by Michael Otey

JUNE 2012

The Essential Guide to Optimizing, Scaling and Securing Microsoft SQL Server 2012

SQL Server is Microsoft's premier relational and analytical data platform and the importance of SQL Server as a core IT infrastructure component continues to grow with each new release of SQL Server. The new SQL Server 2012 release provides many new features that you can use to optimize and secure your mission critical SQL Server database workloads. In this Essential Guide you'll learn how AlwaysOn Availability Groups can be used to increase the availability of your mission critical databases as well as increase the ROI of your database servers and improve enterprise scalability by distributing your reporting and backup workloads to multiple SQL Server instances. In addition, you'll learn how the new xVelocity column store index can be used to provide up to 100X performance improvements for data warehousing workloads. SQL Server 2012 also provides several important new security enhancements. You'll see how SQL Server 2012 can be installed on Windows Server Core minimizing the system overhead required as well as reducing the potential attack surface and the need to patching. Server level auditing has also been expanded to all editions of SQL Server 2012 and user defined server roles have been added. You'll also learn how the new Contained Databases feature can be used to optimize your SQL Server deployment by enabling databases to be moved between SQL Server instances. In addition, you'll see how Citrix NetScaler integrates with SQL Server to improve application performance, availability and security.

AlwaysOn Availability Groups

One of the most important new features in SQL Server 2012 that enables businesses to

SQL Server PRO

CITRIX®

Special Advertising Supplement to SQL Server Pro magazine

Sponsored by CITRIX

optimize their SQL Server installations and increase the availability of their mission critical applications is the new AlwaysOn Availability Groups. The AlwaysOn brand actually refers to two separate but related technologies: AlwaysOn Failover Clustering and AlwaysOn Availability Groups.

AlwaysOn Failover Clustering is essentially the same as SQL Server Failover Clustering which has been available in previous releases. AlwaysOn Failover Clustering is designed to address the issue of unplanned downtime. With AlwaysOn Failover Clustering you install an instance of SQL Server on one of the nodes of Windows Failover cluster. If that node fails then the SQL Server services will be automatically moved to another node in the cluster.

The new SQL Server AlwaysOn Availability Groups is essentially the next generation of database mirroring. SQL Server 2012 AlwaysOn Availability Groups addresses all the main limitations that are found in Database Mirroring. AlwaysOn Availability Groups provide support for four replicas where each replica is located on a

