

1) El perfilamiento del servidor, realizando el test con --prof de node.js. Analizar los resultados obtenidos luego de procesarlos con --prof-process.

Utilizaremos como test de carga Artillery en línea de comandos, emulando 50 conexiones concurrentes con 20 request por cada una. Extraer un reporte con los resultados en archivo de texto.

## con console.log:

[Summary]:

```
ticks total nonlib name
  30  0.0% 100.0% JavaScript
   0  0.0%  0.0% C++
  61  0.1% 203.3% GC
67180 100.0%      Shared libraries
```

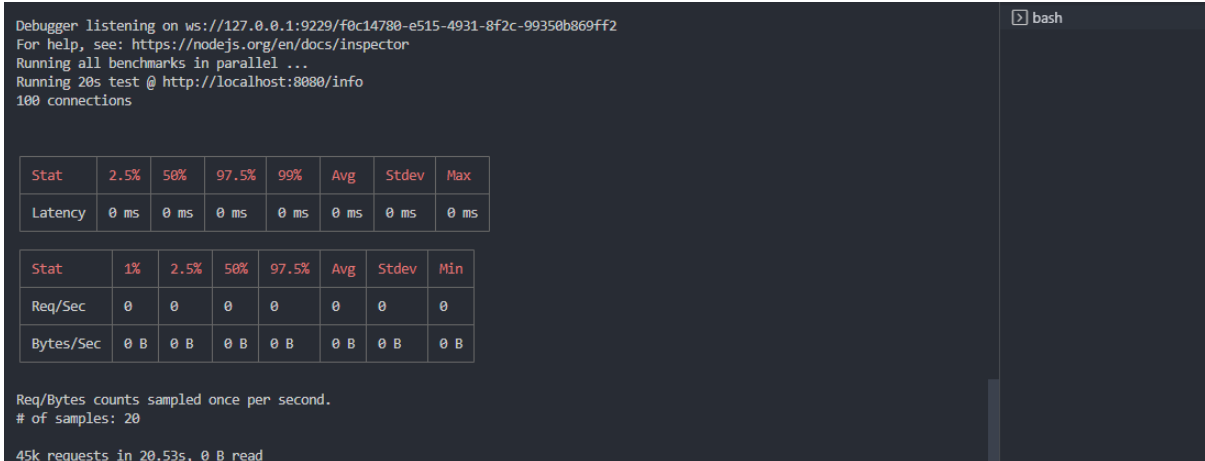
## sin console.log

[Summary]:

```
ticks total nonlib name
  30  0.1% 100.0% JavaScript
   0  0.0%  0.0% C++
  52  0.3% 173.3% GC
20362 99.9%      Shared libraries
```

Luego utilizaremos Autocannon en línea de comandos, emulando 100 conexiones concurrentes realizadas en un tiempo de 20 segundos.

## con console.log



```
Debugger listening on ws://127.0.0.1:9229/f0c14780-e515-4931-8f2c-99350b869ff2
For help, see: https://nodejs.org/en/docs/inspector
Running all benchmarks in parallel ...
Running 20s test @ http://localhost:8080/info
100 connections
```

Stat	2.5%	50%	97.5%	99%	Avg	Stdev	Max
Latency	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms

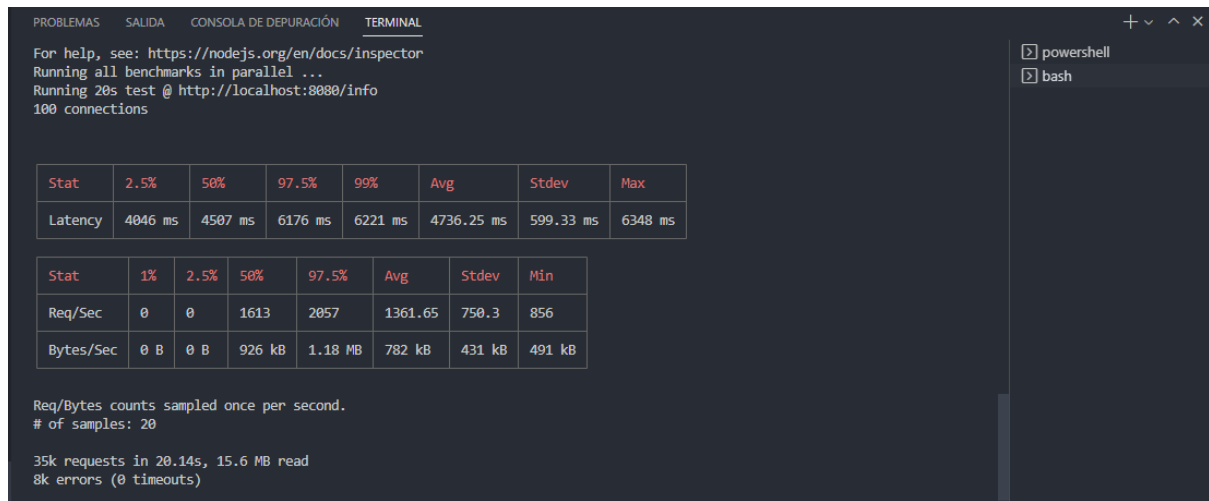
  

Stat	1%	2.5%	50%	97.5%	Avg	Stdev	Min
Req/Sec	0	0	0	0	0	0	0
Bytes/Sec	0 B	0 B	0 B	0 B	0 B	0 B	0 B

