Search Topology Operations in SharePoint Server 2010

Search Topology Operations in SharePoint Server 2010

Firaz Samet, Brion Stone, Nathan Treloar
Microsoft Corporation

April 2010

Applies to: Microsoft SharePoint Server 2010

Summary: This white paper describes how to use the Windows PowerShell™ command-line interface to add components to scale out the search topology in Microsoft® SharePoint® Server 2010. It also describes how to perform a full topology import and export.

Contents

Introduction

The enterprise search system in SharePoint Server 2010 includes a number of important architectural improvements for more efficient scaling across volumes of content and queries. The SharePoint Server 2010 Enterprise Search Evaluation Guide (http://go.microsoft.com/fwlink/?LinkId=191058&clcid=0x409) gives an overview of the new search architecture in SharePoint Server 2010. Additionally, see SharePoint Server 2010

performance and capacity test results and recommendations (http://go.microsoft.com/fwlink/?LinkId=191156&clcid=0x409) to learn how to use available hardware to meet the organization's performance and reliability goals. In this white paper, we'll look at how to perform topology operations by using Windows PowerShell, and by using the corresponding user interface method when available. Later on, we will describe the more holistic approach of full topology import and export.

Permissions required for topology operations

Before you perform any of the operations in this white paper, be sure that the user account that you use to perform them is a member of the SharePoint_Shell_Access role on the configuration database and a member of the Administrators and WSS_ADMIN_WPG local groups on the computer where SharePoint Server 2010 is installed.

Common Windows PowerShell code

Retrieving the search service application object

The search service application object will be used throughout this article; it is at the root of any search administrative operation. To retrieve the list of all search service applications, use `Get-SPEnterpriseSearchServiceApplication` with no parameters. At the Windows PowerShell command prompt, type the following to retrieve the search service application object:

Copy Code

```
$searchApp = Get-SPEnterpriseSearchServiceApplication <Search Application Name>
$searchApp
```

```
Name                  : SearchAppTest
Id                    : 9c3574fa-2856-4f7e-a8d5-a41c30a83e26
ServiceName           : SearchQueryAndSiteSettingsService
QueryTopologies       : {b4f103d2-ddfb-4e0a-ba47-5cd1a8b61f37}
PropertyStores        : {SearchAppTest_PropertyStore}
CrawlTopologies       : {8e79ea17-b70c-46b8-924d-e11e091c50bf}
CrawlStores           : {SearchAppTest_CrawlStore}
SearchAdminDatabase   : SearchAdminDatabase Name=SearchAppTest
Status                : Online
SearchApplicationType : Regular
DefaultSearchProvider : SharepointSearch
Properties            : {Microsoft.Office.Server.Utilities.SPPartitionOptions}
```

Checking the search service instance

Before adding search components, make sure the server you are going to use is ready to host search components. To retrieve a list of all the servers in the farm and the status of their search service instances, use `Get-SPEnterpriseSearchServiceInstance` with no parameters. To check the server, type the following

at the Windows PowerShell command prompt and verify that the status of the search service instance is online:

Copy Code

```
$searchInstance = Get-SPEnterpriseSearchServiceInstance <SearchServerName>
$searchInstance
```

```
TypeName         : SharePoint Server Search
Description      : Index content and serve search queries
Id               : c348a930-cb90-4335-8ac1-b563d5efb9f8
Server           : SPServer Name=<SearchServerName>
Service          : SearchService Name=OSearch14
Role             : None
QueryComponents  :
CrawlComponents  :
AdminComponents  :
Status           : Online
```

If you want to bring a search service instance online, type the following at the Windows PowerShell command prompt:

Copy Code

```
Start-SPEnterpriseSearchServiceInstance <SearchServerName>
```

```
TypeName         : SharePoint Server Search
Description      : Index content and serve search queries
Id               : c348a930-cb90-4335-8ac1-b563d5efb9f8
Server           : SPServer Name=<SearchServerName>
Service          : SearchService Name=OSearch14
Role             : None
QueryComponents  :
CrawlComponents  :
AdminComponents  :
Status           : Provisioning
```

The ‘Status’ of the search service instance will be set to ‘Provisioning’ for a few moments, then it will be set to ‘Online’. Use `Get-SPEnterpriseSearchServiceInstance` to check its status and make sure it is online before using it to host search components.

