Wednesday, March 13. 2013

Performance

Brendan Gregg has written an interesting piece about finding performance problems: "The USE method addresses shortcomings in other commonly used methodologies".

It's a good paper, however ... well let's say, I don't understand why so many people find it especially cool or especially good, because at the end it isn't something really new. Don't understand me wrong: It's good. But not extraordinarily good. Like many methodologies it's basically just codified common sense with a personal spin. So I would prefer to say "My-personal-way-of-doing-stuff" instead of calling it methodology. There is nothing new in it. Just a lot of common sense.

I really think that performance analysis is not so much about a "methodology" you can simply follow that will lead you magically to a result. It's about a mindset how to tackle problems, it's about being structured in the approach, it's about "being prepared", it's a lot about knowing stuff.

As I do performance analysis quite frequently, I have created my own "methodology", or to be more correct ... my own mindset of doing such stuff. I don't call it method or methodologies. Perhaps it's useful for some or the other ... so i write it down here.
By the way - mostly for my german readers: I don't like the overused "-logie" as in "Methodologie". It's outright wrong. The english language does't know the difference between method and methodology in the same way as in the german language (you know, as Richard Porson stated, the life is to short to learn the german language), in german language the "Methodologie" is the science of methods, not a set of methods. Like "Technologie" is the science of "Technik" and not the "Technik" itself. When someones writes "Die Technologie des Flux Condenser", he isn't saying "The technique of the flux condenser" but "The science of the technique of the flux condenser". Perhaps it's derived from the phrase "the technology of the flux condenser". And I hate it even more, when I catch my self saying "Technologie" when I want to say "Technik". But I digress.

Knowledge and the streetlight effectBrendan is using the streetlight effect as an example of doing performance analysis. He is right, when you just know top you look at top, because it's the stuff you know. But essentially all methods I have seen in the past are essentially just that. Methods in performance analysis are maps of streetlight and a tutorial how to walk them. Good methodologies give you a better streetlight map and a better tutorial to use it.

But: There are just two ways you can deviate from this maps. Experience and knowledge. Both are hard to obtain. But there is no substitute for experience and knowledge, except more experience and knowledge. When you always walk the same way from the pub to home and you stumble always at the same place, you know by experience that you may have lost your keys there and you look at this place, even when there is no light. Bad ones give you a streetlight map, but no tutorial how to use it or a good tutorial, but no usable streetlight map.

It's the same with performance analysis. You have to have a lot of knowledge. When you give someone some command lines to check certain stuff in a system , you just erect more streetlights. At the end others just participate on your knowledge and experience because you know from both that this would be a nice place for a streetlight. But does this really help at the end? The nature is infinite in wisdom to find a location between the pub and your home without streetlight as well as nature is infinite in wisdom

So first pro tip: Learn. Learn as much as you can about the inner workings of the dead corpses aka slow systems you expect see in your performance analysis gigs. You don't get a coroner by reading "Pathology for Dummies", at least you don't start with it. You read Gray's Anatomy. Or when you want to analyse the corpses of dead or multi-morbid Solaris systems, start to learn as much about Solaris. My suggestion here is Solaris Internals: Solaris 10 and Open Solaris Kernel Architecture. The next book you should read after that is "Solaris Performance and Tools: DTrace and MDB Techniques for Solaris 10 and OpenSolaris". Both are a tad older, but they give you an good oversight. Don't read it when you have a performance problem, read it when you have time for it. The DTrace-book? Nice read, good read, but read it only if you have learned as much as possible about your application load? When you run an Oracle DB on your system, read a book about Oracle DB internals first before reading a so specialised book like the DTrace-Book. Especially as most problem (perhaps 90%) will show itself without using DTrace on your own.

One observation when doing performance presentation is that the knowledge part is really a problem. I was really

surprised to learn that many admins doesn't know what's the difference between an involuntary and a voluntary context switch? What cross calls really are? The difference between major and minor faults. When starting to do performance analysis you really have to know the basics. There is no way around it and no methodology will help.

