A Guide to
Content Caching
Contents

What is content caching? .................................................................4
How does it work? ........................................................................4
Where are the cached files stored? .............................................5
What content types are supported by the Caching Services? ..........5
What if my network has multiple subnets? .................................6
What if I have multiple public IP addresses? ...............................6
Can I have more than one content cache? .................................10
How do I set up content cache clients, peers, or parents? ..........11
Performance best practices.........................................................14
Content caching best practices ..................................................14
Turn on performance mode for macOS Server.........................14
Configuring advanced content caching settings.......................15
Content Caching service command line tools ..........................21
What is content caching?

Content caching is a macOS service that helps reduce bandwidth usage and speed up software installation on Mac computers, iOS devices, and Apple TV devices. Content caching speeds up downloading of software distributed by Apple and data that users store in iCloud by saving content that local Mac computers, iOS devices, and Apple TV devices have already downloaded. The saved content is stored in a content cache on a Mac, and is available for other devices to retrieve without going out over the Internet.

You can use content caching on networks consisting of a NAT environment for the content cache and all devices, or on networks consisting of publicly routable IP addresses. Apple devices with iOS 7 or later and OS X 10.8.2 or later automatically contact a nearby content cache without any configuration.

Important: It’s strongly recommended that you set up content caching on a Mac that has a single wired Ethernet connection as its only connection to the network. Content caching can use a Wi-Fi connection instead of Ethernet, but performance might be affected.

How does it work?

After you enable content caching on a Mac, it keeps a copy of all content that local networked devices (called clients) download.

For example, when the first client on your network downloads a macOS update, the content cache keeps a copy of the update. When the next client on the network connects to the App Store to download the update, the update is copied from the content cache rather than from the App Store.

Because the local network is normally much faster than the Internet, the second client (and all subsequent clients) download updates much faster.
Where are the cached files stored?
The default location for cached content is the boot volume. You can choose an alternate location and specify how much of the volume is used by content caching. When the disk space for the cached content reaches the maximum you specified, or when the available space on the volume runs low, the content cache deletes content that hasn’t been used recently to make space for the next request.

What content types are supported by the Caching Services?
macOS Server and macOS include built-in support for caching the following software.

<table>
<thead>
<tr>
<th>iTunes and iBooks</th>
<th>macOS</th>
<th>iOS 7 and later &amp; Apple TV</th>
<th>Country and region restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• iTunes 11.0.4 and later (OS X and Windows versions)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iOS app purchases and app updates*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iBooks Store content (iOS 6 and later, and OS X Mavericks and later)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iOS updates downloaded using iTunes are not cached. iOS updates delivered “over the air” are cached instead.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• macOS updates</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Mac App Store (OS X Mountain Lion v10.8.4 or later is recommended)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Other software updates distributed by Apple (including updates to iTunes for macOS and printer drivers)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• GarageBand downloadable content</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Third-party apps from the Mac App Store and their updates</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iCloud data caching (photos and documents) for OS X El Capitan v10.11 and later</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iOS updates (over the air)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Apple TV updates (over the air)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iOS apps, Apple TV apps, and app updates</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• On-demand resources support for iOS 10 and later and tvOS 10 and later.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iCloud data caching (photos and documents) for iOS 9 and later</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iTunes U course materials from the iOS App Store and iBooks Store, as well as uploaded instructor materials such as audio, video, iWork, and iBooks Author files</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Certain mobile assets, such as Siri high quality voices, language dictionaries, and more</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Not all content is cached in all regions.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• As of December 2013, iTunes downloads are not cached in Brazil, Mexico, China, or Portugal. iBooks downloads are not cached in Canada.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• iTunes downloads might not be cached if a client’s IP address isn’t associated with your iTunes Store region.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What if my network has multiple subnets?

By default, content caching is limited to a specific subnet, but you can set it to provide content caching for:

- All combinations of subnets of the local network that share a common public IP address
- Any combination of subnets of publicly accessible IP addresses (with additional DNS setting support)

If your network has multiple subnets that share the same public IP address, all the subnets can take advantage of a single content cache.

What if I have multiple public IP addresses?

You can enable content cache discovery across multiple public IP addresses. If your network uses multiple public IP addresses to connect to the Internet, such that a content cache might register using a different address than a client uses for discovery, you need to provide both the content cache and the clients with a list of those addresses. Apple uses these lists to cross-match registration and discovery requests involving multiple public IP addresses.

To avoid manual configuration of clients, content caching uses DNS TXT records to publish the public IP address information for clients on your network. The TXT record needs to be published in the default DNS search domain used by your clients.

The correct data for the TXT record can be generated automatically or manually. In either case, you need to edit the DNS record, or give the settings to your DNS provider to create or edit the TXT record in the zone file.

