Stress test for Websocket

how I use tsung to stress test Phoenix

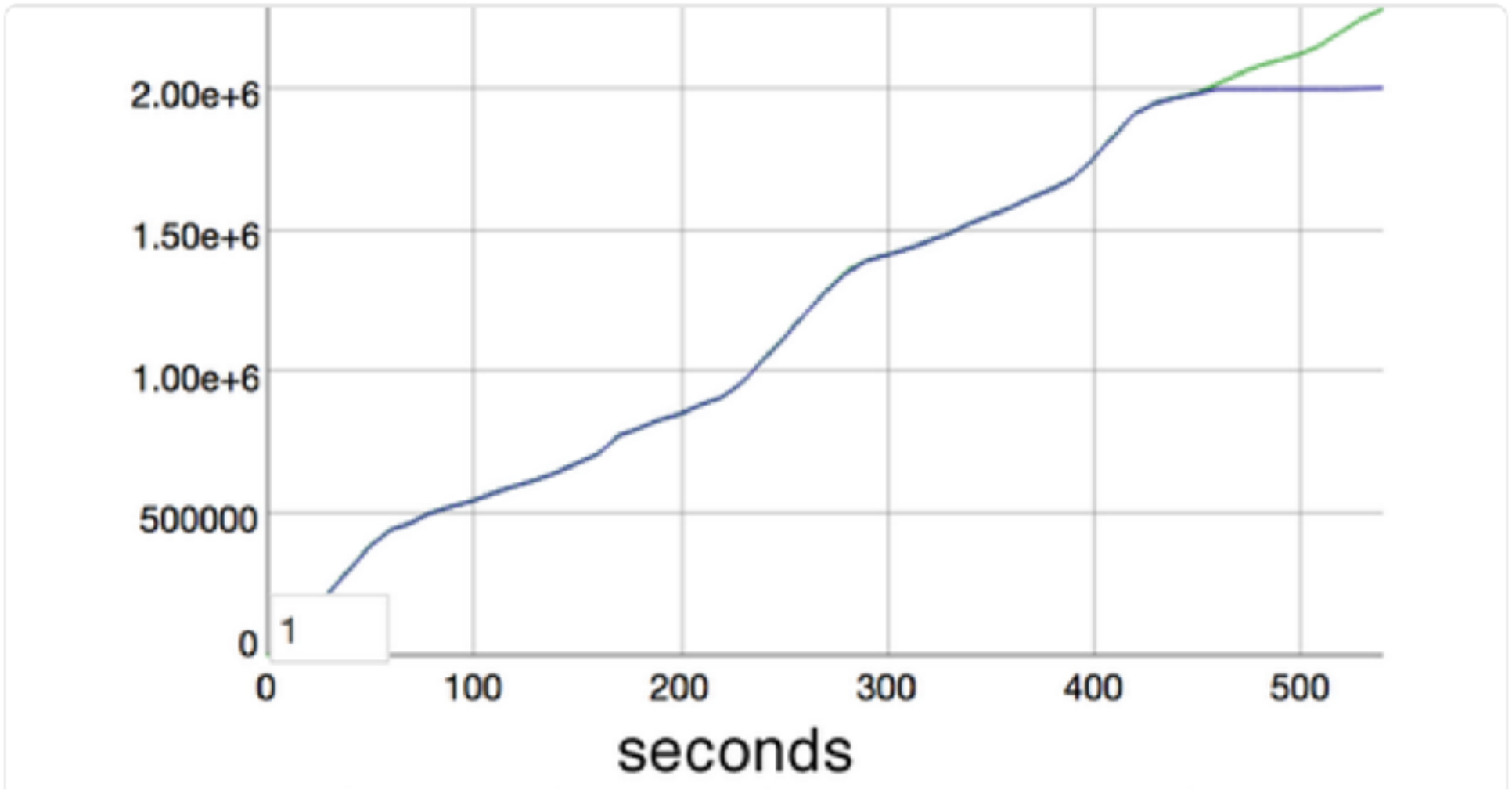
Why?

We have no experience about Elixir/ Phoenix's performance

Chris McCord
@chris_mccord

Following

Final results from Phoenix channel benchmarks on 40core/128gb box. 2 million clients, limited by ulimit
#elixirlang



What they test is not your code

How?

Find hundreds of users for testing your application?

