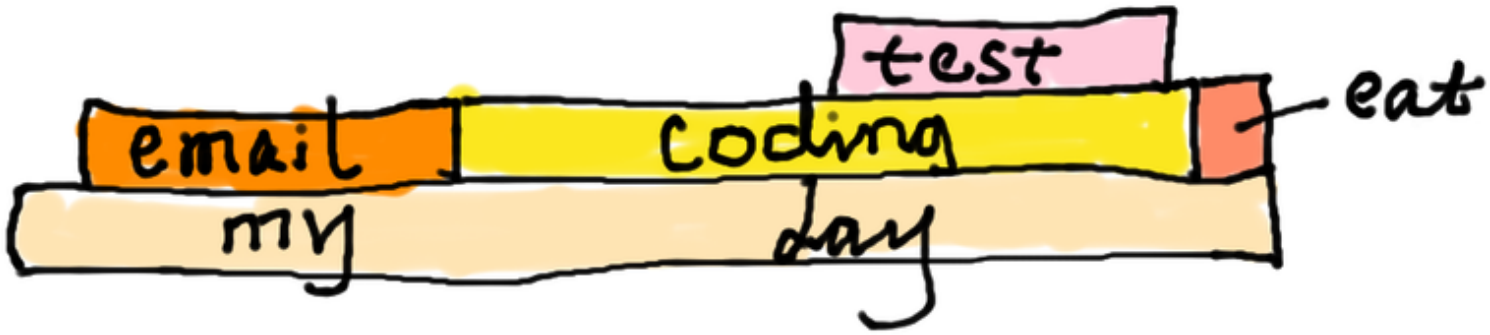
The Way of *Optimizing* and *Troubleshooting* Our Lua Waf

☺ *agentzh@gmail.com* ☺

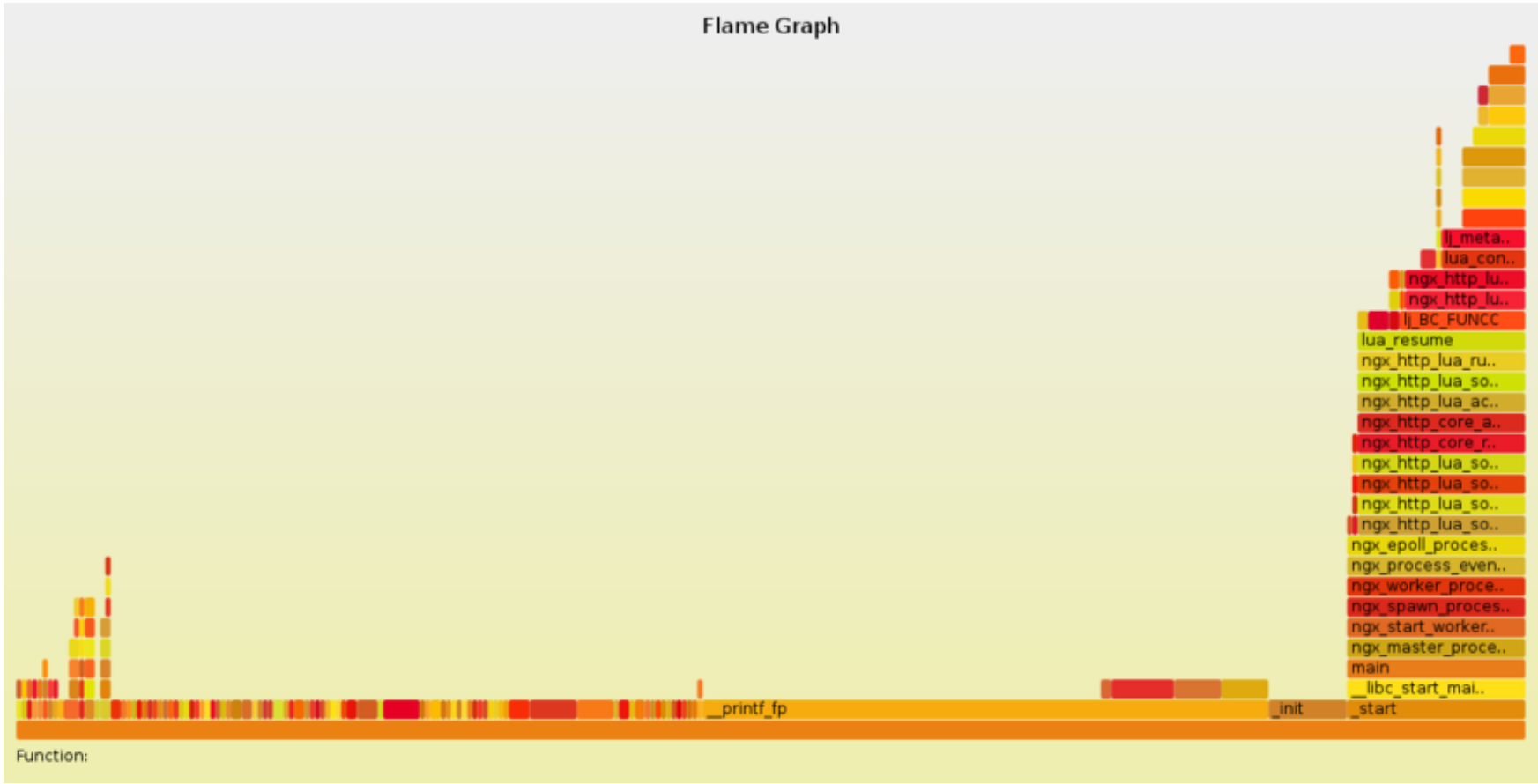
Yichun Zhang (agentzh)