Query scale-out

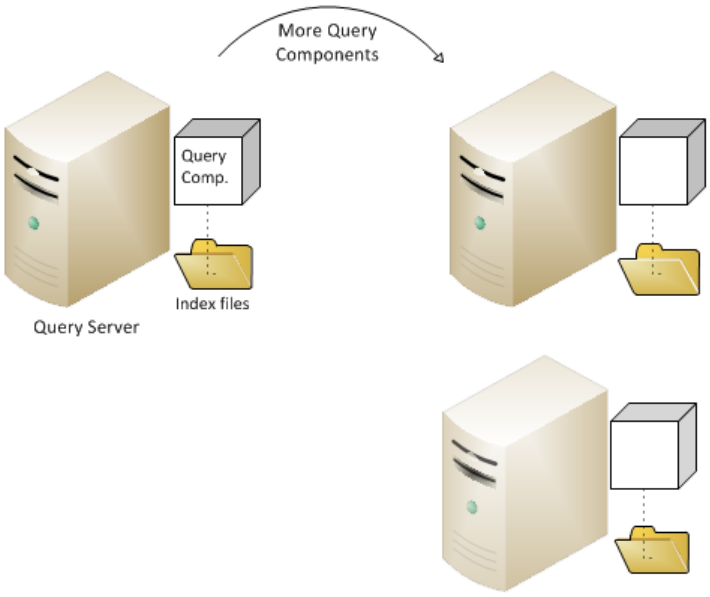
This section will cover only search in SharePoint Server 2010; query scale-out in FAST™ Search Server 2010 for SharePoint has a different approach and will not be covered in this article.

Query topologies

Query topology represents the query subsystem's layout across the servers in the farm, and how they are configured to work together. To change the current query topology, create a new query topology object, make the planned changes (we will describe the options shortly), and then activate the new query topology.

Add query components

Adding query components is typically done either to increase query throughput or to add failover components to support redundancy. To add more query components while keeping the existing ones and their index partitions, clone the active query partition, add the new query component to the desired index partitions, and then activate the cloned query topology. For information about how to do this through the user interface, see Add or remove a query component (http://technet.microsoft.com/en-us/library/ee805953(office.14).aspx).



In this example, we will add a new query component, located on a new server, to the existing topology; both query components will be using the same index partition.

