Subject: Re: Container Test Campaign
Posted by Mark Huang on Fri, 23 Jun 2006 17:31:15 GMT
View Forum Message <> Reply to Message

Cedric Le Goater wrote:
> Did you contribute that feature to vserver ?

The feature is fairly specific to our needs and would not be very useful to the most common vserver use case (shared hosting).

> So you have different containers exposing the same IP address ? How do you
> assign incoming packets to a container ?

We wrote a kernel module that leverages netfilter hooks and ip_conntrack. You're only allowed to send IP, but you can send IP packets through any type of socket (TCP, UDP, raw IP, or even raw packet). As Marc mentioned, this flexibility was an absolute requirement.

The kernel module sits in the input and output path of the stack, and associates every incoming and outgoing packet with an ip_conntrack struct (to which we added container IDs). Once a container sends out an outgoing packet, it is entitled to receive incoming packets associated with that connection. A container may also receive incoming packets associated with ports that it has reserved by calling bind() (the kernel module keeps track of bind() calls). You can think of the kernel module as a local stateful firewall for sockets.

To users, it looks like they can run pretty much anything root would be able to, including programs that use raw IP sockets (ping and traceroute), programs that use raw packet sockets (tcpdump), and regular server apps (Apache, MySQL, etc.). When they run tcpdump, they of course only see packets that they "own" (i.e., packets that are associated with their active connections).

There's technical documentation for the kernel module on our website:

http://www.planet-lab.org/doc/vnet.php

The kernel module does a lot more than this as well, which is another reason that it hasn't been merged into mainline vserver. Recent features include virtualized TUN/TAP and IP aliasing support.

Lastly, you're of course free to browse the code:

http://cvs.planet-lab.org/cvs/vnet/