ANÁLISIS DE PERFORMANCE

Clase 32. Programación Backend

por Marcos Solis Santiago

CODER HOUSE

ANÁLISIS DE PERFORMANCE

INSTRUCCIONES

Sobre la ruta '/info', en modo fork, se agrega y extrae un console.log de la información colectada antes de devolverla al cliente. El child_process de la ruta '/randoms' estará desactivado.

Obtener:

1) El perfilamiento del servidor, realizando el test con --prof de node.js. Análisis de resultados después de procesarlos con --prof-process.
   Extraer reporte con los resultados de utilizar como test de carga Artillery en línea de comandos emulando 50 conexiones concurrentes con 20 request por cada una en archivo de texto.
   Emulando 100 conexiones concurrentes en 20 segundos con Autocannon en línea de comandos, extraer un reporte con los resultados.
2) El perfilamiento del servidor con el modo inspector de node.js --inspect. Revisar el tiempo de los procesos menos performantes sobre el archivo fuente de inspección.
3) El diagrama de flama con 0x, emulando la carga con Autocannon con los mismos parámetros anteriores.

Incluir en este informe sobre incluyendo los resultados de todas las pruebas realizadas y la conclusión obtenida a partir del análisis de los datos.

PERFILAMIENTO DEL SERVIDO CON --prof DE NODE.JS

Comandos del análisis:

```
[root@DESKTOP-0PD4IF7 LOGGERS-GZIP-Performance_analisis]# node --prof main.js
```

```
[root@DESKTOP-0PD4IF7 LOGGERS-GZIP-Performance_analisis]# artillery quick --count 50 -n 20 "http://localhost:8080/info" > result_clg-on.txt
```

```
[root@DESKTOP-0PD4IF7 LOGGERS-GZIP-Performance_analisis]# artillery quick --count 50 -n 20 "http://localhost:8080/info" > result_clg-off.txt
```

Resultados del análisis:

result_clg-on.txt :
Phase started: unnamed (index: 0, duration: 1s) 16:23:55(-0600)

Phase completed: unnamed (index: 0, duration: 1s) 16:23:56(-0600)

Metrics for period to: 16:24:00(-0600) (width: 2.596s)
---

http.codes.200: 1000
http.request_rate: 389/sec
http.requests: 1000
http.response_time:
  min: 2
  max: 282
  median: 83.9
  p95: 138.4
  p99: 228.2
http.responses: 1000
vusers.completed: 50
vusers.created: 50
vusers.created_by_name.0: 50
vusers.failed: 0
vusers.session_length:
  min: 1409.9
  max: 1906.4
  median: 1755
  p95: 1863.5
  p99: 1901.1

All VUs finished. Total time: 3 seconds

Summary report @ 16:23:58(-0600)
---

http.codes.200: 1000
http.request_rate: 389/sec
http.requests: 1000
http.response_time:
  min: 2
  max: 282
  median: 83.9
  p95: 138.4
  p99: 228.2
http.responses: 1000
vusers.completed: 50
vusers.created: 50
vusers.created_by_name.0: 50
vusers.failed: 0
vusers.session_length:
  min: 1409.9
  max: 1906.4
  median: 1755
  p95: 1863.5
  p99: 1901.1

result_clg-off.txt :
Phase started: unnamed (index: 0, duration: 1s) 16:39:59(-0600)

Phase completed: unnamed (index: 0, duration: 1s) 16:40:00(-0600)

Metrics for period to: 16:40:00(-0600) (width: 0.81s)
---

http.codes.200: 239
http.request_rate: 280/sec
http.requests: 280
http.response_time:
  min: 6
  max: 201
  median: 54.1
  p95: 141.2
  p99: 169

http.responses: 239
vusers.created: 41
vusers.created_by_name.0: 41

Metrics for period to: 16:40:10(-0600) (width: 1.702s)
---

http.codes.200: 761
http.request_rate: 424/sec
http.requests: 720
http.response_time:
  min: 2
  max: 289
  median: 80.6
  p95: 133
  p99: 219.2
http.responses: 761
vusers.completed: 50
vusers.created: 9
vusers.created_by_name.0: 9
vusers.failed: 0
vusers.session_length:
  min: 1116.9
  max: 1815.7
  median: 1620
  p95: 1790.4
  p99: 1790.4