Tools

Feature	The Grinder	Gatling	Tsung	JMeter
OS	Any	Any	Linux/Unix	Any
GUI	Console Only	Recorder Only	No	Full
Test Recorder	TCP (including HTTP)	HTTP	HTTP, Postgres	HTTP
Test Language	Python, Clojure	Scala	XML	XML
Extension Language	Python, Clojure	Scala	Erlang	Java, Beanshell, Javascript, Jexl
Load Reports	Console	HTML	HTML	CSV, XML, Embedded Tables, Graphs, Plugins
Protocols	HTTP SOAP JDBC POP3 SMTP LDAP JMS	HTTP JDBC JMS	HTTP WebDAV Postgres MySQL XMPP WebSocket AMQP MQTT	HTTP FTP JDBC SOAP LDAP TCP JMS SMTP

Tsung

- Support HTTP, Websocket
- High Performance(the load can be distributed on a cluster of client machines)
- OS monitoring
- HTML reports(Graph)
- Document is good enough
- Under active development by ProcessOne(1000 stars, last commit 20 days ago)
- Phoenix uses it for stress testing XD

tsung uses XML as config

```
<?xml version="1.0"?>
<!DOCTYPE tsung SYSTEM "/usr/local/share/tsung/tsung-1.0.dtd">
<tsung loglevel="info">
...
</tsung>
```

```
  <clients>
     <!-- use localhost, no distribution -->
     <client host="localhost" use_controller_vm="true"
weight="1" maxusers="50000"/>
     <!-- use remote nodes -->
     <client host="client1" cpu="4" use_controller_vm="false"
weight="1" maxusers="50000" />
   </clients>
```

ulimit -n 2,000,000

```
<servers>
    <server host=“api.foo.com” port="8080" type="tcp"/>
</servers>
```

monitoring

```
 <load>
    <arrivalphase phase="1" duration="1000" unit="second">
      <users maxnumber="40000" arrivalrate="1000"
unit="second"/>
    </arrivalphase>
</load>
```

```
<options>
    <option name="ports_range" min="1025" max="65535"/>
</options>
```

```
 <sessions>
    <session name="websocket" probability="100"
type="ts_websocket">
      <request subst="true">
        <websocket type="connect" path="/guest_socket/
websocket?email=user%%ts_user_server:get_unique_id%
%@stress.test"/>
      </request>
    </session>
  </sessions>
```

```
      <request subst="true">
         <websocket type="message">{"topic":"event:general:user%
%ts_user_server:get_unique_id%%@stress.test",
"event":"phx_join", "payload": {}, "ref":"1"}</websocket>
      </request>
```

```
        <for var="i" from="1" to="1000" incr="1">
            <request>
              <websocket ack="no_ack"
type="message">{"topic":"phoenix","event":"heartbeat","
payload":{},"ref":"3"}</websocket>
            </request>
            <thinktime value="30" random="true"/>
        </for>
```

$ tsung -f config.xml -k start

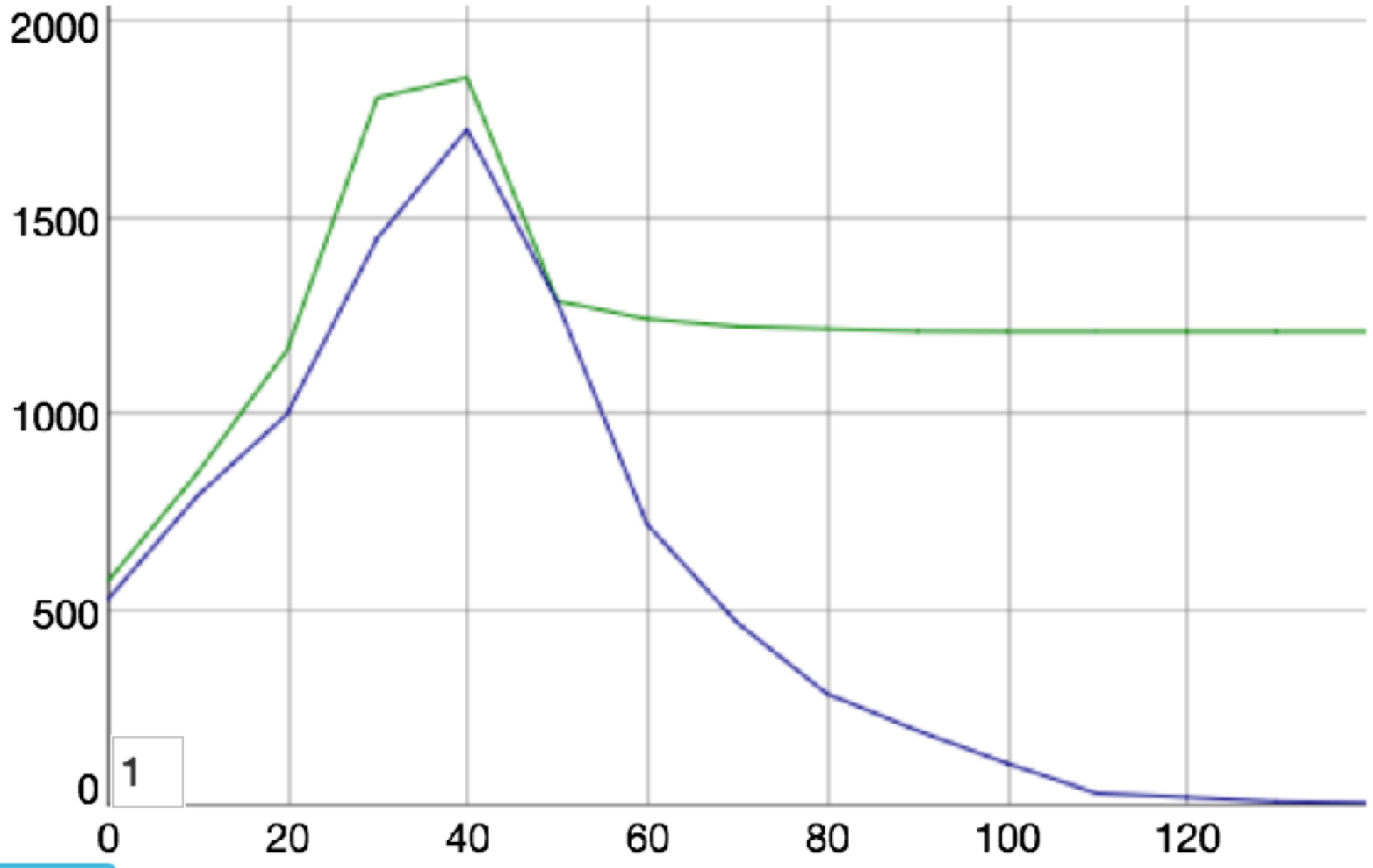
On a 2 cores, 4G memory, 20GB storage
(AWS t2.medium)

