Ускоряем Интернет запросы. Спим спокойно.

Сергей Федоров
DevOops, Санкт-Петербург
30 октября, 2019

NETFLIX

@sfedov

Быстрые запросы - почему это важно?

+100ms = 1%

задержка — падение продаж

A 100 ms latency penalty implies a 1% sales loss.

Amazon, 2009

Liddle, J.: Amazon Found Every 100ms of Latency Cost Them 1% in Sales. http://goo. gl/BUJgV

Быстрые запросы - почему это важно?

+3s = -53%

Загрузка страницы

Трафик пользователей

Starting in July 2018, page speed will be a ranking factor for mobile searches.

Google, 2018

https://webmasters.googleblog.com/2018/01/using-page-speed-in-mobile-search.html

Быстрые запросы - почему это важно?

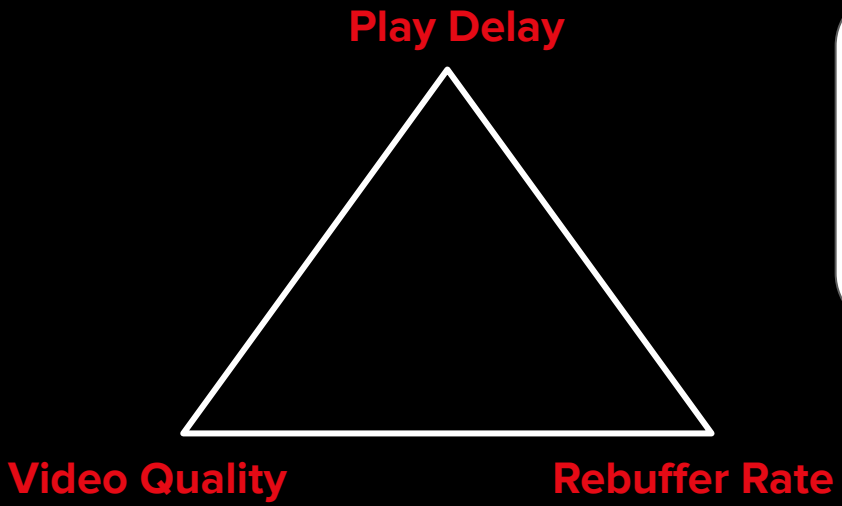
$400M на прокладку кабеля, чтобы уменьшить задержку между Нью-Йорком и Лондоном на **6мс**

$66M за миллисекунду!

The reduction in latency can improve performance, and even improve profitability and increase sales for some customers.

Hibernia Networks, 2015
https://www.submarinenetworks.com/en/systems/trans-atlantic/project-express/hibernia-express-connects-new-york-to-london-in-under-58-95ms

Быстрые запросы - почему это важно?



We study tradeoffs amongst QoE metrics: do members prefer faster playback start with lower video quality or do they prefer to wait a bit longer but start at a higher quality?

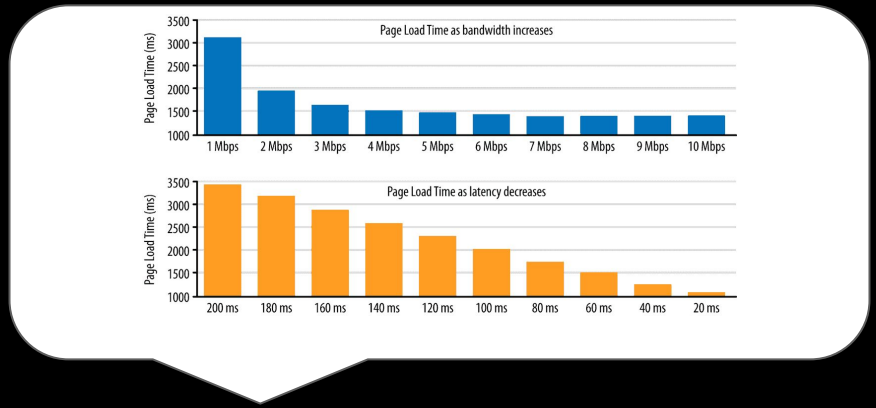
Netflix, 2017

https://medium.com/netflix-techblog/a-b-testing-and-beyond-improving-the-netflix-streaming-experience-with-experimentation-and-data-5b0ae9295bdf

Быстрый Интернет - это вообще что?

Скорость сети выше **5 Mbps** не влияет на время загрузки типового сайта.

Зависимость между задержкой и временем загрузки сайта линейна.



Mike Belshe
https://hpbn.co/primer-on-web-performance/

О чем будем разговаривать?

- Как уменьшить задержку (latency) Интернет запросов.
- Как Netflix ускоряет запросы.
- О хорошем сне: operational excellence.

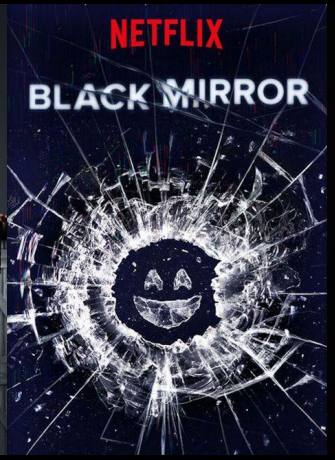
Что вы узнаете?

- МатЧасть: как сетевые протоколы влияют на задержку.
- Практический подход к решению задачи в стиле Netflix.
- Как проектировать, поддерживать и мониторить fault-tolerant системы.

Netflix: фильмы и сериалы

Stranger Things

House of Cards

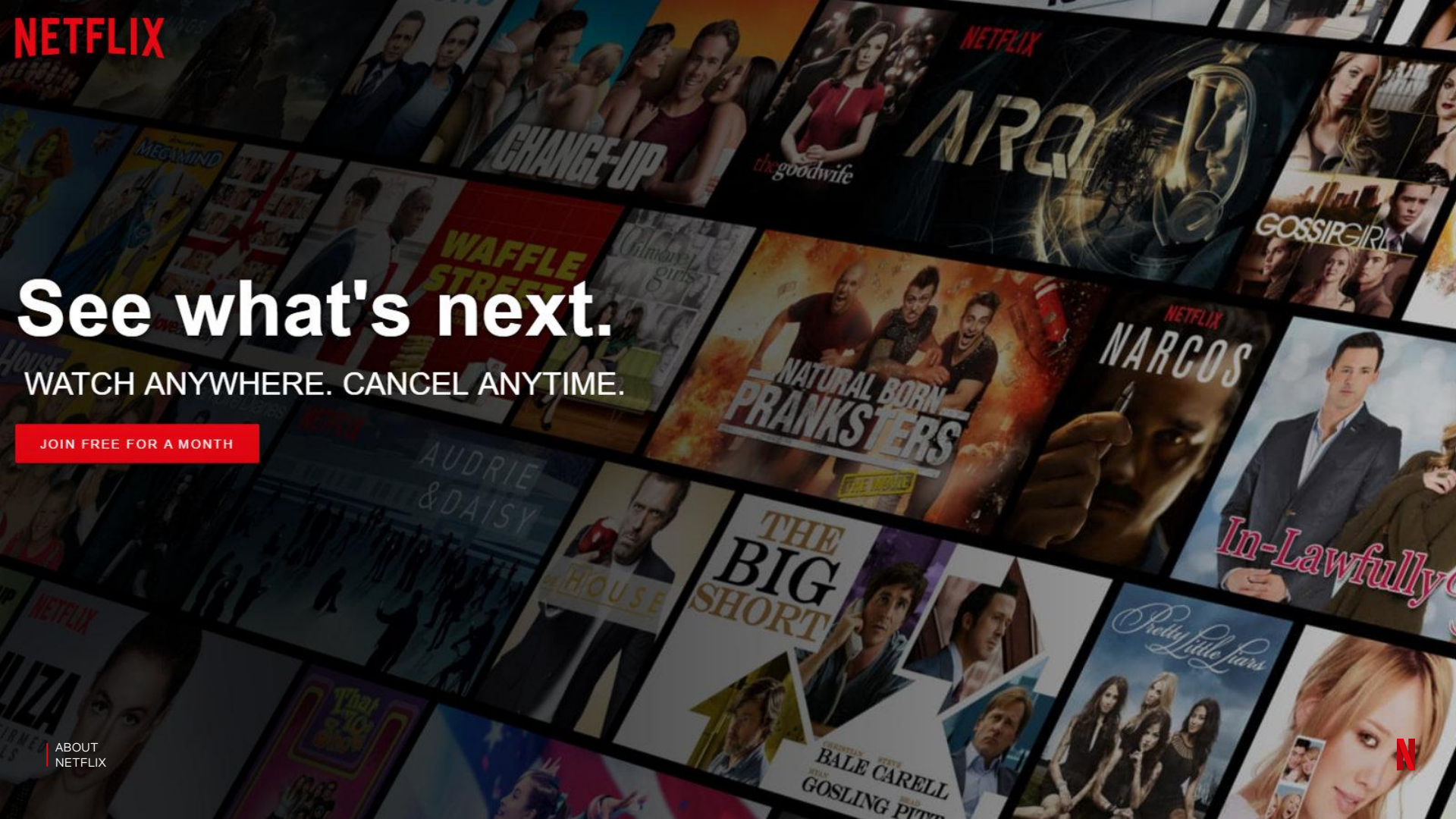


Black Mirror

El Camino
A Breaking Bad Movie

Daredevil

The Crown


NETFLIX
See what's next.
WATCH ANYWHERE. CANCEL ANYTIME.
JOIN FREE FOR A MONTH
ABOUT NETFLIX

Внутри Netflix

INSIDE
NETFLIX

Netflix Client

Тысячи типов устройств

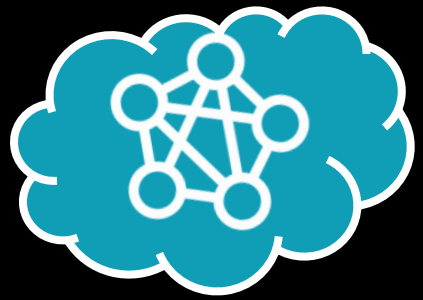
Персонализированный UI

4 отдельные команды:

- iOS, Android, TV, Web

Сотни AB тестов

HTTP APIs



AWS Cloud

Сотни микросервисов

Персонализация

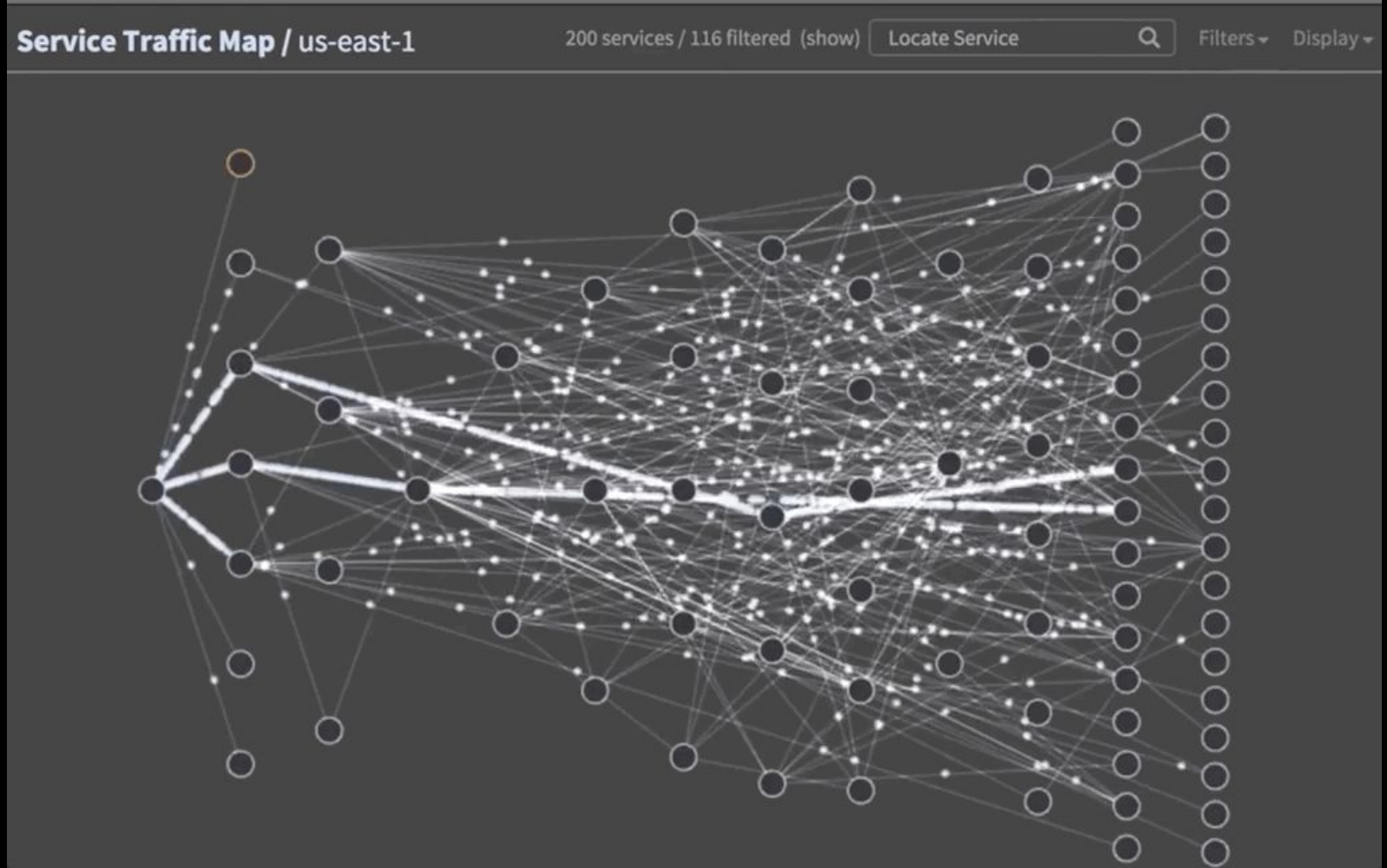
Control plane

Telemetry

Big Data

Encoding

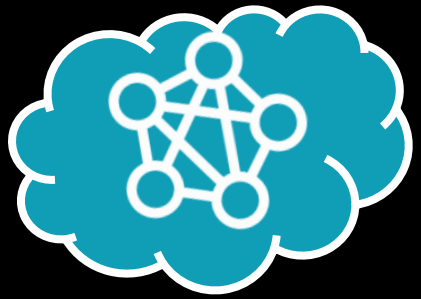
VIZCERAL


Service Traffic Map / us-east-1
200 services / 116 filtered (show)
Locate Service
Filters
Display

Netflix CDN

HTTP APIs

CDN



Видео Стриминг

Статические Объекты

DNS

Netflix CDN
Open Connect

OCA

Open Connect Appliance

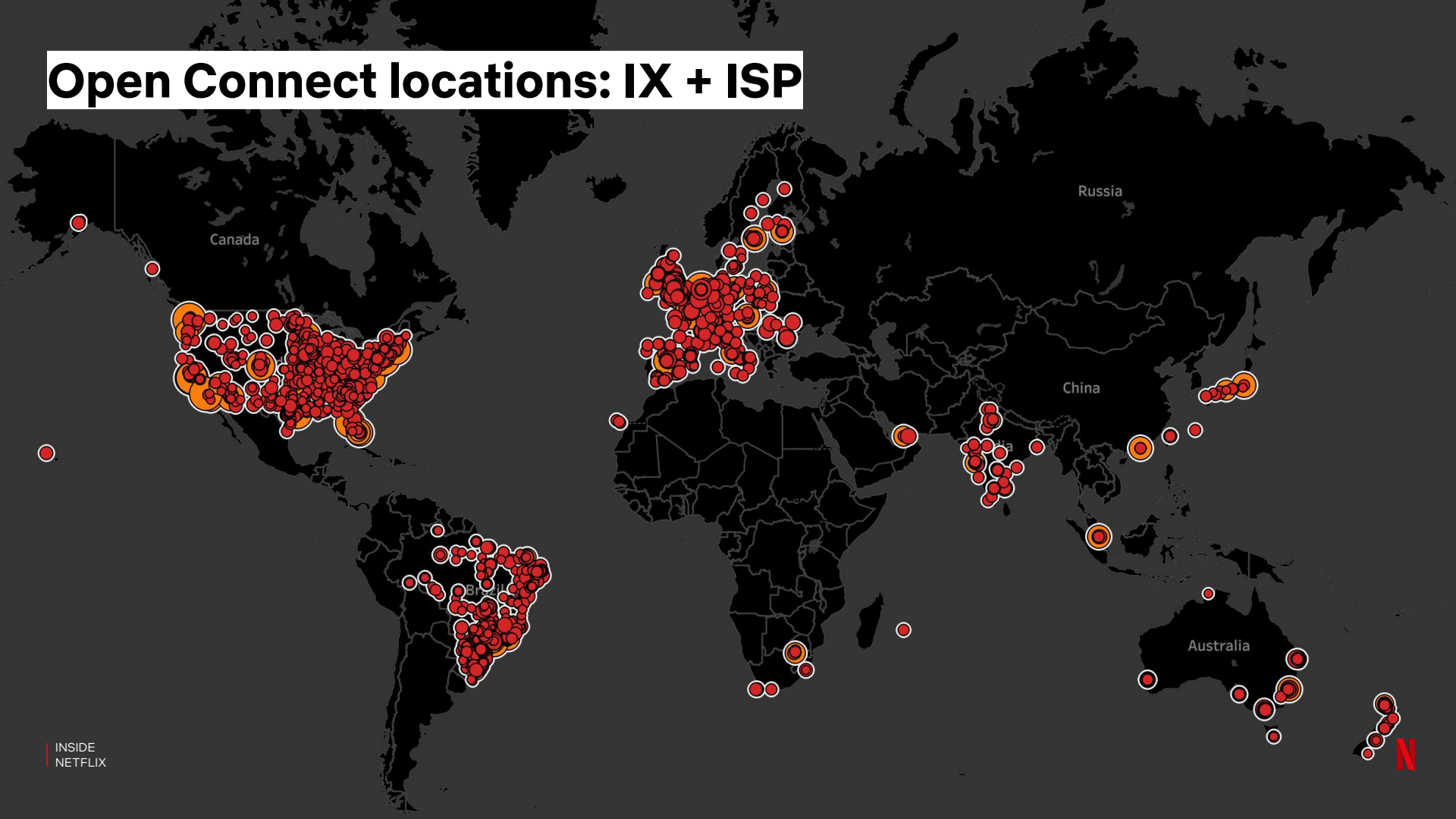
CDN Edge Server:

- FreeBSD
- Nginx
- Lots of SSDs/HDDs
- Optimized for max network throughput

Такая стенка может доставлять около 1% Интернет трафика.