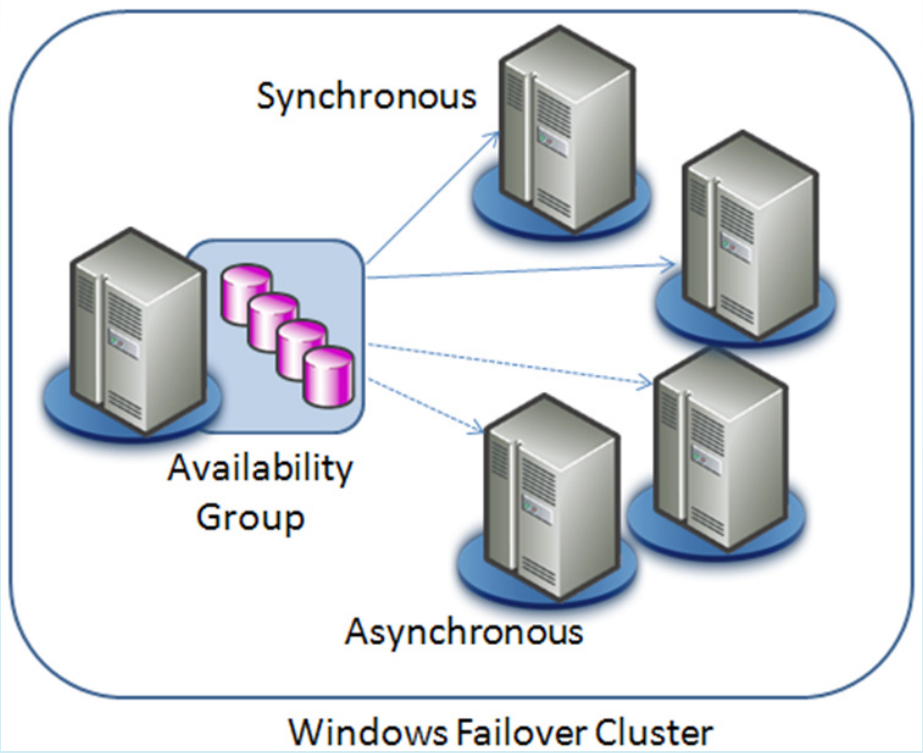

Figure 1 - An Overview of AlwaysOn Availability Groups

separate SQL Server instance running on different Windows Failover Cluster nodes. More importantly, AlwaysOn Availability Groups can contain multiple databases all of which can be automatically failed over as a unit. This means that AlwaysOn Availability Groups can protect multiple related databases and fail them over simultaneously. Another important advantage with AlwaysOn is that you don't have to choose between the asynchronous mode and synchronous mode like you needed to do with Database Mirroring. AlwaysOn Availability Groups can have both synchronous and asynchronous replicas at the same time. Synchronous connections are typically used in high availability scenarios where there is automatic failover. Asynchronous connections are typically used in disaster recovery scenarios where there is geographical distance between the different servers. AlwaysOn Availability Groups enable you to have both types on protection simultaneously and can support a maximum of two synchronous replicas. In addition, with AlwaysOn Availability Groups the replica databases are able to provide read-only access. This enables the replicas to be used for both reporting as well as backup purposes potentially offloading some of the workload and I/O from the primary server. You can see an overview of SQL Server 2012's AlwaysOn Availability Groups in Figure 1.

Clients connect to the Availability Group via the Availability Group Listener. The Availability Group Listener is designed to allow fast application failover after an availability group fails over by redirecting the application to one of the secondary replicas. The listener is configured as a DNS entry and it initially points to the primary replica. In the event of a failover the entry is updated to point to the secondary replica. Client connections use the Availability Group Listener name in their connection strings to connect to the databases in the

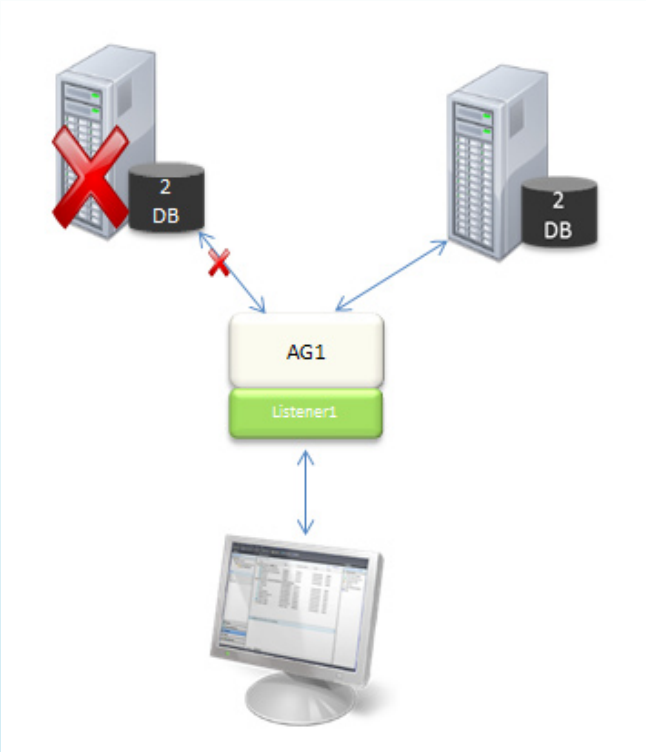

Figure 2 – Connecting to the Availability Group Listener

Availability Group. You can see the role of the Availability Group Listener in Figure 2.

Requirements for Implementing AlwaysOn Availability Groups

On the hardware side AlwaysOn Availability Groups requires at least two server systems and can work with as many as four systems. These can be either physical server systems or they can be VMs. In addition, AlwaysOn Availability Groups requires Windows Failover Clustering using Windows Server 2008 or 2008 R2 Enterprise Edition or higher. The Windows Server 2008 / R2 Standard edition doesn't support Failover Clustering. In addition you need to use the SQL Server 2012 Enterprise edition to get support for AlwaysOn Availability Groups.

