



# Тестирование производительности SportsBook 2.0 с использованием k6

Daniil Matafonov

October 2020

[www.altenar.com](http://www.altenar.com)







## Об авторе



**Daniil Matafonov**

QA Automation Engineer

Занимаюсь автоматизацией тестирования SportsBook 2.0 в компании Altenar, которая является поставщиком программного обеспечения, для лицензированных игорных операторов

**Email:** [daniil.matafonov@altenar.com](mailto:daniil.matafonov@altenar.com)



## О компании

**Altener** — международная B2B компания, с офисами в России, на Мальте и в Греции. Основным направлением является разработка платформы для лицензированных букмекеров, оперирующих в Европе и Южной Америке.

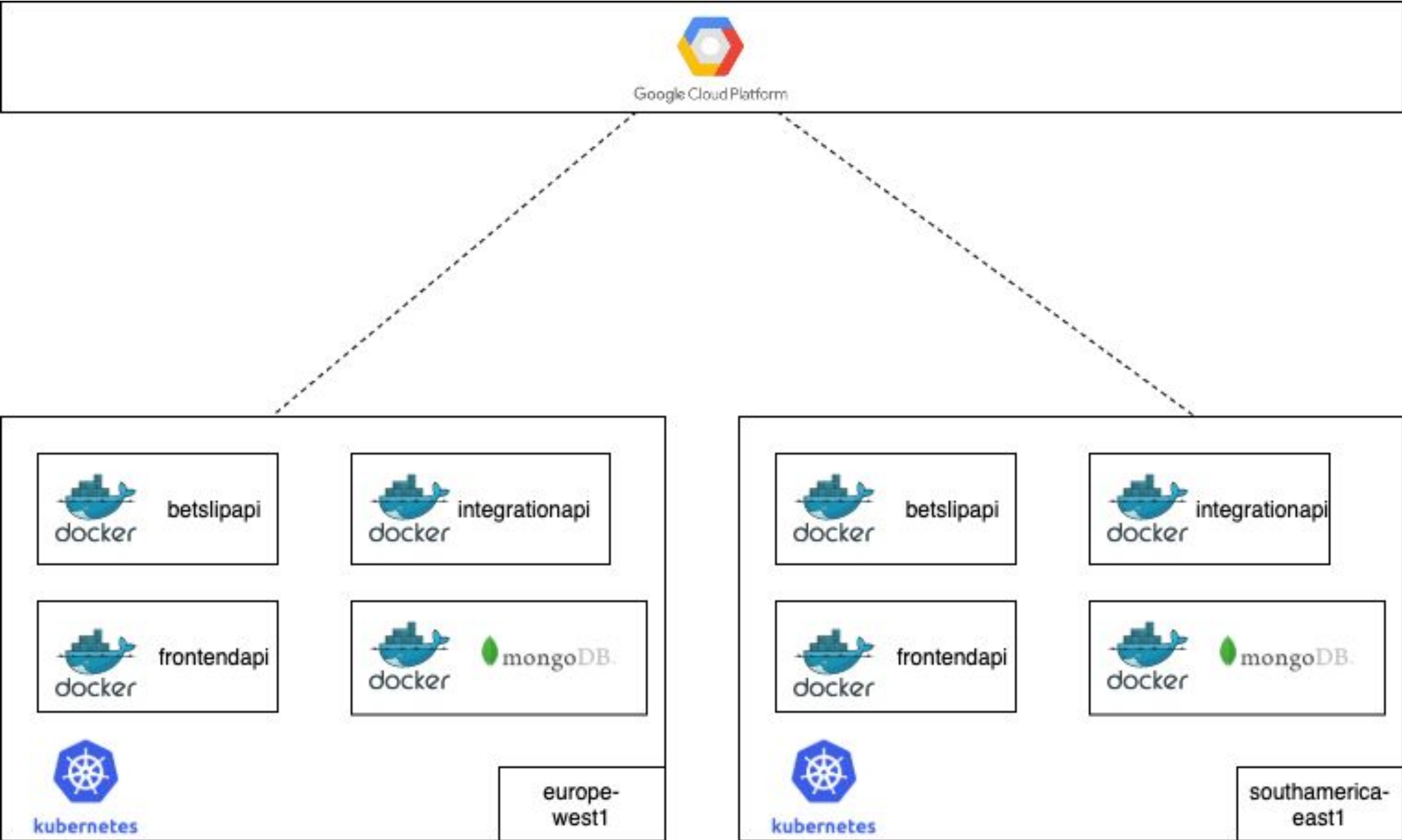


## О продукте

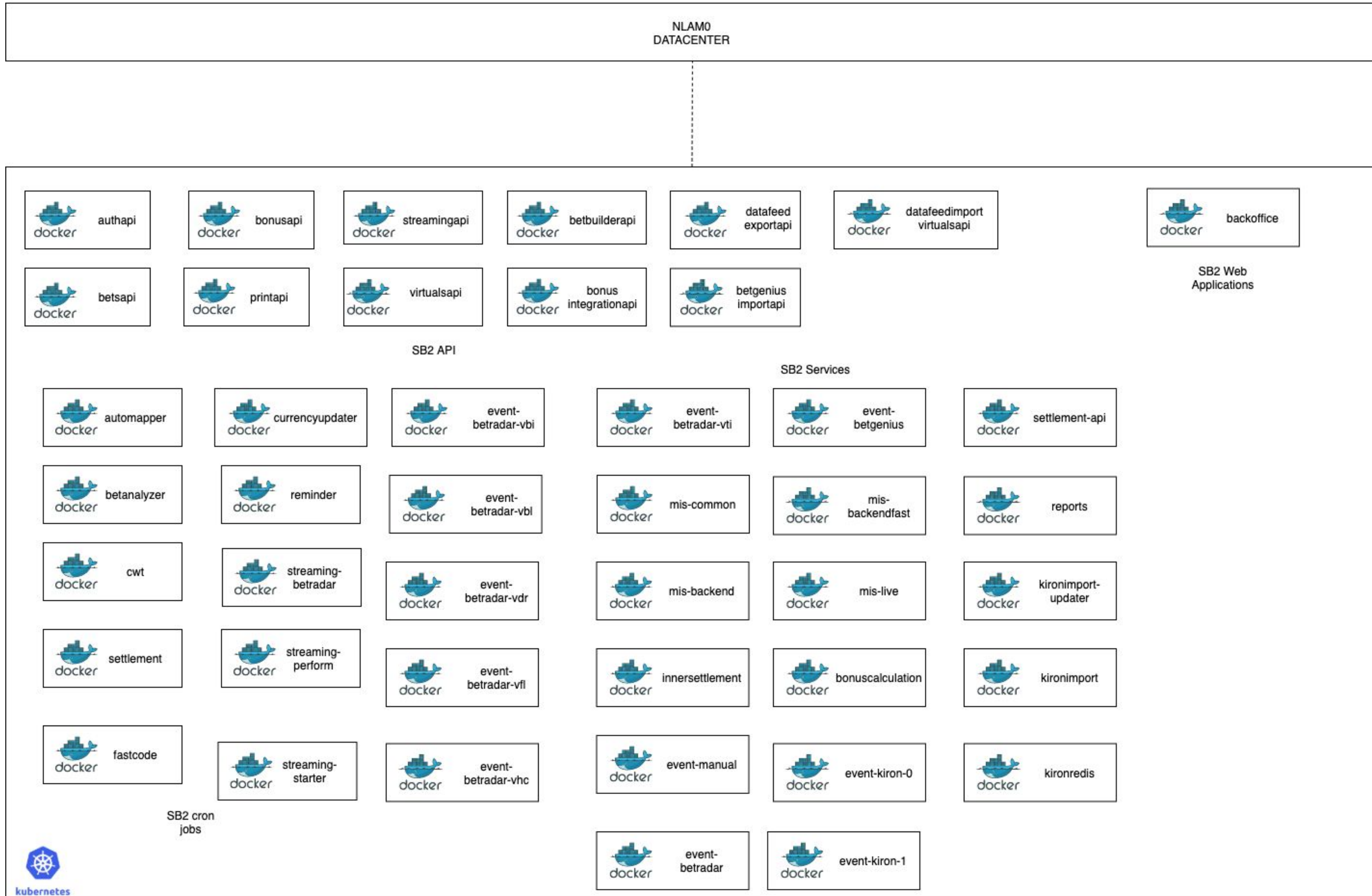
**SportsBook** - программное обеспечение, предназначенное для выполнения ставок на различные спортивные соревнования, включая гольф, футбол, баскетбол, бейсбол, хоккей, скачки, виртуальные виды спорта и другие.



