WIS2 Global Services testing

World Meteorological Organization

Date: 2024-05-24

Version: 2024-03-28

Document location: https://community.wmo.int/wis2-global-services-testing

Standing Committee on Information Management and Technology (SC-IMT)[1]

Commission for Observation, Infrastructure and Information Systems (INFCOM)[2]

Table of Contents

Chapter 1. Abstract

The subject of this Report is the results of testing and experimentation of WIS2 Global Services during the pre-operational phase of WIS2. Global Services testing is coordinated by the WIS2 Architecture and Transition team and provides results and recommendations on testing performance, availability and functionality.

[1] https://community.wmo.int/governance/commission-membership/commission-observation-infrastructures-and-information-systems-infcom/commission-infrastructure-officers/infcom-management-group/standing-committee-information-management-and-technology-sc-imt

[2] https://community.wmo.int/governance/commission-membership/infcom

Chapter 2. Executive Summary

TODO

Chapter 3. Scope

This report presents the testing framework put forth as part of the pre-operational phase of Global Services testing. This report also discussses the results and presents a set of conclusions and recommendations.

Chapter 4. Terms and definitions

This document uses the terms defined in OGC Policy Directive 49, which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this Standard and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the 'ModSpec'. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

4.1. Abbreviated terms

API

Application Programming Interface

GB

Global Broker

GC

Global Cache

GDC

Global Discovery Catalogue

GISC

Global Information System Centre GM: Global Monitor

HTTP

Hypertext Transfer Protocol

HTTPS

Hypertext Transfer Protocol Secure

JSON

JavaScript Object Notation

OGC

Open Geospatial Consortium

MQTT

Message Queuing Telemetry Transport

WCMP2

WMO Core Metadata Profile 2

WIS	WMO Information System
WMO	World Meteorological Organization
WNM	WIS2 Notification Message
WTH	WIS2 Topic Hierarchy

Chapter 5. References

- WMO: WMO Core Metadata Profile (2024) [1]
- WMO: WIS2 Notification Message (2024) [2]
- WMO: WIS2 Topic Hierarchy (2024) [3]
- Draft guidance on technical specifications of WIS2 (2024) [4]
- Draft guidance on transition from GTS to WIS2 (2024) [5]

[1] https://wmo-im.github.io/wcmp2

[2] https://wmo-im.github.io/wis2-notification-message

[3] https://wmo-im.github.io/wis2-topic-hierarchy

[4] https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html

[5] https://wmo-im.github.io/wis2-transition-guide/transition-guide/wis2-transition-guide-DRAFT.html

Chapter 6. High Level Architecture

The focus of testing is to evaluate functionality to ensure all WIS2 components perform as defined by the architecture. Testing is designed to enable core workflows:

- WIS2 Nodes providing data and metadata
- WIS2 Global Brokers subscribing to WIS2 Nodes
- WIS2 Global Caches providing data and metadata for core data and all metadata
- WIS2 Global Discovery Catalogues providing a search API for published discovery metadata
- WIS2 Global Monitors scraping metrics from WIS2 Global Services, and providing metrics/insights on WIS2 performance

