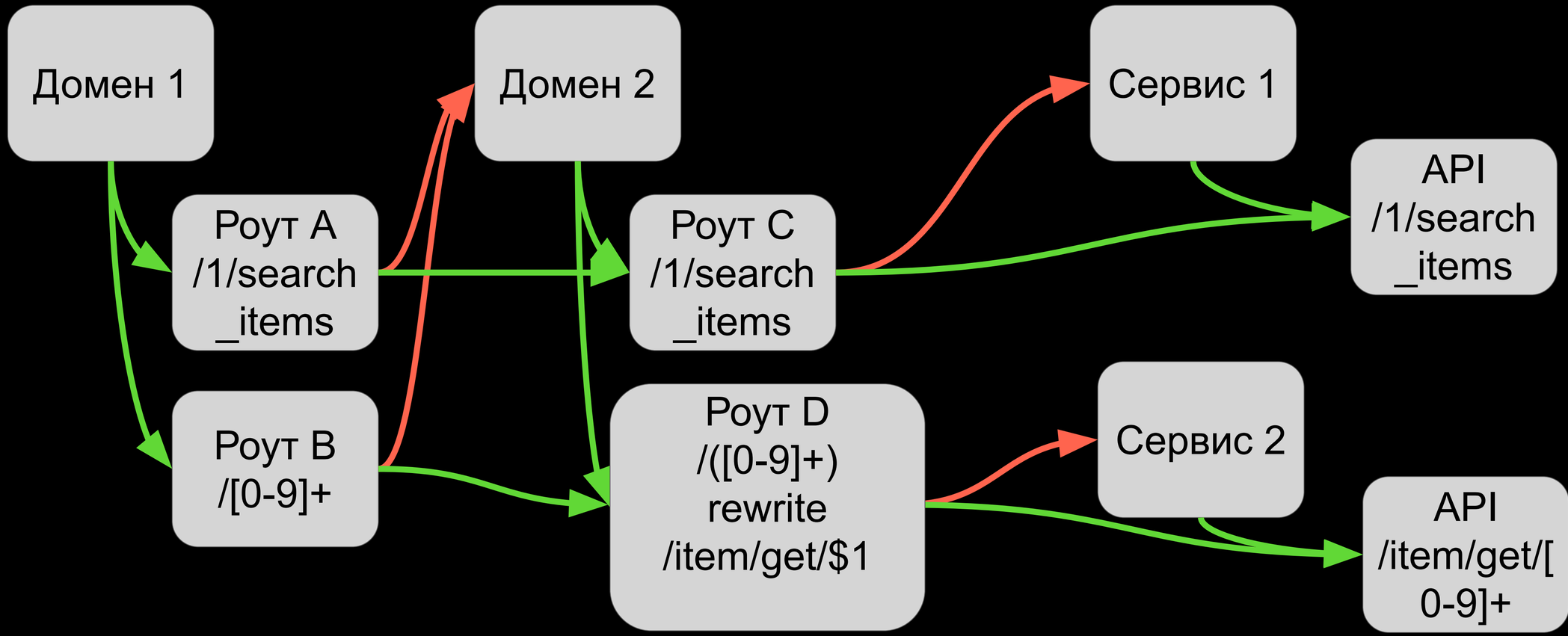
Небольшой пример



Небольшой пример

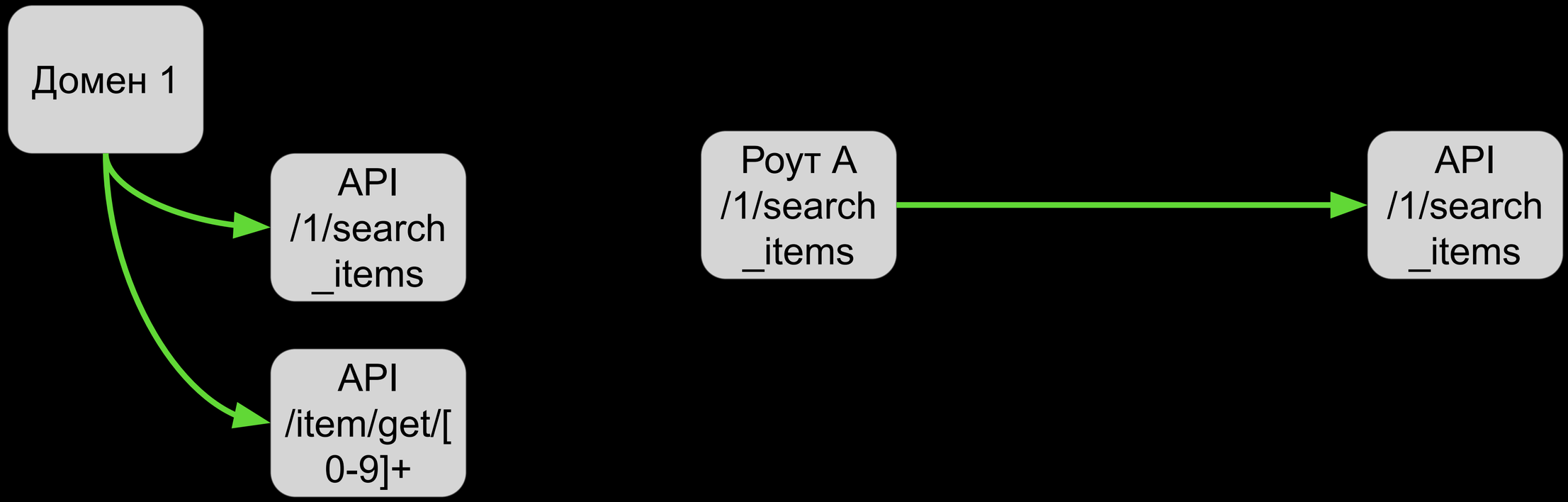


Небольшой пример



Собираем роутинг воедино: The Core

- Для каждой вершины **v** хранить только вершины, которые никуда не ведут.



Первые результаты

- Для отладки пришлось сделать фронтенд.

