APPLISCALE

XProf Solves All Your Performance Issues

EUC 2017 - Stockholm (08.06.2017)

APPLISCALE

~ # whoami

Péter Gömöri

Software Engineer @ Appliscale

10 years with Erlang

Hire us! www.appliscale.io/contact
Join us! www.appliscale.io/career

Agenda

- Introducing XProf
  Features and loads of examples
- Battle stories
  Real-world use-cases
- Under the hood
  Architecture and overhead

What is profiling

"A form of **dynamic** program analysis that measures, for example, the space (memory) or time complexity of a program, the usage of particular instructions, or the **frequency** and **duration** of function calls. Most commonly, profiling information serves to aid program **optimization**." (Wikipedia)

What is XProf

An open-source production-safe real-time visual profiler for Erlang and Elixir

Single-function profiling

The team: Michal Niec, Pawel Pikula, Wojciech Gawronski, Péter Gömöri

https://github.com/Appliscale/xprof

What is XProf

Motivation

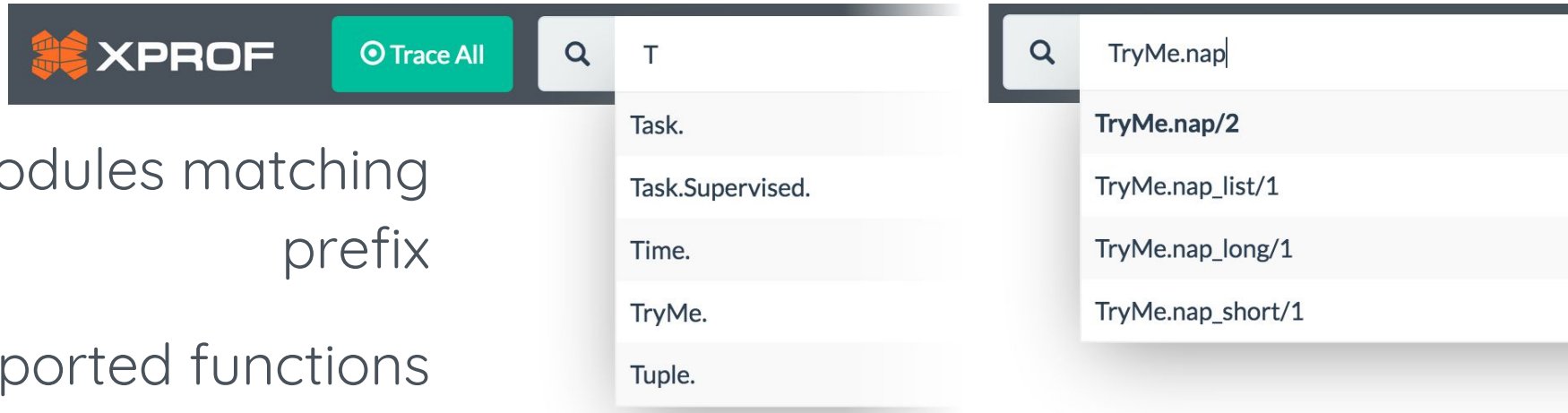
- ~~Instrumentation, metrics~~ ⇒ ad-hoc measure anything
- ~~All functions~~ ⇒ overhead, distortion
- ~~Call count and time~~ ⇒ more details, distribution
- More details, args+return
- Intuitive visual feedback
- Real-time results
- Extra: shared session

Compared to other tools

- General profilers (fprof, eflame)
- Production-safe tracing tools
  (redbug, recon, Tap, exrun)
- O&M software (Appsignal, Wombat)

Features

Function browser



Modules matching prefix

Exported functions

Private functions

Features

duration:

percentiles

mean

min/max

call count

show/hide

Features

Capture long calls

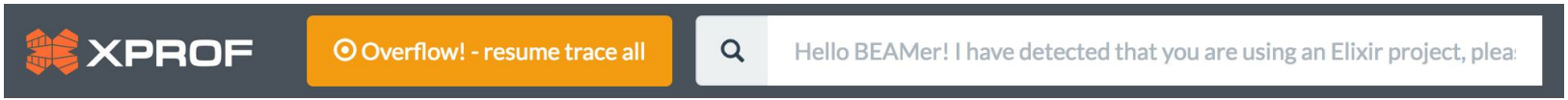
Slow calls tracing

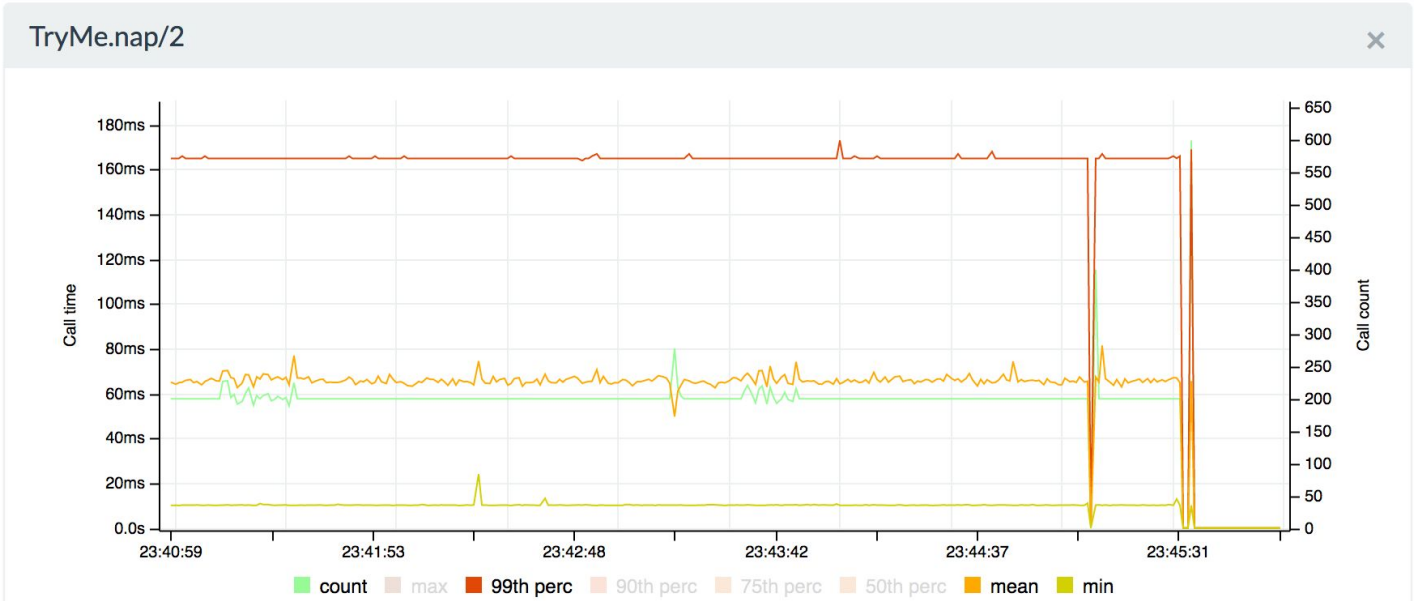
Treshold 120 ms Limit 3 calls Start

	Call time	Pid	Args	Response
›	132580 µs	#PID<0.29219.34>	[:long, 132]	:ok
›	133054 µs	#PID<0.29364.34>	[:long, 132]	:ok
›	132479 µs	#PID<0.29408.34>	[:long, 132]	:ok

