Subject: Linux Memory Overcommit in OpenVZ
Posted by charlesl on Thu, 19 Jun 2008 13:00:37 GMT
View Forum Message <> Reply to Message

After thinking I'd fully understood the three main UBC memory parameters vmguarpages, privvmpages and oomguarpages, I find myself asking another question as follows (note for anyone unfamiliar with the standard Linux kernel overcommit, this is a good explanation: Linux Memory Overcommit).

So we have the following barriers/limits:
vmguarpages - The guaranteed memory for the VE (when allocating memory)
privvmpages - Sets an upper bound on the memory for the VE (when allocating memory)
oomguarpages - Usually the same as vmguarpages, sets the guaranteed limit under OOM condition (when de-allocating memory).
Further, these are the accounting values:
vmguarpages - unused
privvmpages - The total number of pages allocated + used
oomguarpages - The total number of pages used in RAM + swap
physpages - The total number of pages used in RAM

Okay, so that's how I understand things to work (please tell me if I'm wrong!).

Now, my question is about the default Linux memory overcommit - in an unmodified kernel, this allows an application to make any mallocs it likes successfully, then (in some OS dependent manner) deny applications memory when they come to actually use it if there are insufficient resources. BUT, OpenVZ counts allocations in its privvmpages held value so places an upper bound on the amount which may be allocated (even if that is not being used).

So these seem to be conflicting: on the one hand the Linux kernel doesn't care about allocations (only about used memory) and allows virtually all mallocs to succeed regardless of the memory situation, while on the other OpenVZ seems to account for (and prevent?) allocations above the barrier of privvmpages.

I also notice the OpenVZ wiki on UBC says that the total of all privvmpages barriers can be above the total RAM + Swap of the system for the very reason that the kernel doesn't care about mallocs, only about used memory. Does this mean a VE can safely allocate over privvmpages barrier (indeed over RAM + swap capacity of the machine) provided it is not all used, or will OpenVZ not allow this?

So which of these is true: is the OpenVZ kernel modified so that it really does take account of allocations too, or is it true that in an OpenVZ VE all mallocs will always succeed and it doesn't limit them?

Thanks in advance for any thoughts/explanations.