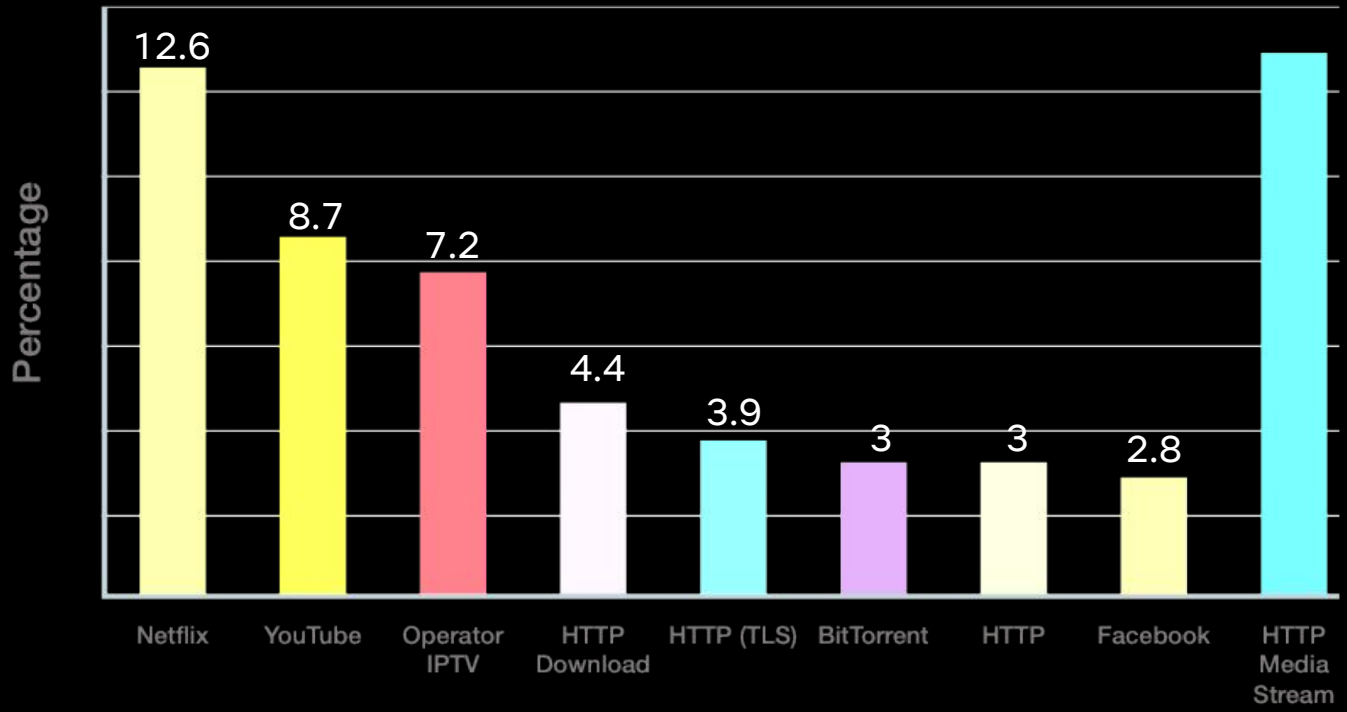
Open Connect locations: IX

Open Connect locations: IX + ISP



Open Connect Backbone Network

Sandvine report on Netflix traffic



12.6% of worldwide downstream traffic

Source: Sandvine 1H2019 Global Internet Phenomena Report

The Open Connect Team

Немного обо мне

- ННГУ им. Лобачевского’12
- Microsoft Imagine Cup’09
- Past: Intel/Microsoft
- Netflix, 7 years
  - CDN Monitoring
  - FAST.com
  - API acceleration
  - QoE optimization

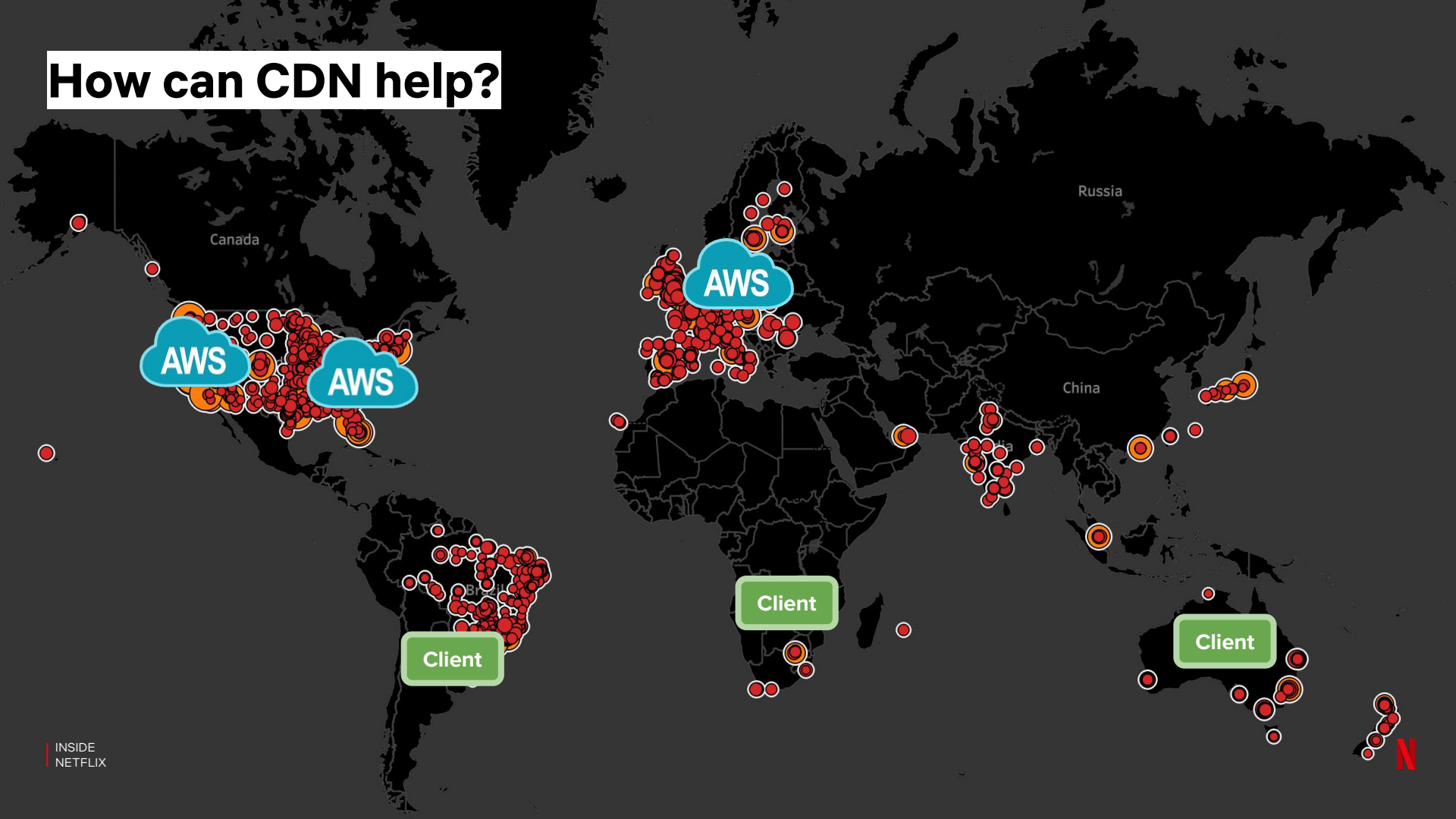
INSIDE
NETFLIX

Так что там про ускорение Интернета?

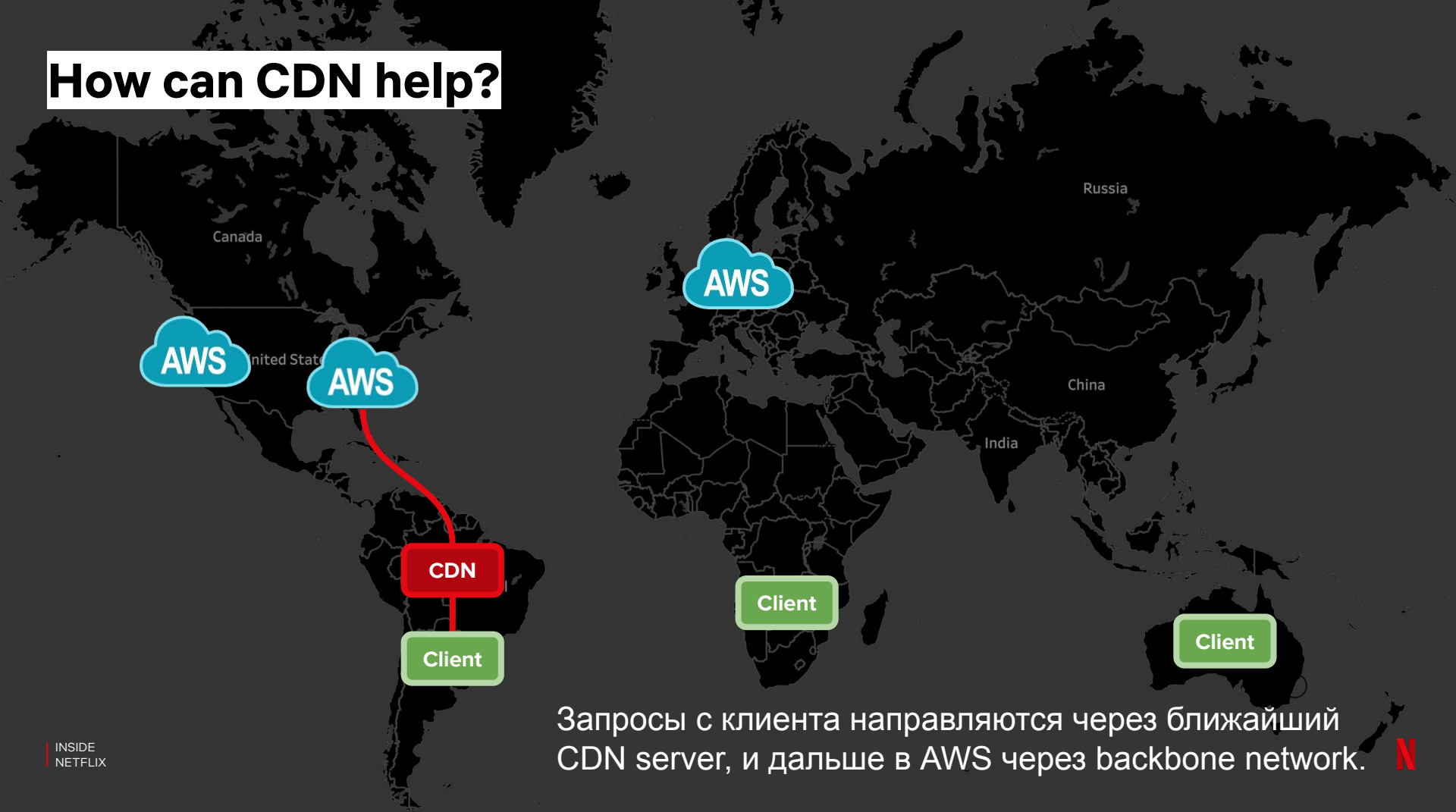
INTERNET
ACCELERATION

Netflix: 3 AWS regions

How can CDN help?



How can CDN help?



Запросы с клиента направляются через ближайший CDN server, и дальше в AWS через backbone network.

Может ли CDN прокси ускорить запросы в облако?

Да

Нет

Может ли CDN прокси ускорить запросы в облако?

Да

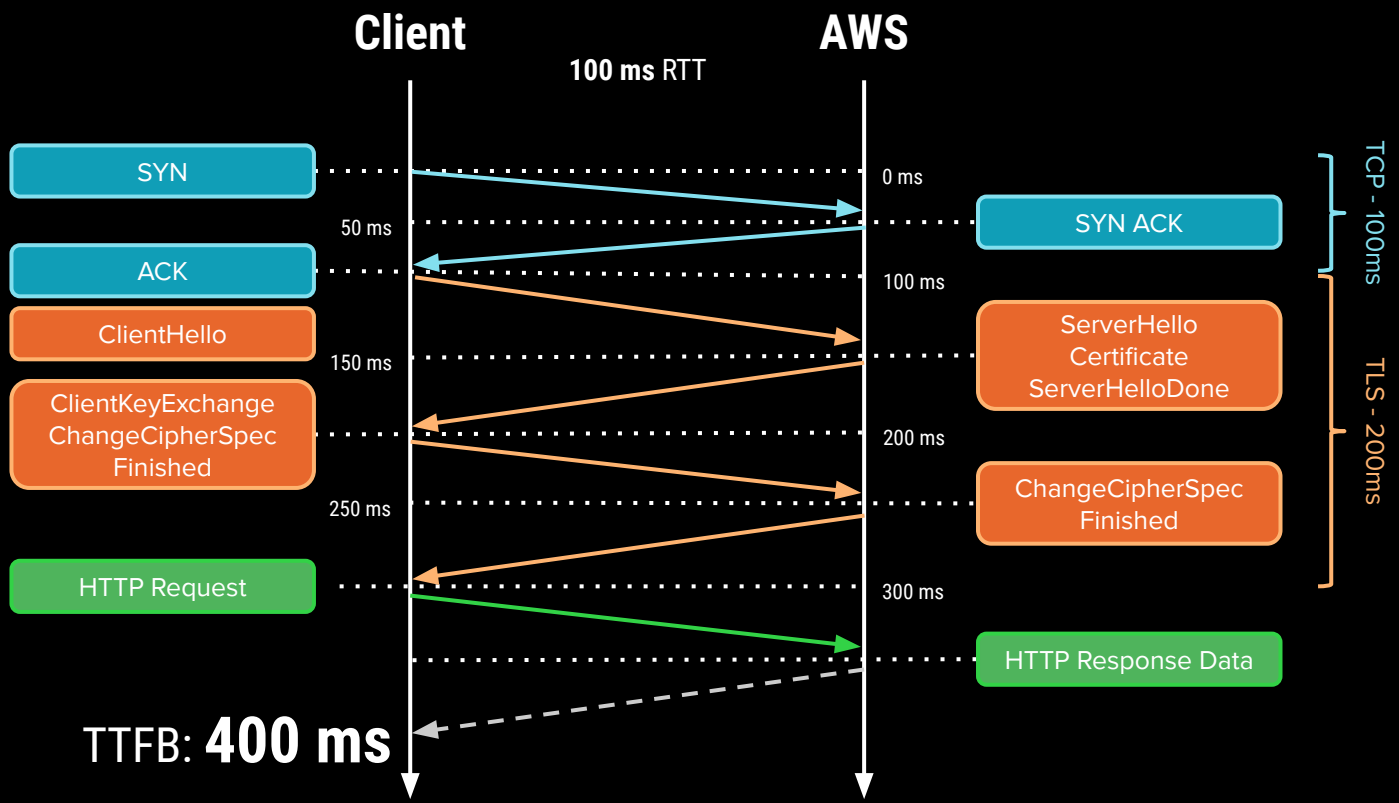
Нет

Может ли CDN прокси ускорить запросы в облако?

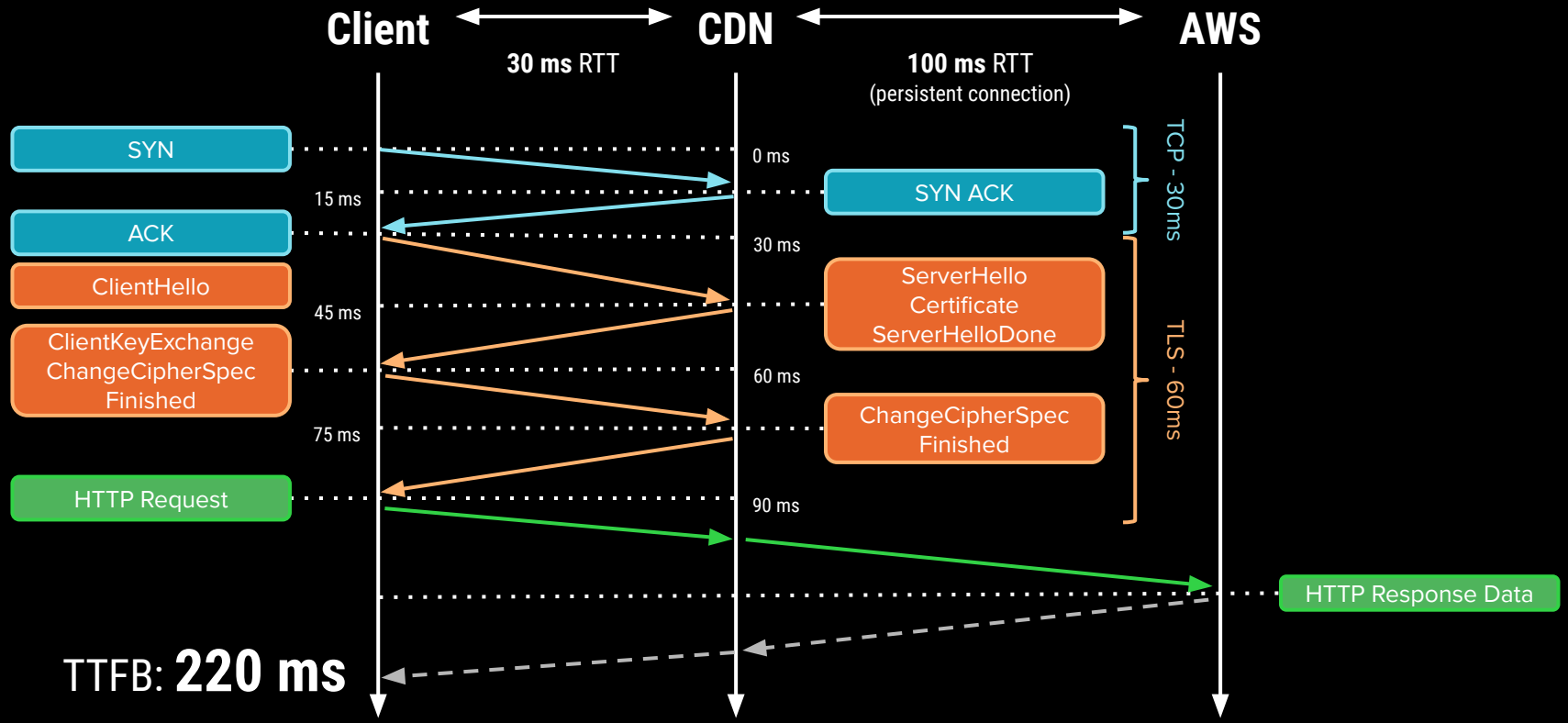
Да

Нет

Матчасть: TCP + TLS

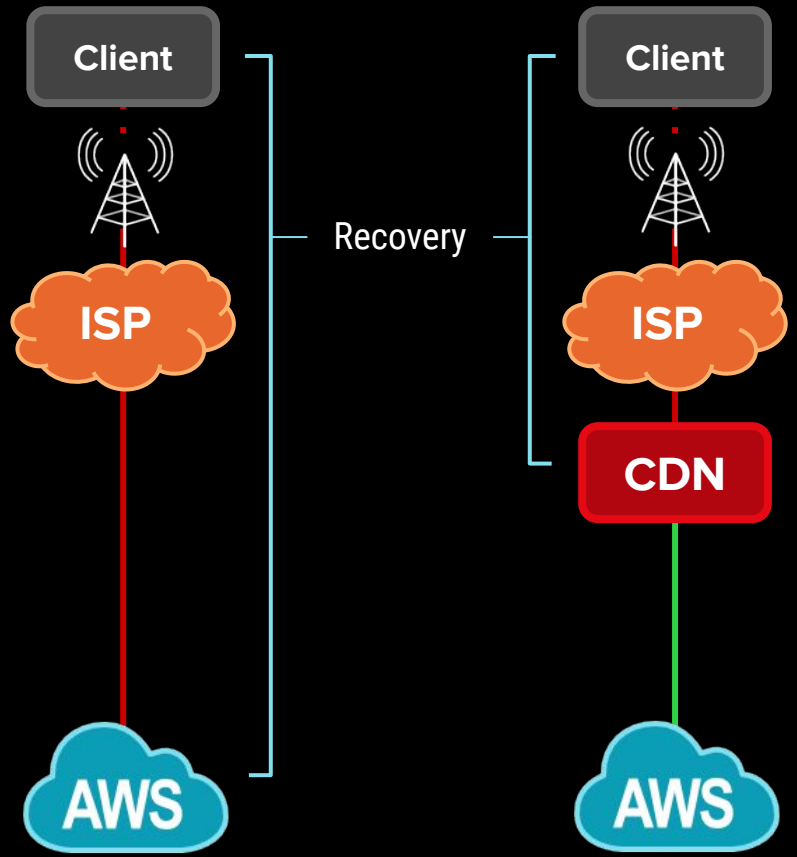


Матчасть: TCP + TLS



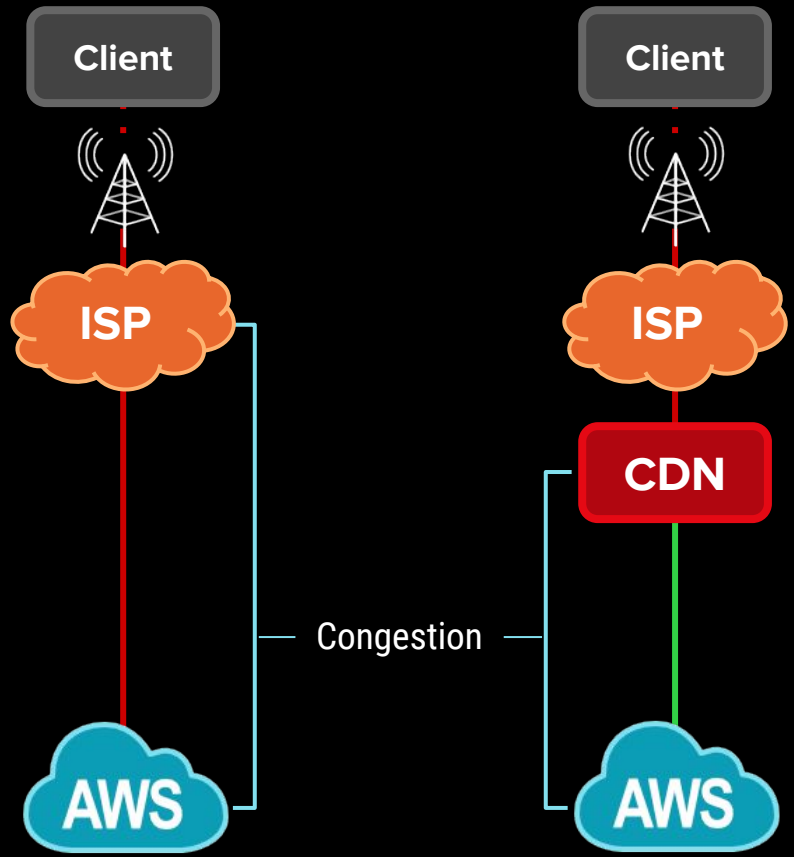
Матчасть: TCP Loss Recovery

- Потери данных чаще всего происходят в домашней сети (last mile): WiFi, Traffic Shaping, плохой роутер.
- TCP урезает congestion window на каждую потерю пакета:
  - 1+ RTT, чтобы обнаружить потерю
  - 1.5+ RTT, чтобы восстановить
- Чем ниже RTT, тем меньше влияние потерь.