# Архитектура SportsBook 2.0 Google Cloud Cluster









# Нагрузочное тестирование с использованием JMeter

Цель:

Выполнить стресс-тестирование проекта Hybrid и получить показатели производительности таких запросов как PlaceBet, BetHistory, CalculateCashout , Cashout.

Окружение: Stage

Конфигурация: 3 VMs in VMWare cluster with 4 vCPUs, 4 gb.



# Недостатки JMeter

- **Ограниченный мониторинг тестов** (трудоемкость анализа логов запуска тестов);
- **Трудоемкость создания тестового сценария** (необходим опыт для понимания элементов JMeter, регулярных выражений, обработки сеансов и т.д.);
- **Ограниченные возможности создания тестовых сценариев с помощью кода**





# Анализ требований

## Проблемы

- Ограниченный мониторинг тестов
- Трудоемкость создания тестового сценария
- Ограниченные возможности создания тестовых сценариев
- Трудоемкость использования тестов на Jmeter в сборках

## Решения

- Инструмент с возможностью поддержкой различных режимов отладки (просмотр тела ответов, возможность отключения отладочных логов)
- Инструмент, который облегчит вовлечение новых участников в процесс разработки тестов (простой и понятный подход для разработки сценариев, поддержка GitOps практик [обзор кода, процесс утверждения с помощью запросов pull и так далее])
- Инструмент, позволяющий разрабатывать тесты на простом и понятном языке программирования
- Поддержка CI / CD



# Анализ инструментов

Функциональные особенности	<a href="#">Jmeter</a>	<a href="#">Gatling</a>	<a href="#">k6</a>	<a href="#">Locust</a>
Распределенное выполнение тестов	Подход Master/slave. Основной узел для запуска подчиненных узлов, имитирующих трафик.	Вручную	Вручную и с использованием executors в версии k6 0.27. See <a href="#">#1007</a> below	Подход Master/slave. Основной узел для отображения статистики в реальном времени и запуска подчиненных узлов, имитирующих трафик
Документация / Поддержка	JMeter - это проект Apache. Один из старейших инструментов, благодаря которому он имеет большое сообщество. Подробная документация.	Инструмент, поддерживаемый одноименной компанией, имеет версию с открытым исходным кодом. Подробная документация.	Проект с открытым исходным кодом, поддерживаемый <a href="#">LoadImpact</a> . Подробная документация	Locust достаточно активно развивается, группой независимых разработчиков. Подробная документация. <a href="https://docs.locust.io/en/stable/">https://docs.locust.io/en/stable/</a>
Протоколы [gRPC, WebSocket, HTTP]	Из коробки: <ul style="list-style-type: none"> <li>• HTTP [1.1]</li> </ul> Через внешний плагин: <ul style="list-style-type: none"> <li>• WebSocket</li> </ul>	Из коробки: <ul style="list-style-type: none"> <li>• HTTP [1.1/2]</li> <li>• WebSocket</li> </ul> Через внешний плагин:	Из коробки: <ul style="list-style-type: none"> <li>• HTTP [1.1/2]</li> <li>• WebSocket</li> </ul> Note: See <a href="#">#441</a> below	Из коробки: <ul style="list-style-type: none"> <li>• HTTP</li> </ul> Note: See <a href="#">#17</a> and <a href="#">#83</a> below





