RunAs Radio

http://www.runasradio.com

RunAs Radio is a weekly Internet Audio Talk Show for IT Professionals working with Microsoft products. The full range of IT topics is covered from a Microsoft-centric viewpoint.

Text Transcript of Show #117
(Transcription services provided by PWOP Productions)

Phil Peery On the Powers and Pitfalls of 64 Bit Operating Systems!
July 13, 2009

[Music]

Brandon Wenn: From runasradio.com, you're listening to RunAs Radio, the Internet audio talk show for IT professionals with Richard Campbell and Greg Hughes. This is Brandon Wenn, announcing show #117, with guest Phil Peery, recorded Thursday, July 2, 2009. RunAs Radio is produced each week by PWOP Productions, providing professional media and podcasting services online at pwop.com. You can follow the boys on Twitter at twitter.com/runasradio.

Richard Campbell: Thank you, Brandon. This is Richard Campbell. You're listening to RunAs Radio. With me as always, my co-host, Greg Hughes.

Greg Hughes: How are you doing?

Richard Campbell: I'm well and it's been a while since we got together. I know for the listeners we've just been in a steady stream of shows, but we got ahead of the game for a while there, didn't we?

Greg Hughes: That's right. We tend to do these things in waves as best we can.

Richard Campbell: Yeah and then all of a sudden, I think a month went by we didn't have to record any shows so I feel like I missed you, buddy.

Greg Hughes: Yeah, I missed you too, you know. We're not doing our regular, pretty regular set of recordings and calls. So for the people who don't know, we try to schedule guests in blocks because it's really the best way to produce the best shows.

Richard Campbell: Absolutely.

Greg Hughes: And it works out really well that way from a planning perspective and then we just release the shows every Wednesday.

Richard Campbell: And so now the coffers are empty and it's time to fill them back up again.

Greg Hughes: That's right. So here we go.

Richard Campbell: Yeah, let me introduce our guest. Phil Peery is a member of the Microsoft Premier Field Engineering group in the New York Metro area and has been with the company for 5 1/2 years. Phil is an Active Directory, Windows Platforms, and Performance engineer, and provides Microsoft customers with services such as ADRAP's, and teaches a number of Workshops including the Active Directory Troubleshooting and Windows 2008 Networking. Passionate about delivering the message on 64-bit Windows and Windows 2008, Phil has been instrumental in driving the development of Windows 2008 Chalk Talks, delivering the talks to many customers in the New York Area, showing the value and new features of the Windows 2008 platform. Welcome, Phil.

Phil Peery: Thanks guys.

Greg Hughes: Howdy.

Phil Peery: How are you doing?

Greg Hughes: What's ADRAP or Adrap, how do you say that, what is that?

Phil Peery: Produced AD-RAP. AD-RAP stands for Active Directory Risk Assessment Program.

Greg Hughes: Ah, okay.

Phil Peery: It's essentially a service that we deliver to our customers. It's a detailed analysis of a deployed running Active Directory, checking how it conforms to best practices, things like that.

Richard Campbell: Cool.

Greg Hughes: Really. I've not heard of that before but we'll have to talk about that sometime in the future.

Richard Campbell: That's it.

Greg Hughes: It's pretty darn interesting.

Richard Campbell: Speaking of shows, that's a show I'd like to know more about that as well, because today's show I think we want to focus on this whole 32-bit versus 64-bit operating systems, right.

Phil Peery: Okay. Yup, absolutely.

Richard Campbell: So the question number one is I should I be running anything other than 64-bit? Isn't 64-bit the way?

Phil Peery: Well, listen. My personal opinion is 64-bit is here, it's been here for awhile. Any new machine that you buy, from whether it be an Intel processor or an AMD processor, is 64-bit now.

Richard Campbell: Right.

Greg Hughes: Sure.

Phil Peery: AMD don't even make 32-bit processors anymore. So from the perspective of what you run, it's there, and there are substantial performance gains from 32-bit to the 64-bit space.

Richard Campbell: Just by running the other operating system, we're going to get a substantial gain?

Phil Peery: Sure, yeah. If you deploy a 32-bit version of Windows, whether it be 2003 or 2008, hopefully it's 2008...

Richard Campbell: Right.