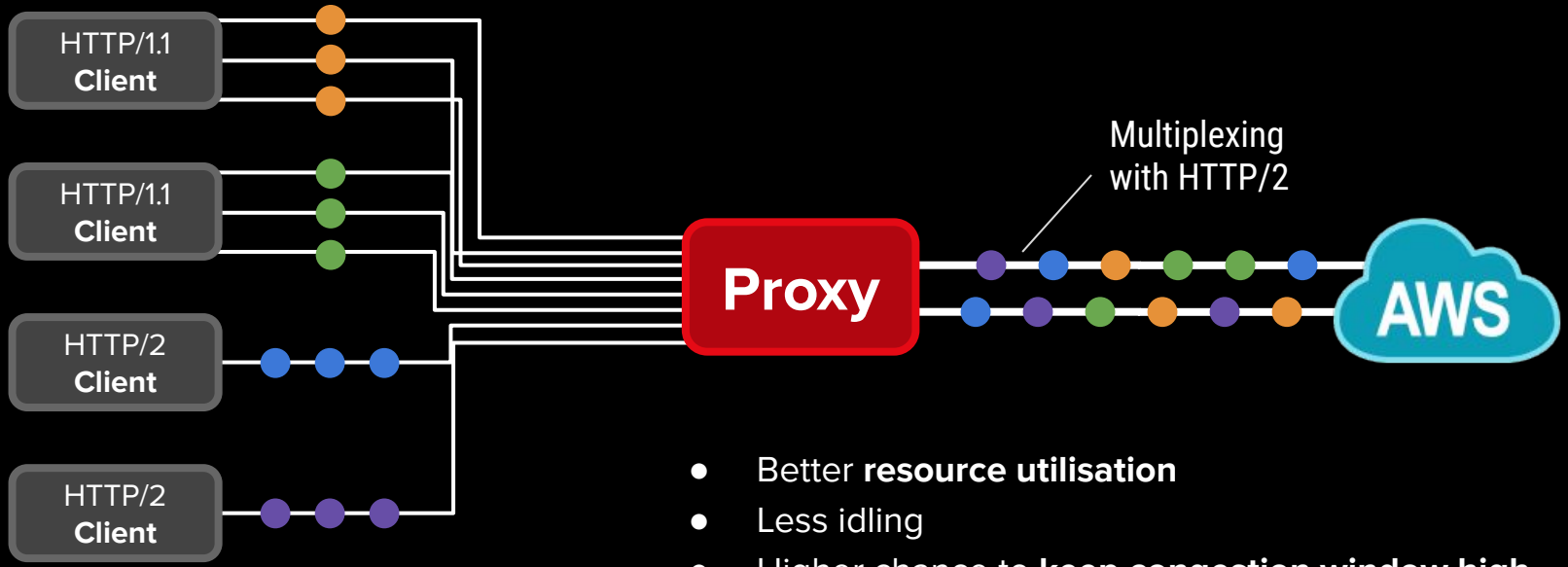


Матчасть: Internet Congestion

- “Долгая” часть RTT - от Интернет провайдера до AWS.
- “Пробки” на соединениях в Интернете, особенно в час-пик.
- Конкуренция с “тяжелым” трафиком - стриминг, обмен файлами. Отсутствие контроля над конфигурацией провайдеров.
- Backbone сеть позволяет настроить QoS по типу трафика.



Матчасть: HTTP 2 (+)

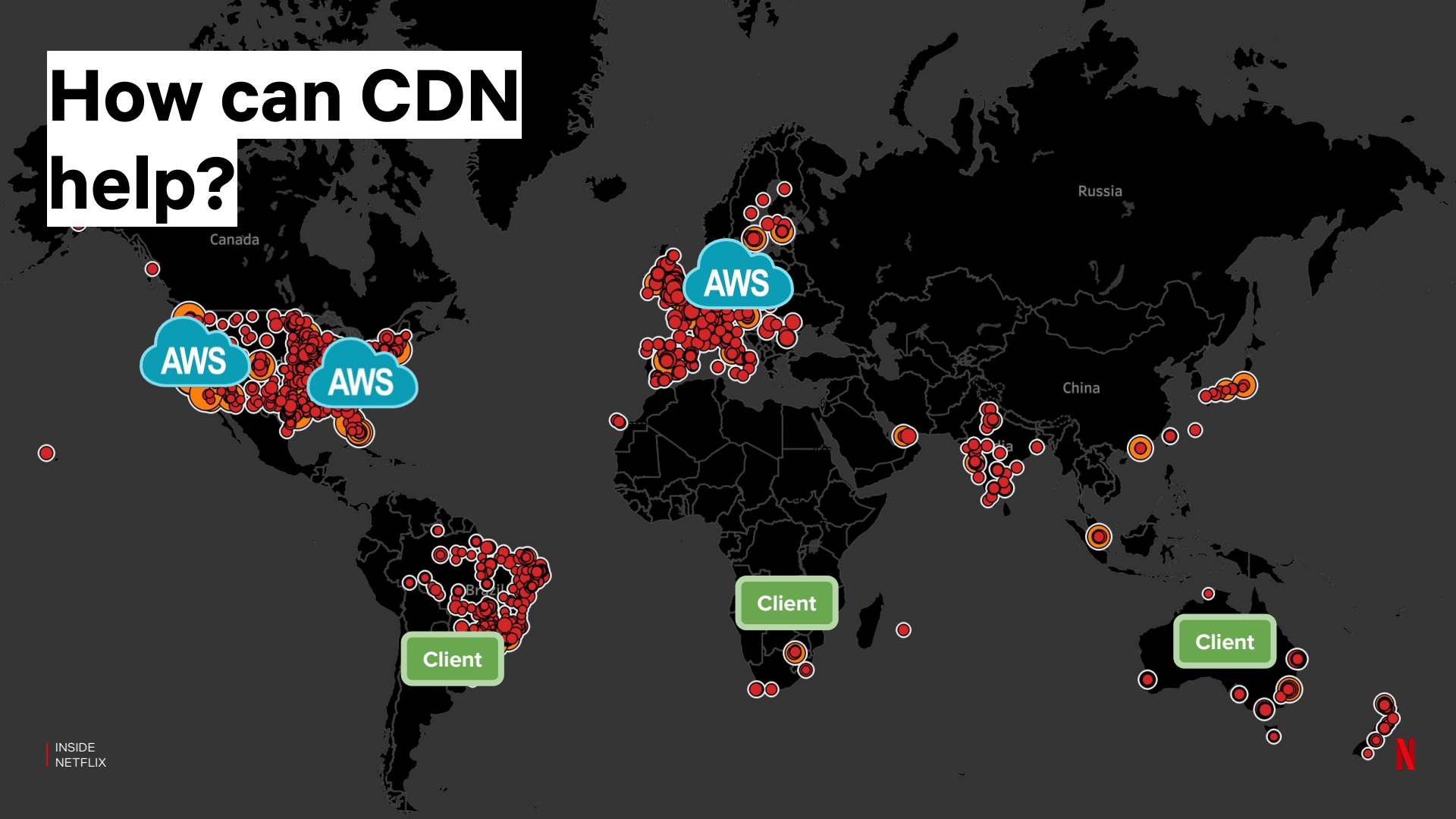


- Better **resource utilisation**
- Less idling
- Higher chance to **keep congestion window high**
- Out of the box **header compression**
- **No head-of-line blocking** at HTTP layer

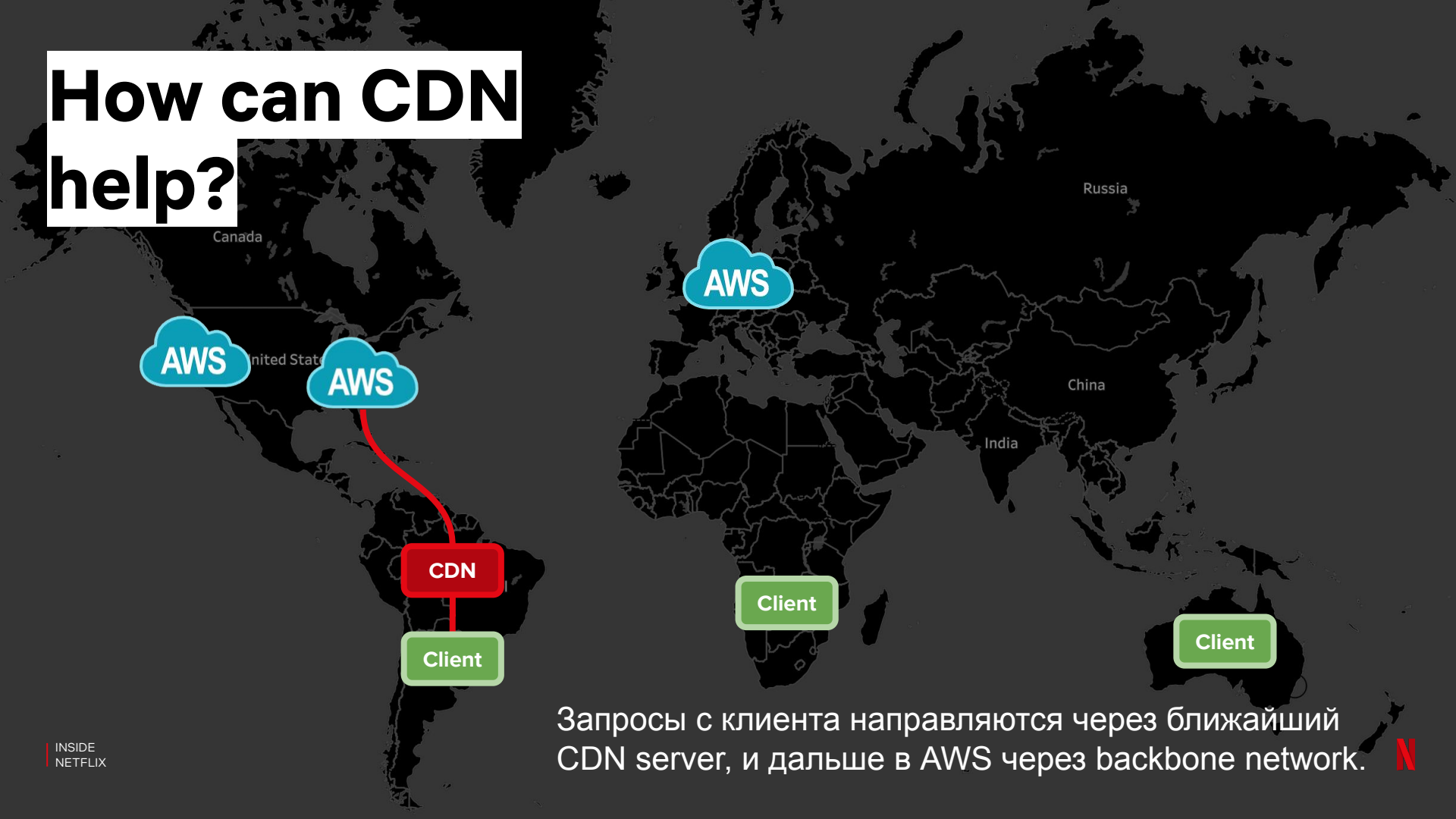
От теории к практике.

INTERNET
ACCELERATION

How can CDN help?



How can CDN help?

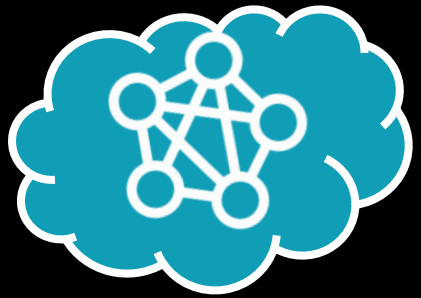


Запросы с клиента направляются через ближайший CDN server, и дальше в AWS через backbone network.

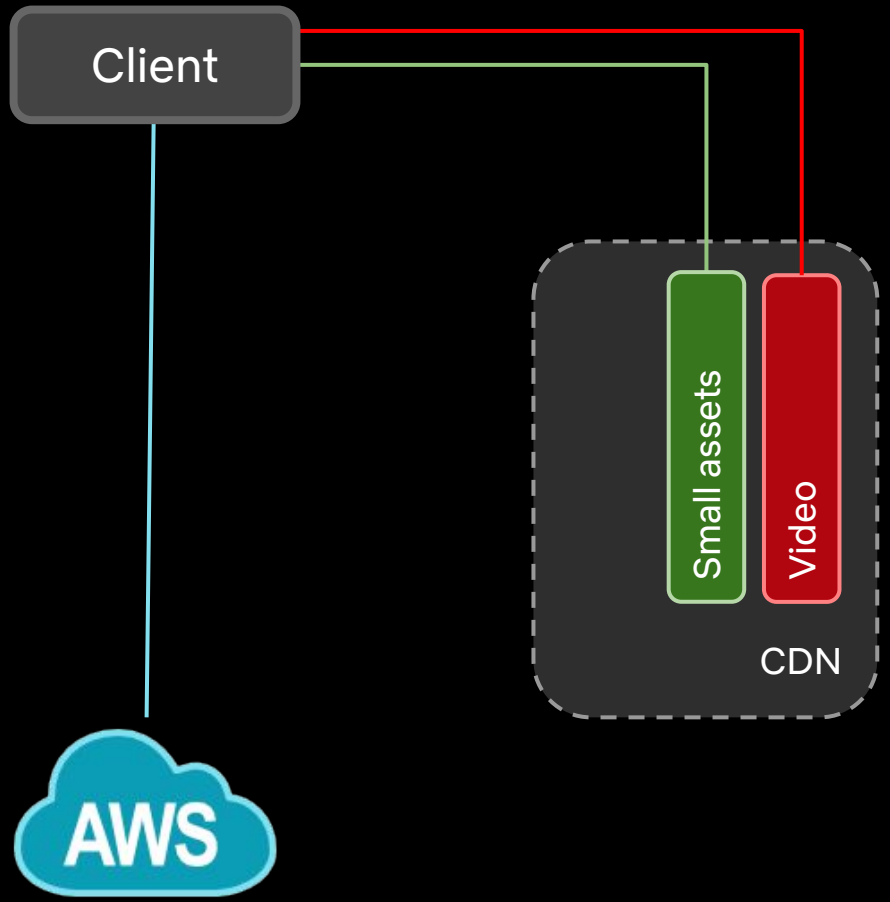
Традиционная архитектура

HTTP APIs

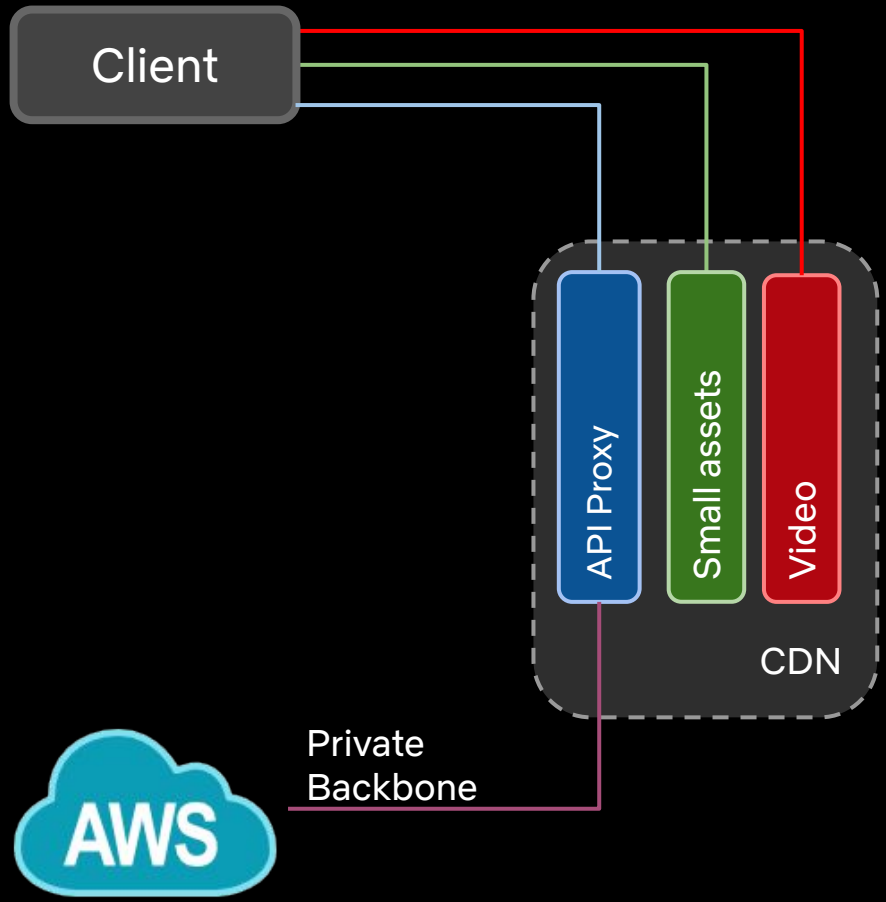
CDN



Традиционная архитектура



Идея: reverse proxy на CDN серверах



Может ли CDN прокси ускорить запросы в облако?

Да

Нет

Может ли CDN прокси ускорить запросы в облако?

Да

Нет

Может ли CDN прокси ускорить запросы в облако?

Да

Нет

Измеряй.

Не гадай -
измеряй

На какие вопросы нужно ответить?

- **Скорость:** будет ли прокси быстрее?
- **Надежность:** будет ли чаще ломаться?
- **Сложность:** как интегрировать с приложениями?
- **Стоимость:** стоимость дополнительной инфраструктуры?
- ...

На какие вопросы нужно ответить?

- **Скорость:** будет ли прокси быстрее?
- Надежность: будет ли чаще ломаться?
- Сложность: как интегрировать с приложениями?
- Стоимость: стоимость дополнительной инфраструктуры?
- ...