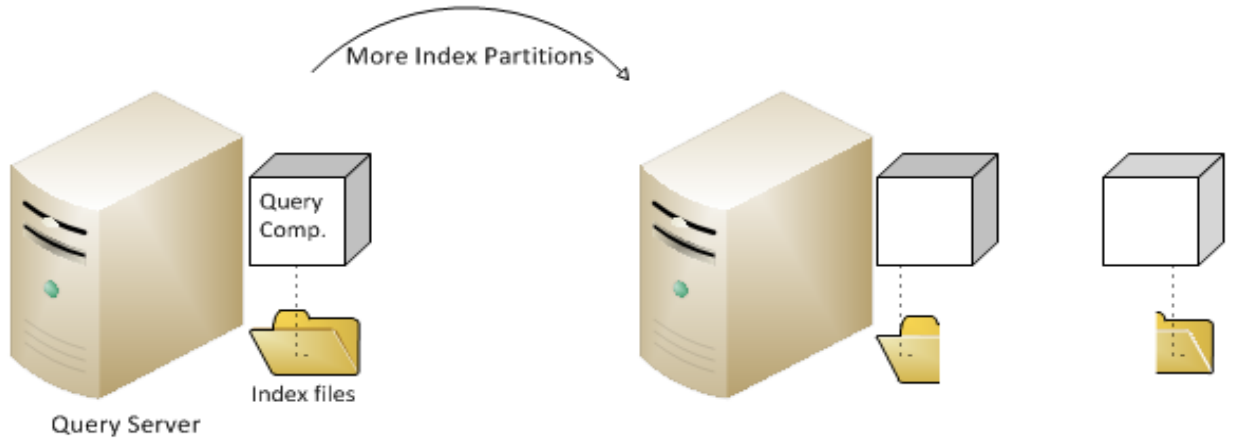
Copy Code

```
# Retrieve the active topology
$InitialQueryTopology = $searchApp | Get-SPEnterpriseSearchQueryTopology -Active
$InitialQueryTopology

# Clone the topology
```

```
$QueryTopology = $searchApp | New-SPEnterpriseSearchQueryTopology -Clone -QueryTopology
$InitialQueryTopology


# Use the following code to add a new query component on a specific partition
# Repeat the following three lines for every new query component
$searchInstance = Get-SPEnterpriseSearchServiceInstance <SearchServerName>
$IndexPartition= ([array](Get-SPEnterpriseSearchIndexPartition -QueryTopology
$QueryTopology))[0]
$QueryComponent0 = New-SPEnterpriseSearchQuerycomponent -QueryTopology $QueryTopology -
IndexPartition $IndexPartition -SearchServiceInstance $searchInstance -IndexLocation
$searchInstance.DefaultIndexLocation

# Activate the new query topology
$QueryTopology | Set-SPEnterpriseSearchQueryTopology -Active
Write-Host -ForegroundColor white Waiting for the old query topology to become inactive
do {write-host -NoNewline .;Start-Sleep 10;} while ($InitialQueryTopology.State -ne
"Inactive")

# Delete the old query topology
$InitialQueryTopology | Remove-SPEnterpriseSearchQueryTopology
```

Index partitions

Creating new index partitions (and repartitioning) is usually done to scale-out the farm in order to support a larger number of items in the index. To add a new index partition, create a new query topology, and then add the desired index partitions to it. For information about how to do this through the user interface, see Add or remove an index partition (http://technet.microsoft.com/en-us/library/ee805955(office.14).aspx).



In the following example, we will create a new query topology with two index partitions to replace the initial query topology that has one index partition.

