

## Re: RSS accounting (was: Re: 2.6.19-rc1-mm1)

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-10/msg03922.html>

---

- *From:* [ebiederm@xxxxxxxxxxxxx](mailto:ebiederm@xxxxxxxxxxxxx) (Eric W. Biederman)
  - *Date:* Wed, 11 Oct 2006 06:07:28 -0600
- 

Arjan van de Ven <[arjan@xxxxxxxxxxxxx](mailto:arjan@xxxxxxxxxxxxx)> writes:

On Tue, 2006-10-10 at 17:54 -0600, Eric W. Biederman wrote:

For processes shared pages are not special.

Actually the above is not quite true you can map a shared page twice into the same process but in practice it rarely happens.

depends on what question you want to answer with RSS.  
If the question is "workload working set size" then you are right. If the question is "how much ram does my application cause to be used" the answer is FAR less clear....

There are two basic concerns. How do you keep an application from going crazy and trashing the rest of your system? A question on what number do you need to implement a resource limit.

The other question is how do I get good information so I can effectively understand what kind of resources a given application is using and hopefully predict what kind of resources that application will use in the future.

The last time I tried to answer the question "how much ram does my application cause to be used" I had to slowly start up additional copies and watch how much free memory decreased.

Having a couple of additional counts in addition to RSS would probably be the most help in understanding resource usage. Counting the number of private dirty resident pages would be interesting. As would counting the number of pages that are resident in the process but not resident in any other process.

It might also help to have a per page report on which file it backs

Re: RSS accounting (was: Re: 2.6.19-rc1-mm1)

and how many user it has. Unfortunately that is totally overwhelming detail and the act of reporting it would quite likely change the result as it would take so much memory to store the result.

You seem to have an implicit definition on what RSS should mean; but it's implicit. Mind making an explicit definition of what RSS should be in your opinion? I think that's the biggest problem we have right now; several people have different ideas about what it should/could be, and as such we're not talking about the same thing. Lets first agree/specify what it SHOULD mean, and then we can figure out what gets counted for that ;)

Well I tried to defined it in terms of what you can use it for.

I would define the resident set size as the total number of bytes of physical RAM that a process (or set of processes) is using, irrespective of the rest of the system.

By physical RAM I mean that if a single page (if shared) is used twice by a single process it will be counted only once. This definition works well for shared and private pages.

COW pages are probably the most subtle. They are both shared and not shared. Since that sharing is application visible and at application discretion I would count COW pages as shared until they that sharing is broken, and then I would count them as private pages.

The principle is that you don't find the ``owner" of a page and charge the page to the ``owner". Instead you find the users of a page and charge all of them for the page exactly once.

By and large most of that usage comes from pages in the page tables so they are the most interesting items to count.

Things like file descriptors, inodes, page tables and other kernel memory are interesting but are generally overshadowed by the page table users, and generally kernel data structures don't have reverse maps which makes it difficult to charge all of the users.

So I think the counting should be primarily about what is mapped into the page tables. But other things can be added as is appropriate or easy.

The practical effect should be that an application that needs more pages than it's specified RSS to avoid thrashing should thrash but it shouldn't take the rest of the system with it.

The biggest instance of system memory that an application does not

Re: RSS accounting (was: Re: 2.6.19-rc1-mm1)

Re: RSS accounting (was: Re: 2.6.19-rc1-mm1)

seem to have true control over is the page cache. Some kind of limit that prevents one application from destroying everything another application doing seems interesting. But I expect the solution there are I/O limits and not memory limits.

Eric

—

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

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

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