The requirement for Windows Failover Clustering means additional complexity over database mirroring. Fortunately, Windows Failover Clustering became much easier to setup beginning with Windows Server 2008. It's important to note that while AlwaysOn Availability Groups require a Windows Failover Cluster they do not require installing SQL Server as a clustered application. Installing a SQL Server instance on the cluster is AlwaysOn Failover Clustering. Instead, with AlwaysOn Availability Groups standalone instances of SQL Server are running on the different cluster nodes. A basic overview of the steps to set up SQL Server 2012 with AlwaysOn Availability Groups on a Windows Failover Cluster follows:

1. On all nodes use Server Manager to install the Windows Failover Cluster Feature
2. Use the Failover Cluster Manager to create a new Windows Failover Cluster
3. Install a new standalone instance of SQL Server on each cluster node
4. Enable AlwaysOn High Availability for the SQL Server Service (MSSQLSERVER)
5. Use the New Availability Group Wizard to configure one or more Availability Groups
6. Specify an Availability Group Listener
7. Perform the initial database synchronization

Restrictions

There are several restrictions regarding which databases can be a part of an Availability Group. Availability Groups can only be created with user databases. Systems

databases cannot be used. The database must be read-write. Read-only databases are not supported. The database must be a multiuser databases and it must not use the AUTO_CLOSE feature. In addition to these restrictions there are a few additional basic requirements. All databases participating in Availability Groups must use the full recovery model and there must be a full backup of the database. A given database can only be in a single Availability Group and the database cannot be configured to use database mirroring. It is also recommended that the file path used by the database file be the same on the primary and the secondary severs.

Managing Availability Groups

After the Availability Group has been configured you can manage it using T-SQL, PowerShell or the SQL Server Management Studio. To manage Availability Groups in SQL Server Management Studio connect Object Explorer to the SQL Server Database Engine in either the primary or one of the secondary replicas then expand the Management node then the Availability Groups node. In Figure 3 you can see the new AlwaysOn Availability Group shown in the SQL Server Management

The Availability Replicas node lists all the primary and secondary replicas. The Availability Databases node lists the databases that are included in the Availability Group. And the Availability Group Listeners node displays the listeners that have been configured for the Availability Group. Right clicking on the nodes and the individual Availability Group components displays different context menus that enable you to interact with the Availability Group.

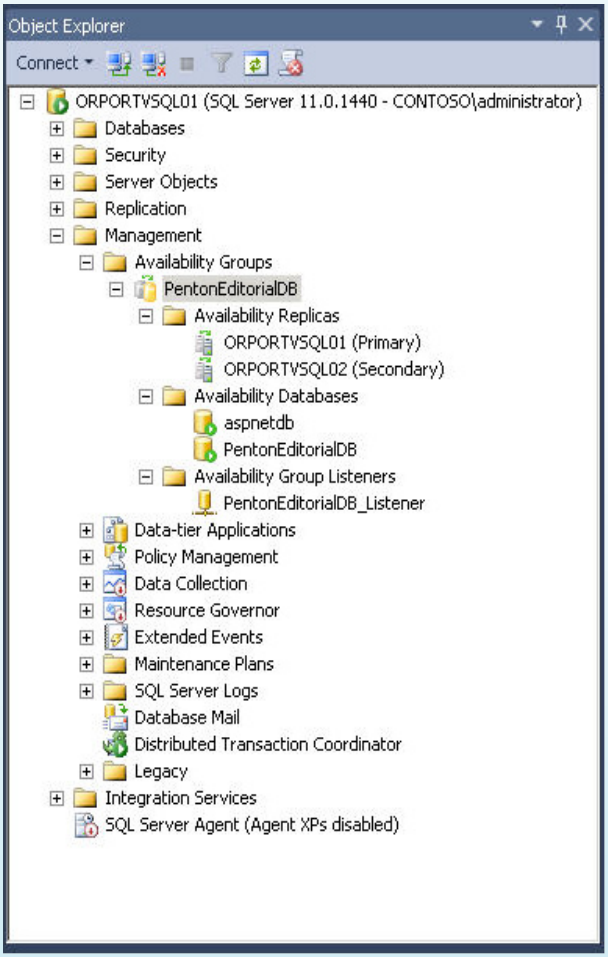

Figure 3 - Managing Availability Groups in SQL Server Management Studio

xVelocity Column Store Index

One of the most significant performance and scalability enhancements in SQL Server 2012 is the addition of the new xVelocity technology. xVelocity is a memory optimized column store index that is intended to increase the performance of data warehousing queries. xVelocity is built on the VertiPaq in-memory data compression technology that Microsoft first introduced with PowerPivot for Excel with the SQL Server 2008 R2 release. For SQL Server 2012 Microsoft has taken this in-memory data compression capability and moved it into the relational database engine. Microsoft reports the xVelocity can provide queries speed improvements by 4X, 10X or even 100X under certain conditions. Figure 4 shows the results of xVelcocity benchmark testing be Microsoft where a set of complex data warehousing queries were run. The first set of tests shown by the blue bars was performed without using xVelocity. The second set of tests shown by the red bars showed the query performance when using xVelocity columnstore indexes. In most cases the use of the new xVelocity indexes resulted in a significant performance improvement for the data warehousing queries.