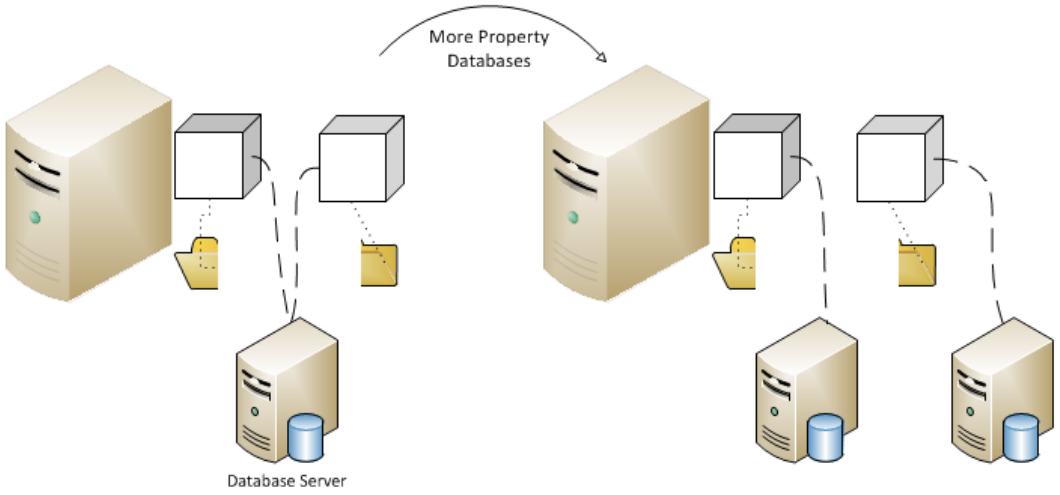
Copy Code

```
# Retrieve the active topology
$InitialQueryTopology = $searchApp | Get-SPEnterpriseSearchQueryTopology -Active
$InitialQueryTopology
```

```
# Create a new query topology
$QueryTopology = $searchApp | New-SPEnterpriseSearchQueryTopology -Partitions 2

# Retrieve the index partition objects
$IndexPartition0= (Get-SPEnterpriseSearchIndexPartition -QueryTopology $QueryTopology)[0]
$IndexPartition1= (Get-SPEnterpriseSearchIndexPartition -QueryTopology $QueryTopology)[1]

# Add at least a query component to each index partition

# Add a query component to the first index partition, located on the first search server
$searchInstance0 = Get-SPEnterpriseSearchServiceInstance <searchServerName>
$QueryComponent0 = New-SPEnterpriseSearchQuerycomponent -QueryTopology $QueryTopology -
IndexPartition $IndexPartition0 -SearchServiceInstance $searchInstance0 -IndexLocation
$searchInstance0.DefaultIndexLocation

# Add a query component to the second index partition, located on the same search server

$QueryComponent1 = New-SPEnterpriseSearchQuerycomponent -QueryTopology $QueryTopology -
IndexPartition $IndexPartition1 -SearchServiceInstance $searchInstance0 -IndexLocation
$searchInstance0.DefaultIndexLocation

# Reuse the existing property database
$PropertyDatabase0 = ([array]($searchApp | Get-SPEnterpriseSearchPropertyDatabase))[0]

# Assign both index partitions to the property database
$IndexPartition0 | Set-SPEnterpriseSearchIndexPartition -PropertyDatabase
$PropertyDatabase0
$IndexPartition1 | Set-SPEnterpriseSearchIndexPartition -PropertyDatabase
$PropertyDatabase0

# Activate the new query topology
$QueryTopology | Set-SPEnterpriseSearchQueryTopology -Active
Write-Host -ForegroundColor white Waiting for the old query topology to become inactive
do {write-host -NoNewline .;Start-Sleep 10;} while ($InitialQueryTopology.State -ne
"Inactive")

# Delete the old query topology
$InitialQueryTopology | Remove-SPEnterpriseSearchQueryTopology
```

Property database scale-out

Creating new property databases (and refactoring) is usually done to scale out the farm in order to eliminate bottlenecks or support more metadata for the items in the index. To use more property databases, create the desired property databases, create or clone a query topology, create and add query components if necessary, and assign each partition to the desired property database. For information about how to do this through the user interface, see Add or remove a property database (http://technet.microsoft.com/en-us/library/ee805954(office.14).aspx).



In the following example, we will clone the existing query topology containing two index partitions, create two new property databases, and assign each partition to a property database.

Copy Code

```
# Retrieve the active topology
$InitialQueryTopology = $searchApp | Get-SPEnterpriseSearchQueryTopology -Active
$InitialQueryTopology

# Clone the topology
$QueryTopology = $searchApp | New-SPEnterpriseSearchQueryTopology -Clone -QueryTopology
$InitialQueryTopology

# Retrieve the index partition objects
$IndexPartition0= (Get-SPEnterpriseSearchIndexPartition -QueryTopology $QueryTopology)[0]
$IndexPartition1= (Get-SPEnterpriseSearchIndexPartition -QueryTopology $QueryTopology)[1]

# The cloned query topology already has query components.

# Create 2 new property databases
$PropertyDatabase0 = $searchApp | New-SPEnterpriseSearchPropertyDatabase -DatabaseName
<PropertyDbName1>
$PropertyDatabase1 = $searchApp | New-SPEnterpriseSearchPropertyDatabase -DatabaseName
<PropertyDbName2>

# Assign each index partition to a new property database
$IndexPartition0 | Set-SPEnterpriseSearchIndexPartition -PropertyDatabase
$PropertyDatabase0
$IndexPartition1 | Set-SPEnterpriseSearchIndexPartition -PropertyDatabase
$PropertyDatabase1

# Activate the new query topology
$QueryTopology | Set-SPEnterpriseSearchQueryTopology -Active
Write-Host -ForegroundColor white Waiting for the old query topology to become inactive
do {write-host -NoNewline .;Start-Sleep 10;} while ($InitialQueryTopology.State -ne
"Inactive")

# Delete the old query topology
$InitialQueryTopology | Remove-SPEnterpriseSearchQueryTopology
```

```
# Delete unused property databases
$searchApp | Get-SPEnterpriseSearchPropertyDatabase | Where-Object { !($_.IndexPartitions)
} | remove-spenterprisesearchpropertyDatabase
```

