Brendan Gregg's Blog   home

node.js Flame Graphs on Linux

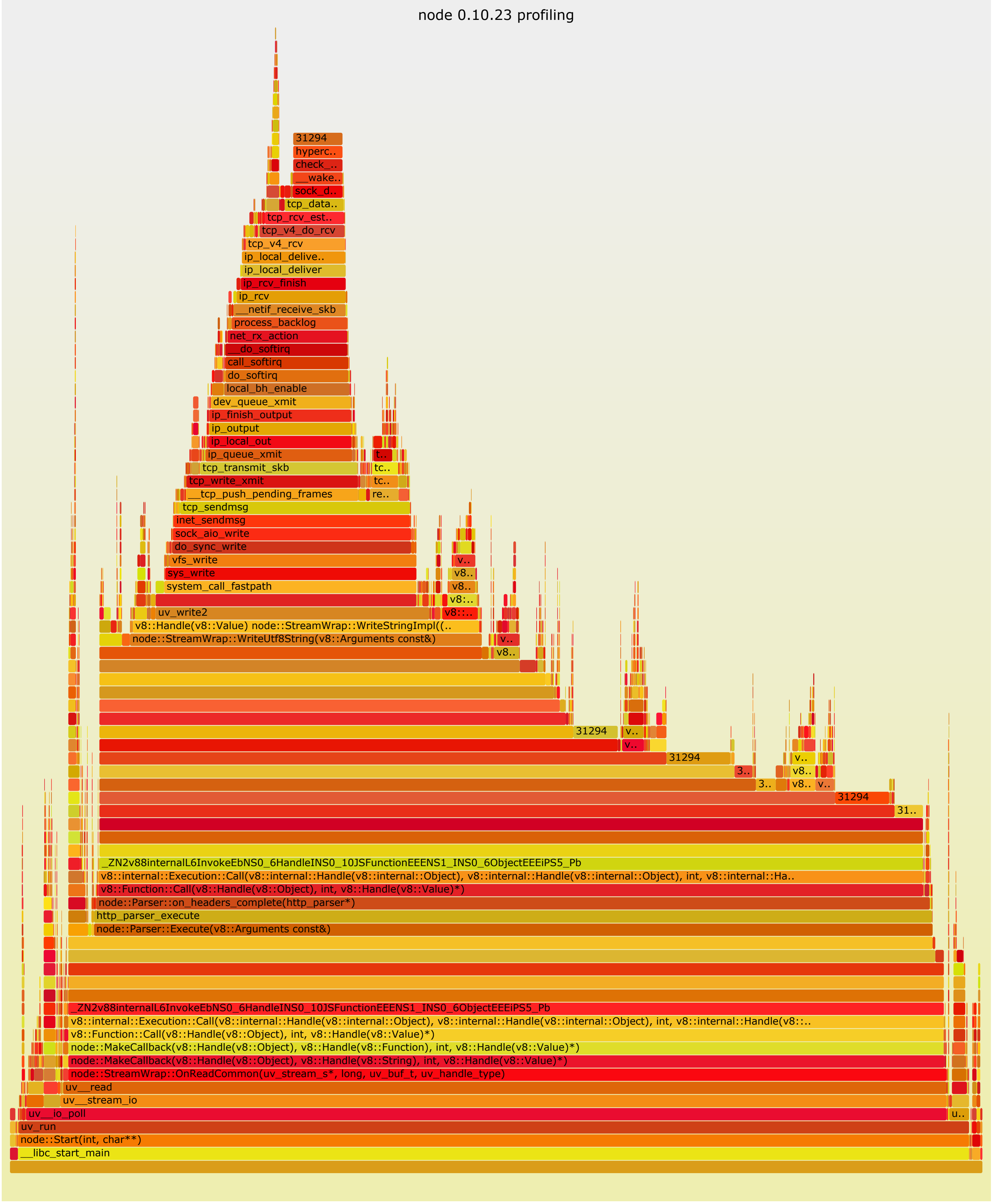
17 Sep 2014

CPU flame graphs are a useful visualization application stack traces, allowing you to quickly identify and quantify what to tune to improve performance. For Node.js they have solved countless problems on systems which have DTrace for sampling stack traces. But what about Linux?

At Netflix we have node.js in production at scale, on Linux instances in AWS EC2, and we create flame graphs using Linux perf_events and v8's --perf_basic_prof option (also works as --perf-basic-prof). In this quick blog post, I'll share how it works and how you can do it, and what needs to be fixed to improve it further.

1. The problem

Using perf_events to profile CPU usage on node.js 0.10.23:



It's interactive: mouse-over elements for details, and click the SVG to zoom. The CPU flame graphs page explains how to interpret these, and this was created using the instructions in the Linux perf section.