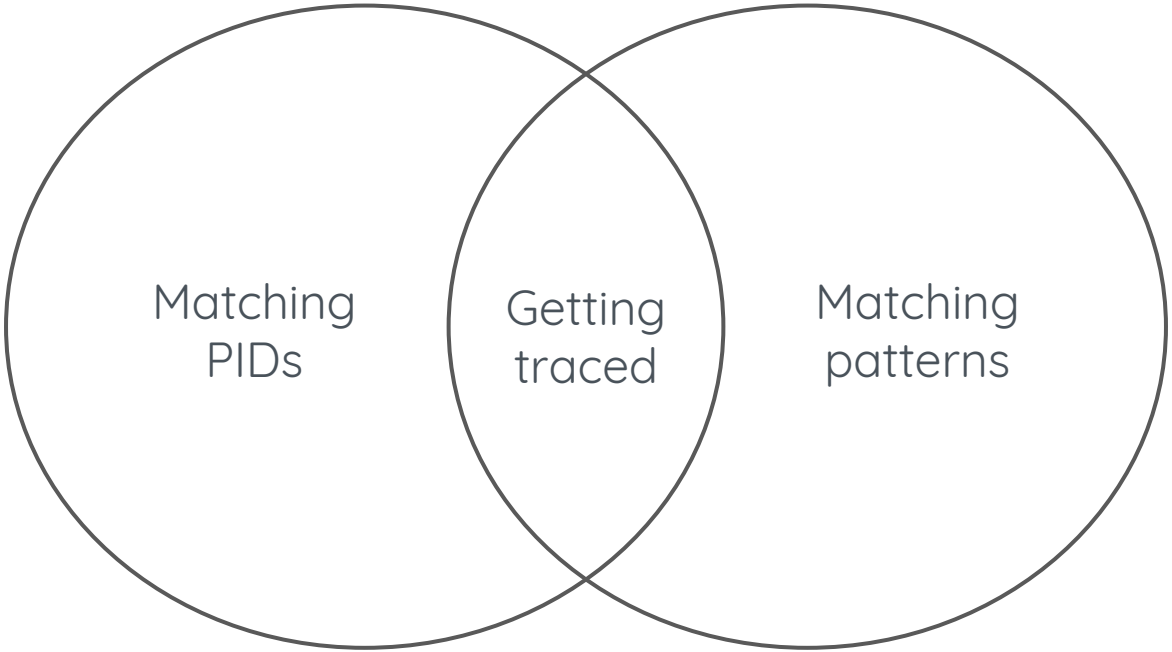
Features

Overload protection



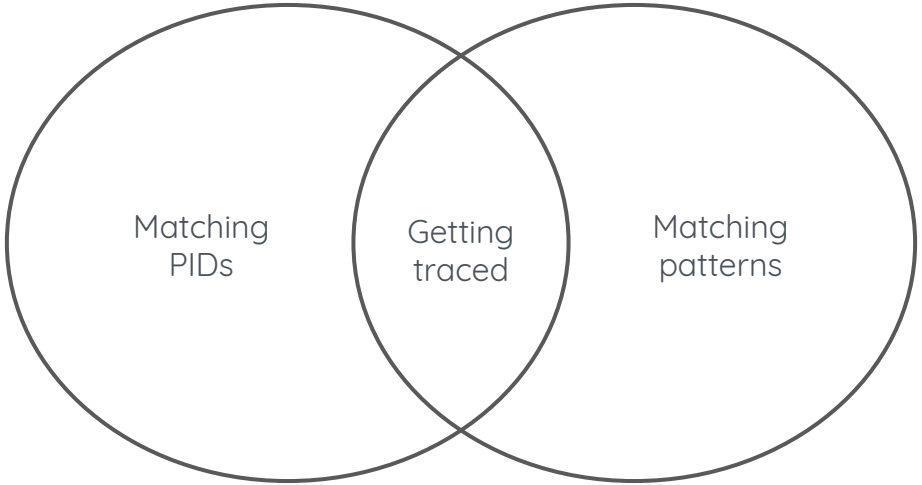


Tracing in Erlang



Tracing in Erlang

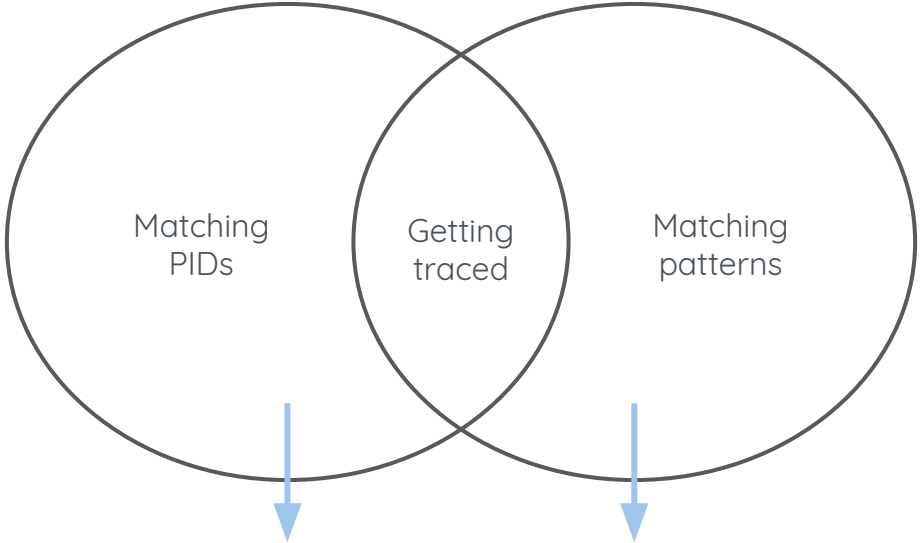
specific PID
all
existing
new



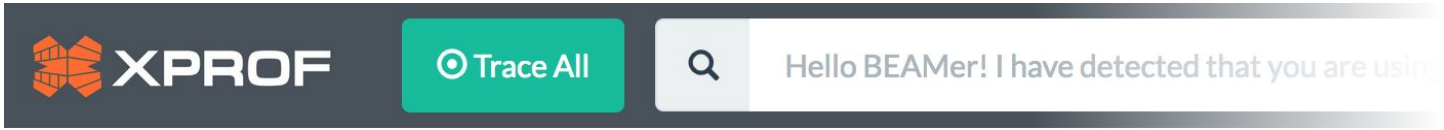
module function arity
module function all arities
module + all functions
match specifications

Tracing in Erlang

all
none



Module function arity
Match specifications



XProf-flavoured match specs

```
Module.function/arity
```

```
Module.function thing
```

- Inspired by redbug's Restricted Trace Patterns (RTP)
- Narrow down what to measure
- Full power of Erlang match-specs

XProf-flavoured match specs

Only arguments

```
ets:lookup(data, _)
```

```
Registry.lookup(MyApp.Registry, _)
```

XProf-flavoured match specs

Arguments and guards

```
gen_tcp:connect(_, Port, _, _)
    when Port =:= 80; Port =:= 443

Registry.dispatch(MyApp.Registry, topic, _)
    when topic in ["topic1", "topic2"]
```

XProf-flavoured match specs

Arguments and body

```
lists:subtract(_, _) -> message(caller())
```

XProf-flavoured match specs

Arguments, guard and body

```
ets:insert(_, Data) when element(1, Data) =:= data ->
    message(element(3, Data))

Enum.fetch(list, index) when is_list(list) ->
    message([length(list), index])
```