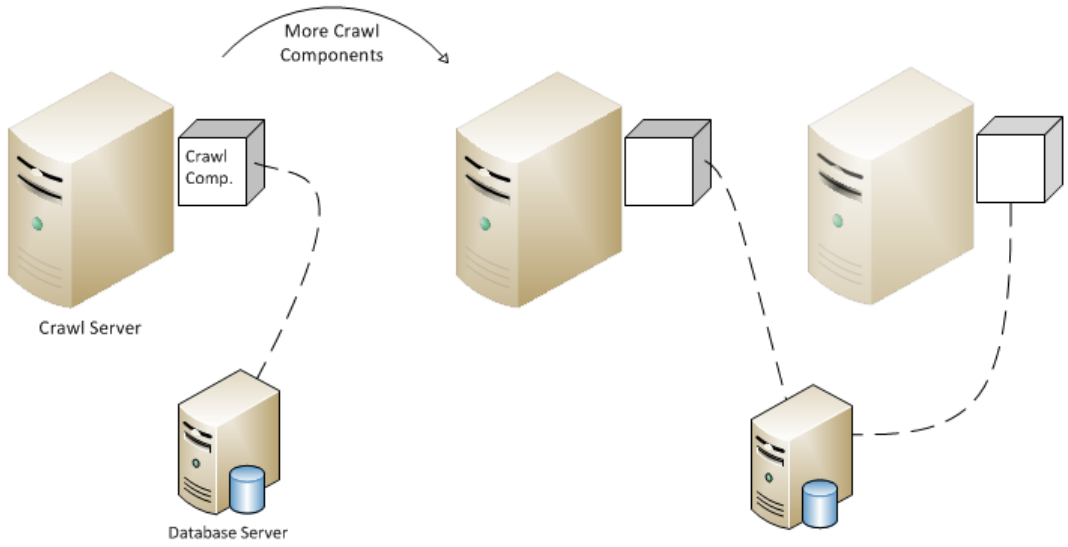
Crawl scale-out

Crawl topologies

Crawl topology represents the crawl subsystem's layout across the servers in the farm and how they are configured to work together. To change to the current crawl topology, create a new crawl topology object, make the planned changes (we will describe each of them shortly), and then activate the new crawl topology.

Crawl component

Adding crawl components is usually done to either increase crawl throughput or to add components on different servers for redundancy. To add more crawl components while keeping the existing ones and their crawl databases, clone the active crawl topology, add the new crawl component to the desired crawl database, and then activate the cloned crawl topology. For information about how to do this through the user interface, see Add or remove a crawl component (http://technet.microsoft.com/en-us/library/ee805950(office.14).aspx).



In this example, we will add a new crawl component, located on a new server, to the existing topology; both crawl components will be using the same crawl database.

