

Re: Is VMS Security being dumbed-down for Java?

Source: <http://unix.derkeiler.com/Newsgroups/comp.os.vms/2006-03/msg01221.html>

- *From:* "Richard Maher" <mahe_rj@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 25 Mar 2006 12:04:49 +0800
-

Hi Steve,

Thanks again for the replies. If I can hold your interest for a couple of clarification questions, that you (and possibly someone else?) could help me out with that would be great.

"Hoff Hoffman" <hoff@xxxxxxxxx> pre-empted something a lot like :-
"Event flags are crap! And that's going in the Programming Concepts Manual :-)"

Then proceeded: -

As for the question, you can only use inner-mode-safe calls. Pretty or not, like it or not, want it or not, you can only use calls that are safe from inner-mode code. RTL calls are not safe from inner-mode code.

Can I just pin you down on your definition of an RTL? Is a call to SY\$GETUAI or SY\$PERSONA_CREATE in sys\$library:seureshp.exe a "RTL call"? So as not to try and trick you, when previously (couple of years ago?) I asked "Is it safe to call sys\$getuai from a UWSS?" your answer was "AFAIK No.". I am now of the opinion that what we're walking away with here is "It is safe to call other UWSS shareable images (including SECURESHRP) but not other user-mode RTLs." Is that right?

But if problems do arise and the report gets to OpenVMS Engineering, my input is going to be "not supported".

You can't get much plainer than that, and I appreciate your candour. But the image I'm stuck with is that of the poor developer's face who, after just spending a big chunk of his life reinventing a new heap-management wheel, stumbles across ACMS or Rdb (or PCA\$COLLECTOR) code that's called LIB\$GET_VM_PAGE from year dot and has had oodles of VMS engineering involvement. He's gutted!!!

Re: Is VMS Security being dumbed-down for Java?

(We need a witch-hunt to track down these RTL callers. I've heard they float! (Or are lighter than a duck :-)) Either way burn them!)

But if the VMS documentation set did not consistently refer to LIB\$GET_VM_PAGE being called from inner mode then I suspect that support and engineering would not be hounded about it for all eternity. Will these references be removed from the doc set?

also see the following documentation in the message files and in the documentation on writing User Written System Services.

NOSHRIMG, privileged shareable image cannot have outbound calls

When did it become legal to drop the /PROTECT qualifier on the \$LINK command? Are we agreed that a re-visit of the relevant documentation should have taken place at that time and, sadly, didn't?

obviously the kernel-mode C library, are safe to call.

Once again, can you please tell me the file-spec for this library? (And are the routines as well documented as the EXE\$ routines) Are there kernel mode equivalents of malloc, and realloc et al?

Inner-mode calls to anything other than exec-based system services are not generally considered supported.

Anyone got just *one* example of a non exec-based system service call being specifically supported?

Oh, and why does the image activator treat LIBRTL differently?

Thanks agin.

Cheers Richard Maher

PS, You also said: -

As for the question, you can only use inner-mode-safe calls. Pretty or not, like it or not, want it or not, you can only use calls that are safe from inner-mode code.

So when is VMS engineering going to provide a half-decent heap-manager that can be called from inner-mode(s)? May I humbly suggest that it moves up on

Re: Is VMS Security being dumbed-down for Java?

Re: Is VMS Security being dumbed-down for Java?

the priority list somewhere above the clamour to introduce the much needed /PAGE qualifier on the \$SHOW ZONE command? After the documentation has gone to such great lengths to try to stop people from using existing tools, it would be ludicrous to suggest it wasn't needed. (And that it hasn't been need for about 20 years)

"Hoff Hoffman" <hoff@xxxxxxxx> wrote in message [news:pxgUf.5109\\$TL3.2713@xxxxxxxxxxxxxxxxxxxxxx](mailto:news:pxgUf.5109$TL3.2713@xxxxxxxxxxxxxxxxxxxxxx)

In article <dvrhs6\$irf\$1@xxxxxxxxxxxxxxxxxxxxxx>, "Richard Maher"

<maher_rj@xxxxxxxxxxxxxxxxxxxxxx> writes:

:Let's look at LIB\$_EF. Even within the inner mode of a UWSS, would it

not

:be nice to support the convention of allocating event flags via

LIB\$GET_EF

:thus avoiding potential conflict with the User-Mode code.

