Subject: Linux Containers : next steps
Posted by Cedric Le Goater on Wed, 26 Jul 2006 16:46:38 GMT
View Forum Message <> Reply to Message

All,

Here's a brief summary of what i've gathered at ksummit/ols. Follows some thoughts on possible next steps.

Globally, there's a quite a good feeling from the community. They like the idea and are ready to help to get things in mainline. The code touches the core kernel and it will need a lot of reviews before it is accepted but, most of all, we need to agree on it.

The first steps are cleanups on the mainline kernel. Following are patchsets that should provide small enough features to be reviewed on lkml. Andrew said he would merged them in -mm if there is agreement like he did before. I think he is going to push what is already in -mm (ipc, utsname) in mainline. he expects us to port our projects or products on top of these patchsets or say what is wrong with them, why they fail to meet the requirements.

However, i've also heard many times that we should agree before flooding lkml. So I guess we should use the vserver, openvz, lxc-devel mailing-list (eric please subscribe to one) before sending our agreement or disagreement on lkml.

vserver@list.linux-vserver.org
devel@openvz.org
lxc-devel@lists.sourceforge.net

Here are some notes from Ksummit/OLS :

* what is a container

 - containers vs. namespace

   namespaces are interesting objects but they are heavily
   correlated. so we need a container object to aggregate them.

 - hierarchical containers

   not much to say. not useful I would say

 - explicit container

   It was stated that containers don't have to support

unmodified distros, which means that we can have restrictions and fix them later.

- user api

enter is the minimal api

* filesystems

- r/o bind mounts

being worked on by dave. hch will help.

- /proc and /sys isolation/virtualization

we should be able to mount different /proc in containers. hch said he add ideas on the topic and would help. /proc does not need to be complete in the first steps and unsupported file could be empty, which is also a way to clean up /proc

- shared subtree done

in 2.6.15

- shared mounts

patches exist

- union fs

patches also exist but there is resistance from the community

* namespaces

namespaces are fine if they are part of a container. this concept is to new to be carved in linux without some experiments. i'm convinced that eric will make sure that we don't take bad shortcuts on our way to namespace perfection :)

fast status on current work :

- utsname : is in -mm

- ipc : is in -mm

- user is still in discussion

I think the last fixes I have done fit with openvz and vserver in a container environment. I will resend. Then we can extend to vfsmount, etc, but this is huge.

- pid is the in attic

if we fix pid(1) it should be usable.

* network

we either try to have a fully virtualized interface in a container (VM approach) or we put some restrictions in place and follow the solaris zones approach. In fact i don't think we need to make a choice. both ideas are useful but may be the second one will be faster to push. I think Kirill had a 3rd ?

Kirill and Daniel agreed on making a first approach with route namespace, TCP socket tagging + iptables for incoming traffic in order to choose the right namespace. That will bring level 3 isolation. Eric agreed but he will want to have several implementations to order to study the performance/isolation

Jamal proposed first to ask on netdev and compile the advantages and drawbacks of the layer 2 - 3 approaches in a white paper.

Daniel is collecting information on the different approaches of the existing solutions (openvz, vserver, mcr, xen, bsd jails, ...). That will be a the information base for netdev.

it needs to be addressed by the network guys !!

* resource management

this would be the first real use of a basic container.

The people at the BOF said to keep it simple and stupid. Start with a process aggregation mechanism (containers) and build on top a resource management system for each linux subsytem.

Very good feedbacks on the UBC framework of OpenVZ but it will need to be splitted.

Here's what I think we should work on to move forward :

* first : have minimal container

 this is really important. without that concept in place we
 won't be able to port our products or projects and this is why
 we are working for mainline.

 - fix and merge existing namespaces in a container
   object
 - fix previous pid namespace and merge with container
 - user API : find a clean way to create, destroy, enter a
   container

* and next : have a useful mininal container

 resource management should be split to use previous
 framework. not my job, i'm a newby.

* to be sorted out fast :

 network
 container freeze

* long term :

 container c/r

comments ?

C.