- Гипотеза:
  - если маршрут обрывается и никуда не ведёт — это ошибка парсинга.
- Реальность:
  - десятки маршрутов, которые никуда не вели (на 404 страницу).

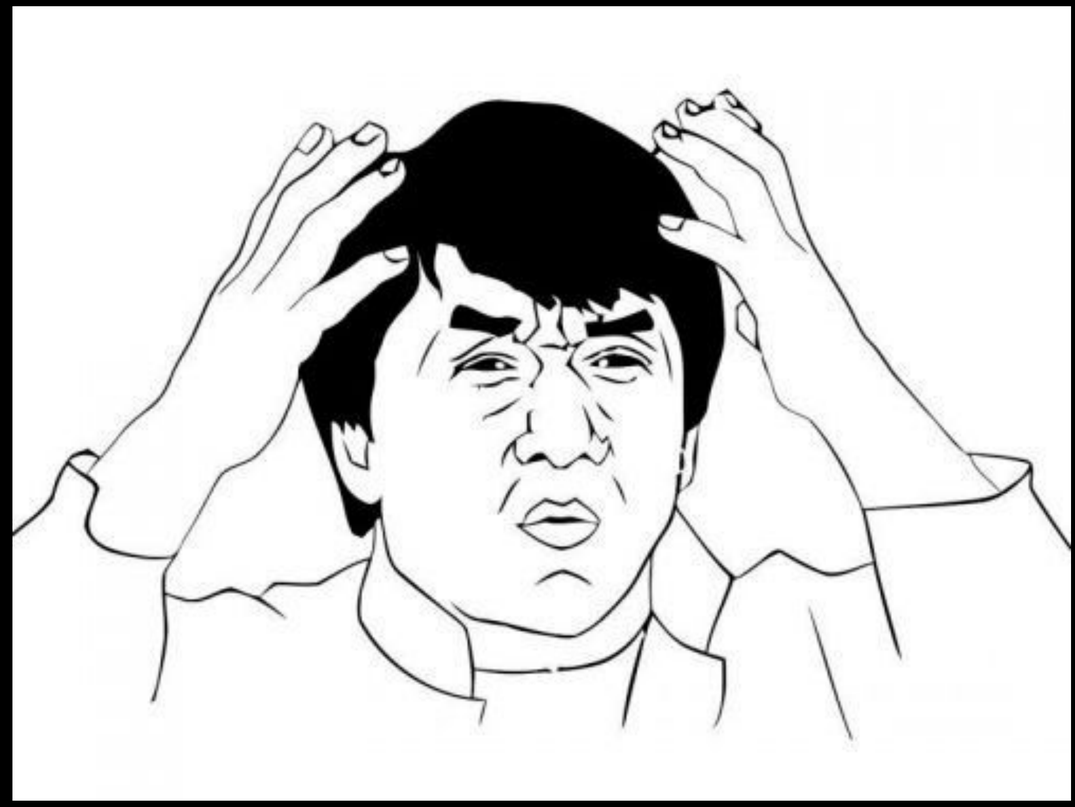
Оценка покрытия: подходы

- Сравнить по URL с логами nginx.
- Сравнить по URL и микросервису с трейсингом.