And then - when you hunt down the really hard problems - there is a point where you need even more knowledge and even more knowledge: About the dependencies of the layers, some visible and obvious. Small caches inside the application, high amount of physical accesses.some subtle and very indirect. Like the dependencies of processor sets and garbage collection in Java. Like Out of this reason finding performance problems is often a team sport. Or you should have some good experts in the background: A DBA that knows a little bit about the system, and a Sysadmin that knows a little bit about the DB are most often a really nice combination and most often you learn a lot of stuff in oder to do such performance diagnosis gigs on your own.

From top to downNext pro tip: Start as highest as possible in the stack: Yeah ... i know it's fscking cool to say to your fellow sys admin colleagues "I shaved of 0.1 ms of 0.5 ms disk access time by some arcane tuning". But when you shave away 900 of 1000 disk accesses by ensuring that your Oracle Database is choosing a proper query execution plan, you did a much better job. The higher you get in the stack, the easier it gets to get larger savings in resource usage and large increases in performance.

When one of the departments of your company storms into your department that one part of their application is running slow, it makes really no sense to analyse kstat, vmstat, iostat or something like that ... looking into an AWR report is a much better place. And perhaps it that is even to low in the stack. If your application delivers decent usage statistics, search there first. And then work down the stack when you don't find any anomalies in the higher layers of the stack.

From client to server
Even more important, work your way in from the client (which may be another server) to the server. Don't start to analyse your server, before you haven't found out that there may be a problem in the interconnecting network, or a problem with the client.

Doing a ping to check the network status at first is no a bad idea because it's nonsense in itself, it's just a bad idea because it doesn't measure network latency correctly. A ping is a good idea, because it answers one question directly: Is there at least a basic network connection between two nodes. And when a ping doesn't answer that answered in the past, you have a start.

House M.D. commemoration derivation
The next tip is based on the article of Brendan, he describes it as one of the anti-pattern, the "blame-someone-else"-pattern. But the first derivation of this pattern is more important to you, i will call it the "House M.D. commemoration derivation". Everybody lies. Always assume, that other groups are using the anti-pattern, never assume that the analysis of other groups is correct. Check it! I can't count the number of situation where i finally found the problem in an area that was excluded by "No, we checked that by experts. Everything is okay there." at the beginning. And i hate it, that I almost can not count the number of situations as well, where I tended to believe the customers it and searched in a different area. I just know that the number of such occasions was reducing with the time and experience.

Question everything!The anti pattern and the first derivation can be distilled down to: At the beginning there are no innocents in performance analysis. Of course you can round up the usual suspects at first to show some action and perhaps save some time by a lucky hit. But at the end you have to check everything.

An additional pro tip: When a customer tells you that two systems are identically configured, do not believe it. Check it!

Some time ago I ran into two systems a customer was thinking, they should be almost identical and thus expected similar behaviour. However especially this customer had changed several technologies and just configured them in a way that they thought it would be similar. The problem is: The knowledge of systems shows some gaps. So sometimes the assessment of "similarity" is somewhat faulted already in it's foundation.

For example Direct I/O. 2013 and still some people are not aware at all, any many people of all the implications of direct I/O. Recently i had a customer that said to me "Both systems are identical, they should deliver similar performance" ... and then you find out that one system is using a filesystem without the equivalent of Direct I/O properly configured and one system with properly configured Direct I/O. Thus the available cache sizes of the old system were significantly larger and thus the assumption of the customer of "systems are comparable" just took a large hit. A system with 2 GB with Direct I/O and a system with 2G SGA with 32 GB file system cache are not equal.