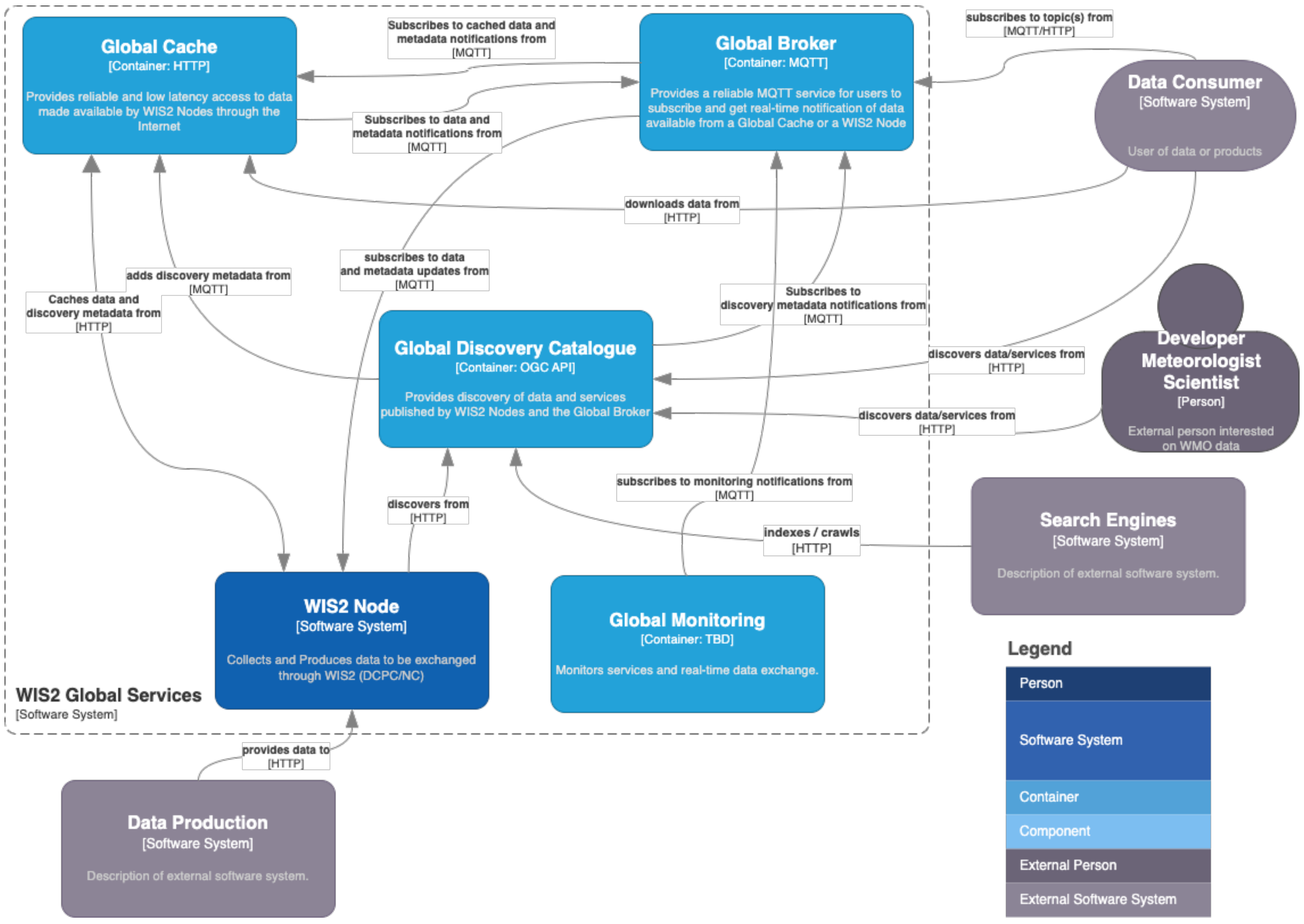
Global Cache
[Container: HTTP]
Provides reliable and low latency access to data made available by WIS2 Nodes through the Internet
Subscribes to cached data and metadata notifications from [MQTT]
Global Broker
[Container: MQTT]
Provides a reliable MQTT service for users to subscribe and get real-time notification of data available from a Global Cache or a WIS2 Node
subscribes to topic(s) from [MQTT/HTTP]
Data Consumer
[Software System]
User of data or products
Subscribes to data and metadata notifications from [MQTT]
downloads data from [HTTP]
subscribes to data and metadata updates from [MQTT]
adds discovery metadata from [MQTT]
Caches data and discovery metadata from [HTTP]
Subscribes to discovery metadata notifications from [MQTT]
Global Discovery Catalogue
[Container: OGC API]
Provides discovery of data and services published by WIS2 Nodes and the Global Broker
discovers data/services from [HTTP]
Developer Meteorologist Scientist
[Person]
External person interested on WMO data
discovers data/services from [HTTP]
subscribes to monitoring notifications from [MQTT]
discovers from [HTTP]
indexes / crawls [HTTP]
Search Engines
[Software System]
Description of external software system.
WIS2 Node
[Software System]
Collects and Produces data to be exchanged through WIS2 (DCPC/NC)
Global Monitoring
[Container: TBD]
Monitors services and real-time data exchange.
WIS2 Global Services
[Software System]
provides data to [HTTP]
Data Production
[Software System]
Description of external software system.
Legend
Person
Software System
Container
Component
External Person
External Software System

Figure 1. High Level Overview of the WIS2 Architecture

The rest of this section describes the components deployed and standards implemented as part of WIS2.

6.1. WIS2 Specifications

6.1.1. WIS2 Topic Hierarchy (WTH)

WTH defines the structure of the WIS Topic Hierarchy. Topics are utilized by WIS Nodes, Global Broker services, and data/metadata subscribers.

6.1.2. WIS2 Notification Message (WNM)

WNM defines the content, structure, and encoding for the WIS2 Notification Message Encoding. WNMs are provided as MQP payloads by WIS2 nodes, Global Broker services, as well as Replay API services (optional OGC API - Features services for data notifications).

6.1.3. WMO Core Metadata Profile (WCMP2)

WCMP2 defines the content, structure, and encoding for WMO resources. WMO resources include, but are not limited to, data (NWP models, observations, forecasts and warnings, etc.), services/APIs, and processes.

6.2. WIS2 Components

6.2.1. Global Broker

WIS2 incorporates several Global Brokers, ensuring highly resilient distribution of notification messages across the globe.

6.2.2. Global Cache

A Global Cache provides a highly available data server from which a Data Consumer can download Core data, as specified in the WMO Unified Data Policy, Resolution 1 (Cg-Ext(2021)).

6.2.3. Global Discovery Catalogue

A Global Discovery Catalogue enables a data consumer to search and browse descriptions of data published by each WIS2 Node. The data description (i.e., discovery metadata) provides sufficient information to determine the usefulness of data and how one may access it.

6.2.4. Global Monitor

A Global Monitor tracks what data is published by WIS2 Nodes, whether data can be effectively accessed by Data Consumers, and the performance of components in the WIS2 system.

6.3. Testing framework

6.3.1. Data

TODO

6.3.2. Environment

The WIS2 development environment will be used as the target network for executing tests.

6.3.3. Performance testing

Ensure WIS2 Global Services are able to operate under various loads.

6.3.4. Functional testing

Ensure WIS2 Global Services operate with one another as expected and meet requirements.

Chapter 7. Tests

7.1. Global Broker Service testing

7.1.1. Functional tests

7.1.1.1. Functional test title

Purpose

TODO

Steps

1. step 1
2. step 2

7.1.2. Performance tests

7.1.2.1. Performance test title

Purpose

TODO

Steps

1. step 1
2. step 2

7.2. Global Cache Service testing

7.2.1. Functional Tests

7.2.1.1. MQTT Connectivity

Purpose

An MQTT client must be able to connect to the local broker of the Global Cache on port 8883 using the MQTT protocol version 5 with TLS (i.e., mqtts protocol) and username/password authentication. **Source:** Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.2: A Global Cache shall operate a message broker.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.

2. Attempt to connect the MQTT client to the local broker of the Global Cache on port 8883.
3. Check if the connection is successful. If the connection is successful, the test passes. If the connection is not successful, the test fails.
4. Close the connection to the broker after the test.