Note: These records are necessary only for your internal network. External DNS doesn’t require the additional record.
Configure content caching to support multiple IP addresses
1. Choose Apple menu > System Preferences, then click Sharing.
2. Select Content Caching, then press and hold the Option key and click Advanced Options.
3. Click Clients.
4. Click the “My local networks” pop-up menu, then choose one of the following:
   - Use one public IP address
   - Use custom public IP addresses
5. If you chose “use custom public IP addresses,” click the Add button, then enter a range of public IP addresses.
   Repeat for any additional IP address ranges you want to enter.
6. You must create a DNS text record (see the tasks below), which describes the public IP addresses, for your client to use. Click DNS Configuration to get the DNS record settings.
7. When you finish the configuration, click OK.

Generate a DNS text record (Option 1)
When you set up content caching clients, if you choose “use custom public IP addresses” for “My local networks,” you have the option of generating the TXT record that can be copied and pasted into the zone file.
1. After adding the network ranges, click DNS Configuration.
2. Choose your DNS server type (BIND or Windows).
3. If you use BIND9 DNS, copy the generated TXT record and paste it into your DNS zone file.
4. If you use Windows DNS, replace the zone name variable and run the generated command on your Windows DNS computer.
Create a DNS text record manually (Option 2)

The syntax for specifying TXT records, and non-ASCII characters in TXT records, will vary for your DNS server. Both IPv4 and IPv6 are accepted, but only IPv4 is supported.

1. Add one or more TXT records to the zone file for your local domain on your DNS server. These records have the same format as DNS-SD TXT records (key-value pairs):

   ```
   name._tcp       10800 IN TXT    "[prs|prn]=addressRanges"
   ```

   For Example:
   ```
   _aaplcache._tcp 10800 IN TXT    \x2aprs=17.53.22.2-17.53.22.254,93.184.216.119
   _aaplcache._tcp 10800 IN TXT    \x12prn=\x24\x11\x35\x16\x02\x11\x35\x16\x0e\x14\x5d\xb8\xd8\xa77
   ```

   Here’s an example of three chained records:
   ```
   _aaplcache._tcp 10800 IN TXT    \x2bprs=17.250.1.1,17.250.2.1-17.250.2.254,more
   _aaplcache1._tcp 10800 IN TXT    \x0eprn=\x24\x11\xfa\03\x01\x11\xfa\03\x0e+
   _aaplcache2._tcp 10800 IN TXT    \x0eprs=17.250.4.5
   ```

2. Name the first record _aaplcache._tcp and subsequent records from _aaplcache1._tcp up to _aaplcache24._tcp, for a maximum of 25 chained records. The values of these records have the same format as DNS-SD TXT records (key-value pairs). Two keys are supported: prs and prn.

3. Choose which key to use:
   - **prs**: The value of the prs key is a sequence of comma-separated ranges of IP addresses in presentation format (ASCII dot notation). This syntax is for easy configuration. A range consists of either a single IP address or two IP addresses separated by a hyphen.
   - **prn**: The value of the prn key is a sequence of concatenated ranges of IP addresses in binary network-byte-order format. This syntax is for range sequences that are too long for a DNS record when specified in presentation format. Each range consists of one of these four items:
     - 0x14 single-IPv4-address
     - 0x16 single-IPv6-address
     - 0x24 first-IPv4-address last-IPv4-address
     - 0x26 first-IPv6-address last-IPv6-address

4. Chain the records together by putting a continuation marker on all but the last TXT record.
   - The prs and prn syntaxes may be mixed between records in the chain.
   - With the prs syntax, append ",more" to the end of the record value. With the prn syntax, append "+" (0x2b) to the end of the record value. The first record lacking such a continuation marker ends the chain.
   - A maximum of 25 records may be chained together. Records are resolved in batches of five at a time—that is, _aaplcache._tcp and _aaplcache1._tcp through _aaplcache4._tcp are resolved in parallel first, and if they all end with continuation markers, then _aaplcache5._tcp through _aaplcache9._tcp are resolved next, and so on.

   Here’s an example of three chained records:
   ```
   _aaplcache._tcp 10800 IN TXT    \x2bprs=17.250.1.1,17.250.2.1-17.250.2.254,more
   _aaplcache1._tcp 10800 IN TXT    \x0eprn=\x24\x11\xfa\x03\x01\x11\xfa\x03\x0e+
   _aaplcache2._tcp 10800 IN TXT    \x0eprs=17.250.4.5
   ```

The syntax for specifying TXT records, and non-ASCII characters in TXT records, may vary based on your DNS server. Some servers don’t need the leading length byte (\x2a, \x12, \x2b, \x0e, and \x0e in the examples, respectively) because they prepend it automatically. The examples are for illustration only.
Add TXT records to DNS zone file

Add the DNS TXT record to the zone that:

- Is authoritative for the domain
- Matches the default search domain for network clients

For example, if your organization provides DNS service for your own domain and is the source of authority for the host names for example.com, you put the caching TXT record in the example.com zone file.

Important: If you don’t host the authoritative DNS service for your domain, you can’t add the TXT record yourself. Coordinate with your DNS provider to have them add the TXT record provided.

For BIND9-based DNS on Linux, this file is in the /etc/bind/ directory, and the zone file name has been defined in /etc/bind/named.conf (most likely, “db.example.com.”).

