

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

## Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-06/msg03329.html>

---

- *From:* Shailabh Nagar <nagar@xxxxxxxxxxxxxxxx>
  - *Date:* Mon, 12 Jun 2006 09:28:10 –0400
- 

Martin Peschke wrote:

Jay Lan wrote:

Shailabh Nagar wrote:

Balbir Singh wrote:

Andrew,

The only other new set of patches to be discussed in this context are the statistics–infrastructure patches from Martin Peschke.

That infrastructure cannot meet the needs of delay accounting, CSA etc. because  
– it only provides "user pull" model of getting stats whereas "kernel push" is needed for delay accounting

Doesn't taskstats interface provide "user pull" request–reply model also? Serious accounting needs to push accounting data as soon as possible.

– it uses a relatively slow interface unsuitable for high volumes of data.

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

By design.

I think it would be fatal to report every event relevant to statistical data gathering up to user space. It's fine to have the kernel maintain counters and to provide preprocessed data.

Given that, is there a need for a high-speed interface for a huge amount of unprocessed statistical data?

Broadly speaking, yes. Inserting policy into the kernel will no doubt save the data being sent to userspace but also

- limits flexibility of what userspace can do with it and
- adds to the kernel code base unnecessarily

Specifically for taskstats, there is a need for a high-speed interface because of the potential volume of data resulting from

- large number of tasks in a single kernel
- high frequency of task exits

Some kernel-based preprocessing, such as per-tgid aggregation, can help cut down the volume but as long as we have a need for getting per-task data, an efficient interface will matter, atleast for our needs.

However, the user interface is a just one building brick, which can be enhanced or replaced with moderate effort, if there is a need.

True. This is not to suggest statistical infrastructure's interface choice isn't correct.. just that its not enough for the needs we seek to serve.

A filesystem based interface has plenty of usability benefits so its primarily a question of which stats you want to export using its interface.

>> Each statistic has its own definition,

Allowing users to restrict accounting to what they need in their particular case. Sensible defaults are usually available.

needs to be read separately using ASCII,  
reading data continuously means open/read/close each  
time.....all of  
which is not very conducive to large structures being sent to  
userspace.

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

Debugfs file are fine for larger structures.

Unless one keeps reading statistics dozens of times per second,  
I don't see an issue with that.

Since delay accounting stats can be exploited for resource management at user space, if one wants to get data for all tasks/processes periodically, it could add up to a fairly high demand for user<->kernel bandwidth even if the frequency need for reading one task's stats isn't that high. Its a question of scalability as number of tasks increase.

For systemwide stats, your point is well taken...unlikely to be an issue, unless of course, one needs to read lots of them.

The question is: what are the requirements to be covered?

Yup...I think the infrastructures are serving differing needs.

Yes, i second the point. It won't be able to catch up the traffic.

– its oriented towards sampled data whereas taskstats isn't.

So, we have a good consensus from existing/potential users  
of taskstats and would  
very much appreciate it being included in 2.6.18.

Andrew, it has become clear that the community wants to see accounting data processing being moved to userspace. Thus there is a need for a common accounting interface to provide minimal works at kernel (via hooks at fork and exit) and deliver data to userspace.

Both, the statistics infrastructure on behalf of its exploiters as well as the exploiters of the taskstats interface do data preprocessing, that is, maintain counters in the kernel.  
User space counters won't perform, of course.

AFAICS, actual differences are:

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

Re: Merge of per task delay accounting (was Re: 2.6.18 –mm merge plans)

- triggers for data delivery to user space  
(statistics infrastructure: when user reads statistics through file,  
taskstats: on certain task related events, right?)

Taskstats data delivery is triggered by

- user asking for data (i.e. akin to reading through a file, only done through a command–response interface)
- on task exit event

- and, therewith, frequency of data delivery to user space

Martin

--Shailabh

–

To unsubscribe from this list: send the line "unsubscribe linux–kernel" in  
the body of a message to majordomo@xxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo–info.html>

Please read the FAQ at <http://www.tux.org/lkml/>