Оценка покрытия: первая цифра

- Сравнить по URL с логами nginx.
- Сравнить по URL и микросервису с трейсингом.

2%



https://www.meme-arsenal.com

Пригодность данных для DAST

- Приношу коллеге роутинг, показываю: «Вот, красивое, работает, строит граф, список ручек доступных из под домена отдаёт».
- После запуска DAST на этих данных коллега спрашивает: «А ты учёл, что эти ручки доступны в домене по другим путям, а не таким же как в самом микросервисе?».

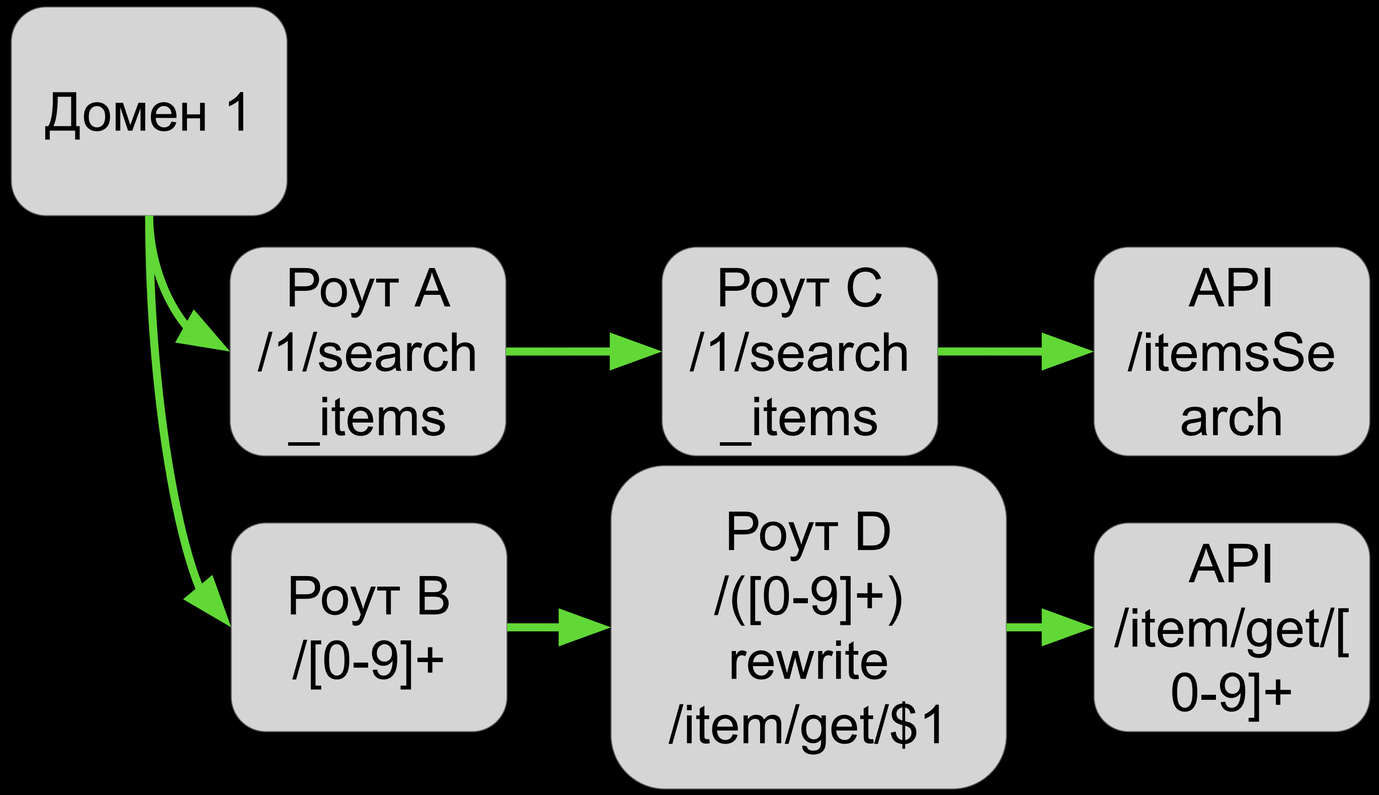
https://www.meme-arsenal.com

Собираем роутинг воедино: The Core

- Нужен весь путь от вершины до вершины.
- Чтобы видеть историю преобразований пути и получать внешний URL для внутренней ручки API.