If you use Windows DNS, do one of the following:

- Copy the generated command from the Client Configuration sheet, replace the zone name variable, and then run the command on your Windows DNS computer.
- Enter the TXT record information manually using the Windows Server administration tools.

If you’re hosting DNS on the Server app, do the following:

1. Copy the generated TXT record from the Client Configuration sheet.
2. Open a command-line text editor (for example, vi, emacs, or pico).
3. Open the zone file for editing (for example, /Library/Server/named/db.example.com).
4. At the end of the configuration file, paste the TXT record.
5. Update the serial number in the file.
6. Save the zone file.
7. Restart the service.

To test if the TXT record has been entered correctly, you can use the following command in Terminal:

dig -t TXT _aaplcache._tcp.<domain>

Example:

```
$ dig -t TXT _aaplcache._tcp.hcstechgroup.com
; <<>> DiG 9.9.7-P3  <<>> -t TXT _aaplcache._tcp.hcstechgroup.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64367
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4000
;; QUESTION SECTION:
;aaplcache._tcp.hcstechgroup.com.  IN  TXT

;; ANSWER SECTION:
aaplcache._tcp.hcstechgroup.com. 259200  IN  TXT  "prs=24.187.207.178-24.197.207.182"

;; Query time: 0 msec
;; SERVER: 192.168.110.229#53(192.168.110.229)
;; WHEN: Fri Dec 22 12:35:16 EST 2017
;; MSG SIZE  rcvd: 107
```
Can I have more than one content cache?

You can have multiple content caches for your network. If you have more than one, you can specify peer and parent relationships for the content caches. Content caching uses these relationships to determine which content cache is queried to fulfill a content request.

The following examples show two different ways the peer and parent relationships between multiple content caches in a network might be defined. In each example, the network is organized into a three-level hierarchy that has parent and grandparent content caches. The difference between them is how the peer content caches are defined. In the first example, peers are defined at each level of the hierarchy. In the second example, peers are defined only at the lowest level of the hierarchy.

You might choose a configuration matching the first example to maximize sharing among caches. If one of the content caches in a location is unavailable, another might already have the same content cached. Content caches 1-6 and parent caches 1-5 can use the parent selection policies first-available, random, round-robin, or sticky-available.

You might choose a configuration matching the second example to maximize the total size of the cache. Parent caches 1-5 do not share content with each other, and neither do main caches 1-3. Content caches 1-6 and parent caches 1-5 can use the parent selection policy url-path-hash.
How do I set up content cache clients, peers, or parents?

**Set up Clients**

On larger networks it is important to ensure that a content cache receives requests only from clients that are nearby. You can specify ranges of client IP addresses that a content cache is best positioned to serve, and optionally make it exclusive to those clients by choosing the "devices using custom local networks" option.

1. Choose Apple menu > System Preferences, then click Sharing.
2. Select Content Caching, then press and hold the Option key and click Advanced Options.
3. Click Clients.
4. Click the “Cache content for” pop-up menu, then choose one of the following options:
   - Devices using the same public IP address
   - Devices using the same local networks
   - Devices using custom local networks
   - Devices using custom local networks with fallback
5. If you chose “content caches using custom local networks,” click the Add (+) button, then enter a range of local IP addresses of peer content caches.
   Repeat for any additional IP address ranges you want to enter.
6. Click OK.

### Content caching client options

<table>
<thead>
<tr>
<th>Cache content for</th>
<th>Select an option that describes which devices this computer will cache content for:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Devices using the same public IP address: This computer caches content for devices that use the same public IP address as this computer.</td>
</tr>
<tr>
<td></td>
<td>• Devices using the same local networks: This computer caches content for devices that use the same local networks as this computer.</td>
</tr>
<tr>
<td></td>
<td>• Devices using custom local networks: This computer caches content for devices that use the specified local networks.</td>
</tr>
<tr>
<td></td>
<td>• Devices using custom local networks with fallback: This computer caches content for devices that use the specified local networks, and for devices that use the same public IP address as this computer when their preferred content cache is unavailable.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>My local networks</th>
<th>Select an option that describes your network:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Use one public IP address: Select this option when you have just one public IP address. The public IP address will be discovered automatically.</td>
</tr>
<tr>
<td></td>
<td>• Use custom public IP addresses: Select this option to use a specific public IP address, or if you use multiple public IP addresses. If you select this option, additional DNS configuration is required.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Add (+)</th>
<th>To specify a custom local network, click the Add button (+), then enter the starting and ending IP addresses. The addresses can be the same, to create a range of just one address.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Remove (-)</th>
<th>To remove a custom local network, select the address range, then click the Remove button (-).</th>
</tr>
</thead>
</table>

| DNS Configuration         | Click to generate configuration for a DNS TXT record. This button is available only if you select "use custom public IP addresses" for the “My local networks” option. |
**Set up peers**

You can use multiple content caches for your network. Content caches on the same network are called peers, and they share content with each other.