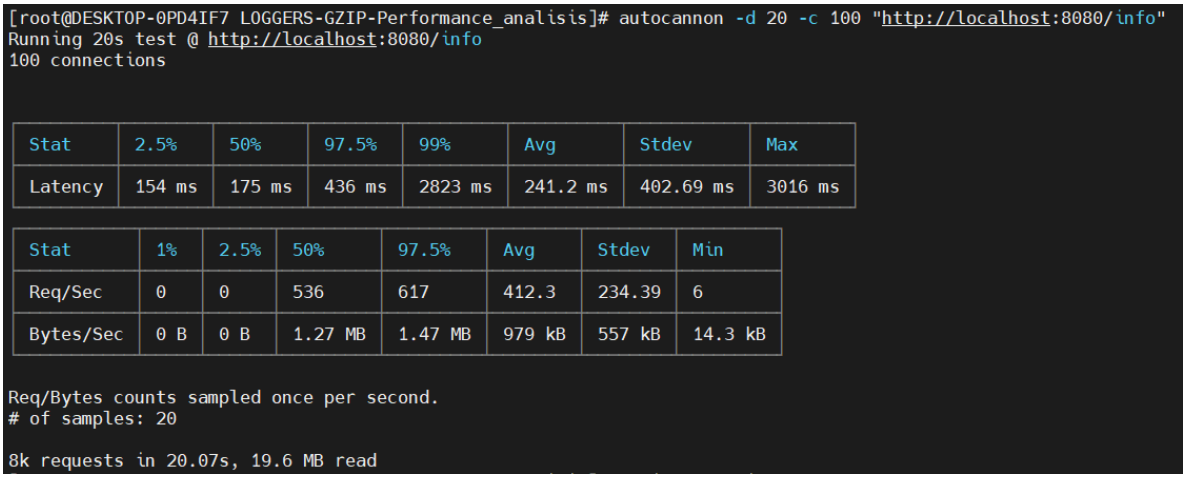
All VUs finished. Total time: 3 seconds

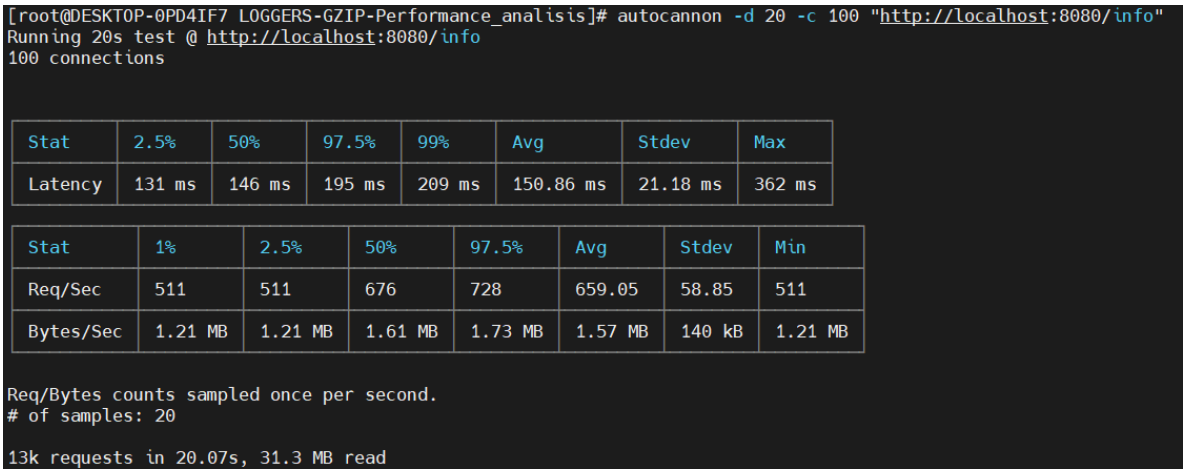
Summary report @ 16:40:02(-0600)
---

http.codes.200: 1000
http.request_rate: 352/sec
http.requests: 1000
http.response_time:
  min: 2
  max: 289
  median: 77.5

p95: 141.2
p99: 210.6
http.responses: 1000
vusers.completed: 50
vusers.created: 50
vusers.created_by_name.0: 50
vusers.failed: 0
vusers.session_length:
min: 1116.9
max: 1815.7
median: 1620
p95: 1790.4
p99: 1790.4

