

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

Source: <http://coding.derkeiler.com/Archive/Assembler/alt.lang.asm/2005-04/msg00508.html>

- *From:* "Beth" <BethStone21@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 16 Apr 2005 05:09:50 GMT
-

Randy wrote:

> Beth wrote:

>> This sounds more like a kind of "aliasing" thing than "lazy
> binding"...

>

> Okay, different concepts, then. Sorry, I've not been paying *real*
> close attention to what's being said in this thread.

Ah, it's okay...it does get confusing sometimes because either the terminology differs between platforms...or they do something "similar" but which isn't quite the same...or it is possible but is a very obscure feature that you had no idea was there because no-one ever uses it or mentions it (e.g. the PE file format does have some "delay-load import table", which is similar to "lazy binding"...but goodness knows what linker / linker option makes that actually work...or if any program has actually ever used it for anything)...

Anyway, I was initially wrong myself to say that it couldn't be done at all with Windows...the PE format does have support for something "similar" ("delay-load import table", section 5.8 of the PE spec ;)...but, you know, I've never seen any "linker options" or any discussion or any actual examples of this "feature" actually being used anywhere...hence my "presumption" that it was flatly not possible...rather, the "support" is there in the PE for it, but it seems not to be generally used by anyone for anything...

So, you know, though you mixed up "weak externals" with "lazy binding", you aren't weren't wrong that the PE allows something similar...you just pulled out the wrong name for it...and it's good that you did mention this because it made me actually look at the PE format specification to see that (among the more "obscure" section types offered) there is something there...

Indeed, you only got the name wrong...it's "delay-load", not "weak external"...you were actually right to say "are you sure?" to my assertion that it was completely absent from Windows...you made me actually look at the PE format and, yeah, there is "support" for it...even if I've never actually seen it used or mentioned anywhere else besides that PE specification, to have made me think it wasn't there anywhere...

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

>> "Lazy binding", in Linux, happens at run-time and is simply a case
>> of handling the "dynamic linking" process on a "per call" basis,
>> when that call is actually made...
>
> Isn't that just "load on demand"?

Basically, yes...but it's called "lazy binding" over here...

Also, not to be confused with "demand loading", which both OSes provide by way of the "paging mechanism" to only load those pages which are actually accessed...

As such, just to be "careful", it might be best to call it "bind on demand" or "late linking" or something like this, to differentiate it from the "demand loading" via paging...a facility both OSes provide (indeed, any OS with "paging" is likely to offer this, which doesn't even restrict it to being an x86 OS, as "paging" exists on other platforms...it's a natural "optimisation": Rather "load", you actually just "map" the file to memory and leave the pages "not present" ...if accessed, THEN you actually load them up from disk on a "per page" basis...any OS with paging – which includes both Windows and Linux – is almost certainly going to do things this way)...

It's complicated by the fact that, basically, the same exact "mechanism" is being used on more than one "level"...usually "demand loading" is referring to applying this "load it when first accessed" concept regard paging (and, thus, is "page-based")...

While this "lazy linking" / "late binding" / "delay-load importing" (so many names! ;) thing is more or less the same basic idea (only bother to load / link when first accessed) but applied on the DLL / .so "level" and is about the linking more than the "loading" part (indeed, because of "demand loading" from "paging", it won't strictly be "loaded"...it'll be "mapped" and then pages "loaded" as and when first accessed)...

So, yeah, this can be confusing because there's so many names thrown around for what is exactly the same...and many of these names sound similar but mean something entirely different ("weak linking"? "lazy linking"? It sounds like it should be the same, for sure ;)...and there are many "mechanisms" which, in fact, are more or less doing the same thing but doing it at a different "level" (e.g. the term "demand loading" usually refers to the use of "paging" to load pages on first access...on the other hand, "load on demand" would be a reasonable name to also call "lazy binding"...but, you know, "load on demand" and "demand loading" actually referring to different things? Oh dear...that's not a terribly good idea...so, as I say, it's probably best to "be careful" and emphasise that it's LINKING with this "lazy" stuff but "loading" with the page stuff ;)...)

