

Average-load testing: A beginner's guide

Grafana Labs Team • 30 Jan, 2024 • 4 min



This content was originally published on k6.io.
30 Jan 2024

On this page

When to run an average-load test
Considerations
Average-load testing example
Results analysis



Grafana Cloud

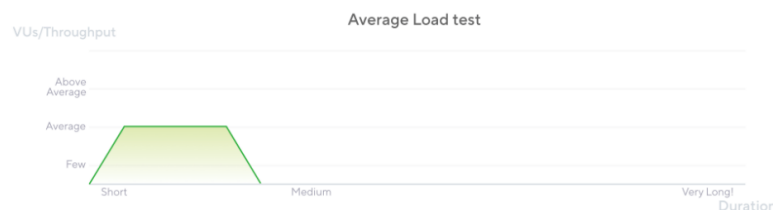
- ✓ Grafana, of course
- ✓ 10k series Prometheus metrics
- ✓ 50 GB logs
- ✓ 50 GB traces
- ✓ 50k frontend sessions
- ✓ 500 VUh k6 testing

[Create free account](#)

No credit card needed, ever.

An average-load test is a type of [load testing](#) that assesses how the system performs under typical load. Typical load might be a regular day in production or an average moment.

Average-load tests simulate the number of concurrent users and requests per second that reflect average behaviors in the production environment. This type of test typically increases the throughput or VUs gradually and keeps that average load for some time. Depending on the system's characteristics, the test may stop suddenly or have a short ramp-down period.



Since the term load test can often refer to all [types of load testing](#) that simulate traffic, this guide uses the name *average-load test* to avoid confusion. In some testing conversation, an average-load test also might be called a *day-in-life test* or *volume test*.

When to run an average-load test

Average-load testing helps understand whether a system **meets performance goals** on a typical day (commonplace load). *Typical day* here means when an average number of users access the application at the same time, doing normal, average work.

You should run an average-load test to:

- Assess the performance of your system under a typical load.
- Identify early degradation signs during the ramp-up or full-load periods.
- Assure that the system still meets the performance standards after system changes (code and infrastructure).

Considerations

When you prepare an average-load test, consider the following:

- **Know the specific number of users and the typical throughput per process in the system.** To find this, look through APMs or analytic tools that provide information from the production environment. If you can't access such tools, the business must provide these estimations.
- **Gradually increase load to the target average.** That is, use a ramp-up period. This period usually lasts between 5% and 15% of the total test duration. A ramp-up period has many essential uses:
 - It gives your system time to warm up or auto-scale to handle the traffic.
 - It lets you compare response times between the low-load and average-load stages.
 - If you run tests using a cloud service, such as [Grafana Cloud k6](#), a ramp up lets the automated performance alerts understand the expected behavior of your system.
- **Maintain average for a period longer than the ramp up.** Aim for an average duration at least five times longer than the ramp-up to assess the performance trend over a significant period of time.

- **Consider a ramp-down period.** The ramp down is when virtual user activity gradually decreases. The ramp down usually lasts as long as the ramp up or a bit less.

Average-load testing example

Note: If this is your first time running load tests, we recommend starting small or configuring the ramp-up to be slow. Your application and infrastructure might not be as rock solid as you think. We've had thousands of users run load tests that quickly crash their applications (or staging environments).

The goal of an average-load test is to simulate the average amount of activity on a typical day in production. The pattern follows this sequence:

1. Increase the script's activity until it reaches the desired number of users and throughput.
2. Maintain that load for a while
3. Depending on the test case, stop the test or let it ramp down gradually.

Here's how to configure load with the open source [load testing tool](#) Grafana k6 in the `options` object:

```
JavaScript Copy

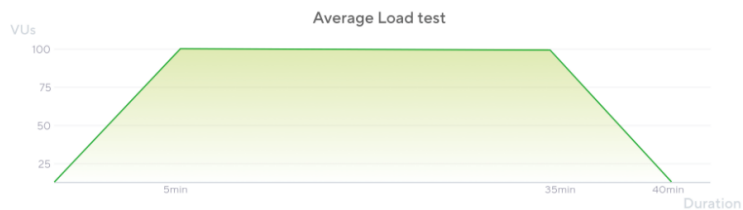
import http from 'k6/http';
import { sleep } from 'k6';

export const options = {
  // Key configurations for avg load test in this section
  stages: [
    { duration: '5m', target: 100 }, // traffic ramp-up from 1 to 100 users over 5 minutes.
    { duration: '30m', target: 100 }, // stay at 100 users for 30 minutes
    { duration: '5m', target: 0 }, // ramp-down to 0 users
  ],
};

export default () => {
  const urlRes = http.get('https://test-api.k6.io');
  sleep(1);
  // MORE STEPS
  // Here you can have more steps or complex script
  // Step1
  // Step2
  Expand code
```