XProf-flavoured match specs

Multiple clauses

```
gen_tcp:connect(_, 80, _, _) -> true;
               (_, 443, _, _) -> true

Registry.dispatch(MyApp.Registry, "topic1", _) -> nil;
                 (MyApp.Registry, "topic2", _) -> nil
```

XProf-flavoured match specs

String prefix (query starts with)

Future `emysql:execute(_, "SELECT f1"++_)`

Now `emysql:execute(_, [$S,$E,$L,$E,$C,$T,$\s,$f,$1|_])`

XProf-flavoured match specs

Sampling (using trace control word)

```
xprof_web_handler:handle_req
  (_,_,_) when get_tcw() >= 9 -> set_tcw(0);
  (_,_,_) -> set_tcw(get_tcw() + 1), message(false)
```

XProf-flavoured match specs

...and a lot more

Reference:
http://erlang.org/doc/apps/erts/match_spec.html

Restrictions:
http://erlang.org/doc/man/ms_transform.html

Battle stories

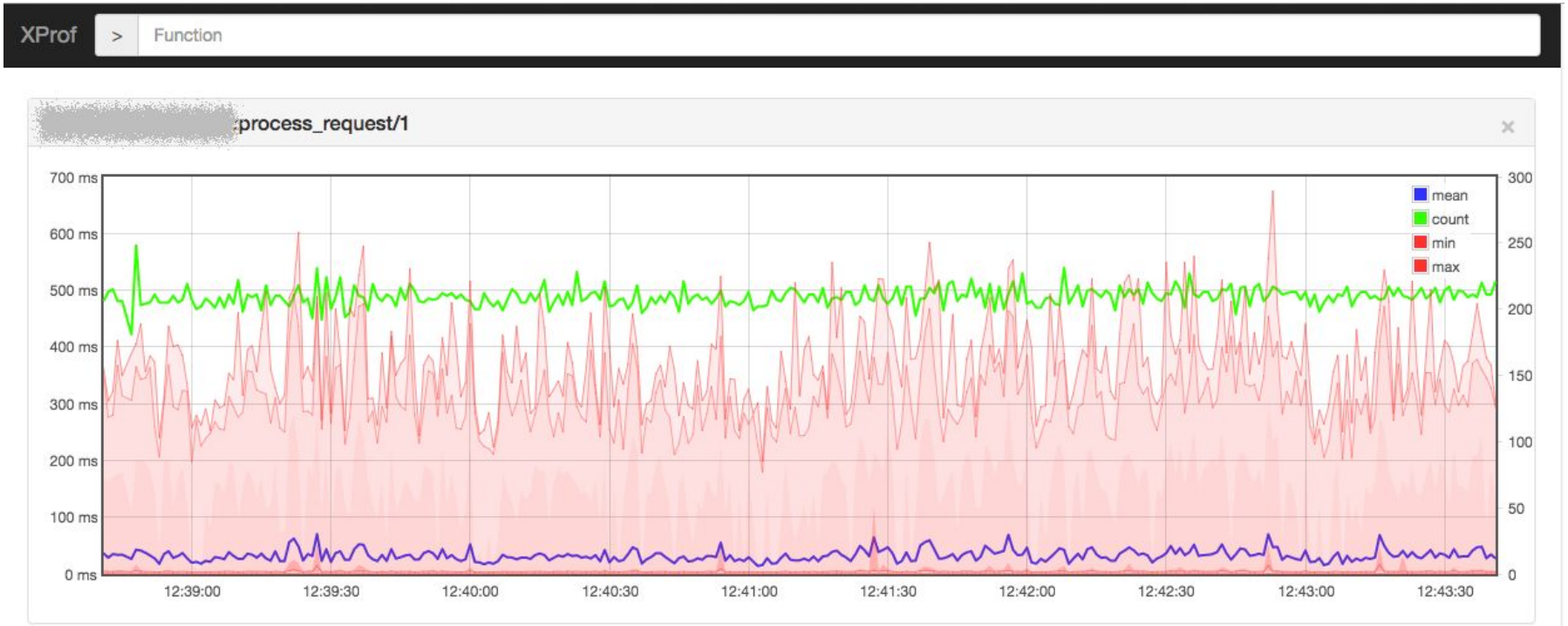
Fat requests

Increased response times/general slowness

Object ↔ business rule (many-to-many)

Preparation phase: fetch business rules associated with requested object

Fat requests



Fat requests

Objects with too many business rules (100s)