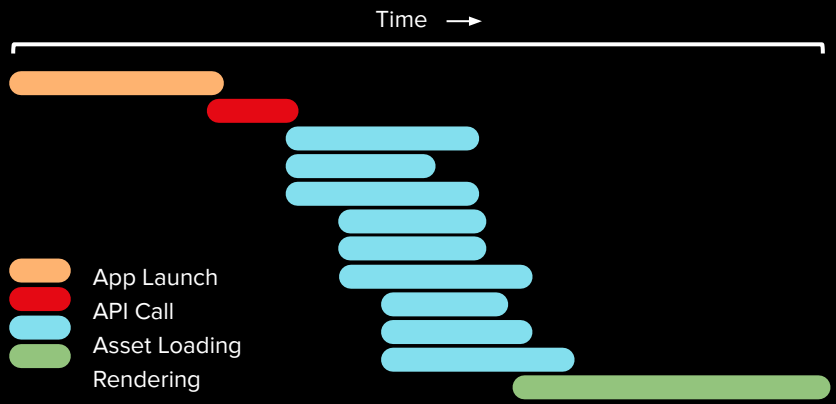
Идеальный результат

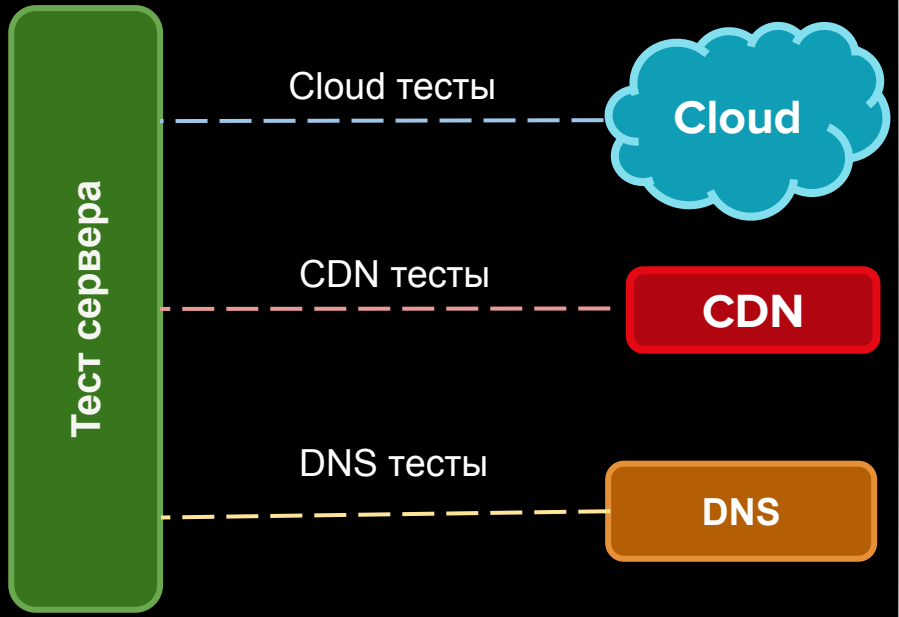
- Точная оценка от текущих пользователей
- Полное покрытие устройств
- Без лишних затрат на разработку
- Не сломать production

RUM: Пассивное измерение запросов.

100% гео
100% устройств

Нестабильный сигнал
Только production трафик





Лабораторные тесты.

Полный контроль
Четкий сигнал

Неполный гео
Неполное покрытие устройств

Пробы на клиенте = RUM + Лаб

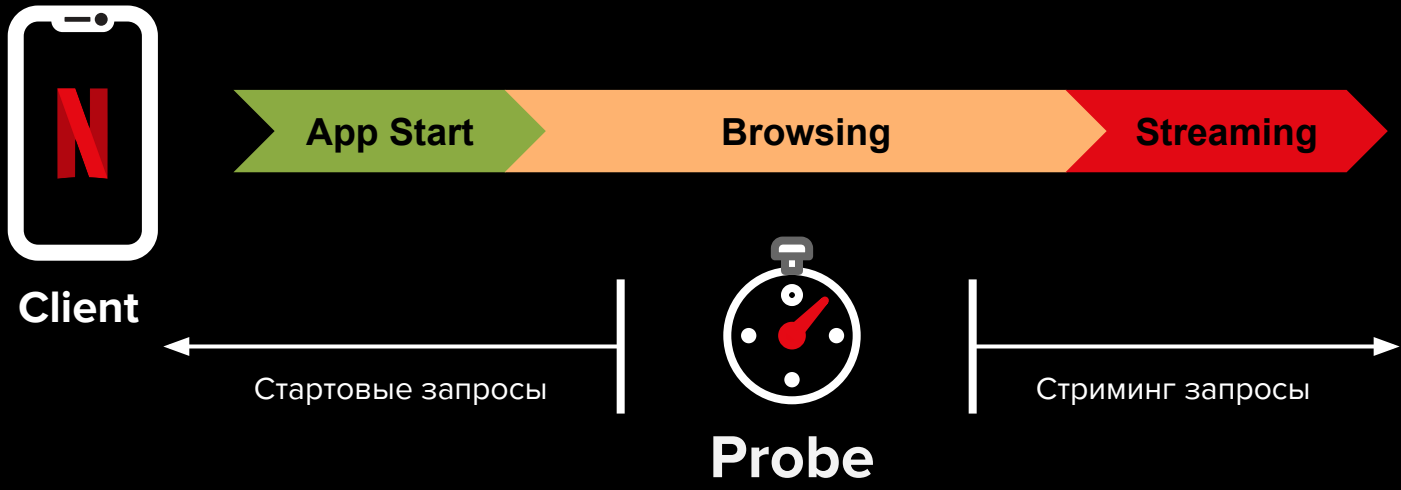
Probe

Получаем рецепт

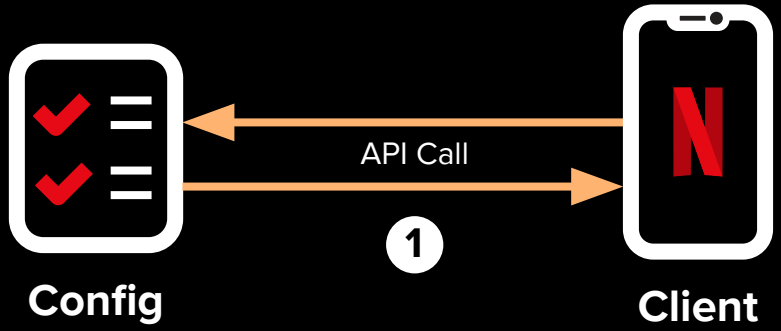
Запросы и измерения

Репорт с результатами

Workflow


App Start
Browsing
Streaming
Client
Стартовые запросы
Probe
Стриминг запросы

Шаг 1: клиент получает рецепт



Рецепт описывает что нужно измерить.

Config

```
type: HTTP GET
name: reachability test
targets:
  http://target1.test.me/probe
  http://target2.test.me/probe
  http://target3.test.me/probe
payload: 5KB
pulses: 3
delay: 5s
```

Рецепт описывает что нужно измерить.

Config

```
type: HTTP GET
name: reachability test
targets:
  http://target1.test.me/probe
  http://target2.test.me/probe
  http://target3.test.me/probe
payload: 5KB
pulses: 3
delay: 5s
```

```
type: HTTPS GET
name: large payload test
targets:
  https://target1.test.me/probe
  https://target2.test.me/probe
  https://target3.test.me/probe
payload: 100KB
pulses: 3
delay: 10s
```

```
...
```

Можно тестировать несколько тестов параллельно.

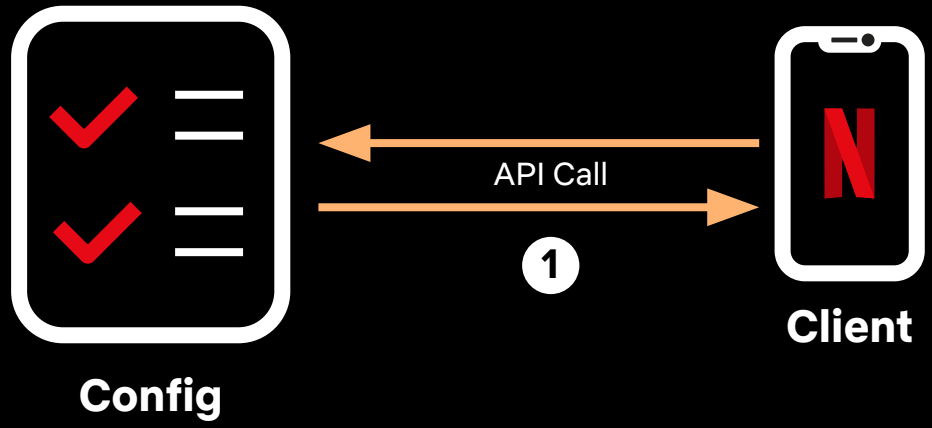
Выбираем на основе “веса” каждого теста

20% Reachability Test

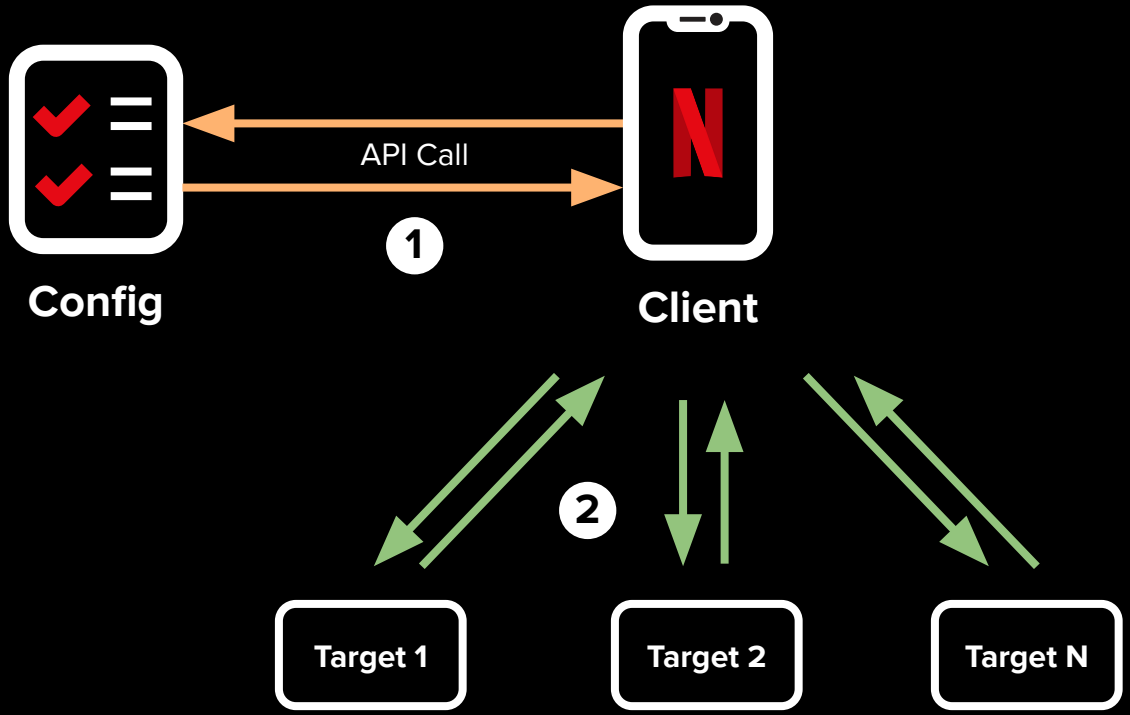
10% DNS Providers Test

1% CDN Test

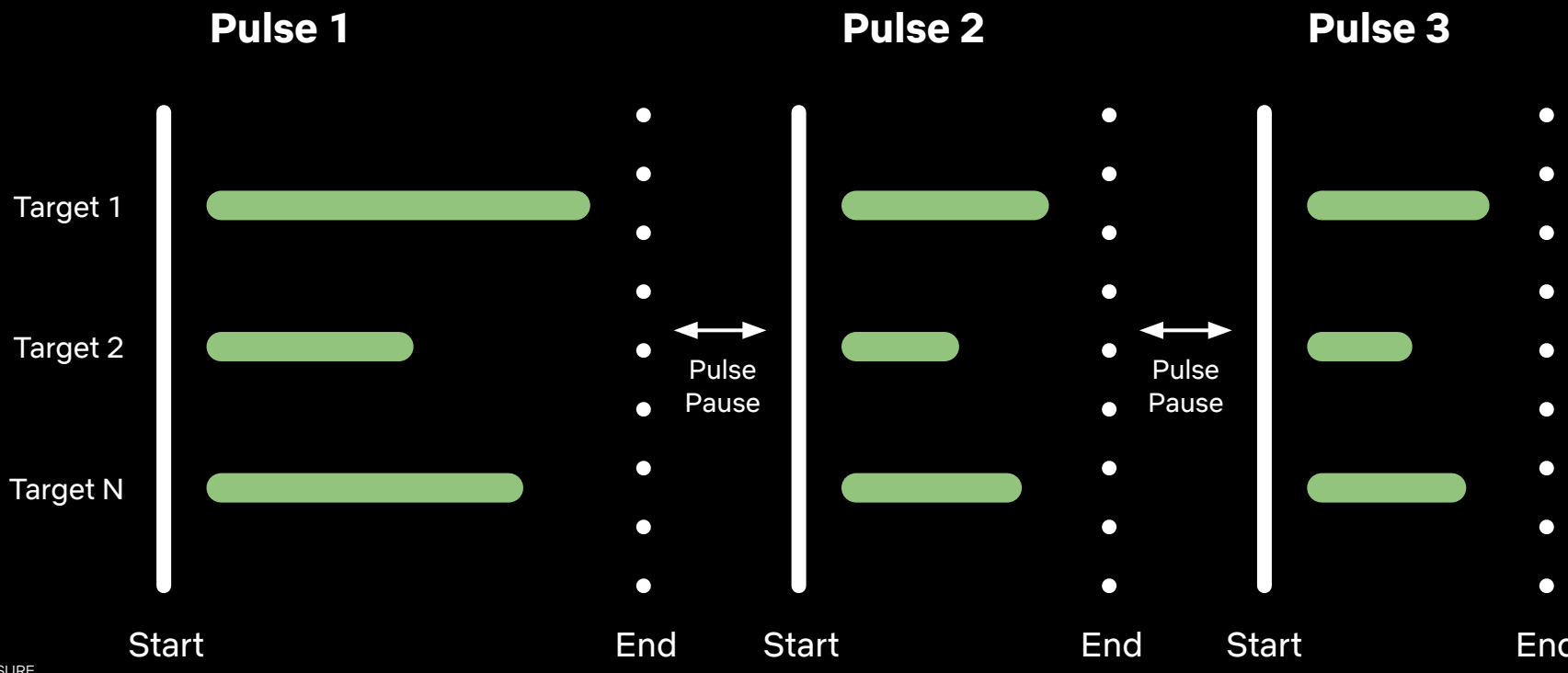
2% Large Payload Test



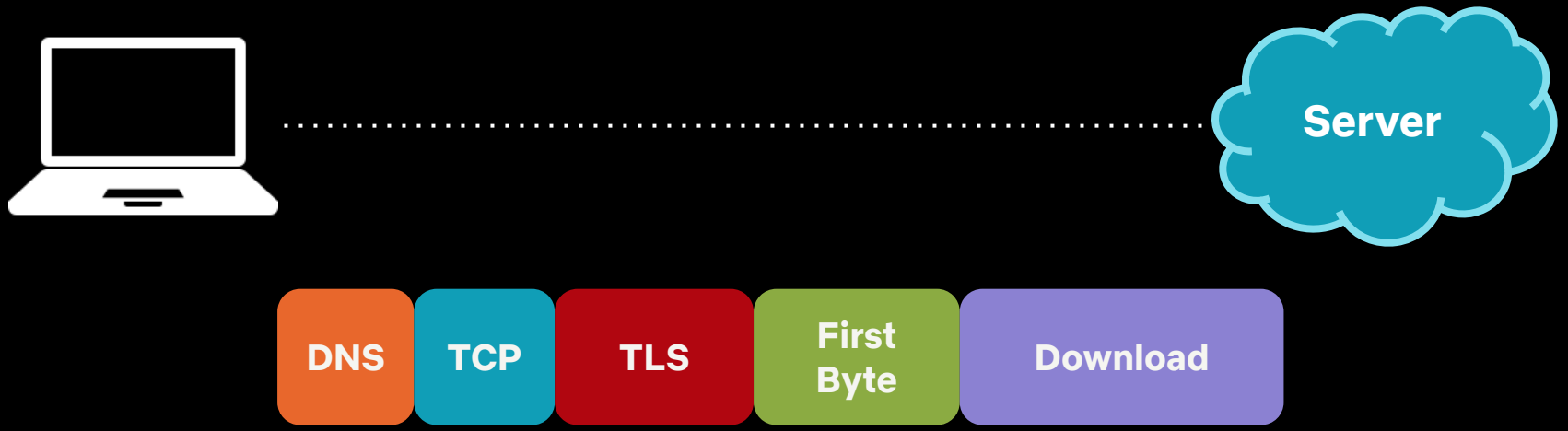
Шаг 2: делаем запрос и измеряем.



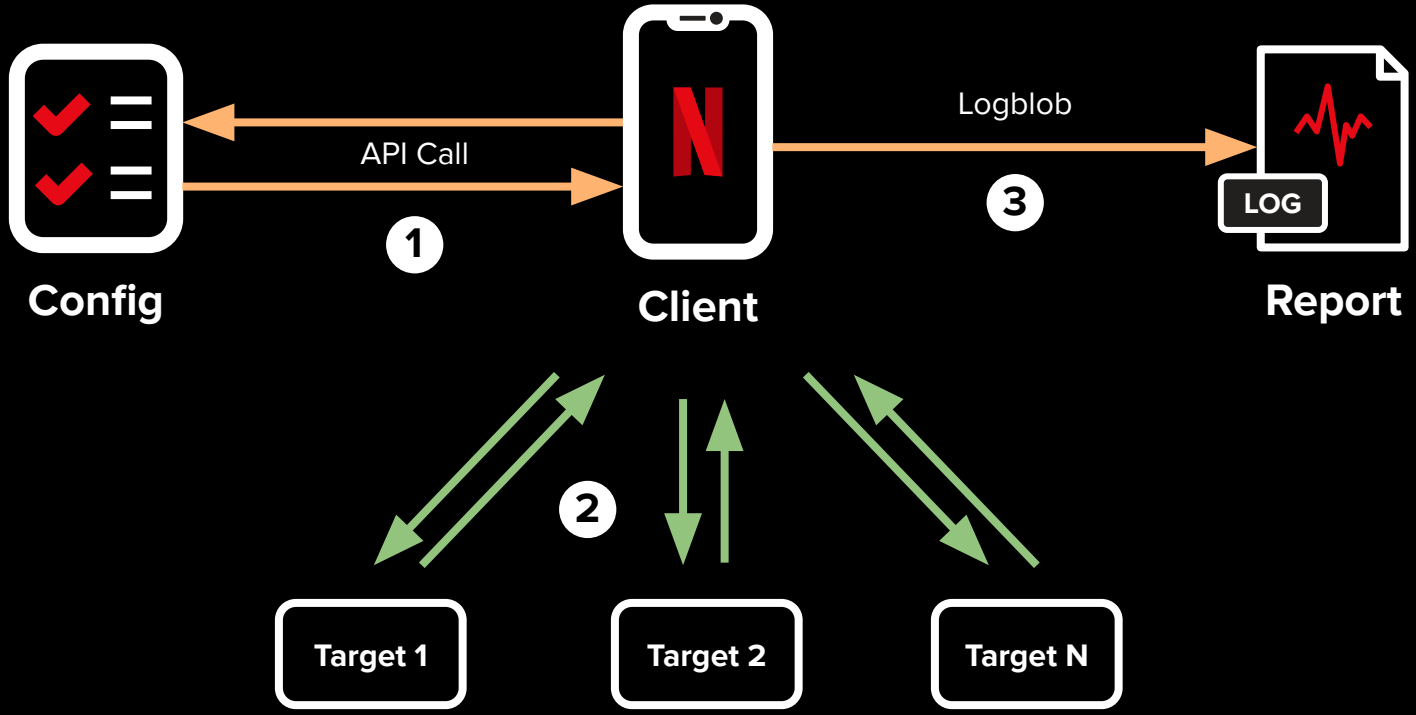
Запросы и пульсы.



Метрики.


Server
DNS
TCP
TLS
First Byte
Download
Response Code: OK / FAIL / ...

Шаг 3: загружаем результаты



Пробы в Netflix.

6K+
проб в секунду

14
рецептов

1K+
устройств

100M+
локаций

$1M+ в год
за услуги вендоров

Пробуй.

Доверяй, но
проверяй.

Что будем измерять?
Нужен прототип будущей системы.

- **Proxy:** надо создать прототип на CDN.
- **Steering:** как выбрать CDN сервер для клиента?
- **Comparison:** сравниваем AWS-direct запросы с proxy.

