Is BranchCache right for remote, serverless software distribution?

1E Technical Whitepaper

Microsoft BranchCache and System Center Configuration Manager 2007

Abstract

BranchCache is a new feature available in Windows Server 2008 R2 and Windows 7 that reduces WAN bandwidth usage and improves application responsiveness when workstations in a remote location access content from the head office or data center by downloading and caching content on the local network as it is requested, making it immediately available to other clients that subsequently request the same content.

This paper examines the BrachCache functionality specifically in the context of software distribution using System Center Configuration Manager 2007 to determine if it is an optimal solution for the deployment of software, patches and operating systems to remote, serverless branches.

Key Findings

- BranchCache requires content to be hosted on Windows Server 2008 R2 and participating clients to have Windows 7 or Windows Vista with BITS 4.0 installed
- BranchCache does not require a local server when operated in Distributed Cache mode, which is ideal for environments of around 50 clients. Microsoft recommend the Hosted Cache mode (which requires a local Windows Server 2008 R2 server in the branch) for remote locations that include laptops
- BranchCache is a very efficient method for distributing 'ad-hoc' content, including software distributions and updates that are initiated by the user at an arbitrary time (i.e. non-mandatory Advertisements that do not have a scheduled install time).
- When the same content is requested simultaneously by several clients (as is the case with scheduled software and patch deployments), some of the content may be downloaded across the WAN by several clients. If the content is HTTP, BITS is used which may result in link saturation when multiple clients simultaneously download

Is BranchCache right for Software Distribution?

Copyright © 1E LTD 2010

Version 1.0 document revision 1

Author: Simon Burbidge & Dave Fuller

Contents

Introduction

BranchCache is a new feature of Microsoft Windows Server 2008 R2 and Windows 7 (also available on Windows Vista with BITS 4.0 installed). It provides a software-based WAN optimization solution for branch environments enabling remote content requested by clients in the branch to be cached locally, either on a dedicated server or among peer workstations, so that it is available locally to other clients in the branch that may require the same content later rather than having them download it again over the WAN.

Once set up, BranchCache is transparent to the application and is used to manage HTTP, HTTPS and SMB2 traffic, so any applications that use these protocols can benefit from BranchCache. Microsoft included BranchCache support in the SP2 release of System Center Configuration Manager (ConfigMgr) 2007, enabling its software distribution, software updates and image deployment features to take advantage of this feature in environments where Windows 2008 R2 and Windows 7 have been deployed and the BranchCache feature configured.

This paper takes a look at the BranchCache feature in detail, focusing on how the feature performs in remote software distribution, patch management and image deployment scenarios. Although BranchCache can be configured to use a local Windows Server 2008 R2 server to store the cache (explained in *BranchCache Modes*), this paper focuses on the server-less Distributed Mode, on the assumption that if a local server were available, it would make sense to host a ConfigMgr Secondary Site or Distribution Point on that server anyway.

How does BranchCache work?

BranchCache Modes

BranchCache can operate in one of two modes: Hosted Mode or Distributed Mode.

Hosted Mode requires a local Windows Server 2008 R2 server in the branch and clients to be configured with the FQDN of this server. When a client requests content from a remote server that is not already present on the local BranchCache server, the client downloads the content but then it is moved to the local server (the content is actually advertised to the server and the server downloads it from the client). The local cache is maintained on the server and subsequent requests for the same content from other clients in the branch can be served from this cache.

Distributed Mode does not require a local server. Instead, content is cached on the workstation clients and is shared with peer clients when they subsequently require the same content (remember only Windows 7 or Windows Vista with BITS 4.0 clients can participate). Distributed Mode is suitable for smaller environments (Microsoft recommends up to 50 clients). Clients participating in peer-to-peer content transfer must be in the same subnet.

Microsoft recommends using Hosted Mode for branches with more than 50 client and branches that have laptops that may be removed from the network.