2013.04.19

♡ Dane now *loves* to
open a conversation with me
via a Flame Graph.

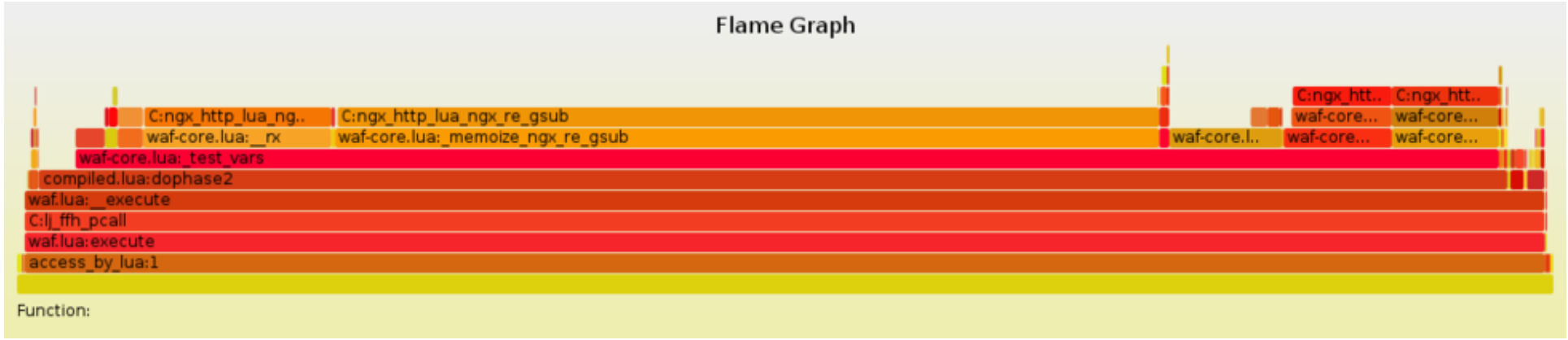

test
email
coding
eat
my
day

♡ One day,
Dane walks to me with a *weird*
C-level Flame Graph
for our online WAF...


Flame Graph
lj_meta..
lua_con..
ngx_http_lu..
ngx_http_lu..
lj_BC_FUNCC
lua_resume
ngx_http_lua_ru..
ngx_http_lua_so..
ngx_http_lua_ac..
ngx_http_core_a..
ngx_http_core_r..
ngx_http_lua_so..
ngx_http_lua_so..
ngx_http_lua_so..
ngx_http_lua_so..
ngx_epoll_proces..
ngx_process_even..
ngx_worker_proce..
ngx_spawn_proces..
ngx_start_worker..
ngx_master_proce..
main
__libc_start_mai..
__printf_fp
_init
_start
Function:

♡ About 80% of the C-level backtraces are *bad* in the graph.

♡ Fortunately we have a good *Lua-land* Flame Graph for the same process.


Flame Graph
C:ngx_http_lua_ng..
C:ngx_http_lua_ngx_re_gsub
C:ngx_htt..
C:ngx_htt..
waf-core.lua:__rx
waf-core.lua:_memoize_ngx_re_gsub
waf-core.l..
waf-core...
waf-core...
waf-core...
waf-core...
waf-core.lua:_test_vars
compiled.lua:dophase2
waf.lua:__execute
C:lj_ffh_pcall
waf.lua:execute
access_by_lua:1
Function:

♡ We are using the PCRE Just-In-Time compiler.

JITted code does *not* have debug information required by a proper C-level backtrace.

♡ We now know *PCRE* is busy!

We now know our *regexes* are slow!

