This paper examines how to measure and optimize the performance of a SolarWinds Orion installation.
Copyright © 1995-2012 SolarWinds Worldwide, LLC. All rights reserved worldwide.

No part of this document may be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the written consent of SolarWinds. All right, title, and interest in and to the software and documentation are and shall remain the exclusive property of SolarWinds and its respective licensors.

SOLARWINDS DISCLAIMS ALL WARRANTIES, CONDITIONS OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON SOFTWARE AND DOCUMENTATION FURNISHED HEREUNDER INCLUDING WITHOUT LIMITATION THE WARRANTIES OF DESIGN, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. IN NO EVENT SHALL SOLARWINDS, ITS SUPPLIERS, NOR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY EVEN IF SOLARWINDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The SolarWinds, the SolarWinds & Design, ipMonitor, LANsurveyor, Orion, and other SolarWinds marks, identified on the SolarWinds website, as updated from SolarWinds from time to time and incorporated herein, are registered with the U.S. Patent and Trademark Office and may be registered or pending registration in other countries. All other SolarWinds trademarks may be common law marks or registered or pending registration in the United States or in other countries. All other trademarks or registered trademarks contained and/or mentioned herein are used for identification purposes only and may be trademarks or registered trademarks of their respective companies. Microsoft®, Windows®, and SQL Server® are registered trademarks of Microsoft Corporation in the United States and/or other countries.

Document Revised: 07/05/2012 (AJM)
Orion Component Overview

Orion installations can be divided into 3 components, the Orion Core and application server, the MS SQL database, and the database storage. These components constantly interact to perform the 3 primary functions of an NMS, data retrieval, data storage, and information presentation. This is depicted below.

Each component is dependent on the other components and their functions. For example, in order to view or report on monitored elements, the data must be available from data retrieval, properly stored, and retrieved to the web in a timely manner. If any one of these steps fails, or performs poorly, the desired result will be impacted. The level of impact depends on the performance and availability of all the functions involved. As the performance of any particular function decreases, the level of impact will increase to a point where all the components have appeared to fail. Because of these component interdependencies, it can be difficult to determine where the problem lies and how to fix it. This paper will focus on the common causes of poor performance, and provide steps to isolate and rectify the issue.

SQL data storage is central to most functions of Orion, and so a poorly performing SQL server will greatly affect the user experience. For example, the below graph depicts the impact to Orion functions caused by a poorly performing SQL server.
Web and Report Loading as a Function of SQL Performance

As seen in the above, as SQL performance degrades a variety of symptoms can result, from slow loading to complete failure. As the symptoms worsen it can appear that other Orion functions, such as the web engine have failed. Making assumptions about the cause of performance issues can cause wasted time examining components that are performing well. While the root cause of the web failing to load may look like a web failure, in the above, the actual cause is poor SQL performance. For this reason, when a failure occurs it is recommended to follow a step-by-step, documented troubleshooting procedure to isolate the cause of a performance issue.

Performance should be thought of as having a faster-the-better overall goal; performance improvements can always be implemented even though the system runs according to user expectations. Performance is not measured simply as good/bad, but as having varying degrees. Using this approach to performance has many benefits, including:

- Cost avoidance – When the NPM system is optimized, more elements can be added than would be possible in a nominally performing system.
- Outage avoidance – Keeping ahead of performance issues will help to eliminate outages in the Orion system caused by a resource overload.
- More accurate planning – When you know what the system is capable of, you can properly plan for growth without surprises.
- Ease of use – While the Orion web interface is very intuitive, any system experiencing slowness degrades the user experience. Keeping the web interface responsive helps you find and isolate network issues easily and quickly.
Managing Orion Component Performance

The overall performance of Orion is dependent on the performance of each of the components. Therefore, accurately measuring Orion performance involves measuring the performance of the components as well as the Orion Web Console responsiveness. MS SQL performance is involved in most Orion performance issues, so that is a logical place to begin.

**SQL Performance - Measuring and Monitoring a Production Server**

For SQL 2005 servers, Microsoft makes the performance counter accessible through the MS Server Performance Monitor in Administrative Tools menu. In SQL 2008 servers the tool is called Reliability and Performance Monitor. The tool can be launched from either version using the perfmon (perfmon.exe) command. This tool makes it easy to locate and observe the counters you will use to measure SQL server performance.

