

Re: Getting a fully-qualified path from a PID

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2004-07/0251.html>

From: Joe Marcus Clarke (marcus_at_marcuscom.com)

Date: 07/21/04

To: Dan Nelson <dnelson@allantgroup.com>

Date: Wed, 21 Jul 2004 12:27:13 -0400

On Wed, 2004-07-21 at 11:12, Dan Nelson wrote:

> *In the last episode (Jul 20), Joe Marcus Clarke said:*
> > *What is the canonical way for a userland application to get the*
> > *fully-qualified path of an executable from its running PID? I know I*
> > *can do a readlink(2) on /proc/pid/file, but procfs is deprecated on*
> > *5.X, correct? Is there a more appropriate way to do this? Thanks.*
>
> *realpath(argv[0]) works for commands not run from \$PATH. Commands found*
> *through a PATH earch will just have the basename in argv[0] so you*
> *would have to check each PATH element until you found it. Note that*
> */proc/pid/file won't work if vn_fullpath() fails (say the original file*
> *has been unlinked, or the filename has expired from the kernel's*
> *cache).*
>
> *If you are examining another process, you can use the kvm_getargv() and*
> *kvm_getenvv() functions to fetch argv[0] and PATH out of the target*
> *process.*

Okay, I was thinking about that. What I was specifically interested in was processes spawned from \$PATH, so realpath isn't going to be much good to me there. I didn't know if there was a better way than getting the environ+argv with kvm, then searching each path element. Thanks for the clarification.

Joe

--

PGP Key : <http://www.marcuscom.com/pgp.asc>

- application/pgp-signature attachment: [This is a digitally signed message part](#)