♡ ngx-pcre-stats is a simple tool based on systemtap.

```
$ ./ngx-pcre-stats -p 24528 --exec-time-dist
Tracing 24528 (/path/to/nginx/sbin/nginx)...
Hit Ctrl-C to end.
^C
Logarithmic histogram for pcre_exec running time distribution (us):
value |-------------------------------------------------- count
    0 |                                                      0
    1 |                                                      0
    2 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@                    981
    4 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  1479
    8 |                                                     16
   16 |                                                     18
   32 |                                                      1
```

```
$ ./ngx-pcre-stats -p 24528 --total-time-top --luajit20
Tracing 24528 (/path/to/nginx/sbin/nginx)...
Hit Ctrl-C to end.
^C
Top N regexes with longest total running time:
1. pattern /WEB_ATTACK/: 15103us (total data size: 82184)
2. pattern / __cf__\d+/: 11143us (total data size: 25916)
3. pattern /[^\x01-\xff]/: 10233us (total data size: 102825)
4. pattern /\b(?:coalesce\b|root\@)/: 7017us (total data size: 78230)
5. pattern /(Content-Length|Transfer-
Encoding)/: 6766us (total data size: 17871)
...
```

```
$ ./ngx-pcre-stats -p 24528 --worst-time-top --luajit20
Tracing 24528 (/path/to/nginx/sbin/nginx)...
Hit Ctrl-C to end.
^C
Top N regexes with worst running time:
1. pattern /\.cookie\b.*?\;\W*?domain\W*?\=/: 98us (data size: 36)
2. pattern /(Content-Length|Transfer-Encoding)/: 89us (data size: 14)
3. pattern / __cf__\d+/: 63us (data size: 8)
4. pattern /[^\x01-\xff]/: 53us (data size: 13)
5. pattern /\b(background|dynsrc|href|lowsrc|src)\b\W*?
=/: 53us (data size: 5147)
6. pattern /(?i:<embed[ /+\t].*?SRC.*?=)/: 47us (data size: 304)
7. pattern /(fromcharcode|alert|eval)\s*\(/: 45us (data size: 24)
8. pattern /\bselect\b.*?\bto_number\b/: 40us (data size: 5147)
```

John Graham-Cumming: This is very *helpful*.

John Graham-Cumming: And today I used it to kill off four of them.

♡ Another day, Dane comes to me with another Flame Graph and tells me that some Nginx worker processes are *hot spinning*!


Flame Graph
index2.
luaL_checklstring
memchr
str_find_aux
lj_cf_string_find
lj_BC_FUNCC
lj_BC_RET
lua_resume
ngx_http_lua_run_thread
ngx_http_lua_socket_tcp_resume
ngx_http_lua_access_handler
ngx_http_core_access_phase
ngx_http_core_run_phases
ngx_http_lua_socket_handle_success
ngx_http_lua_socket_tcp_read
ngx_http_lua_socket_read_handler
ngx_http_lua_socket_tcp_handler
ngx_event_process_posted
ngx_process_events_and_timers
ngx_worker_process_cycle
ngx_spawn_process
ngx_start_worker_processes
ngx_master_process_cycle
main
__libc_start_main
_start
Function:

♡ The Lua code seems to be spining around some *string.find()* calls.

♡ Let's find out
the *Lua*-land backtrace!

♡ After GDB *crashes* our processes for 6 times, I quickly write another tool named ngx-lua-bt based on systemtap...

```
$ ./ngx-lua-bt -p 7599 --luajit20
WARNING: Tracing 7599 (/path/to/nginx/sbin/nginx) for LuaJIT 2.0...
C:lj_cf_string_find
@/usr/local/nginx-waf/lua/waf-core.lua:201
@/usr/local/nginx-waf/lua/waf-core.lua:676
@/usr/local/nginx-waf/lua/waf-core.lua:1467
@/usr/local/nginx-waf/lua/waf-core.lua:1074
@/usr/local/nginx-waf/lua/rules/oldwaf/compiled.lua:371
@/usr/local/nginx-waf/lua/waf.lua:57
C:lj_ffh_pcall
@/usr/local/nginx-waf/lua/waf.lua:50
access_by_lua:1
```

♡ The problematic *infinite* loop in John Graham-Cumming's Lua code...

```
while start do
    local stop = string.find(v, "}", start, true)
    if stop then
      ...
    end
end
```

John Graham-Cumming: Well, that's the *perfect* bug report.

Even contains the fix!

☺ *Any questions*? ☺

The Way of *Optimizing* and *Troubleshooting* Our Lua Waf

☺*agentzh@gmail.com*☺

Yichun Zhang (agentzh)

2013.04.19