# Достоинства и недостатки

- Гибкость тестового сценария

JMeter	Gatling	K6	Locust
Средняя	Высокая	Высокая	Средняя

- Производительность

JMeter	Gatling	K6	Locust
Средняя	Высокая	Высокая	Высокая

- Язык скриптов

JMeter	Gatling	K6	Locust
Java + GUI	Scala	JavaScript	Python

- Порог вхождения

JMeter	Gatling	K6	Locust
Низкий	Высокий	Средний	Средний

- Вывод статистики

JMeter	Gatling	K6	Locust
UI	Console, UI	console, influxDB, cloud, json, kafka, DataDog, StatsD	Console, UI

- Распределенное тестирование

JMeter	Gatling	K6	Locust
Да	В платной версии	Да	Да



# Выводы

К6 как оптимальный инструмент для нагрузочного тестирования

- Преимущества:
  - поддержка протоколов http1.1 / 1.2; gRPC;
  - кроссплатформенность (Linux, Mac, Windows, Docker образ)
  - разработка тестов на JS
  - поддержка GitOps практик
  - возможность использования в CI сборках
  - возможность запуска из cloud
  - развитое комьюнити





[k6](#) - современный инструмент для нагрузочного тестирования, разработанный компанией [Load Impact](#).

## Features:

- CLI-инструмент с удобным для разработчика API
- [HTTP/1.1](#), [HTTP/2](#), [WebSocket](#) поддержка протоколов
- Разработка тестов на JavaScript ES2015/ES6 - с поддержкой локальных и удаленных модулей
- **Automation-friendly**: [checks](#) (like asserts) and [thresholds](#) for easy and flexible CI configuration!
- **TLS features**: клиентские сертификаты, настраиваемые SSL/TLS
- Вывод результатов тестов: [InfluxDB](#) (+Grafana), [JSON](#), [k6 Cloud](#), Kafka, StatsD, DataDog
- Облачное выполнение и распределенные тесты (на инфраструктуре, управляемой Load Impact)



# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
K6_VUS	<code>--vus , -v</code>	VUS	1

ВРЕМЕННО.





# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
K6_HTTP_DEBUG	<code>--http-debug</code> , <code>--http-debug=full</code>	<code>httpDebug</code>	<code>false</code>

`--http-debug=full.`

необходимо использовать



# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
<code>K6_DURATION</code>	<code>--duration, -d</code>	<code>duration</code>	<code>null</code>

Задаёт общую продолжительность выполнения теста.

В течение этого времени каждый VU будет выполнять сценарий в цикле. Доступен в K6 run и K6 cloud.





# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
K6_ITERATIONS	<code>--iterations, -i</code>	<code>iterations</code>	1

Параметр, задающий фиксированное число итераций для выполнения сценария, в течение которого сценарий будет выполняться в цикле.



# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
<code>K6_MIN_ITERATION_DURATION</code>	<code>--min-iteration-duration</code>	<code>minIterationDuration</code>	<code>0</code> (disabled)

ии)функции по умолчанию. Любые тавшегося времени, пока не будет





# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
K6_COMPATIBILITY_MODE	--compatibility-mode	N/A	"extended"

Поддержка запуска сценариев с разными режимами запуска

Более подробно про режимы запуска [JavaScript Compatibility Mode documentation](https://k6.io/docs/using-k6/options).



# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
<code>K6_DISCARD_RESPONSE_BODIES</code>	<code>--discard-response-bodies</code>	<code>discardResponseBodies</code>	<code>false</code>



# k6 Options

ENV	CLI	CODE / CONFIG FILE	DEFAULT
N/A	N/A	<code>scenarios</code>	<code>null</code>

тераций, исполняющими различные функции (кроме default) и использующими различные переменные окружения, теги и другое.





## Scenarios (k6 v.0.27)

- shared-iterations
- per-vu-iterations
- constant-vus
- ramping-us
- constant-arrival-rate
- ramping-arrival-rate
- externally-controlled

The k6 logo, which is a purple triangle with the text 'k6' inside in white.