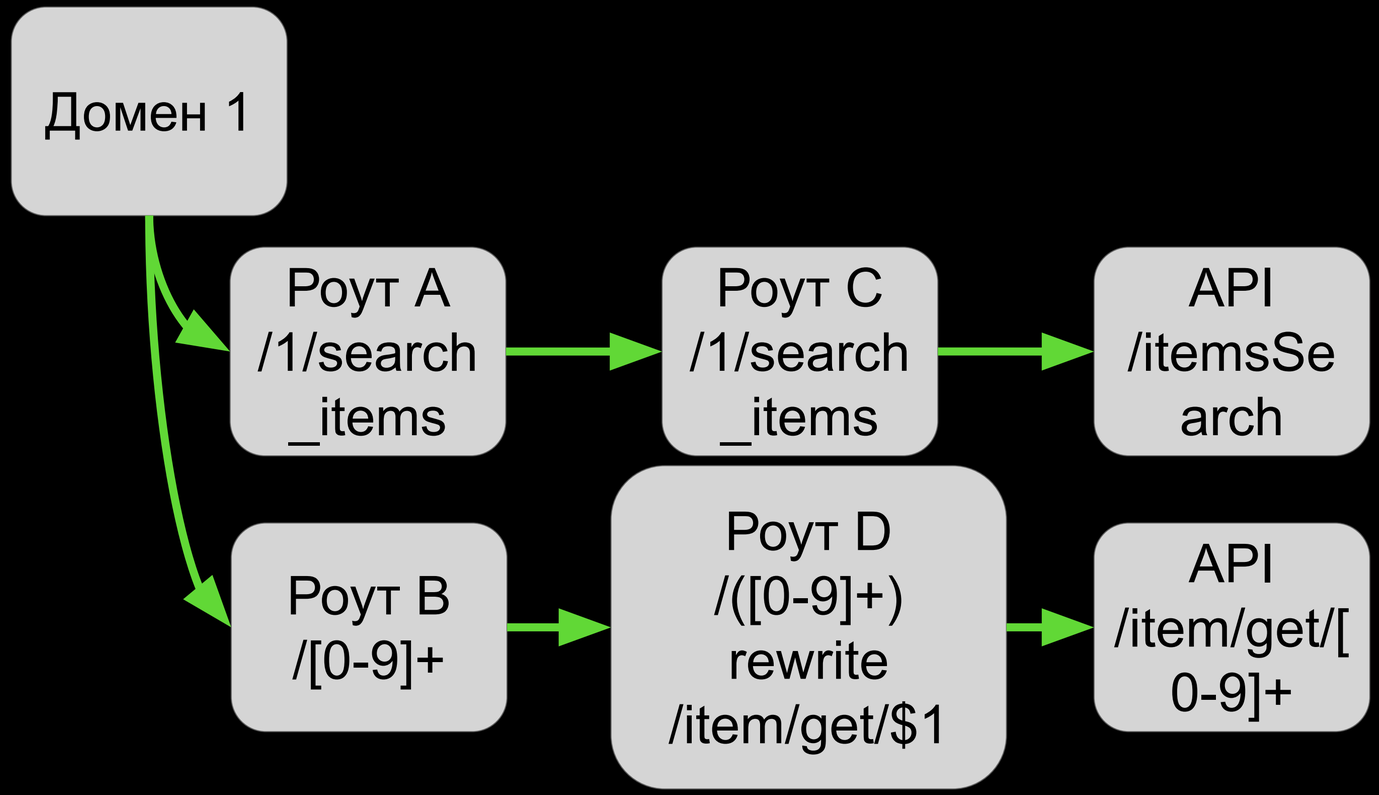
Собираем роутинг воедино: The Core

- Нужен весь путь от вершины до вершины.
- Чтобы видеть историю преобразований пути и получать внешний URL для внутренней ручки API.
- Для каждой вершины **v** нужно построить граф **G**, состоящий из всех вершин которые достижимы из **v**.