7.2.1.2. Client subscription

Purpose

A Global Cache must allow connected MQTT clients to subscribe to the cache/a/wis2/# topic.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the local broker of the Global Cache.
3. Once the connection is successful, attempt to subscribe to the `cache/a/wis2/#` topic.
4. Check if the subscription is successful. If the subscription is successful, the test passes. If the subscription is not successful, the test fails.
5. Close the connection to the broker after the test.

7.2.1.3. Global Broker subscription

Purpose

A Global Cache must connect to at least 2 Global Brokers using the MQTT protocol with TLS and username/password authentication (everyone/everyone) and subscribe to the following topics:

- `origin/a/wis2/+/metadata/#`
- `origin/a/wis2/+/data/core/#`
- `cache/a/wis2/+/metadata/#`
- `cache/a/wis2/+/data/core/#`

Steps

1. Initialize the GC with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication for the connection to both GB as well as the subscription information to the listed topics. The specific means to achieve this depends on the implementation of the GC
2. Connect the GC as described above to the Global Brokers.
3. Verify that the connection is established by either the Log Files of the GC, the Log Files of the GB or by inspecting network traffic whichever seems fit.
4. Check if the subscription is successful. Do this by
   a. For each listed topic and each listed GB publish one matching WNM
   b. Verify if each WNM is received by the GC. Do this by either inspectng the Log Files of the GC

or by verification if the GC is trying to open a connection to the URL listed as canonical in the WNM.

5. If all messages are received, the test passes. If not, the test fails.
6. Close the connections to the brokers after the test.

Source: Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.3: A Global Cache shall subscribe to notifications about the availability of discovery metadata records and core data for real-time or near-real-time exchange. Duplicate notifications are discarded; clause 4.5.3: A Global Cache shall subscribe to at least one Global Broker for notifications concerning core data and discovery metadata […].

Source: Guide to WIS (WMO No. 1061), Vol II, clause 2.7.1. Procedure for registration of a new Global Service: “Each Global Cache connected to at least two (2) Global Broker and should be able to download the data from all WIS2 Nodes providing Core data” https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_procedure_for_registration_of_a_new_global_service

7.2.1.4. Notification processing

Purpose

The intention here is to test that the GC functions as expected under normal conditions. The Global Cache should be able to process an incoming notification message, download the data at the canonical link, validate the integrity of the message, publish a new notification message on the proper topic using the proper message structure, and update the necessary metrics. This case/test assumes the data object and included checksum is valid, a separate test is included to test the integrity check.

The cache notification message published by the Global Cache should be identical to the notification message received, except for the following:

- the canonical link (a link object’s `href` property where `rel=canonical`)
- the unique identifier of the message (id)
- the topic (`cache/a/wis2/…`) (note the incoming message may be on this same topic if it is from another GC)

The notification messages will follow the standard structure (see Manual on WIS (WMO-No. 1060), Volume II, Appendix E: WIS2 Notification Message).

The notification messages will be published on the standard topic structure in their local message brokers (see Manual on WIS (WMO-No. 1060), Volume II, Appendix D: WIS2 Topic Hierarchy) on topic `cache/a/wis2/…`.

On successful completion, the following metrics should be modified:

- `wmo_wis2_gc_download_total` should be incremented by 1 (one).
- `wmo_wis2_gc_dataserver_status_flag` for the source data server (i.e., the centre ID from where the data item was downloaded from) should be set to 1 (one).
- `wmo-wis2_gc_dataserver_last_download_timestamp_seconds` for the source data server should be

set to the current time.

Steps

These tests rely on mock data and/or curated batches of real data. Specifically notification messages, with accompanying data objects, whose characteristics are known and can be used to validate the GC's behavior. Upon processing the batch of notification messages, the following will be evaluated and compared against the expected results:

- The total number of cache notification messages published by the GC
- The total number of data objects cached by the GC
- The validity of the notification messages published by the GC
- The validity of the data objects cached by the GC (i.e., the data object is accessible via the canonical link)
- The following metrics are updated (or not) as expected per the curated test data set:
  - `wmo_wis2_gc_download_total`
  - `wmo_wis2_gc_dataserver_status_flag`
  - `wmo_wis2_gc_dataserver_last_download_timestamp_seconds`

Source: Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.4: Based on the notifications it receives, a Global Cache shall download and store a copy of discovery metadata records and core data from [WIS2 Nodes] and other Global [Services]; clause 3.7.5.7: A Global Cache shall publish notifications via its Message Broker about copies of the discovery metadata records and core data it makes available. A Global Cache shall use a standardized topic structure when publishing notifications; clause 4.5.2: A Global Cache shall download core data and discovery metadata from [WIS2 Nodes] and other Global [Services] to provide for reliable, low-latency access to those resources via WIS; clause 4.5.4: Based on received notifications, a Global Cache shall download core data from [WIS2 Nodes] or other Global [Services] and store them for a minimum duration of 24 hours; clause 4.5.5: Based on its received notifications, a Global Cache shall download discovery metadata records from [WIS2 Nodes] or other Global [Services] and store them for a minimum duration of 24 hours; clause 4.5.7: A Global Cache shall publish notifications to a Message Broker indicating the availability of data and discovery metadata resources from the Global Cache and shall use the format and protocol specified […].

Source: Guide to WIS (WMO No. 1061), Vol II, clause 2.7.4.1. [Global Cache] Technical considerations https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_technical_considerations_2; clause 2.7.4.2. [Global Cache] Practices and procedures https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_practices_and_procedures_2

7.2.1.5. Cache false directive

Purpose

Where a Global Cache receives a notification message with *properties.cache* set to false, the Global Cache should publish a notification message where the data download link (a link object's `href` property where `rel=canonical`) refers to the source data server.