Phil Peery: On a machine that is 64-bit capable and you take that same machine and you deploy the 64-bit version of the operating system with 64-bit drivers and full support for the 64-bit OS, the performance will be definitely on the order of 30% to 40% better than what you get from the 32-bit version of the OS.

Greg Hughes: So can you explain in layman's English, you know, like just sort of think of me as a really below average guy and explain to me why is that true? What are the real world reasons? Where do I really see the performance increases and where is it maybe kind of the same?

Phil Peery: So from the perspective of just the way the operating systems are designed, so a 32-bit OS basically deals with information and 32-bit kind of bunches.

Richard Campbell: Right.

Phil Peery: It's what the processor is designed to handle and the way the operating system is compiled to run. When you move from 32 bits to 64 bits, the code of the operating system as long as it's fully 64-bit capable, because you can't run 32-bit applications on a 64-bit Windows box, as long as everything is 64 bits across the board basically you're able to process twice the processor word data at any given clock cycle. So right from just the change from 32-bit to 64-bit hardware, you automatically are getting a larger amount of data that you can execute at any given time. From the perspective of memory and this is really, in my opinion, is probably where you gain the biggest bang for the buck, is with Windows 32-bit you basically have a 4-gig cap in the amount of addressable memory space that you have.

Greg Hughes: Right, yeah.

Phil Peery: You know 2 bits to the 32nd power; fuzzy math -- that's roughly 496, 4-gig. With 64 bits, the theoretical memory space is 16 exabytes. I mean, it's huge. Now you show me a machine that supports that much physical memory and I have a bridge that I can sell you in Brooklyn, but they just don't exist yet. But from the perspective of the virtualized memory space and what you have available to you there, the differences are huge. For example, in 32-bit Windows we basically divide the memory space up into 2 chunks. From zero to 2-gig is kernel space. That's private to the operating system. That's where we have the page pool and the non-page pool, memory pools that the operating system manages. From 2-gig to 4-gig is user mode memory space. This is where applications execute. This is where your notepad runs, where SQL Server runs, where Active Directory runs. When you go from 32-bit to 64-bit, that processor address space jumps from 2 gigabytes to 8 terabytes.

Greg Hughes: Slight difference there.

Phil Peery: So it's a huge amount of processing room for processors to run in. So specifically just from the memory perspective, you've got this huge amount of address space that you can now run your processes.

Greg Hughes: If you throw the hardware at it then you have that capability.

Phil Peery: Exatly. Well, that's the thing. You have to have a machine that supports that much physical memory.

Greg Hughes: Sure.

Phil Peery: I think right now you could get machines with hundreds and hundreds of gigabytes but nobody is producing a box that has that 16 exabytes memory in it which is too expensive I suppose.

Greg Hughes: But in the real world like, you know, you might think of it as, I don't know what, maybe you could call it a commodity level server in today's day and age, that if you had 16 gigs or 32 gigs of RAM, it sounds like you're going to be able to do a lot more with the 64-bit architecture than you can with the 32-bit architecture.

Phil Peery: Absolutely. Take case in point. An Active Directory domain controller.

Greg Hughes: Okay.

Phil Peery: In Active Directory, the main process that makes a domain controller -- a domain controller is Lsass.exe. This is the process that's responsible for handling network request to the database. It also caches the database and the memory, and basically it's managing Active Directory for you. In 32-bit Windows, if you use the standard memory model or your 2-gig and 2-gig, 2-gig for kernel and 2-gig for user mode, that basically means that the most Lsass can cache into memory at any one point, it's about a gig-and-a-half or so of Active Directory data. If your Active Directory database is

bigger than a gig-and-a-half, that means to satisfy any LDAP look up request, you're going to disk which as we know is way slower than if the data is cache in memory.

Greg Hughes: Sure, sure.

Phil Peery: So if you take that same environment and you put it on a 64-bit machine and you have lots and lots of physical memory available to get that Active Directory database cache into memory, Active Directory performance is greatly enhanced by that.

Greg Hughes: Interesting.

Phil Peery: We do have customers, we have a lot of customers that actually run with really big Active Directdory databases on 32-bit hardware and it is a performance issue.

Greg Hughes: So I mean I imagine there are a lot variables that go into how many users/groups or how many objects are in your directory, but is there sort of maybe just a big paintbrush kind of number that says, "If your organization is larger than X, then chances are you're going to benefit from doing this." I mean...