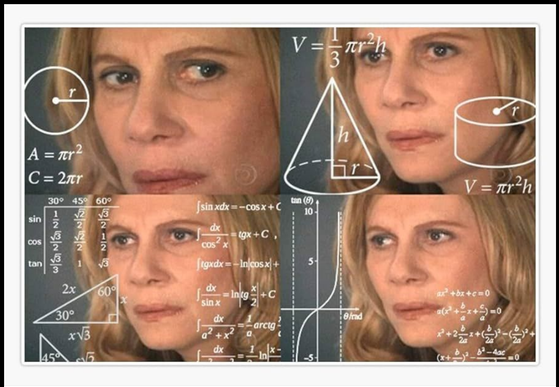
Собираем роутинг воедино: The Core

- Нужен весь путь от вершины до вершины.
- Чтобы видеть историю преобразований пути и получать внешний URL для внутренней ручки API.
- Для каждой вершины **v** нужно построить граф **G**, состоящий из всех вершин которые достижимы из **v**.
- **Да это же задача построения транзитивной редукции!**

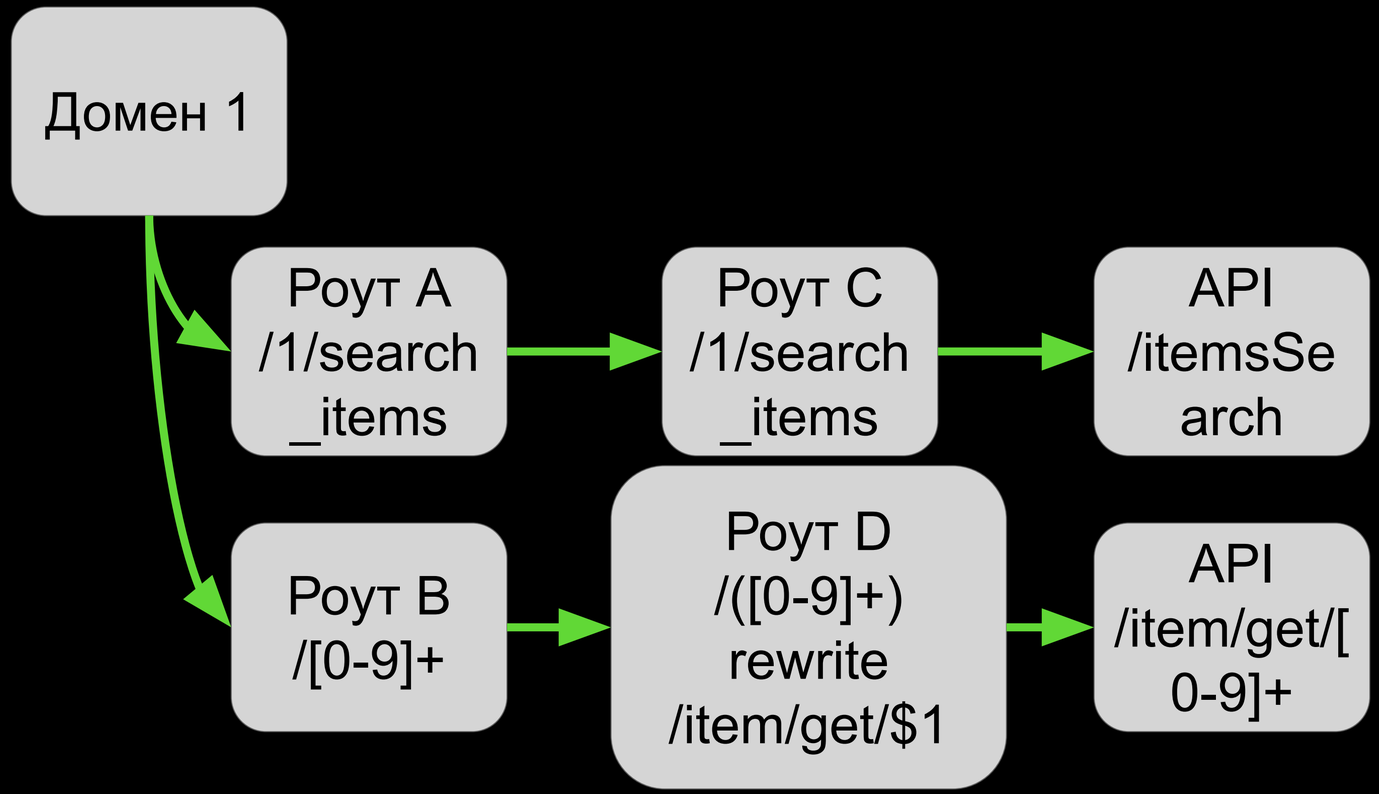


Собираем роутинг воедино: The Core

- Нужен весь путь от вершины до вершины.
- Чтобы видеть историю преобразований пути и получать внешний URL для внутренней ручки API.
- Для каждой вершины **v** нужно построить граф **G**, состоящий из всех вершин которые достижимы из **v**.
- **Да это же задача построения транзитивной редукции!**