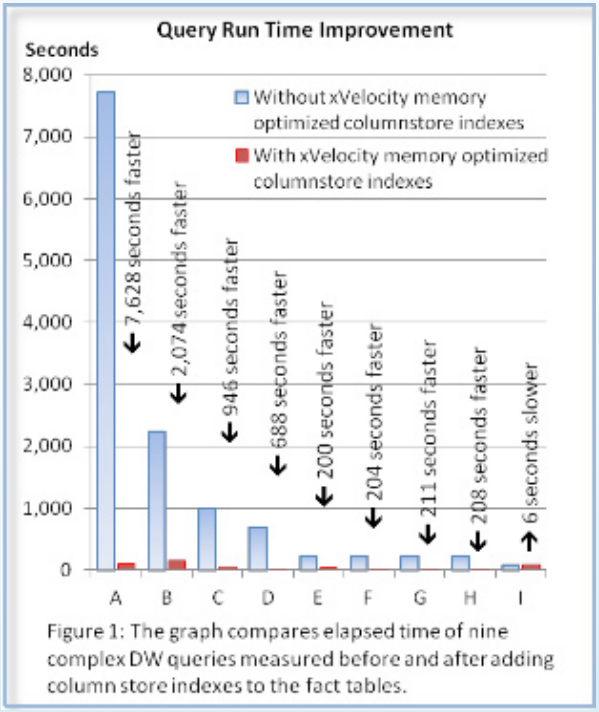

Figure 4 – Performance benchmarks for DW queries using xVelocity

xVelocity columnstore indexes are optimized for typical data warehousing queries that utilize large fact tables and moderate to small dimension tables joined in a star schema where the data is grouped and aggregated. Unlike traditional indexes which store and sort data row-wise xVelocity indexes store data column-wise. Internally, xVelocity columnstore indexes group and store data one column at a time. They can provide significant performance improvements because only the columns that are needed to satisfy the query must be read. This results in less data being read from disk. In addition, xVelocity indexes utilize extreme data compression which reduces the number of bytes that need to be pro-

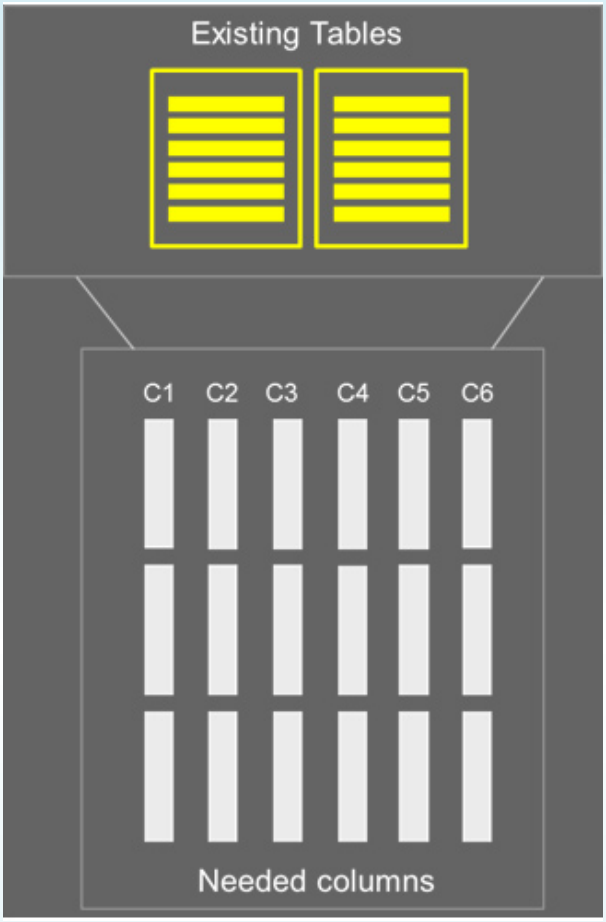

Figure 5 – An overview of the xVelocity columnstore index

cessed. Advanced query execution technology enables SQL Server to process data from xVelocity indexes in chunks of columns called batches which results in reducing CPU usage. You can see an overview an xVelocity columnstore index in Figure 5.

