White Paper

Performance White Paper: SQL Server® 2016 Always On Availability Groups on HPE Proliant DL380 Gen9 Servers using HPE 12Gbs SAS Write-Intensive 2.5” SSDs

Ultra-High Performance | Ultra-Low Latency | Sustained High Availability
Balanced | Scalable | Consistent | Predictable

SanDisk®
Western Digital Technologies, Inc.
951 SanDisk Drive, Milpitas, CA 95035
www.SanDisk.com

Table of Contents

Executive Summary

Enterprise database applications demand unprecedented performance, availability, affordability, and flexibility. This document characterizes an architecture and options that satisfy these requirements. With a focus on performance, this white paper describes the server, storage, and software configurations for the following:

- HPE Proliant DL380 Gen9 2U rack server
- HPE 12G SAS Write-Intensive 2.5” 800GB SSDs
- Microsoft® SQL Server 2016 Always On Availability Groups (AGs), configured and validated for a high transaction OLTP TPCC-like workload.

This configuration highlights the outstanding performance obtained by SQL Server AGs on state-of-the-art yet affordable HPE hardware, including high-performance and ultra-low-latency storage. Combined with improved and new features in SQL Server 2016—especially the game-changing enhancements to AG log transport performance—performance and high availability (HA)/disaster recovery (DR) service levels can be met that were previously not possible. All this is achieved in a straightforward, easy-to-implement, and affordable configuration.

The companies and organizations collaborating to make this work possible include HPE, Western Digital (SanDisk® brand technologies), and Microsoft.

Business Value

The system was tuned until the following requirements were satisfied:

- *Consistent high performance*: Over 70,000 transactions/second (over four million transactions/minute) were achieved, a high performance by any standard. Leveraging best practices delivered these transactions predictably, within a narrow range compared to typical deployments.

The *Performance* section documents this as well as much higher numbers gathered from ultra high-performance testing. Both scenarios delivered I/O latency with consistent microsecond latencies for both data and log volumes.

- *HA and DR*: Business transactions were hardened on secondary replica servers in less than a second.

In addition, system components and configuration were chosen to offer the following:

- *Affordability*: HPE servers and storage offer unmatched value. Leveraging the design characterized here maximizes investment, including SQL Server licensing dollars. The in-server flash eliminates the need for SAN infrastructure: footprint, cost, licensing, and a dedicated skill set is no longer required.
- *Flexibility*: This easy-to-implement configuration offers generous resource bandwidth, particularly CPU and I/O, providing flexible deployment options, subsequently described more fully.

Key Sections

Among the most valuable sections in this document are:

- The *Performance* section highlights the remarkable capabilities of SQL Server 2016 on HPE servers and in-server flash storage.
- *Appendix A: Bill of Materials* which documents system affordability. In addition, *Solution Configuration: Options and Details* fully characterizes the flexible deployment options leveraged by the architecture.

This section is necessary for defining the configuration that maximizes IT investment in terms of hardware and software, particularly SQL Server licensing.

- *Appendix B: Lessons Learned* provides guidance for implementing best practices and avoiding missteps in the optimal configuration and deployment.

Configuring SQL Server on HPE servers hosting storage on HPE SSDs moves the bottleneck off the storage tier. Combined with the log transport performance enhancements in SQL Server 2016, enterprises can fully leverage HPE hardware and attain high performance, achieve high availability, and prepare for disaster recovery.

Leveraging SQL Server best practices on a balanced hardware configuration including HPE SSDs provides reliable, consistent performance. This outcome is an enterprise solution with interesting and valuable possibilities.

In summary, this architecture provides the following benefits making it a compelling solution for real-world, tier 1 enterprise implementation:

- Consistent high performance
- High Availability and Disaster Recovery
- Affordability
- Flexibility

Audience

This document is for Database Architects, DBAs, IT Directors, and others who influence or make decisions. It provides guidance for designing enterprise solutions requiring predictable high performance and high availability.

Architecture Background

This HPE configuration offers guidance for implementing a highly available, performant, affordable, and flexible solution, providing the consistency and durability required by real-world Tier 1 OLTP applications.

The following characterizes historic challenges and the solution, including flexible deployment options.

Test goals:

- Consistent, predictable, high performance in terms of business transaction
- Ultra-low I/O (µsec) latency for both data and log volumes throughout workload execution.

In addition, this configuration

- Frees up resource bandwidth, including CPU
- Exposes a number of options in terms of expanded storage capacity
- Adds the ability to host multiple AGs for multiple applications, eliminating the need to leverage readable secondary replicas (and concomitant licensing costs)

See the section below and the *Solution Configuration: Options and Details* section for more information

Moving the Performance Bottleneck from the Storage Tier

For decades, the performance bottleneck for most applications was storage, whether local, direct attached storage, or in a SAN. Leveraging flash, specifically in-server HPE SSDs—which provide high IOPs, high throughput, and ultra-low latency—moves the bottleneck out of the storage tier.

Moving the Perf Bottleneck from the Log Transport in SQL Server 2016 AGs

In SQL Server 2016, Always On Availability Groups benefited from a wide spectrum of enhancements. The AG network log transport mechanism is responsible for transferring changes from primary database(s) to secondary replica(s). In SQL Server 2012 and SQL Server 2014, AGs were not fully optimized for performance. An internal bottleneck in the AG log transport resulted in limited AG traffic across the network from primary to secondaries.

In SQL Server 2016, the Microsoft SQL Server program group set out to provide AG throughput within a very small margin of a stand-alone server performance. In close collaboration with Windows® and Azure® groups, the entire AG network pipeline was reworked, end-to-end.

See *Appendix C: SQL Server 2014 vs. 2016 AG Network Transport Enhancements* to see the performance delta between SQL Server 2014 and 2016. This delta resulted from the new log transport enhancements combined with in-server flash:

- CPU utilization is increased. In these tests, it was by more than 4x.
- Additional processor utilization maps directly to more application transaction throughput. Depending on the workload, these tests showed an increase of 4x – 7x.
- More transactions per server mean more customers served on fewer hardware resources.
- Application throughput is not only significantly higher, it's delivered reliably and consistently.

This testing was done on the same hardware—the only test variable was the version of SQL Server. In other words, simply upgrading availability group implementations to SQL Server 2016 provides significant performance benefits.

In SQL Server 2016, the Always On AGs are fully enterprise-ready. They can be hosted on flash capable for the first time of supporting world-class workloads, such as high-throughput OLTP applications, data warehouse ETL and other data loads, and maintenance such as large index rebuilds.

SQL Server 2016 AG Redo: The Next Opportunity for Optimization

In addition to the performance improvements to AG log transport, enhancements were also made to AG log redo on secondary replicas. For example, it is now a multi-threaded operation. Of the two components, log transport was the most urgent issue to address. Now, data from the primary arrives at the secondary replica logs and is hardened there in near real time. This is critical for Recovery Point Objectives (RPO).

The enhancements to the log transport were a direct result of the SQL Server Product Team listening to customer requests. These outcomes enabled a real-world production HA scenario delivering approximately one million transactions/min, sustainable for an arbitrarily long period of time, which also expands the configuration's flexibility. Our sustained HA test results will be available in a subsequent document.

However, limitations to AG secondary log redo are now the bottleneck. The Product Team is likewise working to remediate this. See *Appendix D: SQL Server 2016 Log Redo Performance* for an analysis and internals insights related to the challenges.

In the meantime, the *Solution Configuration: Options and Details* section provides insight into viable workarounds, many of which have been implemented in production by customers.

Enterprise Workloads, HA/DR, and RPO

In SQL Server 2012 and SQL Server 2014, transactions queue on the primary server while awaiting transport to secondary replicas. These queues are often minutes, sometimes hours long. Enhancements in SQL Server 2016 result in real-time transport of transactions to secondary replicas, where the data is immune to failure of primary servers. SQL Server 2016 has optimized AGs for HA/DR scenarios, making it easier to meet RPO targets.

SQL Server 2016 AGs

In SQL Server 2016, Always On Availability Groups (AGs) benefited from a wide spectrum of enhancements. The solution in this paper leverages those improvements, balancing hardware needs (CPU, memory, storage, and network) and showcasing a high-performance mode.

Solution Configuration Options

The High Availability mode described here involves several options to consider. It is critical to define the configuration that maximizes IT investment in terms of hardware and software, particularly SQL Server licensing. Below is a summary of the information found in the *Solution Configuration: Options and Details* section.

- Preserve SQL Server licensing costs by leveraging fewer processors:
  - Populate the server with only one CPU.
  - Use two CPUs, but with fewer processors.
- If the consistent, high-performance, ultra-low I/O latency documented here isn't demanded, consider a single I/O controller.

- SSD capacity options using the 1.6TB version of the 800GB SSDs used here and/or leveraging more of the 24 slots in the HPE Proliant DL380 G9. Allocating a mirrored pair for the OS leaves 22 slots. Raw capacity using 1.6TB SSDs exceeds 35TB.
- Further leverage the hardware via consolidation. For example, create a separate AG, consider doing so on a separate SQL Server instance (or a virtual machine), affinitizing each instance (or VM) to separate processors.
- Consider considering discrete instances of SQL Server hosting a separate AG on one or both of this configuration's secondary replicas. The configuration documented in this paper provides two secondary replicas, formerly known as "passive" nodes (see this post from Microsoft MVP Allan Hirt for more information and guidance).
- The resource bandwidth available on the primary may make readable secondaries unnecessary, thus saving hundreds of thousands in SQL Server licensing costs per server.
- Consider Basic Availability Groups, a hobbled implementation of AGs new to SQL Server 2016 supporting only one database and one non-readable secondary replica.

See the *Solution Configuration: Options and Details* section for more information.

About the Components

About HPE ProLiant DL380 Gen9 Servers

http://www8.hp.com/us/en/products/proliant-servers/product-detail.html

http://ssl-product-images.www8-hp.com/digmedialib/prodimg/lowres/c04412084.png

The HPE ProLiant DL380 Gen9 Server delivers reliability, serviceability, and near continuous availability, backed by a comprehensive warranty. It supports the most basic to mission critical applications, making it ideal for any server environment. This 2U server leverages Intel's latest E5-2600 v4 processors, the latest HPE 2400 MHZ DDR4 SmartMemory supporting 3.0TB, and supports 12Gb/s SAS (externally) and PCIe (internally) for I/O, as well as 40Gb/s networking (80Gb/s with Windows NIC teaming).

The HPE DL380 Gen9 Server has a flexible redesigned chassis, including new HPE Universal Media Bay configuration options with 8 to 24 2.5" small form factor (SFF) and 4 or 12 large form factor (LFF) drive options along with NVMe options and additional rear drive support.

Customers can choose an embedded 4x1GbE, HPE FlexibleLOM or PCIe standup 1GbE to 40GbE Adapters, providing flexibility of networking bandwidth and fabric to adapt and grow for changing business needs.

The HPE DL380 Gen9 Server supports industry-standard Intel® Xeon® E5-2600 v3 and E5-2600 v4 processors with up to 22 cores, 12G SAS, and 5TB of HPE DDR4 SmartMemory. Depending on optional risers, the servers host up to nine PCIe slots.

Learn more about the HPE ProLiant DL380 Gen9 Server here: www.hpe.com/servers/dl380gen9

About HPE 800GB 12G SAS Write Intensive 2.5" SSDs

HPE provides the most complete portfolio of choices across SAS and SATA and first to market with latest technologies. HPE Solid State Drives (SSDs) leverage NAND flash and deliver exceptional performance and endurance while reducing power consumption for applications requiring high random read and write IOPs. The drives do so while reducing datacenter energy consumption and delivering more compute per watt.

HPE SSDs deliver substantially higher performance, better latency, and more power-efficient solutions compared to traditional rotating media.

The HPE 800GB 12G SAS Write Intensive-1 SFF (2.5in) SSD used in this solution is joined in the HPE storage portfolio by HPE 12Gb SAS Mixed Use and Write Intensive SSDs. Available in 800GB and 1.6TB capacities and categorized by Read Intensive (RI), Mixed Use (MU), and Write Intensive (WI), customers can choose the right SSD that tailored for workload demands. HPE SmartSSD Wear Gauge management tools, such as HPE Smart Storage Administrator, can be used to monitor the SSD life. Self-describing LEDs reduce drive activity confusion.

HPE SAS SSDs deliver reliability via a dual-port interface. They provide power loss protection that continues to protect data even when the datacenter loses power. HPE firmware is written specifically to provide compatibility with the ProLiant Server series and HP controllers for consistency and high performance. Members of the HPE SSD storage portfolio have undergone 2.4 million hours of rigorous testing to achieve high quality standards, including high shock and vibration testing.

The following table summarizes characteristics of the SSDs used in this testing.

HPE 800GB 12G SAS Write Intensive 2.5" SSD (846430-B21)		
Performance	*Data Sheet*	*Quick Specs*
Sequential Reads (MB/s)	1,080	1,000
Sequential Writes (MB/s)	580	565
Random Reads (IOPs)	100,000	66,000
Random Writes (IOPs)	68,000	64,000
Interface and Physical Characteristics		
Interface	12 GB/s SAS	
Dual Port	Yes	
Hot Pluggable	Yes	
Sector Size (bytes)	512	
Form Factor	SFF	
Max Power (Watts)	9	
Operating Temperature (ºC)	0 – 60	
Reliability		
Endurance (Drive Writes/Day)	10	
MTBF (hours)	2,000,000	
Unrecovered Bit Error Rate (UBER)	<1 sector in 10^{18} bits read	

Table 1. Specifications for HPE 800GB 12G SAS Write Intensive 2.5" SSD

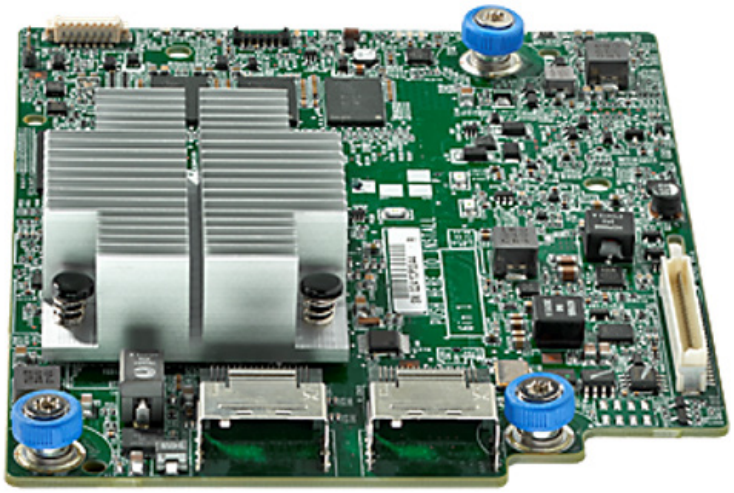
For more information about the SSDs tested here, see HPE 800GB 12G SAS Write Intensive-1 SFF (2.5in), HPE product number 846430-B21:
www8.hp.com/us/en/products/server-solid-state-drives/product-detail.html?oid=1008829775

For more information on other HPE SSDs, see the following resources:

- Data Sheet: HPE Solid State Drives (SSDs) (12 pages)
  https://www.hpe.com/h20195/v2/GetDocument.aspx?docname=4AA4-7186ENW
- Quick Specs: HPE Solid State Drives (SSDs) (97 pages)
  https://www.hpe.com/h20195/v2/GetDocument.aspx?docname=c04154378

Note: For information specific to the 846430-B21 SSDs used in this architecture, see page 5 in the Data Sheet and page 87 in the Quick Specs.

