Subject: Re: Mapping PIDs from parent-&gt;child namespaces
Posted by Mike Heffner on Wed, 05 Jan 2011 04:50:47 GMT
View Forum Message <> Reply to Message

On 01/04/2011 05:13 PM, Daniel Lezcano wrote:
> On 01/04/2011 08:57 PM, Mike Heffner wrote:
>> On 01/04/2011 11:04 AM, Daniel Lezcano wrote:
>>> On 01/04/2011 12:02 AM, Mike Heffner wrote:
>>>> Hi,
>>>>
>>>> Is it possible for a process running in a parent PID namespace to map
>>>> the PID of a process running in a child's namespace from the
>>>> parent->child's namespace? For example, if I span the process "myproc"
>>>> with CLONE_NEWPID then a call to getpid() inside myproc will return "1"
>>>> whereas in the parent's namespace that process could actually be PID
>>>> "23495". I'd like to be able to know that 23495 maps to 1 in the new
>>>> NS.
>>>> Obviously, just mapping the first PID is straightforward since I can
>>>> just look at the result of clone(). However, mapping the PIDs of
>>>> processes subsequently forked from "myproc" -- in this example -- I
>>>> haven't been able to figure out.
>>>
>>> AFAIK, it is not possible.
>>>
>>> That would be very nice to show the pid<-> vpid association.
>>>
>>> The procfs is a good candidate to show these informations.
>>>
>>> That would makes sense to show the content of /proc/<pid>/status with
>>> the pid relatively to the namespace.
>>>
>>> Let me give an example:
>>>
>>> Assuming the process '1234' creates a new pid namespace, and the child
>>> which is '1' in the new namespace has the real pid '4321'. This one
>>> mounts its /proc.
>>>
>>> If the process '1234' looks at /proc/4321/root/proc/1/status, it sees:
>>>
>>> ...
>>> Tgid: 1
>>> Pid: 1
>>> PPid: 0
>>> ...
>>>
>>>
>>> It could be:
>>>

>>> ...
>>> Tgid: 4321
>>> Pid: 4321
>>> PPid: 1234
>>> ...
>>>
>>> as the file is inspected from the parent namespace. Of course, if the
>>> file is looked from the child namespace context, we will see '1', '1'
>>> and '0'.
>>>
>>> I suppose the patch in the kernel should very small also.
>>>
>>> Thoughts ?
>>
>> Would that mean that finding the pid->vpid association for a real PID
>> X requires checking all files /proc/<X>/root/proc/<Y>/status where Y
>> is all vpids until you find the one where Pid == X? It would be nice
>> to have a have a way to check a single file for the association where
>> vpid is not known beforehand -- unless I'm misunderstanding your
>> solution.
>
> Hmm, right. But how do you know a pid is belonging to a specific pid
> namespace ? I mean you can have a single process creating several pid
> namespaces. So while looking at the /proc/<pid>/status, you can see
> several times the same vpid, no ?
> I am not sure the kind of informations you want to collect but it is not
> really a problem to build an association table from the userspace by
> browsing the /proc/<pid>/root/proc/<vpids> and their corresponding pid
> from the 'status' file information.

Yeah, I see how that could be a problem. I guess I was coming from an
assumption that you knew which namespace a real PID came from just not
which vpid it was in that namespace.

>
> Do you have an example for a pid -> vpid association without looking at
> more informations from /proc ?
>

I actually did discover another solution. My original requirement was
for an application monitoring solution where procs in a child namespace
need to contact an agent running in the root namespace. In this example,
the agent needs to know the real PID in order to query stats from /proc
and to know when the process has exited.

I originally tested SO_PEERCRED, but that just returned the vpid.
However, it does look like this was patched in 2.6.36 with:

http://www.spinics.net/lists/linux-containers/msg20944.html

SO_PEERCRED now returns the realpid in the agent while my application
can communicate the vpid over the socket.

____