

Re: Cannot activate sbcl

Source: <http://coding.derkeiler.com/Archive/Lisp/comp.lang.lisp/2007-11/msg00391.html>

- *From:* "Dimitar \"malkia\" Stanev" <malkia@xxxxxxxxxx>
 - *Date:* Tue, 06 Nov 2007 19:00:43 -0800
-

It's an interesting suggestion. It still requires enough contiguous free address space to eventually map the array though. BSS addresses are reserved at load time (using VirtualAlloc).

I could be wrong, but I think what happens is that Windows sees that your executable has a DATA section that might overlap a DLL's address space then it would reallocate that DLL (or others) to different address space. This happens before your application is started, or even fully loaded (not sure here, the wine source code might reveal something in that direction).

As you speculated, the difference between this approach and using VirtualAlloc dynamically in your code is that your program and all its DLLs were loaded first and possibly fragmented the space you could have used.

Something like that. Off course if your "array" was say 1.9gb, and there wasn't memory for the DLLs to fit with it, it won't run. What can be done here (another tricky thing) is to have many tiny executables with different BSS sections in them, which then load a bigger executable in the form of a DLL (e.g. the real executable is in DLL).

That's not a nice solution, and there might be way better solution than this one. But it worked for us, on one specific tool that required such big contiguous address space.

It is NOT safe for deployment – at least not by itself. Fragmentation is a function of application load order and it is not limited to the set of currently running applications – their loading may have been influenced by other programs that are no longer executing. You would still have to guarantee that SBCL was loaded at a time when there is enough contiguous global address space to satisfy the reservation.

Each application has its own virtual address space. The DLL's are usually loaded at some "standard" addresses, which is all for benefit of not having more than one copy in memory. In the hack I'm describing, it would make those DLL's to be reallocated (and probably wasting some real memory).

I mean really. In a perfect world, you would say – load all my dlls from any address, and this way make sure

Re: Cannot activate sbcl

that I have contiguous memory. But what is done by Windows (and I guess other OS) is a kind of "heh premature" (not really... or may be) optimization, for the sake of saving memory – e.g. use the same code for any other application – and because it's loaded in the same address space – the relocations are the same...

Here is an example to test the idea. Write the next one to file called a.c, and use the Microsoft Visual Studio compiler to compile it, just type "cl a.c". (If you use cygwin or mingw, remove the pragmas, and add the libraries to the command–line).

```
#include "windows.h"
```

```
unsigned char a[1024*1024*1024 + 850*1024*1024]; // 1.85GB contiguous memory that can be used for  
allocs.
```

```
#pragma comment(lib,"GDI32")  
#pragma comment(lib,"USER32")  
#pragma comment(lib,"KERNEL32")
```

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int  
nCmdShow)  
{  
MSG msg;  
while (GetMessage(&msg, NULL, 0, 0))  
{  
TranslateMessage(&msg);  
DispatchMessage(&msg);  
a[0] = 10; // This is simply so that the compiler/linker includes the array a  
}  
return msg.wParam;  
}
```

Now compile that one to say "a.exe", then run depends.exe on a.exe and press F7. Sort then by Actual Base. Here's what I've got:

<http://img222.imageshack.us/my.php?image=99627412ne2.png>

Then change the source code of a.c into a2.c by modifying the array a to be just a[1]:

```
unsigned char a[1];
```

Do the same and see the results:

<http://img144.imageshack.us/my.php?image=a2oh2.png>

As you can see when the buffer is 1.85GB three DLLs were reallocated – The Google Desktop one, WS2_32 and WS2HELP.DLL. in the second case (a[1]) the Google Desktop DLL was just sitting "in–the middle" of the virtual space. Once it's there after your application is loaded, I don't think you can move it.

Note: If that number a[1024*1024*1024 + 850*1024*1024] is too high for your system, decrease it (otherwise it would say something like "Access Denied").

Re: Cannot activate sbcl

Re: Cannot activate sbcl

Yes this is the kind of problem you would end up if you want to deliver it, this number would vary on different systems, but you can either come up with requirements for the application to the client, or find some other solution (for example different executables, or executable that's generated on the fly, or something even more low-level).

I guess that's now quite off-topic Common Lisp :)

.