

# Some devfs and tty issues

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2008-03/msg00092.html>

---

- *From:* Arthur Hartwig <[Arthur.Hartwig@xxxxxxxxxx](mailto:Arthur.Hartwig@xxxxxxxxxx)>
  - *Date:* Mon, 17 Mar 2008 18:28:22 +1000
- 

I'm developing a driver for a USB hardware modem. I want to allow dialup logins from the modem. The driver currently allows me to use `cu` to set and examine modem parameters, make outgoing calls and converse with the called system. I'm more interested in allowing logins over an incoming call to the modem. The driver is modeled on the `uplcom` driver. If I plugin the modem and start `getty` on the `ttyU0` device (`# /usr/libexec/getty std.9600 ttyU0`) and then pull the adapter out of the USB socket a crash follows.

I'm happy to file a PR and supply stack traces or otherwise assist in debugging. A recent message to this list and its replies suggest others have also found the interactions between the `tty` driver and `devfs` to be somewhat obscure so I'm posting this in the hope that some other eyes or old hands might be able to point to point me to something I've missed.

These remarks apply to FreeBSD 6.3 RELEASE.

In `destroy_devl()` in `kern_conf.c` I think the call to `devfs_destroy()` appears too early in the function. The following scenario in `destroy_devl()` is possible:

1. in process A, `devfs_destroy()` in `fs/devfs/devfs_devs.c` called, clearing `CDP_ACTIVE`.
2. `msleep()` called; `devmtx` released
3. context switch to process B which issues an `open()` which results in a call to `devfs_populate()` which calls `devfs_populate_loop()` which finds `CDP_ACTIVE` clear and calls `dev_rel()` which results in the device structure getting freed.
4. sleep time expires and process A resumes, but retains pointer to now freed device structure.

The `devfs_destroy()` call would be better moved to somewhere towards the end of `destroy_devl()`, say after `SI_ALIAS` is cleared. The `dev` structure is still safe to reference after calling `devfs_destroy()` because the `devmtx` mutex is still held preveneting the freeing of the `dev` structure

After I moved the `devfs_destroy()` call down past the `msleep()` call I could still provoke a problem by the following

1. Plug in USB modem
2. `# cu -l cuaU0`

- to set modem parameters to auto answer
3. `kill cu`

## Some devfs and tty issues

4. dial in from PSTN
5. remove USB modem from USB socket.

Now kernel repeatedly reports:  
Purging 4294967245 threads from cuaU0

I expect it will take a long while to purge that many threads :-). At least longer than I was prepared to wait.

When this message was being output, the `si_threadcount` field of the `cuaU0` cdev structure contained `0xfffffcd` while much of the rest of the structure contained sensible looking values.

On repeating the scenario I observed:  
All threads purged from cuaU0  
Purging 4294967232 threads from ttyU0  
Purging 4294967231 threads from ttyU0

On a second repeat of the same scenario, the `cuaU0` cdev structure had `0xfffffd3` in the `si_threadcount` field when it was passed to `destroy_devl()`.

Tomorrow I'll set hardware watchpoints on the `si_threadcount` field of the cdev structures for both `ttyU0` and `cuaU0` in an attempt to catch where they are being modified to these extravagant values.

Arthur

---

freebsd-arch@xxxxxxxxxxx mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@xxxxxxxxxxx"