1. Choose Apple menu > System Preferences, then click Sharing.
2. Select Content Caching, then press and hold the Option key and click Advanced Options
3. Click Peers.
4. Click the “Share content with” pop-up menu, then choose one of the following options:
   - Content caches using the same public IP address
   - Content caches using the same local networks
   - Content caches using custom local networks
5. If you chose “content caches using custom local networks,” click Add (+), then enter a range of local IP addresses of peer content caches.
   Repeat for any additional IP address ranges you want to enter.
6. Click OK.

### Content caching peer options

| Share content with                  | Select an option that describes which content caches this content cache will share content with:
|------------------------------------|-----------------------------------------------------------------------------------
| Content caches using the same public IP address: This computer shares content with other content caches that use the same public IP address as this computer.
| Content caches using the same local networks: This computer shares content with other content caches that use the same local networks as this computer.
| Content caches using custom local networks: This computer shares content with other content caches that use the same custom local networks as this computer.

<table>
<thead>
<tr>
<th>Add (+)</th>
<th>To specify a custom local network, click the Add button, then enter the starting and ending IP addresses. The addresses can be the same, to create a range of just one address.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove (-)</td>
<td>To remove a custom local network, select the address range, then click the Remove button (-).</td>
</tr>
</tbody>
</table>
Set up parents
You can arrange your content caches in a hierarchy. These content caches are called parents, and they provide content to their children.
1. Choose Apple menu > System Preferences, then click Sharing.
2. Select Content Caching, then press and hold the Option key and click Advanced Options.
3. Click Parents.
4. Click Add (+), then enter the local IP address of a parent content cache.
   Repeat for any additional IP address ranges you want to enter.
5. Select a parent policy.
6. Click OK.

Content caching parent options

<table>
<thead>
<tr>
<th>Parent IP Addresses</th>
<th>The list of local IP addresses for parent caches. Enter the addresses in the order you want them used when a parent cache is selected.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>To add an IP address, click Add.</td>
</tr>
<tr>
<td></td>
<td>To delete an IP address, click Remove.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parent policy</th>
<th>The policy to use when selecting a parent cache to use. Click the pop-up menu, then select a parent policy:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• First available: Always use the first parent in the parents list that is available. This is useful for designating permanent primary, secondary, and subsequent parents.</td>
</tr>
<tr>
<td></td>
<td>• Hash: Hash the path part of the requested URL so that the same parent is always used for the same URL. This is useful for maximizing the size of the combined caches of the parents.</td>
</tr>
<tr>
<td></td>
<td>• Random: Choose a parent at random. This is useful for load balancing.</td>
</tr>
<tr>
<td></td>
<td>• Round robin: Rotate through the parents in order. This is useful for load balancing.</td>
</tr>
<tr>
<td></td>
<td>• Sticky available: Starting with the first parent in the parents list, always use the first parent that is available. Use that parent until it becomes unavailable, then advance to the next one. This is useful for designating floating primary, secondary, and subsequent parents.</td>
</tr>
</tbody>
</table>
Performance best practices
Content caching is primarily affected by two main factors: connectivity and hardware configurations. You get the best performance from your content cache by connecting it to your network using Gigabit Ethernet. The content cache can serve hundreds of clients concurrently, which can saturate a Gigabit Ethernet port. Therefore, in most small to medium scale deployments, the performance bottleneck is usually the bandwidth of your local network.

To determine if your Mac is the performance bottleneck when a large number of clients are accessing the content cache simultaneously, check the processor usage for the AssetCache process in Activity Monitor (open Activity Monitor, choose View > All Processes, then click CPU). If the processor usage is constantly at or near the maximum, you may want to add additional content caches to distribute the requests across multiple computers.

Also, if your Mac is in an environment where clients download large amounts of a wide variety of content, be sure to set the cache size limit high enough. This prevents the content cache from deleting cached data frequently, which may necessitate downloading the same content multiple times, thereby using more Internet bandwidth.

Content caching best practices
The following are best practices for content caching. Whenever possible, you should follow these recommendations:

• Allow all Apple push notifications.
• Don’t use manual proxy settings.
• Don’t proxy client requests to content caches.
• Bypass proxy authentication for content caches.
• Specify a TCP port for caching.
• Manage inter-site caching traffic.
• Block rogue cache registration.

Turn on performance mode for macOS Server
Performance mode changes the system parameters of your Mac. These changes take better advantage of your hardware for demanding server applications. A Mac with macOS Server that needs to run high-performance services can turn on performance mode to dedicate additional system resources for server applications.

OS X El Capitan 10.11 and later
To turn on performance mode in OS X El Capitan 10.11 and later, use the nvram command to adjust the boot-args NVRAM variable. If you reset NVRAM, this setting is cleared.

