Improve Linux Perf tool to account for task sleep
Improve Linux Perf tool to account for task sleep
Agenda

- Problem Statement: Linux Perf tool not monitoring sleeping tasks

- Perf tool architecture:
  - Sample collection
  - No sample when target is sleeping

- Solutions:
  - Calculate sleep time and add to total sample count
  - Capture sleep sample
Problem Statement: Linux Perf tool not monitoring sleeping task

Perf tool doesn’t collect samples if target process is in sleep state which leads to:

- Incorrect ‘CPU usage’ calculation:
  If target task was in sleep state for around 50% of the time, the CPU usage represented by the perf tool does not account for the same.

- No ‘task sleep time’:
  As the perf tool does not provide any sleep sample, it is not possible to determine for how long the task was in sleep state.

---

**CPU usage calculation:**

- Function 1 has sample 1, 6 = 2 samples
- Function 2 has sample 2, 3, 4, 5, 7, 8 = 6 samples
  Total samples = 8

- Function 1 CPU usages = (2/8)*100 = 25%
- Function 2 CPU usages = (6/8)*100 = 75%
Problem Statement: Linux Perf tool not monitoring sleeping task

Perf tool doesn’t collect samples if target process is in sleep state which leads to:

- Incorrect ‘CPU usage’ calculation:
  If target task was in sleep state for around 50% of the time, the CPU usage represented by the perf tool does not account for the same.

- No ‘task sleep time’:
  As the perf tool does not provide any sleep sample, it is not possible to determine for how long the task was in sleep state.

Goal: Perf tool to consider task’s sleep time for CPU usage calculations
Problem Statement: Linux Perf tool not monitoring sleeping task

- Function 1 has sample 1, 6 = 2 samples
- Function 2 has sample 2, 3, 4, 5, 7, 8 = 6 samples
  Total samples = 8

CPU usage calculation:
- Function 1 CPU usages = (2/8)*100 = 25%
- Function 2 CPU usages = (6/8)*100 = 75%

- Task Sleeping
- Task Running

- Function 1 has sample 1, 6 = 2 samples
- Function 2 has sample 2, 3, 4, 5, 7, 8 = 6 samples
  Total samples = 8 + 4 (skipped samples) = 12

CPU usage calculation:
- Function 1 CPU usages = (2/12)*100 = 16.6%
- Function 2 CPU usages = (6/12)*100 = 50%
  Sleep time = (4/12)*100 = 33.3%
Perf tool architecture: Sample collection

Execution flow: Perf collecting samples

Target task:
- perf_swevent_init_hrtimer()
- perf_swevent_start_hrtimer()
- perf_swevent_hrtimer() → perf_get_regs_user()
- perf_swevent_hrtimer() → perf_get_regs_user()
- perf_swevent_stop_hrtimer()

Perf record task:
- init
- start timer
- save to perf.data
- save to perf.data
- stop timer

start timer
save to perf.data
stop timer
Perf tool architecture: Sample collection

Execution flow: Perf collecting samples
Perf tool architecture: No sample when target is in sleep

Execution flow: Perf not collecting samples when target task is in sleep
Perf tool architecture: No sample when target is in sleep

Execution flow: Perf not collecting samples when target task is in sleep
Perf tool architecture: No sample when target is in sleep

Execution flow: Perf not collecting samples when target task is in sleep
Perf tool architecture: No sample when target is in sleep

Execution flow: Perf not collecting samples when target task is in sleep
Perf tool architecture: No sample when target is in sleep

Execution flow: Perf not collecting samples when target task is in sleep
Solution #1: Capture sleep sample
Solution #1: Capture sleep sample

Target task

- sleep → perf_swevent_stop_hrtimer()
- wake up → perf_swevent_start_hrtimer()
Solution #1: Capture sleep sample

Target task

- sleep → perf_swevent_stop_hrtimer()
- wake up → perf_swevent_start_hrtimer()
Solution #1: Capture sleep sample

```
Solution #1: Capture sleep sample

Target task

- sleep
- perf_swevent_stop_hrtimer()
- wake up
- perf_swevent_start_hrtimer()

soft irq

- perf_swevent_hrtimer()
- perf_get_regs_target()

Perf record task

- sleep sample
- save to perf.data
```
Solution #1: Capture sleep sample

Target task

Sleep

perf_swevent_stop_hrtimer()

Wake up

perf_swevent_start_hrtimer()

soft irq

perf_swevent_hrtimer()

perf_get_regs_target()

sleep sample

Perf record task

save to perf.data

Disadvantages: size of perf.data will increase.
Solution
Solution #2: Calculate sleep time and add to total sample count
Solution #2: Calculate sleep time and add to total sample count

Target task

- sleep → perf_swevent_stop_hrtimer()

- wake up → perf_swevent_start_hrtimer()

Perf record task

- save to perf.data
Solution #2: Calculate sleep time and add to total sample count

Target task

- sleep → perf_swevent_stop_hrtimer() → stop_time = local_clock();