Phil Peery: Ultimately it comes down to how big physically the LDAP database is, the NTDS.DIT file. Like you said, there are some variables there, but generally if you boot your domain controllers without a switch called 3GB, which we will talk about in a minute...

Greg Hughes: Okay.

Phil Peery: So that you have a 3-gig user mode addressable memory space, you can cache the, Lsass process could cache up to about 1.4 or so gig of Active Directory data into memory, and the reason why it can't go to the full 2-gig is because the process has to have a little bit of headroom to actually process request. So it's going to use a little bit of that remaining memory to actively process all that request. Now, what you could do in some cases is you can use the 3GB switch on a domain controller with some caveats that basically what 3GB does is it takes that 4-gig address space and it divides it a little differently. Basically, what happens is you add 3GB to your Boot.ini and the kernel memory space, which is normally 2-gig, is reduced to one gig. So the kernel has to squeeze into a gig of memory.

Greg Hughes: Okay.

Phil Peery: That extra gig of RAM is given to user mode processing, and of course Lsass Active Directory is a user mode process. So we can now give Lsass a little more headroom to cache data into. We actually only recommend you do this under certain circumstances, and the circumstances are if you have a full physical 4-gig of memory in the domain controller and if your Active Directory database is bigger than a gig-and-a-half, and at that point if you decide to throw that switch and you meet that criteria, if you decide to put that switch in Boot.ini you basically are allowing Lsass to cache up to about 2-1/2-gig of Active Directory database. Now, what do you do – so here's the negative effect to using that 3GB switch, because there's a bit of a performance implication. When we reduce the kernel size down to a gig, one of the things that happens is the page pool and the non-page pool caches, which are memory caches that are in the kernel address space, are cut by about half. Within the kernel, there is a thing called the page table. The page table contains page table entries. Basically, what a page table entry does is it tracks every process that has a memory allocation that's currently active, and what that address space is is the physical hexadecimal address of the memory for that given process so that we know who's got what in memory. The reason why we have to do that is let's say you fire up notepad, you're going to go down the local computer store and you're going to buy some memory and hard drive, so you're typing up your shopping list. As soon as notepad fires up, he requests memory from the memory manager in order to store his process private data, your shopping list.

Greg Hughes: Right.

Phil Peery: When he does that request, a page table entry is created with the physical address of that page of memory that is being allocated to notepad. So notepad knows where his memory is, knows where his data is as a function of the page table entry. Let's say you type away, you type away and then you stop, you take a break and you go out to lunch or what have you, after a period of time there's a process called the working set tremor that is keeping track -- and this is a component of the memory manager, that is keeping track of what every process is doing in regard to accessing their active data, those pages of memory. If the notepad process stops actively accessing a page of memory for a given period of time, I believe it's about 10 minutes or so, what will happen is the memory manager will say, "Hey, notepad, you're not using the memory right now. The OS process needs that memory to cache Active Directory database into memory, I'm going to take that memory away from you and I'm going to put your data on the page file."

Greg Hughes: Got you.

Phil Peery: Actually, first it goes to something called the modified list. The modified list is

another area in memory that if you come back to notepad after this operation happens and we move to the modified list, we don't have to go all the way up to the page file to retrieve the data. It's a little bit faster. What happens is we move that data to the modified list and we update the page table entry to reflect that the data is no longer in main memory for that process. We have to keep track of where the data is. Once it sits in the modified list for a period of time and isn't ask for again or isn't touch, then it goes out to the page file. When it goes out to the page file, we again update the page table entry for notepad and we say this data is now on the page file. You come back from lunch and you start typing away on your shopping list again, you're going to add a laser printer or something to your shopping list, and notepad goes to access that page in memory and it's gone. It's no longer there. He goes, "Hey, memory manager, where is my data?" The memory manager goes to the page table entry list and says, "Wait a minute, notepad, your data is out on the page file. Let me go retrieve it for you." The memory manager goes and retrieves that page of data, puts it back into memory probably in a different physical location, updates the page table entry, tells notepad, "Hey, your data is here now," dismisses the exception that notepad sent to the memory manager and notepad is able to continue on to his merry way using that data. Now, here's why I go through this lengthy description of this. Just by that description you can see how important a page table entry is to the normal operation of Windows machine.