xVelocity columnstore index are created very much like regular indexes using the CREATE INDEX statement. The primary difference is the use of the new COLUMNSTORE directive. You can see how to create an xVelcoity columnstore index in the code listing below.

```
CREATE NONCLUSTERED COLUMNSTORE INDEX
mycolumnstoreindex ON mytable (col1,
col2, col3);
```

The COLUMNSTORE directive tells SQL Server that this will be a columnstore index rather than a standard index. Then you supply a name of the index as well as defining the table and columns that will be used.

Requirements and Restrictions

The xVelocity technology does not have any particular hardware or operating system requirements. However, it is only available in the SQL Server 2012 Enterprise edition. In addition, there are several restrictions in using the xVelocity columnstore index. First, it's important to realize that the xVelocity columnstore index is a data warehousing / BI / reporting feature. It is not meant for OLTP operations. Underlining this idea is the fact that tables that have a columnstore index cannot be updated. In order to update the base table the columnstore index must be dropped. In addition, tables cannot have more than 1024 columns. Columnstore indexes cannot be clustered and they cannot be a unique index nor can they be created over a View or Indexed View. They also cannot include spare columns and they cannot

act as a primary or foreign key. There are also restrictions on the data types that can be used. The following data types cannot be used in a columnstore index:

- binary and varbinary
- FILESTREAM
- ntext, text, and image
- varchar(max) and nvarchar(max)
- uniqueidentifier
- rowversion (and timestamp)
- sql_variant
- decimal (and numeric) with precision greater than 18 digits
- datetimeoffset with scale greater than 2
- CLR types (hierarchyid and spatial types)
- xml

Running SQL Server 2012 on Windows Server Core

One of the most important new security features in the SQL Server 2012 release is the ability to run SQL Server 2012 on Windows Server Core. Windows Server Core is the ideal platform for backend infrastructure applications like SQL Server. Windows Server Core provides all the vital Windows Server infrastructure services that are found in the full installation of Windows Server but without the overhead of the graphical management shell which you don't really want or need on a backend server. Running on the leaner Windows Server Core requires less overhead which makes it a more efficient server platform but more importantly it is also more secure because it has a smaller attack vector and requires significantly less patching. Microsoft has stated that Windows Server Core reduces the need for patching by almost 60% over a full Windows Server installation.

Today the new release of SQL Server 2012 will run on a Windows Server 2008 R2 Server Core installation. In the future it will also be supported on the Windows Server 2012 implementation of Server Core. The Server Core implementation that's coming in Windows Server 2012 will be significantly enhanced over the current Windows Server 2008 R2 implementation. With Windows Server 2008 R2 you have to choose either a full installation or a Server Core installation and you can't easily change between them. The upcoming Windows Server 2012 Server Core will be feature that you can easily switch on and off.

Installing SQL Server 2012 on Server Core

Like you might guess setting up SQL Server 2012 isn't like installing SQL Server on a full Windows Server because Server Core does not support the graphical end user interface that's used by the SQL Server Installation Center.

1. Perform the basic Server Core configura-

tion with sConfig. If this is the first time you've used the Server Core system you'll need to use options 8) Network Settings, 1) Domain/Workgroup then 2) Computer Name to perform the basic server setup. Next use option 4) Configure Remote followed by 2) Enable Windows PowerShell

2. Enable PowerShell and the .NET Framework on Server Core. PowerShell and the .NET Framework are both required by SQL Server 2012. To enable PowerShell 2.0 and the .NET Framework 4.0 run the following commands from the Windows Server Core command prompt:

```
DISM /Online /Enable-Feature /
FeatureName: NetFx2-ServerCore
DISM /Online /Enable-Feature /
FeatureName: NetFx3-ServerCore
DISM /Online /Enable-
Feature /FeatureName:
MicrosoftWindowsPowerShell
dotnefFx40_Full_x86_x64_SC.exe /
passive /promptrestart
```

3. Open the Server Core firewall ports for SQL Server using netsh. The netsh tool allows you to configure the Windows firewall from the command prompt. To open the port required by SQL Server run the following commands in the Server Core command prompt:

```
netsh firewall set portopening TCP
1433 "SQLServer"
netsh firewall set portopening TCP
1434 "SQL Admin Connection"
```

Depending on the SQL Server features that you're using you may need to open more ports. For more information you can refer to:http://msdn.microsoft.com/en-us/library/cc646023.aspx.