# Checks

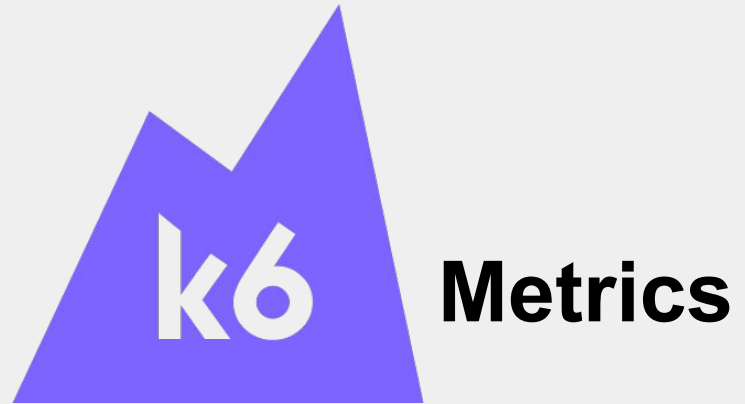
Механизм проверок схож с assertions, но основное отличие в том, что **checks** не останавливают выполнение, а просто сохраняют результат проверки (pass / fail ) и позволяют скрипту продолжить выполнение.

Проверки отлично подходят для кодификации утверждений, относящихся к HTTP-запросам/ответам, например, что код ответа равен 2xx:



Метрика	Тип	Описание
http_reqs	Counter	Общее количество сгенерированных http запросов на тест
http_req_blocked	Trend	Время, затраченное на блокировку (ожидание tcp-соединения) перед отправкой запроса
http_req_connecting	Trend	Время, затраченное на установление TCP-соединения с удаленным хостом.
http_req_tls_handshaking	Trend	Время, затраченное на установление сеанса TLS с удаленным хостом
http_req_sending	Trend	Время, затраченное на отправку данных на удаленный хост
http_req_waiting	Trend	Время, затраченное на ожидание ответа от удаленного хоста.( время до первого байта)
http_req_receiving	Trend	Время, затраченное на получение данных ответа от хоста.
http_req_duration	Trend	Общее время выполнения запроса.( <b>http_req_sending</b> + <b>http_req_waiting</b> + <b>http_req_receiving</b> )





- Custom
  - [Counter](#) (суммарная метрика)
  - [Gauge](#) (хранит только последнее значение)
  - [Rate](#) (отслеживает процент значений, отличных от нуля)
  - [Trend](#) (собирает статистику(min/max/avg/percentiles) для ряда значений)



## Windows

Вы можете вручную загрузить и установить [official .msi installation package](#) или, если вы используете [chocolatey package manager](#), следуйте [ЭТИМ ИНСТРУКЦИЯМ](#) для установки latest версии k6 из репозитория.

## Docker

```
docker pull loadimpact/k6
```

## Linux (deb and rpm packages)

<https://k6.io/docs/getting-started/installation#linux-deb-and-rpm-packages>

## Mac (brew)

<https://k6.io/docs/getting-started/installation#mac-brew>



## Проблемы в k6 (version 0.26.0)

- При запуске теста с количеством VUS > 2000 k6 прерывал выполнение с ошибкой. FATA[0052] stream error: stream ID 5; INTERNAL\_ERROR
- Ограниченная поддержка параллелизации тестов (платная поддержка в k6 cloud)
- Недостаточно гибкая настройка конфигурации сценариев





# Нагрузочное тестирование с использованием k6

## Цель:

Выполнить нагрузочное тестирование проекта SportsBook 2.0 и получить показатели производительности таких методов как signIn, GetEvents, place, betHistory, PrintCoupon, Cashout, OpenBetsCount.

