Memory Allocation

We currently migrate to MediaWiki from our old installation, but not all content has been migrated yet. Take a look at the Wiki Team page for instructions how to help or browse through our new wiki at wiki.linux-vserver.org to find the information already migrated.

How to allocate memory amongst Vservers

This is the 'trying to understand about memory' page!

relevant files

```
/proc/virtual/<xid>/limits
/etc/vservers/<vserver>/flags
```

examples

Note: this is modelled on a real vserver, but i've substituted the name `vs1234` with xid `1234`.

```
$ sudo cat /proc/virtual/1234/limit
PROC:           22              45              -1          0
VM:          58472          113395              -1          0
VML:             0               0              -1          0
RSS:         21787           58289              -1          0
ANON:         8506            8662              -1          0
FILES:         273             476              -1          0
OFD:           172             188              -1          0
LOCKS:           1               9              -1          0
SOCK:           12              12              -1          0
MSGQ:            0               0              -1          0
SHM:            79              79              -1          0

$ sudo cat /etc/vservers/vs1234/flags
cat: /etc/vservers/vs1234/flags: No such file or directory
```

what does this mean?

Hmmm... I'm not entirely sure! I had an hour long tutorial with Bertl, which i've summarised on the Memory Management page. There is also another page on the vserver wiki which explains some of this (the `/proc/virtual/.../limit` file) - but runs out just before it gets to the crucial information, and is unfinished (Resource Limits page).

Still, I have managed to figure out some of it. From the Resource Limits page, the five columns listed above are:

1. (short) name
2. current value
3. observed maximum
4. hard limit (maximum)
5. number of hits (max)

and, from the same page, the columns:

```
        PROC:  number of processes
        VM:    virtual memory pages     = the sum of all virtual pages inside the guest
        VML:   pages locked into memory = the sum of all virtual pages locked into memory
        RSS:   resident set size        = the number of pages currently present in RAM
        ANON:  anonymous memory pages   = the number of anonymous pages
        FILES: number of file handles     }
        OFD:   number of file descriptors }    These options are
        LOCKS: file system locks          }    not relevant to
        SOCK:  number of sockets          }    discussions about memory
        MSGQ:  message queue size         }
        SHM:   shared memory pages      = the shared memory (ipc)
```

of these, it is possible to limit some of them:

limit	config

PROC:	/etc/vservers/<vserver>/nproc
VM:	/etc/vservers/<vserver>/as
VML:	/etc/vservers/<vserver>/memlock
RSS:	/etc/vservers/<vserver>/rss
ANON:	--
FILES:	/etc/vservers/<vserver>/nofile
OFD:	--
LOCKS:	/etc/vservers/<vserver>/locks
SOCK:	--
MSGQ:	--
SHM:	--

I guess some of this is self explanatory - limiting the number of processes, for instance (although, is this really necessary? I guess so, sometimes...)

However, 5 am not sure about all of them, so I'll only talk about the ones I understand. before that, I also think it will be helpful to give some more basics on how memory works - I've been doing some more reading and have a bit more of an understanding, expecially with regards to high_mem.

using memory above 4GB

Most computers are intel architecture CPUs which are 32 bit. This basically means that the CPU is able to map out up to 4GB data - 2^32 = 4294967296 = 4GB. So, how best to use this? The default linux method has been to allocate 1GB to the kernel (for mapping out memory) and 3GB for user applications. Also, by default, the kernel starts at 0xc0000000 - which is 3GB (in hexidecimal). note: *starts*.

Ok, so high_mem allows that to be expanded. but to enable this, some of the (kernel space) memory needs to be used for mapping it out, as it's further away and consequently takes more time to access (is more _expensive_). Thus, the kernel space ends up with 896MB and the remainder of that GB is the mapping portion. Yeah, ok, I'm bullshitting and oversimplifying here.

So, it is then theoretically possible that a single application could use up all the high_mem - could even crash the system if the exchanging - swapping - between the kernel space and the high_mem space such that the high_mem space was so slow to swap out that the kernel space filled up and gave an 'out of memory' type error. Of course, if you limit individual applications - or, in the case of linux-vserver, contexts - then you can efficiently use the higher memory.

How is this done? How exactly does that all translate into the configuration options we have available to us? What I really know right now are just a few things, particularly which are helped by this:

```
VSZ=VIRT = number of pages currently mapped
RSS=RES  = number of pages currently in RAM
```

- taken from vserver-stat and vtop/top commands, http://www.linux-vserver/Memory+Management

Thus, I take it that VSZ-RSS=[amount in high_mem]

From here, it seems that we need to share the total amount of pages in RAM fairly between the vservers (note: this doesn't mean "equally" as it may be clear that one vserver requires a lot less, consistently, whereas another has an above average requirement on the memory). I guess it's time to start experimenting! here's what I figure the values should be about:

PROC:	/etc/vservers/<vserver>/nproc	probably unnecessary to limit number of processes, i feel
VM:	/etc/vservers/<vserver>/as	the sum of all virtual pages inside the guest
VML:	/etc/vservers/<vserver>/memlock	is the sum of all virtual pages locked into memory
RSS:	/etc/vservers/<vserver>/rss	the number of pages currently present in RAM
ANON:	-	the number of anonymous pages
FILES:	/etc/vservers/<vserver>/nofile	don't know what this is about
OFD:	-	-
LOCKS:	/etc/vservers/<vserver>/locks	don't know what this is about
SOCK:	-	-
MSGQ:	-	-
SHM:	-	the shared memory (ipc)

how to progress from here

My feeling is that the next step is to ensure adequate monitoring of individual vservers memory consumption using munin (or similar) to track the memory usage in terms of RSS and VSZ (either using vserver-stat or, more probably, polling the information directly from /proc/virtual/<xid>/limits to create an rrd graph) so that there is a historical comparison to any changes made. Then, you can start experimenting!

references and resources

- http://linux-vserver.org/Memory+Management
- http://linux-vserver.org/Resource+Limits
- http://linux-vserver.org/Caps+and+Flags
- http://linux-vserver.org/Memory+Management
- http://linux-mm.org/MemoryHierarchy
- http://linux-mm.org/VirtualMemory
- http://linux-mm.org/HighMemory

Document last modified Sat, 15 Apr 2006 22:34:52