Proof-of Concept: CDN Reverse Proxy

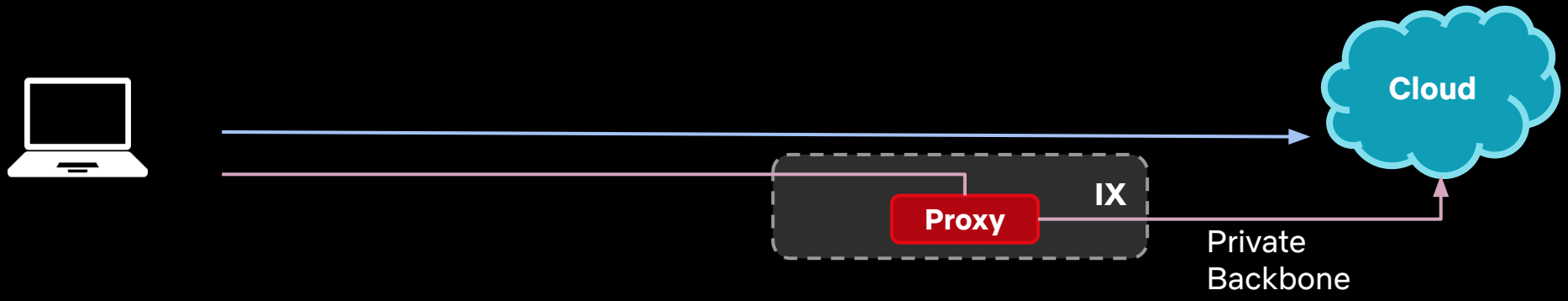
- Go-based
- На каждом CDN, static binary
- HTTP2 connection pooling and request multiplexing
- Просто, как велосипед: базовая функциональность, используем стандартные компоненты.

Client to AWS Routing Options: Direct Link

Cloud Steering:

- 3 AWS региона
- geoDNS для балансировки

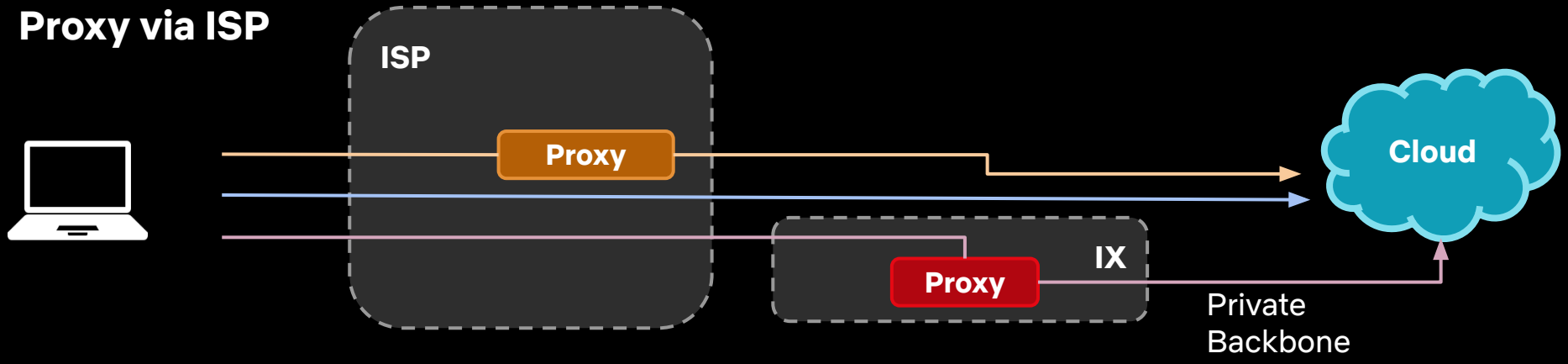
Client to AWS Routing Options: Proxy via IX CDN



IX CDN Steering:

- TCP Anycast
- Single IP
- 70+ sites

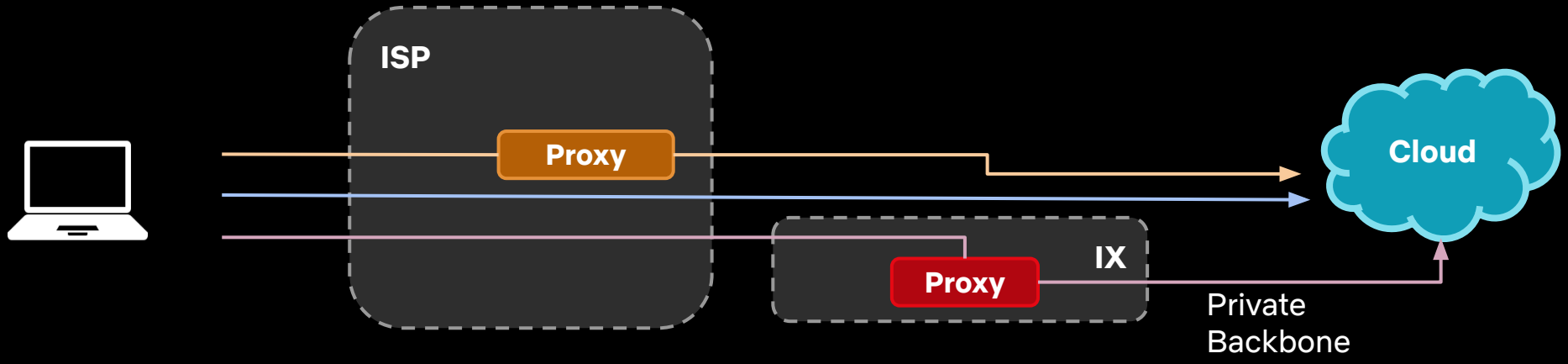
Client to AWS Routing Options: Proxy via ISP



ISP CDN Steering:

- Тот же сервер что используется для видео
- 1K+ sites

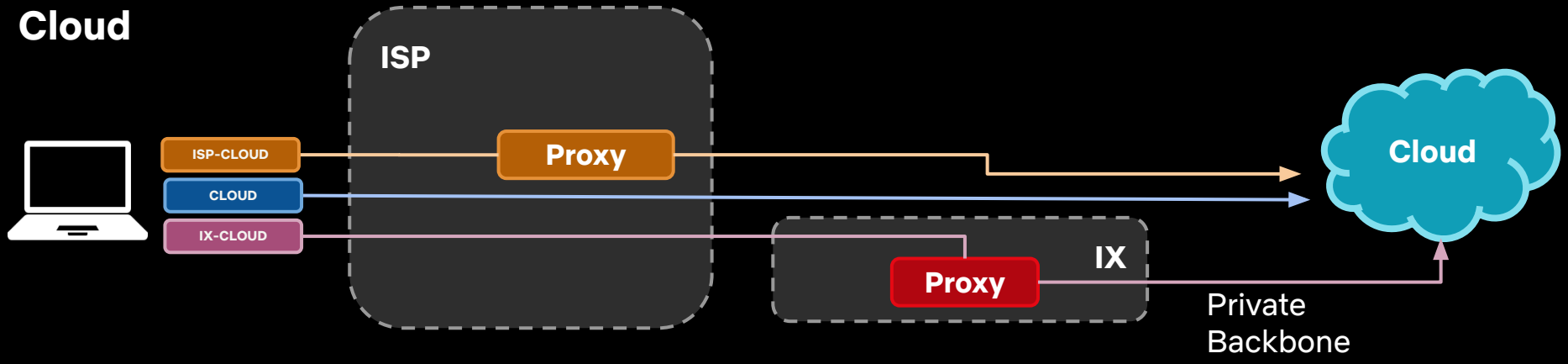
Client to AWS Routing Options



Какой путь быстрее?

Какой путь более надежен?

3 Network Paths to Reach the Cloud




```
type: HTTP GET
name: steering test
targets:
  cloud.test.me/probe
  ix-cloud.test.me/probe
  isp-cloud.test.me/probe
```

→

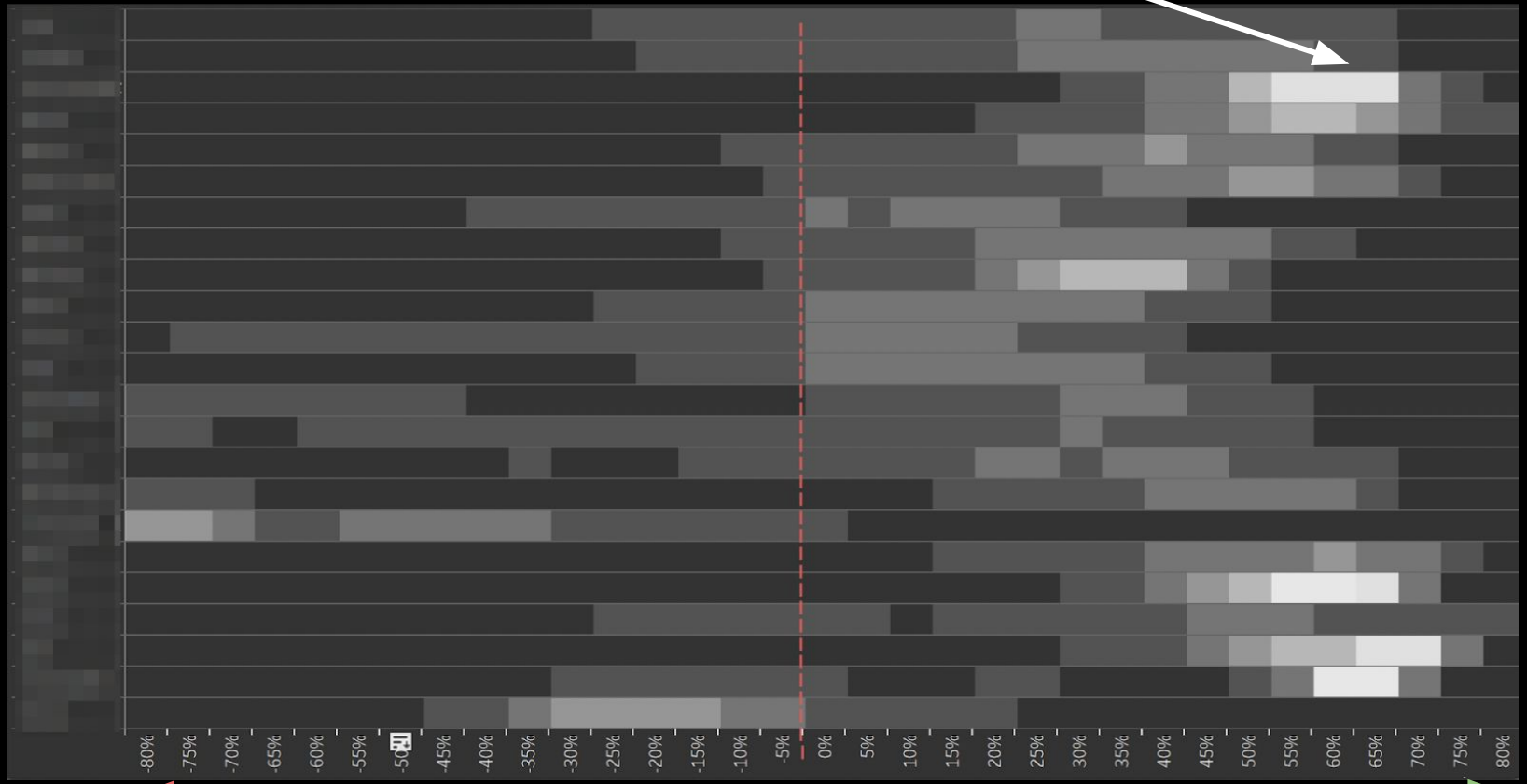
```
cloud:        time / OK / FAIL
ix-cloud:     time / OK / FAIL
isp-cloud:    time / OK / FAIL
```

Может ли CDN прокси ускорить запросы в облако?

Да

Нет

Где **светлее** = **больше** клиентов



Proxy **быстрее**

Proxy **медленнее**

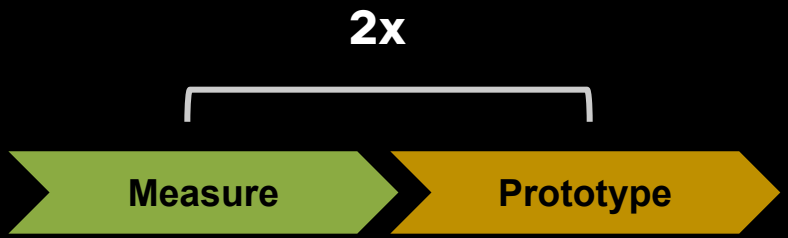
INTERNET
ACCELERATION

Обобщаем.

- Простая proxy недостаточна.
- Оценили ожидаемую производительность с клиентов.
- **Не трогали (не ломали) production.**

Пробуй.

Доверяй, но проверяй.



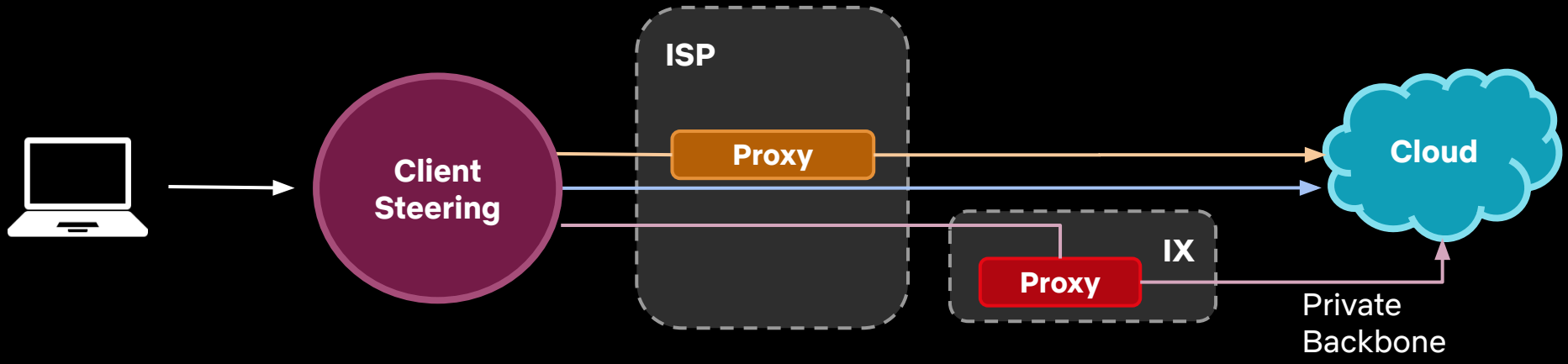
Нету однозначного быстрого решения. Нужно делать steering.

Goal: надо выбрать самый быстрый путь из трех опций:

- AWS-direct
- IX-proxy (TCP Anycast)
- ISP-proxy (Unicast)

Решение: Client Steering

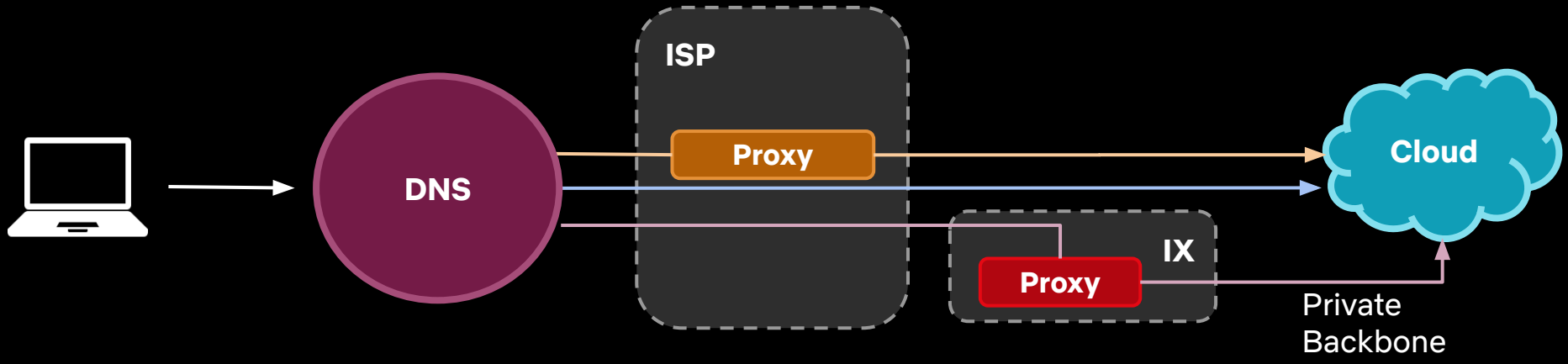
Client Steering



Цель: выбрать самый быстрый путь:

- Cloud vs IX vs ISP
- Без дополнительных API запросов
- Минимум изменений на клиенте

Client DNS Steering.

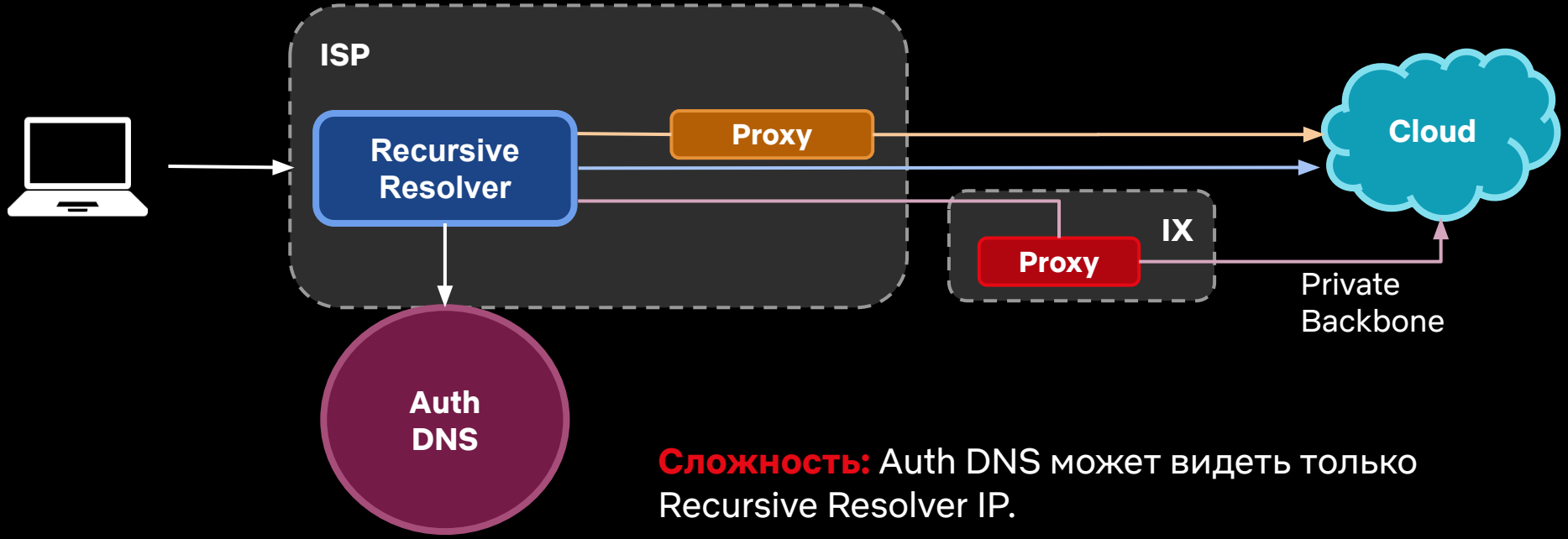


Решение: DNS

api.netflix.com -> IP сервера по самому быстрому пути.