This command displays the boot-args NVRAM variable. If you see serverperfmode=1, performance mode is already turned on.

```
nvram boot-args
```

If performance mode isn’t already turned on, you can enable it by setting serverperfmode=1 in the boot-args NVRAM variable with this command:

```
sudo nvram boot-args="serverperfmode=1 $(nvram boot-args 2>/dev/null | cut -f 2 -)"
```

To turn off performance mode, use this command:

```
sudo nvram boot-args="$(nvram boot-args 2>/dev/null | sed -e $'s/boot-args\t//;s/serverperfmode=1/\t/')"
```
Configuring advanced content caching settings
You can fine-tune content caching for your network configuration. You can set advanced configuration settings for the content cache either by using the command line in Terminal or by modifying the value of keys in the /Library/Preferences/com.apple.AssetCache.plist file. For some changes to take effect, content caching must be stopped and restarted.

You can set advanced configuration parameters using the command:
```
sudo -u _assetcache defaults write /Library/Preferences/com.apple.AssetCache.plist
```
followed by the command
```
sudo AssetCacheManagerUtil reloadSettings.
```
Use the command AssetCacheManagerUtil settings to view the standard (non-advanced) settings. For more information about the AssetCacheManagerUtil command, see the AssetCacheManagerUtil(8) man page.

Use the defaults command
You can set both simple and complex keys with the defaults command. For example, to set the Interface key to en1, execute this command as an administrator:
```
$ sudo -u _assetcache defaults write /Library/Preferences/com.apple.AssetCache.plist Interface -string en1
```
ListenRanges is a complex key that takes an array of dictionaries. For example, execute this command as an administrator to set two IP address ranges for the ListenRanges key:
```
$ sudo -u _assetcache defaults write /Library/Preferences/com.apple.AssetCache.plist ListenRanges '{ { first = 10.0.0.1; last = 10.0.0.254; }, { first = 10.1.0.1; last = 10.1.0.254; } }'
```
After using the defaults command, be sure to run the following command to reload the content cache settings:
```
$ sudo AssetCacheManagerUtil reloadSettings
```
For more information about the defaults command, see the defaults(1) man page.

Earlier versions of macOS
To turn on performance mode in OS X Mountain Lion, OS X Mavericks, or OS X Yosemite, use the serverinfo command in Terminal.

To get the current mode, use the command:
```
serverinfo --perfmode
```
To set high performance mode:
```
serverinfo --setperfmode 1
```
To turn off high performance mode:
```
serverinfo --setperfmode 0
```
Transitioning to or from high performance mode requires a restart.
Caching configuration plist keys and values

Important: Don’t change any settings in the com.apple.AssetCache.plist file other than the ones described in the table.

A key can have a value that’s clamped between two values. The key value can be any number in the range between the low and high values. If it’s set below the lower-bound value, the lower-bound value is used. If it’s set above the upper-bound value, the upper-bound value is used. For example, PeerDownloadTimeout is clamped between 5 and 300. If it’s set to 301 or 1000, then the value is set to 300. If it’s set to 4 or -10, then the value is set to 5.

Some changes take effect after you run AssetCacheManagerUtil reloadSettings; others require that you stop and then restart content caching. The only keys that support reloadSettings are those that can also be set in Content Caching preferences. To set values in Content Caching preferences, choose Apple menu > System Preferences, click Sharing, then click Content Caching.

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgeForLowSpaceAlert</td>
<td>When content is purged from the content cache because it’s low on disk space, and the purged content was added to the content cache less than this many days ago, you receive a low space alert.</td>
<td>30 (days)</td>
</tr>
<tr>
<td>AllowCacheDelete</td>
<td>Allow content to be purged from the cache automatically when the computer needs disk space for other apps.</td>
<td>Yes</td>
</tr>
<tr>
<td>AllowImports</td>
<td>Allow import (upload) requests.</td>
<td>Yes</td>
</tr>
<tr>
<td>AllowPersonalCaching</td>
<td>Allow the caching of users’ iCloud data. At least one of the AllowPersonalCaching or AllowSharedCaching keys must be yes. You can also set this value in Content Caching preferences.</td>
<td>Yes</td>
</tr>
<tr>
<td>AllowSharedCaching</td>
<td>Controls the caching of non-iCloud content, such as apps and software updates. At least one of the AllowPersonalCaching or AllowSharedCaching keys must be yes.</td>
<td>Yes</td>
</tr>
<tr>
<td>AllowWirelessPortable</td>
<td>Allow portable computers that have only Wi-Fi network connections to run content caching.</td>
<td>Yes</td>
</tr>
<tr>
<td>CacheLimit</td>
<td>The maximum number of bytes of disk space that will be used for the content cache. You can also set this value in Content Caching preferences.</td>
<td>0 (unlimited)</td>
</tr>
<tr>
<td>DatabaseUpdateInterval</td>
<td>How often the content cache saves changes to its on-disk database. Raising the interval increases the risk of losing cached content after a power failure. The maximum is 3600 seconds (1 hour). An interval of 0 means always update the database immediately, with no delay, which decreases performance.</td>
<td>5 (seconds)</td>
</tr>
<tr>
<td>DataPath</td>
<td>The path to the directory used to store cached content. Changing this setting manually does not automatically move cached content from the old to the new location. To move content automatically, use Content Caching preferences (see Select a volume for caching). You can also set this value in Content Caching preferences.</td>
<td>/Library/Application Support/Apple/AssetCache/Data</td>
</tr>
<tr>
<td>ImportMaxRate</td>
<td>The maximum number of bytes per second at which the content cache receives data from each client. A value of 0 indicates an unlimited number of bytes per second.</td>
<td>0 (bytes per second)</td>
</tr>
<tr>
<td>ImportMinRate</td>
<td>The minimum number of bytes per second that clients must sustain while importing (uploading) content. The content cache stops imports that transfer data slower than this rate. The minimum rate is 100 bytes per second.</td>
<td>2000 (bytes per second)</td>
</tr>
</tbody>
</table>
## Key

