GVP Deployment Guide

Optimizing GVP Performance Through HTTP Caching

12/19/2025

Optimizing GVP Performance Through HTTP Caching

This topic describes how HTTP caching works in GVP and how to configure caching to improve GVP performance.

HTTP Caching in GVP

HTTP caching is an important aspect of GVP deployment configuration, because it has significant impact on the performance, scalability, and robustness of the deployment. Properly designed caching rules govern properties such as, freshness and cacheability.

These rules enable the majority of HTTP requests to be fulfilled by the cache, while limiting the number of HTTP GET requests to those that are sent when the resource cannot be cached, cannot be found in the cache, or the expired cache entry requires revalidation.

As a result, caching lessens the load on the HTTP server, reduces network traffic, improves fetch response time, and increases fault-tolerance if the HTTP server becomes unavailable.

GVP can perform the caching function itself by using the Fetching Module's in-memory caching or the Squid Caching proxy (See GVP Caching), or it can use an external server a caching appliance, or a web proxy server.

Caching Within the Media and Call Control Platforms

The implementation of HTTP caching on the Media Control Platform and Call Control Platform complies fully with the HTTP 1.1 specification. When the Media Control Platform or Call Control Platform make an HTTP request, the platform handles the request in one of the following ways:

- The requested resource is not found in the cache. The Media Control Platform or Call Control Platform sends an HTTP GET request, and the HTTP server returns an HTTP response. If the response is cacheable, it is added to the cache.
- The requested resource is found in the cache, and the cache meets all the cache freshness requirements. The cache is used to satisfy the request, and an HTTP GET request is not sent.
- The requested resource is found in the cache, but it does not satisfy the freshness requirements. The Media Control Platform or Call Control Platform sends a conditional HTTP GET request, which contains an If-Modified-Since header, to validate the cache. The platform also sends the E-Tag header if the E-

Tag header is provided in the cached HTTP response.

- If the resource has not changed, the HTTP server returns an HTTP 304 response code. The HTTP response does not contain a message body, and the Media Control Platform or Call Control Platform uses the cache to satisfy the request.
- If the resource has changed, the HTTP server returns a new HTTP response. The Media Control Platform or Call Control Platform removes the old cache entry and, if the new response is cacheable, adds it to the cache.

Controlling Cache Size

The in-memory cache has a limited size. Therefore, Genesys recommends that you use the following Media Control Platform and Call Control Platform configuration options to control the size of the cache, the size of each cache entry, and the number of cache entries:

- `[fm]cachemaxsize`
- `[fm]cachemaxentrysize`
- `[fm]cachemaxentrycount`

Internally, the platform manages cache entries in an ordered list. A cache entry is moved to the front of the list when it is being accessed, and the cache entry that was used least recently is removed when the cache runs out of space.

Real-Time Cache Clearing

You can clear the internal in-memory cache without restarting. Framework can send a custom message to the MCP or CCP, and the receiving program then clears the cache. Following is the configuration procedure, for the Windows and Linux operating systems.

Procedure: Clearing the Realtime Cache (Windows)

Enable clearing the internal in-memory cache without restarting, using MCP or CCP.

1. Verify that Genesys Management Framework 8.1 or above is installed.
2. Copy the file `mlcmd.exe` into the same directory that holds the `clearfmcache.bat` and `clearfmcache.txt` files.
3. Edit the `clearfmcache.bat` shell scripts in the MCP `/bin` directory (or the CCP `/bin` directory) as follows:
   a. Populate the `CSUSER`, `CSPASSWORD`, and `CSAPPNAME` variables with the same Management Framework credentials that you use to log in to the Genesys Administrator.
   b. Populate the `SCSHOST` variable with the address of name of the machine hosting the Solution Control Server (SCS).
   c. Modify the `SCSPORT` variable, if it is deployed on a non-default port. The configuration is complete.
4. To clear the cache, run the script clearfmcache.bat at the command prompt.

Procedure: Clearing the Realtime Cache (Linux)

Enable clearing the internal in-memory cache without restarting, using MCP or CCP.

1. Verify that Genesys Management Framework 8.1 or above is installed.
2. Go to the Genesys Management Framework - Solution Control Server `/bin` directory.
3. Copy the *mlcmd* command into your to your MCP `/bin` directory (or the CCP `/bin` directory).
   - For Linux, that file is named `mlcmd_64`.
4. Edit the `clearfmcache.sh` shell scripts in the MCP `/bin` directory (or the CCP `/bin` directory) as follows:
   a. Populate the `CSUSER`, `CSPASSWORD`, and `CSAPPNAME` variables with the same Management Framework credentials that you use to log in to the Genesys Administrator.
   b. Populate the `SCSHOST` variable with the address of name of the machine hosting the Solution Control Server (SCS).
   c. Modify the `SCSPORT` variable to point to your SCS, if it is deployed on a non-default port.