Event flags are not a construct I prefer to use, having found these to be largely non-modular and exceedingly difficult to scale, and variously quite difficult to debug and control. As I have stated elsewhere with some regularity, I avoid event flags in all but the most trivial cases, having been badly burned by this construct across several projects. I now use EFN\$_ENF quite commonly, avoiding event flag zero, and I have been using ASTs and threads (either POSIX Threads or KP Threads), and bitflags, semaphores, locks or queues as required.

As for the question, you can only use inner-mode-safe calls. Pretty or not, like it or not, want it or not, you can only use calls that are safe from inner-mode code. RTL calls are not safe from inner-mode code.

As a general rule, system services are (usually) safe, and most any calls that use LIB\$ or SMG\$ or other calls can be quite problematic. I am aware of some applications that have "gotten lucky" here, too, and haven't seen problems. But if problems do arise and the report gets to OpenVMS Engineering, my input is going to be "not supported".

:> and known to have

:> various failures

:

:Please point out any that come to mind. (I'm not questioning they exist!

On

Re: Is VMS Security being dumbed-down for Java?

:the contrary, I know they're out there and could have missed many)

OpenVMS traditionally doesn't have to, need to, or even want to explain why some construct, platform or operation is unsupported. In this case, the particular case I've slammed into was page protection errors ---

pages

of memory within the heap end up owned by inner modes, and badness can then ensue --- maybe not at all, or maybe not immediately. I fully

expect

there are other cases of badness that can arise outside of the heap,

too.

To avoid Larry Kilgallen's circular reference to me at/via:

<http://h71000.www7.hp.com/wizard/swdev/ovms-shexe-cook.html>

also see the following documentation in the message files and in the documentation on writing User Written System Services.

NOSHRIMG, privileged shareable image cannot have outbound calls

Facility: SYSTEM, System Services

Explanation: Privileged shareable images, also known as user-written

system

services or protected images, cannot reference other

shareable

images. This check is made when the privileged shareable

image

is activated. If a reference to another image is detected, the entire activation is aborted and this error message is issued.

User Action: Rewrite the privileged shareable image so that it does not reference other shareable images.

Re: Is VMS Security being dumbed-down for Java?

and see:

http://h71000.www7.hp.com/doc/731FINAL/5841/5841pro_082.html

"As a protected image, your program does not have the entire operating system programming environment at its disposal. Unless a module has the prefix SYS\$ or EXE\$, you must avoid calling it from an inner mode. In particular, do not call LIB\$GET_VM or LIB\$RET_VM from an inner mode. You can call OpenVMS RMS routines from executive mode but not from kernel mode."

Hmmm. "lib\$ret_vm"? Looks like either I had a typo in what I gave the writer, or there was an editing error downstream. As of about five minutes ago, there's a report of that error logged. (That call should be lib\$free_vm, obviously.) Most sys\$ and exe\$ calls, and obviously the kernel-mode C library, are safe to call.

OpenVMS does not have a particularly good nor complete set of documentation for inner-mode coding, though I do try to cover that topic in a few of my presentations. The only major piece of documentation for this environment presently available is the driver manual -- that manual is, however, the central set of documentation for the supported inner-mode coding on OpenVMS.

----- #include

<rtfaq.h> -----

For additional, please see the OpenVMS FAQ --

www.hp.com/go/openvms/faq

----- pure personal

opinion -----

Hoff (Stephen) Hoffman OpenVMS Engineering hoff[0100]hp.com