The cache notification message published by the Global Cache should be identical to the notification message received, except for the following:

- the unique identifier of the message (id)
- the topic (`cache/a/wis2/…`) (note the incoming message may be on this same topic if it is from another GC)

```
Is the above assessment correct?
```

7.2.1.6. Steps

These tests rely on mock data and/or curated batches of real data. Specifically notification messages, with accompanying data objects, whose characteristics are known and can be used to validate the GC's behavior. In this case a known number of messages will have the cache directive set to false. Upon processing the batch of notification messages, the following will be evaluated and compared against the expected results:

- The total number of cache notification messages published by the GC
- The total number of data objects cached by the GC
- The validity of the notification messages published by the GC, taking into consideration the bullets above (difference between the original and the published message).
- The following metrics are updated (or not) as expected per the curated test data set:
    - `wmo_wis2_gc_download_total`
    - `wmo_wis2_gc_dataserver_status_flag`
    - `wmo_wis2_gc_dataserver_last_download_timestamp_seconds`
    - `wmo_wis2_gc_no_cache_total` (incremented by 1 for each notification message where the cache directive is set to false)

```
wmo_wis2_gc_no_cache_total is a proposed new metric
```

7.2.1.7. Source download failure

Purpose

Where a Global Cache is unable to download a data item from the location specified in a notification message (i.e., the source data server), the `metric wmo_wis2_gc_dataserver_status_flag` for the source data server should be set to 0 (zero).

7.2.1.8. Steps

1. step 1
2. step 2

7.2.1.9. Cache override

Purpose

Where a Global Cache determines that it is unable to cache a data item, the Global Cache should publish a notification message where the data download link (a link object's `href` property where `rel=canonical`) refers to the source data server, and the metric `wmo_wis2_gc_cache_override_total` is incremented by 1 (one).

More details needed about the notification message; format, content, topic.

7.2.1.10. Steps

1. step 1
2. step 2

7.2.1.11. Data integrity failure check

Purpose

Where a notification message provides an integrity value for a data item (`properties.integrity`), a Global Cache should validate the integrity of the resources it caches and only accept data which matches. A Global Cache should calculate the hash of the data object instance [once downloaded into the cache?] using the method specified in `properties.integrity.method`. Where the calculated hash does not match the value specified in `properties.integrity.value`: The data item should be removed from the cache if already downloaded No notification message should be published The `metric wmo_wis2_gc_download_errors_total` should be incremented by 1 (one). The `metric wmo_wis2_gc_integrity_failed_total` should be incremented by 1 (one).

7.2.1.12. Steps

1. step 1
2. step 2

7.2.1.13. Duplicate notification discarding

Purpose

A Global Cache must ensure that only one instance of a notification message with a given unique identifier (id) is successfully processed.

Test this by sending two identical notification messages, ideally from different sources, and verify that the second notification message is discarded.

Source: Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.3: A Global Cache shall subscribe to notifications about the availability of discovery metadata records and core data for real-time or near-real-time exchange. Duplicate notifications are discarded.

7.2.1.14. Steps

1. step 1
2. step 2

7.2.1.15. Duplicate notification discarding (alternative)

Purpose

Where a Global Cache fails to process a notification message with a given unique identifier (id), a Global Cache must attempt to process subsequently received notification messages with the same unique identifier.

Test this by sending two almost identical notification messages, the first of which should include an unresolvable data download link (a link object's `href` property where `rel=canonical`) (or simply missing a `canonical` link object?). This will force processing of the first message to fail. The second notification message should be processed successfully, with the data item being copied into the cache.

7.2.1.16. Steps

1. step 1
2. step 2

7.2.1.17. Duplicate data discarding

Purpose

A Global Cache must ensure that only one instance of a data item, designated with a given unique identifier (`properties.data_id`) and publication time (`properties.pubtime`) in the associated notification message, is successfully processed.

Test this by sending two notification messages each with a unique identifier (id) but both with the same data identifier (`properties.data_id`) and publication time (`properties.pubtime`). Ideally the notification messages should simulate data being made available at different locations (i.e., an origin WIS2 Node and another Global Cache) with differing data download links (a link object's `href` property where `rel=canonical`).

7.2.1.18. Steps

1. step 1
2. step 2

7.2.1.19. Duplicate data discarding (alternative 1)

Purpose

Where a Global Cache fails to process a notification message relating to a given unique data object (`properties.data_id` + `properties.pubtime`), a Global Cache must attempt to process subsequently received notification messages with the same unique data identifier.

Test this by sending two notification messages each with a unique identifier (id) but both with the same data identifier (`properties.data_id`). The first message should include an unresolvable data download link (a link object's `href` property where `rel=canonical`) (or simply missing a `canonical` link object?). This will force processing of the first message to fail. The second notification message should be processed successfully, with the data item being copied into the cache.

7.2.1.20. Steps

1. step 1
2. step 2

7.2.1.21. Duplicate data discarding (alternative 2)

Purpose

A Global Cache should treat notification messages with the same data item identifier (`properties.data_id`), but different publication times (`properties.pubtime`) as unique data items. Data items with the same `properties.data_id` but a later publication time should be copied into the cache (see test Notification processing). Data items with the same `properties.data_id` but earlier or identical publication times should be ignored (see test Duplicate link discarding).

Source: Guide to WIS (WMO No. 1061), Vol II, clause 2.7.4.2. [Global Cache] Practices and procedures: "Verify if the message points to new or updated data by comparing the pubtime value of the notification message with the list of data_ids". https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_practices_and_procedures_2

7.2.1.22. Steps

1. step 1
2. step 2

7.2.1.23. Client data download

Purpose

An HTTP client (i.e., a Web browser) must be able to connect to the HTTP server of the Global Cache on port 443 using HTTP 1.1 with TLS but without any authentication and be able to resolve the URL provided in a data download link (a link object's `href` property where `rel=canonical`) from a notification message published by the Global Cache within the previous 24 hours; i.e., download a cached data item.