**Окружение:** Stage

**Конфигурация:** VM (24 CPU / 25 GB RAM)

## Тестовый сценарий:

- Выполнение теста разделено на группы
- 1 группа вызывает методы SignIn, GetEvents, place, BetHistory, PrintCoupon, Cashout, OpenBetsCount (10% от общего числа VUS на тест)
- 2 группа SignIn, BetHistory, Cashout, OpenBetsCount (30% от общего числа VUS на тест)
- 3 группа SignIn, BetHistory, OpenBetsCount (60% от общего числа VUS на тест)

## Реализация тестового сценария. Вызов метода SignIn

```
group("Bets-Thread-1", function () {
  group("SignIn", function () {
    let authResp = auth();
    let authCheckResult = check(authResp, {
      "status is 200 OK": (authResp) => authResp.status === 200,
      "login successful, accessToken is not empty":
        trycatch(f: (authResp) =>
          authResp.json("Result").hasOwnProperty(v: 'AccessToken'))
    });
    errorRate.add(!authCheckResult);
    accessToken = authResp.json("Result.AccessToken");
    signInDuration.add(authResp.timings.duration);
  });
});
```

# ALTENAR Реализация тестового сценария. Функция auth()



auth.js

```
import http from "k6/http";
import {options} from "../../config.js";
import { authApiUrl, originStaticUrl } from "../../endpoints.js";

export {options};

export function auth() {
  let token = `${__ENV.TOKEN}` + __ITER + __VU;
  let walletCode = "newPI";
  let url = authApiUrl + "/Auth/SignIn";
  let referer = originStaticUrl + "?token=" + token + "&walletcode=" + walletCode;
  let signInBody = {
    timezoneOffset: '-180',
    langId: '8',
    skinName: 'betsonic',
    configId: `${__ENV.CONFIG}`,
    deviceType: "Desktop",
    numformat: "en",
    culture: 'en-GB',
    countryCode: "",
    token: token,
    walletcode: walletCode,
    grant_type: "platformIntegration",
    special: "",
    period: "null",
    country: 'en-us'
  };
  let json = JSON.stringify(signInBody);
  let resp = http.post(url, json, { headers: {
    "Connection": "keep-alive",
    "Accept": "*/*",
    "Accept-Encoding": "gzip, deflate",
    "Origin": originStaticUrl,
    "Cache-Control": "no-store",
    "Sec-Fetch-Dest": "empty",
    "Sec-Fetch-Mode": "cors",
    "Sec-Fetch-Site": "same-site",
    "Content-Type": "application/json; charset=UTF-8",
    "Referer": referer}});
  return resp;
}
```





## Реализация тестового сценария. Функция placeBet()

placeBet.js

```
import http from "k6/http";
import { options } from "../../config.js";
import { betsApiUrl } from "../../endpoints.js";

export {options};

/* placeBet */
export function placeBet(accessToken, placeObj){
    let placeBetUrl = betsApiUrl + "/Bets/place";
    let reqBody = JSON.stringify(placeObj);
    let resp = http.post(placeBetUrl, reqBody,
        { headers:
            {Authorization: "Bearer " + accessToken, "Content-Type": "application/json"}
        });
    return resp;
}
```





```
export let options = {
  httpDebug: 'full',
  minIterationDuration: '1m',
  scenarios:{
    thread_groups_placement_scenario:{
      executor: 'constant-vus',
      exec: 'placement',
      vus: 100,
      duration: '3m',
      gracefulStop: '0s',
      tags: { my_custom_tag: 'placement' },
    },
  },
};
```



## Сборка тестов



```

> sb2.k6.betting.tests.load@1.0.0 webpack /home/LoadTest/sb2.k6.betting.tests.load
> webpack

Hash: e1f668413aaf5f39bf09
Version: webpack 4.44.1
Time: 2991ms
Built at: 09/22/2020 7:40:37 AM