client.host:8091/es/ts_web:graph

<users maxnumber="4000"
arrivalrate="100" unit="second"/>

Make it work, then make it better

Simultaneous Users



Info »

log: ~/.tsung/log/

log: ~/.tsung/log/

- Error: Unknown msg {tcp_closed, #Port<0.3140>} receive in state wait_ack, stop
- Stop in state wait_ack, reason= {function_clause,
- ts_client:(6:<0.672.0>) connection closed while waiting for ack

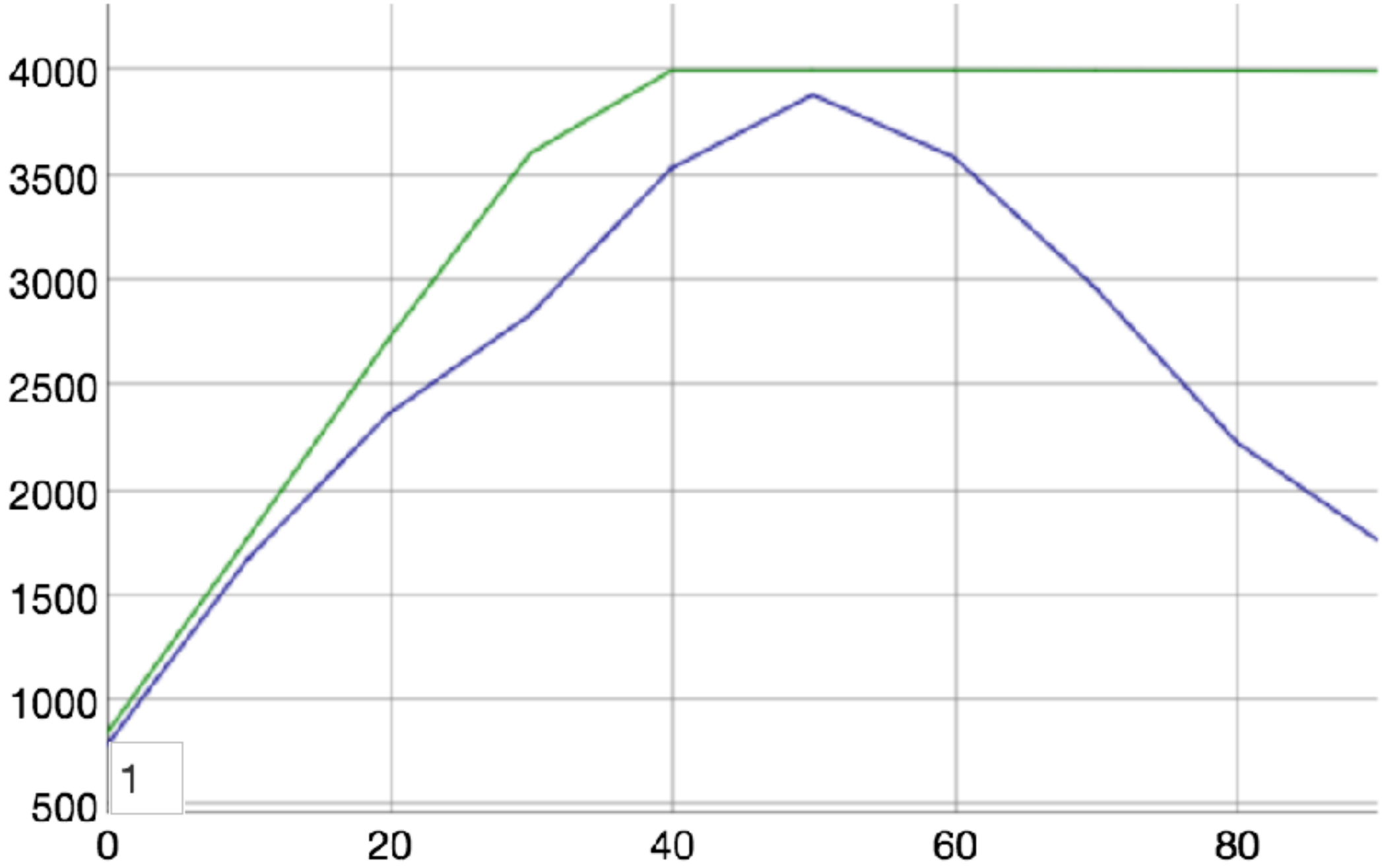
nginx error:

768 worker_connections are not enough

```
events {
   worker_connections 20000;
}
```

```
$ sudo service nginx reload
```

Simultaneous Users



Info »

“By default, the connection will be closed if the proxied server does not transmit any data within 60 seconds. This timeout can be increased with the proxy_read_timeout directive.”

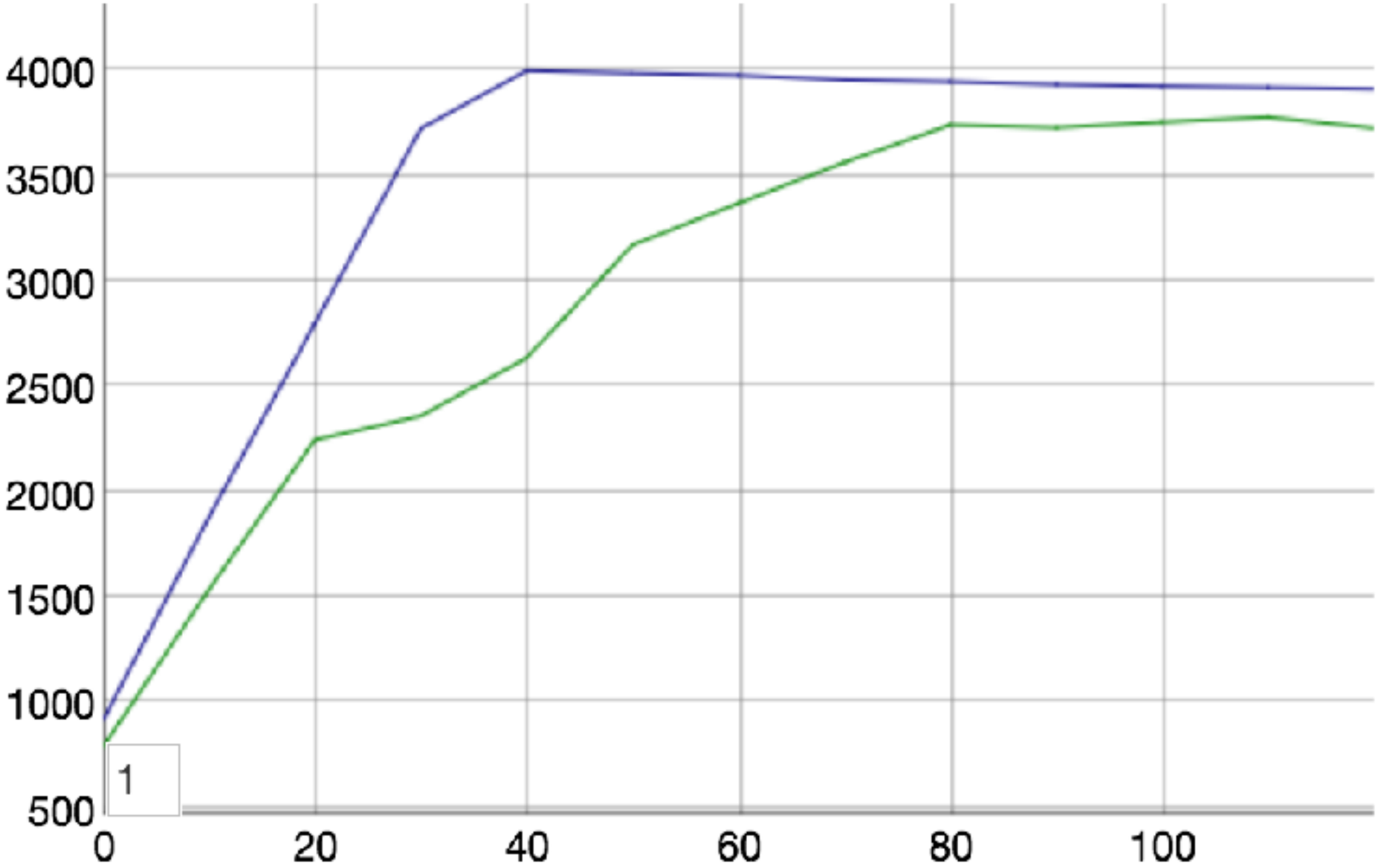
–http://nginx.org/en/docs/http/websocket.html

```
proxy_read_timeout 1d;
```