   The configuration is complete.
5. To clear the cache, run the script `./clearfmcache.sh` at the command prompt.

Cache Control

You can apply cache control settings on either the server side or the client side. In general, it is not necessary to apply the settings on both sides. The decision is usually based on system access policies and maintenance considerations. For example, an application developer who does not have control of the web server settings must apply the settings on the client side.

Server-Side Cache Control

HTTP servers, such as Microsoft Internet Information Service (IIS) and Apache, enable you to apply the following cache control settings to the HTTP resources:

- Expires immediately.
- Expires after a specific amount of time for example, after 5 minutes.
- Expires at a specific date and time. For example, `May 2, 2011 at 9:00 AM`.
- Expires a specific amount of time after the content was last modified. (Not currently supported by IIS.)

Based on these settings, the HTTP Server will add either an Expires header, or a Cache-Control: max-age header to the HTTP response. The Expires header indicates that the response will expire at the specified time, and the Cache-Control: max-age header indicates that the response will expire after the max-age amount of time, which is expressed in seconds.

> **Tip**
> If the Media Control Platform or Call Control Platform receives an HTTP response with both the Expires and Cache-Control: `max-age` headers, the platform ignores the Expires header, in accordance with the HTTP 1.1 specification.

Client-Side Cache Control

GVP uses properties or attributes that specify maximum age and staleness (Maxage and Maxstale Attributes) to implement cache control by VoiceXML and CCXML applications.

VoiceXML and CCXML applications support the following cache control properties or attributes:

VoiceXML

- The `audiomaxage/audiomaxstale` properties or the `maxage/maxstale` attributes of `<audio>`.
- The `datamaxage/datamaxstale` properties or the `maxage/maxstale` attributes of `<data>`.
- The `documentmaxage/documentmaxstale` properties or the `maxage/maxstale` attributes of `<choice>`, `<goto>`, `<link>`, `<subdialog>`, or `<submit>`.
- The `grammarmaxage/grammarmaxstale` properties or the `maxage/maxstale` attributes of `<grammar>`.
- The `scriptmaxage/scriptmaxstale` properties or the `maxage/maxstale` attributes of `<script>`.
- The `maxage/maxstale` SIP REQUEST-URI parameters for the initial VoiceXML page.

CCXML

- The `maxage/maxstale` attributes of `<createccxml>`, `<fetch>`, `<dialogprepare>`, `<dialogstart>`, or `<script>`.
- The `maxage/maxstale` parameters in the HTTP POST request to the `createsession` processor.
- The `maxage/maxstale` SIP REQUEST-URI parameters for the initial CCXML page.

Maxage and Maxstale Attributes

Maxage

- Maxage Indicates that the client does not use cached content that is older than the specified time (in seconds). Setting the maxage attribute to a non-zero value enables you to force the platform to get a fresh copy of a resource before the cached copy expires. A fresh copy can be unconditionally requested by setting maxage value to 0.

> **Tip**
> If the client specifies the maxage attribute, and the cache already contains an expiration time that is calculated based on the Expires or Cache-Control: max-age header from the HTTP response, the more restrictive rule (in other words, the rule that results in an earlier expiration time) takes effect.

Maxstale

- `Maxstale` Indicates that the document does not use cached content that exceeds its expiration time by the specified amount of time (in seconds). When the `maxstale` attribute is used, an expired cache that is not too stale (according to the rules of HTTP 1.1) can be used. For example, an author who does not have direct server-side control of the expiration dates of large static files can avoid unnecessary fetching by allowing cached copies to be used after expiration.

The `maxage` or `maxstale` attribute value is first used to calculate the freshness of a cache entry. If the cache entry is not fresh enough, the `maxage` or `maxstale` attribute value is also sent in the Cache-Control header of the HTTP request, so that the HTTP proxy and server can generate the response, based on these settings.

Non-Cacheable Substrings

The Media Control Platform and Call Control Platform support the `[fm]no_cache_url_substring` configuration option, which defines a comma-delimited list of substrings. If an HTTP REQUEST-URI parameter contains any of these substrings, its response is not cached. The default list of non-cacheable substrings is: `cgi-bin, jsp, asp, ?`

Configuration Recommendations

This section provides recommendations for configuring cache control for dynamic and static resources.

Identifying Dynamic and Static HTTP Resources

HTTP resources are categorized as:

- Dynamic Refers to those resources that generate responses, based on the information that is provided in the HTTP requests. Java Server Page (JSP) and Common Gateway Interface (CGI) pages are typical examples of dynamic resources.

- Static Refers to those resources that are predefined. The same content is returned, regardless of the HTTP requests.

