Troubleshooting modern Linux in the cloud

Martin Bach

09/2017

Abstract

This article is intended to provide an outlook to the upcoming presentation "Troubleshooting modern Linux in the cloud", to be held during the annual DOAG conference in November.

1 Linux in the context of Operating Systems for the Oracle Database

The Linux ecosystem has come a long way from modest beginnings to contributing to all major compute environments both on premises and on the cloud. No other operating environment sees such rapid development, and there is none to the author's knowledge that runs on such a variety of platforms. Linux supports almost all hardware, from the smallest embedded systems to some of the largest clusters that have been ever assembled.

One advantage of the Linux system as a platform is its openness. There are sheer endless numbers of tools for everything there is to do with a system. This can both be a blessing and a curse. When it comes to profiling and tracing applications for the Linux platform this is particularly noticeable. Having played catch-up with other platforms, with Solaris as the benchmark, according to some Linux is now in a position to finally surpass the benchmark and lead the field. On the downside, there are far too many tracers out there, and almost all are still used and see development.

When it comes to running the Oracle database, there are few platforms still available today, and their number is definitely not increasing. Using Linux on x86-64 architecture appears to be a safe bet: performance of the architecture is formidable, and for quite some time it is possible to ramp up the DRAM capacity of servers in a way that was possible only on high-end UNIX platforms.

For the purpose of this article it does not matter too much whether the Linux system you are running is hosted on-premises, or in the cloud. There are some details worth knowing when it comes to virtualised Linux and tracing; these will be pointed out where applicable.

2 A word about licensing

Many performance troubleshooting tools the Oracle database offers require separate licenses. Before applying any of the techniques in this article and its corresponding presentation during the conference to your environment ensure that you are appropriately licensed!

3 Gross simplification of performance analysis on the Oracle RDBMS

Thankfully for everyone working with the Oracle database it is one of the the best instrumented pieces of enterprise software to the author's best of knowledge. The instrumentation can broadly be categorised into 2 areas, both provided by the database itself:

- The Wait Interface
- Session counters

There is a third level of instrumentation-operating system profiling and tracing- which will be covered later.

3.1 The Wait Interface

When the Oracle Wait Interface has been integrated with the software one of the principle goals of software engineering could be fulfilled: measuring performance of applications written for the Oracle database. Without information provided by the Wait Interface performance analysts had very little data available to do their jobs properly. Somewhat odd metrics from back in the day are still circulating discussion forums and even the official documentation! Take the infamous buffer cache hit ratio for example: they were the straws one had to clasp to before the introduction of the Wait Interface.

In the interest of a scientific approach to problem solving it is greatly appreciated that the early days are over.

The Wait Interface provides details about ongoing operations in many places: V$- Views, (extended) SQL Trace and more recently in Active Session History (ASH) and its persisted form, the Automatic Workload Repository (AWR).

Using the Wait Interface, many problems can be solved, potentially contributing to the majority of issues that can be encountered with the Oracle database.

3.2 The Wait Interface sometimes isn't enough

There are situations when the information provided by the Wait Interface is not enough. Sometimes more details are required for which the Wait Interface does not provide sufficient data. Session statistics as exposed by tools such as Tanel Poder's Session Snapper or Adrian Billington's mystats package (to name just a few) can provide more insights.

It is usually advantageous to combine the output of both Waits and session counters unless it is known beforehand that only session counters should be investigated. To give you an idea about the wealth of instrumentation, consider the following:

The Wait Interface in 12.2.0.1.170814 shows a total of 1811 waits grouped into 14 classes.

```
SQL> select count(*), wait_class from v$event_name
  2  group by rollup(wait_class);

  COUNT(*) WAIT_CLASS
---------- ----------------------------------------------------------------
        57 Administrative
        17 Application
        68 Cluster
         4 Commit
        49 Concurrency
        27 Configuration
       135 Idle
        29 Network
```

```
      1314 Other
         9 Queueing
        10 Scheduler
        35 System I/O
        57 User I/O
      1811

14 rows selected.
```

Additionally, there are 1804 performance counters Oracle tracks per session.

```
SQL> select count(*) from v$statname;

  COUNT(*)
----------
      1804
```

You can see for yourself how these relate by starting a full scan on a reasonably large segment in one session, and sampling its activity with Tanel Poder's session snapper in another session. You are going to see lots of details about the actual physical and logical I/O operations, blocks fetched, and many more that you might not have been aware of previously.

Sadly the majority of statistics are not documented, those that are can be found in Appendix E of the Database Reference Guide.

The combined information provided by the Wait Interface and session counters is enough detail for most occasions.

3.3 Using the operating system to diagnose performance issues

In the rare situation where Waits and session counters aren't sufficient to solve the performance puzzle, the operating system might help. Very often though you are in a position where your hands are tied and the issue you see is a problem with the database. Oracle Support is quite likely the only one to help. Understanding the bottleneck on the other hand and being able to steer support into the right direction can be very rewarding though.

If you think about it, the Oracle database is "just an application" making use of services offered by the operating system. And the work performed by the operating system can be observed. You should be warned though: attaching to a process, especially one that is in trouble, can have very undesired consequences. Some tracing methods are so intrusive they take the process down. This might also bring the database down!

Over time a number of options have become available to the performance analyst: basic tools such as strace are widely known and can provide a good first overview of the problem.

There has been a lot of research involving Linux tracing tools such as systemtap and perf in troubleshooting the most tricky Oracle performance problems. Other popular tracers include ftrace and most recently, eBPF. Solaris DTrace has been ported to Linux as well, but it does not feature the same number of probes as on Solaris, where it originates.

In the author's opinion Systemtap suffers from an inherent issue in that it is largely based on inserting a loadable kernel model into a system. Many administrators might not like the idea.

The ftrace tool looks quite interesting, but it's interface is difficult to master for the casual performance analyst/DBA.

The perf tool on the other hand appears reasonably easy to use, and there are a great many examples out there, including ways to visualise performance data using flame graphs. Due to its widespread availability and adoption, perf appears to be one of the winners for the contest of the most popular tracer, at least for now.

Most recently enhancements have been made to BPF, or Berkeley Packet Filters allowing it to perform work similar to DTrace. At the moment eBPF tracing, although very powerful, is hard to use. Most performance analysts are best off using the examples and tools provided on the Internet. In addition the use of eBPF requires a very modern kernel.

4 Outlook

The presentation during the annual conference aims at refreshing the audience's memory about both Wait Interface and session counters before digging deeper and explaining important Linux tracers.

The focus is on practical use cases and demonstrations to help everyone attending understand Oracle performance when Wait Interface and session counters both cannot provide the information needed to work on an issue.