```

Asset	Size	Chunks	Chunk Names
app.bundle.js	30.8 KiB	0 [emitted]	main
app.bundle.js.map	140 KiB	0 [emitted] [dev]	main

```

Entrypoint main = app.bundle.js app.bundle.js.map
[0] external "k6" 42 bytes {0} [built]
[1] external "k6/http" 42 bytes {0} [built]
[2] external "k6/metrics" 42 bytes {0} [built]
[6] ./manual_events/event1.json 1.27 KiB {0} [built]
[39] (webpack)/buildin/global.js 475 bytes {0} [built]
[71] ./scenarios/bets.js + 13 modules 49.4 KiB {0} [built]
| ./scenarios/bets.js 26.1 KiB [built]
| ./insights.js 1.05 KiB [built]
| ./config.js 1.22 KiB [built]
| ./helpers/helpers.js 13 KiB [built]
| ./api/upcoming.js 640 bytes [built]
| ./api/auth/auth.js 1.2 KiB [built]
| ./api/bets/placeBet.js 473 bytes [built]
| ./api/bets/betHistory.js 1.24 KiB [built]
| ./api/print/printCoupon.js 789 bytes [built]
| ./api/bets/getBetDetails.js 783 bytes [built]
| ./api/events.js 959 bytes [built]
| ./api/bets/cashout.js 809 bytes [built]
| ./api/bets/getOpenBetsCount.js 761 bytes [built]
| ./endpoints.js 417 bytes [built]
+ 66 hidden modules

```

## КОМПИЛЯЦИЯ:

npm run-script webpack





## запуск тестов

```
#!/bin/bash

influxUrl=
influxPort='8086'
dbName=
vus='100'
LOGS_DIR="./logs"
k6 run
  -e K6_COMPATIBILITY_MODE=base
  -e CHAMP_ID=2946
  -e TOKEN=
  -e CONFIG=9
  -e AUTH_URL=
  -e PRINT_URL=
  -e BETSLIP_URL=
  -e BETS_URL=
  -e FE_URL=
  -e STATIC_URL=
  -o influxdb=$influxUrl:$influxPort/$dbName
  build/app.bundle.js > "$LOGS_DIR/bets-stg-$vus-$vus-$(date +%Y-%m-%d %H:%M:%S').log" 2>&1 &
```



```
█ Bets-Thread-3
  █ SignIn
    ✓ status is 200 OK
    ✓ login successful, accessToken is not empty
  █ betHistory
    ✓ status is 200 OK
  █ OpenBetsCount
    ✓ status is 200 OK
█ Bets-Thread-2
  █ SignIn
    ✓ login successful, accessToken is not empty
    ✓ status is 200 OK
  █ betHistory
    ✓ status is 200 OK
  █ OpenBetsCount
    ✓ status is 200 OK
█ Bets-Thread-1
```

```
█ OpenBetsCount
  ✓ status is 200 OK
█ SignIn
  ✓ status is 200 OK
  ✓ login successful, accessToken is not empty
█ GetEvents
  ✓ status is 200 OK
  █ placeBet Event 0
    ✓ status is 200 OK
    ✓ Bets has successfully returned
    ✓ Bets count eq 1
  █ placeBet Event 1
    ✓ status is 200 OK
    ✓ Bets has successfully returned
    ✓ Bets count eq 1
  █ betHistory
    ✓ status is 200 OK
  █ printCoupon
    ✓ status is 200 OK
    ✓ content-type is application/json
```