In the Orion environment, the single most important SQL server performance metric is disk queue length. Queue length is a measurement of the SQL writes that are waiting to be written to disk. When disk write queues start lengthening and there is a steady load on the SQL writes, the queues may snowball to the extent that write requests get dropped. This leads to gaps in Orion data and will affect the overall performance of the SQL server and Orion.

A good rule of thumb is that disk queue length should not exceed 2 times the number of effective spindles in the SQL storage array. The effective spindle count is the number of striped spindles. For a RAID 10 direct attached storage unit with 8 total disks the effective spindle count is 4. 4 of the spindles in this array are the primary striped array and the other 4 are secondary striped mirrors of the 4 primary spindles. Since no performance gain is achieved by mirroring disks, only the primary striped set is used to measure performance.

Keeping in mind that this is a rule of thumb, and that performance is a continuum rather than good/bad assessment, the recommendation does not mean that if you have 5 effective spindles and you peak the queue at fifteen, you will have performance issues. Also this number should not be used to determine the number of spindles required to lower the queue length. This is because the effect of queue length on performance is not linear. Adding 1 effective spindle to a 4 effective spindle array may lower the queue length by fifty percent or more.
The below table provides some other important SQL performance counters and their target ranges.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
<th>Target Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Server: Access Methods - Page splits/sec</td>
<td>Index or data page is split and partially allocated to a new page. May cause IO issues such as excessive disk queue length. Can be addressed by increasing SQL fill factor.</td>
<td>The fewer the better. A static page splits number with low disk queue length indicates this is not an issue.</td>
</tr>
<tr>
<td>SQL Server: Access Methods – Full Scans/sec</td>
<td>Table index failures. When table indexes are not used the result is a disk operation that could have been avoided. High full scan rates will impact transaction rates and overall SQL performance.</td>
<td>The lower the better. Near 0 rates are to be expected for a well performing SQL server.</td>
</tr>
<tr>
<td>SQL Server: SQL Statistics - Batch Requests/Sec</td>
<td>A measurement of transaction rate. High batch requests may cause a heavy CPU load.</td>
<td>Static rate. Usual rates are in the thousands and are not a problem unless CPU utilization is too high. Monitor for a slow creep up or any sudden jump that does not recover to normal levels.</td>
</tr>
<tr>
<td>SQL Server: Locks: Dead locks/sec</td>
<td>Dead lock rate.</td>
<td>None - Dead locks rates higher than 0 should be investigated using MS SQL tools.</td>
</tr>
<tr>
<td>SQL Server: Buffer Manager – Buffer Cache hit ratio</td>
<td>The percent of requests that can be served from cache rather than disk. Although writes are more important than reads in an Orion environment, this metric should be tracked as an indicator of memory issues.</td>
<td>The more, the better.</td>
</tr>
<tr>
<td>Physical disk – Reads/sec, writes/sec</td>
<td>The read and write rates. This should be measured for logs and data. Because these measure these actual read and write rates they are not an indicator of the maximum I/O rates unless they reach a level where disk queue lengths also increase.</td>
<td>Static levels.</td>
</tr>
</tbody>
</table>

While this is not a complete list of the metrics you may wish to monitor, these do a good job of detecting bottlenecks for disk I/O, memory and CPU. The thwack.com user community has posted several times
Along with monitoring the above SQL counter metrics, some basic metrics for the SQL server should also be monitored. These include the following:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database size</td>
<td>Total size of the Orion database. Normally less than 20GB for NPM with moderate syslog data and no NetFlow data. NetFlow can increase the database size sever hundreds of GB. Larger databases require more disks, memory and CPU resources. Large numbers of syslog messages will also increase the database size.</td>
<td>Balance data retention settings with database size and performance.</td>
</tr>
<tr>
<td>SQL Server Memory</td>
<td>The amount of RAM used by SQL.</td>
<td>Increase the minimum SQL memory setting to allocate eighty to ninety percent of the total memory to SQL.</td>
</tr>
<tr>
<td>SQL Server CPU</td>
<td>CPU utilization</td>
<td>Keep the average CPU utilization as low as possible. CPU peaks of one hundred percent are acceptable as long as the average stays low and the peaks are brief.</td>
</tr>
<tr>
<td>AWE Memory</td>
<td>AWE memory allow for more efficient use of memory on 32-bit SQL.</td>
<td>Enable AWE for all 32-bit SQL. See the Microsoft SMDN Library for details.</td>
</tr>
</tbody>
</table>