Discovery and Download

BranchCache is transparent to the application. When the application running on a BranchCache enabled client requests content from a BranchCache enabled server over HTTP/HTTPS or SMB2, the BranchCache client will request content metadata from the remote server. Metadata is generated on the source server each time it is requested by the client and consists of block hashes, segment hashes and a segment key, a segment being a collection of blocks. Typically the hashing results in a compression ratio of 2000:1, meaning that the metadata is typically 1/2000 (or 0.05%) of the size of the source data. A BranchCache client will always request this metadata from the source server regardless of whether the data already exists locally to ensure that the source content has not changed.

Once the metadata has been downloaded, in Distributed Mode clients perform a local discovery (using the Branch Cache Discovery Protocol, which in turn uses the WS-Discovery protocol) to determine if any of the segments are available within the branch before resorting to downloading these from the remote server. HTTP/HTTPS content is downloaded using Background Intelligent Transfer Service (BITS).

Interestingly, when using SMB2, BranchCache can be configured to be invoked according to connection latency (a threshold set using NETSH) so allowing BranchCache to only be used when the connection to the server is performing poorly. In contrast, BranchCache using BITS does not possess this intelligence and is always invoked if enabled.

Note that BranchCache does not download files smaller than 64KB (the requesting application will receive these through normal SMB2 / HTTP transfer).

Caching

When BranchCache is implemented in Distributed Mode, each client is allocated a proportion of the total (not free) disk size. By default this is 5% but can be adjusted using NETSH. When the BranchCache service first starts, the cache is initialized with sparse files (0 data files with a .PDS extension). These will eventually host the downloaded package and hash data. During testing it was seen that when BranchCache is actually used for the first time, it appears to instantly reserve half its cache allocation with the 0 byte data .PDS file being expanded to 50% allocated cash size (regardless of the size of the actual package being downloaded).

It should be noted that the BranchCache is completely independent from the ConfigMgr client cache. The ConfigMgr client will populate its own cache from wherever BranchCache gets the content from. Using BranchCache with ConfigMgr therefore requires sufficient disk space on the client to host the content in both the BranchCache and the ConfigMgr cache.

Setup and configuration

As long as your source servers are running Windows Server 2008 R2 and your workstations are on Windows 7 (or Vista with BITS 4.0), setting BranchCache up is relatively straightforward. BranchCache is installed as a Feature in the Server Manager interface. If you want to support file transfer (SMB2) with BranchCache you need to add the *BranchCache for network files* role in the File Services server role (also managed through Server Manager).

On clients, BranchCache is disabled by default. Most organisations would use Group Policy to enable and configure BranchCache, typically defining the cache mode, size and location. BranchCache can also be enabled and managed using the NETSH command-line utility. The NETSH command-line tool can also be used to manage the local cache on a client (such as flushing it) and can be used to display configuration settings and cache content.

The Remote Branch Software Distribution Challenge

Remote branch offices will typically range from a handful of clients to perhaps a couple of hundred. Often these will be a healthy mix of desktops and laptops, with some laptops being off the network for periods of the time. In many cases there is no facility (or desire) to host servers in these locations, and often network bandwidth to the head offices and datacenters is highly contested.

The distribution of software (applications, patches and OS images) in these locations is somewhat different to the user-initiated applications (e.g. accessing documents from a SharePoint server) commonly used to demonstrate BranchCache functionality.

Many software and patch deployments require content to be distributed to all machines in the branch at the same time (typically the case with Software Updates, but also with the planned roll-out of a new application). Software applications and updates may be distributed in the form of a single 'self-extracting' file (e.g. in MSI or EXE format), or may involve the distribution of several files. Distribution of OS images requires the transfer of very large WIM files and the process of deploying the image to a workstation will typically require additional package content (applications) not included in the image.

These deployments have to contend with other applications using the WAN and must avoid adversely impacting the normal flow of business relying upon it. In order to achieve deployment within an acceptable time period, WAN traffic should be minimised and bandwidth used efficiently. This involves ensuring the content only traverses the WAN once and is shared out within the branch (rather than each client obtaining it from the source server), and ensuring the download process is aware of existing traffic levels and is able to throttle demand on the link accordingly, taking advantage of 'quiet' periods but backing off when other traffic requires the bandwidth.