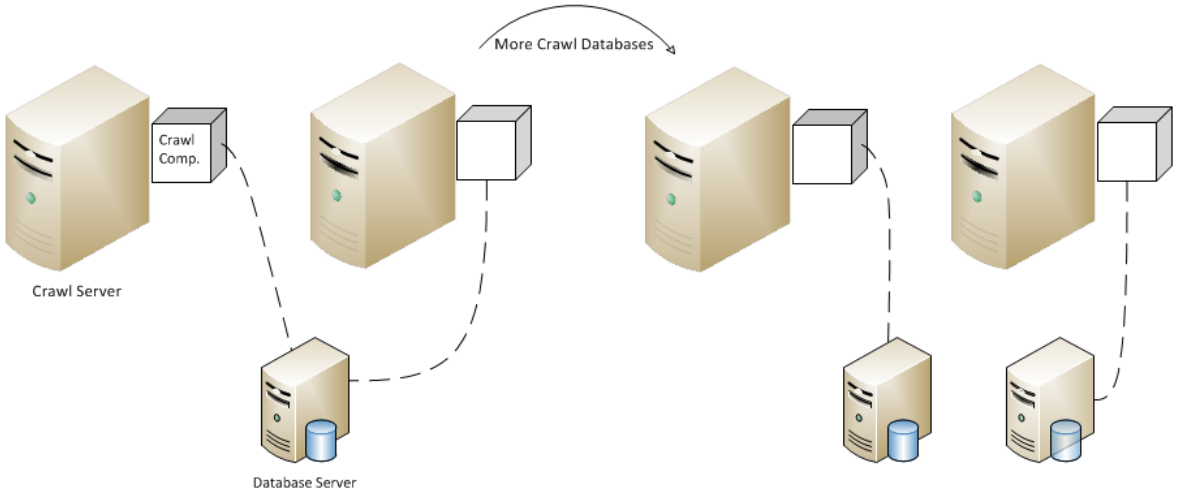
Copy Code

```
# Retrieve the active topology
$InitialCrawlTopology = $searchApp | Get-SPEnterpriseSearchCrawlTopology -Active
$InitialCrawlTopology

# Clone the topology
$CrawlTopology = $searchApp | New-SPEnterpriseSearchCrawlTopology -Clone -CrawlTopology
```

```
$InitialCrawlTopology

# Create the new crawl component
$CrawlDatabase0 = ([array]($searchApp | Get-SPEnterpriseSearchCrawlDatabase))[0]
$searchInstance0 = Get-SPEnterpriseSearchServiceInstance <searchServerName>
$CrawlComponent0 = New-SPEnterpriseSearchCrawlComponent -CrawlTopology $CrawlTopology -
CrawlDatabase $CrawlDatabase0 -SearchServiceInstance $searchInstance0 -IndexLocation
$searchInstance0.DefaultIndexLocation

# Activate the new crawl topology
$CrawlTopology | Set-SPEnterpriseSearchCrawlTopology -Active
Write-Host -ForegroundColor white Waiting for the old crawl topology to become inactive
do {write-host -NoNewline .;Start-Sleep 10;} while ($InitialCrawlTopology.State -ne
"Inactive")

# Delete the old crawl topology
$InitialCrawlTopology | Remove-SPEnterpriseSearchCrawlTopology
```

Crawl database scale-out

Creating new crawl databases (and doing host redistribution) is usually done to scale out the farm in order to eliminate bottlenecks or support crawling more items. To use more crawl databases, create the desired crawl databases, create or clone a crawl topology, create and add crawl components if necessary, and assign each component to the desired crawl database. For information about how to do this through the user interface, see Add or remove a crawl database (http://technet.microsoft.com/en-us/library/ee805952(office.14).aspx).



In the following example, we will create a new crawl topology, create two new crawl databases, and create a crawl component for each crawl database.