~~proxy_read_timeout 1d;~~

```
        <for var="i" from="1" to="1000" incr="1">
            <request>
              <websocket ack="no_ack"
type="message">{"topic":"phoenix","event":"heartbeat","
payload":{},"ref":"3"}</websocket>
            </request>
            <thinktime value="30"/>
          </for>
```

Simultaneous Users

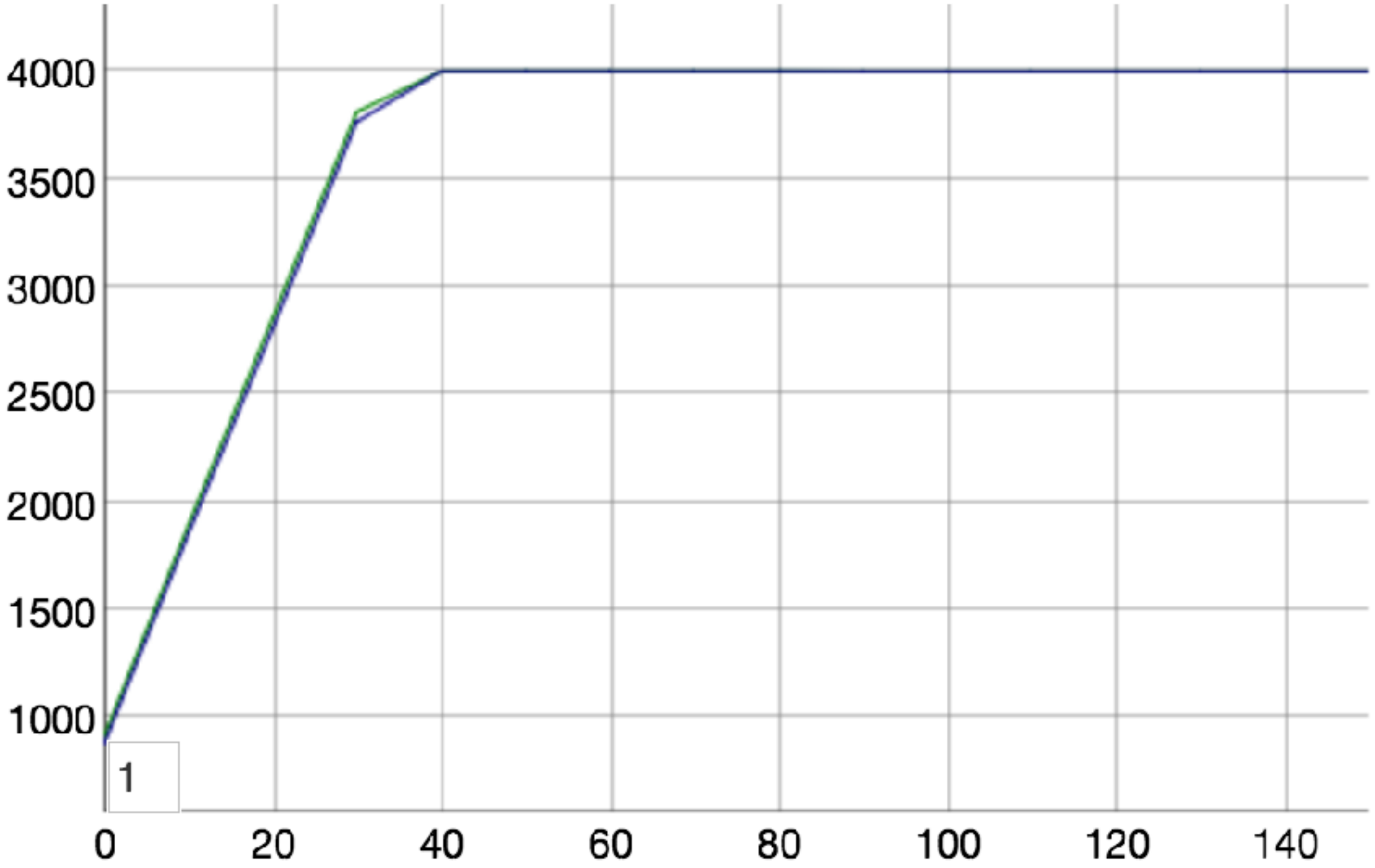


Info »