About HPE Smart Array P440 and P440ar I/O Controllers



http://www8.hp.com/us/en/products/iss-controllers/product-detail.html?oid=7274889

The HPE Smart Array P440ar PCIe I/O controller (and functionally identical P440 integrated I/O controller, also used in this testing) supports up to eight 12Gb/s SAS links performance and boasts 2GB of flash-backed write cache. The card's Smart Storage battery supports both read-ahead and write-back caching, with indefinite write cache data retention in the case of unexpected power outage. An overview of these and other features can be found here:
http://www8.hp.com/us/en/products/iss-controllers/product-detail.html?oid=7274889

See also the P440ar Quick Specs:
http://www8.hp.com/h20195/v2/getpdf.aspx/c04346299.pdf?ver=1.0

About SQL Server 2016 Always On Availability Groups (AGs)

SQL Server 2016 is widely considered to be the most important SQL Server release since SQL Server 2005. Significant improvements and new features have been engineered into the product, such as data visualization, business intelligence, big data, R integration, broad Hekaton improvements, columnstore index improvements for data warehousing and OLTP databases, and support for the new Operational Analytics feature.

The most significant aspect may be the performance improvements to the Always On Availability Groups features, including parallel log redo and especially log transport enhancements.

SQL Server 2016 AG enhancements are listed below.

Dependencies

- SQL Server 2016 is 64-bit only, no 32-bit version
- .NET Framework 3.5x independent
- SQL Server Standard Edition support
- Domain-independent Availability Groups

Security

- Increased support for transparent data encryption (TDE)
- Group-managed service accounts (gMSAs) support
- Default endpoint encryption changed from RC4 to AES

Failover

- Automatic failover replicas increased from two to three
- Database-level failover options

Miscellaneous

- Enhanced Azure integration
- Distributed transaction coordinator (DTC) support (limitations apply)
- Availability group "Direct Seeding"
- Distributed availability groups
- Enhanced online operations

Readable Secondaries

- Round-robin load balancing in readable secondaries
- Enhanced Azure support
- Clustered Columnstore support

Performance

- Log transport compression behavior
- Parallel log redo
- Log transport

For more information on SQL Server 2016, see the Microsoft SQL Server home page: www.microsoft.com/sql

For highlights on SQL Server 2016, see the "#DataDriven" release keynote from Microsoft CEO Satya Nadella and the accompanying 33 videos on specific features from SQL Server program group PMs on the SQL Server 2016 launch site:
www.microsoft.com/en-us/cloud-platform/data-driven

For information on SQL Server 2016's high performance capabilities, see the SQL Bits session by SanDisk Data Propulsion Laboratory Architect Niall MacLeod (co-authored by colleague Brian Walters) – SQL Server 2016 Availability Groups – Replicating data fast:
www.sqlbits.com/Sessions/Event15/SQL_Server_2016_Availability_Groups_Replicating_data_at_the_speed_of_flash

For an overview of the SQL Server 2016 Always On Availability Group enhancements cited above, see the Professional Association for SQL Server 24 Hours of PASS (PASS 24 HoP) session delivered by SanDisk Data Propulsion Laboratory Architect Jimmy May:

SQL Server 2016 Always On Availability Groups Enhancements
www.sqlpass.org/24hours/2016/edp/Sessions/Details.aspx?sid=48777

Configuration Components

This section describes the specific hardware, software, and configurations to replicate the reference environment.

Overview

This table is an overview of the reference environment, consisting of three physical servers deployed as follows.

Component	Description	Quantity per Server
Server	HPE ProLiant DL380 Gen9	
Operating System	Microsoft Windows Server 2012 R2 Standard	
SQL Server Version Edition	SQL Server 2016 (RTM) Developer Edition (64-bit)	
CPU	Intel Xeon E5-2697 v4 (Broadwell-EP) 2.6GHz	2 sockets 16 physical cores each 32 logical cores each 64 logical cores total
Drives (HDD)	HPE 1.2TB 6G SAS 10K 2.5"	2
Drives (SSD)	HPE 800GB 12G SAS Write-Intensive 2.5"	10 (6 data; 4 log)
Storage Controller	HPE SmartArray P440ar	1
RAM	DDR4-2133MHz	256GB 8 x 32GB DIMMs
NIC	Mellanox ConnectX-3	2

Table 2. Overview of architecture physical specifications.

Hardware

The following sections describe the hardware subsystems in this configuration.

Server

The architecture was comprised of three 2U HPE ProLiant DL380 Gen9 servers. Each hosted a single instance of SQL Server 2016, with an Always On Availability Group configured with three replicas—one primary and two secondaries. See *About HPE ProLiant DL380 Gen9 Servers* for more info.

The BIOS was accessed remotely to set the following settings, best practices configured to maximize performance.

System Utilities > System Configuration > BIOS/Platform Configuration (RBSU)

- System Options
  - Processor Options > Intel Hyperthreading: [Enabled]
  - Virtualization Options > Virtualization Technology: [Disabled]
- Power Management
  - Power Profile: [Maximum Performance]
- Advanced Options > Fan and Thermal Options
  - Thermal Configuration: [Increased Cooling]

Settings were validated after rebooting.

Processor

Each 2U HPE server featured two sockets, both populated with Intel Xeon E5-2697 v4 (Broadwell-EP) 2.6GHz 64-bit processors. Each processor had 16 physical cores. Hyper-threading exposed 32 logical cores per processor, i.e., a total of 64 logical cores per server.

Intel's ARK database provides more information on this processor:

http://ark.intel.com/products/91768

Memory

Each server contained eight 32GB DIMMs (DDR4-2400 MHz) for a total of 256GB of RAM.

Storage

All storage was internal, small-form-factor (SFF) 2.5" drives. The OS was hosted on a mirrored pair of HPE 1.2TB 6G SAS 10K HDDs. The database data and log files utilized the HPE Smart Array P440, which includes an integrated P440 controller to host the data files and a PCIe P440ar controller to host the log files.

- Firmware version: 4.02
- Driver version: 63.12.0.64

SQL Server 2016 data and log files were hosted on ten HPE 800GB 12G SAS Write-Intensive SSDs. SQL Server 2016 data files were configured on six SSDs configured as RAID 10. The log files were hosted on four SSDs, also configured as RAID 10. All SSDs were updated with HPE firmware version HPD5.

Volumes for SQL Server 2016 data and log files were explicitly configured, using `diskpart.exe`, as GPT volumes with a 1MB offset and 64KB file allocation unit size.

The data and log arrays were configured using the HPE Smart Storage Administrator or the HPE `ssacli` command- line utility. The following defaults were adopted:

- Default stripe size / full stripe size
- 32 sectors/track
- SSD over-provisioning option

Network

For bandwidth and redundancy, each server was supplied with two Mellanox ConnectX-3 Pro PCIe cards connected to a 40Gb/s backbone. The operating system was configured for NIC teaming for redundancy and performance which provided bandwidth aggregation at 80Gb/s; see the NIC Teaming section for more information.

- Firmware version: 2.35.5100
- Driver version: 5.10.11345.0

Operating System: Windows Server 2012 R2

The operating system for all servers was Microsoft Windows Server 2012 R2 Standard.

Power

The `powercfg.cpl` applet in the Windows Control Panel was configured for "High performance". To access this applet from the Control Panel, go to Control Panel > Hardware > Power Options > High performance.

Local Security Policies

In the Local Security Policy console (`secpol.msc`), the SQL Server service account was added to:

- Lock pages in memory
- Perform volume management tasks

Once the console is open, these settings can be located at Security Settings > Local Policies > User Rights Assignment.

NIC Teaming

NIC Teaming (load balancing and failover—LBFO) was configured on the dual, redundant Mellanox ConnectX-3 NICs. Doing so provided the following benefits:

- Bandwidth aggregation
- Traffic failover to prevent connectivity loss in the event of a network component failure

Each NIC port provided 40Gb/s; aggregating two ports provided 80Gb/s. Two NICs provided the redundancy demanded by a Tier 1 enterprise solution.

To learn more about NIC teaming in Windows Server 2012 R2, including configuration guidance, see:

Windows Server 2012 R2 NIC Teaming User Guide – A Guide to Windows Server 2012 R2 NIC Teaming for the novice and the expert at https://gallery.technet.microsoft.com/windows-server-2012-r2-nic-85aa1318

SQL Server 2016

SQL Server Instance

SQL Server 2016 (RTM) Developer Edition was used for the testing. Developer Edition shares the same full-featured capabilities as Enterprise Edition. TheSQL Server administrative related configuration options included:

- `Show advanced options: 1`
- `Backup checksum default: 1`
- `Remote access: 1`
- `Remote admin connections: 1`

SQL Server performance related configuration options were as follows:

- `min server memory (MB): 235814` (230GB of 256GB total physical memory)
- `max server memory (MB): 235814` (230GB of 256GB total physical memory)
- `backup compression default: 1`

Note that `max degree of parallelism` was set to the default of `0`. Parallelism was set within the test database using `ALTER DATABASE SCOPED CONFIGURATION` introduced in SQL Server 2016. See the *Test Database* section below for additional information.

Startup trace flags included:

- `-k750`
- `-T1204`
- `-T1222`

Note that `-k` is a fully supported, though little-known option. Its purpose is to throttle SQL Server checkpoints, mitigating the impact of flooding often overwhelming the disk I/O subsystem. This trace flag is indispensable for reliable, consistent application performance. Please review a detailed performance analysis in *Re-creating AG on New WFSC*.

Some aggressive means were used in providing a robust solution. Tests often included rebuilding the entire environment from scratch. A PowerShell framework was created to expedite this work. However, after destroying the Windows Server Failover Cluster (WSFC), the framework failed to surface an error which occurs when re-recreating the AG:

```
Msg 41105, Level 16, State 0, Line 112
Failed to create the Windows Server Failover Clustering (WSFC) resource with name
'HADR_AG'
and type 'SQL Server Availability Group'.
The resource type is not registered in the WSFC cluster.
The WSFC cluster may have been destroyed and created again.
To register the resource type in the WSFC cluster, disable and then enable Always On in
the SQL Server Configuration Manager.
```

A manual rebuild exposed the error, and the solution was to simply disable "Enable Always On Availability Groups". This is found in the AlwaysOn High Availability tab of the instance's SQL Server properties, in the SQL Server Configuration Manager dialog (`SQLServerManager13.msc`).

Note: In contrast to SQL Server 2012 and 2014, in SQL Server 2016 RTM uses "Always On"—with a space in the name. This change has not yet been propagated to all product components.

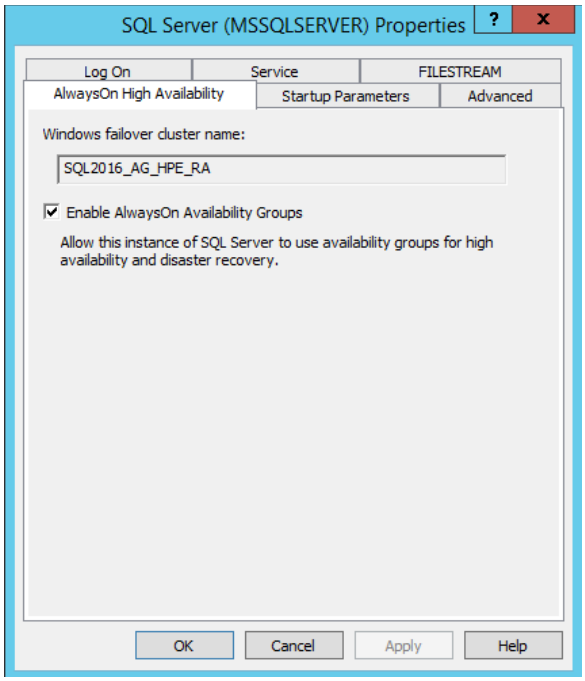


Figure 1. Enabling Always On Availability Groups in SQL Server Configuration Manager properties dialog.

Schema Design: Smaller Databases in Multiple AGs

If your business model is compatible, consider splitting your application into multiple databases configured in multiple AGs. This will better leverage the hardware investment and act as a workaround to the SQL Server 2016 log redo limits. A noted entertainment company has done so with great success. See the *Performance* section and *Appendix B: Lessons Learned* for additional insight into this invaluable feature as an alternative option.

`-T1204` and `-T1222` were leveraged for deadlock troubleshooting. These were necessary while tuning the high-performance workload; once resolved, no such challenges arose again.

Trace flags `-T1117` and `-T1118` were not used, following a long-standing best practice for concurrency, especially in tempdb. See the *Tempdb* and *Test Database* sections for more information.

Finally, other commonly adopted trace flags for performance and scalability, such as `-T2371` and `-T8048`, also were not used. These and others are integrated into SQL Server 2016 and are no longer required to be specified.

Tempdb

To comply with best practices for tempdb concurrency enhancements, eight equally-sized, right-sized files were configured (each data file was 10GB totaling 80GB plus a 10GB log file).

Note that in the context of tempdb, startup trace flags `-T1117` and `-T1118` are "baked in" to SQL Server 2016. These options manage AutoGrow in multi-file filegroups and shared global allocation map (SGAM) utilization, respectively. Optimal settings are automatically enabled for tempdb and these trace flags are ignored by SQL Server 2016. (Settings for user databases are now managed separately; see the next section.)

