Subject: Re: [RFC PATCH 0/5] Resend - Use procfs to change a syscall behavior
Posted by Pavel Machek on Thu, 10 Jul 2008 08:54:06 GMT
View Forum Message <> Reply to Message

On Thu 2008-07-10 09:42:03, Nadia Derbey wrote:
> Pavel Machek wrote:
>> Hi!
>>
>>
>>>>> An alternative to this solution consists in defining a new field in the
>>>>> task structure (let's call it next_syscall_data) that, if set, would change
>>>>> the behavior of next syscall to be called. The sys_fork_with_id() previously
>>>>> cited can be replaced by
>>>>> 1) set next_syscall_data to a target upid nr
>>>>> 2) call fork().
>>>>
>>>>
>>>> ...bloat task struct and
>>>>
>>>>
>>>>
>>>>> A new file is created in procfs: /proc/self/task/<my_tid>/next_syscall_data.
>>>>> This makes it possible to avoid races between several threads belonging to
>>>>> the same process.
>>>>
>>>>
>>>> ...introducing this kind of uglyness.
>>>>
>>>> Actually, there were proposals for sys_indirect(), which is slightly
>>>> less ugly, but IIRC we ended up with adding syscalls, too.
>>
>>
>>> I had a look at the lwn.net article that describes the sys_indirect()
>>> interface.
>>> It does exactly what we need here, so I do like it, but it has the same
>>> drawbacks as the one you're complaining about:
>>> . a new field is needed in the task structure
>>> . looks like many people found it ugly...
>>
>>
>>> Now, coming back to what I'm proposing: what we need is actually to
>>> change the behavior of *existing* syscalls, since we are in a very
>>> particular context (restarting an application).
>>
>>
>> Changing existing syscalls is _bad_: for backwards compatibility
>> reasons.
>

> I'm sorry but I don't see a backward compatibility problem: same interface,
> same functionality provided. The only change is in the way ids are
> assigned.

If you don't see a backward compatibility problem here, perhaps you
should not be hacking kernel...? The way ids are assigned is certainly
part of syscall semantics (applications rely on), at least for open.

If you want to claim that your solution is better than adding milion
of syscalls, I guess you need to list the milion of syscalls, so we
can compare.

> Actually, one drawback I'm seeing is that we are adding a test to the
> classical syscall path (the test on the current->next_syscall_data being
> set or not).
>
>> strace will be very confusing to read, etc...
>
> We'll have the 3 following lines added to an strace output each time we
> fill the proc file:
>
> open("/proc/15084/task/15084/next_syscall_data", O_RDWR) = 4
> write(4, "LONG1 100", 9)   = 9
> close(4)   = 0
>
> I don't see anthing confusing here ;-)

No, that part is just very very ugly.

close(5)
close(6)
open("foo") = 6

_is_ confusing to me.
Pavel
--
(english) http://www.livejournal.com/~pavelmachek
(cesky, pictures) http://atrey.karlin.mff.cuni.cz/~pavel/picture/horses/blog.html