Req/Bytes counts sampled once per second.  
# of samples: 20

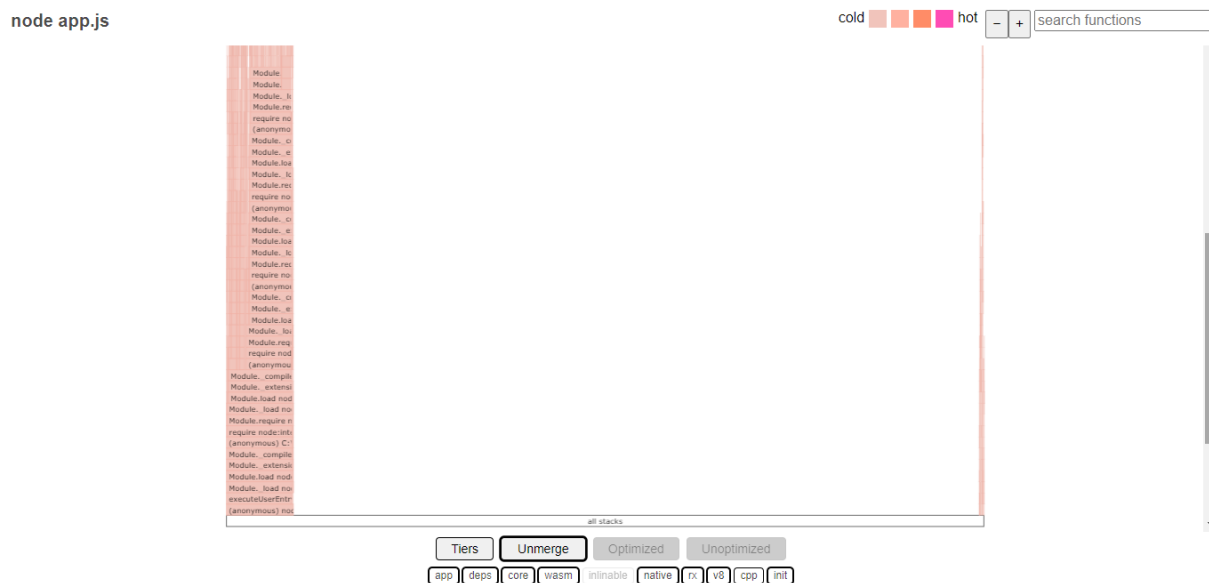
45k requests in 20.53s, 0 B read

## sin console.log

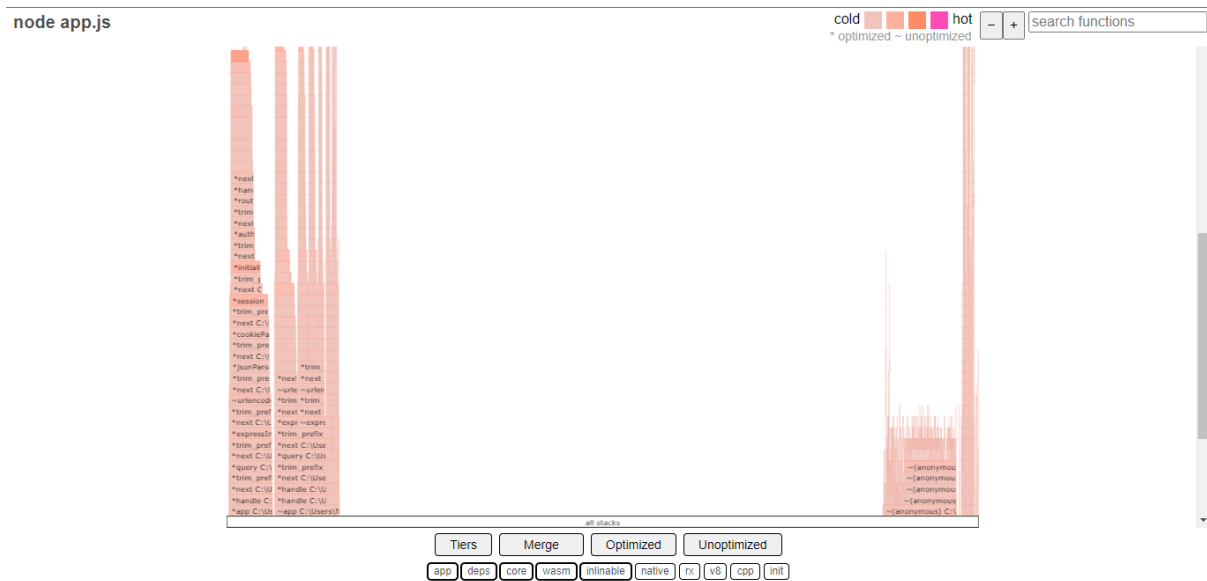


3) El diagrama de flama con 0x, emulando la carga con Autocannon con los mismos parámetros anteriores.

## con console.log



## sin console.log



**conclusión:**

Como podemos ver las operaciones sincrónicas afectan el rendimiento de nuestro servidor, es por esto que debemos evitar usarlas.