Much Ado About… Migrations!

Francesco Ciraolo (University of Turin)
Dario Faggioli (SUSE)
Enrico Bini (University of Turin)

Linux Plumbers Conference

September 20th, 2021

The problem

- The motivation of migrations could be useful for various purposes, e.g. testing and documentation writing
- Currently the only way is to manually dive through traces' millions records and track back the migration's cause
- This method is not very *scalable*, since it requires a relatively large time for each migration
- An automatic way to deduce the reason of each migration in a trace would be highly desirable

Current goals

1. To give some depth to the record representation:
   - modeling the data to offer both common and event (or plugin) specific API
   - defining fine-grained attributes filtering
   - merging continuous information spread among several records
2. To provide interactive research support
   - finding the next record filters compliant
   - *semantic* information collection from the skipped records
   - plain history grouped by core
3. To offer some global or local statistics
   - CPU time percentage per process and per core
   - processes pinning chance
4. To label each migration with its root cause, in an automatic way

The way

- Building composite objects over of the records, splitting common parts and specific record-type information
- Handling the trace in a functional-like way; trace as a stream of records
- Defining stackable, and customizable, computation blocks

Some examples

```
#########################
Current filters
#########################

1) Filter by PID: 4849
2) Filter by type: sched_migrate_task

Press enter to continue [q to exit]:


#########################
Search module
#########################

<idle>-0      [003]   406.626370:   sched_migrate_task:
  comm=while1 pid=4849 prio=120 orig_cpu=9 dest_cpu=3
  1) Search next record
  2) Reload trace from beginning
  3) Cores running tasks
  4) Cores' idle time
  5) Print last records
  6) Save found records
  7) Show stacktrace
  8) Main menu

Insert menu index [q to exit]:
```

```
#########################
Running tasks per core
#########################

[000] => swapper/0:0
[001] => swapper/1:0
[002] => swapper/2:0
[003] => swapper/3:0
[004] => swapper/4:0
[005] => swapper/5:0
[006] => swapper/6:0
[007] => swapper/7:0
[008] => swapper/8:0
[009] => kworker/9:2:335
[010] => swapper/10:0
[011] => swapper/11:0
```

```
#########################
Cores' idle time
#########################

[000] => 100.00%
[001] => 100.00%
[002] => 100.00%
[003] => 100.00%
[004] => 100.00%
[005] => 100.00%
[006] => 100.00%
[007] => 100.00%
[008] => 99.99%
[009] => 0.00%
[010] => 100.00%
[011] => 100.00%
```

Use case

- `while(1)` in unloaded system migrations analysis
- A CPU bound process, even less extreme than this, avoids to leave willingly the running state and no process, in the system, requires more cores than available; therefore little to no migrations should occur
- Unexpectedly the process experiences a non-negligible number of migrations and the most of this migration could be counterintuitive

Use case II

System setup

- Kernel version: 5.15-rc1
- CPU: i7-9750H
  - 6 cores
  - 12 threads

Tool	Min	Mean	SD	Max
stress-ng	51.11	52.59	0.89	54.01
stress-ng†	50.98	53.29	1.52	56.86
sysbench	97.72	101.46	1.35	102.77
sysbench†	99.34	101.56	0.88	102.62

```
7361.081433:   sched_migrate_task: comm=while1 pid=5659 prio=120 orig_cpu=4 dest_cpu=10
```

CPU10			CPU4		
Process	Switch delta	Time	Process	Switch delta	Time
while1:5659	0.000	99.9987%	swapper/4:0	0.000	99.9455%
kworker/10:2:559	0.832	0.0012%	llvmpipe-8:2329	1.929	0.0538%
migration/10:73	3.112	0.0000%	kworker/4:1:183	0.032	0.0005%
			llvmpipe-11:2332	1.931	0.0001%
			llvmpipe-9:2330	1.932	0.0001%
			migration/4:37	3.108	0.0000%

```
7367.129463:   sched_migrate_task: comm=while1 pid=5659 prio=120 orig_cpu=10 dest_cpu=4
```

† Pinned execution

Thank you!

Open to questions and call for feedback

Could this tool be useful?
Would it be worthy to try to make the labeling automatic?
Any suggestion about its development path?

Francesco Ciraolo

- University of Turin
- francesco.ciraolo@edu.unito.it

Dario Faggioli

- SUSE
- dfaggioli@suse.com

Enrico Bini

- University of Turin
- enrico.bini@unito.it