In a serverless branch, the dependency is on the client workstations to provide the local cache. It is therefore important that the workstation that hosts the cache is available at all times, or that the cache is replicated on several machines to provide resilience.

The ideal solution for software distribution to remote branches would have the following attributes

- No requirement for a local server

- Efficient use of available bandwidth, making use of bandwidth as it becomes available but ensuring it is never saturated and ensuring content is only transferred to the branch once
- No dependency for any one client to be available at all times
- Efficient use of available disk capacity on local workstations

How does BranchCache measure up?

1E conducted research into the use of BranchCache in the software distribution scenario, specifically with the use of System Center Configuration Manager (ConfigMgr) 2007 Service Pack 2, which introduced specific integration with BranchCache. This section summarises the results of our testing in relation to the ideal attributes listed above.

No requirement for a local server

Implementing BranchCache in Distributed mode supports this requirement. Windows 7 and Windows Vista clients with BITS v4.0 can participate in the peer-caching. Microsoft recommends that Distributed mode be used for branches up to 50 clients. If clients span multiple subnets then Distributed mode results in one download per subnet.

In the *Key BranchCache Design Points* section of the *Microsoft Windows Server 20087 R2 BranchCache Design Guide*[1] it states that the BranchCache data can only be decrypted by authenticated clients that are members of the same domain as the content server, although we have not verified this requirement.

Efficient use of available bandwidth

When multiple clients request the same content simultaneously, as is the case when a deployment is scheduled at a specific time, we observed that all clients begin to download data from the source server simultaneously. (The BranchCache performance counters were used to measure the volume of data downloaded from the WAN, downloaded from a peer cache and served to others on the network).

This is expected as every client must obtain the metadata from the source to ensure data integrity, and as previously noted this should amount to 0.05% of the content size if 2000:1 compression is achieved. However, we observed that overall, the amount of data downloaded across the WAN to the branch was between 2 and 11% greater than the content size (i.e. the overhead was between 2 and 11%), suggesting that clients were simultaneously downloading content as well as metadata. The tables below show some examples of our test results.

- **Bytes from server** is the amount of data transferred across the WAN from the remote server
- **Bytes from Cache** is the amount of data that the client obtained from the cache on another local workstation
- **Bytes Served** is the amount of data that each client provided to other clients from its own cache

5MB Single file package deployment

Client	Bytes from Server	Bytes from Cache	Bytes Served
A	204KB	4.8MB	1.6MB
B	347KB	4.6MB	196KB
C	5MB	0	7.6MB

In this test, all clients began downloading from the server simultaneously. After about 20 seconds, A and B switch to peer caching. The bytes served figures indicates that A and B cached from C as well as from each other. As noted earlier, the file metadata (data that must be first downloaded from the server) should be using a compression ratio of 2000:1 (0.05% of file size) or 2.5KB for the test example above. The results show that clients downloaded substantially more than this amount from the server, suggesting that either the desired compression ratio

[1] http://www.microsoft.com/downloads/details.aspx?FamilyID=CBF75A21-BC09-4824-B128-1A24EE71A9AA&displayLang=en

was not achieved, or clients were downloading content as well as metadata from the server, which is inefficient.

15MB package containing 10 unique files

Client	Bytes from Server	Bytes from Cache	Bytes Served
A	5.1MB	7.9MB	9.9MB
B	8MB	6MB	13.5MB
C	2.2MB	12MB	2.5MB

In this scenario it is obvious that the clients are simultaneously downloading significant amounts of content directly from the server. As these clients are using BITS to download the content, it is not possible to dynamically make optimum use of the available WAN bandwidth[2], and as more clients participate in the download, the chances of saturating the WAN link are increased.

5MB 10 file package (all files had identical content)

Client	Bytes from Server	Bytes from Cache	Bytes Served
A	4KB	5MB	0
B	500KB	3.8MB	458KB
C	570KB	3.2MB	500KB

The figures illustrate an interesting feature of BranchCache. Although we have a 5MB package, we have only downloaded a total of approximately 1MB from the server. The Program ran successfully with the entire package present within the ConfigMgr client cache. BranchCache was able to recognise the fact that the all files shared the same content and effectively reused data when supplying the clients. BranchCache data discovery behaviour further illustrates this point. Within this scenario, despite there being 10 files in the source, clients issued only 2 discovery requests as opposed when dealing with unique files, when a discovery request appears to be issued per file.

