Contribution ID: **67** Type: **not specified**

How can we make procfs safe?

Wednesday, September 14, 2022 11:00 AM (30 minutes)

Thanks to openat2(2), it is now possible for a container runtime to be absolutely sure that they are accessing the procfs path they intended by using RESOLVE_NO_XDEV|RESOLVE_NO_SYMLINKS (the main limitation before this was the fact that there was no way to safely do the equivalent of RESOLVE_NO_XDEV in userspace on Linux, and implementing the necessary behaviour in userspace was expensive and bug-prone).

However, this method does not help if you need to access magiclinks in procfs (RESOLVE_NO_XDEV blocks all magiclinks and even if we allowed magiclink-jumps within the same vfsmount this wouldn't help with any of the magiclinks we care about since they all cross the vfsmount boundary). Of particular concern are:

- /proc/self/fd/*
- /proc/self/exe
- When introspecting other processes, /proc/<pid>/ns/*, /proc/<pid>/cwd and /proc/<pid>/root.

The primary attack scenario is that we have seen attacks where not-obviously-malicious Kubernetes configurations have been used to get the container runtime to silently create unsafe containers (we need to access several procfs files when setting up a container and if any of the paths are redirected to "fake" procfs files, we would be silently creating insecure containers) – ideally it should be possible to detect these kinds of attacks and refuse to create containers in such an environment.

In this talk, we will discuss proposed patches to fix some of these endpoints (primarily /proc/self/fd/* through open(fd, "", O_EMPTYPATH)) and open up to a general discussion about how we might be able to solve the rest of them.

I agree to abide by the anti-harassment policy

Yes

Primary author: SARAI, Aleksa (SUSE LLC)

Presenter: SARAI, Aleksa (SUSE LLC)

Session Classification: Containers and Checkpoint/Restore MC

Track Classification: LPC Microconference: Containers and Checkpoint/Restore MC