Note: testing provision of access via HTTP 1.1 - "at least one of the protocols".

Source: Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.5: A Global Cache shall provide highly available access to copies of discovery metadata records and core data it stores; clause 3.7.5.6: A Global Cache shall retain a copy of the discovery metadata records and core data it stores for a duration compatible with the real-time or near-real-time schedule of the data and not less than 24 hours; clause 4.5.2: A Global Cache shall download core data and discovery metadata from [WIS2 Nodes] and other Global [Services] to provide for reliable, low-latency access to those resources via

WIS; clause 4.5.6: Data and discovery metadata available for download from a Global Cache shall be accessible via a URL using at least one of the protocols specified […].

7.2.1.24. Steps

1. step 1
2. step 2

7.2.1.25. Certificate validation

Purpose

A Global Cache must use a valid certificate.

7.2.1.26. Steps

1. step 1
2. step 2

7.2.1.27. Metric publication

Purpose

A Global Cache must publish the following metrics using the OpenMetrics standard:

- `wmo_wis2_gc_download_total`
- `wmo_wis2_gc_download_errors_total`
- `wmo_wis2_gc_dataserver_status_flag`
- `wmo_wis2_gc_dataserver_last_download_timestamp_seconds`
- `wmo_wis2_gc_cache_override_total`
- `wmo_wis2_gc_integrity_failed_total`

Source: https://github.com/wmo-im/wis2-metric-hierarchy/blob/main/metrics/gc.csv

7.2.1.28. Steps

1. step 1
2. step 2

7.2.2. Performance tests

7.2.2.1. Notification processing rate

Purpose

A Global Cache shall be able to successfully process 1000 notification messages, averaging xxx bytes, including caching the associated data item and publishing the new notification message, within xxx seconds.

7.2.2.2. Steps

1. step 1
2. step 2

7.2.2.3. Notification processing time

Purpose

A Global Cache shall successfully process a notification message, including caching the associated data item and publishing the new notification message, within xxx seconds.

Note: A Global Cache may decide to ignore the request to cache a data item if it will take excessively long to process. See test Cache override for details.

7.2.2.4. Steps

1. step 1
2. step 2

7.2.2.5. Concurrent client downloads

Purpose

1000 HTTP clients concurrently download data items from the Global Cache, with HTTP response time not exceeding xxx seconds, at a rate exceeding xxx bytes/second.

Source: Manual on WIS (WMO No. 1060), Vol II, clause 3.7.5.5: A Global Cache shall provide highly available access to copies of discovery metadata records and core data it stores; clause 4.5.1: A Global Cache shall operate a highly available storage and download service; clause 4.5.2: A Global Cache shall download core data and discovery metadata from [WIS2 Nodes] and other Global [Services] to provide for reliable, low-latency access to those resources via WIS. **Source:** Guide to WIS (WMO No. 1061), Vol II, clause 2.7.2.2. Service levels, performance indicators and fair-usage policies: “A Global Cache should support a minimum of 1000 simultaneous downloads” https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_procedure_for_registration_of_a_new_global_service

7.2.2.6. Steps

1. step 1
2. step 2

7.2.2.7. Storage volume

Type of test

Performance

Purpose

A Global Cache shall be able to store at least 100GB of Core data items.

Source: Guide to WIS (WMO No. 1061), Vol II, clause 2.7.2.2. Service levels, performance indicators and fair-usage policies: “A Global Cache should support a minimum of 100 GB of data in the cache” https://wmo-im.github.io/wis2-guide/guide/wis2-guide-DRAFT.html#_procedure_for_registration_of_a_new_global_service

7.2.2.8. Steps

1. step 1
2. step 2

7.2.3. System-wide tests

7.2.3.1. Single Global Broker failure

Purpose

Pre: At least 2 Global Brokers have subscribed to notification messages from a given WIS2 Node. Pre: Global Cache is subscribed to at least two Global Brokers. Pre: Global Cache is successfully downloading data items into its cache from the WIS2 Node.

In the event that one of the Global Brokers subscribing to the WIS2 Node fails (i.e., goes offline), notification messages from the WIS2 Node are still received (and processed) by the Global Cache.

7.2.3.2. Steps

1. step 1
2. step 2

7.2.3.3. Origin node unresolvable

Purpose

Pre: A given WIS2 Node is publishing notification messages and Core data. Pre: At least 2 Global Caches are receiving notification messages from the WIS2 Node (via a Global Broker). Pre: Global Cache #1 is able to resolve HTTP URLs from the WIS2 Node. Pre: Global Cache #2 is not able to resolve HTTP URLs from the WIS2 Node.

Core data items published by the WIS2 Node are successfully cached by Global Cache #2, by way of downloading from Global Cache #1.

7.2.3.4. Steps

1. step 1
2. step 2

7.2.4. Considerations

7.2.4.1. General Testing Strategy

The testing strategy for the Global Cache (GC) will leverage both mocked data and curated real data. This approach ensures a comprehensive evaluation of the GC’s functionality under various scenarios.

1. **Mocked Data:** This data is artificially created to simulate specific scenarios that might not be easily reproducible with real data. It allows us to test edge cases, error conditions, and unusual data patterns.
2. **Curated Real Data:** This data is derived from actual use cases and provides a realistic representation of what the GC will encounter in a production environment. It allows us to test the GC’s performance and reliability.