Determining SQL Server Performance Abilities

The above section is used for measuring the performance of the SQL server in operation. You may still want to know the maximum rates that you can expect from your system; this will give you a sense of when your server is reaching its limits. RAM and CPUs operate at predetermined speeds, but storage subsystem performance is dependent on several factors. Several tools can be used to assess the maximum performance of your storage subsystem. Perhaps the 2 most common tools are Microsoft’s SQLIO (free) and IO Meter (shareware). These tools give you 2 crucial measurements for your subsystem, the read/write speeds and the disk queue lengths. They will also tell you the maximum throughput rates in MB/s. These tests should be performed on new subsystems in a lab environment, not against an active production server. In general, a well performing subsystem should have disk write speeds of 1 to 5 ms or better for log volumes and 20 ms or better for data volumes. These volumes are discussed in the next section.
**Optimizing SQL Server for Orion**

Once you understand the important metrics to measure, you can address areas in SQL to maximize performance. Again, as SQL is very storage intensive, this section will focus a great deal on SQL storage enhancement. Here are some of the basic recommendations for using SQL with Orion. Many of these can be found in the Administrator’s Guide for NPM.

<table>
<thead>
<tr>
<th>Item</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dedicated SQL Server</td>
<td>Except for small NPM installations (SL250 or lower) without NTA we recommend a dedicated hardware server be used.</td>
</tr>
<tr>
<td>RAID Array</td>
<td>RAID 10. The more drives the faster the maximum read and write rates. RAID 1 alone offers no performance increase over a single spindle. Do not use RAID 5 or 6.</td>
</tr>
<tr>
<td>Hardware RAID Controller</td>
<td>The RAID controller should always be hardware based and server quality.</td>
</tr>
<tr>
<td>Disk Speed</td>
<td>15 KRPM (15K). While the cost of 15K drives is higher, this will partially be offset by the need for fewer spindles.</td>
</tr>
<tr>
<td>SQL Version</td>
<td>SQL 2005 and 2008 are supported. 64-bit SQL should be used for all Orion installations over SL 250 or smaller Orion license levels with NetFlow.</td>
</tr>
<tr>
<td>RAM</td>
<td>SL 500 and below – 2GB, SL 2000, 5GB+, SLX – 20GB+. The more RAM the better. For SL 2000 and above set the minimum SQL memory to eighty to ninety percent of the total server memory, but allow sufficient total memory outside this setting for the OS.</td>
</tr>
<tr>
<td>CPU</td>
<td>3 GHz, dual core processor or better. This will work for all size licenses. Do not use less than a 2 GHz processor for any Orion SQL server.</td>
</tr>
</tbody>
</table>

You might wonder, is RAID 01 is a good choice for your SQL database storage? RAID 01 and RAID 10 both yield the same number of effective spindles for any given even number of drives greater than 4. RAID 10 is actually shorthand for RAID 1+0, and RAID 01 is actually RAID 0+1. If you think of the RAID arrays in this way the difference is easier to understand. RAID 1 is mirrored drives. Raid 0 is striped drives. For a RAID 10, 4 drive array the 2 primary drives are mirrored with the 2 secondary drives, then the mirrored pairs are striped to primary and secondary striped sets. With Raid 01 the drives are striped in 2, 2 drive sets, and then those sets are mirrored to the secondary 2 striped drives.

The important thing to remember is that in RAID 10 you can lose one or more drives out of either striped set and the mirror of that drive will continue. In RAID 01 if you lose one drive from a striped set the set is dead. If you lose one drive from both sets, the storage array is dead.

This is shown in the below graphic.
SQL server can be further optimized using several methods discussed below.

**Enable Advanced Windows Extension (AWE) for 32-bit SQL.**
AWE allows MS SQL enhanced access to memory under 4GB. Users enabling AWE may see dramatic improvements in MS SQL 32-bit server performance. AWE is not used for 64-bit servers. Enabling AWE depends on the version of Windows server you are running.

