Vegeta HTTP load testing

Targets stdin or file → `attack` → **Results** binary stream → `report / plot / encode` → **Output** text, JSON, HTML, CSV

CONCEPTS

Vegeta sends a constant request rate, regardless of how slow the server is. Latency grows instead of throughput dropping.

Results are a binary stream: write it to a file with > or tee, since the report cannot be regenerated otherwise.

Vegeta reads targets, not URLs: even a single URL needs a GET line piped in or in a targets file.

ATTACK

`vegeta attack`
Fire requests from stdin targets

`-rate=<n>/<unit>`
Requests per time unit (default 50/1s)

`-duration=<t>`
How long to attack (0 = forever)

`-targets=<file>`
Read targets from a file, not stdin

`-output=<file>`
Write binary results to a file

ATTACK (cont.)

`-rate=0 -max-workers=<n>`
Go as fast as possible

`-header="<Name: value>"`
Add a request header (repeatable)

`-body=<file>`
Body file for every request

`-timeout=<t>`
Per-request timeout (default 30s)

`-workers=<n>`
Initial workers (default 10, grows)

`-max-workers=<n>`
Cap the worker pool

`-name=<name>`
Label the attack in results

`-insecure`
Ignore invalid TLS certificates

`-http2=false`
Force HTTP/1.1

`-keepalive=false`
Open a new connection per request

`-redirects=<n>`
Max redirects (-1 = don't follow)

`-format=json`
Targets are JSON lines

TARGETS

HTTP targets file

```
# one request per block, blank line between
GET http://localhost:8080/items?x=1
X-Account-ID: 99

POST http://localhost:8080/things
Content-Type: application/json
@./newthing.json
```

JSON targets (one per line)

```
{"method":"GET", "url":"http://localhost/", "header":{"X-Id":["1"]}, "body":"<base64>"}
```

In the HTTP format a body must be a file reference with @path; inline bodies are not supported. JSON targets take base64 bodies.

Lines starting with # in a targets file are comments.

REPORT

`vegeta report <file>.bin`
Text summary of an attack

`cat <file>.bin | vegeta report`
Read results from stdin

`-type=json`
Metrics as JSON

REPORT (cont.)

`-type=hist[0,100ms,200ms]`
Latency histogram with buckets

`-type=hdrplot`
HDR histogram plot data

`-every=<t>`
Print a report at each interval

`vegeta report *.bin`
Merge several runs by timestamp

Reading the text report

Field	Meaning
Requests	Total, rate sent, throughput
Latencies	Min, mean, p50/p90/p95/p99, max
Success	Share of 200-399 responses
Status Codes	Count per code; 0 = send failed
Error Set	Distinct errors seen

PLOT AND ENCODE

`vegeta plot <file>.bin > <file>.html`
Interactive latency-over-time chart

`vegeta plot -title=<text>`
Set the chart title

PLOT AND ENCODE (cont.)

`vegeta plot a.bin b.bin`
Compare runs in one chart

`vegeta encode -to=csv <file>.bin`
Convert results to CSV

`vegeta encode -to=json <file>.bin`
Convert results to JSON lines

GLOBAL

`vegeta -version`
Print version

RECIPES

Quick smoke test

```
echo "GET http://localhost:8080/" | \
  vegeta attack -rate=100 -duration=10s | \
  tee results.bin | vegeta report
```

Compare two rates on one chart

```
echo "GET http://localhost/" > t.txt
vegeta attack -targets=t.txt -name=50qps -rate=50 -duration=10s > r50.bin
vegeta attack -targets=t.txt -name=100qps -rate=100 -duration=10s > r100.bin
vegeta plot r50.bin r100.bin > plot.html
```