Part II - Advanced Operating Systems: Lab 2 - IPC

Dr Robert N. M. Watson

2020-2021

This is Part II Lab 2. If you are an L41 student, please see the other lab variant.

Your lab assignment will compare several configurations of the IPC benchmark, exploring (and explaining) performance differences between them. Do ensure that your experimental setup suitably quiesces other activity on the system, and also uses a suitable number of benchmark runs.

Hypotheses

In this lab, we provide you with three hypotheses that you will test and explore through benchmarking:

1. *Larger IPC buffer sizes improve IPC performance regardless of IPC model selected.*
2. *The kernel's automatic socket-buffer sizing achieves better performance than a fixed socket-buffer size.*
3. *The probe effect associated with HWPMC is negligible.*

We will test these hypotheses by measuring net throughput between two IPC endpoints in two different threads. We will use DTrace and HWPMC to establish the causes of divergence from these hypotheses, and to explore the underlying implementations leading to the observed performance behavior. As we have considered the probe effect associated with DTrace in our prior lab, we will consider only the probe effect of HWPMC in this lab.

Approach

The following questions are with respect to a fixed total IPC size (the default 16MiB). As with Lab 1, take measurements across a spectrum of powers-of-two buffer buffer sizes between 1 byte and the total size. Use `2thread` mode in all experiments. You will use local socket IPC for all experiments (`-i local`), and configure it in one of two modes:

Automatic socket-buffer sizing The default configuration for this benchmark, the kernel will detect when socket-buffer sizes become full, and automatically expand them.

Fixed socket-buffer sizes When run using the `-s` argument, the benchmark will automatically set the sizes of the send and receive socket buffers to the buffer size passed to the benchmark.

Experimental questions

The questions first request an exploration using DTrace, and then extend your data collection and analysis to use HWPMC.

1. Performance

Create a plot illustrating IPC bandwidth across the full range of buffer sizes, using both fixed and automatically sized socket buffers.

2. Kernel IPC statistics

The benchmark is able to capture process information regarding performed IPC operations using `getrusage(2)`:

- Create a plot showing how the `msgsnd` and `msgrcv` counters vary across the full range of buffer sizes.
- Succinctly summarise how the different socket-buffer configurations and parameterisations interact with the number of unique send and receive operations that are performed. Comment on the potential implications for the performance results you have measured.

For the purposes of this question, you can work with a single run of the benchmark for each buffer size.

3. Kernel tracing

Using DTrace's syscall provider, measure the distribution of `read(2)` and `write(2)` returned lengths in both IPC configurations across the range of buffer sizes:

- Create a plot comparing the distributions of `read(2)` returned lengths for each of the two socket IPC variants across the range of buffer sizes.
- Create a plot comparing the distributions of `write(2)` returned lengths for each of the two socket IPC variants across the range of buffer sizes.
- Describe how this gathered data relates to the corresponding performance data, and, as relevant, suggest possible reasons for this.

For the purposes of this question, you can work with a single run of the benchmark for each buffer size.

Please include your D scripts as part of your submission.

4. Microarchitectural counters

Using the `-P` *`mode`* command-line argument to the benchmark to capture conter statistics relating to memory access: architectural or speculative load, store, and instruction fetch; L1I, L1D, and L2 cache hits and misses; ITLB and DTLB misses; and memory and bus accesses. You may disregard other counters, such as those relating to branch prediction and exception handling.

- Create a set of plots comparing corresponding values for each of the two IPC configurations, one per counter mode (i.e., one each for `arch`, `dcache`, `instr`, and `tlbmem`).
- Using this counter data, and data collected earlier in the lab about OS behaviour, propose a theory as to why the two IPC configurations perform the way they do relative to one another.

For the purposes of this question, you can work with a single run of the benchmark for each buffer size.

5. Probe effect

For each HWPMC counter set, gather performance benchmark results across the buffer-size parameter space for the default socket-buffer sizing configuration:

- Plot the four sets of performance results on the same graph, labelling them clearly.
- Explain what the probe effect for each of the HWPMC counter sets is.
- If you measure significant probe effect, explain why we might still have confidence in the analysis we have performed.

For the purposes of this question, you can work with a single run of the benchmark for each buffer size.

Notes

Graphs and tables should be used to illustrate your measurement results. Ensure that, for each question, you present not only results, but also a causal explanation of those results – i.e., why the behaviour in question occurs, not just that it does. For the purposes of graphs in this assignment, use achieved bandwidth, rather than total execution time, for the Y axis, in order to allow you to more directly visualise the effects of configuration changes on efficiency. All plots in this lab should use a log X axis.