Autocannon:



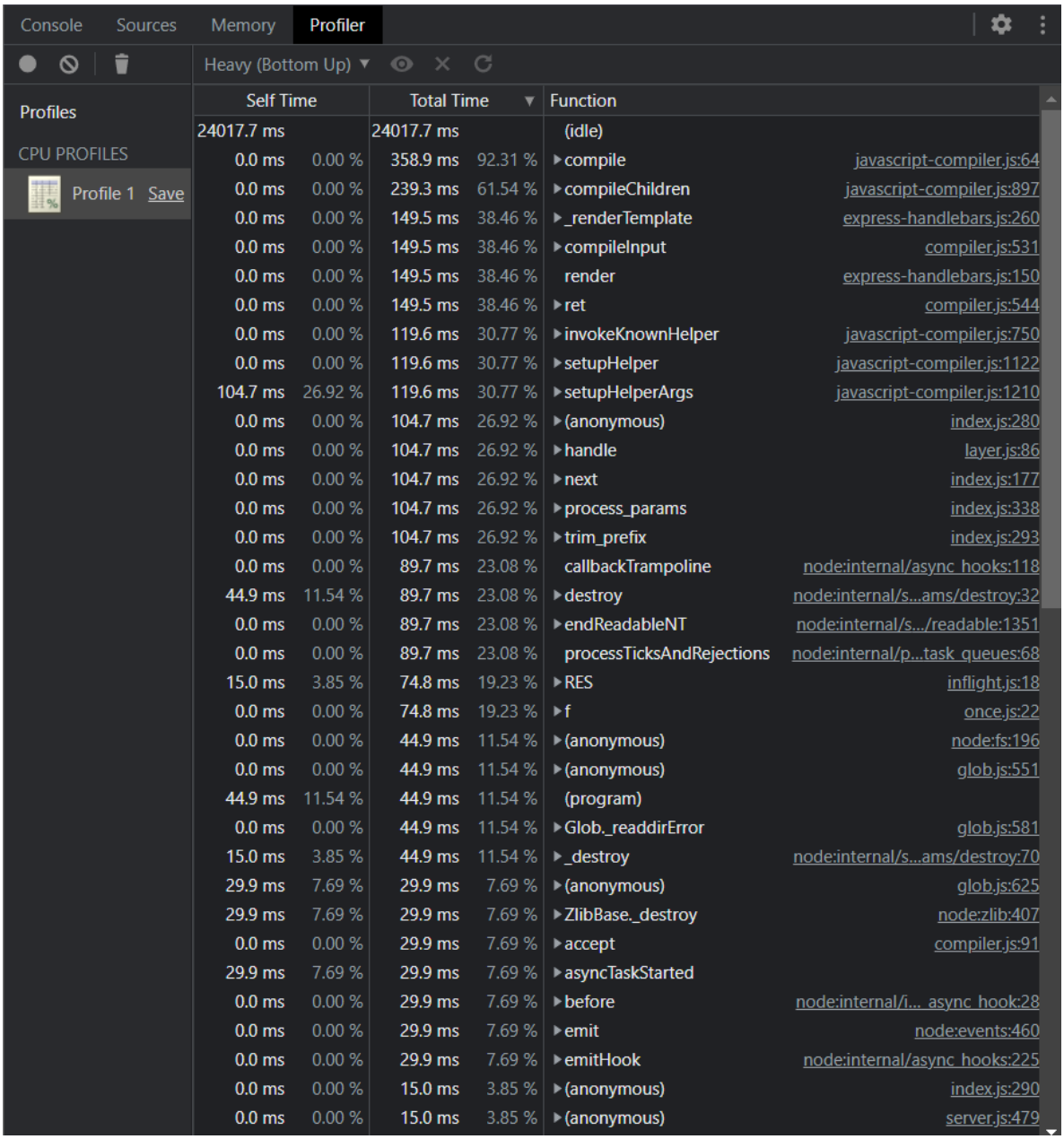


PERFILAMIENTO DEL SERVIDOR CON NODE.JS --INSPECT

Comandos del análisis:

```
[root@DESKTOP-0PD4IF7 LOGGERS-GZIP-Performance_analisis]# node --inspect main.js
Debugger listening on ws://127.0.0.1:9229/1fd6bf63-8d96-43a7-b379-fc38889efce7
For help, see: https://nodejs.org/en/docs/inspector
{"level":"info","message":"PID worker: 5539 - Server mode: FORK, listening at: http://localhost:8080"}
Debugger attached.
```