Copy Code

```
# Retrieve the active topology
$InitialCrawlTopology = $searchApp | Get-SPEnterpriseSearchCrawlTopology -Active
```

```
$InitialCrawlTopology

# Clone the topology
$CrawlTopology = $searchApp | New-SPEnterpriseSearchCrawlTopology

# Create a new crawl component on the first crawl database
$searchInstance0 = Get-SPEnterpriseSearchServiceInstance <searchServerName0>
$CrawlDatabase0 = $searchApp | New-SPEnterpriseSearchCrawlDatabase -DatabaseName
<CrawlDbName0>
$CrawlComponent0 = New-SPEnterpriseSearchCrawlComponent -CrawlTopology $CrawlTopology -
CrawlDatabase $CrawlDatabase0 -SearchServiceInstance $searchInstance0 -IndexLocation
$searchInstance0.DefaultIndexLocation

# Create a new crawl component on the second crawl database
$searchInstance1 = Get-SPEnterpriseSearchServiceInstance <searchServerName1>
$CrawlDatabase1 = $searchApp | New-SPEnterpriseSearchCrawlDatabase -DatabaseName
<CrawlDbName1>
$CrawlComponent1 = New-SPEnterpriseSearchCrawlComponent -CrawlTopology $CrawlTopology -
CrawlDatabase $CrawlDatabase1 -SearchServiceInstance $searchInstance1 -IndexLocation
$searchInstance1.DefaultIndexLocation

# Activate the new crawl topology
$CrawlTopology | Set-SPEnterpriseSearchCrawlTopology -Active
Write-Host -ForegroundColor white Waiting for the old crawl topology to become inactive
do {write-host -NoNewline .;Start-Sleep 10;} while ($InitialCrawlTopology.State -ne
"Inactive")

# Delete the old crawl topology
$InitialCrawlTopology | Remove-SPEnterpriseSearchCrawlTopology

# Delete unused crawl databases
$searchApp | Get-SPEnterpriseSearchCrawlDatabase | Where-Object { !($_.CrawlComponents) }
| remove-spenterprisesearchcrawlDatabase
```

Administration topology change

The administration component can be moved from one server to another to support scale-out, or a 'failed server' scenario. In this example, we will move it to a new server.


```
# Retrieve the search server that will host the administration component
$searchInstance = Get-SPEnterpriseSearchServiceInstance <searchServerName>

# Assign the administration component to the new server
$searchApp | get-SPEnterpriseSearchAdministrationComponent | set-
SPEnterpriseSearchAdministrationComponent -SearchServiceInstance $searchInstance
$admin = ($searchApp | get-SPEnterpriseSearchAdministrationComponent)

# Wait for the administration component to be initialized
do {write-host -NoNewline .;Start-Sleep 10;} while (-not $admin.Initialized)
```

In this next example, the server hosting the administration component failed and could no longer be contacted; we will force the move of the administration component to a new server without waiting for the old server to respond.

Copy Code

```
# Retrieve the search server that will host the administration component
$searchInstance = Get-SPEnterpriseSearchServiceInstance <searchServerName>

# Assign the administration component to the new server
$searchApp | get-SPEnterpriseSearchAdministrationComponent | set-
SPEnterpriseSearchAdministrationComponent -SearchServiceInstance $searchInstance -Force
$admin = ($searchApp | get-SPEnterpriseSearchAdministrationComponent)

# Wait for the administration component to be initialized
do {write-host -NoNewline .;Start-Sleep 10;} while (-not $admin.Initialized)
```

Topology import and export

Cmdlets are available to export and import topologies to support offline search application creation (via XML), or export-import scenarios. This feature is also useful when you need to perform many topology manipulations at the same time. The following example shows how to export and import topology XML.

Copy Code

```
# Export the current search topology to an XML file.
$searchapp | Export-SPEnterpriseSearchTopology -Filename c:\Topology.xml

# Edit the topology settings XML and save it to c:\EditTopology.xml

# Import the updated topology settings XML file.
$searchapp | Import-SPEnterpriseSearchTopology -Filename c:\EditedTopology.xml
```