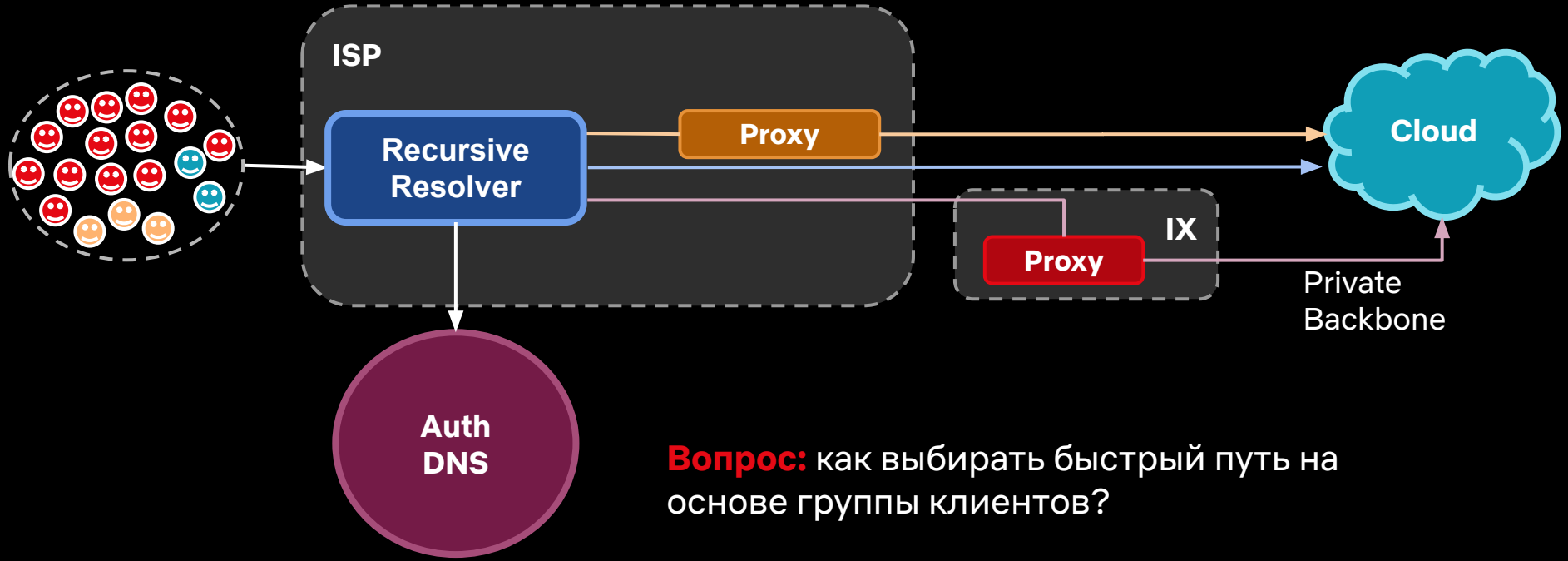
- Cloud : AWS Region IP
- IX : Anycast IP
- ISP : Site Unicast IP

DNS Resolver Based Steering.



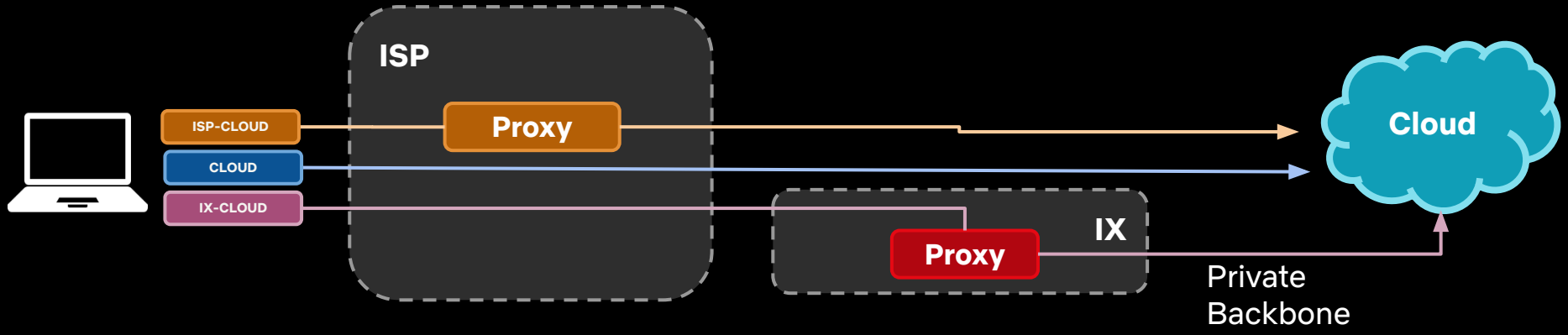
Сложность: Auth DNS может видеть только Recursive Resolver IP.

DNS Resolver Based Steering.



Вопрос: как выбирать быстрый путь на основе группы клиентов?

Шаг 1: измеряем время по каждому из путей, с клиента.

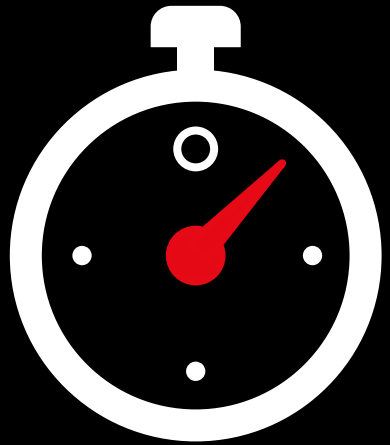



```
type: HTTP GET
name: steering test
targets:
  cloud.test.me/probe
  ix-cloud.test.me/probe
  isp-cloud.test.me/probe
```

→

```
cloud:       time / OK / FAIL
ix-cloud:    time / OK / FAIL
isp-cloud:   time / OK / FAIL
```

Шаг 2: агрегируем Пробы по Recursive Resolver.



Probe

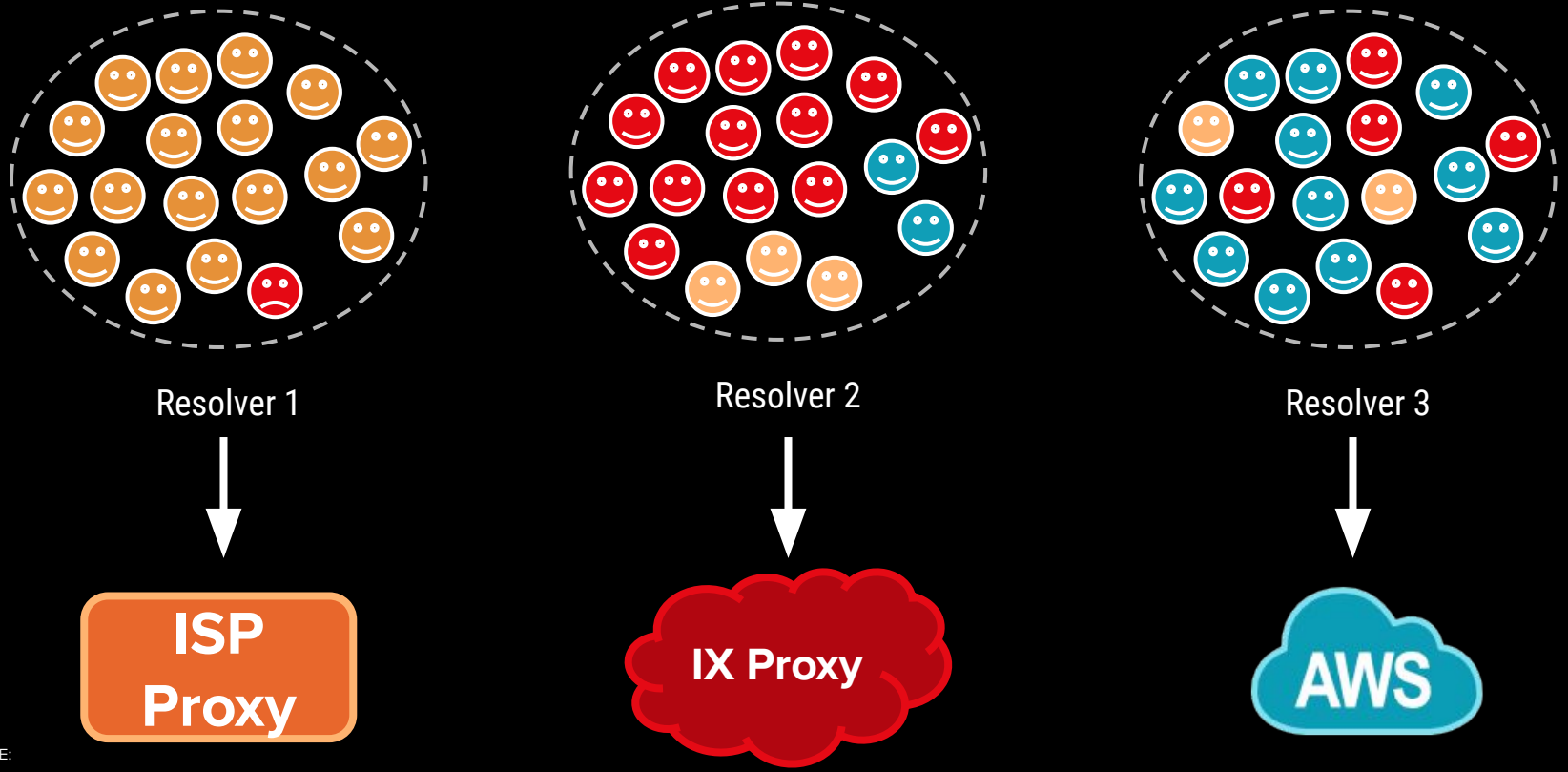
Targets:

- ISP Proxy
- IX Proxy
- Cloud

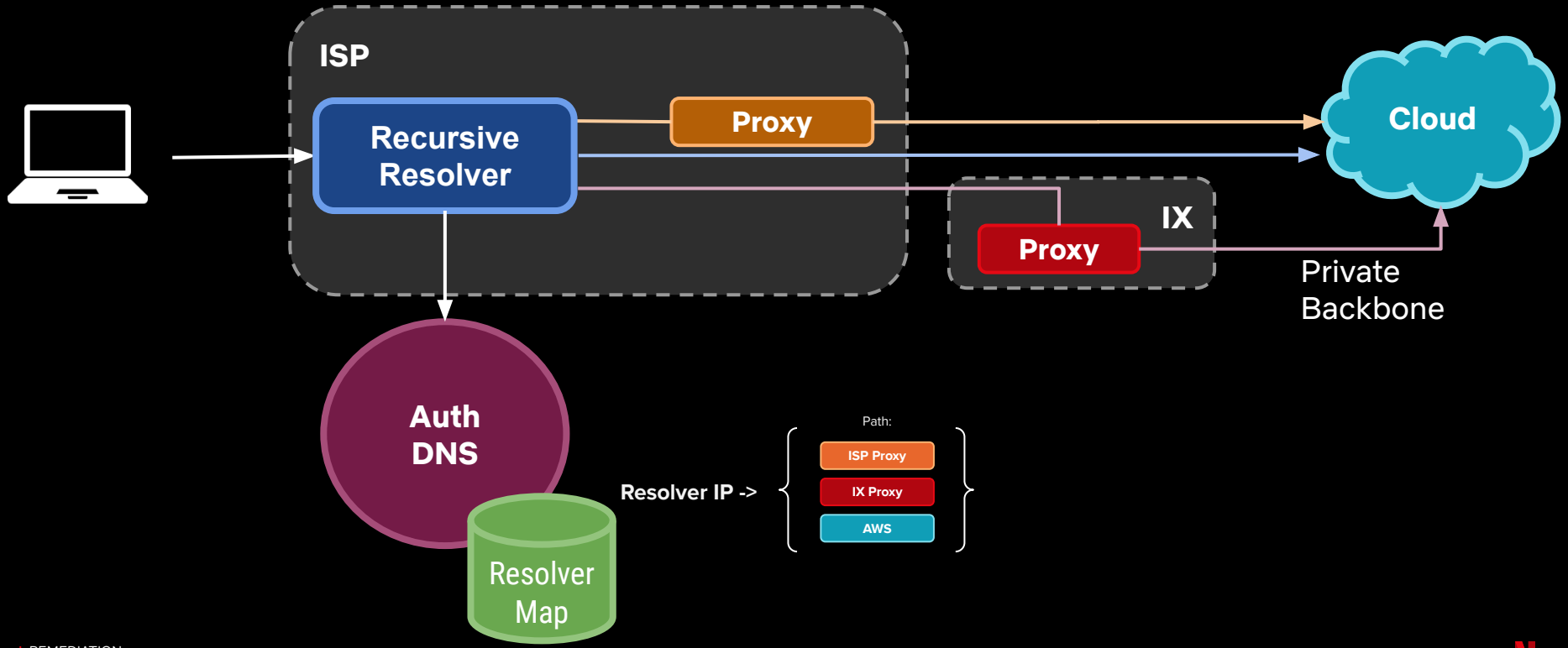
GROUP BY resolver:

resolver 1: IX Proxy
resolver 2: ISP Proxy (stack #)
resolver 3: AWS
resolver 4: IX Proxy
resolver 5: AWS
resolver 6: IX Proxy
...

Шаг 2: агрегируем Пробы по Recursive Resolver.

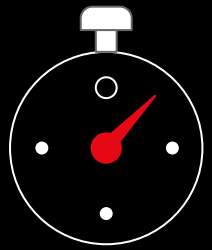


Шаг 3: загружаем модель на Auth DNS и направляем по Recursive Resolver IP.



Автоматический Steering.

Probes



Измеряем задержку по возможным путям в сети

=>

Data Pipeline

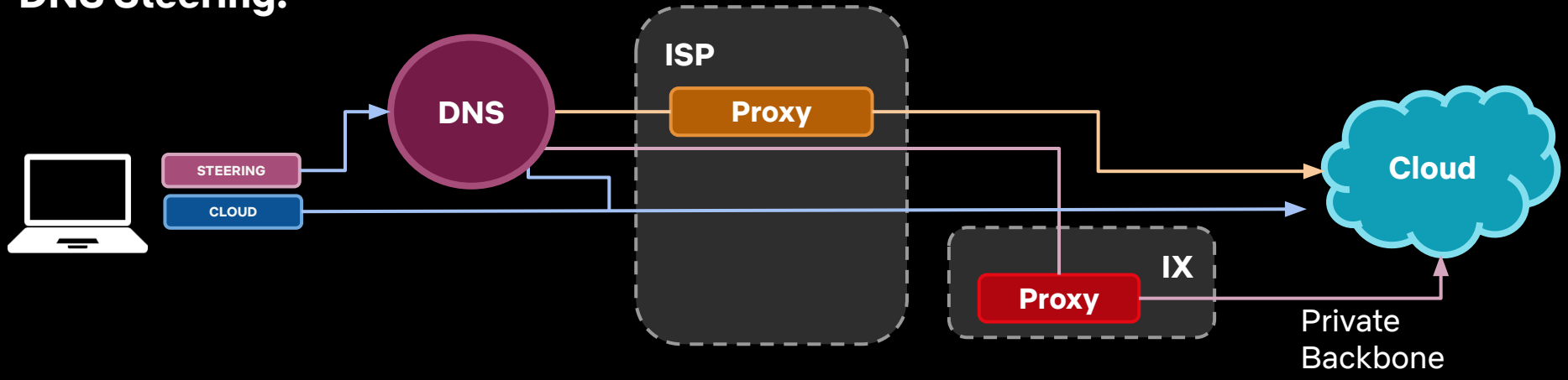
Выбираем самый быстрый путь

=>

Client Steering

Используем выбранный путь для production трафика

Используем Пробы чтобы оценить DNS Steering.

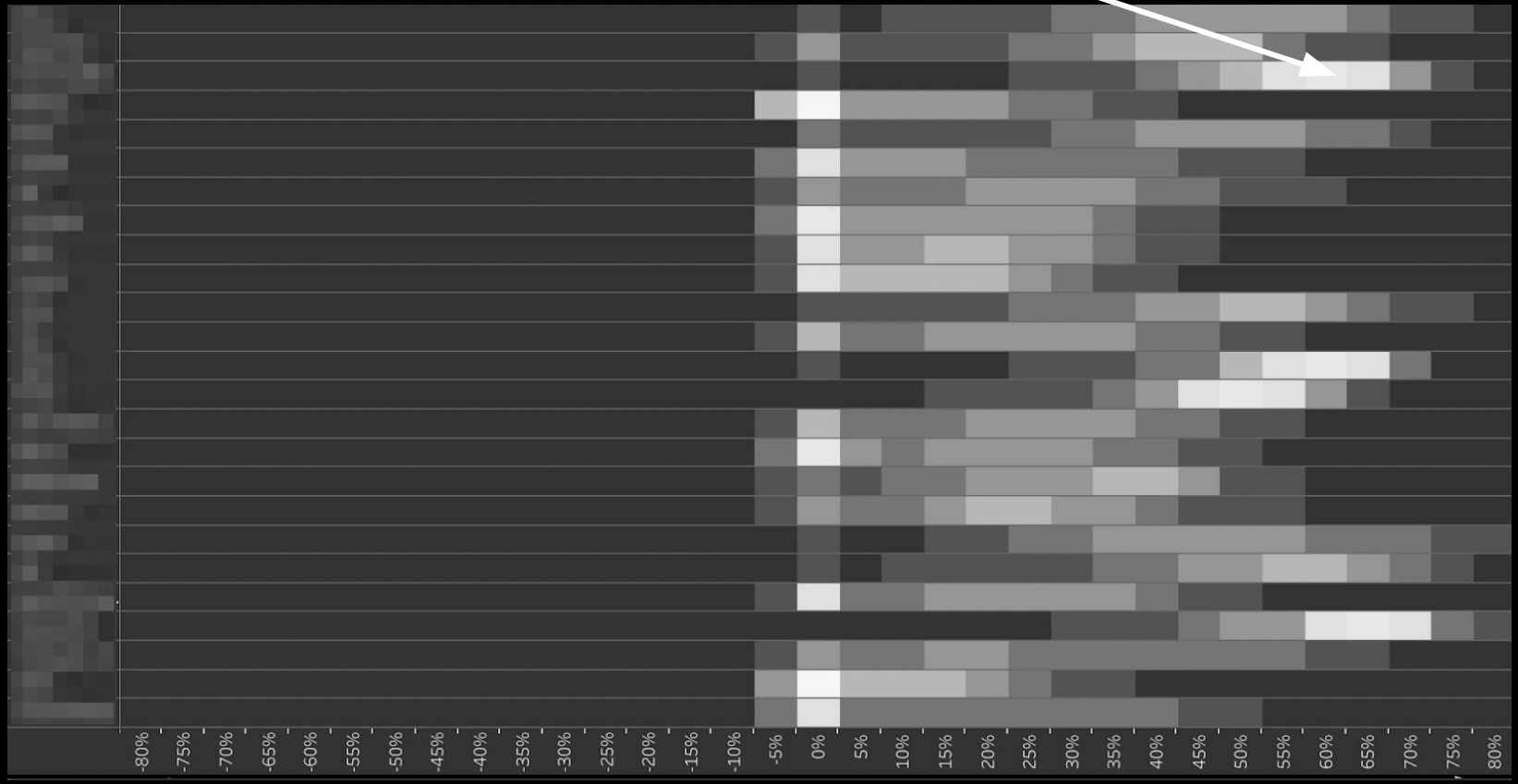



```
type: HTTP GET
name: steering test
targets:
  cloud.test.me/probe
  dns-steering.test.me/probe
```

→

```
cloud:          time / OK / FAIL
dns-steering:   time / OK / FAIL
```

Где **светлее** = **больше** клиентов



Steering **медленнее**

Steering **быстрее**

INTERNET
ACCELERATION

Обобщаем.

- DNS подход сработал.
- Настроили модели агрегации по DNS Resolvers.
- Протестировали и измерили end-to-end решение.
- **Не трогали (не ломали) production.**

Продуктизируй.

Работающий прототип - это далеко не все.

Scope of work

HTTP APIs

CDN

150M пользователей

1K+ видов устройств

10K+ CDN Servers

1K+ локаций

1M+ запросов в секунду

Роутинг через Интернет

100+ микросервисов

100+ deployments в день

Team picture

Как спать спокойно?

