Subject: Hidden process for init (PID 1)?
Posted by signal11 on Mon, 15 Dec 2008 09:05:17 GMT
View Forum Message <> Reply to Message

I'm trying to detect hidden processes within a container (no, you don't need a kernel module to hide a process.. trojan ps and top go a long way already).

I'm aware that OpenVZ already has a hidden process for every visible process in the container, and that there's an offset of 1024 in PID between hidden and visible processes, which allows to identify the legitimate hidden OpenVZ processes.

However, there remains one unaccounted hidden process (i.e. no visible process at hidden PID + 1024). On the other hand there's no obvious hidden process candidate for init (PID 1). Is it safe to presume that the unaccounted hidden process is the one corresponding to PID 1?

Subject: Re: Hidden process for init (PID 1)?
Posted by maratrus on Mon, 15 Dec 2008 11:39:04 GMT
View Forum Message <> Reply to Message

Hello,

Quote:
I'm aware that OpenVZ already has a hidden process for every visible process in the container

No, OpenVZ doesn't have a hidden process for every visible in the container. The process that is shown inside container is the same process that is shown on the HN (you should be able to manipulate your VE from inside the HN). But as we consider container as a standalone system we should support container's own pid space, that's why the process with pid = pid_ve is shown as a pid_ve + 1024 on the HN.
But OpenVZ doesn't change processes' names so we must find on the HN init processes for every VE running on that moment.

Subject: Re: Hidden process for init (PID 1)?
Posted by signal11 on Mon, 15 Dec 2008 11:55:40 GMT
View Forum Message <> Reply to Message

maratrus wrote on Mon, 15 December 2008 06:39Hello,

Quote:
I'm aware that OpenVZ already has a hidden process for every visible process in the container

No, OpenVZ doesn't have a hidden process for every visible in the container. The process that is

shown inside container is the same process that is shown on the HN (you should be able to manipulate your VE from inside the HN). But as we consider container as a standalone system we should support container's own pid space, that's why the process with pid = pid_ve is shown as a pid_ve + 1024 on the HN.
But OpenVZ doesn't change processes' names so we must find on the HN init processes for every VE running on that moment.

That doesn't answer the question, though: is it normal that (exactly) one hidden process does not follow the 'offset by 1024' rule? (and nitpicking aside, within the container it appears as a second, 'hidden' process.. what it really is is another issue).

But as far as I understand, every process in the container is a process on the host, usually with a fixed offset in PID. This would work for every process except init, which always has PID 1, so the associated PID on the host cannot follow the 'offset by 1024' rule, because with several containers you need several processes corresponding to init in these containers. I just wanted to know whether this interpretation is (more or less) correct.

Subject: Re: Hidden process for init (PID 1)?
Posted by maratrus on Mon, 15 Dec 2008 16:22:38 GMT

View Forum Message <> Reply to Message

Hi,

Quote:
That doesn't answer the question, though: is it normal that (exactly) one hidden process does not follow the 'offset by 1024' rule?

There are no hidden processes in OpenVZ. Please, describe in more detail what do you have in mind?

Quote:
But as far as I understand, every process in the container is a process on the host, usually with a fixed offset in PID.

You cannot rely on this fact. It could be true for a lot of examples but this is not the rule and could be broken down for example after VE will be migrated.

Subject: Re: Hidden process for init (PID 1)?
Posted by signal11 on Wed, 17 Dec 2008 09:36:01 GMT

View Forum Message <> Reply to Message

maratrus wrote on Mon, 15 December 2008 11:22Hi,

There are no hidden processes in OpenVZ. Please, describe in more detail what do you have in mind?

Quote:
But as far as I understand, every process in the container is a process on the host, usually with a fixed offset in PID.

You cannot rely on this fact. It could be true for a lot of examples but this is not the rule and could be broken down for example after VE will be migrated.

Within the container, you can detect both PIDs for a process, the one for the host process and the one in the VE PID namespace. But within the VE, ps, top, or other standard tools only list the PID in the VE PID namespace. Thus for all practical purposes, for an observer within the VE the host PIDs appear as hidden processes.

This is problematic because: if you want to check from within the container whether your 'ps' has been replaced by a rootkit to hide processes, you need to identify legitimate PIDs that represent host processes corresponding to PIDs in your VE PID namespace.

Insofar the 'offset by 1024' is nice, and as long as it works in the VE I'm concerned with, I don't have a problem with the fact that it might not be true always and everywhere.

The only problem I have, and what my question was about, is whether I have to expect that there will be one host PID (which in my guess would correspond to init in the VE) that will not follow the 'offset by 1024' rule.

Subject: Re: Hidden process for init (PID 1)?
Posted by khorenko on Wed, 17 Dec 2008 12:24:15 GMT

View Forum Message <> Reply to Message

Hi signal11,
Quote:
The only problem I have, and what my question was about, is whether I have to expect that there will be one host PID (which in my guess would correspond to init in the VE) that will not follow the 'offset by 1024' rule.

As maratrus has already said above in general you can _not_ relay on the "offset by 1024" rule for every process, not only "init". Moreover in recent dev-t kernels (2.6.26-x) rule "offset by 1024" almost never works.

BTW, /proc/$PID/status will show you both pid and vpid, so you can you this info for your investigations, for example:

```
# cat /proc/27179/status |egrep -i "env|pid"
```

```
Pid:    27179
PPid:   24599
TracerPid:      0
envID:  101
VPid:   28203
```

--
Konstantin