The testing process will be automated through scripts. These scripts will perform the following steps:

1. **Data Publication:** The scripts will publish a batch of messages to the dev MQTT broker. These messages will represent a mix of scenarios based on the mocked and curated real data.
2. **GC Subscription:** The GC will be subscribed to the MQTT broker to receive the published messages. This simulates the GC’s real-world operation where it subscribes to Global Brokers to receive notifications. (Remy has something already in the works here)
3. **Result Validation:** After the GC processes the received messages, the scripts will validate the results. This includes checking if the GC correctly stored the data, published notifications, and updated metrics as expected.

7.2.4.2. General Performance Testing Strategy

The performance testing strategy for the GC will primarily focus on the time taken from when a notification message is published to when the associated cache message is received by the test process. This approach ensures a comprehensive evaluation of the GC’s performance under various scenarios.

1. **Notification Publication:** The test process will publish a notification message to the MQTT broker. This message will represent a specific scenario based on the mocked or curated real data.
2. **Start Timer:** The test process will start a timer immediately after the notification message is published. Multiple timers can be used for multiple notification messages.
3. **GC Subscription and Processing:** The GC, which is subscribed to the dev MQTT broker, will receive the published notification message. It will then process the message, which may include storing the data, publishing a cache notification, and updating metrics as expected.
4. **Cache Message Receipt:** The test process, which is also subscribed to the MQTT broker, will receive the cache message published by the GC.
5. **Stop Timer:** The test process will stop the timer immediately after the cache message is received.

6. **Result Validation:** The test process will validate the results. This includes checking if the GC correctly processed the notification message and published the cache message, and if the time taken (as measured by the timer) is within the acceptable performance limits.
7. **Data Size Consideration:** The size of the cached data objects will also be considered. The performance of the GC can be evaluated based on the bytes per second processed. This will help in understanding the GC's efficiency in handling different sizes of data objects.

7.2.4.3. Addition of `wmo_wis2_gc_no_cache_total` metric

- This metric will be used to capture `properties.cache=false` cases. It will be incremented by 1 (one) for each notification message where the `properties.cache` property is set to `false` or where the Global Cache determines that it is unable to cache a data item.

7.2.4.4. Message uniqueness = `properties.data_id` + `properties.pubtime`

- The unique identifier of a data item is a combination of the data identifier (`properties.data_id`) and the publication time (`properties.pubtime`). This is to ensure that the Global Cache does not store multiple copies of the same data item AND to support the ability to update/correct data items.
- Are other folks in agreement with this approach and already implementing it?

7.2.4.5. Max data object size

- What is the maximum size of a data object that a Global Cache should be able to process and store?

7.2.4.6. Data Integrity Checks

- How are folks implementing the data integrity check? Downloading first or any other approach, perhaps a rolling hash?

7.2.4.7. Best practices/best effort

Retry/Redrive strategy

- Simple: failed download attempts where we retry same URL. (immediate, and/or after a backoff as these solve different problems).
- Redrive based on messages with redundant `properties.data_id�s in the event of a download failure. This would require caching all messages for a certain amount of time. This way the Global Cache can reprocess the message with the same `properties.data_id` + `properties.pubtime` if the download fails and 'redundant' messages with different download links exist.
    - supporting update/correction of data items per GTS?

7.3. Global Discovery Catalogue Service testing

7.3.1. Test setup

The GDC test setup consists of the following:

- A test data bundle to consist of:
  - 100 valid WCMP2 records
  - 1 broken JSON WCMP2 record
  - 1 invalid WCMP2 record
  - 102 WNM documents, each of which pointing to the related WCMP2 records
  - 1 WNM document specifying a WCMP2 record deletion
- all WCMP2 records stored on an HTTP server
- all WNM documents updated to point to the correct HTTP server for proper HTTP dereferencing

7.3.2. Functional tests

7.3.2.1. Global Broker connection and subscription

Purpose

A Global Discovery Catalogue must connect to a Global Broker using the MQTT protocol with TLS and username/password authentication (everyone/everyone) and subscribe to the following topic:

- `origin/a/wis2/+/metadata/#`

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. Close the connections to the broker after the test.

On successful completion, the following metrics should be modified:

- `wmo_wis2_gdc_connected_flag` for the centre-id from which the Global Discovery Catalogue connected to should be set to 1 (one).

7.3.2.2. Notification and metadata processing (success)

Purpose

The Global Discovery Catalogue should be able to process valid WCMP2 metadata record of core data published by a WIS2 Node.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
    - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
    a. find the canonical link object in the `links` array.
    b. from the matching link, issue a HTTP GET request against the matching `href` value.
    c. parse the HTTP response:
        i. validate against the WCMP2 Executable Test Suite (ETS)
        ii. publish the WCMP2 record to the catalogue
        iii. publish an ETS validation report as a notification message to the GDC MQTT broker.
    d. using the WCMP2 identifier (i.e. `$WCMP2_ID`), construct a path to the record on the GDC (`https://HOST/collections/wis2-discovery-metadata/$WCMP2_ID`).

On successful completion:

- the resulting WCMP2 record should be available on the GDC API and contain an MQTT link / channel (using `cache/a/wis2`) foreach Global Broker
- the following metrics should be modified:
    - `wmo_wis2_gdc_passed_total` for the centre-id from where the metadata was published from should be incremented by 1 (one).
    - `wmo_wis2_gdc_core_total` for the centre-id from where the metadata (core data policy) was published from should be incremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

7.3.2.3. Notification and metadata processing (failure; record not found)

Purpose

The Global Discovery Catalogue should be able to process failing (record not found) WCMP2 metadata published by a WIS2 Node.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.