- The network of my mac got stuck
- No obvious errors found

Run test on a Linux server(client)

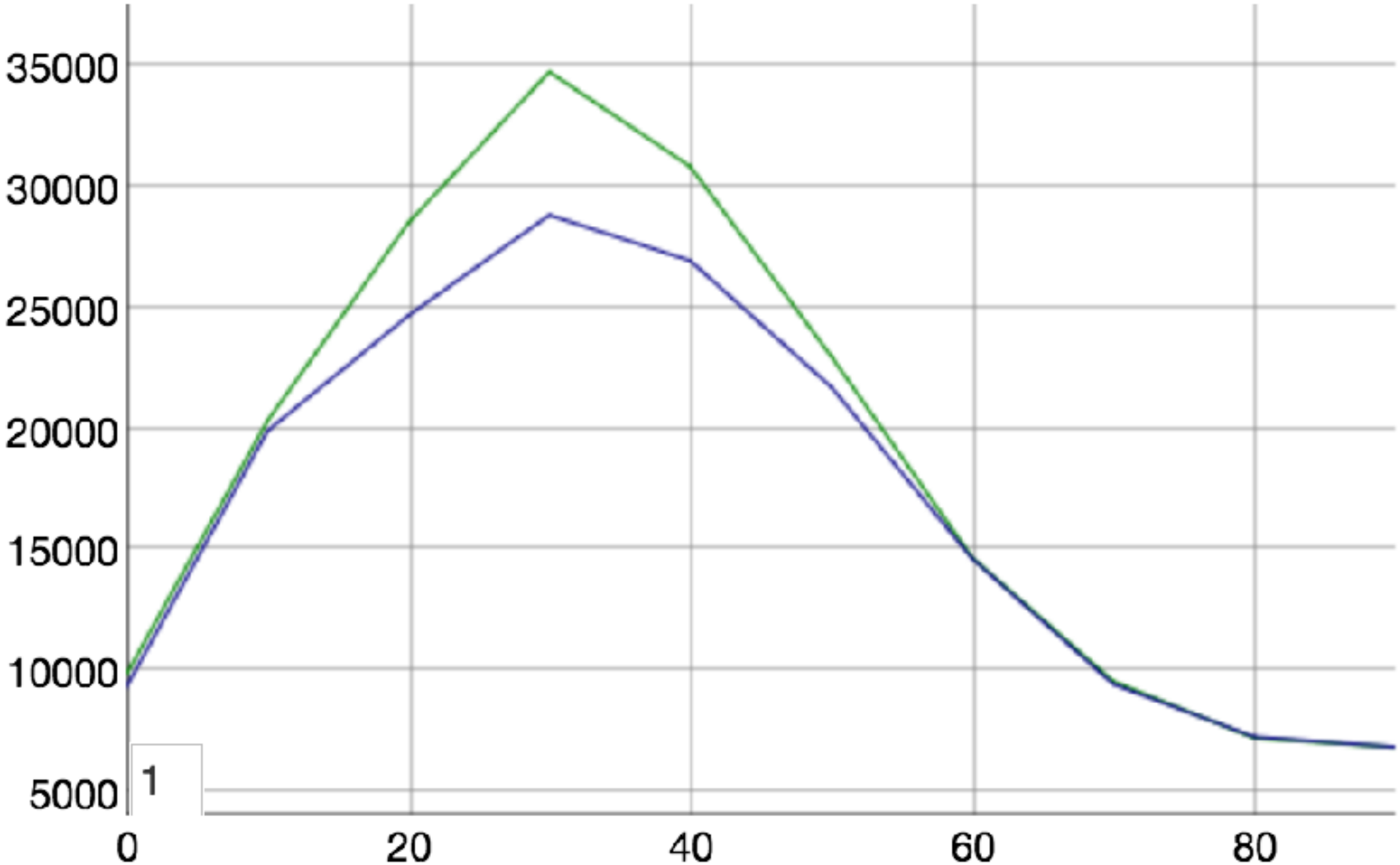
Simultaneous Users



Info »

```
    <arrivalphase phase="1" duration="1000" unit="second">
        <users maxnumber="40000" arrivalrate="1000"
unit="second" />
    </arrivalphase>
```

