What do the counters in /proc/[pid]/io mean?

Asked 14 years, 3 months ago Modified 3 years, 11 months ago

Viewed 30k times

41

I'm creating a plugin for Munin to monitor stats of named processes. One of the sources of information would be `/proc/[pid]/io`. But I have a hard time finding out what the difference is between `rchar`/`wchar` and `read_bytes`/`written_bytes`.

They are not the same, as they provide different values. What do they represent?

linux linux-kernel procfs

Share

Improve this question

Follow

edited May 30, 2018 at 17:22

Ciro Santilli OurBigBook.com

380k ● 117 ● 1.3k ● 1.1k

asked Sep 3, 2010 at 5:41

Kvisle

708 ● 1 ● 5 ● 12

Hope you don't mind the changes, some (me in particular) are not familiar with minority projects, less so those in ancient

languages ;) – Matt Joiner Sep 3, 2010 at 8:19

1 I can live with that, but it's not that small, really. I consider it well deployed. – Kvisle Sep 4, 2010 at 11:27

1 Answer

Sorted by: Highest score (default)

73

While the proc manpage is woefully behind (and so are most manpages/documentation on anything not relating to cookie-cutter user-space development), this stuff is fortunately documented completely in the Linux kernel source under `Documentation/filesystems/proc.rst`. Here are the relevant bits:

```
rchar
-----

I/O counter: chars read
The number of bytes which this task has caused to
be read from storage. This
is simply the sum of bytes which this process
passed to read() and pread().
It includes things like tty IO and it is
unaffected by whether or not actual
physical disk IO was required (the read might have
been satisfied from
pagecache)


wchar
-----

I/O counter: chars written
The number of bytes which this task has caused, or
shall cause to be written
to disk. Similar caveats apply here as with rchar.
```

```
read_bytes
----------

I/O counter: bytes read
Attempt to count the number of bytes which this
process really did cause to
be fetched from the storage layer. Done at the
submit_bio() level, so it is
accurate for block-backed filesystems. <please add
status regarding NFS and
CIFS at a later time>
```

Share Improve this answer

Follow

edited Dec 29, 2020 at 18:17

Community Bot

1 ● 1

answered Sep 3, 2010 at 8:24

Matt Joiner

118k ● 117 ● 386 ● 541

cancelled_write_bytes ---------- The big inaccuracy here is truncate. If a process writes 1MB to a file and then deletes the file, it will in fact perform no writeout. But it will have been accounted as having caused 1MB of write. In other words: The number of bytes which this process caused to not happen, by truncating pagecache. A task can cause "negative" IO too. If this task truncates some dirty pagecache, some IO which another task has been accounted for (in its write_bytes) will not be happening. We *could* just subtract that from the truncating task's write_bytes...
– Massimo Sep 4, 2018 at 16:19

1 It's important to know that IO done via the use of memmap will not be displayed in those counter and there's currently no way to easily measure such access – Kiwy Oct 5, 2018 at 12:25