Resultados del análisis:



Console | Sources | Memory | Profiler

Heavy (Bottom Up)

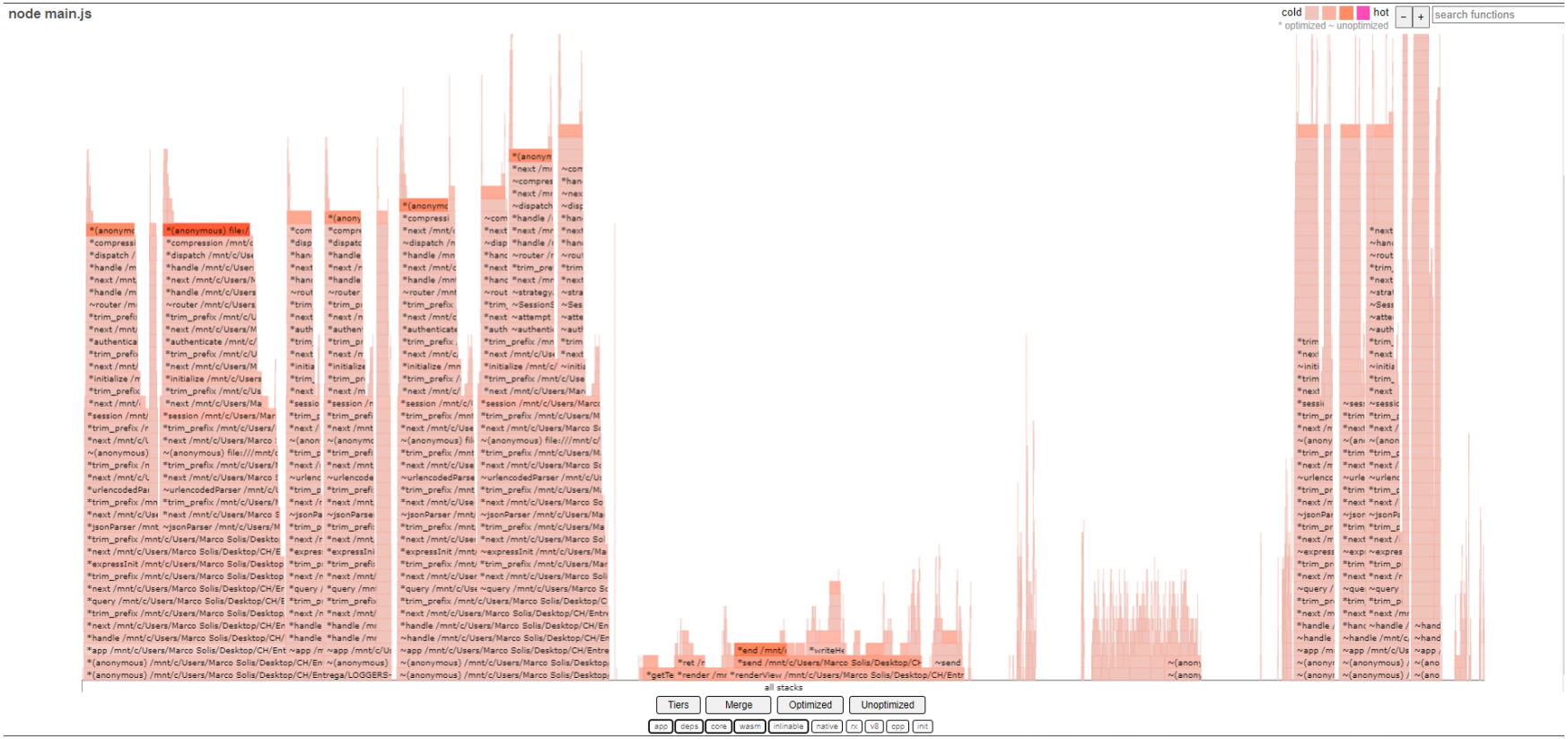
Profiles

CPU PROFILES

Profile 1 Save

Self Time		Total Time		Function	
24017.7 ms		24017.7 ms		(idle)	
0.0 ms	0.00 %	358.9 ms	92.31 %	compile	javascript-compiler.js:64
0.0 ms	0.00 %	239.3 ms	61.54 %	compileChildren	javascript-compiler.js:897
0.0 ms	0.00 %	149.5 ms	38.46 %	_renderTemplate	express-handlebars.js:260
0.0 ms	0.00 %	149.5 ms	38.46 %	compileInput	compiler.js:531
0.0 ms	0.00 %	149.5 ms	38.46 %	render	express-handlebars.js:150
0.0 ms	0.00 %	149.5 ms	38.46 %	ret	compiler.js:544
0.0 ms	0.00 %	119.6 ms	30.77 %	invokeKnownHelper	javascript-compiler.js:750
0.0 ms	0.00 %	119.6 ms	30.77 %	setupHelper	javascript-compiler.js:1122
104.7 ms	26.92 %	119.6 ms	30.77 %	setupHelperArgs	javascript-compiler.js:1210
0.0 ms	0.00 %	104.7 ms	26.92 %	(anonymous)	index.js:280
0.0 ms	0.00 %	104.7 ms	26.92 %	handle	layer.js:86
0.0 ms	0.00 %	104.7 ms	26.92 %	next	index.js:177
0.0 ms	0.00 %	104.7 ms	26.92 %	process_params	index.js:338
0.0 ms	0.00 %	104.7 ms	26.92 %	trim_prefix	index.js:293
0.0 ms	0.00 %	89.7 ms	23.08 %	callbackTrampoline	node:internal/async_hooks:118
44.9 ms	11.54 %	89.7 ms	23.08 %	destroy	node:internal/s...ams/destroy:32
0.0 ms	0.00 %	89.7 ms	23.08 %	endReadableNT	node:internal/s.../readable:1351