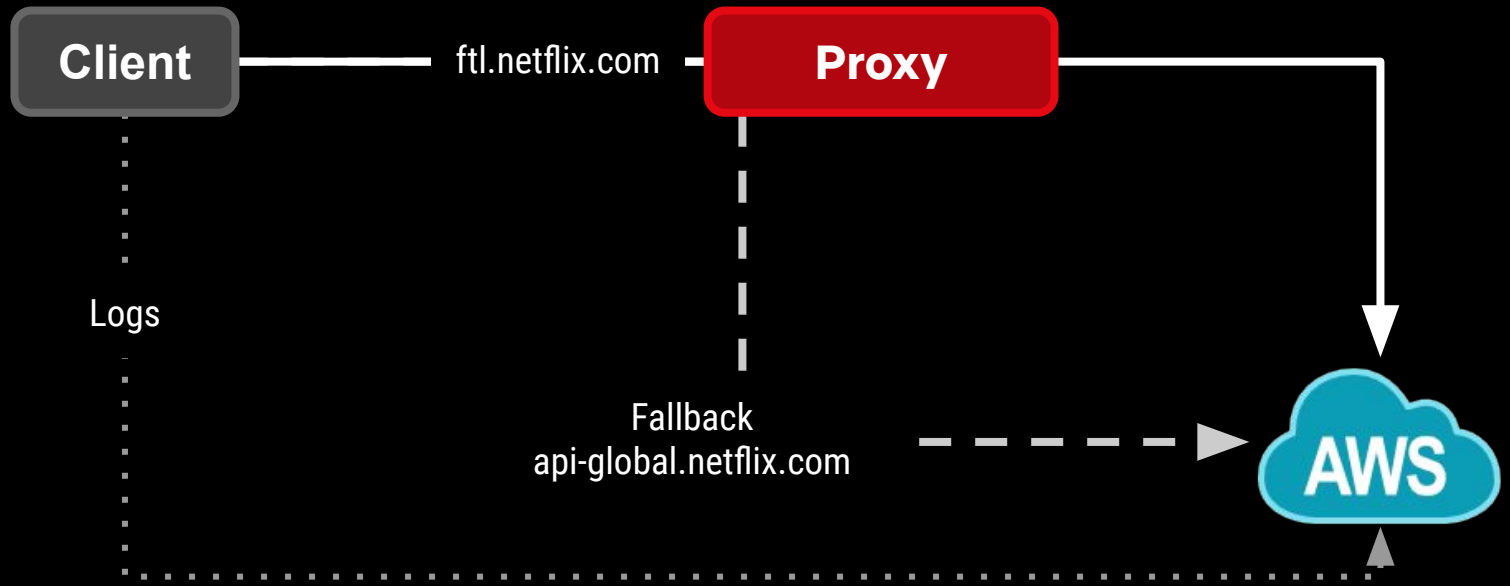
Как продолжать разработку, а не тратить все время на поддержку?

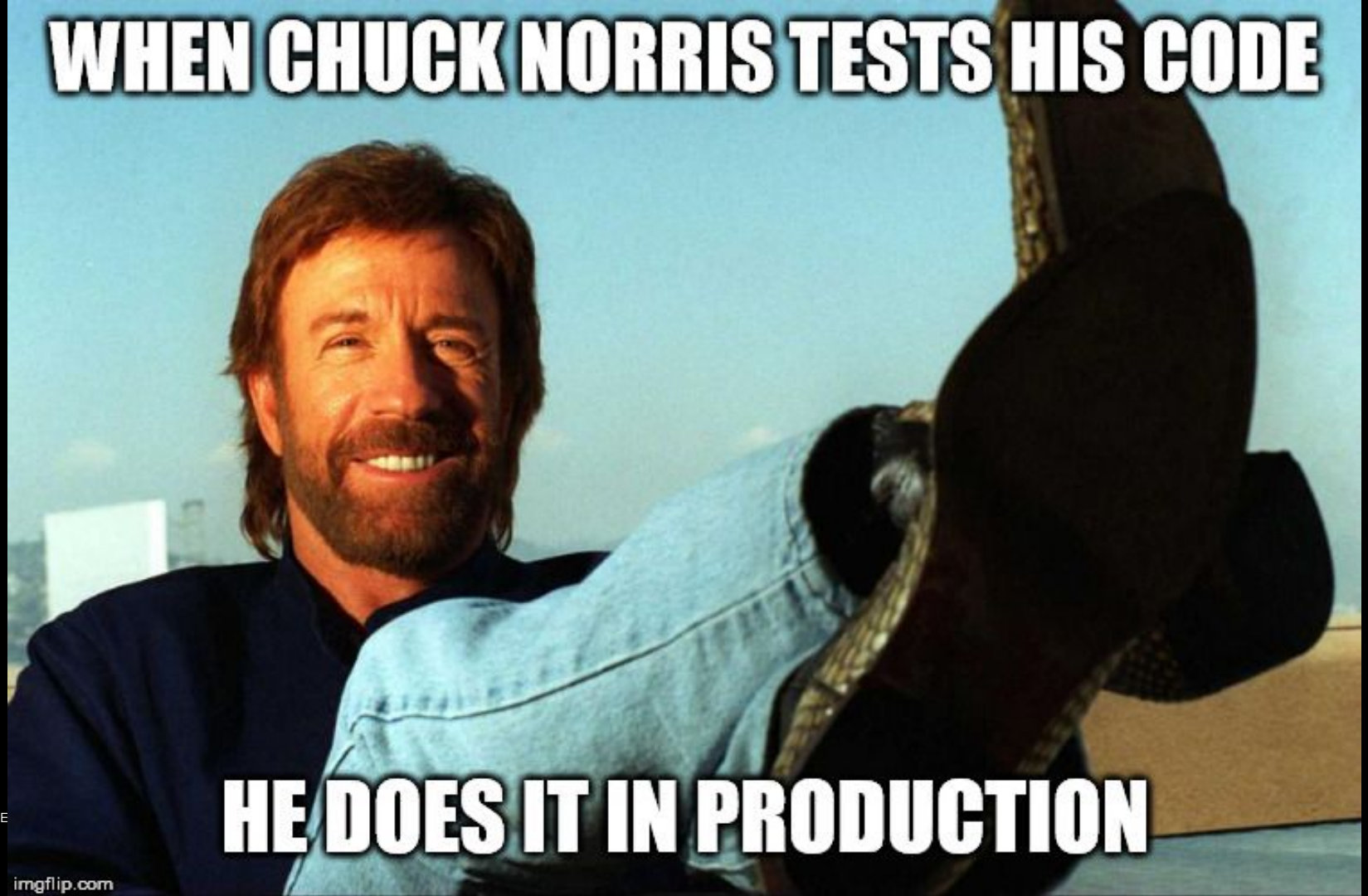
Уменьшай масштаб поломок (blast radius).

Думай об ошибках и поломках в момент проектирования.

Постепенная деградация (graceful degradation).

Думай о поломках в момент проектирования.



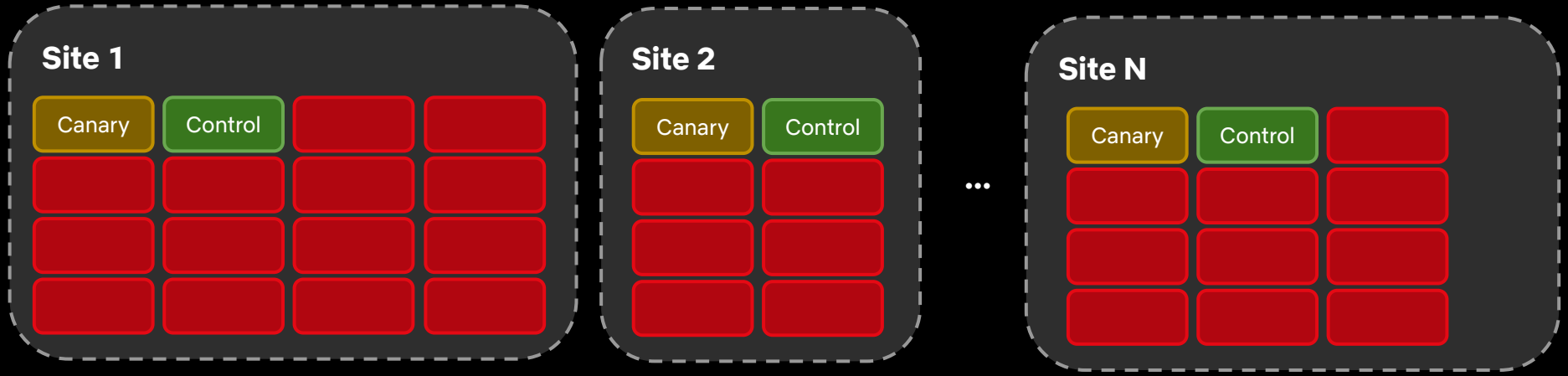

WHEN CHUCK NORRIS TESTS HIS CODE
HE DOES IT IN PRODUCTION
imgflip.com

Маленькие изменения. Частые deployments. Постепенный release.

Workflow:

1. Тест на Пробах.
2. AB тест или Canaries.
3. Постепенный выпуск (progressive rollout).
4. Done.

Canary deployments.



Canary analysis.

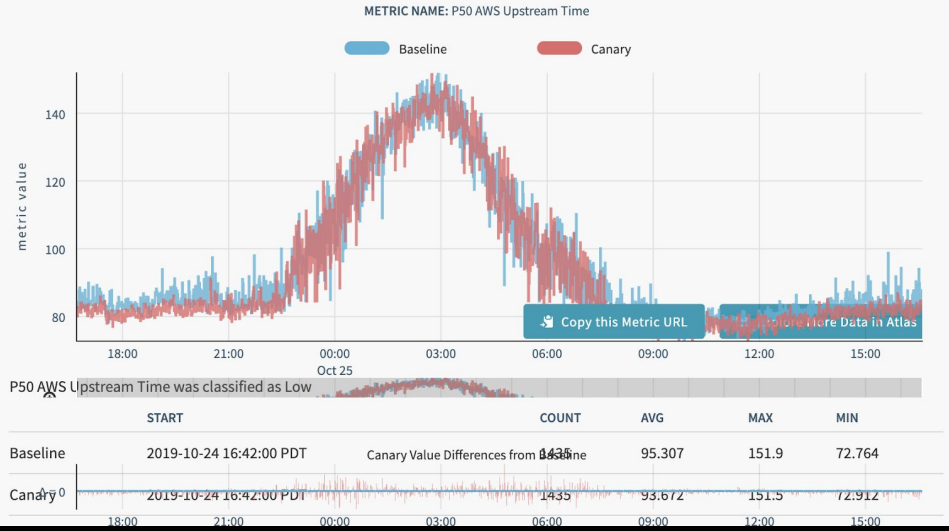
0
FAIL

svm-canary-proxy

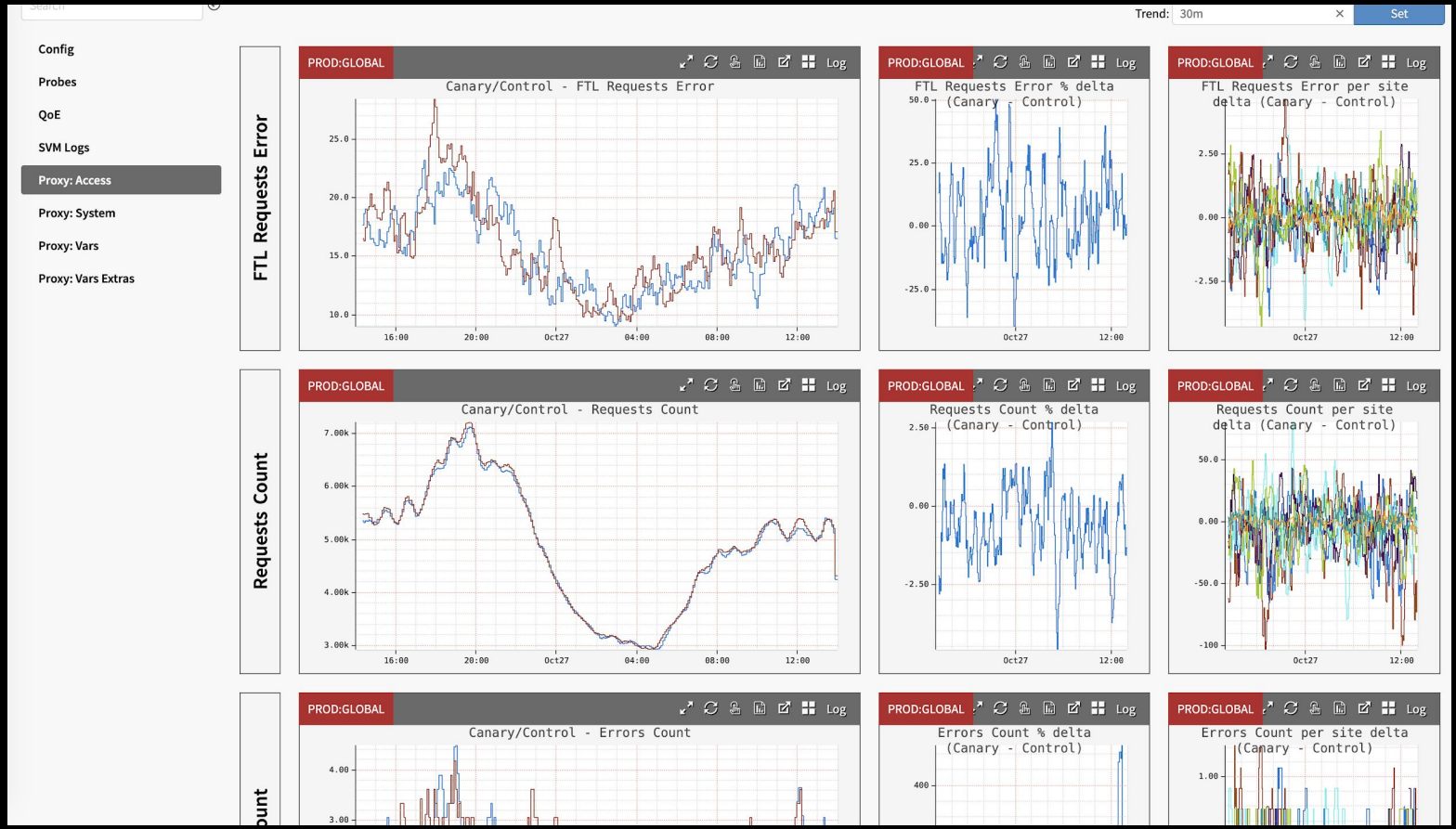
BASELINE	CANARY	TIME	THRESHOLD	SOURCE
SCOPE	SCOPE	START	MARGINAL	Report
cacheName,c198.atl001.ix|c336.dfw001.ix|c508.ord001.ix|c0…	cacheName,c196.atl001.ix|c339.dfw001.ix|c513.ord001.ix|c0…	2019-10-24 16:42:42 PDT	75	Metrics
LOCATION	LOCATION	END	PASS	
		2019-10-25 16:37:42 PDT	95	
		STEP		
		1 min		

ALL | ACCESS | FTLPROXY | LOGS | SYSTEM | VARS

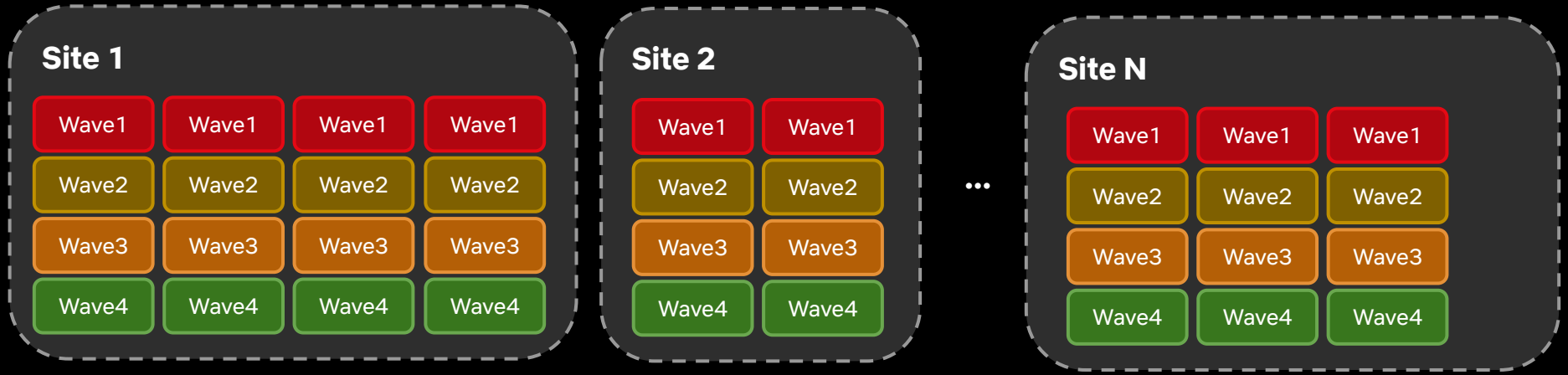
METRIC NAME	DEVIATION	RESULT
Errors Count	+2.1%	Pass
FTL Requests Error	-1.0%	Pass
P25 AWS Upstream Time	-2.1%	Pass
P5 AWS Upstream Time	-9.9%	Pass
P50 AWS Upstream Time	-1.7%	Low
P75 AWS Upstream Time	-1.0%	Pass
P95 AWS Upstream Time	-0.0%	Pass
Requests Count	-0.3%	Pass

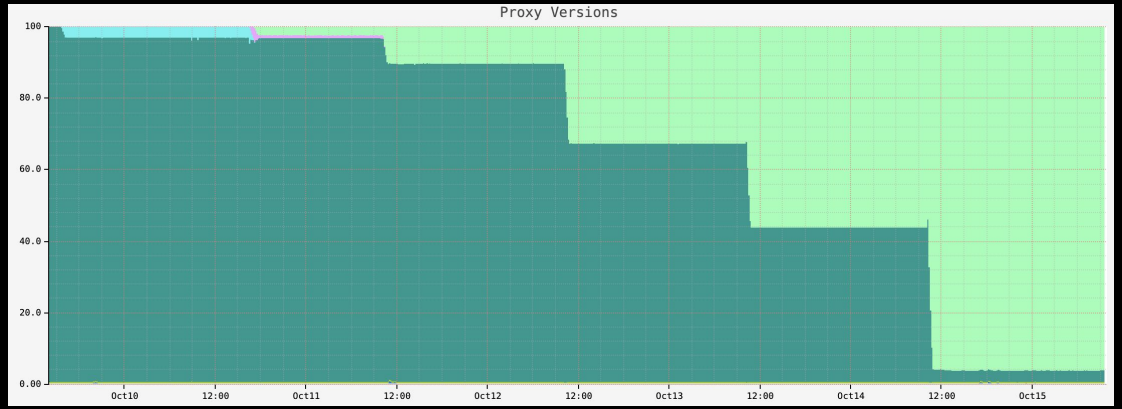


Canary dashboard.

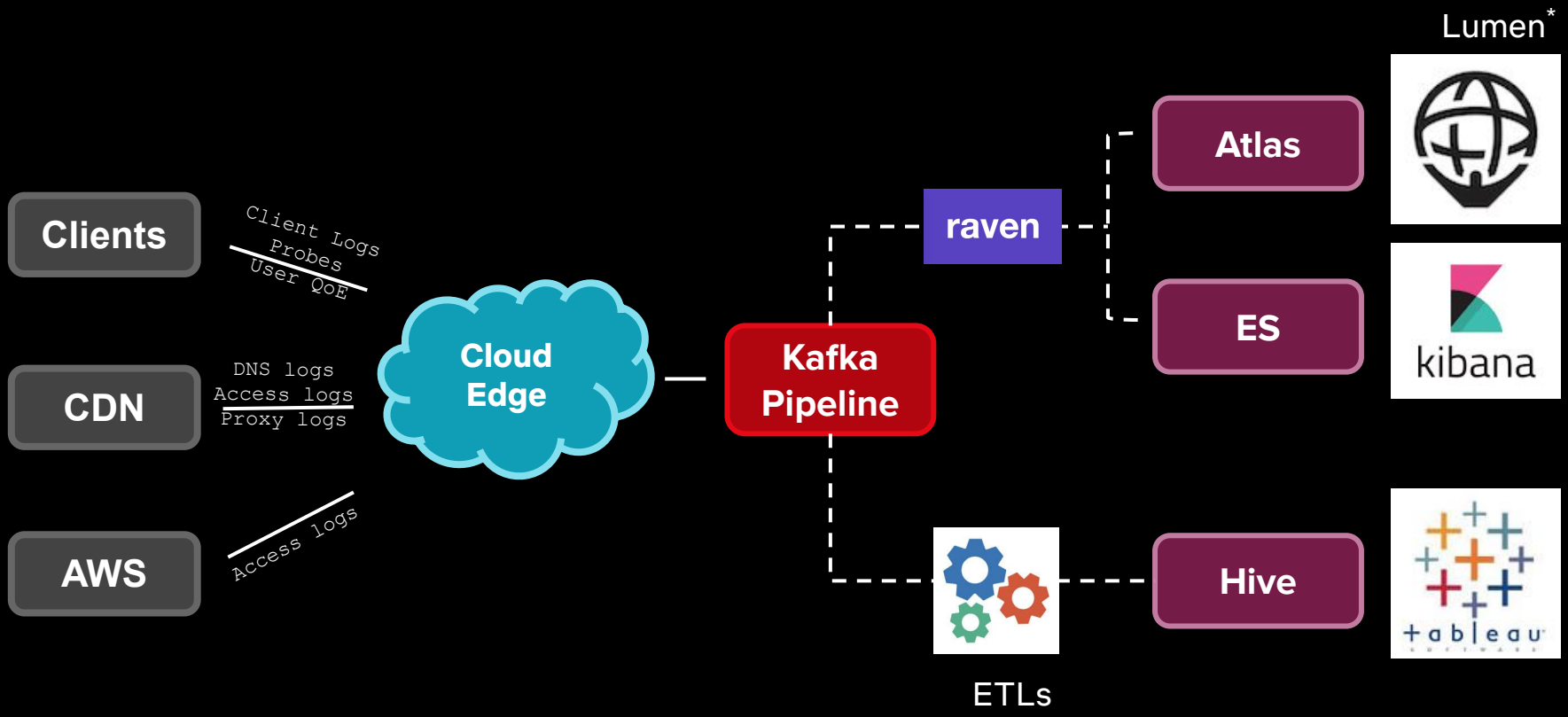


Canary your deployments.