10s MB data fetched for these requests

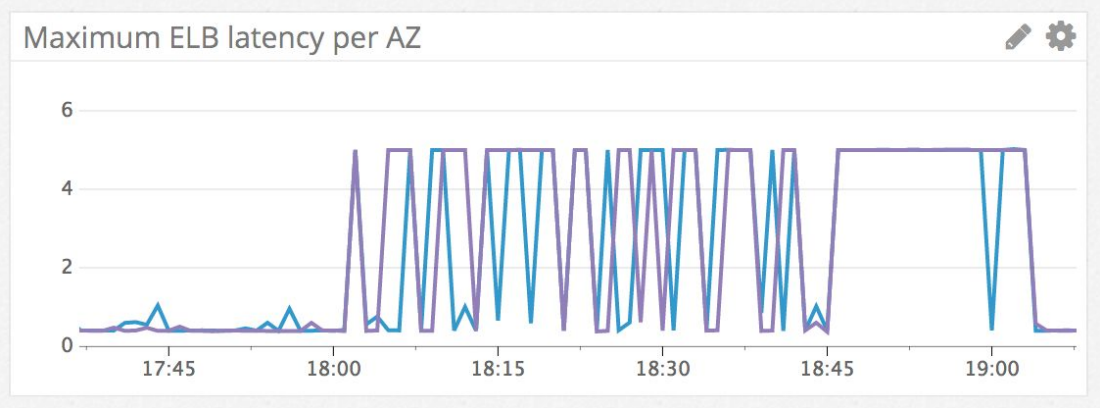
> 300ms to prepare

XProf showed these requests are frequent

MySQL woes 1

Max response time:
5 sec spikes

Every ~2h



MySQL woes 1

5 seconds

No free connection

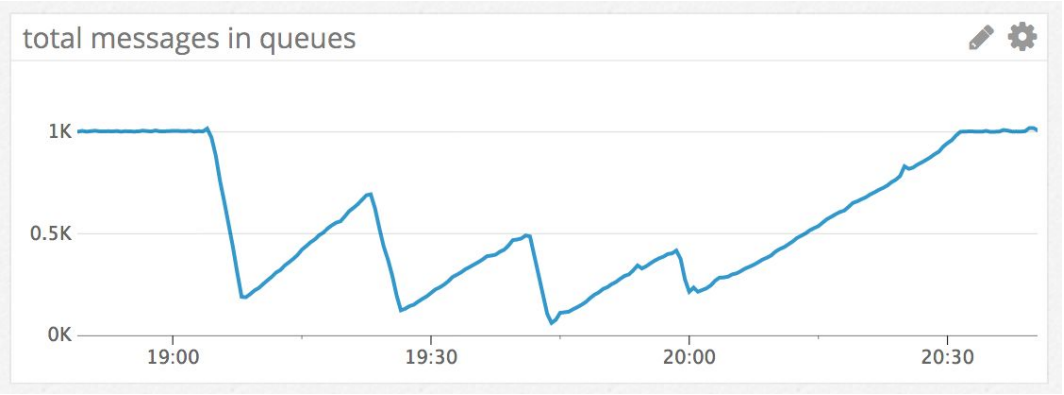
Increased pool size

MySQL woes 2

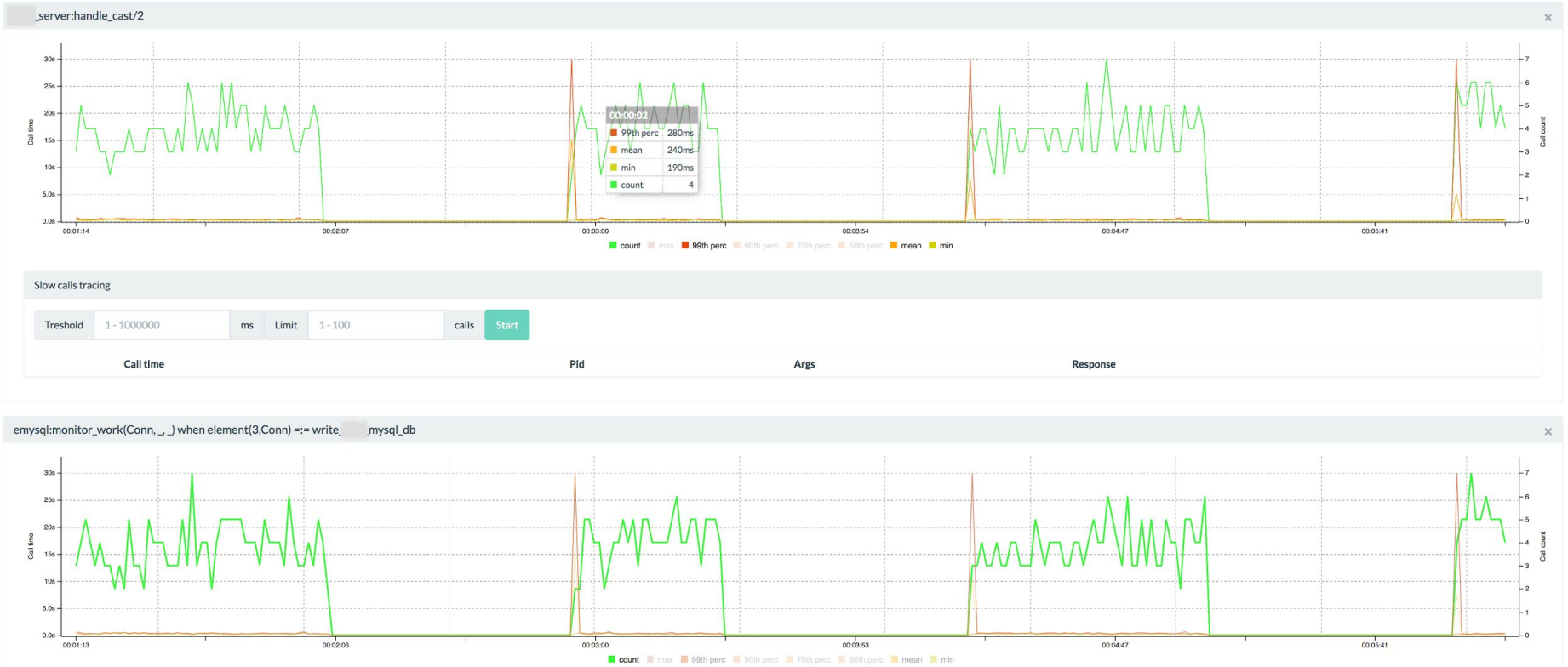
Message queue build-up

Bottleneck gen_server

Freeze



MySQL woes 2


_server:handle_cast/2
Call time
Call count
00:00:02
99th perc 280ms
mean 240ms
min 190ms
count 4
count
99th perc
mean
min
Slow calls tracing
Treshold
1 - 1000000
ms
Limit
1 - 100
calls
Start
Call time
Pid
Args
Response
emysql:monitor_work(Conn, _, _) when element(3,Conn) =:= write_mysql_db
Call time
Call count
count
99th perc
mean
min

APPLISCALE

MySQL woes 2

Slow calls tracing

Treshold 2000 ms Limit 3 calls Start

	Call time	Pid	Args	Response
>	50717657 µs	<0.2866.0>	[{emysql_connection,"#Port<0.4485>"	{error_packet,1,1205,<<"HY000">>, "Lock wait timeout exceeded; try restarting transaction"}