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImportRateAttenuation</td>
<td>The percentage of attenuation added to the upload time. Clamped minimum is 0% attenuation. Values too large will exceed the ImportTimeout and cause failures.</td>
<td>0.20 (percentage)</td>
</tr>
<tr>
<td>ImportTimeout</td>
<td>How long, in seconds, to allow an import (upload) from a client to sit idle before giving up. The minimum is 10 seconds.</td>
<td>300 (seconds)</td>
</tr>
<tr>
<td>Interface</td>
<td>The BSD name of a network interface to be used by the content cache.</td>
<td>Listen on all interfaces</td>
</tr>
<tr>
<td>ListenRanges</td>
<td>An array of dictionaries describing the range of client IP addresses to serve.</td>
<td>none</td>
</tr>
<tr>
<td>ListenRangesOnly</td>
<td>If ListenRangesOnly is set to true, the content cache provides content only to clients in the ranges specified by the ListenRanges key. If you want to use the ListenRangesOnly key, you must also specify the ListenRanges key.</td>
<td>No</td>
</tr>
<tr>
<td>LocalSubnetsOnly</td>
<td>Whether or not the content cache should offer content only to clients on the same immediate local network as the content cache, rather than to clients on all local networks reachable by the content cache.</td>
<td>Yes</td>
</tr>
<tr>
<td>LogClientIdentity</td>
<td>Determines whether or not the content cache should log the IP address and port number of the clients that request content.</td>
<td>No</td>
</tr>
<tr>
<td>MaxConcurrentClients</td>
<td>This limit is to prevent content caching from running out of file descriptors. Apple does not guarantee that a content cache can achieve 3400 concurrent clients.</td>
<td>3400</td>
</tr>
<tr>
<td>MaxParentDepth</td>
<td>The maximum number of times, for a single request, that a child content cache will forward the request to a parent content cache. Requests that are too deep (forwarding chain is too long) are forced to the origin rather than to a parent.</td>
<td>8</td>
</tr>
<tr>
<td>MaxPeersToQuery</td>
<td>The maximum number of peer content caches to ask for content.</td>
<td>0 (unlimited)</td>
</tr>
<tr>
<td>OriginDownloadTimeout</td>
<td>How long, in seconds, to allow a download from Apple's servers to sit idle before giving up (and possibly trying the download again). Clamped between 5 and 300 seconds, inclusive.</td>
<td>60</td>
</tr>
<tr>
<td>OriginUploadTimeout</td>
<td>How long, in seconds, to allow an upload to an origin server to sit idle before giving up. (Clamped between 5 and 3600 seconds, inclusive.)</td>
<td>600</td>
</tr>
<tr>
<td>ParentDownloadTimeout</td>
<td>How long, in seconds, to allow a download from a parent content cache to sit idle before giving up (and possibly trying the download again). Clamped between 5-300 seconds, inclusive.</td>
<td>60</td>
</tr>
<tr>
<td>ParentRetryInterval</td>
<td>How long, in seconds, to ignore parent content caches after they have accrued five consecutive network failures or server errors. Clamped between 30-3600 seconds, inclusive.</td>
<td>900</td>
</tr>
<tr>
<td>Parents</td>
<td>A list of the local IP addresses of other content caches from which this cache should download or upload content instead of downloading from or uploading to Apple directly. Invalid addresses and addresses of computers that are not content caches are ignored. Parent caches that become unavailable are skipped according to the ParentRetryInterval. If all parent content caches become unavailable, the content cache will download from or upload to Apple directly until a parent content cache becomes available again. You can also set this value in Content Caching preferences.</td>
<td>none</td>
</tr>
<tr>
<td>Key</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| ParentSelectionPolicy       | The policy to use when choosing among more than one configured parent content cache. With every policy, parent caches that are temporarily unavailable are skipped.  
                                 | The policies are:  
                                 | • first-available: Always use the first parent in the Parents list that is available. This is useful for designating permanent primary, secondary, and subsequent parents.  
                                 | • url-path-hash: Hash the path part of the requested URL so that the same parent is always used for the same URL. This is useful for maximizing the size of the combined caches of the parents.  
                                 | • random: Choose a parent at random. This is useful for load balancing.  
                                 | • round-robin: Rotate through the parents in order. This is useful for load balancing.  
                                 | • sticky-available: Starting with the first parent in the Parents list, always use the first parent that is available. Use that parent until it becomes unavailable, then advance to the next one. This is useful for designating floating primary, secondary, and subsequent parents.  
                                 | You can also set this value in Content Caching preferences.                                                                                                                                          | round-robin |
| ParentUploadTimeout         | How long, in seconds, to allow an upload to a parent content cache to sit idle before giving up. Clamped between 5-3600 seconds, inclusive.                                                                 | 600         |
| PeerDownloadTimeout         | How long, in seconds, to allow a download from a peer content cache to sit idle before giving up (and possibly trying the download again). Clamped between 5 and 300 seconds, inclusive.                                                             | 30          |
| PeerFilterRanges            | When PeerFilterRanges is an array (of entries like those for ListenRanges), the content cache filters and sorts its list of peers according to the ranges in the array.  
                                 | The content cache only queries peers that are in the PeerFilterRanges. The filtering and sorting are applied before truncating the list of peers at MaxPeersToQuery entries (if that setting is present).  
                                 | When PeerFilterRanges is an empty array, the content cache will not query any peers.  
                                 | When PeerFilterRanges is Boolean true, the content cache does the same as above, but uses the ListenRanges rather than the PeerFilterRanges value.  
                                 | When PeerFilterRanges is any other type or the value is missing, the content cache neither filters nor sorts its list of peers before truncating the list at MaxPeersToQuery entries.  
                                 | PeerFilterRanges only affects the list of other content caches this content cache queries for content and downloads. It has no effect on incoming requests for content from any other content cache.  
                                 | You can also set this value in Content Caching preferences.                                                                                                                                          | none        |
## PeerListenRanges