This flame graph is partially-useful, as I can see system and v8 library symbols. However, it is missing JavaScript symbols (the blank rectangles), since v8, like the JVM, compiles and places symbols just in time (JIT).

2. Linux perf_events JIT support

In 2009, Linux perf_events added JIT symbol support, so that symbols from language virtual machines like the JVM could be inspected. It works in the following amazingly simple way:

1. Your JIT application must be modified to create a /tmp/perf-*PID*.map file, which is a simple text database containing symbol addresses (in hex), sizes, and symbol names.
2. That's it.

perf already looks for the /tmp/perf-*PID*.map file, and if it finds it, it uses it for symbol translations. So only v8 needed to be modified.

3. v8 --perf-basic-prof support

In November 2013, v8 added perf_events support, enabled using the --perf-basic-prof option. This made it into node v0.11.13. It works like this:

```
# ~/node-v0.11.13-linux-x64/bin/node --perf-basic-prof hello.js &
[1] 31441
# ls -l /tmp/perf-31441.map
-rw-r--r-- 1 root root 81920 Sep 17 20:41 /tmp/perf-31441.map
# tail /tmp/perf-31441.map
14cec4db98a0 f Stub:BinaryOpICWithAllocationSiteStub(ADD_CreateAllocationMementos:String*Generic-
14cec4db9920 f Stub:BinaryOpICWithAllocationSiteStub(ADD_CreateAllocationMementos:String*String->
14cec4db99a0 f Stub:BinaryOpICWithAllocationSiteStub(ADD_CreateAllocationMementos:String*Smi->Str
14cec4db9a20 22c LazyCompile:~nextTick node.js:389
14cec4db9cc0 156 Stub:KeyedLoadElementStub
14cec4db9e80 22 KeyedLoadIC:
14cec4db9f20 22 KeyedLoadIC:
14cec4db9fc0 56 Stub:DoubleToIStub
14cec4dba080 10c Stub:KeyedStoreElementStub
```

This text file is what perf_events reads.

