

---

Subject: Re: insmod problem

Posted by [adobriyan](#) on Fri, 25 Apr 2008 10:27:37 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

There must be some confusion about what kernel modules are, what they do, what OpenVZ does and what OpenVZ does not.

OpenVZ implements so-called container-style virtualization. This means only 1 (one) kernel image is in memory and controls the system and all, let's call them, virtual machines. The situation is no different from normal unpatched Linux kernel.

Modules allow administrator to load additional functionality at live kernel. This can be device driver, allowing operating system to interact with hardware, or file system driver, allowing to mount new filesystems and so on.

Module's code is loaded and linked against already existing kernel image in memory and possibly against other modules. It's loaded with the very same privileges as the rest of the kernel which are THE privileges.

Kernel module can access all kernel memory, the rest of the kernel can access. Kernel module can change many data structures without problems and all data structures with minimal workarounds. I'm talking about exported symbols here.

Corollary #1: kernel module can royally screw up the system in particularly disgusting ways. It can seamlessly give root privileges to any process. Which would be the least of your troubles, BTW.

Corollary #2: loading of kernel modules from inside VE/container is not allowed. Root of VE is viewed as untrusted here, as such, giving him such powerful weapon as `init_module(2)` is simply out of question. If you want to load something, let administrator of VE0 do it.

Corollary #3: if you want containerise some functionality your module gives to you, you most certainly should change your module. Depends on functionality it provides.

Other types of virtualization such as KVM allow you to have multiple operating system kernels, as such, you can load different modules into different kernels. In theory and modulo bugs in virtualization layer and in hardware, this can't screw

up master kernel. OpenVZ doesn't operate like this by design nor it was ever promised for OpenVZ to operate like this.

So, what your module actually does? In some case problem is fixed by allowing VE to use major:minor pairs and creating device nodes in.

---