> Again, I'm not an expert on Win32

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

- > DLLs, but I was under the impression that they provided a facility to
- > link a pointer to a stub routine that would load a DLL on demand and
- > patch the pointer thereafter (i.e., dynamic linking). Is this what
- > you're looking for?

Yeah – by whatever one six million names that people call it (even over on Linux, doing a Google search produces "lazy binding", "lazy linking", "late binding", "late linking" and a whole host of different names...and that's BEFORE we also start counting in the Windows' names like "delay-load import" ;) – THIS is at least THE IDEA we're talking about...and my last post did confirm – upon reviewing the PE file format – that this mechanism IS there with Windows (though, does anyone actually use it? And how to ask the linker to do it, as it does require a special "table" in the PE file, which would be the linker's job to construct and pop into your PE?)...

Note that this facility is NOT really that important, though...

The only reason it has turned up in this thread is because ("unless specified otherwise") LD does this kind of "lazy linking" by default...so, you know, as we're having "linking troubles" then T.M. was right to point out that we shouldn't forget that LD is using "lazy linking", if we didn't tell it to do otherwise...

It has a "consequence" on where and when the "error messages" would appear, you see...with "static linking", you'll get your "file not found" errors at LINK-TIME...with "(non-lazy) dynamic linking", you'll get your "file not found" at LOAD-TIME...and, if "lazy (dynamic) linking", then the "file not found" appears at RUN-TIME (when you "first access" any of that library's functions, which might happen anywhere – including potentially nowhere sometimes (you know, if that section of code is never executed because the user never selected the "option" which executes it ;) – in the program at any time whilst running ;)...

It's not really that we necessarily "want" this "lazy linking" (I'm dubious as to just how much use it could actually be...especially because – for the "plug-in" modular architecture we're suing – LuxAsm will be doing "manual dynamic loading" an awful lot, anyway...actually, more than that: LuxAsm will, by definition, also be immediately calling the "Init" function in that "module" straight after loading it, anyway)...it's just that this is what LD offers "by default" and, as T.M. was right to point out, we shouldn't be forgetting this while testing our code...

Indeed, when it comes to LuxAsm, we're going to have our own kind of linking, anyway...you know, to support the "modular architecture" idea that they can "plug-in" to the IDE or directly access one another's "services"...which actually, as the plan stands at the moment, will be a form of "stripped down COM", where the "modules" explicitly send each other "interfaces" (also known as, in its simplest term: "a jump table"...though, you might also call this a "virtual method table", if the procedures in the "jump table" accept a point to the jump table itself as a parameter

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

("this")...the LuxAsm "interfaces" we'll be using deliberately do not specify, unlike COM, whether these procedures do or do not have "this" parameters, so that, you know, either (or both) can be used, as is appropriate)...

>> You know, when Windows loads a DLL, it sorts out all the "dynamic linking" there and then during load-time...with "lazy binding", >> this is delayed until the symbol is actually called (which >> potentially means that if the code in the library is never >> referenced at run-time – say, a library is only accessed when >> the user chooses some menu option to run a specific function >> that uses that library (and no other part of the code uses it), >> >> True, but I was under the impression that Windows provides a facility >> to load the DLL when you first call the function in question. Does such >> a facility not exist?

In re-reviewing the PE specification, then, yes, this does exist...and I retracted my earlier assertion that it doesn't in the last post I made to the thread (somewhere around here ;)...

The difference, though, is that this is LD's "default", unless specified otherwise...which, as far as I know, it's the "other way around" with most Windows linkers, yes? That, by default, you don't get it unless you explicitly ask for it...

>> Ah, the names sound similar but "lazy binding" and "weak externals" >> aren't actually the same thing in this case..."weak externals" is >> a kind of "alias" mechanism (useful, as you mention, for, say, >> being able to alternatively link in some "integer only" library >> that "emulates" replacements for floating-point routines in a >> floating-point library...using an "aliasing" kind of mechanism)... >> while "lazy binding" – at regard Linux and LD, anyway – refers to >> an alternative method of "late linking"... >> >> Well, weak externals is slightly more than that, but you're essentially >> correct here.

Well, this was just a basic example...and I'm just going by what the PE file format specification itself says about how the PE's "weak externals" work...