5. Run the SQL Server command-line installation. In addition to the SQL Server Installation Center you can also run the SQL Server setup program using the command line setup.exe program. The setup.exe program is able to run unattended and it takes a number of different parameters that control the SQL Server installation options. Run the following command for the Server Core command prompt:

```
Setup.exe /qs /ACTION=Install /
FEATURES=SQLEngine,Replication /
INSTANCENAME=MSSQLSERVER /
SQLSVCACCOUNT="<DomainName\
UserName>" /SQLSVCPASSWORD="<St
rongPassword>" /SQLSYSADMINACCO
UNTS="<DomainName\UserName>" /
AGTSVCACCOUNT="NT AUTHORITY\
Network Service" /TCPENABLED=1 /
IACCEPTSQLSERVERLICENSETERMS
```

Note that the parameters can change depending on the features you want to install. You can refer to http://msdn.microsoft.com/en-us/library/hh231669.aspx for more information.

Enable SQL Server remote access – Run SQLCMD from the Server Core command prompt. Then run the following commands in the SQLCMD window.

```
EXEC sys.sp_configure N'remote access', N'1'
GO
RECONFIGURE WITH OVERRIDE
GO
```

Auditing, User Defined Server Roles and Contained Databases

SQL Server 2012 has a number of other security related enhancements that serve to make it the most secure version of SQL Server to date. One of the most important changes to security was the ability to perform server auditing in all of the different SQL Server 2012 editions. Database level auditing is still limited to Enterprise, Developer, and Evaluation editions. In previous editions of SQL Server auditing was limited to just the Enterprise edition and higher. With SQL Server 2012 SQL Server Audit is now also more resistant to failures writing to the audit log target. In the event of a write failure it can now reconnect to the audit log. In addition, you can now cap the number of audit files without rolling over the log files. There is also support for support the monitoring of contained database users.

Another important security related enhancement in SQL Server 2012 is support for user defined server roles. In previous version of SQL Server the server roles were fixed. The new SQL Server 2012 allows you much more flexibility in the way you manage SQL Server by allowing you to create your own custom server roles. To work with SQL Server 2012's user-defined server roles you can use the new CREATE SERVER ROLE, ALTER SERVER ROLE, and DROP SERVER ROLE T-SQL statements. You can also add and remove members from all server roles using ALTER SERVER ROLE with its WITH ADD MEMBER parameter.

The new Contained Database feature makes it much easier to deploy databases and to move them between different SQL Server instances. With Contained Databases the logins are moved from the SQL Server instance level to the database level. In addition the metadata that describes the database is moved out of the Master database and into the user database.

To use the new SQL Server 2012 Contained Databases first you need to configure the SQL Server instance to support Contained Databases using the sp_configure stored procedure as you can see below.

```
EXEC sp_configure 'contained database authentication', 1;
```

Next, after the database has been created
ALTER DATABASE [MyContainedDB] SET CONTAINMENT = PARTIAL;

CREATE USER [MyUser] WITH PASSWORD = 'MyPwd';

Using Citrix NetScaler DataStream with SQL Server

Citrix's NetScaler is an Application Delivery Controller (ADC) appliance that optimizes how network clients connect to infrastructure servers like SQL Server. NetScaler is deployed as a proxy between network clients and your application servers and it acts to ensure application availability, accelerate application performance as well as reducing bandwidth requirements and securing the remote connections.