Test Database

Database compatibility was set to 130 (SQL Server 2016 compatibility). AutoGrow was enabled, and database data and log files were right-sized per best practices.

Though not used, these two new options introduced in SQL Server 2016 were set to manage AutoGrow:

- The database-level option default value for the `MIXED_PAGE_ALLOCATION` setting is `ON`, eliminating shared global allocation map (SGAM) generation.
- For each filegroup, the filegroup-level option controlling file AutoGrow was set to `AUTOGROW_ALL_FILES`.
- As noted earlier in the *SQL Server Instance* section, the instance-level option max degree of parallelism was kept at the default 0. Parallelism was set within the test database using `ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 1` as introduced in SQL Server 2016.

Books Online discusses the utility of this syntax for setting maximum degree of parallelism, parameter sniffing, clearing the procedure cache, and toggling query optimization hot fixes at the granularity of the database.

SQL Server Availability Group Configuration

The test scenario was designed to emulate a typical enterprise HA/DR scenario. This consisted of an AG comprised of three replicas (one primary and two secondaries) in Synchronous Commit Availability Mode. An AG listener—a virtual network name—was optional in these tests to support application connectivity for failover. Secondaries were not readable (the default configuration was accepted: `ALLOW_CONNECTIONS = NO`). AG traffic was dedicated to a private network capable of supporting 80Gb/s. The AG configuration is shown below.

Appendix E: Availabilibity Group Code Samples provides guidance for replicating the AGs used for this architecture. The following figure provides a schematic overview.

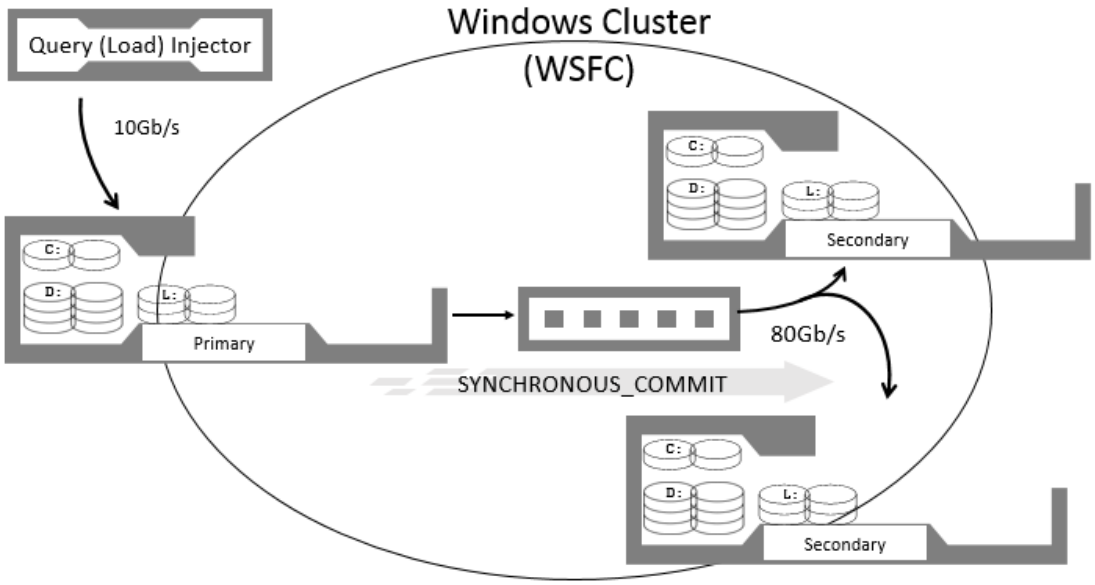


Figure 2. Architecture diagram. The workload is consumed by the primary replica.

The Log-transparent traffic is transmitted in synchronous commit, at an aggregated bandwidth of 80Gbs, to two secondary replicas. The diagram shows the storage layout. OS volumes (C:) were configured as two-HDD RAID 1. The Data (D:) volumes were six-SSD RAID 10. Log volumes were configured as four-SSD RAID 10. Not shown: The OS and Data volumes shared a P440 SmartArray I/O controller; the Log volumes enjoyed dedicated controllers.

Workload

A TPCC-like, OLTP workload was used with 5,000 logical warehouses employed. Data files totaled 1.4TB, and the log file was 600GB, for a total database size of 2.0TB.

For all tests, these goals were paramount:

- High performance in terms of business transactions, delivered at a predictable rate within a narrow range compared to typical deployments
- High I/O performance
- Ultra-low I/O latency for both data and log volumes
- AG log transport rate from the primary replica equivalent to log redo rate on the secondary replicas, maintaining secondary recovery queues at or near zero

The objective was to provide high yet balanced performance while maintaining conditions for near-instantaneous failover.

Solution Configuration: Options and Details

As outlined in the *Executive Summary*, this solution provides several options to maximize HPE hardware and Microsoft software investments. This solution offers high and consistent performance, HA/DR, cost savings, and the valuable enterprise features and functionality offered by SQL Server 2016. This creates an unparalleled opportunity for enterprise customers to host their business applications and meet customer needs.

Leveraging hardware as fully as possible and maximizing CPU utilization moves the bottleneck to the CPU—where it belongs. SQL Server licenses are sold by the core, so an idle CPU isn't doing nothing—it's burning licensing dollars. Higher CPU utilization translates to better value for enterprise investments in SQL Server licensing as well as hardware.

This section should be studied thoroughly. Conserving SQL Server licensing costs on just one server may result in significant savings. It may be possible to do so while still maximizing high availability and performance. For example, reducing processor count or eliminating readable secondaries can save hundreds of thousands of dollars. Careful consideration should be given to the potential for risk, including additional complexity.

The note in the *Processor Options* section immediately below is particularly important.

Below are details for additional solution configuration options.

Processor Options

For workloads that can be supported by doing so, reduce SQL Server licensing costs by:

- Installing only one processor. Depending on your licensing model, this can save over $100,000 per primary replica server. See *Appendix B: Bill of Materials* for information on retail pricing.
- Using two processors but with lower core counts. For example, the 2.7 GHz 16-core Broadwell processor used here could be replaced by a 2.1 GHz 8-core Broadwell or even Haswell processors. See Broadwell-based Xeon Processors or Haswell-based Xeon Processors for additional information.

Note: Low CPU utilization may occur on the primary, limited by secondary replica log redo performance on the secondaries. In ultra-high performance workloads, the secondaries cannot keep up. Driving up CPU utilization thus unbalances the architecture. A future SQL Server incremental release will increase redo performance. Until then, if the secondaries cannot keep up, a customer can reduce core count on the primary to drive up CPU utilization and simultaneously reduce the SQL per-core licensing cost. When that future release delivers increased redo performance, updating the primary with additional processors would be a simple way to scale the primary's performance while continuing to match CPU and SQL Server licensing cost to the performance delivered by the balanced system. See *Appendix D: SQL Server 2016 RTM Log Redo Performance* for insight into SQL Server internals.

I/O Controller Options

- If the consistent, high performance documented here isn't demanded, consider using a single controller.

Capacity Options

- Consider using the 1.6TB version of the 800GB SSDs featured in this solution. The usable capacity of the current 6-disk RAID 10 array hosting the data volume could be expanded to 5.4TB.
- The HPE DL380 G9 can host 24 2.5" disks. Allocating a mirrored pair for the OS leaves 22. Raw capacity using optional 1.6TB SSDs exceeds 35TB.

Consolidation Options

- Further leverage the hardware via consolidation. For example, create a separate AG on a separate SQL Server instance (or a virtual machine), affinitizing each instance (or VM) to separate processors.

Secondary Replica Servers as Primaries for Other AGs

- Consider discrete instances of SQL Server hosting a separate AG on one or both of this configuration's secondary replicas. The configuration documented in this paper provides two secondary replicas (formerly known as "passive" nodes). Resource utilization is trivial. These aren't "wasted resources"—HA is a valuable business requirement. The additional risk that comes with co-mingling applications may or may not be be acceptable and appropriate.

Readable Secondaries

- The resource bandwidth available on the primary makes readable secondaries unnecessary, saving hundreds of thousands of dollars in SQL Server licensing costs per server.
- For eligible implementations consider Basic Availability Groups in SQL Server Books Online, a hobbled implementation of AGs. This is new to SQL Server 2016, supporting only one database and one non-readable secondary replica. Basic AGs are supported on SQL Server Standard Edition. Microsoft introduced them as an upgrade path from Database Mirroring (DBM), which is marked for deprecation.
- See the following for more information:

  Basic Availability Groups (Always On Availability Groups)
  https://msdn.microsoft.com/en-us/library/mt614935.aspx

Performance

This section demonstrates the performance achieved in this solution. These outstanding results were a function of:

- A balanced HPE hardware configuration including three 2U DL380 Gen9 servers
- Intel Xeon E5-2697 v4 (Broadwell-EP) 2.7GHz 64-bit processors
- HPE I/O controllers with firmware optimized specifically for HPE SSD storage
- Enhancements in SQL Server 2016 AGs
- Leveraging SQL Server best practices
- Leveraging the –k SQL Server startup trace flag to throttle checkpoint flushes
- Adjusting the maximum transaction throughput of the primary AG to match the performance constraint imposed by SQL Server 2016 secondary replica log redo. This keeps the recovery queue at or near zero, providing nearly immediate recovery and high availability on failover

The following results are documented from the architecture of three replicas (one primary, two secondaries) with AGs in synchronous commit mode:

- AG Maximum Performance: Maximum performance was documented, while maintaining ultra-low latency. This test shows that the AG log transport moves data across the wire in virtually real time—no queuing on the primary. Perfomance exceeds 70,000 transactions/sec (over 4 million transactions/min). Even at this performance, read/write latencies for both data and log volumes were consistently in the sub-millisecond range.

Ultra-High Maximum Performance

The testing documented here demonstrates the impressive potential of a system comprised of 2U servers and the new capabilities of SQL Server 2016. Ultra-high performance was demonstrated in transaction throughput and disk performance, including ultra-low latency.

For a one-hour test, consistent asnd sustained transaction throughput exceeded 70,000 transactions/sec. This maps to over 4 million transactions/min, and over 250 million for the duration of the test.

Few applications require support for over a quarter-billion transactions, even for "bursty" workloads (data loads, peak production times, etc.). Yet this architecture does so with a strong performance profile.

- Workload: The SQL Server 2016 workload supported over 200MB/sec of log generation (`Log Bytes Flushed/sec`).
- AG log transport: The SQL Server 2016 Always On Availability Group log transport generated combined throughput to both secondary replicas exceeding 400MB/sec of data replication across the network. This reflects the fantastic enhancements introduced in SQL Server 2016. On the primary replica, the value of `Bytes Sent to Transport/sec` is twice that of `Log Bytes Flushed/sec`.
- Even with this performance, read/write latencies for both data and log volumes were consistent in the sub-millisecond range.

Important: Log backups were not taken during this test. Also, this workload significantly exceeds the capabilities of AG secondary log redo. This configuration is not highly available, in the sense that immediate recovery after failover is not possible. Though the AG log transport mechanism hardens the log on

secondary replicas in near real-time, the AG log redo mechanism cannot keep up in the RTM release of SQL Server 2016. On failover, limits to AG log redo rate may render the system unavailable for a recovery that could last up to three hours. Subsequent collateral will detail a true HA configuration.

Ultra-High Performance: Relevant Perfmon Metrics

Counter	Metric
% Processor Time	80.8
Transactions/sec Transactions/min	71,171 4,270,260
Log Bytes Flushed/sec	213,463,921
Bytes Sent to Transport/sec	428,131,725
Percent Log Used	Rising

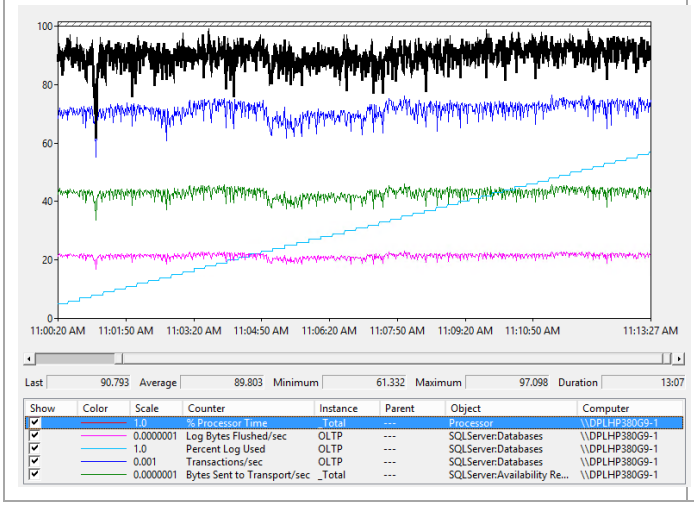
Table 3. Relevant system performance metrics for high performance mode.

In the table above, note the high CPU utilization as well as impressive log and AG redo-related metrics. Also, the Percent Log Used counter in this test is rising, reflecting the absence in this test of backups, focusing instead on raw performance.

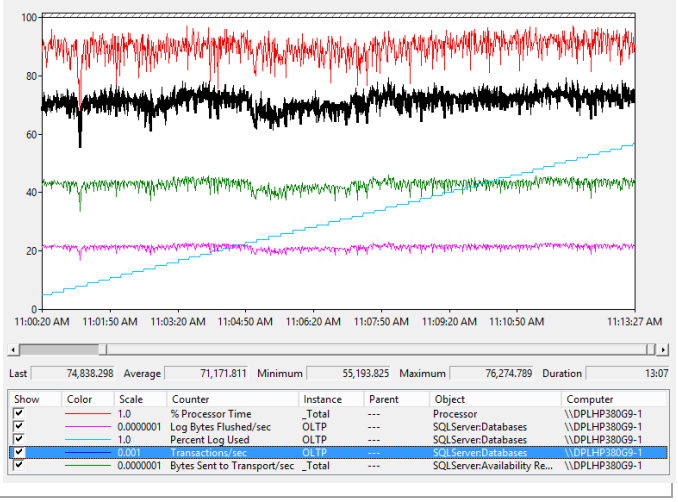
Ultra-High Performance: Relevant Perfmon Graphics

The ultra-high performance mode shown here documents the remarkable performance this configuration can temporarily sustain. In these tests, over 4 million transactions/min are sustained for the capacity of the log files, for both primary and secondary replicas. In testing, the 600GB log file on the secondary overcame the ability of the max AG log redo rate in approximately 90 minutes.