https://www.meme-arsenal.com



Результаты

Покрытие в сравнении с логами Nginx
Внешние ручки по основному Авито

97±2%

Покрытие по всем ручкам,
включая внутренние в сравнении
с трейсингом

87±2%

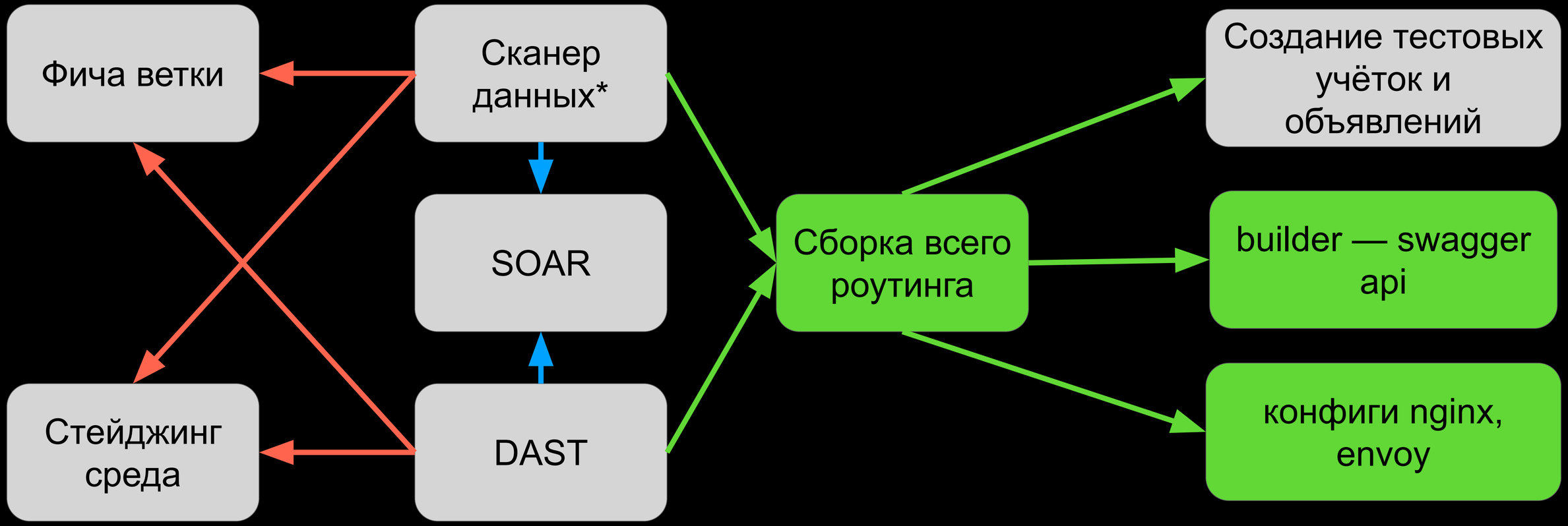
Цифра ниже из-за нескольких
внутренних сервисов с особым
роутингом

Стабильное покрытие парсинга в течение года

Результаты

- Благодаря системе проблемы в новых релизах ищет автоматика, а не пентестер.
- Найдено и пофикшено N расхождений в ratelimit.
- Удалены десятки неиспользуемых API.
- Мы видим появление новых API на периметре и внутри.

Часть корабля



* Смотрите в следующих сериях

The Future

Покрытие по всем внешним доменам

75%

Александр Трифанов

ведущий в безопасное будущее инженер

Спасибо! Вопросы ???