Citrix's NetScaler is deployed as either a physical appliance or as a virtual appliance. The physical appliances are the NetScaler MPX and NetScaler SDX platforms while the virtual appliance is the NetScaler VPX platform. NetScaler MPX is a high performance hardware appliance that can provide up to 50 Gbps of network throughput. The MPX platform is designed to support enterprise and cloud datacenters. The NetScaler SDX platform uses virtualization to support up to 40 independent NetScaler instances on the same appliance. The NetScaler SDX platform is expressly designed for multi-tenant support and each instance of NetScaler can support up 18 Gbps of bandwidth. The NetScaler VPX platform is a software-based virtual appliance. It supports Windows Server 2008 R2 with Hyper-V, Citrix XenServer 5.6 or later, and VMware ESX/ESXi 3.5 or later. VPX is designed for medium and smaller businesses that require flexibility and the ability to have NetScaler instances which can be dispersed for small to medium workloads, without the need for new hardware. NetScaler VPX enables organizations to rapidly deploy new NetScaler ADC instances using their existing virtualization infrastructure. All of the NetScaler appliances provide exactly the same feature set and they are managed exactly the same. You can deploy NetScaler in a multi-tiered approach for both front end web/application services as well as DataBase Services using a combination of the Hardware and Software versions as needed. NetScaler 10 offers multiple mechanisms for scaling in any environment (scale-up, scale-in, scale-out). Other options for deployment include pay-grow, SDX and Clustering.

NetScaler was initially designed to optimize web server traffic but it has evolved to support other types of network traffic

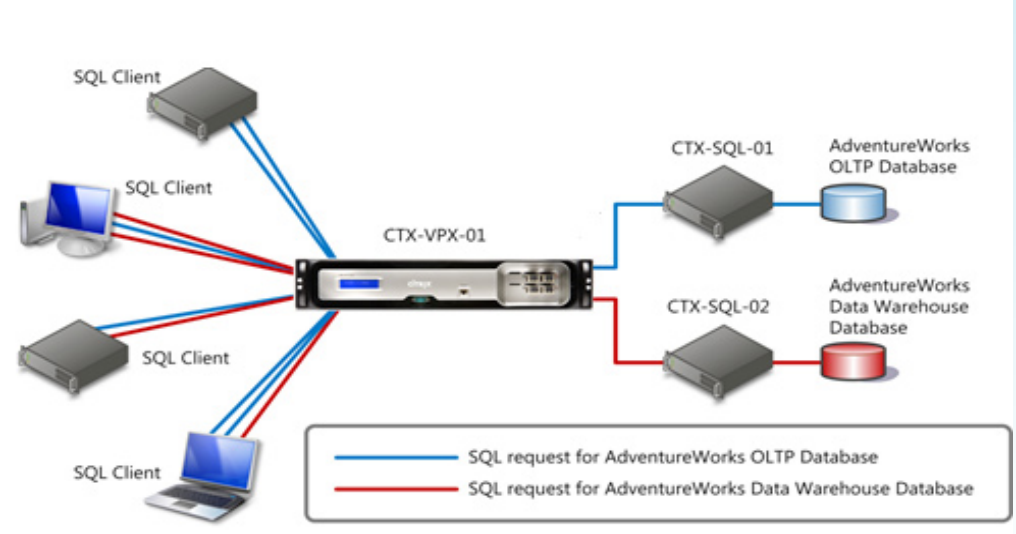

Figure 6 – An overview of a single instance NetScaler deployment

as well including SQL Server. In fact the NetScaler DataStream technology provides native on-the-wire support for SQL Server's TDS protocol. The DataStream technology in Citrix NetScaler provides traffic management, automated failover, and health monitoring for the data tier in the same way that Citrix NetScaler provides these capabilities for the web and application tier.NetScaler DataStream technology allows for real-time query level visibility and analytics along with the other benefits mentioned in this guide.

NetScaler Architecture

NetScaler's native support for the TDS protocol enables it to increase scalability, availability, security, performance and manageability of the data tier. NetScaler acts as a SQL Server proxy for the supported versions of SQL Server. Instead of connecting directly to the backend SQL Server instance network clients connect to the NetScaler. NetScaler in turns connects to SQL Server and it manages the network connections to the SQL Server system. NetScaler will support all versions of Microsoft SQL from 2005 through 2012. You can see an overview of how NetScaler's DataStream technology can be deployed with SQL Server in Figure 6.

In Figure 6 you can see that all of the different network SQL Server clients are actually connecting to the NetScaler appliance.

Clients make connections through the application server to a Virtual IP address provided by NetScaler. NetScaler's native TDS support allows the network clients to think that they are connecting to a SQL Server instance. NetScaler is managing the multiple incoming connections and intelligently routing the traffic and pooling the connections as needed. NetScaler's high performance network support ensures that the incoming client connections continue to have optimal network response time while combining connections on the backend optimizes SQL Server utilization and reduces network bandwidth requirements.

DataStream Features

NetScaler's multi-tier architecture provides a number of important advantages for database connections.