% Processor Time: 80.8



Transactions/sec: 71,171
Transactions/min: 4,270,260



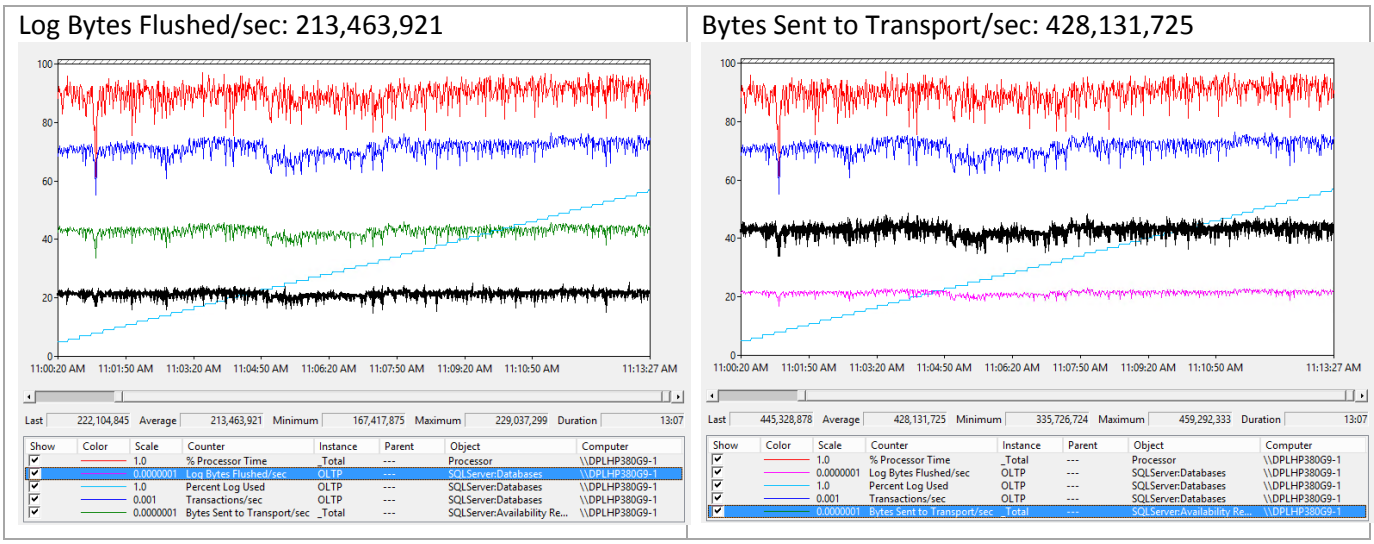


Figure 3. Relevent Perfmon metrics characterizing the ultra-high performance provided by this configuration.

In the figure above, note the consistency of metrics owing largely to the ability of the HPE flash to absorb the insult of SQL Server checkpoints.

Disk Performance: Data and Log File IOPs, Throughput, and Latency

One of the highlights of this architecture is the impressive performance of HPE SSD storage. The results below reflect the high-performance configuration just described.

The workload generated I/O that would demanded by few production workloads in the world, yet the storage IOPS and throughput had significant headroom. The storage performed at microsecod read and write latencies for both data and log volumes.

Combined throughput for both volumes exceeded over 1GB/sec, 60GB/min, 3.6TB/hr while sustaining ultra-low, microsecond latency.

Note: I/O latency was so low in testing that the default scale for perfmon shows 0ms for reads and writes for both data and log volumes. The figure below provides an example taken during a test generating over 1GB/sec of I/O. *In the summary graphics farther down, the perfmon scale for latency has been changed from 1.0 to 10,000.*

Read and write response was similar and, in all cases, ultra-low. For simplicity, the Transfer results are shown, reflecting both reads and writes.

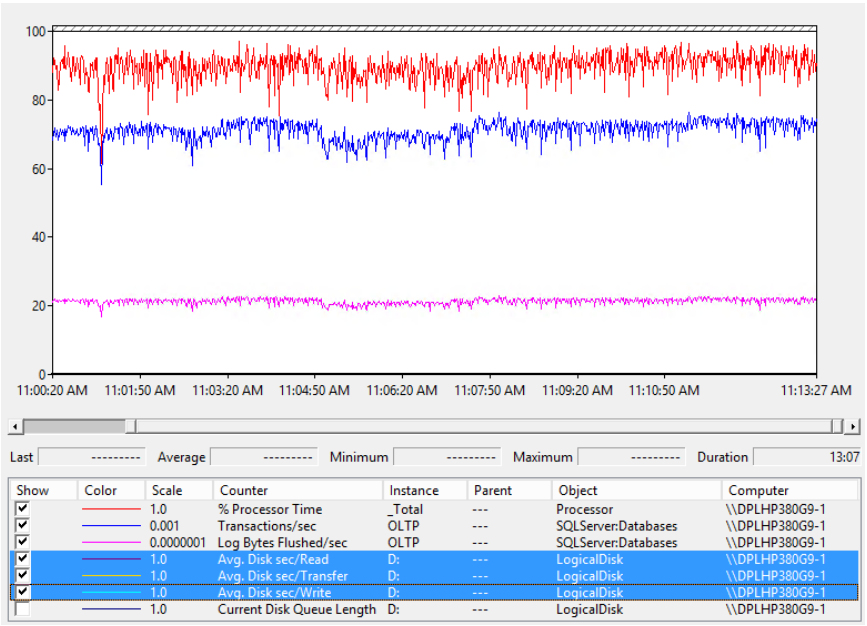


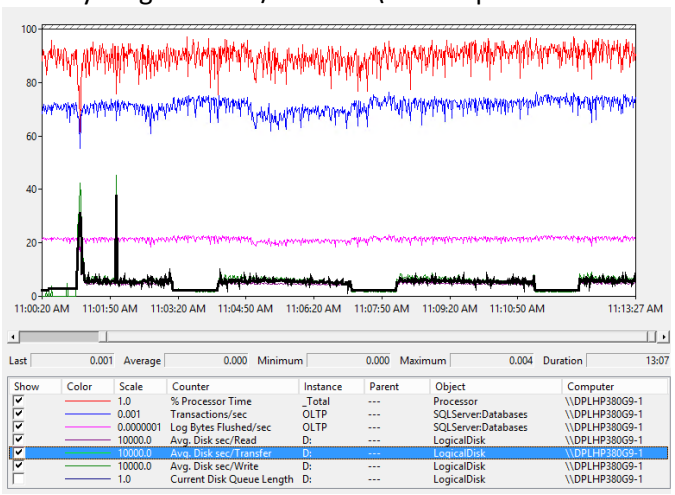
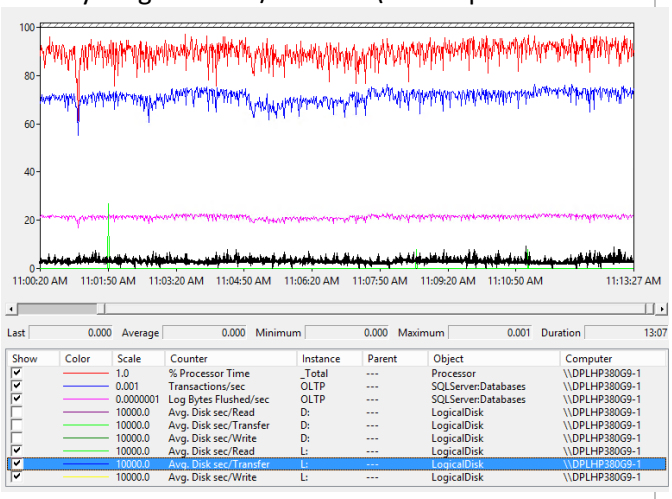
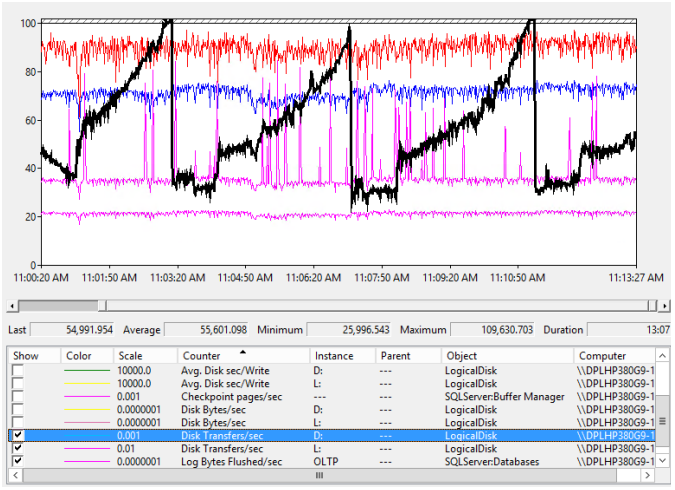
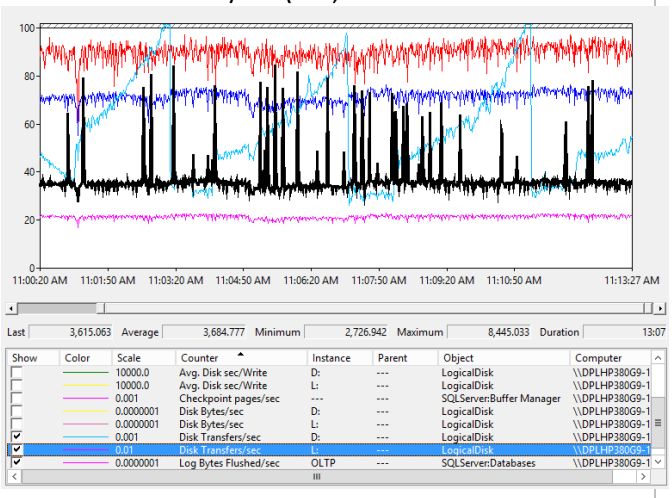
Figure 4. Perfmon graphic demonstrating HPE SSD storage delivering latency of 0ms (dark line at the bottom of the chart) while driving over 1GB/s of throughput.

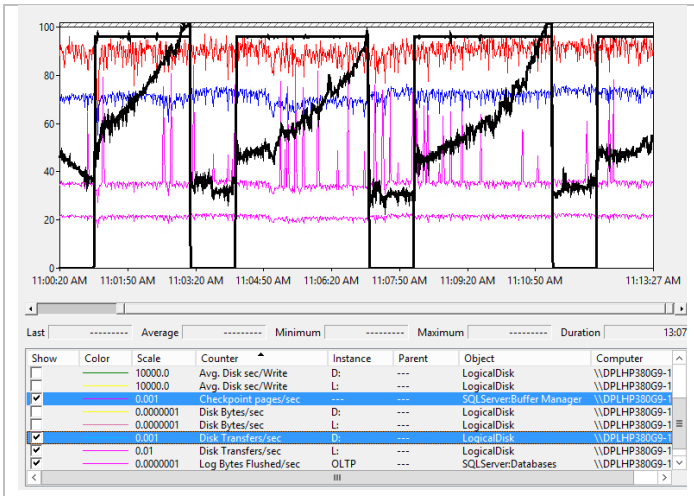
These results were captured using the default scale for latency of 1.0. (In the images below, the scale has been changed to 10,000 to show more granular results.)

The following table and perfmon graphics summarize performance during a test such as described above:

Counter	Value	Data (D:)	Log (L:)	Total
% Processor Time	80.8			
Transactions/sec Transactions/min	71,171 4,270,260			
Log Bytes Flushed/sec	213,463,921			
Bytes Sent to Transport/sec	428,131,725			
Percent Log Used	Rising			
Latency		~400µs	~200µs	
IOPs		55,601	3,685	59,286
Throughput (Bytes/sec)		869,119,319	213,464,874	1,082,584,193

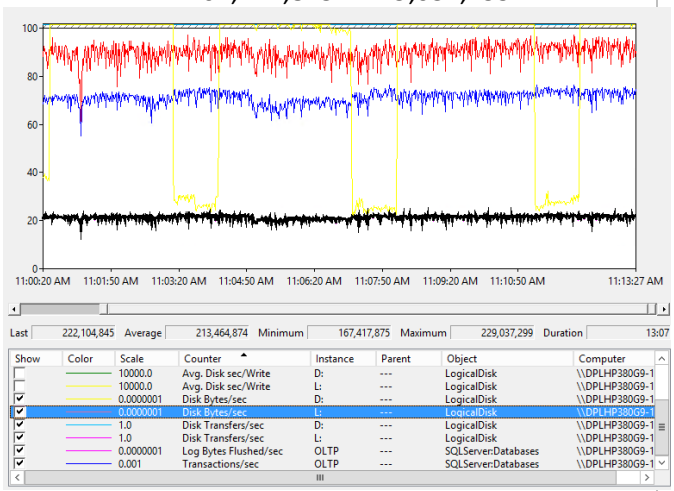
Table 4. Relevant performance metrics for high performance mode. For convenience, the table includes not only the system data from the preceding section as well as disk-related metrics.

Data Volume (D:)	**Log Volume (L:)**
Latency: Avg. Disk sec/Transfer\D: ~400µs	Latency: Avg. Disk sec/Transfer\L: ~200µs
	
IOPs: DiskTransfers/sec\D: 55,601	IOPs: DiskTransfers/sec\L: 3,685
	
To infer correlation, IOPs and Checkpoints counters are juxtaposed	



Throughput: Disk Bytes/sec \D:
avg 869,119,319
min 212,963,683 :: max 1,791,358,898

Throughput: Disk Bytes/sec\L:
avg 213,464,874
min 167,417,875 :: 229,037,299



To infer correlation, Throughput and Checkpoints counters are juxtaposed.