Simultaneous Users



Info »

Simultaneous Users

nginx error log

- upstream timed out (110: Connection timed out) while connecting to upstream

- recv() failed (104: Connection reset by peer) while reading response header from upstream

- upstream prematurely closed connection while reading response header from upstream, client: , server: , request: "GET /socket/websocket?token= HTTP/1.1", upstream: "http://127.0.0.1:8080/socket/websocket?token=", host: ""

erlang log

- 15:41:40.173 [error] #PID<0.3785.2> running Phoenix.Transports.WebSocket terminated

- Server: xxx:80 (http)

- Request: GET /socket/websocket?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

- ** (exit) exited in: :gen_server.call(:jose_server, {:json_module, Poison})

- ** (EXIT) time out

erlang log

- 16:16:21.243 request_id=bbjbc1mi2jipdiihi5oob6v71ocqjrih [info] Sent 201 in 62ms
  ...
  16:16:54.145 request_id=scfe8i945c2tspipuasuf05omp3bv78p [info] POST /api/sessions
- 16:16:54.145 request_id=guhuhbasm07ujoudfi3b0fu8en8gj4oo [info] Sent 201 in 8202ms
- 16:16:54.146 [warn] Error: {:badarg, ["undefined"]}
- 16:16:54.147 [warn] Error: {:badarg, ["undefined"]}
- 16:16:54.147 request_id=8pi819hb1gf3nn5hdvncga8tj6u6uo9m [info] Sent 201 in 8209ms
- 16:16:54.147 request_id=2un3ki4q9it6niaag3r3001ma813a5l8 [info] Sent 201 in 8205ms

erlang log

- Request: POST /api/sessions
- ** (exit) exited in: :gen_server.call(Fluensay.Repo.Pool, {:checkout, #Reference<0.0.262146.227512>, true}, 5000)
- ** (EXIT) time out

nginx log

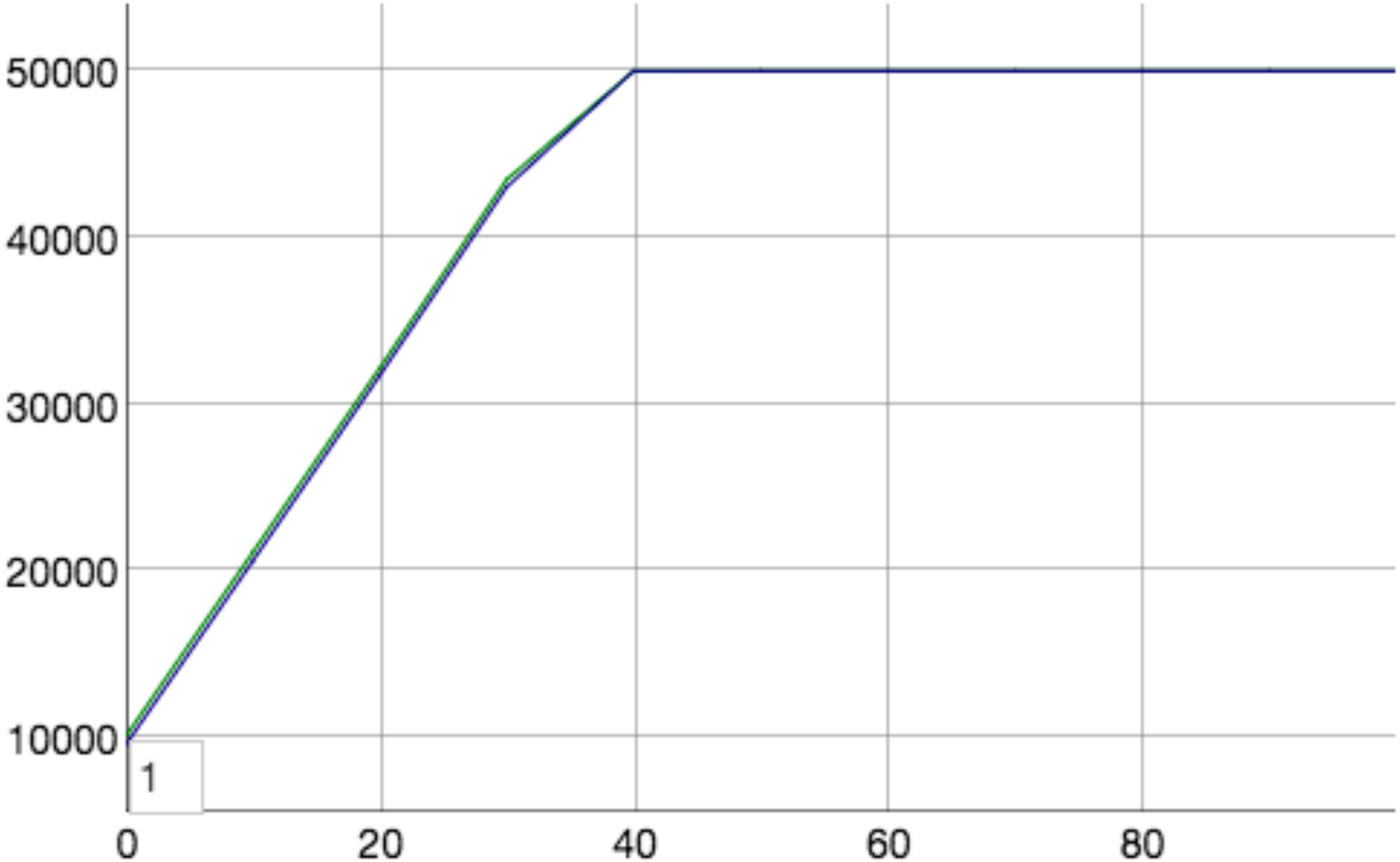
- "POST /api/sessions HTTP/1.1" 201 608 "-" "tsung" 0.004
- "POST /api/sessions HTTP/1.1" 201 614 "-" "tsung" 0.128
- ...
- "POST /api/sessions HTTP/1.1" 201 620 "-" "tsung" 2.506
- "POST /api/sessions HTTP/1.1" 201 620 "-" "tsung" 6.094
- "GET /socket/websocket?token=... HTTP/1.1" 101 503 "-" "-" 90.045