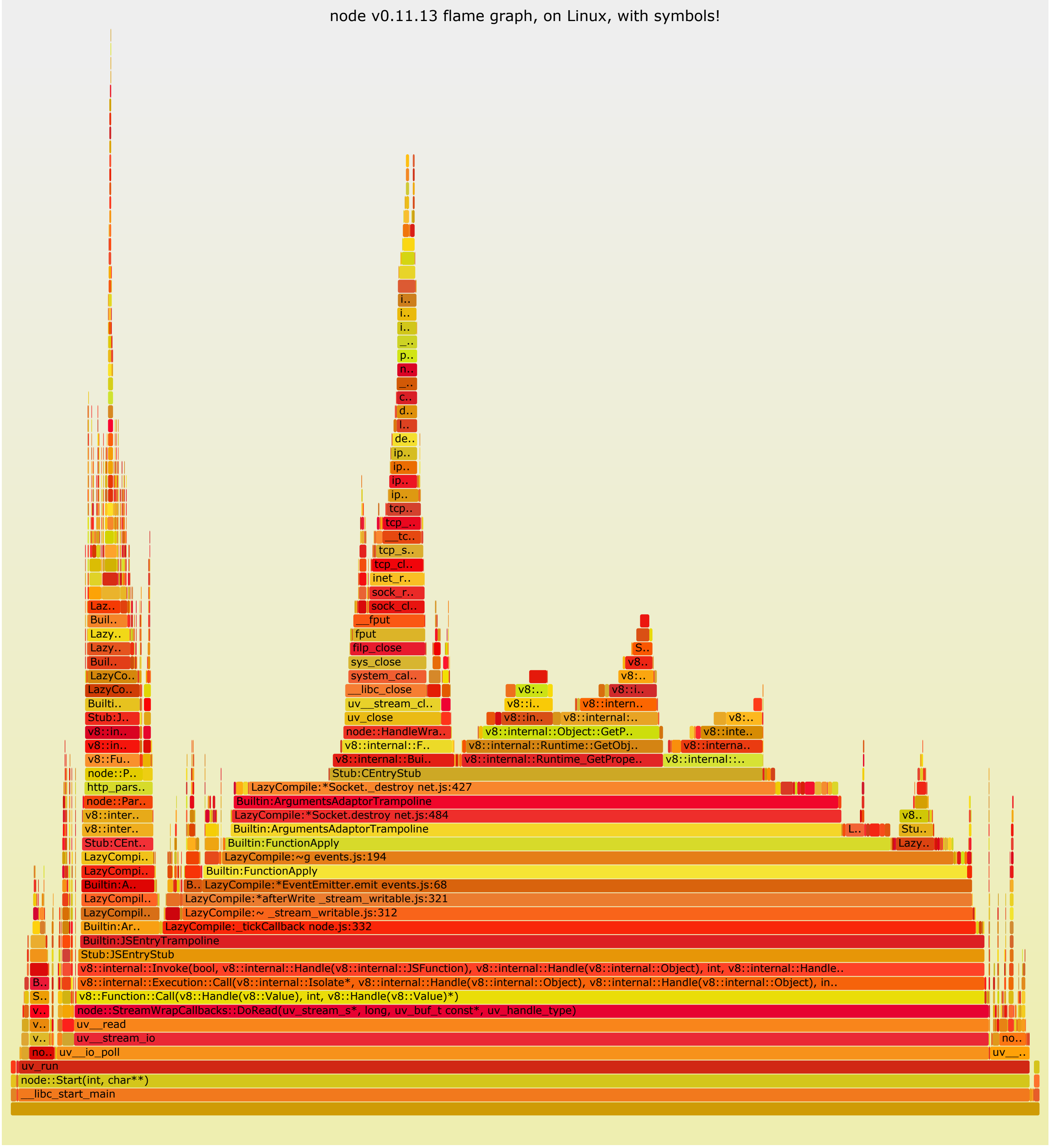
4. node.js Flame Graphs

Now that we have node 0.11.13+ running with --perf-basic-prof, we can create a flame graph using:

```
$ sudo bash
# perf record -F 99 -p `pgrep -n node` -g -- sleep 30
# perf script > out.nodestacks01
# git clone --depth 1 http://github.com/brendangregg/FlameGraph
# cd FlameGraph
# ./stackcollapse-perf.pl < ../out.nodestacks01 | ./flamegraph.pl > ../out.nodestacks01.svg
```

You can also use stackvis, by Dave Pacheco, a node.js implementation which has extra features.

Here's an example result:



Note the JavaScript symbols are now readable. Click the SVG to zoom in. This actual flame graph isn't very interesting, as I'm just testing a dummy app to test out --perf-basic-prof.

Thanks to Trevor Norris for first posting the instructions for doing this in a short gist, which you may find useful to read. He also provides a script to facilitate this.

WARNING: map file growth

We can currently only use --perf-basic-prof for short periods (hours), due to bug 3453: the perf.map file can grow endlessly, eating Gbytes in a few days. It looks like symbols are moving location (they are supposed to stay put with --perf-basic-prof), causing the map file to keep growing.

UPDATE (2016): A new option, --perf_basic_prof_only_functions (or --perf-basic-prof-only-functions) was introduced to address this bug by only logging interesting types of symbols, cutting down on map file growth. If map file growth is a problem for you, try out this option instead.

More

We're doing more at Netflix with node.js analysis. Stay tuned, and also see the Netflix Tech Blog.

Comments for this thread are now closed ×

8 Comments **Brendan Gregg's Blog** 1 **Login**

Recommend 1 Tweet Share Sort by Best

atul pahuja • 3 years ago

Hi, I am generating flame graph for my loopback application, I have followed below steps :

1. sudo bash
2. ~/node-v0.11.13-linux-x64/bin/node --perf-basic-prof hello.js &
3. perf record -F 99 -p `pgrep -n node` -g -- sleep 30
4. perf script > out.nodestacks01

After step 4 I am getting out.nodestacks01 with 0kb. due to this I am getting below error while generating the flame graph on step 5.

5. ./stackcollapse-perf.pl < ../out.nodestacks01 | ./flamegraph.pl > ../out.nodestacks01.svg .
bash: ../out.nodestacks01: No such file or directory

ERROR: No stack counts found

Please help me out

1 • Share ›

abcde ➜ atul pahuja • 3 years ago

I had that same problem. It was because the system was idle while you generated the nodestacks01 file. Try processing something or trigger requests on app while profiling.

• Share ›

Alejandro Oviedo • 2 years ago

Hey Brendan, apparently the bug 3453 for map file growth is fixed. Are you still having issues with that?

Thanks,
Alejandro

• Share ›

Dory Zidon • 3 years ago

I tried to run this exact process, but all I got was a very simple flame graph with very few levels, most of which weren't even node.js functions. What might I be missing? http://stackoverflow.com/qu...

• Share ›

Tushar Mathur • 4 years ago

Doesn't work - ➜ ~ perf script
perf: 'script' is not a perf-command. See 'perf --help'.

• Share ›

Techy Ashutosh • 4 years ago

I was trying to generate so many times but not able to get exact way!
Can you help me, How to generate a Node.js flame graph on CentOS?

• Share ›

Tony Brown • 4 years ago

This is really useful, thanks

• Share ›

brendangregg Mod • 4 years ago

Apart from perf_basic_prof, a new and different way to extract symbol information from node is being developed: https://gist.github.com/rf/... , which can also be used for making flame graphs. The potential advantages of this way is that it works on older node.js versions, and might avoid the growth bug. I haven't used it yet.

• Share ›

You can comment here, but I can't guarantee your comment will remain here forever: I might switch comment systems at some point (eg, if disqus add advertisements).