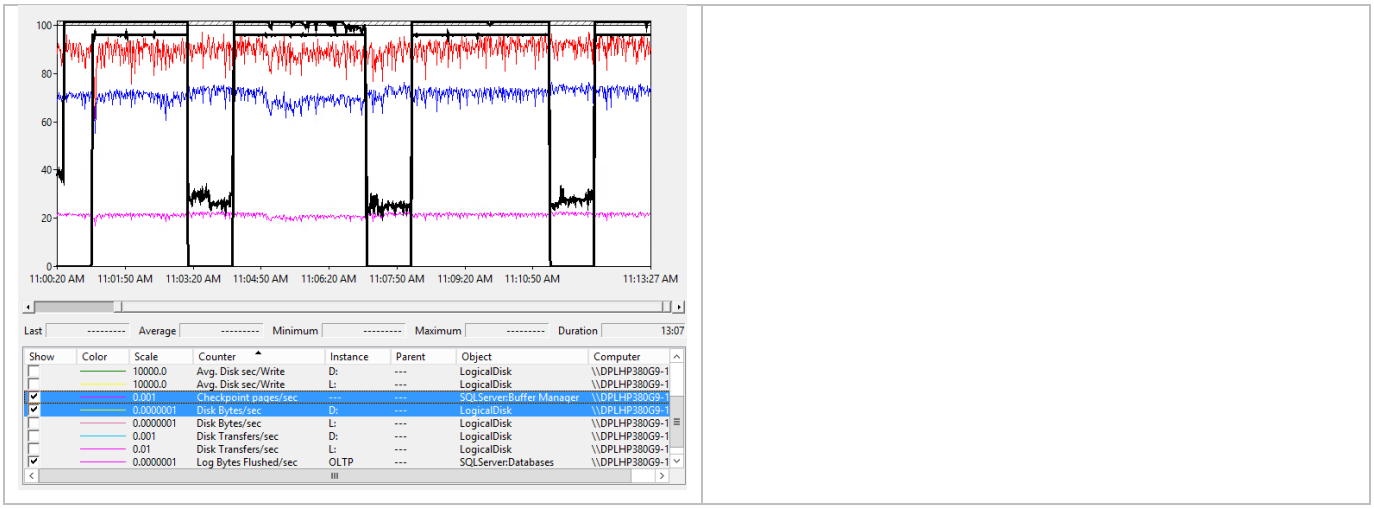


Figure 5. Representative disk metrics showing the performance of HPE flash storage used in this configuration.

The table above summarizes the information from the other tables. In these images, the scale for latency has been changed from the historical default of 1.0 to 10,000. At the default scale, latency appears to be 0ms. Changing the scale exposes the µsec latency provided by the storage, even during checkpoint, while delivering impressive throughput and IOPs. The figure also highlights representative metrics describing the IOPs and throughput. Juxtaposing IOPs and throughput with checkpoints reinforces the ability of the storage to absorb I/O extremes while sustaining ultra-low latency.

Summary

The configuration documented here maximizes affordable, high-performance hardware as well as reduced software licensing investment. This provides consistent, robust, and reliable performance for both sustainable high availability and ultra-high performance.

This document provides guidance for implementing best practices. This includes improved features in SQL Server 2016 such as the enhanced AG log transport mechanism, as well as insights for creating an optimal configuration and deployment. Highlights of the business value include:

- Consistent high performance
- High Availability and Disaster Recovery
- Affordability
- Flexibility

Appendix A: Bill of Materials

The following table includes all components of this solution (server racks and network switch excluded). These prices reflect full retail; substantial discounts are commonly available.

The cost to license Windows Server 2012 R2 Standard Edition for all three servers is $882 per server, for a total of $2,646 (full price, no discounts, no Software Assurance). For current information, see:

Windows Server 2012 R2 Licensing Datasheet
http://download.microsoft.com/download/F/3/9/F39124F7-0177-463C-8A08-582463F96C9D/Windows_Server_2012_R2_Licensing_Datasheet.pdf

The purpose of this appendix is not to provide authoritative guidance but to offer a high-level overview of two non-discounted SQL Server licensing examples. The examples are for the architecture characterized here, where all three servers have two 16-core CPUs, for a total of 32 physical cores per host.

For Open Licensing, all physical cores on all replicas must be licensed. At present, the full price for SQL Server 2016 Enterprise Edition license with no discounts and without Software Assurance is $7,128 per core. For the three-replica configuration described here, licensing the 32-core primary for SQL Server 2016 Enterprise Edition costs $228,096 (32 cores * $7,128/core, no discounts, no Software Assurance). So, for the three replicas in this configuration, licensing costs for three, at $228,096/server, is $684,288.

Software Assurance (SA) MSRP is $10,105 per core and provides support for one "passive" server at no additional licensing charge. SA licensing for a 32-core replica for SQL Server 2016 Enterprise Edition is $323,360 (32 cores * $10,105/core, no discounts, with Software Assurance, non-readable secondaries). Because only two servers in this configuration require paid licenses, total licensing costs for the three replicas are 2 servers * $323,360/server = $646,720.

Consult with your licensing professional for additional details, including Open License, Software Assurance, Volume License, and other questions. For current information, see:

SQL Server Licensing
www.microsoft.com/en-us/cloud-platform/sql-server-pricing
or
Licensing Support:
800.426.9400

The table below provides the hardware Bill of Materials. The total comes to $178,908.12 for all three servers, again with no discounts.

Hardware	Quantity	Description	List	Total
HPE Proliant DL380 Gen9				
767032-B21	3	HP DL380 Gen9 24SFF CTO Server	2,107.00	6,321.00
817955-L21	6	HP DL380 Gen9 E5-2697v4 FIO Kit	4,359.00	26,154.00
805351-B21	24	HP 32GB DR x4 DDR4-2400 Kit	719.00	17,256.00
749974-B21	3	HP Smart Array P440ar/2G FIO Controller	599.00	1,797.00
726821-B21	3	HP Smart Array P440/4G FIO Controller PCIe 3.0 x8	533.18	1,599.54

783009-B21	3	HPE SAS Internal Cable Kit	102.92	308.76
719073-B21	3	HP DL380 Gen9 Secondary Riser	99.00	297.00
AF556A	6	HP 1.83m 10A C13-UL US Pwr Cord	10.00	60.00
727250-B21	3	HP 12Gb DL380 Gen9 SAS Expander Card	699.00	2,097.00
764286-B2	3	HP IB QDR/EN 10Gb 2P 544+FLR-QSFP Adptr	995.00	2,985.00
666988-B2	3	HP 2U Security Bezel Kit	49.00	147.00
720863-B2	3	HP 2U SFF BB Rail Kit	119.00	357.00
726116-B21	3	HP 8GB microSD EM Flash Media Kit	79.00	237.00
719082-B21	3	HP DL380 Gen9 Graphics Enablement Kit	119.00	357.00
720620-B2	6	HP 1400W FS Plat Pl Ht Plg Pwr Spply Kit	429.00	2,574.00
720865-B2	3	HP 2U CMA for BB Rail Kit	65.00	195.00
BD505A	3	HP iLO Adv incl 3yr TS U 1-Svr Lic	469.00	1,407.00
768900-B2	3	HP DL380 Gen9 Sys Insght Dsply Kit	129.00	387.00
HPE SSD Storage				
718162-B21	6	HP 1.2TB 6G SAS 10K 2.5in DP ENT SC HDD	879.00	5,274.00
846430-B21	30	800 GB WI-1 HPE 800GB 12G SAS Write Intensive-1 SFF (2.5in)	3,429.00	102,870.00
Networking				
MCX354A-FCCT	6	ConnectX-3 Pro VPI Adapter, Dual-Port QSFP, FDR IB	1,010.47	6,062.82
MAM1Q00A-Q	6	Mellanox Cable Module, Eth 10GBE, 40Gb/s to 10Gb/s, QSFP to SFP+	27.50	165.00
Hardware Grand Total				***$178,908.12***

Table 5. Harware Bill of Matierals (BoM) for three-replica architecture including servers, processors, NICs, storage, and cabling. Note that prices reflect MSRP; discounts are widely available.

Appendix B: Lessons Learned

A number of challenges were encoutered before arriving at the final configuration. Over 150 tests were executed and documented, ranging from 15 minutes to 24 hours. The following items may helpful when duplicating this architecture.

Hardware Validation

Network and storage throughput need to be separately isolated and validated, outside the context of application testing. Doing so prior to full system tests and deployment has long been a best practice. It is often useful for short-circuiting troubleshooting efforts, including in the tests documented here.

Firmware

SSD firmware (version HPD5) is specifically optimized for the HPE P440ar I/O controller. The firmware version is exposed in the HPE Smart Storage Administrator or the HPE `ssacli` command-line utility. Updating SSD firmware boosted test results in terms of transactions/sec by 19.6% and resulted in significantly improved latency, from ms to µsec.

RAID Configuration

Several disk/controller configurations and two different strip/stripe sizes were tested. As documented, the final configuration was a 6-disk RAID 10 for data, with a 4-disk RAID 10 for the log. Both leveraged the default strip/stripe sizes.

Workload Optimization

The TPCC-like workload stored procedure distribution weights were optimized. This eliminated deadlocking and locking, and it maximize CPU utilization and transaction throughput. Startup trace flags `-T1204` and `-T1222` were enabled to expedite deadlock troubleshooting.

Load Injector CPU

Only one injector was necessary in these tests. CPU utilization on the 24-logical core injector was necessary to confirm the absence of a bottleneck, especially during high-performance baselining.

Backup Options

Our work typically leverages aggressive parameters for both database and log backup and restores, such as:

- 8 disk devices
- `MAXTRANSFERSIZE = 4194304 -- (4MB is the max value permitted)`
- `BUFFERCOUNT = 256 -- (guidance: 4x core count for max perf)`

However, *application consistency* was the primary objective, not maximum throughput. HPE SSD storage provided the flexibility to achieve both. Extensive testing was done with database backup and log backup options, to find the optimal balance between backup times. This kept the log sufficiently cleared for reuse (including substantial capacity to accommodate possible downtimes such as maintenance or emergencies). Also important was the need for ultra-low disk latency (especially on the log volume) to provide consistent transaction performance.

Backup tuning resulted in the following scripts, with the best-suited requirements for this architecture under real-world conditions:

```
--database
BACKUP DATABASE OLTP
     TO DISK = 'NUL:'
      , DISK = 'NUL:'
      , DISK = 'NUL:'
      , DISK = 'NUL:'
  WITH MAXTRANSFERSIZE = 65536
     , BUFFERCOUNT       = 64
     , COMPRESSION
     , CHECKSUM
     , STATS             = 10;

--log
BACKUP LOG OLTP
     TO DISK = 'NUL:'
  WITH MAXTRANSFERSIZE = 65536
     , BUFFERCOUNT       = 16
     , COMPRESSION
     , CHECKSUM
     , STATS             = 10;
```

Note the use of the `'NUL:'` backup device in order to stress the HPE SSDs to the fullest extent possible, removing network and destionation I/O as potential bottlenecks. Also, the database backup was assigned four devices, whereas the log backup had only one. The SQL Server default `MAXTRANSFERSIZE` of 64KB was used instead of our typical choice for high performance of 4MB. `BUFFERCOUNT` was also reduced from 256 to 64, and 16 for the database and log backups, respectively. Doing so sacrificed maximum possible backup performance to satisfy our priority of consistent application performance.

For context, a full database backup statement leveraging the maximum backup performance options completes in 119 seconds; the tuned statement above finishes in 158 seconds. The somewhat slower backup performance should be an easy tradeoff for consistent end-user application performance. Testing is recommended to determine the optimal values for your environment.

SAN Offline diskpart Issue

During testing, the environment was torn down and re-created multiple times. During one such effort, SSDs had to be manually brought online after each reboot. Though no SAN was involved in the configuration, the remedy was simply changing the disk's "SAN policy" using `diskpart.exe`:

```
DISKPART> san
SAN Policy: Offline Shared
DISKPART> san policy = onlineall
DiskPart successfully changed the SAN policy for the current operating system.
```

Re-creating AG on New WFSC

Some aggressive means were used in providing a robust solution. Tests often included rebuilding the entire environment from scratch. A PowerShell framework was created to expedite this work. However, after destroying the Windows Server Failover Cluster (WSFC), the framework failed to surface an error which occurs when re-recreating the AG:

```
Msg 41105, Level 16, State 0, Line 112
Failed to create the Windows Server Failover Clustering (WSFC) resource with name
'HADR_AG'
and type 'SQL Server Availability Group'.
The resource type is not registered in the WSFC cluster.
The WSFC cluster may have been destroyed and created again.
To register the resource type in the WSFC cluster, disable and then enable Always On in
the SQL Server Configuration Manager.
```

A manual rebuild exposed the error, and the solution was to simply disable "Enable Always On Availability Groups". This is found in the AlwaysOn High Availability tab of the instance's SQL Server properties, in the SQL Server Configuration Manager dialog (`SQLServerManager13.msc`).

Note: In contrast to SQL Server 2012 and 2014, in SQL Server 2016 RTM uses "Always On" – with a space in the name. This change has not yet been propagated to all product components.

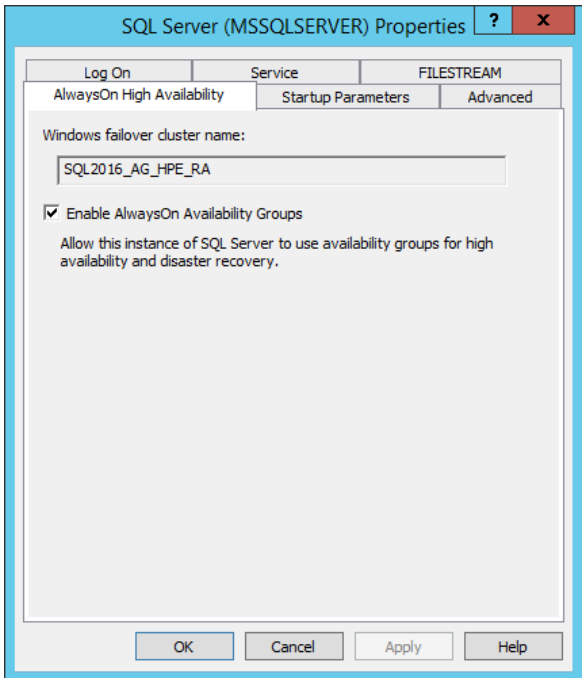


Figure 6. Enabling Always On Availability Groups in SQL Server Configuration Manager properties dialog.

Throttling Checkpoints with the –k Startup Trace Flag

Checkpoint throttling, via the fully supported SQL Server startup trace flag, was used to throttle checkpoint bursts. In the testing, checkpoints were throttled at 750MB/s (`-k750`). Doing so was a key to success, particularly in obtaining the consistent performance profile demanded by enterprise OLTP applications.

`-k` is a fully supported though little-known option. Its purpose is to throttle SQL Server checkpoints, mitigating the impact of flooding that often overwhelms the disk I/O subsystem. Mitigation against checkpoint bursts typically requires massive storage overprovisioning. This trace flag is indispensable for reliable, consistent application performance.

