Utilizing tools made for *Big Data* to analyse Ftrace data: making it fast and easy

Yordan Karadzhov

VMware Inc. - OSTC
Motivation

a. Ftrace: The official tracer of the Linux kernel (No need to explain this)

b. Is the Ftrace data *Big Data*?
   * Not necessarily. It depends how you use it.
   * Extremely sophisticated instrument. Large variety of use cases.
Motivation

a. OK, I have a nontrivial or very user-specific problem.

b. I have recorded a lot of tracing data.

c. What should I do now?
Motivation

a. OK, I have a nontrivial or very user-specific problem.

b. I have recorded a lot of tracing data.

c. What should I do now?

KernelShark

Not going to explain it here. See Steven Rostedt’s presentation at OSS NA 2019.
KernelShark (Something More)

Stolen slide from Steven’s presentation at OSS NA 2019.
Steven is doing:

* Switching to **Marker A** and clicking at the right event.
* Switching to **Marker B** and clicking at the right event.
* Getting the latency value that shows up in **A B Delta**.
Steven is doing:

* Switching to **Marker A** and clicking at the right event.
* Switching to **Marker B** and clicking at the right event.
* Getting the latency value that shows up in **A B Delta**.

Now imagine doing this 10K times - Ugh!!!
There must be a better way to get this job done.

Imagine having something like this:
#!/usr/bin/env python3

import ksharkpy as ks
...
ks.open_file('trace.dat')
data = ks.load_data()
data_size = ks.data_size(data)
...
for i in range(data_size):
    if data['event'][i] == my_event_a:
        action1
    elif data['event'][i] == my_event_b:
        action2
...
ks.close()
print('some summary of the results')
plot('some cool histograms or graphs for my presentation')
NumPy

a. **General purpose languages:** C, Perl, Python ...

b. **Numerical languages:** Fortran, MATLAB, R, ...
   * Written mostly for scientific numerical use.

Python + Scientific computing = NumPy
NumPy is not built in to the Python language. It is a library.

It provides:

1. Powerful densely packed N-dimensional arrays of homogeneous type.

2. Large collection of high-level mathematical functions to operate on these arrays.


4. Complementary packages like:
   a. Matplotlib - plotting package that provides MATLAB-like plotting functionality.
   b. SciPy - library that adds functionalities for optimization, linear algebra, integration, interpolation, FFT, signal and image processing.
NumPy

* Many Numpy operations are implemented in C.

* In fact the Numpy arrays are very similar to the C arrays.

* Numpy array can be initialized from C-computed array without data copying - **COOL!!!**
NumPy

* Many Numpy operations are implemented in C.

* In fact the Numpy arrays are very similar to the C arrays.

* Numpy array can be initialized from C-computed array without data copying - COOL!!!

Let’s use something that is already (almost ;-) available in KernelShark

libkshark.so
Example of loading data using libkshark.so:

```c
#include "libkshark.h"

int main(int argc, char **argv)
{
    struct kshark_context *kshark_ctx = NULL;
    struct kshark_entry **data = NULL;
    int data_size;

    kshark_instance(&kshark_ctx);
    kshark_open(kshark_ctx, "trace.dat");
    data_size = kshark_load_data_entries(kshark_ctx, &data);

    for (r = 0; r < data_size; ++r) {
        if (data[i]->event_id == my_event_a)
            action1;
        if (data[i]->event_id == my_event_b)
            action1;
    }
}
... /* Free the memory. */
for (r = 0; r < data_size; ++r)
    free(data[r]);
free(data);

/* Close the file. */
kshark_close(kshark_ctx);

/* Close the session. */
kshark_free(kshark_ctx);

printf("some summary of the results")
/*
 * Unfortunately, no simple way to show cool histograms/graphs here :( 
 */

return 0;
}
Summary

a. PoC NumPy interface for accessing Ftrace data in Python (via NumPy arrays).

b. The implementation is just a tiny wrapper around libkshark
Summary

a. PoC NumPy interface for accessing Ftrace data in Python (via NumPy arrays).

b. The implementation is just a tiny wrapper around libkshark

Let’s see some examples.
Example 1

Trying to reproduce the study done by Prof. Dr. Wolfgang Mauerer and Daniel Wagner.

See:

*Cyclic Tests Unleashed: Large-Scale RT Analysis with Jitterdebugger*
Open Source Summit Japan 2019
Example 1

Goal:

* Statistical estimate of the probability of exceeding the Worst Case Execution Time (WCET)
* Remember that this is just an example demonstrating the PoC NumPy interface for Ftrace data. The whole credit for the development of the analysis itself goes to Wolfgang and Daniel.
Example 1: Jitterdebugger - Idle machine
Example 1: Jitterdebugger - Idle machine
Example 1: Jitterdebugger

To make it more interesting, let’s do the test on a heavy loaded system.

Hackbench: stress test for the Linux kernel scheduler.
Example 1: Jitterdebugger

Extreme Value Theory: Peak over Threshold (PoT) approach
Example 1: Jitterdebugger

\[ f(x) = \left(1 + \frac{\xi(x - \mu)}{\sigma}\right)^{-(1+1/\xi)} \]

**Generalized Pareto distribution:** A continuous probability distributions. It is often used to model the tails of other distributions.
Example 1: Jitterdebugger $\sim 2.4M$ cycles

\[ \xi = 0.393 \pm 0.061 (15.53\%) \]
\[ \sigma = 1.124 \pm 0.031 (2.77\%) \]
\[ \mu = 10 \text{ (const)} \]
\[ \text{goodness-of-fit: 1.797} \]
Example 1: Jitterdebugger $\sim 2.4M$ cycles

$\xi = 0.393 \pm 0.061 (15.53\%)$

$\sigma = 1.124 \pm 0.031 (2.77\%)$

$\mu = 10$ (const)

goodness-of-fit: 1.797
Example 1: Jitterdebugger

Idle vs. Hackbench
Example 1: Jitterdebugger

Let’s see some real code
KernelShark session description file (JSON)

Allows loading predefined sessions.

```json
{
  "type": "kshark.config.session",
  "Data": {
    "type": "kshark.config.data",
    "file": "trace-jitter-load.dat",
    "time": 1567097989
  },
  "Model": {
    "type": "kshark.config.model",
    "range": [5831246982148, 5831250072340],
    "bins": 1000
  }
...
```
"Markers": {
    "type": "kshark.config.markers",
    "markA": {
        "isSet": true,
        "row": 8467891
    },
    "markB": {
        "isSet": true,
        "row": 8469063
    },
    "Active": "A"
},
"CPUPlots": [0],
"TaskPlots": [26456],
"ViewTop": 8467886}
Example 1: Jitterdebugger

kernelshark -s max_lat.json
Example 2: Page Faults

Demo