- More efficient networking and improved scalability with connection multiplexing – NetScaler's connection multiplexing allows you to support a larger number of simultaneous SQL Server connections with improved performance. Connection multiplexing enables numerous short-lived SQL client requests to be sent to the database server over a few long-lived SQL server connections. This can reduce the number of SQL Server connections by a ratio of 50:1 or more depending on the application. Offloading TCP connections to NetScaler enables more efficient utilization of server processor and memory. It also improves the performance of client connections because it eliminates the need to open each connection. Connection multiplexing can also be very important for scalability. SQL Server supports a maximum of 32,000 simultaneous connections per server which may present a potential bottleneck in various use cases and installations. NetScaler's connection multiplexing drastically reduces the number of SQL Server connections that are required effectively enabling SQL Server to support a much greater number of connections than it could natively.
- Improved performance with load balancing and caching – NetScaler is able to perform SQL Server load balancing where application performance can be improved by distributing client requests between multiple SQL Server instances. NetScaler provides scale-up using TCP offloading and connection pooling. Scale Out is achieved thru the ability to understand the TDS protocol and switch on anything within the SQL query. NetScaler's native support for TDS allows it to perform connection routing based on the context of the T-SQL statements that are

being sent across the network. NetScaler can interpret SQL statements, such as SELECT, INSERT, or UPDATE and direct the requests to different SQL Server instances based on the type of request. Beyond just T-SQL statements, NetScaler can actually switch on anything within the datastream including the database name and even table name/value pairs. For example, you can use NetScaler to send read operations to one server while sending write operations to another server or NetScaler can leverage your AlwaysOn Availability Group replicas by simultaneously directing read operations to multiple SQL Server instances. NetScaler's datastream context switching also enables it to support database partitioning (also known as sharding) where the administrator can place the logic to route queries to specific partitions from NetScaler rather than having to add that logic at the application tier. NetScaler also provides in-memory database caching. This caching is controlled by granular policies. For instance, you can direct NetScaler to cache specified read-only queries from a given database for a certain period of time or until a data update operation is sent to the cached database which would cause the cache to be invalidated.

- Increased security using granular user access policies – NetScaler's support for TDS also enables it to parse T-SQL commands and taking action based on the user and the transaction. NetScaler supports the Appflow standard (www.appflow.org) that enables it to gather statistics down to the query level and either consume them using its Action Analytics reporting features or feed those statistics to another monitoring tool such as Splunk or Solarwinds. These reports are actionable and they allow you to add specific policies in place to restrict or allow queries based on your organization's requirements. Policies can be created around what incoming queries are allowed to do with respect to the database. NetScaler can also apply user access policies to each database user. For instance, you could create a NetScaler policy that prohibited end users from issuing DDL statement or other configuration command that they would normally not be expect to perform. NetScaler also stores a consolidated log of all SQL transactions and user access.
- Improved availability with transparent client rerouting -- NetScaler is closely integrated with SQL Server and it provides the ability to monitor the health

of SQL Server instances based on a number of factors, such as replication backlog or response times, or the result of a custom query. Based on the policies in place NetScaler is able to off-load traffic to another server or if a database server failure occurs NetScaler can automatically and transparently reroute client connections to another server with no interruption of client connectivity. NetScaler also fully supports SQL Server 2012's new AlwaysOn Failover Clustering and Availability Groups.

Additional Resources

You can find more information about Citrix NetScaler at:
www.citrix.com/English/ps2/products/product.asp?contentID=21679

You can find more information about Citrix NetScaler DataStream at:
www.citrix.com/English/ps2/products/subfeature.asp?contentID=2309521

You can also download a free evaluation version of NetScaler VPX at:
www.citrix.com/English/ps2/products/feature.asp?contentID=2300361&ntref=bottom_promo_button

To learn more about Citrix NetScaler DataStream and Microsoft SQL Server and the Citrix and Microsoft relationship, please visit
www.microsoftandcitrix.com

ABOUT THE AUTHOR

Michael Otey, technical director for Windows IT Pro and SQL Server Pro, is president of TECA, a software-development and consulting company in Portland, Oregon, and coauthor of Microsoft SQL Server 2008 High Availability with Clustering & Database Mirroring (Osborne/McGraw-Hill). Michael has covered the topic of virtualization extensively for Windows IT Pro, having written several features articles showing how to take advantage of virtualization in the enterprise as well as reviewing all of the major virtualization products.