3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
   a. find the canonical link object in the `links` array.
   b. from the matching link, issue a HTTP GET request against the matching `href` value.
   c. if the response is an HTTP status code of 404:
      i. publish an ETS error report as a notification message to the GDC MQTT broker.

On successful completion:

- the following metrics should be modified:
  - `wmo_wis2_gdc_failed_total` for the centre-id from where the metadata was published from should be incremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

7.3.2.4. Notification and metadata processing (failure; malformed JSON)

Purpose

The Global Discovery Catalogue should be able to process failing (malformed JSON) WCMP2 metadata published by a WIS2 Node.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
   a. find the canonical link object in the `links` array.
   b. from the matching link, issue a HTTP GET request against the matching `href` value.
   c. parse the HTTP response:
   d. if the JSON is malformed:
      i. publish an ETS error report as a notification message to the GDC MQTT broker.

On successful completion:

- the following metrics should be modified:
    - `wmo_wis2_gdc_failed_total` for the centre-id from where the metadata was published from should be incremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

7.3.2.5. Notification and metadata processing (failure; invalid WCMP2 record)

TODO: we need more examples of invalid WCMP2

Purpose

The Global Discovery Catalogue should be able to process failing WCMP2 metadata published by a WIS2 Node.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
    - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
    a. find the canonical link object in the `links` array.
    b. from the matching link, issue a HTTP GET request against the matching `href` value.
    c. parse the HTTP response:
        i. validate against the WCMP2 Executable Test Suite (ETS)
        ii. publish an ETS error report as a notification message to the GDC MQTT broker.

On successful completion:

- the following metrics should be modified:
    - `wmo_wis2_gdc_failed_total` for the centre-id from where the metadata was published from should be incremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

7.3.2.6. Metadata ingest centre-id mismatch

TODO: validate workflow for regional distribution of partner metadata

Purpose

A Global Discovery Catalogue should detect a mismatch between an incoming message topic's centre-id and the centre-id as part of a WCMP2 record identifier.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
   a. capture the centre-id from the topic (4th token split on `/`).
   b. find the canonical link object in the `links` array.
   c. from the matching link, issue a HTTP GET request against the matching `href` value.
   d. parse the HTTP response:
   e. extract the centre-id from WCMP2 record identifier (3rd token split on `:`).
6. compare the centre-id from the topic and the centre-id of the WCMP2 record identifier.
7. publish an ETS error report as a notification message to the GDC MQTT broker.

On successful completion, the following metrics should be modified:

- `wmo_wis2_gdc_failed_total` for the centre-id from where the metadata was published from should be incremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

7.3.2.7. Notification and metadata processing (record deletion)

Purpose

The Global Discovery Catalogue should be able to process valid WCMP2 metadata record deletion of core data published by a WIS2 Node.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`

4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On an incoming message:
   a. find the deletion link object in the `links` array.
   b. from the matching link, issue a HTTP GET request against the matching `href` value.
   c. parse the HTTP response:
      i. validate against the WCMP2 Executable Test Suite (ETS)
      ii. delete the WCMP2 record from the catalogue
      iii. publish a notification message to the GDC MQTT broker.
   d. using the WCMP2 identifier (i.e. `$WCMP2_ID`), construct a path to the record on the GDC (`https://HOST/collections/wis2-discovery-metadata/$WCMP2_ID`).

On successful completion:

- the WCMP2 record should be removed from the GDC API
- the following metrics should be modified:
  - `wmo_wis2_gdc_passed_total` for the centre-id from where the metadata was published from should be decremented by 1 (one).
  - `wmo_wis2_gdc_core_total` for the centre-id from where the metadata (core data policy) was published from should be decremented by 1 (one).
- a notification message should arrive from the Global Broker under `monitor/a/wis2/CENTRE_ID_global-discovery-catalogue/centre-id`)

BEGIN TODO test deletion WITHOUT metadata_id END

BEGIN TODO add test case for metadata zipfile usage on cold start END

7.3.2.8. WCMP2 metadata archive zipfile publication

Purpose

Validate that a GDC API publishes a metadata archive zipfile.

Steps

1. Construct a path to the GDC endpoint (`https://HOST/collections/wis2-discovery-metadata`).
2. Issue a HTTP GET request on the path.
3. Parse the HTTP response.
4. Check that the record includes a `links` array.
5. In the `links` array, check that a metadata archive zipfile link is available (where a link object's `rel=archives` and `type=application/zip`).
6. In the matching link, issue a HTTP GET request on the associated `href` value.
7. Unzip the content of the HTTP response.

8. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
9. Connect the MQTT client to the Global Discovery Catalogue.
10. Once the connection is successful, attempt to subscribe to the following topics:
    - `origin/a/wis2/centre-id/metadata/#` (where `centre-id` is the centre identifier of the Global Discovery Catalogue).
11. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
12. If the MQTT client is able to successfully subscribe to the topic on the Global Discovery Catalogue, the test passes. If not, the test fails.
13. On receipt of a notification message, issue a HTTP GET requerst on the canonical link (a link object's `href` property where `rel=canonical`)
14. Unzip the content of the HTTP response.

On successful completion:

- the resulting HTTP response should be a zip encoded data, which, when unzipped, contains a directory of JSON files of WCMP2 metadata.

7.3.2.9. OpenMetrics publication

Purpose

Validate that a GDC API publishes an OpenMetrics endpoint.

Steps

1. Construct a path to the GDC endpoint (`https://HOST/collections/wis2-discovery-metadata`).
2. Issue a HTTP GET request on the path.
3. Parse the HTTP response.
4. Check that the record includes a `links` array.
5. In the `links` array, check that a metadata archive zipfile link is available (where a link object's `rel=related`, `type=text/plain` and `title=OpenMetrics`.
6. In the matching link, issue a HTTP GET request on the associated `href` value.
7. Parse the HTTP response.