- wake up → perf_swevent_start_hrtimer()

Perf record task

- save to perf.data
Solution #2: Calculate sleep time and add to total sample count

```
#2: Calculate sleep time and add to total sample count

Target task

sleep  ➔  perf_swevent_stop_hrtimer()  ➔  stop_time = local_clock();

wake up  ➔  perf_swevent_start_hrtimer()  ➔  start_time = local_clock();
  sleep_time = start_time – stop_time

Perf record task

sleep_time sample  ➔  save to perf.data
```
Solution #2: calculate sleep time and add to total sample count

Perf report task

reader_process_events() → process_sample_event() → add to sample list

reader_process_events() → process_sleep_event() → add to sleep_time
Solution #2: calculate sleep time and add to total sample count

sample_time = 1 second / sample_per_second

samples_time = sample_count * sample_time

total_time = samples_time + sleep_time
Solution #2: calculate sleep time and add to total sample count

```
sample_time = 1 second / sample_per_second
samples_time = sample_count * sample_time
total_time = samples_time + sleep_time
```

CPU usage % = (samples_time / total_time) * 100

sleep % = (sleep_time / total_time) * 100
Perf tool snapshots:

```c
int main(int argc, char* argv[]) {
    int pwm = strtol(argv[1], NULL, 0);
    while (1) {
        struct timeval stop, c;
        gettimeofday(&stop, NULL);
        stop.tv_sec++;
        usleep(((100 - pwm) * 10000);
        while (1) {
            gettimeofday(&c, NULL);
            if (c.tv_sec > stop.tv_sec)
                break;
            if (c.tv_sec == stop.tv_sec && c.tv_usec > stop.tv_usec)
                break;
        }
    }
    return 0;
}
```

```
./a.out 10 (90% sleep, 10% duty cycle)
```

```
perf record --call-graph dwarf -p `pidof a.out` -F 10 -- sleep 4
```

Samples: 4 of event 'cpu-clock:ppp', Event count (approx.): 40000000

<table>
<thead>
<tr>
<th>Children</th>
<th>Self</th>
<th>Command</th>
<th>Shared Object</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>- 100.00% 100.00%</td>
<td>a.out [vdso]</td>
<td></td>
<td>__vdso_gettimeofday</td>
<td></td>
</tr>
</tbody>
</table>

Total expected samples = 40
Actual sample collected = 4 (which is 10% of 40)
Perf tool snapshots:

**Existing Perf tool**

<table>
<thead>
<tr>
<th>Children</th>
<th>Self</th>
<th>Command</th>
<th>Shared Object</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.00%</td>
<td>100.00%</td>
<td>a.out</td>
<td>[vdso]</td>
<td>__vdsogettimeofday</td>
</tr>
</tbody>
</table>

```
_start
__libc_start_main
0x7f7fe0e45189
main
__vdsogettimeofday
```

<table>
<thead>
<tr>
<th>Children</th>
<th>Self</th>
<th>Command</th>
<th>Shared Object</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>a.out</td>
<td>__start</td>
</tr>
<tr>
<td>100.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>lib.so.6</td>
<td>__libc_start_main</td>
</tr>
<tr>
<td>100.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>lib.so.6</td>
<td>0x00007f7fe0e5189</td>
</tr>
<tr>
<td>100.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>a.out</td>
<td>main</td>
</tr>
</tbody>
</table>

**Solution #2: Perf tool**

<table>
<thead>
<tr>
<th>Children</th>
<th>Self</th>
<th>Command</th>
<th>Shared Object</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.00%</td>
<td>10.00%</td>
<td>a.out</td>
<td>[vdso]</td>
<td>__vdsogettimeofday</td>
</tr>
</tbody>
</table>

```
_start
__libc_start_main
0x7f7fe0e45189
main
__vdsogettimeofday
```

<table>
<thead>
<tr>
<th>Children</th>
<th>Self</th>
<th>Command</th>
<th>Shared Object</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>a.out</td>
<td>__start</td>
</tr>
<tr>
<td>10.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>lib.so.6</td>
<td>__libc_start_main</td>
</tr>
<tr>
<td>10.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>lib.so.6</td>
<td>0x00007f7fe0e5189</td>
</tr>
<tr>
<td>10.00%</td>
<td>0.00%</td>
<td>a.out</td>
<td>a.out</td>
<td>main</td>
</tr>
</tbody>
</table>
Solution #2: Calculate sleep time and add to total sample count

```c
perf_swevent_start_hrtimer()
start_time = local_clock();
sleep_time = start_time – stop_time
```

Perf record task

```c
sleep_time sample
save to perf.data
```

Target task

```c
sleep
perf_swevent_stop_hrtimer()
stop_time = local_clock();
perf_get_regs_user()
```

```c
wake up
perf_swevent_start_hrtimer()
start_time = local_clock();
sleep_time = start_time – stop_time
```
HAVE ANY QUESTIONS AND DOUBTS
Thanks