Greg Hughes: Sure.

Phil Peery: It's a critical part of the memory management of Windows. What happens is when you throw that 3GB switch, whether it be on a workstation or a domain controller or SQL Server or what have you, what happens is, like I said, the kernel memory gets reduced down to a gig. When we reduced that kernel memory down to a gig, we cut our non-page, you know, page pool or memory pool approximately in half. The data structure that holds our page table doesn't get reduced by half. It actually gets reduced much more aggressively, something in the order of like four times more aggressively. So what happens is if you have a Windows machine that boots and doesn't run anything with 2-gig of RAM in it, or 4-gig of RAM in it, you may have up to 200,000 free page table entries when that machine boots. That same machine with 3GB may start with only 40,000 page table entries.

Greg Hughes: Okay.

Phil Peery: And page table entries are part of every memory allocation. Now, what happens is if you go below 10,000 free page table entries, you're kind of getting into that warning zone. You're like that centerfield who's running the catch a fly ball and instead of keeping your eye on the centerfield bench, you're watching the ball coming at you. If you go below 8,000 free page table entries, your right foot had just hit the warning track in centerfield. If you go below 6,000 free page table entries, you've just run into the centerfield wall and basically you're out of the critical memory resource on that machine.

Greg Hughes: Right.

Phil Peery: And what we typically see is when you have page table exhaustion, what we see is we see operating system instability and sometimes the machine will just hang. This is one of the reasons why when you're all looking at 3GB, when I go to customers that are considering implementing it or have already implemented it, I always make sure that they have some kind of instrumentation on that machine that's monitoring their free page table entry...

Greg Hughes: To keep track of stuff.

Phil Peery: Because it's critical for that reason. Now, when you get to the 64-bit box, for page table entry address space you have 128-gig available to you. So it's very simply no longer an issue.

Greg Hughes: Right. So 64-bit architecture and a 64-bit version of Windows running, that really solves all the problems that we've been talking about for the last 10 minutes.

Phil Peery: Exactly and to be quite honest with you, the whole 3GB methodology, because of the huge address space in 64-bit Windows, simply doesn't exist. You don't have to do it because you've got all this memory.

Richard Campbell: Phil, the question I have here is is this only going to work if I have more memory? If I take exactly the same hardware, both with 4 gigs of RAM, run 32-bit versus 64-bit, don't I get any problems with 64-bit just because the pointers are all bigger and I don't have anymore memory to play with?

Phil Peery: Good question. Basically, bear in mind that Windows is a demand page virtual memory operating system. So when you boot a Windows box, a Windows 32-bit box, whether or not you have 256-meg of memory in it or 4-gig of memory in it, it's going to think that virtually it has 4-gig of memory in it.

Richard Campbell: Right.

Phil Peery: The same concept applies with 64-bit Windows. We think we have that whole

address space and because address space is virtualized and by using the page file we can make applications think that they have an 8-terabyte process address space, so any given process with 64-bit is going to think it has got that 8 terabytes of memory available and depending on what it needs to do from a processing perspective, the address ranges are going to all be virtualized and there will be a little bit of paging going on potentially, and speaking of the page file, one of the things that is also important is the more memory you get in Windows machine, whether it be 32-bit or 64-bit, the smaller a page file, you really technically gain.

Richard Campbell: Right.

Phil Peery: Because if you could get all of your data into physical memory and all your processes into physical memory...

Greg Hughes: Nothing to page out.

Phil Peery: Nothing has to page out, and what goes to the page file anyway, processed data.

Richard Campbell: Right.

Phil Peery: So going back to our notepad example, if you're typing away and you have to swap that data out for the page file that's notepad is on, does notpad.exe get paged out? No, it doesn't because it's already on disk in the form of EXE. So we don't actually have to page out executables in DLLs. We only page out processed private data when it needs to be, on demand.

Greg Hughes: Makes sense.

Phil Peery: So hopefully that answers your 32-bit versus 64-bit question in regards to memory and with machines that have the same amount of physical RAM.

Richard Campbell: And obviously the thing to do as soon as you get 64-bit is to add more memory...

Phil Peery: Yeah.