>	50681729 µs	<0.2866.0>	[{emysql_connection,"#Port<0.4485>"	{error_packet,1,1205,<<"HY000">>, "Lock wait timeout exceeded; try restarting transaction"}
>	51160348 µs	<0.2866.0>	[{emysql_connection,"#Port<0.4485>"	{error_packet,1,1205,<<"HY000">>, "Lock wait timeout exceeded; try restarting transaction"}

MySQL woes 2


emysql:monitor_work(Conn, _, _) when element(3,Conn) =:= write_ _mysql_db
Call time
30s
25s
20s
15s
10s
5.0s
0.0s
Call count
11
10
9
8
7
6
5
4
3
2
1
0
00:23:44
00:24:38
00:25:32
00:26:26
00:27:20
00:28:14
count
max
99th perc
90th perc
75th perc
50th perc
mean
min
Slow calls tracing
Treshold
2000
ms
Limit
3
calls
Stop
Call time
Pid
Args
Response
49048469 µs
<0.2866.0>
[{emysql_connection,"#Port<0.4485>",write_ _mysql_db,utf8,#Port<0.4485>, <<"5.6.29-log">>,25867,65535,8, {0,nil}, 837584,true,0,1495837584,#Ref<0.0.11069.193977>,false}, 900000,
{ok_packet,1,1,0,2,0,"(Rows matched: 1 Changed: 1 Warnings: 0"}