The following is a sample content of a topology settings XML file:

```
<?xml version="1.0" encoding="utf-8"?>
<TopologySettings xmlns="http://schemas.microsoft.com/office/2009/SearchTopology">
  <ApplicationId>dce97454-ae28-4410-ac28-b69b5d69d46d</ApplicationId>
  <ObjectId>9c3574fa-2856-4f7e-a8d5-a41c30a83e26</ObjectId>
  <AdminTopology>
    <AdminComponentServer>SearchServerName</AdminComponentServer>
    <AdminDatabase>
      <Server>DatabaseServerName</Server>
      <Name>SearchApplicationName</Name>
      <DatabaseId>f7f9ac79-532d-4be8-bf4e-9f0c34f62cf3</DatabaseId>
    </AdminDatabase>
  </AdminTopology>
  <QueryTopology>
    <PropertyStores>
      <PropertyStore>
        <Id>DatabaseServerName-PropertyDatabaseName</Id>
        <Database>
          <Server>DatabaseServerName</Server>
          <Name>PropertyDatabaseName</Name>
          <DatabaseId>9caa108a-9fce-46be-bed4-65a6f1678e5b</DatabaseId>
        </Database>
      </PropertyStore>
    </PropertyStores>
    <QueryPartitions>
      <QueryPartition>
        <QueryComponents>
```

```
          <QueryComponent>
            <ServerName>SearchServerName</ServerName>
            <IndexLocation>E:\Program Files\Microsoft Office Servers\14.0\Data\Office
Server\Applications</IndexLocation>
            <ComponentId>6d801313-54f4-466c-a0be-ee7daa8b1de4</ComponentId>
            <FailoverOnly>False</FailoverOnly>
          </QueryComponent>
        </QueryComponents>
        <PropertyStore>DatabaseServerName-PropertyDatabaseName</PropertyStore>
      </QueryPartition>
    </QueryPartitions>
  </QueryTopology>
  <CrawlTopology>
    <CrawlStores>
      <CrawlStore>
        <Id>DatabaseServerName-SearchApplicationName_CrawlStore</Id>
        <Database>
          <Server>DatabaseServerName</Server>
          <Name>SearchApplicationName_CrawlStore</Name>
          <DatabaseId>4117bf9e-9eac-4e2d-b4ad-4bce6b4b7dd8</DatabaseId>
        </Database>
        <IsDedicated>False</IsDedicated>
      </CrawlStore>
    </CrawlStores>
    <CrawlComponents>
      <CrawlComponent>
        <ServerName>SearchServerName</ServerName>
        <IndexLocation>F:\Program Files\Microsoft Office Servers\14.0\Data\Office
Server\Applications</IndexLocation>
        <ComponentId>f0c93f38-6799-4c47-9383-5c8ca0a84322</ComponentId>
        <CrawlStore>DatabaseServerName-SearchApplicationName_CrawlStore</CrawlStore>
      </CrawlComponent>
    </CrawlComponents>
  </CrawlTopology>
</TopologySettings>
```

Conclusion

Search in SharePoint Server 2010 offers major design improvements that allow very flexible system deployment. Many scale-out options give you more control to improve search performance. Whether the goal is to improve crawl performance, query performance, or both; there are plenty of options to reach that goal.

Additional resources

Manage search topology (http://technet.microsoft.com/en-us/library/ee805956(office.14).aspx)

Technical diagrams (SharePoint Server 2010) (http://technet.microsoft.com/en-us/library/cc263199(office.14).aspx)

Search cmdlets (SharePoint Server 2010) (http://technet.microsoft.com/en-us/library/ee906563(office.14).aspx)

About the authors

- Firaz Samet is a Software Development Engineer in Test and has been with Microsoft since 2006. He worked on search administration and topology in SharePoint Server 2010.

- Brion Stone is a Senior Program Manager in the Microsoft Enterprise Search Group and has been with Microsoft since 2000. His current focus is on SharePoint search topology, manageability, and performance, building on his past experience in Windows Live services and Internet client applications.
- Nathan Treloar is Principal Search Technology Evangelist in the Microsoft Enterprise Search Group where he has responsibility for the group's technology innovation and evangelism programs.