This script logic has only one request (to open a web page). Your test behavior likely has more steps. If you would like to see more complex tests that use groups, checks, thresholds, and helper functions, you can find more [load testing examples](#) in our [Grafana k6 documentation](#).

The virtual users (VU) or throughput chart of an average-load test looks similar to this:



Results analysis

An initial outcome for the average-load test appears during the ramp-up period to find whether the response time degrades as the load increases. Some systems might even fail during the ramp-up period.

The test validates if the system's performance and resource consumption stay stable during the period of full load, as some systems may display erratic behavior in this period.


Once you know your system performs well and survives a typical load, you may need to push it further to determine how it behaves at above-average conditions. Some of these above-average conditions are known as [stress tests](#).

Grafana Cloud is the easiest way to get started with Grafana k6 and performance testing. We have a generous forever-free tier and plans for every use case. [Sign up for free now!](#)

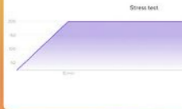
Tags

[k6](#) [Performance testing](#) [Load testing](#)

Up next

k6

What is stress testing?




Grafana Labs Team · 30 Jan 2024 · 4 min read

Stress testing: A beginner's guide

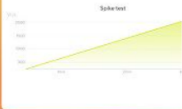
k6

Load testing

Performance testing

k6

What is spike testing?




Grafana Labs Team · 30 Jan 2024 · 4 min read

Spike testing: A beginner's guide

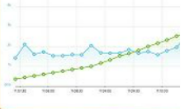
k6

Load testing

Performance testing

k6

What is soak testing?



Grafana Labs Team · 30 Jan 2024 · 4 min read

Soak testing: A beginner's guide

k6

Load testing

Performance testing



Sign up for Grafana stack updates

Email

Subscribe

Note: By signing up, you agree to be emailed related product-level information.



Grafana

[Overview](#)
[Deployment options](#)
[Plugins](#)
[Dashboards](#)

Products

[Grafana Cloud](#)
[Grafana Cloud Status](#)
[Grafana Enterprise Stack](#)
[Grafana Cloud Application Observability](#)
[Grafana Cloud Frontend Observability](#)
[Grafana Cloud IRM](#)
[Grafana Cloud k6](#)
[Grafana Cloud Logs](#)
[Grafana Cloud Metrics](#)
[Grafana Cloud Profiles](#)
[Grafana Cloud Synthetic Monitoring](#)
[Grafana SLO](#)

Open Source

[Grafana](#)
[Grafana Loki](#)
[Grafana Mimir](#)
[Grafana OnCall](#)
[Grafana Tempo](#)
[Grafana Agent](#)
[Grafana Alloy](#)
[Grafana k6](#)
[Prometheus](#)
[Grafana Faro](#)
[Grafana Pyroscope](#)
[Grafana Beyla](#)
[OpenTelemetry](#)
[Grafana Tanka](#)
[Graphite](#)
[GitHub](#)

Learn

[Grafana Labs blog](#)
[Documentation](#)
[Downloads](#)
[Community](#)
[Community forums](#)
[Community Slack](#)
[Grafana Champions](#)
[Community organizers](#)
[Grafana ObservabilityCON](#)
[GrafanaCON 2024](#)
[The Golden Grot Awards](#)
[Successes](#)
[Workshops](#)
[Videos](#)
[OSS vs Cloud](#)
[Load testing](#)

Company

[The team](#)
[Press](#)
[Careers](#)
[Events](#)
[Partnerships](#)
[Contact](#)
[Getting help](#)
[Merch](#)

[Grafana Cloud Status](#)

[Legal and Security](#) [Terms of Service](#) [Privacy Policy](#) [Trademark Policy](#)

Copyright 2024 © Grafana Labs