When PeerListenRanges is an array of dictionaries where each dictionary represents an IP address range, the content cache will only successfully respond to peer cache queries from content caches with an IP address contained within this array of ranges.

When PeerListenRanges is an empty array, the content cache will respond with an error to cache queries from any other content cache.

When PeerListenRanges is Boolean true, the content cache will use the ListenRanges value rather than the PeerListenRanges value to decide which other content caches it will successfully respond to cache queries from.

When PeerListenRanges is any other type or the value is missing, the content cache will successfully respond to cache queries from all other content caches.

PeerListenRanges only affects which content caches this content cache will successfully respond to cache queries from. It has no effect on the list of peers this content cache will query for content and download content from.

When a content cache responds with an error from a cache query, the querying content cache marks the responding content cache as unfriendly and does not attempt to query it again until the PeerRetryInterval has elapsed.

You can also set this value in Content Caching preferences.

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeerListenRanges</td>
<td>When PeerListenRanges is an array of dictionaries where each dictionary represents an IP address range, the content cache will only successfully respond to peer cache queries from content caches with an IP address contained within this array of ranges. When PeerListenRanges is an empty array, the content cache will respond with an error to cache queries from any other content cache. When PeerListenRanges is Boolean true, the content cache will use the ListenRanges value rather than the PeerListenRanges value to decide which other content caches it will successfully respond to cache queries from. When PeerListenRanges is any other type or the value is missing, the content cache will successfully respond to cache queries from all other content caches. PeerListenRanges only affects which content caches this content cache will successfully respond to cache queries from. It has no effect on the list of peers this content cache will query for content and download content from. When a content cache responds with an error from a cache query, the querying content cache marks the responding content cache as unfriendly and does not attempt to query it again until the PeerRetryInterval has elapsed. You can also set this value in Content Caching preferences.</td>
<td>none</td>
</tr>
<tr>
<td>PeerLocalSubnetsOnly</td>
<td>Whether or not the content cache should only peer with other content caches on the same immediate local network, rather than with content caches that use the same public IP address as this computer. When PeerLocalSubnetsOnly is true, the content cache only queries and successfully responds to peer queries from content caches on the same immediate local network. When PeerLocalSubnetsOnly is true, it overrides the configuration of PeerFilterRanges and PeerListenRanges. When PeerLocalSubnetsOnly is false, the content cache defers to PeerFilterRanges and PeerListenRanges for configuring the peering restrictions. When PeerLocalSubnetsOnly is true and the network changes, the local network peering restrictions update appropriately. You can also set this value in Content Caching preferences.</td>
<td>Yes</td>
</tr>
<tr>
<td>PeerNotifyTimeout</td>
<td>How long, in seconds, to wait for replies from peer content caches when pinging them on startup. Clamped between 5 and 300 seconds, inclusive.</td>
<td>30</td>
</tr>
<tr>
<td>PeerQueryTimeout</td>
<td>How long, in seconds, to wait for replies from peer content caches when asking them about content in their caches. Clamped between 1 and 60 seconds, inclusive.</td>
<td>5</td>
</tr>
<tr>
<td>Key</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>PeerRetryInterval</td>
<td>How long, in seconds, to ignore peer content caches after they have accrued three consecutive notify or query failures. After the retry interval has passed, peer content caches are restored to the list of peers to query for content. Clamped between 30 and 3600 seconds, inclusive.</td>
<td>900 (seconds)</td>
</tr>
<tr>
<td>PersonalCacheLimit</td>
<td>Limit how much disk space the content cache uses for cached iCloud data, in bytes. The PersonalCacheLimit must not exceed CacheLimit.</td>
<td>0 (unlimited)</td>
</tr>
<tr>
<td>Port</td>
<td>The TCP port number on which content caching accepts requests for uploads or downloads.</td>
<td>0 (use a random port)</td>
</tr>
<tr>
<td>PruneAffinitiesAge</td>