This option specifies the value in MB at which to throttle checkpoint I/O. Testing included values ranging from `-k500`, `-k750`, `-k1000`. Results using `-k750` struck the optimal balance, maximally flushing pages without

negatively impacting I/O performance. The perfmon counter is Checkpoint pages/sec in the instances Buffer Manager performance object; 750MB maps to 96,000 data pages.

Support for `-k` has been built into the product since SQL Server 2008 and was originally announced for SQL Server 2005 here:
https://support.microsoft.com/en-us/kb/929240

`-k` throttles checkpoints at the server level, desirable for this implementation. For information on checkpoints in SQL Server, see this post from Mike Ruthruff of the Microsoft SQL Server Customer Advisory Team (SQL CAT):

Changes in SQL Server 2016 Checkpoint Behavior
https://blogs.msdn.microsoft.com/sqlcat/2016/08/03/changes-in-sql-server-2016-checkpoint-behavior

The post references Indirect Checkpoints, a database-level option that is the default in SQL Server 2016.

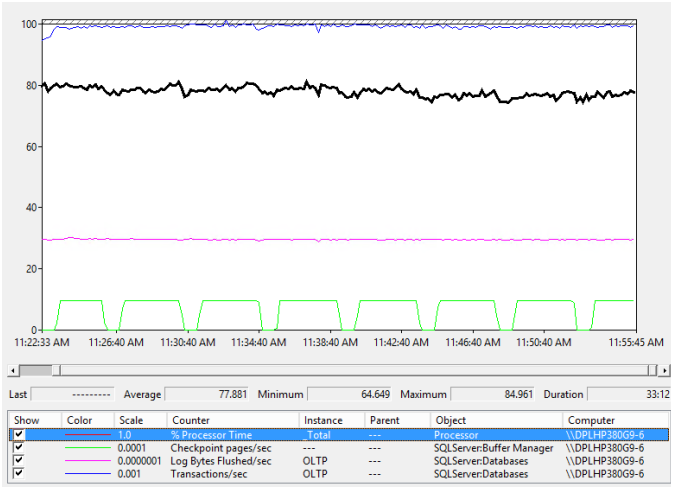
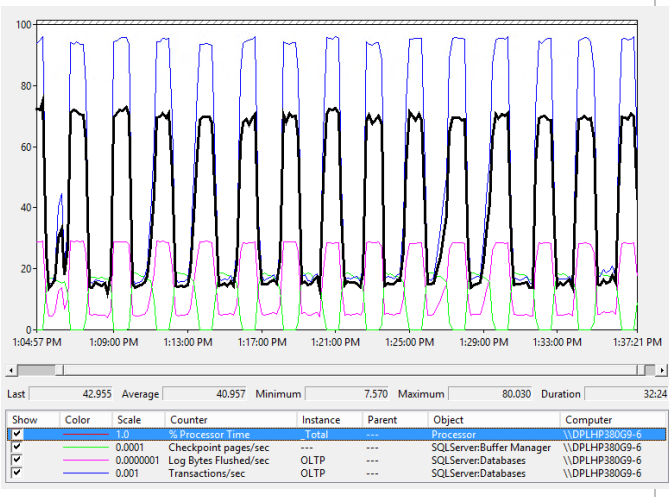
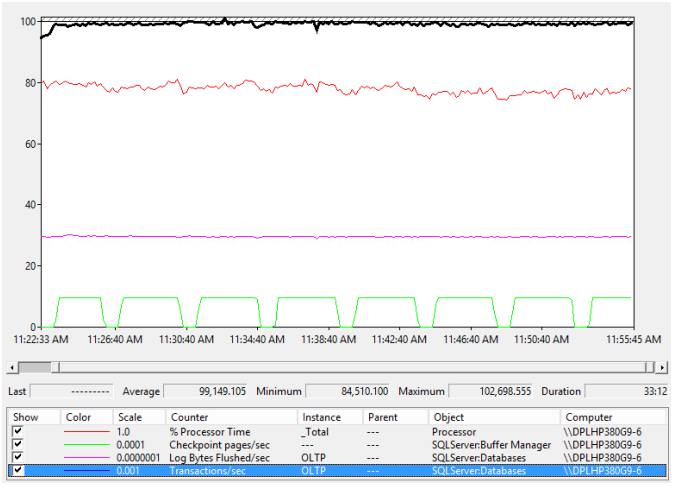
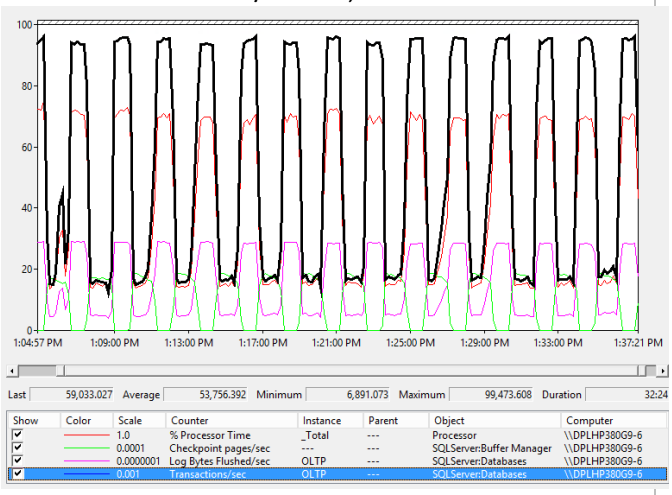
The following table and graphics show the dramatic impact on performance of the default checkpoint behavior, compared to the more consistent behavior provided by checkpoint throttling. SQL Server's default checkpoint behavior leads to wild fluctuations in performance.

Enabling `-k` provides not only the ability to deliver transactions consistently, but also significant improvement in overall performance. Transactions/sec were nearly doubled for the implementation of **`-k750`**: 99,149 vs. 53,756.

	Default Checkpoint Behavior	**`-k750` Startup Trace Flag**
Counter	Avg [min | max]	Avg [min | max]
% Processor Time	41.0 [7.6 | 80.0]	77.9 [64.6 | 85.0]
Transactions/sec	53,756 [6,891 | 99,474]	99,149 [84,510 | 102,699]
Log Bytes Flushed/sec	161,949,800 [21,952,746 | 302,174,397]	296,020,649 [253,760,882 | 307,057,318]
Checkpoint pages/sec	96,324 [0 | 232,286]	73,076 [0 | 96,483]

Table 6. Relevant system metrics required to highlight the benefits of leveraging the `-k` startup trace flag.

Throttling checkpoint/pages per second at a user-defined value prevents the storage system from being overwhelmed, allowing the system to provide consistent performance, especially in terms of business transactions.

-k750: Startup Trace Flag	**Default Checkpoint Behavior**
-k750: % Processor Time: 77.9	Default: % Processor Time: 41.0
	
-k750: Transactions/sec: 99,149	Default: Transactions/sec: 53,756
	

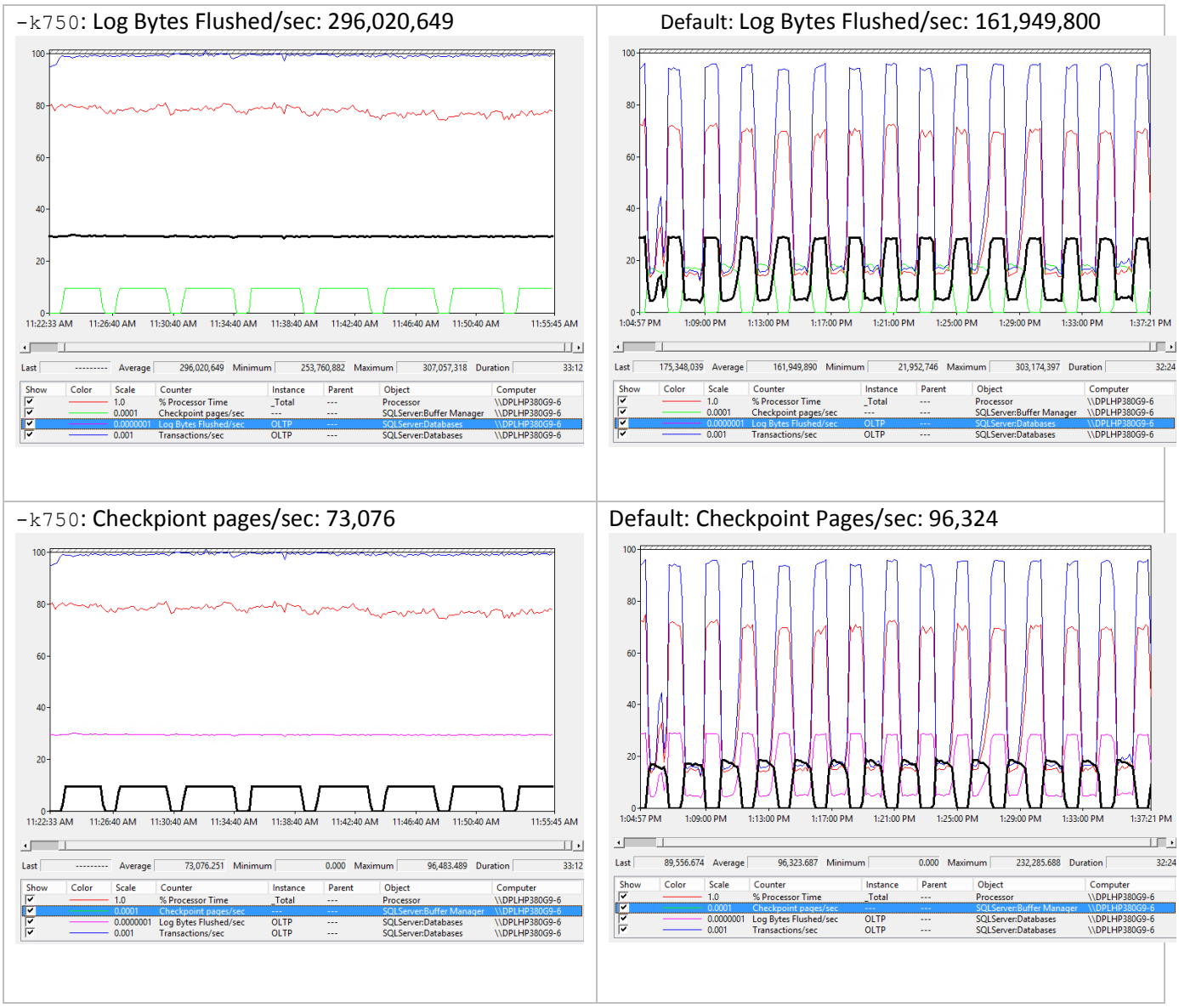


Figure 7. These perfmon metrics graphically demonstrate the benefits of leveraging the `-k` startup trace flag to throttle checkpoint pages at a user-defined value.

As seen in the figures above, the default checkpoint behavior subjects the server and application to wide swings in performance. Contrast this with the behavior provided by `-k`, which allows the system to provide more consistent performance.

Appendix C: SQL Server 2014 vs. 2016 AG Network Transport Enhancements

This work was done previously at the Data Propulsion Laboratory (DPL), demonstrating one of the most significant outcomes in the release of SQL Server 2016, namely AG log transport performance. The log transport is responsible for moving AG log data from the primary replica to secondary replicas.

To appreciate the enhancements in SQL Server 2016 AGs, it's useful to compare performance between the two versions. In these tests, the very same hardware was used for both experiments. The only difference in the protocol is the version of SQL Server.

SQL Server 2016 enables the product to do more of what it's supposed to do. It leverages more processor power (~4x CPU) to drive more workload (>5x transactions/sec), increasing the potential of the storage (~3x – 4x IOPs & throughput), at near-zero latency. SQL Server 2016 delivers far more data across the wire (>4x) for HA/DR (`Log Bytes Received`).

To reiterate: simply upgrading your AG implementation to SQL Server 2016 provides more performance. In SQL Server 2012 and 2014, the SQL Server Log Transport mechanism had a bottleneck that was remediated in SQL Server 2016. For more evidence related to this and other performance enhancements, see It Just Runs Faster, a blog series from Microsoft's Customer Support Services team.

The 6x transactions/sec delta was a function of enhanced Always On AGs—5x log transfer—which correlates to more than ~80% of the delta. Also, the SQL Server log has hard limits coded in: the lesser of 112 outstanding IO's or 3,840KB throughput. HPE SSD flash can better consume the I/O it encounters, thereby mitigating the log's hard-coded limits cited above.

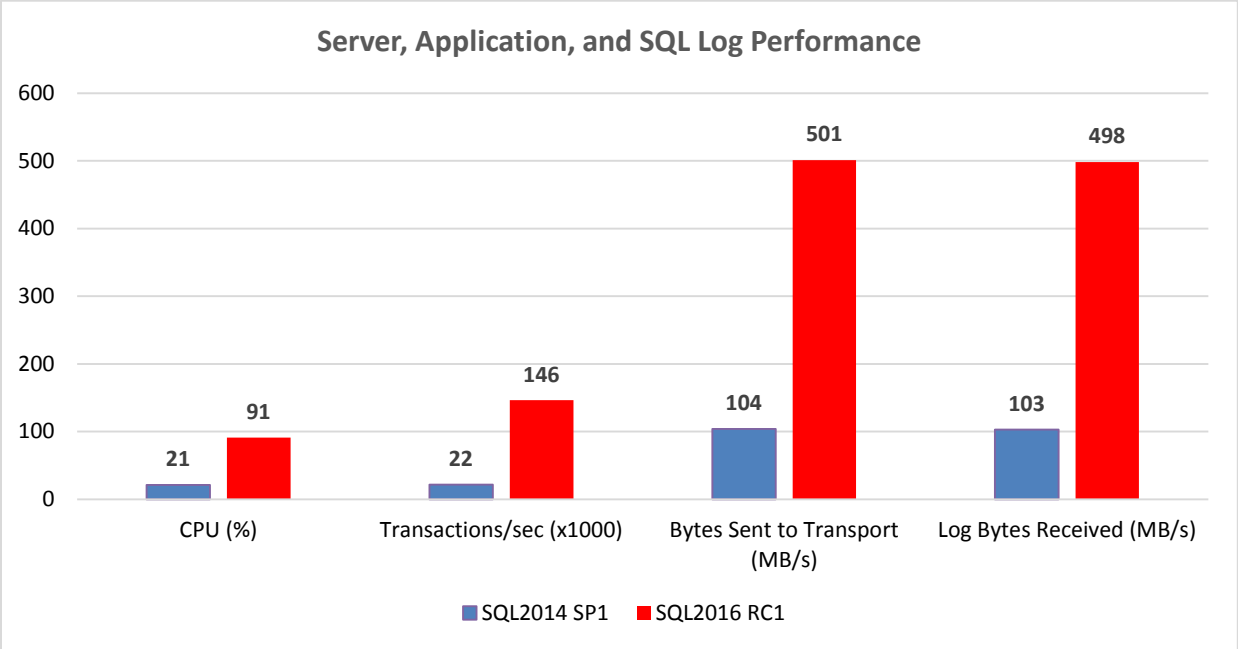


Figure 8. Improved Log Transport mechanism in SQL Server 2016 compared to 2014 provides not only enhanced HA, but also greater hardware utilization. More importantly, higher business application performance is unleashed.