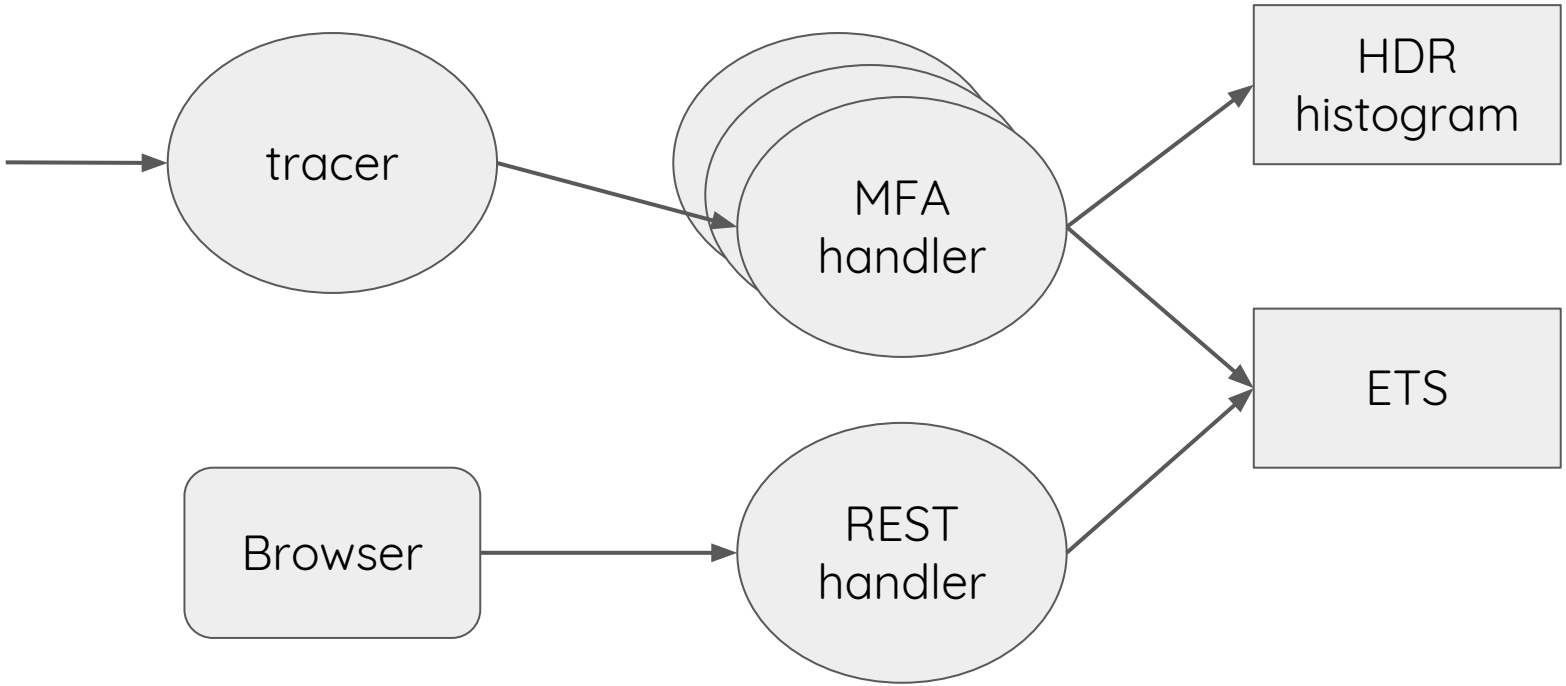
XProf workflow

- Monitoring
- XProf in action
- Code modification
- Profit

- Graph overview
  - Outer → inner functions
  - Compare 2 parallel functions
- Capture arguments
- Narrow down with match-specs

Under the hood

Architecture

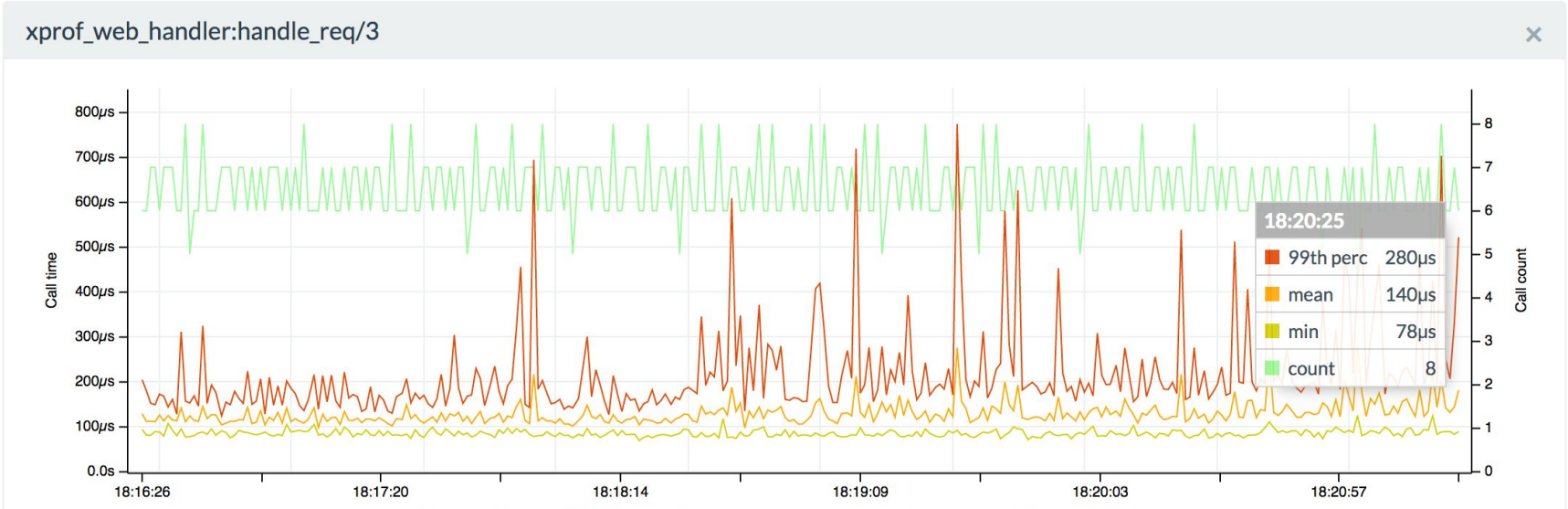


Overhead

- Profile XProf with XProf (GUI/web server impact)

Overhead

- Profile XProf with XProf (GUI/web server impact)