<td>User affinities older than this number of days are removed from the affinities cache automatically. User affinities provide hints to clients as to where their content is cached, for improved performance. Pruning user affinities has no effect on cached content. Clamped minimum is 7 days.</td>
<td>30 (days)</td>
</tr>
<tr>
<td>PruneAffinitiesInterval</td>
<td>How often, in days, the content cache should scan for and remove user affinities older than PruneAffinitiesAge days. User affinities, used only by iCloud, provide hints to clients as to where their content is cached, for improved performance. Pruning user affinities has no effect on cached content. Clamped minimum is one day.</td>
<td>7 (days)</td>
</tr>
<tr>
<td>PruneAssetsAge</td>
<td>Content that has not been requested in this number of days is removed from the content cache automatically.</td>
<td>120 (days)</td>
</tr>
<tr>
<td>PruneAssetsInterval</td>
<td>How often, in days, the content cache should scan for and remove content older than PruneAssetsAge days.</td>
<td>7 (days)</td>
</tr>
<tr>
<td>PublicRanges</td>
<td>Ranges of public IP addresses the cloud servers should use for matching clients to content caches. You can also set this value in Content Caching preferences.</td>
<td>no default</td>
</tr>
<tr>
<td>ReservedVolumeSpace</td>
<td>The minimum number of bytes of free disk space to be maintained for the volume that stores the cached content.</td>
<td>2000000000 (2 GB)</td>
</tr>
<tr>
<td>TerminationTimeout</td>
<td>How long, in seconds, the content cache should try to deregister when it is being stopped. Deregistering informs clients that the content cache is no longer available so they won’t try to use that content cache anymore (or until the content cache is started again). Clamped between 1 and 60 seconds, inclusive.</td>
<td>10 (seconds)</td>
</tr>
<tr>
<td>Verbose</td>
<td>When Verbose=true the content cache logs a little more information about its activities. The increased logging can reduce performance. This setting is not recommended for long-term use. Use the log command in the Terminal app to view content caching logs. See the log(1) man page. Content caching logs to the subsystem com.apple.AssetCache. For example: log show --predicate 'subsystem == &quot;com.apple.AssetCache&quot;' You can also use the Console app to view the logs.</td>
<td>No</td>
</tr>
</tbody>
</table>
About the ListenRanges key
You can use the ListenRanges key to specify preferred content caches in advanced network topologies where multiple content caches are used behind the same public IP address. For example:

- caching1.example.com uses the ListenRanges key to specify a range of 10.0.0.1 through 10.0.0.254 and 10.1.0.1 through 10.1.0.254.
- caching2.example.com uses the ListenRanges key to specify a range of 10.1.0.1 through 10.1.0.39 (note the overlap with the second range of caching1).
- If a client whose IP address is 10.0.0.10 requests content, it's directed to caching1.
- If a client whose IP address is 10.1.0.10 requests content, it's directed to either caching1 or caching2, selected randomly.
- If a client whose IP address is 10.2.0.10 requests content, it's directed to either caching1 or caching2, selected randomly.
- If caching1 is shut down or loses power but caching2 remains available, all clients are directed to caching2.

Content Caching service command line tools
To view the status of the caching service and see the peering server IP's associated with this server run the following:
```
/usr/bin/AssetCacheManagerUtil status
```

To view the settings configured for content caching:
```
/usr/bin/AssetCacheManagerUtil settings
```

To view logging for caching content:
```
log show --predicate 'subsystem == "com.apple.AssetCache"'
```

To revert logging on the caching server back to standard levels run the following:
```
sudo -u _assetcache defaults write /Library/Preferences/com.apple.AssetCache.plist Verbose -bool NO
```

Note: Do not leave verbose logging enabled full time as it can cause excessive load on the server as well as fill up the disk space! More of these settings are documented in the Configure advanced content caching settings

To view the logging, including debug if turned on:
```
log show --predicate 'subsystem == "com.apple.AssetCache"' --debug
```

Extended logging determines whether or not the server should log the IP address and port number of the client requesting each asset. To enable this use:
```
sudo -u _assetcache defaults write /Library/Preferences/com.apple.AssetCache.plist LogClientIdentity -bool YES
```

If the client has multiple external facing IP addresses you will need to enter a TXT record. This is well documented here. To test this DNS TXT entry was successfully inserted you can test this using:
```
dig -t TXT _aaplcache._tcp.<domain>
```