



What is stress testing?



Stress testing: A beginner's guide

Grafana Labs Team • 30 Jan, 2024 • 4 min



On this page

When to perform a stress test

Considerations

Stress testing in k6

Stress test results: analysis



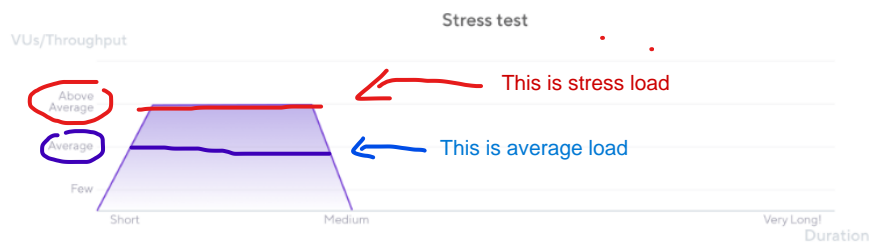
Grafana Cloud

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

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

Stress testing assesses how the system performs when loads are heavier than usual.

The load pattern of a stress test resembles that of an average-load test. The main difference is higher load. To account for higher load, the ramp-up period takes longer in proportion to the load increase. Similarly, after the test reaches the desired load, it might last for slightly longer than it would in the average-load test.



In some testing conversations, stress tests might also be called *rush-hour*, *surge*, or *scale tests*.

When to perform a stress test

Stress tests are a type of *load testing* that verifies the stability and reliability of the system under conditions of heavy use. Systems may receive higher than usual workloads on unusual moments, such as process deadlines, paydays, rush hours, ends of the workweek, and many other behaviors that might cause frequent higher-than-average traffic.

Considerations

When you run a stress test, consider the following:

- Load should be higher than what the system experiences on average.

Some testers might have default targets for stress tests—say an increase of 50 or 100 percent compared to an average load. Overall, there's no fixed percentage.

The load simulated in a stress test depends on the situations that the system may be subjected to during the test. Sometimes this may be just a few percentage points above that average. Other times, it can be 50 to 100 percent higher, as mentioned. Some stressful situations can even be orders of magnitude higher.

In the end, define the load in stress testing according to the risk load patterns that the system may receive.



Only run stress tests after running average-load tests.

Identify performance issues under average-load tests before trying anything more challenging. This sequence is essential.

- **Re-use the average-load test script.**

Modify the parameters to have higher load or VUs.

- **Expect performance to be worse than an average-load test.**

(This test determines how much the performance degrades with the extra load and whether the system survives it.) The goal is a well-performing system that produces consistent response times when handling a constant workload for an extended period.

Stress testing in k6

The load in a stress test starts in a way that resembles load in an average-load test. The difference is that it reaches a higher level of load over the course of the test.

1. **Increase** the script's activity further in a slower ramp-up until it reaches an above-average number of users or throughput.
2. Maintain that load for a while.
3. Depending on the test case, stop or ramp down gradually.

Here is a test script for the open source [load testing tool](#) Grafana k6:

```
JavaScript

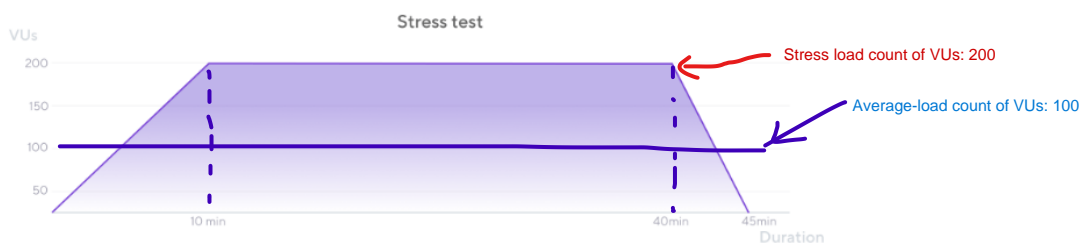
import http from 'k6/http';
import { sleep } from 'k6';

export const options = {
  // Key configurations for Stress in this section
  stages: [
    { duration: '10m', target: 200 }, // traffic ramp-up from 1 to a higher 200 users over 10 minutes
    { duration: '30m', target: 200 }, // stay at higher 200 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
}
```

For more complex behavior, refer to more [load testing examples](#) in our [k6 documentation](#).

The VU or throughput chart of a stress test looks similar to this:



Note that similar to the average-load test, the stress test starts at 0 but increases beyond the point tested in the average-load type. The ramp-up and ramp-down periods are longer to allow for a more realistic response.

Note: Run stress tests only after smoke and average-load tests. Running this test type earlier may be wasteful and make it hard to pinpoint problems if they appear at low volumes or at loads under the average utilization.

Stress test results: analysis

Like the [average-load test](#), an initial outcome for the stress test shows up during the ramp-up period to identify response time degradation as the load increases further than the average utilization. Commonly, the performance degrades, and even the system's stability crashes as we

push the system further than the average-load test.

During the full load period, verification is vital if the system's performance and resource consumption stays stable with a higher load.

Now that you know that your system can handle outstanding load events, the teams generally check if the system performs well over extended periods. That is, they run a soak test.


Smoke test -> Average load test -> Stress test -> Soak test

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
- Load testing
- Performance testing

Up next



What is spike testing?



Grafana Labs Team · 30 Jan 2024 · 4 min read
Spike testing: A beginner's guide

- k6
- Load testing
- Performance testing



What is soak testing?



Grafana Labs Team · 30 Jan 2024 · 4 min read
Soak testing: A beginner's guide

- k6
- Load testing
- Performance testing



What is smoke testing?



Grafana Labs Team · 30 Jan 2024 · 3 min read
Smoke 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