Static resources do get updated, but at different frequencies. For example, each one of the following static resources requires a different update frequency:

- An image file that contains a company logo might never change.
- A video file that contains the latest movie trailers might be updated anytime during normal business hours.
- A VoiceXML page that contains a dynamic TTS prompt for example, to play the customer s account balance, might be updated occasionally at any time.

Registration for ECC Variables Static and Dynamic

The CTI Connector (ICM) supports static and dynamic registration of ECC variables.

Static Registration of ECC Variables

When the CTI Configuration parameter [ICMC]eccvariablelist is populated with the desired ECC variables, during start up the CTI Connector sends a REGISTER_VARIABLES message to ICM for registration. The CTI Connector support configuration of ECC variable names along with their tag values (optional) as a comma-separated string.

Dynamic Registration of ECC Variables

If the new ECC variables are received (i.e., apart from those configured ECC variables through [ICMC]eccvariablelist) during the call setup message from the Media Gateway (incoming call), or during the call setup/transfer/end message from the VXML application in MCP, then CTIC will register these ECC variables on the fly, by a sending REGISTER_VARIABLES message.

The ECC variables format is mandatory: `ICMC_ECC_user<variable name>`
...where CTIC submits to ICM only the `user<variable name>` portion as the ECC variable name.

Squid HTTP Proxy

In GVP 8.1.1 and earlier releases, the Squid HTTP Proxy is a mandatory component, because the Fetching Module relies on Squid to support HTTP 1.0-compliant caching.

In GVP 8.1.2 and later releases, Squid is an optional component, because the Media Control Platform and Call Control Platforms are integrated with an in-process Fetching Module library that supports HTTP 1.1-compliant caching.

However, you might still want to take advantage of the features that are provided by Squid, for example:

- The overall size of cacheable resources is too large to be stored entirely in the in-memory cache of the GVP processes. Squid implements disk-based caching, which is more suitable for caching large amounts of data.
- To make use of the sophisticated access control options that Squid provides.

To use Squid or any other HTTP Proxy, use the [fm]http_proxy configuration option to specify the IP address and port number of the proxy.

Cache Control Settings

As described in Cache Control, cache control settings can be applied either on the server side or the client side.

Dynamic Resources

Never cache dynamic resources. Apply cache control settings for dynamic resources in one of the following ways:

- Server side Configure the resource to expire immediately.
- Client side Ensure that the `[fm]no_cache_ url_substring` configuration option covers the URLs that are used by the dynamic resources.

Static Resources That Never Change

Use caching as much as possible to serve static content that never changes, and avoid unnecessary conditional GET requests. Apply cache control settings for static resources that never change in one of the following ways:

- Server side Configure the cache to expire 30 days after access.
- Client side Configure the `maxage` attribute to a large number, such as `2592000` (30 days, expressed in seconds).

Static Resources That Might Change (Visible Immediately)

If the resource update must be immediately visible to the client, enable the response to be stored in the cache, and configure the `maxage` attribute with a value of `0` to ensure that a conditional GET request is sent for validation. In most cases, an HTTP 304 response is returned, which is still more efficient that receiving a `200 OK` response with the full HTTP message-body.

Apply cache control settings for static resources that might change and that must be visible immediately in one of the following ways:

- Server side Configure the cache to expire immediately after access.
- Client side Configure the `maxage` attribute to a value of `0` (zero).

Static Resource That Might Change (Not Visible Immediately)

Determine the maximum acceptable delay before a resource update is detected, to ensure that a conditional GET request is sent if the cache is older than the amount of time that is configured in the MAX_DELAY attribute.

Apply cache control settings for static resources that might change and that are not visible immediately in one of the following ways:

- Server side Configure the cache to expire MAX_DELAY after access.

- Client side Configure the `maxage` attribute to the value of the MAX_DELAY attribute. Alternatively, if the HTTP response already specifies an Expires or `Cache-Control: max-age` header value that cannot be changed on the server, configure the `maxage` attribute with a value of `0` and the `maxstale` attribute with the same value as the MAX_DELAY attribute.

Considerations and Usage Notes

Consider also, the following usage notes:

Synchronizing the Clocks

- To function properly, the HTTP 1.1 caching algorithm requires that the system clocks on the server, proxy, and client be synchronized. If the clocks are not synchronized, the cache entry might expire sooner than expected.

Gathering Statistics

- After the cache control settings are implemented, there are several ways to gather statistics for tuning and monitoring:
  - The Fetch dashboard in Genesys Administrator displays near real-time Media Control Platform and Call Control Platform fetching statistics. For more information about this dashboard, see the GVP 8.5 User's Guide.
  - The fetch_end metrics log from the Media Control Platform and the fetch_resp metrics log from the Call Control Platform contain information about cache hits and misses, as well as other data. For information about the metrics logs, see the Genesys Voice Platform 8.5 Metrics Reference and the Genesys Voice Platform 8.5 CCXML Reference.