k6

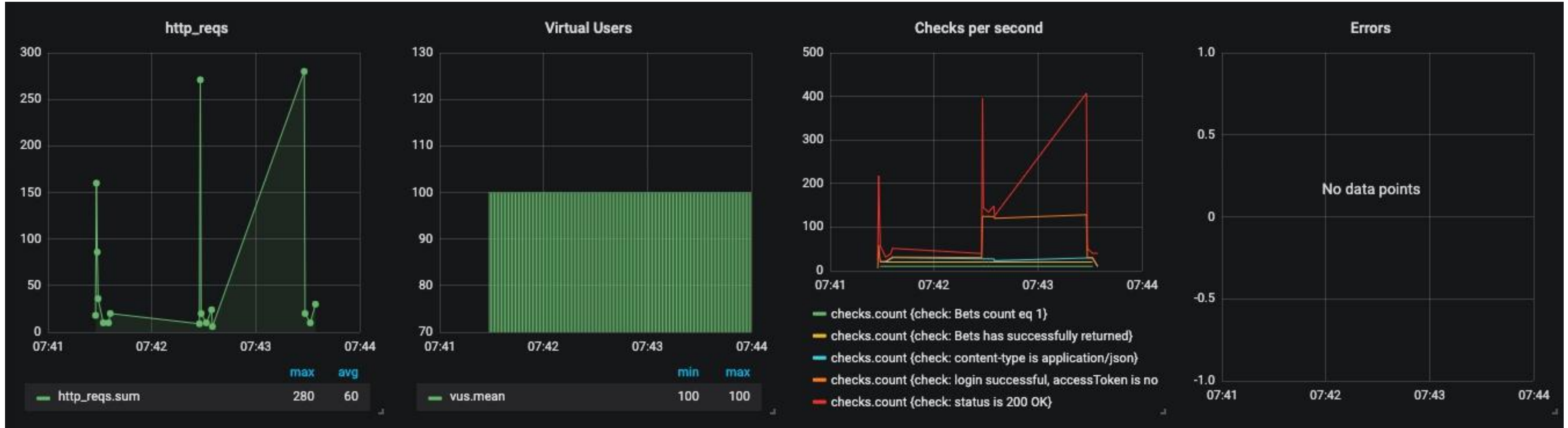
## Метрики. Вывод показателей в console

```

GetEvents.....: avg=99.390759 min=8.94012 med=15.782593 max=368.032159 p(90)=360.318773 p(95)=365.366075
GetOpenBetsCount.....: avg=7.363795 min=3.044463 med=6.186006 max=37.124272 p(90)=12.705848 p(95)=15.502449
GetPrintCoupon.....: avg=424.731275 min=74.916298 med=178.067034 max=1048.347874 p(90)=976.667897 p(95)=994.449579
betHistory.....: avg=9.805701 min=4.014421 med=7.611294 max=69.187101 p(90)=15.583727 p(95)=22.497075
checks.....: 100.00% / 1470 x 0
data_received.....: 4.3 MB 24 kB/s
data_sent.....: 1.3 MB 7.4 kB/s
errors.....: 0.00% / 0 x 1020
group_duration.....: avg=701.61ms min=3.35ms med=203.97ms max=8.8s p(90)=987.5ms p(95)=1.42s
http_req_blocked.....: avg=44.52ms min=455ns med=7.21ms max=1.02s p(90)=18.79ms p(95)=59.2ms
http_req_connecting.....: avg=35.57ms min=0s med=199.46µs max=1s p(90)=693.32µs p(95)=1.18ms
http_req_duration.....: avg=225.41ms min=3.04ms med=10.61ms max=1.04s p(90)=794.89ms p(95)=840.09ms
http_req_receiving.....: avg=122.96µs min=35.2µs med=73.7µs max=2.48ms p(90)=254.5µs p(95)=306.8µs
http_req_sending.....: avg=46.28µs min=9.08µs med=45.78µs max=239.54µs p(90)=83.56µs p(95)=102.93µs
http_req_tls_handshaking....: avg=7.96ms min=0s med=6.86ms max=50.11ms p(90)=12.2ms p(95)=46.54ms
http_req_waiting.....: avg=225.24ms min=2.96ms med=10.44ms max=1.04s p(90)=794.78ms p(95)=839.93ms
http_reqs.....: 1020 5.664932/s
iteration_duration.....: avg=1.52s min=110.53ms med=833.7ms max=8.8s p(90)=2.03s p(95)=7.38s
iterations.....: 300 1.666157/s
place.....: avg=228.390068 min=114.349459 med=179.924176 max=522.932147 p(90)=491.247434 p(95)=495.181721
placement_fail_counter.....: 0 0/s
placement_success_counter...: 60 0.333231/s
signIn.....: avg=651.148543 min=63.359297 med=645.656025 max=898.098645 p(90)=847.030984 p(95)=872.086624
vus.....: 100 min=100 max=100
vus_max.....: 100 min=100 max=100

```







Спасибо за внимание