0.0 ms	0.00 %	89.7 ms	23.08 %	processTicksAndRejections	node:internal/p...task_queues:68
15.0 ms	3.85 %	74.8 ms	19.23 %	RES	inflight.js:18
0.0 ms	0.00 %	74.8 ms	19.23 %	f	once.js:22
0.0 ms	0.00 %	44.9 ms	11.54 %	(anonymous)	node:fs:196
0.0 ms	0.00 %	44.9 ms	11.54 %	(anonymous)	glob.js:551
44.9 ms	11.54 %	44.9 ms	11.54 %	(program)	
0.0 ms	0.00 %	44.9 ms	11.54 %	Glob._readdirError	glob.js:581
15.0 ms	3.85 %	44.9 ms	11.54 %	_destroy	node:internal/s...ams/destroy:70
29.9 ms	7.69 %	29.9 ms	7.69 %	(anonymous)	glob.js:625
29.9 ms	7.69 %	29.9 ms	7.69 %	ZlibBase._destroy	node:zlib:407
0.0 ms	0.00 %	29.9 ms	7.69 %	accept	compiler.js:91
29.9 ms	7.69 %	29.9 ms	7.69 %	asyncTaskStarted	
0.0 ms	0.00 %	29.9 ms	7.69 %	before	node:internal/i... async_hook:28
0.0 ms	0.00 %	29.9 ms	7.69 %	emit	node:events:460
0.0 ms	0.00 %	29.9 ms	7.69 %	emitHook	node:internal/async_hooks:225
0.0 ms	0.00 %	15.0 ms	3.85 %	(anonymous)	index.js:290
0.0 ms	0.00 %	15.0 ms	3.85 %	(anonymous)	server.js:479

EL DIAGRAMA DE FLAMA CON OX



CONCLUSIÓN

Las pruebas de performance nos brindan herramientas útiles para conocer el tiempo que tarda en ejecutarse una aplicación. Conocer esto nos dará una perspectiva más amplia de la eficacia de la ejecución de la misma y los puntos en los que refactorizar sería una opción útil para su desempeño.