**Separate Drives for Orion, SQL log (temp) and SQL data**
Separating the temp files from the data files can help improve performance. A typical installation of this type will look like this:

- C: drive – Orion and OS. 2 drive RAID 1
- E: drive – SQL log files. 4 15K drives RAID 0
- F: drive – SQL data (Orion data). 6 15K drives RAID 10

The size of the drive space for each of the SQL drives can be determined by examining the current SQL database files.

**Using RAMDisk® for SQL log (temp)**
RAMDisk, a third party software package, allows you to place the temp SQL files onto a logical drive that exists completely in RAM. This requires 64-bit SQL and a good amount of RAM. See the RAMDisk documentation for further requirements. This tool is very useful as it takes the most performance intensive part of SQL storage and moves them from physical disk performance levels to RAM performance levels. Physical drive (spindle) IOPS are measured in the hundreds per second where RAM IOPS are measured in the hundreds of thousands per second.

**Solid State Drives (SSDs)**
Solid state drives yield the IOP levels seen in RAM drives. These are becoming more common as the prices continue to drop. Unlike RAM drives, SSDs are non volatile, so the data written to SSDs remains after the system is powered down. SSDs are made of billions of microscopic transistors which store individual data bits by being in an uncharged state or a charged state. This works well for program files and data that is static. One issue with most SSDs is that the barrier that holds the charge within each transistor breaks down upon successive writes. When the transistor changes from a 1 to a 0 or a 0 to a 1 the charge state of the transistor changes and some erosion of the charge barrier occurs. After many write cycles, the transistor looses the ability to hold a charge and becomes useless for storage. SQL is very read/write intensive and so creates challenges for SSDs. New technologies aimed at maximizing SSD life span are coming onto the market. These tools can help high-end SSD arrays reach life expectations up to 8 to 10 years.

**Orion Server Performance**
Beginning with the NPM 10.2 release (Fall 2011) and including all products built on the Orion platform, SolarWinds optimizes monitoring performance by automatically balancing polling rates and polling intervals in heavy polling environments.

To accomplish this balance, when required, SolarWinds automatically extends configured polling intervals to decrease the overall polling rate in your environment. To determine when this polling rate adjustment should occur and when users should be notified, SolarWinds has established two polling rate thresholds: a polling warning level and a maximum polling limit level. These levels are based on an internal calculation that includes both the number of network objects polled and the configured polling interval.
As a result of these polling enhancements, it is no longer necessary for a user to compensate for performance issues by manually setting polling rates higher than required to ensure that data is actually polled as frequently as desired. All polling jobs will be completed. If the polling rate is more than the server can handle, the polling intervals are automatically increased to handle the higher load.

**Poller Performance Measurements**

**Polling Completion**

Polling completion represents the percentage of polls across all pollers that were completed without the need to delay polls. The new Orion poller can delay polls if it detects that the Orion server resources are overtaxed. To make it easier to detect ongoing polling issues, the polling completion number is an average of the last one hundred polling jobs. A polling completion number less than one hundred only indicates delayed polls, not failed polls. Polling rate should normally be about one hundred. Significant declines in polling rate are typically due to the Orion server memory or CPU constraints.

**Polling Rate**

The polling rate indicates the percentage of maximum polling rate possible for your server. This number is derived from the Total Job Weight value and the polling interval settings. Each polling job is assigned a weight to best estimate relative load the job places on the server. If the polling rate reaches eighty five percent or greater the Orion server will give a poller status warning popup. At this point you should add an additional poller or scale back you polling rates if you have increased the rates from the default settings.
Appendix A – Other Resources


thwack Community

The thwack community has thousands of users who regularly exchange ideas and solutions. Chances are that if you are having network management issues, there are thwack users who has already solved those issues. SolarWinds employees from Development, Support and Product Management regularly interact with SolarWinds product users on thwack, so we may also be able to help you solve your problem there too.

The thwack Content Exchange portion of thwack can be used for uploading and downloading various templates. These templates can be imported to your Orion system and customized to fit your environment.

http://thwack.com/media/42/orion-apm-content/application-templates/29793/sql-server-via-wmi/
http://thwack.com/media/42/orion-apm-content/application-templates/29791/sql-server-via-snmp/

A good discussion about measuring Orion performance can be for at this thwack thread.