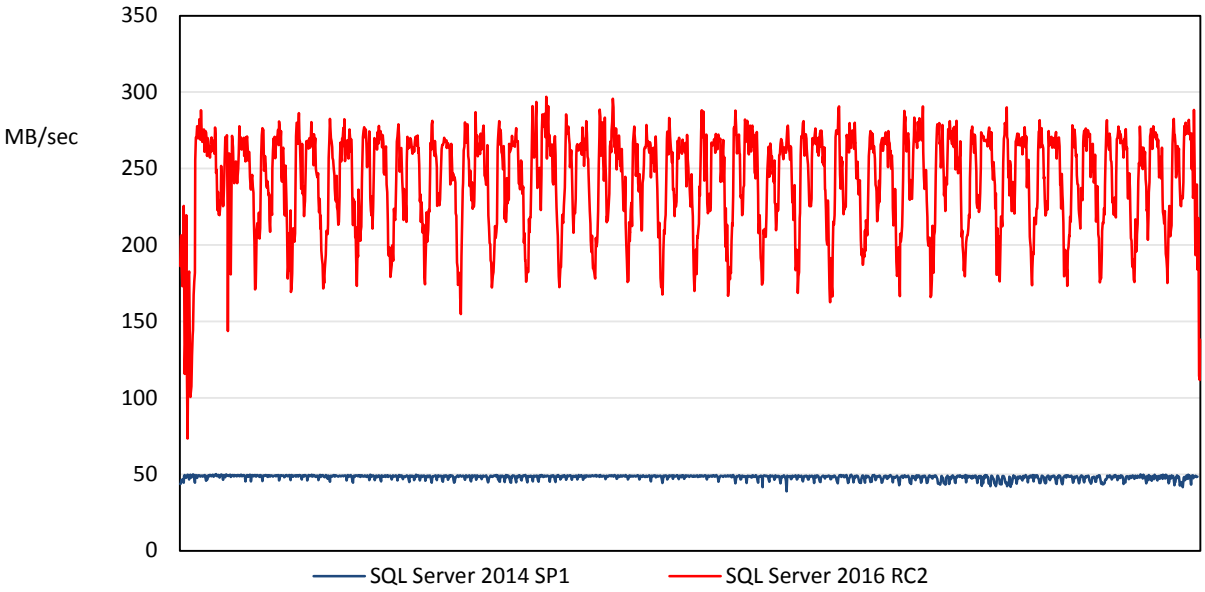


Figure 9. This test reinforces the enhanced network log transport performance for SQL Server 2016 vs. SQL Server 2014.

Appendix D: SQL Server 2016 RTM Log Redo Performance

Performance challenges related to AG log redo were introduced previously in the *SQL Server 2016 AG Redo: The Next Opportunity for Optimization* section of *Architecture Background*. AG log redo is the process used by secondary replicas that takes log data received from the primary replica and applies it from the secondary log to the secondary data files. This could be thought of as a continuous database restore.

Of the two components, log transport was by far the most urgent issue to address: getting the data from the primary to secondary replicas is critical for Recovery Point Objectives (RPO). Performance improvements were also made to AG log redo on secondary replicas. For example, parallel threads are available to apply modifications from the secondary log to database files.

However, limitations to AG secondary log redo remain the system bottleneck. The challenge is that a single-threaded process is still being used to distribute work to the threads that actually implement redo.

In the discussion that follows, counter names are self-explanatory; for additional info, see the following:

- SQL Server, Databases Object
  https://msdn.microsoft.com/en-us/library/ms189883.aspx
- SQL Server, Availability Replica
  https://msdn.microsoft.com/en-us/library/ff878472.aspx
- SQL Server, Database Replica
  https://msdn.microsoft.com/en-us/library/ff878356.aspx

Performance Analysis

In the context of this three-replica configuration, this section examines perfmon metrics from the perspective of the primary replica (DPLHP380G9-1) and one of the two secondary replicas (DPLHP380G9-6). The figures below reflect internal observations.

On the primary, note that `% Processor Time` is an 89.8%, delivering an impressive 71K `Transactions/sec`.

The corresponding `Log Bytes Flushed/sec` is 213 MB/sec. Because there are two replicas, the primary's `Bytes Sent to Transport/sec` is approximately double that total, 428 MB/sec.

The secondary replica `Bytes Received from Replica/sec` aligns well, 213 MB/sec. As expected for an AG with a single database, the OLTP database value for `Log Bytes Received/sec` matches almost exactly, 213 MB/sec.

These values reflect the high performance supported by this configuration and the improvements to SQL Server 2016 AG Log Transport.

There are challenges related to AG log redo. As expected, the AG secondary's `% Processor Time` is lower, here only 7%. The secondary is responsible merely for applying log traffic, hardened on its replica database log, to the database data files. However, the explanation for the secondary's 17K `Transactions/sec`—a fraction of the primary's 71K transaction rate—is because of internal constraints: the SQL Server 2016 RTM log redo cannot restore log traffic as quickly as the data is received from the primary. Specifically, the value for secondary's `Redone Bytes/sec` is only 51 MB/sec. As stated previously, the secondary's `Bytes Received from Replica/sec` is 213 MB/sec. The delta between log traffic received (213 MB/sec) and log data redone (51 MB/sec) is 162 MB/sec. This delta is reflected in the secondary's Recovery Queue average value of 68 million log records, as well as the growth highlighted in the graphic below.

\\DPLHP380G9-1	
Processor	**_Total**
% Processor Time	89.815
SQLServer:Availability Replica	**_Total**
Bytes Sent to Transport/sec	428,072,038.190
SQLServer:Databases	**OLTP**
Log Bytes Flushed/sec	213,433,659.673
Transactions/sec	71,183.190
\\DPLHP380G9-6	
Processor	**_Total**
% Processor Time	7.420
SQLServer:Availability Replica	**_Total**
Bytes Received from Replica/sec	213,632,691.420
SQLServer:Database Replica	**OLTP**
Log Bytes Received/sec	213,435,448.900
Recovery Queue	68,098,496.284
Redone Bytes/sec	51,224,663.369
SQLServer:Databases	**OLTP**
Transactions/sec	16,796.161

Figure 10. Perfmon metrics captured from primary and secondary replicas to contrast AG log transport high performance with AG log redo. The text describes each of these values in detail.

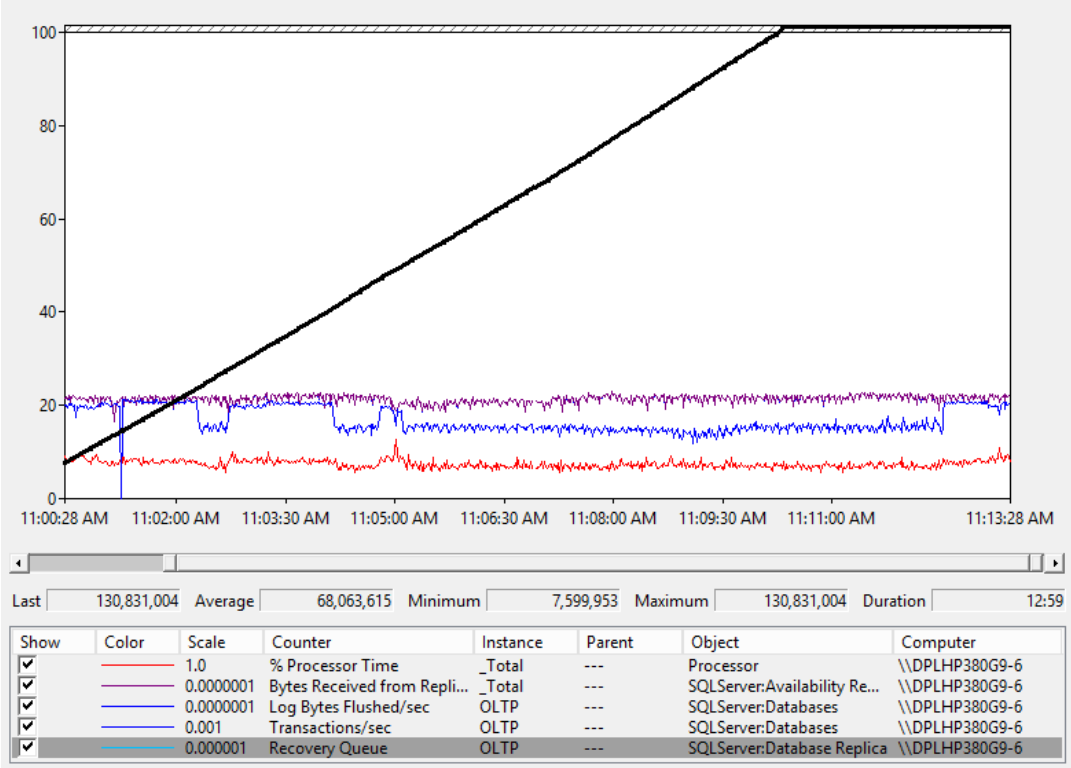


Figure 11. Permon graphic highlighting the growth of the secondary recovery queue when the AG log transport rate exceeds the rate of the AG log redo.

Note: High availability demands that the recovery queue be maintained at or near zero to minimize the time required for AG secondary databases to recover on failover.

The following two figures document processor behavior on the secondary. The arrows in the Task Manager graphic indicate that overall CPU utilization is only 8%. Yet one processor is maxed out at 100%. The subsequent figure shows results from interrogating `sys.sysprocess`. Wait statistics for threads related to AG log redo show the DB STARTUP process (row 1) waiting for CPU cycles, in order to provide work to idle redo threads (rows 2 through 18).

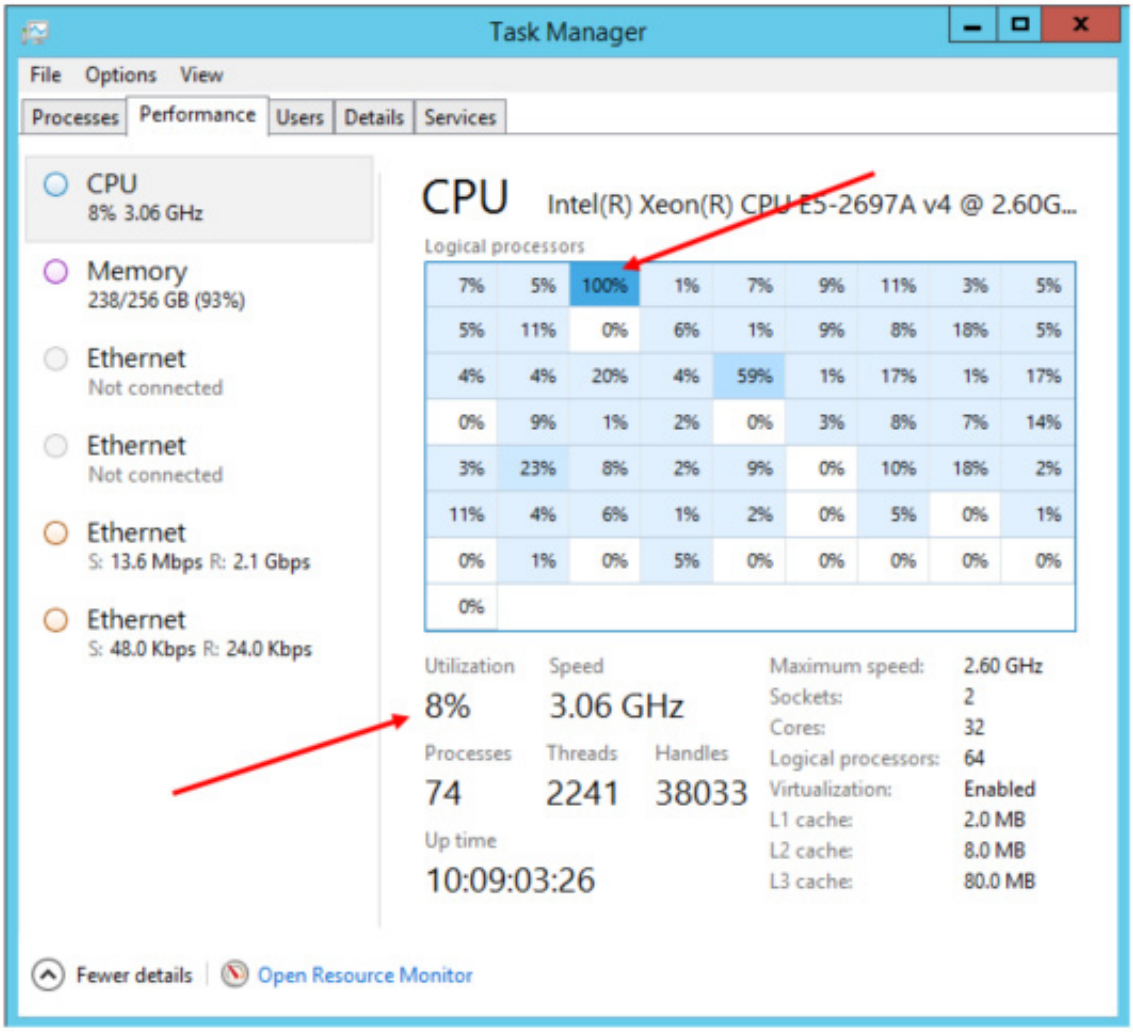


Figure 12. Even though the AG secondary's overall CPU utilization is only 8%, one processor is pegged at 100%. This is hosting the process responsible for distributing work to threads actually implementing AG log redo.

	cmd	spid	kpid	blocked	waittype	waittime	lastwaittype	waitresource	dbid	uid	cpu	physical_io	memusage
1	DB STARTUP	41	7484	0	0x0000	0	SOS_SCHEDULER_YIELD		5	1	476734	1631829	0
2	PARALLEL REDO TA	52	8736	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	7890	4845018	0
3	PARALLEL REDO TA	19	7968	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	7125	5111213	0
4	PARALLEL REDO TA	55	6440	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	5062	4846233	0
5	PARALLEL REDO TA	51	1528	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4953	4834648	0
6	PARALLEL REDO TA	63	7268	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4734	4841183	0
7	PARALLEL REDO TA	59	7636	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4625	4829267	0
8	PARALLEL REDO TA	53	6904	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4562	5120193	0
9	PARALLEL REDO TA	64	8024	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4437	4841504	0
10	PARALLEL REDO TA	57	9728	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	4312	5108502	0
11	PARALLEL REDO TA	54	6624	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3921	5095310	0
12	PARALLEL REDO TA	65	3824	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3734	5109023	0
13	PARALLEL REDO TA	60	7688	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3718	4838362	0
14	PARALLEL REDO TA	56	9808	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3718	4845424	0
15	PARALLEL REDO TA	58	9896	0	0x046F	3	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3359	5106725	0
16	PARALLEL REDO TA	61	10196	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3266	5096338	0
17	PARALLEL REDO TA	62	10044	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	3187	5090060	0
18	PARALLEL REDO HE	66	10004	0	0x046F	4	PARALLEL_REDO_WORKER_WAIT_WORK		0	1	281	0	0

Figure 13. Leveraging sys.sysprocesses to view wait statistics for threads related to AG log redo.