Инструментация на всех уровнях.



* Lumen: Custom, Self-Service Dashboarding For Netflix: https://medium.com/netflix-techblog/lumen-custom-self-service-dashboarding-for-netflix-8c56b541548c

Мониторь.

Знай когда (не) ты во всем виноват.

Когда ты меняешь критический путь - виноват во всем.

150M пользователей

1K+ видов устройств

10K+ CDN Servers

1K+ локаций

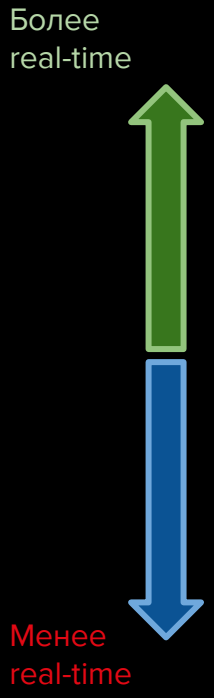
1M+ запросов в секунду

Роутинг через Интернет

100+ микросервисов

100+ deployments в день

Подход к мониторингу.



1. Обнаружение
2. Triage
3. Локализация
4. Диагностика
5. Моделирование

Dashboard статистика:

40 в Lumen

15 в Kibana

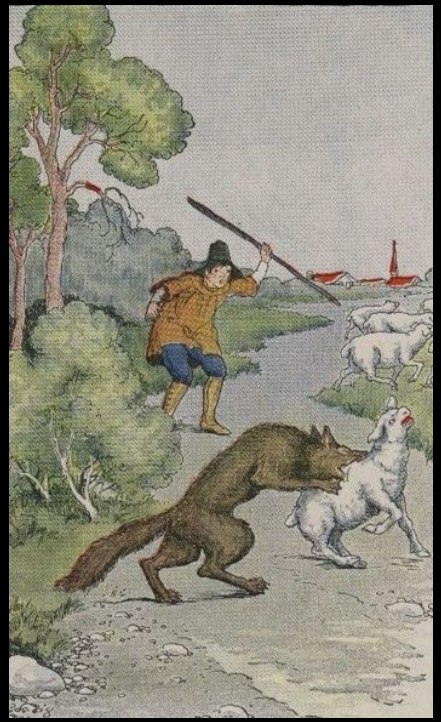
25 в Tableau

Подход к мониторингу.

Время triage: 1-2 минуты

Время на диагностику:
минуты до нескольких часов

Alerting Fatigue

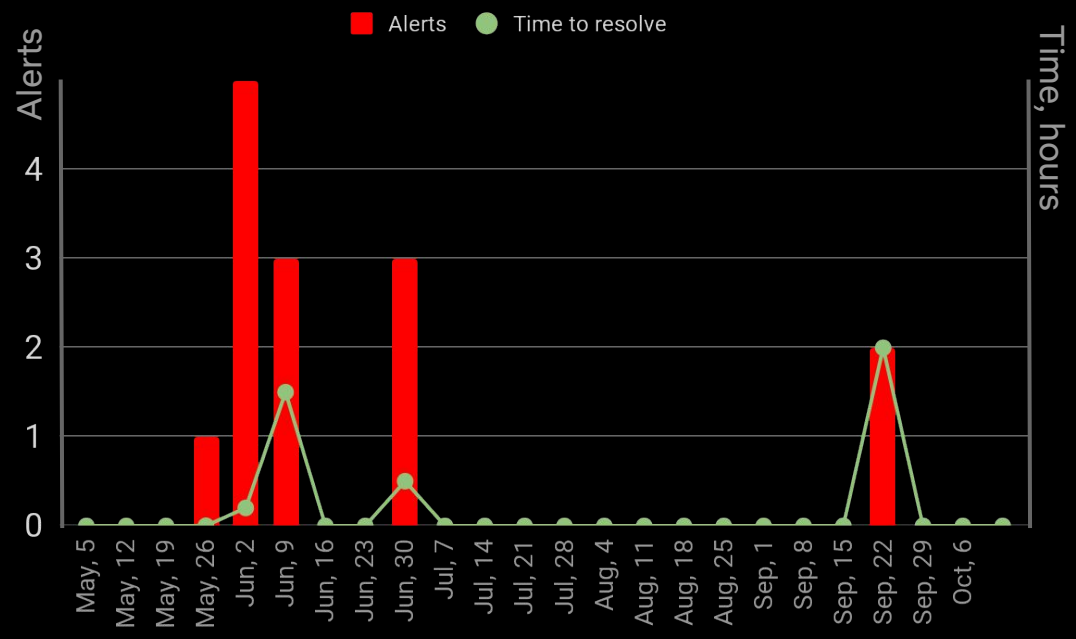


Мальчик который кричал “Волки”

Подход к алертингу.

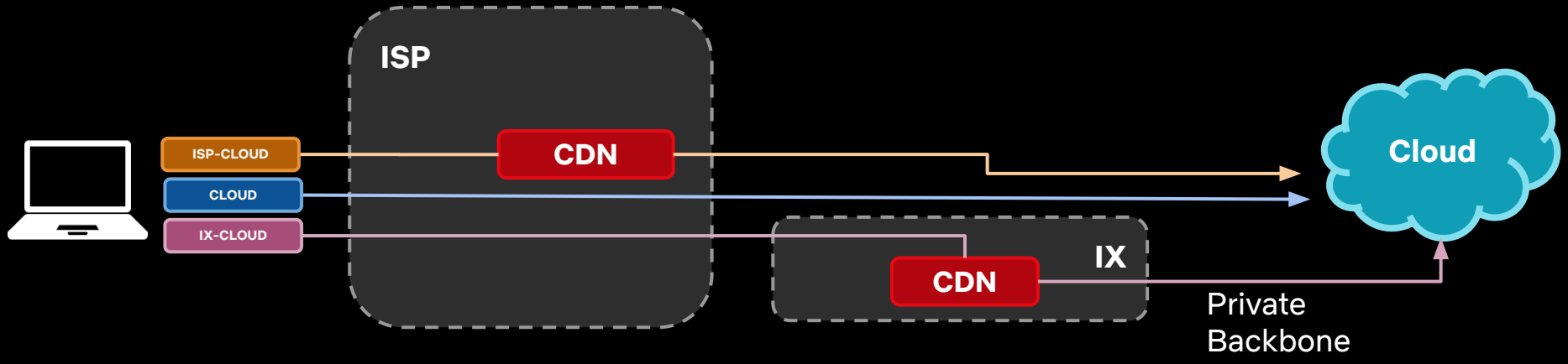
2 **Critical** Alerts:

- Процент Client Fallback
- Процент Probe errors



Количество критических alerts (в неделю) / среднее время починки

Пробы для диагностики ошибок.

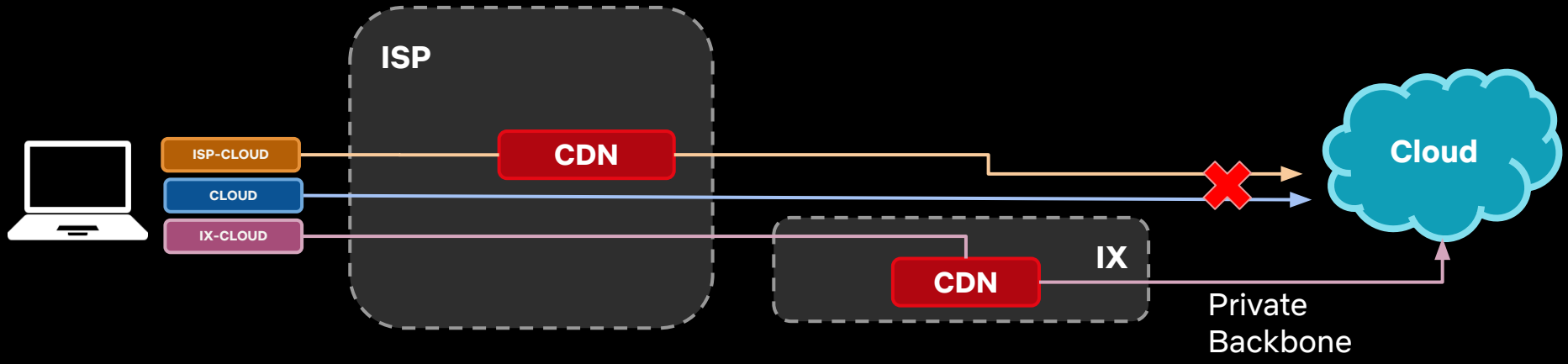



```
type: HTTP GET
name: steering test
targets:
  cloud.test.me/probe
  ix-cloud.test.me/probe
  isp-cloud.test.me/probe
```

→

```
cloud:        time / OK / FAIL
ix-cloud:     time / OK / FAIL
isp-cloud:    time / OK / FAIL
```

Пробы для диагностики ошибок.




```
cloud:          FAIL
ix-cloud:       OK
isp-cloud:      FAIL
```

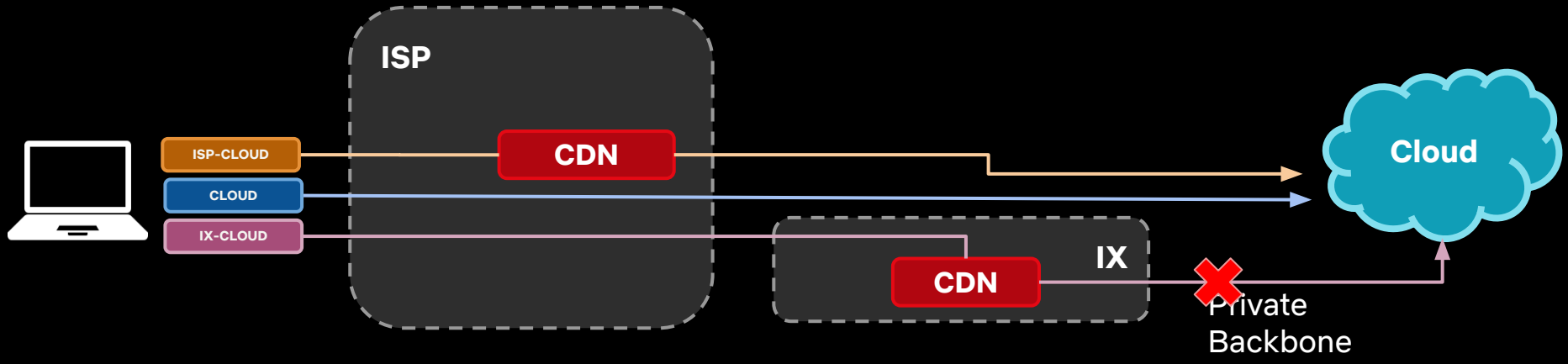
Что сломалось?

- Соединение провайдера в облако

Can we fix it?

- ДА - перенаправляем трафик через IX CDN и Backbone

Пробы для диагностики ошибок.




```
cloud:          FAIL
ix-cloud:       OK
isp-cloud:      FAIL
```

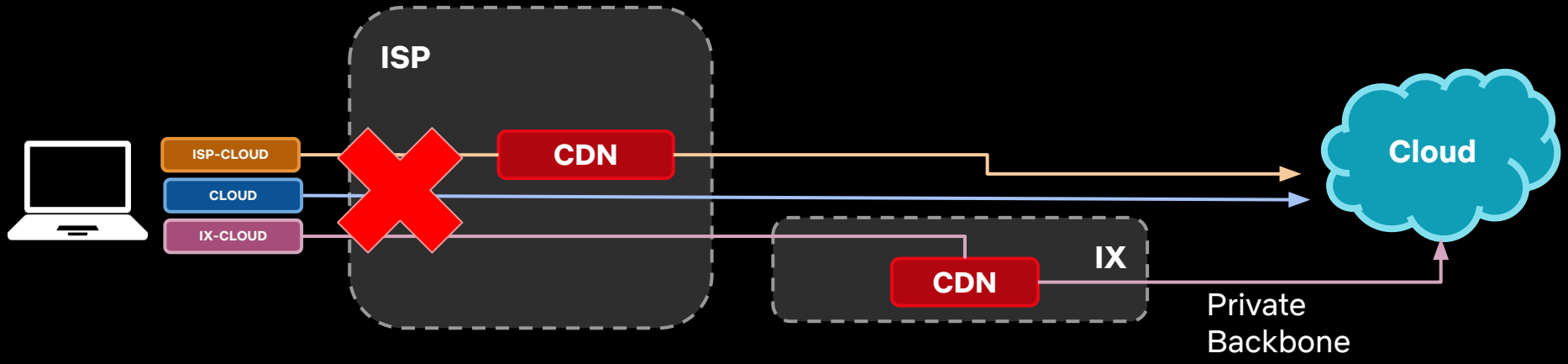
Что сломалось?

- Backbone путь в облако

Можем починить?

- ДА - перенаправить трафик в облако напрямую

Remediation for Full Isolation




```
cloud:          FAIL
ix-cloud:       FAIL
isp-cloud:      FAIL
```

Что сломалось?

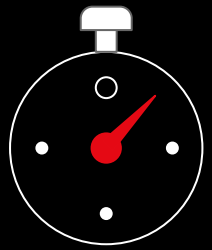
- Инфраструктура провайдера или домашняя сеть

Can we fix it?

- НЕТ (нету работающего пути)

Automatic traffic steering

Probes



Измеряем задержку по возможным путям в сети

=>

Data Pipeline

Выбираем самый надежный путь

=>

Client Steering

Используем выбранный путь для production трафика

Принципы поддержки системы.

- Уменьшаем масштаб поломок.
- Собираем метрики.
- Работаем над dashboards и triage toolset.
- Автоматически чиним поломки если можем.
- Алертинг если большие поломки.

Workflow

Workflow


First prototype
First prod traffic
1mo
2mo
3mo
4mo
2+yr
Probing system
DNS Steering
Gradual deployment


Measure
Prototype
Productize
Monitor

Учись.

Подвергай сомнению свои решения.
Слушай, что говорят данные.

Не верь интуиции.

Примеры где интуиция нас подвела:

- Ускорение задержки с CDN Proxy
- Стабильность TCP Anycast
- Влияние DNS конфигурации
- Влияние TLS настроек
- Распределение трафика
- Нагрузка на инфраструктуру
- ...

Учись на данных из production.

Примеры проблем из production:

- Баги с интеграцией на клиенте
- Лимиты на длину URL
- Legacy TLS конфигурации
- Баги в DNS Recursive Resolvers
- ...

Только ТЫ можешь точно знать что работает для ТВОИХ пользователей.

Наши результаты отличаются от:

- Akamai: DNS resolvers stats
- Uber: HTTP2 performance stats
- Cloudflare: TLS performance stats
- ...

Говори НЕТ модным трендам и фичам, пока не доказал их полезность.

Что мы НЕ сделали:

- Поддержка ECS EDNS0 subnet
- Изменение network policies для anycast
- Real-time PID loop для балансировки
- Кэширование данных
- ...

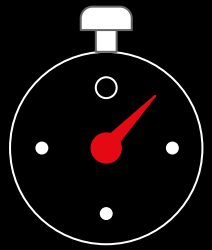
Обращай внимание на новые применения.

Мы начали со снижения задержки, но теперь также делаем:

- Балансировку AWS трафика по регионам
- Моделирование CDN reliability
- Конфигурирование DNS
- Конфигурирование TLS/TCP
- ...

Заключение: Netflix подход к ускорению запросов.

Probes



Измеряем задержку по возможным путям в сети

=>

Data Pipeline

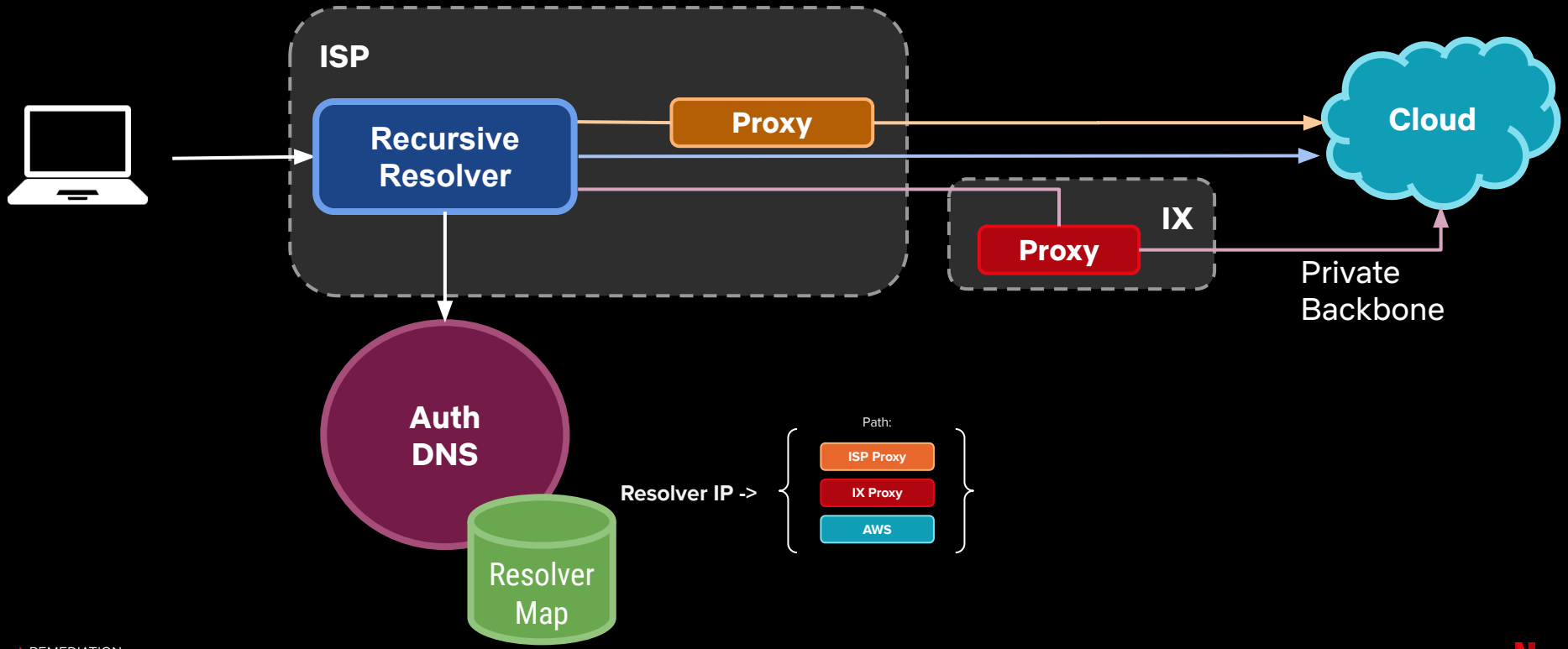
Выбираем самый быстрый и надежный путь

=>

Client Steering

Используем выбранный путь для production трафика

Заключение: Netflix подход к ускорению запросов.



Заключение: что нужно делать Тебе.

- Понимай влияние сети на сервис.
- Меряй то что можешь изменить.
- Начинай с простого.
- Думай о поломках при проектировании.
- Избегай alert fatigue.
- Делай что работает для тебя, я не для других.
- Учись в процессе.

Спасибо.

 sfedorov@netflix.com

 @sfedov