Richard Campbell: But I'm just concern that I'm not going to get much benefit if I don't add more memory.

Phil Peery: Exactly. There will be a benefit to it from a performance perspective.

Richard Campbell: Right.

Phil Peery: One of the things with Windows is it always runs better with more memory when you give it.

Richard Campbell: Yes.

Phil Peery: That's true with 32-bit, it's true with 64-bit.

Greg Hughes: Sure.

Phil Peery: Absolutely.

Richard Campbell: The 3GB switch worries me because once you get that pool down to 40,000 and you know that 10,000 is the threshold, that means that at a 25% mark you're in trouble already. It just seems like such a small number.

Phil Peery: Yes.

Richard Campbell: In what cases is this switch actually a good idea? Because if I have a program that's using that much memory, it's probably using handles as well.

Phil Peery: True. So there are some circumstances that it can be of benefit. For example, we talked a little bit about the Active Directory database size.

Richard Campbell: Right.

Phil Peery: So if your database is physically between 1.4 and 2.5 gig and that machine is only a domain controller, in other words it's not wearing multiple hats, it's not doing double and triple duty and you have four physical gig of memory in it for 32-bit domain controller, yeah, it can be helpful...

Richard Campbell: Right.

Phil Peery: Because our idea is to cache as much of the Active Directory database into physical memory as possible. That's the way we ensure the fastest access to LDAP. If you are less than1.4-gig, then 3GB is of no real value to you. If your database is bigger than 2.5-gig, then your answer is a 64-bit domain controller.

Richard Campbell: Right.

Greg Hughes: So take a look at the file and it will give you an idea.

Phil Peery: One of the things that I do like to recommend, and I think I've mentioned this once before, is if you do choose to throw out the 3GB switch, make sure that you're doing some monitoring of that available free page table entry counter in PerfMon.

Greg Hughes: Sure.

Phil Peery: Because you do have to keep an eye on it. One of the things I did, I did some benchmarking for one of our customers specifically on this topic and we had a 4-gig machine, we threw a 3GB in it and one of the things we did was we monitor different processes, and when their systems take back-ups kicked off we observed approximately 18,000 page table entries get consumed by the NT back-up process like that.

Richard Campbell: Ouch.

Phil Peery: Yeah, because what's the back-up process doing? It's caching stuff into memory to go either to a take or to a back-up file.

Richard Campbell: Right.

Phil Peery: It's allocating a lot of memory and boom, it ate up the page table entry list. So you do have to be cognizant of that particular counter. So other things that may benefit from 3GB is Exchange Servers, it love that extra gig of memory.

Richard Campbell: Right.

Greg Hughes: Right, right.

Phil Peery: So do SQL Servers. If you want to slow down your Windows XP or Windows Vista workstation, you can certainly put 3GB on those operating systems.

Greg Hughes: Yeah, I've tried that.

Phil Peery: But it's not going to buy you much.

Greg Hughes: Yeah.

Richard Campbell: No. It strikes me that this switch makes sense on single purpose machines because more programs running is going to eat up handles really quickly...

Phil Peery: Correct.

Richard Campbell: And it makes it a tightly monitored server. It's just doing Active Directory, it's just doing Exchange. But as soon as I talk about Exchange and SQL Server, I think the answer is 64-bit and 8 gigs of RAM at a minimum.

Phil Peery: Any one, yeah, exactly.

Greg Hughes: Well, you have to with Exchange now, right.

Phil Peery: Yeah. Actually, the new version of Exchange is 64-bit only, I believe.

Greg Hughes: Right.

Phil Peery: But not being an Exchange guy, you know... I believe there's a 32-bit version for testing purposes but the production version, I believe, is 64-bit. SQL Server, I think, we still do both 64 and 32-bit but I agree 100% if you're running a SQL box, if you're running an Exchange box, and if you're running an Active Directory domain controller, I always want to recommend 64-bit not only because of the memory benefits but also because also remember our IO drivers, networking disk drivers are now 64-bit as well so they also perform much better than their 32-bit counterparts as well. So along with getting the big memory space and the larger processor capability, the ability to process twice the processor word data at one clock cycle, you also get greatly improved disk and network IO. So there's, you know, win is almost all the way around here.

Richard Campbell: There's another side to this driver problem. One of the things is sometimes companies don't have 64-bit drivers.