>> Actually, looking through the PE file format document, there IS >> something slightly similar called a "Delay-load Import Table"... >> this delays actually loading a DLL until one of its functions >> are actually called...now, THAT corresponds to the "lazy >> binding" that LD provides by default... >> >> That's what I remember reading about (see above comments).

Yeah, as noted: You got the idea perfectly right, you just used the "wrong

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

name" there...which didn't help "locating" the correct section of the PE format specification that confirms it's there...but, luckily, I did stumble into – "delay-load import table" – while looking through that document to see that you were just "right idea, wrong name for it" here ;)...

[Although, there's so many names floating around that all superficially seem to sound alike: "weak linking" / "weak externals" / "weak binding" / "lazy linking" / "lazy binding" / "lazy linking" / "delay-load import table" / "import table" / "late linking" / "late binding" / "late loading" / "dynamic linking" / "run-time dynamic linking" / "lazy dynamic linking" / "lazy dynamic run-time binding" / etc., etc....some of which here are the same thing and some of which aren't the same thing...not helped when talking "cross platform", as every OS seems to like to conjure up its own "new name" for exactly the same thing ;)...

Well, what was that Arthur C. Clarke short story? Where if all the "many names of God" were all spoken then the world would come to an end? It seems, in the area of linking, they are attempting to "get started" on a similar quest to list all the names possible...I wonder what happens if they are all spoken? Does Windows come to an end? Now, if that were true, then they'd be a flood of people trying to compose that "exhaustive list" ;)]

>> Well, okay, Windows appears that it does have it...but LD does this
>> "lazy" thing by default, while, with Windows...well, does anyone out
>> there know how you get a DLL into the "delay-load import table"
>> instead of the usual plain old "import table"? Windows can do it
>> – I sit corrected – but, well, goodness knows HOW you get LINK.EXE
>> to set this up for you :)...
>
> Don't ask me. But the MS-Link manual is on-line on Webster. OTOH, it
> didn't have a whole lot to say about weak externals...

Well, this was why I initially thought that it just wasn't possible...because I've never heard of anyone actually doing it, never heard a discussion about it, never saw a single example employing it and, when I have looked through the (mostly sparse) documentation for MS's "LINK.EXE", I saw no "switch" or "option" that obviously seem to relate to any such function...

Hence my initial mistake of thinking it was completely absent...it isn't, in fact, the PE format "supports" the possibility...but, well, you know: What's the point of "support" in the PE, if every linker out there doesn't actually provide any "switch" or "option" to make use of it? Short of "hand coding" your PE headers, a la Herbert's code snippets, or something a bit "drastic" like that ;)...

>> Anyway, before I end, I've got to get the bad joke in, haven't I?
>>
>> "You are the weakest link...goodbye!"
>

> Arrggh!

Well, you know how the game works...statistically, you weren't the weakest link...but as the other contestants included Rene, Wannabee, Annie, Herbert and others, you were voted off "tactically"...even though you were the only one remembering to say "bank" before your question to bank the money! ;)

[Ah, it's one of the few British gameshows that was directly exported to America "as is" – including having "evil" Anne Robinson hosting it – that I know I can make this joke with at least some chance of you Americans having actually heard of the show...I don't know if they ever made any French / German / Swedish / etc. versions of the show, though...so I probably am leaving someone out here, who's got no idea – scratching their head – what on Earth the joke's all about...]

Mind you, if they need a replacement for Anne Robinson, then Annie seems a good choice...her name even sounds similar...and she'd have no problem with that whole "ridicule the contestants" section...

Beth :)

• *Follow-Ups:*

- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: Annie*

• *References:*

- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: T.M. Sommers*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: Frank Kotler*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: T.M. Sommers*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: Beth*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: Randall Hyde*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: Beth*
- ◆ ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
◇ *From: randyhyde*

- Prev by Date: ***Re: Why Bush?***
- Next by Date: ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
- Previous by thread: ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
- Next by thread: ***Re: Linux, X, ld, gcc, linking, shared libraries and stuff***
- Index(es):

Re: Linux, X, ld, gcc, linking, shared libraries and stuff

- ◆ Date
- ◆ Thread