No dependency for any one client to be available at all times

The BranchCache Distributed cache model, it was observed that not all participating clients do not cache the entire content. During a test deployment of a 2.7GB OS image to 3 computers, only one client had the full 2.7GB in the BranchCache (although the ConfigMgr client cache had the whole content) while the others had around 1.3GB. The client that had the full image in its BranchCache was then switched off and the image deployed to another client in the subnet. This client had to obtain the balance of the image (1.4GB) from the server.

The other tests described in the previous section also revealed that clients had different volumes of data in their cache, indicating that the caching is indeed distributed and that in order for other clients to obtain all the content during a subsequent deployment, multiple clients would need to be available on the network to enable the client to obtain the total content from the local network.

Efficient use of available disk capacity on local workstations

Although hard disk capacity is pretty cheap these days, using workstation-class hardware to store distributed content should still make efficient use of the available space, especially if the workstation is being used to perform a user’s day job as well.

As noted previously, BranchCache will allocate 5% (default) of the disk capacity for use. When the cache is first used, we observed 50% of the allocated space being ‘used’, regardless of the size of the content. If the BranchCache becomes full, older content will be removed to make room for new content.

[2] The BITS client makes use of bandwidth available on the local network adapter and is not aware of the end-to-end bandwidth available on the WAN. As the local network adapter typically has significantly more bandwidth (e.g. 100Mb/s) than the WAN link (perhaps 1Mb/s), clients can quickly saturate the WAN link. It is possible to configure the BITS client to never exceed a set threshold (e.g. 20Kbps) between two specific times in the day, but this does not allow BITS to take advantage of available bandwidth beyond this threshold during quieter periods.

The BranchCache is separate from the ConfigMgr client cache. When using BranchCache with ConfigMgr for software distribution it is therefore necessary to ensure there is sufficient disk capacity for the content to be cached twice on each client (once in the BranchCache, once in the ConfigMgr cache). In practice, observed behaviour[3] suggests the ConfigMgr cache is populated from either the BranchCache on the local client or from the BranchCache on other clients, so the total capacity required on each client would not always be double the size of the content, but this must be the assumption when planning capacity.

Conclusion

For organizations that have Windows Server 2008 R2 and Windows 7 deployed throughout their environment, BranchCache operating in Distributed Cache mode offers an ideal solution for ad-hoc content retrieval from remote servers, such as obtaining documents from a file share or SharePoint site, accessing web content or running a non-mandatory (i.e. unscheduled) ConfigMgr Advertised Program without the need for a local server. In these scenarios it is very unusual for several clients to simultaneously request the same content.

In the scenario where ConfigMgr is used to schedule the simultaneous deployment of an application, patch or even OS image to a number of machines, our biggest concern was the potential to saturate the network, as it was observed that each client started obtaining significant amounts of content from the remote server before eventually settling down and retrieving remaining content from peer caches. The optimum solution will only download content once across the WAN, but our tests suggested that at least some of the content was downloaded by several clients if the same content is requested simultaneously.

Our tests indicated that once content had been cached in a given subnet, as it was distributed across a number of clients there was a higher chance that later requests for the same content would result in some data being downloaded across the WAN from the remote server if some of the original clients were powered off or removed – perhaps the reason that Microsoft does not recommend using Distributed Mode in locations that include laptops.

Finally, if updates to the branch are frequent, or include OS images for rebuilding faulty workstations, the amount of disk space required on each workstation when used with ConfigMgr can be significant. Content is aged out of the BranchCache as capacity is required, so applications that are only required occasionally in a branch may not be available even if they were previously cached.

[3] We observed that with any particular package deployment, the volume of data in each participating BranchCache was different, whereas the volume of data in each participating ConfigMgr cache was consistent. This suggests that the ConfigMgr cache was populated directly from peer BranchCaches as well as from the BranchCache on the local client.