Phil Peery: Exactly, yeah and for 64-bit Windows the drivers have to be 64-bit digitally signed in order to be able to be installed.

Richard Campbell: I have found that generally when they do have 64-bit drivers, they're pretty good quality because they do have to go through the signing process.

Phil Peery: That is correct, yeah. That definitely allows us to make sure that there's no memory leaks, that the drivers are compiled correctly, and all that.

Richard Campbell: Yeah, it's an interesting advantage in that respect, but like you said all the hardware runs 64-bit these days. The question is I think people get confused around whether 32-bit apps run well on 64-bit OSs or not. What has your experience been? I mean, obviously Active Directory, if you're in 64-bit it just runs in 64-bit.

Phil Peery: That's correct.

Richard Campbell: If you've got custom-built apps, you know, the stuff that's built with .NET and so forth, how well does it behave running on the 64-bit OS?

Phil Peery: Of course it's always going to be an application-by-application evaluation. You do have to do a little bit of testing first.

Richard Campbell: Right.

Phil Peery: But my experience both on my home network and out in the field with customers is that most 32-bit applications run very well with little or minor modifications to run properly on a 64-bit version of the OS and, again, performance is greatly enhanced. Now, there is the whole 64-bit to 32-bit funky process. When we run a 32-bit application, we do have to do the memory address space emulation.

Richard Campbell: Right. They call it Windows on Windows.

Phil Peery: Exactly, exactly and the initial start-up of the application might be impacted a little bit but it's still, in most cases, I found that most 32-bit applications run pretty well.

Richard Campbell: One of the things I've noticed, I'm throwing my developer hat on right now, in the .NET world is that default configuration for compiling a .NET app is mark as compile any which basically says if you're running on a 64-bit OS, run in 64-bit mode. If you're running a 32-bit, run a 32-bit which is very compatible.

Phil Peery: Right.

Richard Campbell: But often we haven't done any testing in 64-bit mode and so when you do that stuff can break and the minor modification to make is just to set the app to say always run as 32-bit, I don't care what operating system you're running on. Then our testing is all valid and it runs Windows on Windows mode and you don't have any compatibility problems that way.

Phil Peery: Right. The Windows on Windows process does a pretty good job of making sure that those 32-bit apps run well.

Richard Campbell: Absolutely and the main thing is just making sure you know you're running a 32-bit rather than running 64-bit. To me it seems silly to run in 64-bit mode if you're not going to address large chunks of memory. If the 32-bit space, if that 4-gig space is enough, make it a 32-bit app.

Phil Peery: Absolutely.

Richard Campbell: Of course, that means talking to developers about it, but just between us IT pros, it's a property setting and then recompile. It's a 5-second fix. I mean, talk about low hanging things, it just decrease grief, that's a good one.

Greg Hughes: Yeah.

Phil Peery: Sure, sure, absolutely and there's been a lot of work in ensuring that that compatibility is there and it is solid, it's solid, it works. I've never had a 32-bit application fail on my 64-bit OSs and I run a lot on 32-bit applications on a couple of my 64-bit machines in my home lab so I never had a problem.

Richard Campbell: Is there any question that we shall always run all our Hyper-V machines as 64-bit?

Phil Peery: Well, basically if you're running Hyper-V on 2008, that's your option, that's your only option. If you've got to run the Hyper-V server, it has to run on a 64-bit machine. It won't run on 32 bits. You can run a 32-bit guest OS but the Hyper-V box itself, the host machine had to be 64-bit.

Richard Campbell: Right. All right, Phil, any final words? We're about out of time.

Phil Peery: I know. I enjoyed talking to you guys. The one thing I'd like to say to all your listeners is kick the tires, 64-bit is here, it's good stuff and one of the things, if you want to do, contact your Microsoft camp to have a PFE come in and maybe do a demo.

Richard Campbell: Absolutely.

Greg Hughes: Yup.

Phil Peery: Yeah, that's what we're here for.

Richard Campbell: You bet.

Phil Peery: We're here to help.

Richard Campbell: Phil Peery, thanks so much for coming on the show.

Greg Hughes: Thanks man.

Phil Peery: Thank you guys. I appreciate it.

Richard Campbell: And we'll talk to you next week on RunAs Radio.