Overhead

- Profile XProf with XProf
- Profile XProf with fprofx

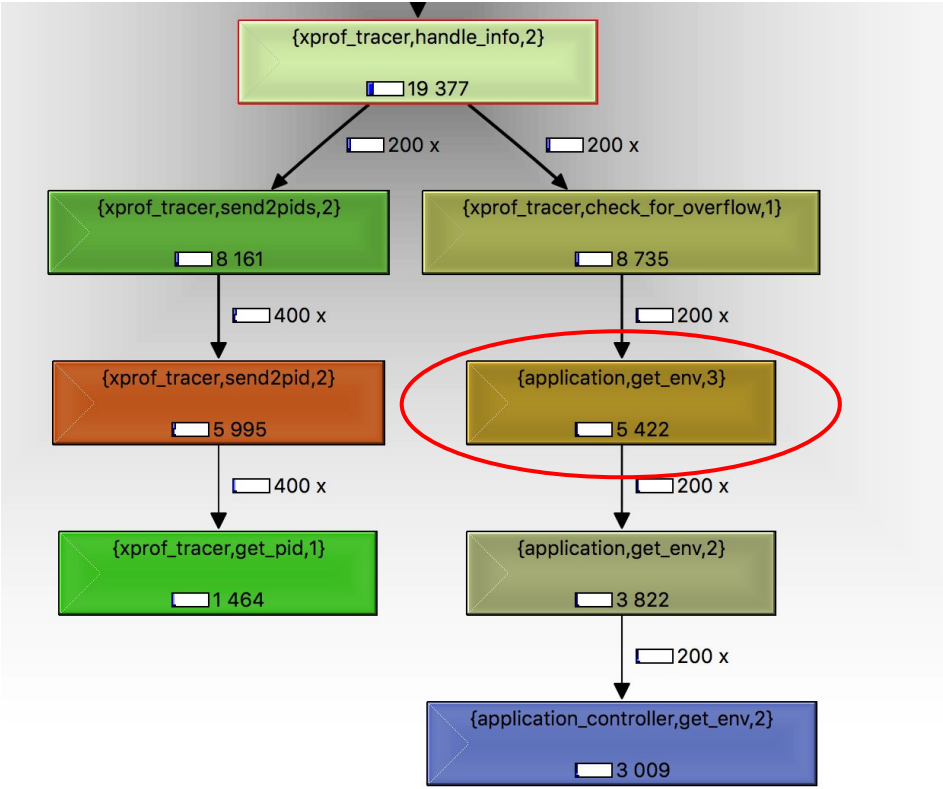
Overhead

- Profile XProf with XProf
- Profile XProf with fprofx

```
…                                                    CNT       ACC       OWN       SUS       GC
{  suspend,                                           17, 1929.107,   0.000,1929.107,    0.000}.
{  {gen_server,try_dispatch,4},                      401, 1008.458,   3.111, 964.728,    0.672}.
{  {erlang,put,2},                                   300,  990.732,   1.272, 963.946,    0.525}.
{  {xprof_tracer_handler,handle_info,2},             201,  990.317,   3.215, 963.946,    0.525}.
{  {gen_server,try_dispatch,3},                      401,   40.060,   1.779,   0.862,    0.641}.
{  {xprof_tracer,handle_info,2},                     200,   20.083,   2.480,   0.706,    0.150}.
{  {xprof_tracer,send2pids,2},                       200,    8.857,   2.167,   0.695,    0.056}.
{  {xprof_tracer,check_for_overflow,1},              200,    8.735,   2.374,   0.000,    0.094}.
{  {xprof_tracer,send2pid,2},                        400,    6.690,   3.573,   0.695,    0.056}.
{  {xprof_tracer_handler,maybe_make_snapshot,1}, 201,    5.937,   3.412,   0.000,    0.058}.
{  {application,get_env,3},                          200,    5.422,   1.600,   0.000,    0.026}.
{  {application,get_env,2},                          200,    3.822,   0.813,   0.000,    0.026}.
{  {xprof_tracer_handler,put_ts_args,3},             100,    3.234,   2.251,   0.000,    0.138}.
```

Overhead

- Profile XProf with XProf
- Profile XProf with fprofx (cachegrind)



Overhead

- Profile XProf with XProf
- Profile XProf with fprofx
- Overhead of tracing
  - Message passing / term copying
  - Greatly improved in OTP 19
  - Further improve with tracer modules

Future ideas

Usability/new features

- Different visualisations (histogram, heatmap)
- Event overlay
- Erlang record syntax support
- Shortcuts/GUI buttons for complex match-specs
- Other trace events (GC, processes, ports)

Reduce overhead

- Sampling processes/events
- Tracer module (NIF) from OTP 19
- LTTng/other dyntrace support

Please tell us more!

Summing up

- Simple ad-hoc profiling
- Immediate visual feedback
- Versatile usage
- Try XProf and contribute!

https://github.com/Appliscale/xprof

https://hex.pm/packages/xprof