History repeating?So, you are in the middle of analysing a system. You look at all the parameters on the system, all the numbers the stat tools are reporting. But there are few numbers that that are really inherently bad when you see them (like scan rate in vmstat larger than 1), most of them are just part of doing business. Where should you start looking? I found it to be most effective to search for history repeating, or - to be exact - history not repeating. When you get to a system, that worked well in the past but is now working sluggish, the first question is what has changed: Is the load pattern different? Is there something that wasn't there in the past? Like an sql statement not used before? Like a different number of processes?

Pro tip: Prepare for the situation when the sky falls down and the CEO is sitting behind you by recording the characteristics of known good performance. Use dimstat, use GUDS or at the end just pipe and hour of all *stat tools you can think of into a file. Be sure to keep the raw data and and not over processed and over compressed data. With Oracle DB keep an AWR report in HTML form somewhere in a safe location.

It's really easier to find a start in your performance diagnosis, when you know what has changed, what's normal and whats not.

VisualizeMost humans i know are much better at detecting patterns in graphics than in long log files. As soon as you have a log file for example on execution times, try to filter out the numbers in there, put them into Excel or R or whatever you prefer to do charts. There is a reason why many customers really love Analytics on the S7000 series.

Pro tip: You will see patterns in it more easily when having a chart. And when you see patterns you can search for them in lists (seeing 10 peaks with a certain size in a certain time frame is a good start to search for the occurrence of 10 peaks in other lists).

Cause and effectIt's important to separate between cause and effect! You see large service times on the disks holding your slow database. But are the service times really the cause for the slowness, or just an effect. The cause could be a suboptimal query plan, a missing index, that leads to more disk accesses leading to overload on your disk system leading to large service times. You have to ask your self "Why is it this way?" until you are sure beyond reasonable doubt that there is no further "Why?". Otherwise you iron out an effect without touching the cause. The problem will come back for sure and your suggestion of using more disks will haunt you.

LightspeedI just write about this because i run into the need to explain this quite often: Yes, it's possible to have a performance problem without seeing a component overloaded. When your large disk array delivers for a certain load data in 1 ms from request send to reply received, you can do 1000 requests per second in one thread doing stuff sequential. When the round trip time between one node and another is 1 ms on you network, you can to 1000 transactions per second on it within a single thread. Come hell or high water. Your CPUs won't be loaded, you disk array is totally bored, tumbleweed is blown through your SAN, your LAN is playing harp under a tree. However your performance is still limited. When diagnosing performance problem never exclude latency because of execution times, the speed a head can move inside a hard disk and the effect of the universal limit of light speed. There is a reason why some companies with deep pockets can't wait that some carriers offer a fibre from london to new york that exactly follows the great circle instead the easiest way or from new york to tokio via the north-west or north-east passage through arctic waters to shave some times of their network latencies.

But please: When measuring latency, please use a tool that is just measuring that. Ping is a bad tool to measure network latency, because the round trip time measured in a ping is a amalgam of a lot of tasks that in a part has nothing to do with the process of moving packets.

ConclusionOkay, i wrote a long article about my way of doing things when finding performance problems. Hey ... i could call all this stuff "Moellenkamp Unbelievable Performance Problem Estimation Technology".

Posted by Joerg Moellenkamp in English, Solaris, The IT Business at 14:18

test
Anonymous on Mar 14 2013, 09:04

Nice article. But Muppet? Who are you, Statler or Waldorf?
Or wait: Dr. Bunsen Honeydew?
Anonymous on Mar 14 2013, 14:24

I'm worse than Statler and Waldorf together
Anonymous on Mar 14 2013, 14:32

I really love your blog. You write about very interesting things. Thanks for all your tips and information.
Anonymous on Mar 14 2013, 16:27

Just one word of caution regarding the Oracle AWR. It's part of the Diagnostic Pack, so either have it licensed or fall back to the older stats pack reports.
Anonymous on Mar 18 2013, 14:07

Yes, you are right In such situations all and any historic information helps ... STATSPACK,AWR, snapper outputs, old execution plans of your mostly executed sql commands
Anonymous on Mar 18 2013, 15:49