On successful completion:

- the resulting HTTP response should be a text file in OpenMetrics format.

7.3.2.10. API functionality

Purpose

Validate that a GDC API performs as expected based on the OGC API - Records standard.

Steps

1. Construct a path to the GDC endpoint (`https://HOST/collections/wis2-discovery-metadata`).
2. Issue a HTTP GET request on the path.
3. Parse the HTTP response.
4. Check that the record includes a `links` array.
5. In the `links` array, check that an items link is available (where a link object's `rel=items` and `type=application/geo+json`).
6. In the matching link, issue a HTTP GET request on the associated `href` value.
7. Parse the HTTP response.
8. Ensure that a `numberMatched` property exists with an integer value greater than 0.
9. Ensure that a `numberReturned` property exists with an integer value greater than 0.
10. Construct a path to the GDC endpoint with a bounding box query parameter (`https://HOST/collections/wis2-discovery-metadata?bbox=-142,42,-53,84`).
11. Issue a HTTP GET request on the path.
12. Parse the HTTP response.
13. Ensure that a `numberMatched` property exists with an integer value greater than 0.
14. Ensure that a `numberReturned` property exists with an integer value greater than 0.
15. Ensure that a `features` array exists.
16. Construct a path to the GDC endpoint with a temporal query parameter (`https://HOST/collections/wis2-discovery-metadata?datetime=2000-11-11T12:42:23Z/..`).
17. Issue a HTTP GET request on the path.
18. Parse the HTTP response.
19. Ensure that a `numberMatched` property exists with an integer value greater than 0.
20. Ensure that a `numberReturned` property exists with an integer value greater than 0.
21. Ensure that a `features` array exists.
22. Construct a path to the GDC endpoint with a full text query parameter (`https://HOST/collections/wis2-discovery-metadata?q=observations`).
23. Issue a HTTP GET request on the path.
24. Parse the HTTP response.
25. Ensure that a `numberMatched` property exists with an integer value greater than 0.
26. Ensure that a `numberReturned` property exists with an integer value greater than 0.
27. Ensure that a `features` array exists.

TODO: measure accuracy

7.3.3. Performance tests

7.3.3.1. Processing timeliness

Purpose

Validate that a GDC is able to process WCMP2 metadata in a timely manner.

Steps

1. Initialize the MQTT client with the necessary parameters such as the MQTT protocol version 5, TLS security, and username/password for authentication.
2. Connect the MQTT client to the Global Broker.
3. Once the connection is successful, attempt to subscribe to the following topics:
   - `origin/a/wis2/+/metadata/#`
4. Check if the subscription is successful. If the subscription is successful, proceed. If the subscription is not successful, the test fails.
5. On all incoming messages:
   a. find the canonical link object in the `links` array.
   b. from the matching link, issue a HTTP GET request against the matching `href` value.
   c. parse the HTTP response:
      i. validate against the WCMP2 Executable Test Suite (ETS)
      ii. publish the WCMP2 record to the catalogue
      iii. publish an ETS validation report as a notification message to the GDC MQTT broker.
   d. using the WCMP2 identifier (i.e. `$WCMP2_ID`), construct a path to the record on the GDC (`https://HOST/collections/wis2-discovery-metadata/$WCMP2_ID`).

On successful completion:

- all WCMP2 records should be processed and published in 5 minutes

Chapter 8. Results

8.1. Global Cache Service results

8.1.1. China

8.1.1.1. Functional test results

8.1.1.2. Performance test results

8.1.1.3. Testing observations

8.1.2. Germany

8.1.2.1. Functional test results

8.1.2.2. Performance test results

8.1.2.3. Testing observations

8.1.3. Japan

8.1.3.1. Functional test results

8.1.3.2. Performance test results

8.1.3.3. Testing observations

8.1.4. Republic of Korea

8.1.4.1. Functional test results

8.1.4.2. Performance test results

8.1.4.3. Testing observations

8.1.5. United Kingdom of Great Britain and Northern Ireland / United States of America

8.1.5.1. Functional test results

8.1.5.2. Performance test results

8.1.5.3. Testing observations

8.2. Global Broker Service results

8.2.1. Brazil

8.2.1.1. Functional test results

8.2.1.2. Performance test results

8.2.1.3. Testing observations

8.2.2. China

8.2.2.1. Functional test results

8.2.2.2. Performance test results

8.2.2.3. Testing observations

8.2.3. France

8.2.3.1. Functional test results

8.2.3.2. Performance test results

8.2.3.3. Testing observations

8.2.4. United States of America

8.2.4.1. Functional test results

8.2.4.2. Performance test results

8.2.4.3. Testing observations

8.3. Global Discovery Catalogue results

8.3.1. Canada

8.3.1.1. Functional test results

8.3.1.2. Performance test results

8.3.1.3. Testing observations

8.3.2. China

8.3.2.1. Functional test results

8.3.2.2. Performance test results

8.3.2.3. Testing observations

8.4. Global Monitor results

8.4.1. China

8.4.1.1. Functional test results

8.4.1.2. Performance test results

8.4.1.3. Testing observations

8.4.2. Morocco

8.4.2.1. Functional test results

8.4.2.2. Performance test results

8.4.2.3. Testing observations

Chapter 9. Discussion

TODO

Chapter 10. Conclusions

TODO

Chapter 11. Future work

TODO

Appendix A: Revision History

Date	Release	Author	Primary clauses modified	Description
2024-03-30	0.1	Kralidis	all	initial version

Bibliography

- ab - Apache HTTP server benchmarking tool (2024) [1]

[1] https://httpd.apache.org/docs/2.4/programs/ab.html