

> On Tue, Sep 04, 2012 at 03:00:07PM -0400, bfields wrote:
>> On Mon, Aug 20, 2012 at 02:18:13PM +0400, Stanislav Kinsbursky wrote:

>>>> Stanislav Kinsbursky <skinsbursky@parallels.com> writes:

>>>>

>>>>> This patch set introduces new socket operation and new system call:

>>>>> sys_fbind(), which allows to bind socket to opened file.

>>>>> File to bind to can be created by sys_mknod(S_IFSOCK) and opened by

>>>>> open(O_PATH).

>>>>>

>>>>> This system call is especially required for UNIX sockets, which has name

>>>>> lenght limitation.

>>>>>

>>>>> The following series implements...

>>>>>

>>>> Hmm. I just realized this patchset is even sillier than I thought.

>>>>

>>>> Stanislav is the problem you are ultimately trying to solve nfs clients

>>>> in a container connecting to the wrong user space rpciod?

>>>>

>>>>

>>> Hi, Eric.

>>> The problem you mentioned was the reason why I started to think about this.

>>> But currently I believe, that limitations in unix sockets connect or

>>> bind should be removed, because it will be useful it least for CRIU

>>> project.

>>>

>>>> Aka net/sunrpc/xprtsock.c:xs_setup_local only taking an absolute path

>>>> and then creating a delayed work item to actually open the unix domain

>>>> socket?

>>>>

>>>> The straight correct and straight forward thing to do appears to be:

>>>> - Capture the root from current->fs in xs_setup_local.

>>>> - In xs_local_finish_connect change current->fs.root to the captured

>>>> version of root before kernel_connect, and restore current->fs.root

>>>> after kernel_connect.

>>>>

>>>> It might not be a bad idea to implement open on unix domain sockets in

>>>> a filesystem as create(AF_LOCAL)+connect() which would allow you to

>>>> replace __sock_create + kernel_connect with a simple file_open_root.

>>>>

>>>>

>>> I like the idea of introducing new family (AF_LOCAL_AT for example)

```
>>> and new sockaddr for connecting or binding from specified root. The
>>> only thing I'm worrying is passing file descriptor to unix bind or
>>> connect routine. Because this approach doesn't provide easy way to
>>> use such family and sockaddr in kernel (like in NFS example).
>>>
>>>> But I think the simple scheme of:
>>>> struct path old_root;
>>>> old_root = current->fs.root;
>>>> kernel_connect(...);
>>>> current->fs.root = old_root;
>>>>
>>>> Is more than sufficient and will remove the need for anything
>>>> except a purely local change to get nfs clients to connect from
>>>> containers.
>>>>
>>>
>>> That was my first idea.
>>
>> So is this what you're planning on doing now?
>
> What ever happened to this?
>
```

Sorry, was busy.
I'll prepare patch today, I hope.

```
> --b.
>
>>
>>> And probably it would be worth to change all
>>> fs_struct to support sockets with relative path.
>>> What do you think about it?
>>
>> I didn't understand the question. Are you suggesting that changes to
>> fs_struct would be required to make this work? I don't see why.
>>
>> --b.
```

--
Best regards,
Stanislav Kinsbursky