In the figure above, note the DB STARTUP processes waiting for CPU cycles in order to provide work to idle redo threads.

The SQL Server product team is working to remediate the AG log redo bottleneck. In the meantime, the *Solution Configuration: Options and Details* section provides insight into viable workarounds, many of which have been implemented in production by customers.

Appendix E: Availability Group Code Samples

```
/*--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--
    --The following Transact-SQL code provides guidance for creating this configuration
characterized in this document
    --This is NOT a comprehensive solution & no pretext is made re: completeness
    --However, the reader may find it helpful to use as a basis for testing AGs in their own
environment.
    --Server & object names, IPs, ports, etc. will require customization
    --Weasel Clause: This script is provided "AS IS" with no warranties, and confers no rights
    --jimmy.may@sandisk.com :: aspiringgeek@live.com :: https://twitter.com/aspiringgeek
--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--*/

--NOTE: ENABLE SQLCMD mode: SSMS menu > Query > SQLCMD

/*
CREATE AVAILABILITY GROUP (Transact-SQL)
    https://msdn.microsoft.com/en-us/library/ff878399.aspx
ALTER AVAILABILITY GROUP (Transact-SQL)
    https://msdn.microsoft.com/en-us/library/ff878601.aspx
Configure Availability Group to Communicate on a Dedicated Network
    http://blogs.msdn.com/b/alwaysonpro/archive/2013/11/01/configuring-a-dedicated-network-for-
replicating-changes-from-primary-to-secondary-replica.aspx
*//* --<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--
--Deploying AG
   0a. Enable Always On
   0b.  Create mirroring endpoint

   0. Destroy existing AG(s)
   1.  Stage Replica DBs
   2.  Create AG & add secondary replica instance(s)
   3.  ALTER db...SET HADR...
   4.  Create AG Listener (out of scope)
   5.  Validate AG health
*/--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--

/*--0a. Enable Always On Availability Groups
    No T-SQL command exists to perform this operation
    PowerShell (not shown)
    Manual method
        1. Open SQL Server 2016 Configuration Manager (SQLServerManager13.msc)
        2. Expose SQL Server Services, then the Properties dialog of the target instance
        3. Click "AlwaysOn High Availability" (sic)
        4. Enable the "AlwaysOn Availability Groups" checkbox (sic); click OK
        5. Restart the service
*/--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--

--0b. create mirroring endpoint on primary replica & all secondary replicas
--scrutinize endpoint state
    --SELECT name, state_desc, port FROM sys.tcp_endpoints WHERE name = 'HADR_AG_EP';
    --SELECT * FROM sys.tcp_endpoints WHERE name = 'HADR_AG_EP';
--start a stopped endpoint
    --ALTER ENDPOINT HADR_AG_EP STATE = STARTED;
:CONNECT DPLHP380G9-1
SET NOCOUNT ON;
--SELECT @@SERVERNAME, * FROM sys.endpoints;
```

```
IF NOT EXISTS (SELECT * FROM sys.endpoints WHERE type_desc = 'DATABASE_MIRRORING')
BEGIN
CREATE ENDPOINT HADR_AG_EP
    STATE = STARTED
    AS TCP ( LISTENER_PORT = 5023
           , LISTENER_IP   = (192.168.18.1) --ip of dedicated network
           )
    FOR DATABASE_MIRRORING
      ( AUTHENTICATION = WINDOWS NEGOTIATE  --WINDOWS NTLM  | WINDOWS KERBEROS
      , ROLE           = ALL                --WITNESS | PARTNER | ALL
      , ENCRYPTION     = REQUIRED ALGORITHM AES    --OFF
      );
END
SET NOCOUNT ON; SELECT @@SERVERNAME, * FROM sys.endpoints;
GO

--endpoint for first secondary replica
:CONNECT DPLHP380G9-4
SET NOCOUNT ON;
--SELECT @@SERVERNAME, * FROM sys.endpoints;
IF NOT EXISTS (SELECT * FROM sys.endpoints WHERE type_desc = 'DATABASE_MIRRORING')
BEGIN
CREATE ENDPOINT HADR_AG_EP
    STATE = STARTED
    AS TCP ( LISTENER_PORT = 5023
           , LISTENER_IP   = (192.168.18.2) --ip of dedicated network
          )
    FOR DATABASE_MIRRORING
      ( AUTHENTICATION = WINDOWS NEGOTIATE  --WINDOWS NTLM  | WINDOWS KERBEROS
      , ROLE           = ALL                --WITNESS | PARTNER | ALL
      , ENCRYPTION     = REQUIRED ALGORITHM AES    --OFF
      );
END
SET NOCOUNT ON; SELECT @@SERVERNAME, * FROM sys.endpoints;
GO

--endpoint for additional secondary replica
:CONNECT DPLHP380G9-6
SET NOCOUNT ON;
--SELECT @@SERVERNAME, * FROM sys.endpoints;
IF NOT EXISTS (SELECT * FROM sys.endpoints WHERE type_desc = 'DATABASE_MIRRORING')
BEGIN
CREATE ENDPOINT HADR_AG_EP
    STATE = STARTED
    AS TCP ( LISTENER_PORT = 5023
           , LISTENER_IP   = (192.168.18.3) --ip of dedicated network
           )
    FOR DATABASE_MIRRORING
      ( AUTHENTICATION = WINDOWS NEGOTIATE  --WINDOWS NTLM  | WINDOWS KERBEROS
      , ROLE           = ALL                --WITNESS | PARTNER | ALL
      , ENCRYPTION     = REQUIRED ALGORITHM AES    --OFF
      );
END
SET NOCOUNT ON; SELECT @@SERVERNAME, * FROM sys.endpoints;
GO
--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--
```

```
--0. destroy AG (execute from primary)
:CONNECT DPLHP380G9-1
SELECT @@SERVERNAME as servername , * FROM sys.availability_groups;
GO
DROP AVAILABILITY GROUP HADR_AG;
GO
SELECT @@SERVERNAME as servername , * FROM sys.availability_groups;
GO
--check status on primary & secondaries
    --note: any pending redo will delay AG drop on secondaries
    --workaround: stop SQL service on each node, delete physical db files, restart SQL services
WAITFOR DELAY '00:00:15';
GO
:CONNECT DPLHP380G9-1
SELECT @@SERVERNAME as servername , * FROM sys.availability_groups;
GO
:CONNECT DPLHP380G9-4
SELECT @@SERVERNAME as servername , * FROM sys.availability_groups;
GO
:CONNECT DPLHP380G9-6
SELECT @@SERVERNAME as servername , * FROM sys.availability_groups;
GO
--<<<<<<<<<<----------------------------------------------------------------------->>>>>>>>>>--

--1. stage replica dbs
    --restore dbs from backup
    --on primary WITH RECOVERY
    --on secondary(ies) WITH NORECOVERY
--<<<<<<<<<<----------------------------------------------------------------------->>>>>>>>>>--

--2. Create AG & add secondary replica instance(s)
:CONNECT DPLHP380G9-1
SELECT @@SERVERNAME as ServerName;
IF NOT EXISTS (SELECT * FROM sys.availability_groups WHERE name = 'HADR_AG')
BEGIN
CREATE AVAILABILITY GROUP HADR_AG
  FOR DATABASE OLTP
  REPLICA ON
    'DPLHP380G9-1'
    WITH  ( ENDPOINT_URL      = N'TCP://192.168.18.1:5022'
          , AVAILABILITY_MODE = SYNCHRONOUS_COMMIT --SYNCHRONOUS_COMMIT | ASYNCHRONOUS_COMMIT
          , FAILOVER_MODE     = AUTOMATIC           --AUTOMATIC | MANUAL --async requires manual
          , BACKUP_PRIORITY   = 50
          , SECONDARY_ROLE (ALLOW_CONNECTIONS = NO) --<get options>
          )
  , 'DPLHP380G9-4'
    WITH  ( ENDPOINT_URL      = N'TCP://192.168.18.2:5022'
          , AVAILABILITY_MODE = SYNCHRONOUS_COMMIT --SYNCHRONOUS_COMMIT | ASYNCHRONOUS_COMMIT
          , FAILOVER_MODE     = AUTOMATIC          --AUTOMATIC | MANUAL --async requires manual
          , BACKUP_PRIORITY   = 50
          , SECONDARY_ROLE (ALLOW_CONNECTIONS = NO) --<get options>
          )
  , 'DPLHP380G9-6'
    WITH  ( ENDPOINT_URL      = N'TCP://192.168.18.3:5022'
          , AVAILABILITY_MODE = SYNCHRONOUS_COMMIT --SYNCHRONOUS_COMMIT | ASYNCHRONOUS_COMMIT
```

```
          , FAILOVER_MODE     = AUTOMATIC          --AUTOMATIC | MANUAL --async requires manual
          , BACKUP_PRIORITY   = 50
          , SECONDARY_ROLE (ALLOW_CONNECTIONS = NO) --<get options>
          )          ;
END
SELECT @@SERVERNAME as ServerName, * FROM sys.availability_groups;
GO
:CONNECT DPLHP380G9-4
SELECT @@SERVERNAME as ServerName;
ALTER AVAILABILITY GROUP HADR_AG JOIN;
SELECT @@SERVERNAME as ServerName, * FROM sys.availability_groups;
GO
:CONNECT DPLHP380G9-6
SELECT @@SERVERNAME as ServerName;
ALTER AVAILABILITY GROUP HADR_AG JOIN;
SELECT @@SERVERNAME as ServerName, * FROM sys.availability_groups;
GO
WAITFOR DELAY '00:00:05';

--<<<<<<<<<<----------------------------------------------------------------------->>>>>>>>>>--
--3. ALTER db...SET HADR... for secondary replicas
--validate readiness of secondary
:CONNECT DPLHP380G9-4
SELECT
    ( SELECT member_state FROM master.sys.dm_hadr_cluster_members
        WHERE member_name = serverproperty('ComputerNamePhysicalNetBIOS')
    )       as member_state__sys_dm_hadr_cluster_members
  ,   1 as expected
  , ( SELECT SERVERPROPERTY('IsHadrEnabled')) as IsHadrEnabled
  ,   1 as expected
  , ( SELECT state FROM master.sys.database_mirroring_endpoints) as endpoint_state
  ,   0 as expected;
GO
--add db to AG
:CONNECT DPLHP380G9-4
ALTER DATABASE OLTP
    SET HADR AVAILABILITY GROUP = HADR_AG;
GO

--validate readinsess of additional secondary
:CONNECT DPLHP380G9-6
SELECT
    ( SELECT member_state FROM master.sys.dm_hadr_cluster_members
        WHERE member_name = serverproperty('ComputerNamePhysicalNetBIOS')
    )       as member_state__sys_dm_hadr_cluster_members
  ,   1 as expected
  , ( SELECT SERVERPROPERTY('IsHadrEnabled')) as IsHadrEnabled
  ,   1 as expected
  , ( SELECT state FROM master.sys.database_mirroring_endpoints) as endpoint_state
  ,   0 as expected;
GO
--add db to AG
:CONNECT DPLHP380G9-6
ALTER DATABASE OLTP
    SET HADR AVAILABILITY GROUP = HADR_AG;
GO
```

```
--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--
--4. Create AG Listener (out of scope)
    --Create or Configure an Availability Group Listener (SQL Server)
    --https://msdn.microsoft.com/en-us/library/hh213080.aspx

--<<<<<<<<<<----------------------------------------------------------------->>>>>>>>>>--

WAITFOR DELAY '00:01:00';
GO

--5. Validate AG health
:CONNECT DPLHP380G9-1
SELECT primary_replica, primary_recovery_health_desc, synchronization_health_desc
    FROM sys.dm_hadr_availability_group_states;

SELECT ag.name, ar.replica_server_name, drs.is_primary_replica
     , adc.database_name, drs.synchronization_health_desc, drs.synchronization_state_desc
     , drs.log_send_rate, drs.log_send_queue_size, drs.redo_queue_size, drs.redo_rate
    FROM sys.availability_databases_cluster as adc
        JOIN sys.availability_groups as ag ON adc.group_id = ag.group_id
        JOIN sys.dm_hadr_database_replica_states as drs ON adc.group_id = drs.group_id
        JOIN sys.availability_replicas as ar ON drs.replica_id = ar.replica_id
    ORDER BY is_primary_replica DESC;
GO

/*
--see these & other DMVs
    --Monitor Availability Groups (Transact-SQL)
    --https://msdn.microsoft.com/en-us/library/ff878305.aspx
SELECT * FROM sys.availability_databases_cluster as adc
SELECT * FROM sys.availability_groups as ag
SELECT * FROM sys.dm_hadr_database_replica_states as drs
SELECT * FROM sys.availability_replicas as ar
*/
```