wrk HTTP load tester

CONCEPTS

> Load test only systems you own or have written permission to test. wrk can take a server down.

> Connections are split across threads (connections / threads each), so -c must be at least -t. Use as many threads as cores, no more.

setup once per thread
↓ start
init per thread, gets args
↓ loop
request / response every call
↓ end
done final summary

How a script hooks into a run

RUNNING A TEST

wrk -t<*n*> **-c**<*n*> **-d**<*time*> <*url*>
Run a benchmark

-t, --threads <*n*>
Worker threads

-c, --connections <*n*>
Open connections in total

-d, --duration <*time*>
Test length (2s, 2m, 2h)

--latency
Print latency distribution

--timeout <*time*>
Timeout per response

-H, --header "<*name: value*>**"**
Add a request header

-s, --script <*file*>
Load a Lua script

> Raise open-file and ephemeral-port limits (ulimit -n) on the client, and keep the server's listen backlog above -c.

READING OUTPUT

Output lines

Line	Meaning
Latency	Per-request time: avg, stdev, max
Req/Sec	Per-thread rate, not total
Requests/sec	Total throughput
Transfer/sec	Bytes read per second
Socket errors	connect/read/write/timeout counts
Non-2xx or 3xx	Responses with error status

> Check for socket errors and non-2xx counts first: fast errors inflate requests/sec.

> With --latency read the 99% line, not the average.

LUA SCRIPTING

wrk.method = "POST"
Set the HTTP method

wrk.body = '<*data*>**'**
Set the request body

wrk.headers["<*name*>**"] = "**<*value*>**"**
Set a header

function request()
Return the next request

wrk.format(method, path, headers, body)
Build a request string

function response(status, headers, body)
Inspect each response

function init(args)
Per-thread setup

function done(summary, latency, requests)
Custom final report

> Static changes (method, path, headers, body) cost nothing. Building a request per call or using response() lowers the load wrk can generate.