But CPU/Memory/Disk IO and pg log(queries can be done in 1ms) are normal

Then I write a guest_socket for only Websocket connection

```
def connect(%{"email" => email} = params, socket) do
    user = Repo.get_by(User, email: email)
    user = if user, do: user, else: create_user(params)
    if user do
      {:ok, assign(socket, :current_user, user)}
    else
      :error
    end
end
```

Simultaneous Users



Info »

Simultaneous Users

Above 50k, the progress will be killed by OOM

Then I try to use wrk to benchmark Phoenix

https://github.com/tony612/phoenix-benchmarker

On the server

```
Running 30s test @ http://localhost:8001/api/sessions
  12 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   247.42ms   43.16ms   1.08s    96.61%
    Req/Sec   336.80     92.95   787.00     74.84%
  Latency Distribution
     50%  244.97ms
     75%  250.57ms
     90%  255.74ms
     99%  404.72ms
  118405 requests in 30.05s, 35.68MB read
Requests/sec:   3940.28
Transfer/sec:      1.19MB
```

On my computer

```
Running 30s test @ http://127.0.0.1:8080/api/sessions
  12 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   186.07ms   50.01ms 360.32ms   79.50%
    Req/Sec   382.16    115.05     1.28k    82.66%
  Latency Distribution
     50%  202.20ms
     75%  212.86ms
     90%  225.60ms
     99%  276.65ms
  136913 requests in 30.10s, 41.41MB read
  Socket errors: connect 0, read 943, write 0, timeout 0
Requests/sec:   4547.97
Transfer/sec:      1.38MB
```

From my computer to the server

```
Running 30s test @ http://remote.host:8001/api/sessions
  12 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.84s     3.57s   28.38s    90.34%
    Req/Sec    88.32     58.00   383.00     63.39%
  Latency Distribution
     50%  293.74ms
     75%    1.68s
     90%    5.20s
     99%   18.58s
  28124 requests in 30.08s, 8.45MB read
  Socket errors: connect 0, read 543, write 0, timeout 0
Requests/sec:    934.90
Transfer/sec:    287.65KB
```

- 05:29:11.591 request_id=esm6c075herne7hsrg2thjlkdfkct29q [info] Sent 201 in 7ms
- 05:29:11.592 request_id=51c359ptohvgli4smr63jhj0jinrgb65 [info] Sent 201 in 8ms
- 05:29:11.606 request_id=9sk220bn5l72hs2atuqdm6qhu8p38fq3 [info] Sent 201 in 20ms
- 05:29:11.607 request_id=kmq605kfo2gbcp3euj1s403n7erm6e6p [info] Sent 201 in 21ms
- 05:29:11.619 request_id=pn7n70bvfe2ith63ofr9qgthplekj9tg [info] Sent 201 in 31ms
- 05:29:11.620 request_id=krppumbiibac7ldto1oo8sna8jvbeq45 [info] Sent 201 in 32ms

So...

一脸懵逼

Never trust benchmarks,
always measure yourself

Q&A