Subject: Re: [RFC][PATCH][cryo] Save/restore state of unnamed pipes
Posted by Matt Helsley on Thu, 19 Jun 2008 23:46:00 GMT
View Forum Message <> Reply to Message

On Thu, 2008-06-19 at 00:59 -0700, sukadev@us.ibm.com wrote:
> Matt Helsley [matthltc@us.ibm.com] wrote:
>
> <snip>
> |
> | > | I don't see anything in the pipe man page, at least, that suggests we
> | > | can safely assume pipefds[0] < pipefds[1].
> | > |
> | > | The solution could be to use "trampoline" fds. Suppose last_fd is the
> | > | largest fd that exists in the final checkpointed/restarting application.
> | > | We could do (Skipping the PT_FUNC "notation" for clarity):
> | >
> | > |
> | > |
> | > | pipe(pipefds); /* returns 5 and 4 in elements 0 and 1 */
> | > | /* use fds after last_fd as trampolines for fds we want to create */
> | > | dup2(pipefds[0], last_fd + 1);
> | > | dup2(pipefds[1], last_fd + 2);
> | > | close(pipefds[0]);
> | > | close(pipefds[1]);
> | > | dup2(last_fd + 1, <orig pipefd[0]>);
> | > | dup2(last_fd + 2, <orig pipefd[1]>);
> | > | close(last_fd + 1);
> | > | close(last_fd + 2);
> | > |
> | > |
> | > | Which is alot more code but should work no matter which fds we get back
> | > | from pipe(). Of course this assumes the checkpointed application hasn't
> | > | used all of its fds. :(
> | > |
>
> It appears that this last_fd approach will fit in easier with current
> design of cryo (where we process one or two fds at a time and don't have
> the src_fds and dest_fds handy).
>
> BTW, we should be able to accomplish the above with a single-unused fd
> right (i.e no need for last_fd+2) ?

Yes, I think that's sufficient:

```
    int pipefds[2];

 ...
```

```
restarted_read_fd = 11;
restarted_write_fd = 12;

...

        pipe(pipefds);

/*
 * pipe() may have returned one (or both) of the restarted fds
 * at the wrong end of the pipe. This could cause dup2() to
 * accidentaly close the pipe. Avoid that with an extra dup().
 */
        if (pipefds[1] == restarted_read_fd) {
  dup2(pipefds[1], last_fd + 1);
  pipefds[1] = last_fd + 1;
}

if (pipefds[0] != restarted_read_fd) {
  dup2(pipefds[0], restarted_read_fd);
  close(pipefds[0]);
}

if (pipefds[0] != restarted_read_fd) {
        dup2(pipefds[1], restarted_write_fd);
  close(pipefds[1]);
}
```

I think this code does the minimal number of operations needed in the
restarted application too -- it counts on the second dup2() closing one
of the fds if pipefds[1] == restarted_read_fd.

Cheers,
 -Matt

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers