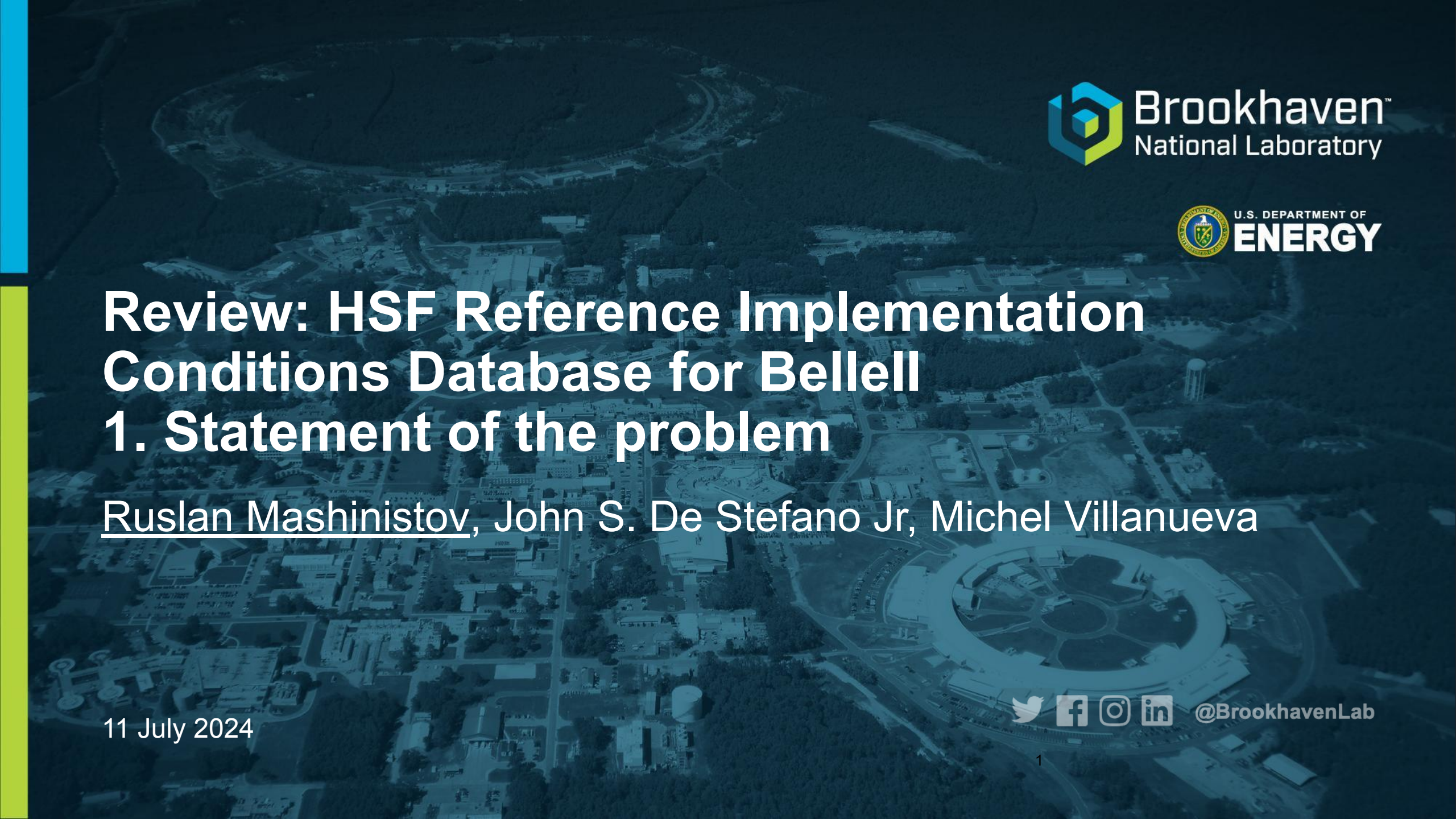


Review: HSF Reference Implementation Conditions Database for BelleII 1. Statement of the problem

Ruslan Mashinistov, John S. De Stefano Jr, Michel Villanueva

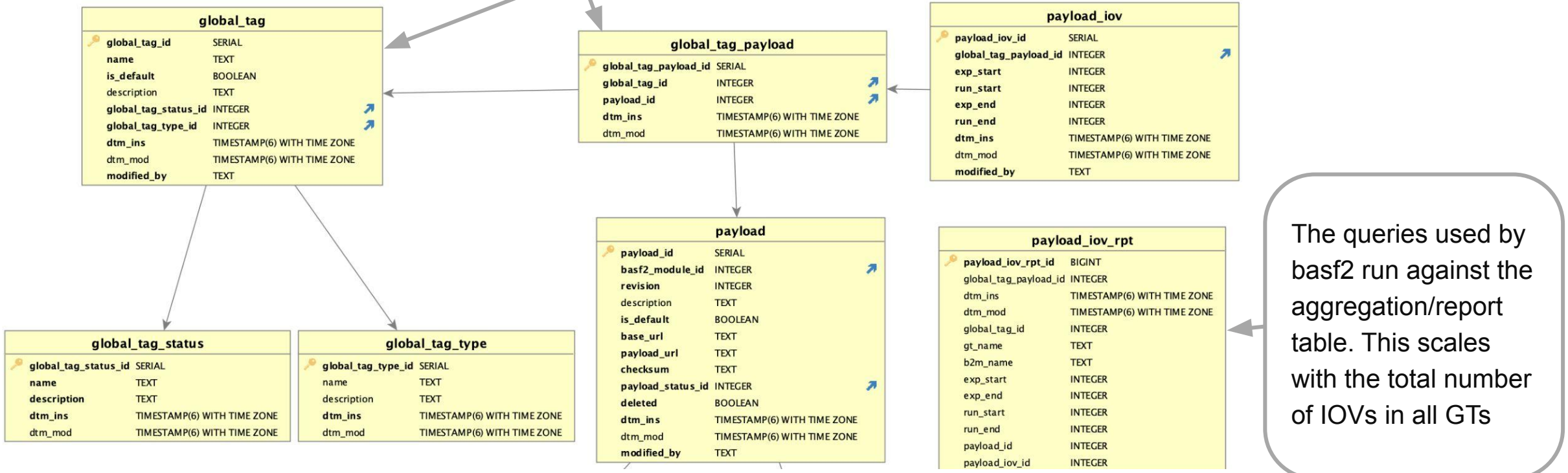
11 July 2024

Belle II CDB Overview

- BNL operates the CDB service since 2018
- During this time, we've made numerous improvements to the server code and infrastructure. Enhancements include:
  - Site Squid caches
  - JWT auth/authz
  - GT state machine (GT statuses and transitions)
  - Bulk PayloadIOVs update API
  - Database cleanup procedures
  - Upgrades of Java, Payara, Kubernetes, PostgreSQL
  - Improve the deployment scripts
  - Read-only GT viewer
  - Grafana monitoring
- Over the past year, we've tackled many issues and challenges, which can be grouped into three categories:
  - Scalability issues with the current DB schema design
  - Service implementation
  - Long-term support model

Design limitations of the database schema

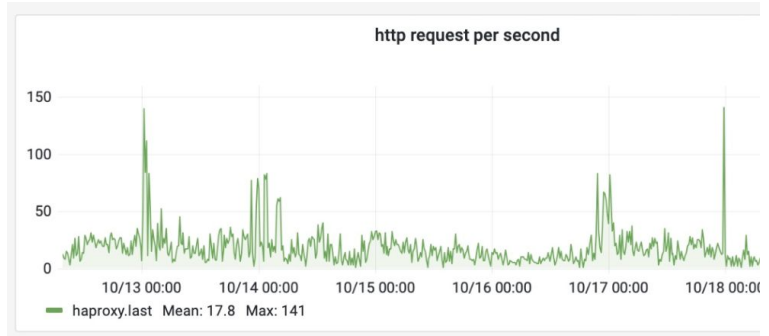
All payloads and IOVs are directed to the GT without intermediate-level table

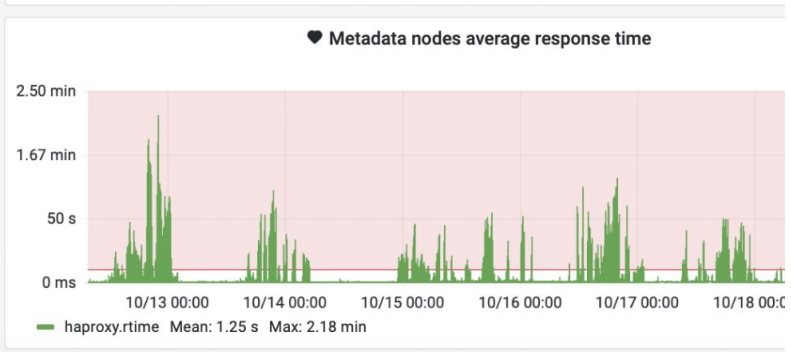


The queries used by basf2 run against the aggregation/report table. This scales with the total number of IOVs in all GTs

Scalability issues. Long response time

- **Long Response Time with High Request Rate**
  - Performance Issues: Response times > 1 minute when handling HTTP requests at a rate of more than 50Hz.
  - Solution: Implemented Squid caching at the largest sites to improve performance and reduce latency



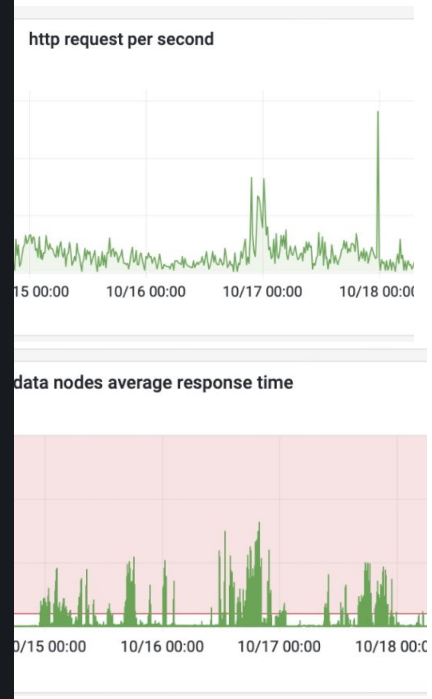


Scalability issues. Long response time

- Long
  - R
  - S

GT Requests by Client Address

Client Location	GT URL	Count
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	23836
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	23722
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	18572
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	15506
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	15486
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	14392
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/mc_production_MC15ri_a	13056
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/mc_production_MC15ri_a	13010
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/Legacy_CollisionAxisCMS	12672
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	12562
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtName=mc_production_MC15ri_a&expNumber=1003&runNumber=0	11234
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtName=mc_production_MC15ri_a&expNumber=1003&runNumber=0	10918
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	10540
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/analysis_tools_light-2406-ragdoll	10330
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	10300
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtName=release-06-00-07&expNumber=1003&runNumber=0	10078



Scalability issues. Long response time

- **Long**
  - R
  - S

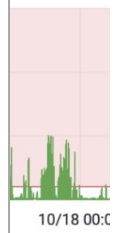
GT Requests by Cl

Client Location	GT URL	
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTagStatus	
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/mc_production_MC15ri_a	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/mc_production_MC15ri_a	
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/Legacy_CollisionAxisCMS	
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/release-06-00-07	
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtName=mc_production_MC15ri_a&expNum	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtName=mc_production_MC15ri_a&expNu=1003&runNumber=0	10918
KEK CC Squid (1)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	
UVic Squid	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/analysis_	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/globalTag/online	
KEK CC Squid (2)	http://belle2db.sdcc.bnl.gov/b2s/rest/v2/iovPayloads/?gtNar	

- Retrieve GT statuses
  - OPEN, TESTING, VALIDATED, PUBLISHED…
  - The list only been changes once or twice for the last 7 years.
- This call used as isAlive check for the CDB service
  - Easy to cache

- Retrieve GT by name
  - includes Payload count

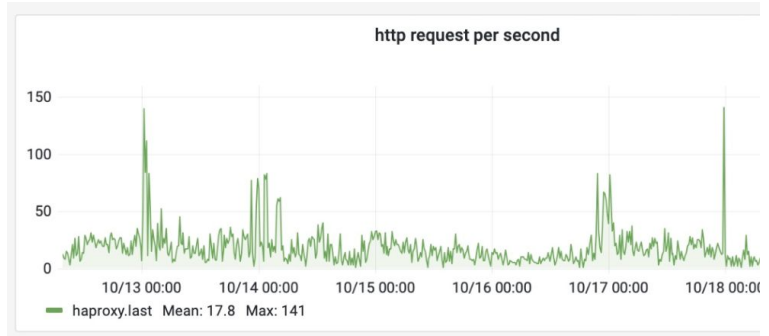
```
select count(gtp)  from GlobalTagPayload gtp
where gtp.globalTag.globalTagId = :globalTagId
```

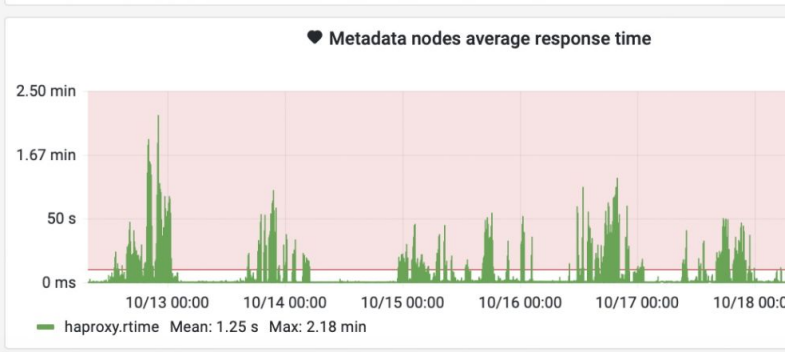


- Retrieve Payload for the given GT and IOVs
  - Request made on report table

Scalability issues. Long response time

- **Long Response Time with High Request Rate**
  - Performance Issues: Response times > 1 minute when handling HTTP requests at a rate of more than 50Hz
  - Solution: Implemented Squid caching at the largest sites to improve performance and reduce latency
- **Explanation of the problem**
  - All the calls from the jobs are read calls
    - Of course, they may interfere with the write activities
    - However, even when considering read calls in isolation, we can explain the scalability issue
  - A certain number of calls are made for particular objects from the dedicated tables.
  - The calls retrieving payload for the given GT and IOVs are made on the report table
    - Then extract Payload object
    - Extract PayloadIOV object
    - Compile result PayloadIOV list





Scalability issues. Cloning huge global tags

- **Cloning Huge Global Tags**
  - Performance Issues: Cloning and managing large Global Tags are slow, timeouts in extreme cases
  - Ineffective "Clone" API: Requires excessive server-side processing time, leading to timeouts

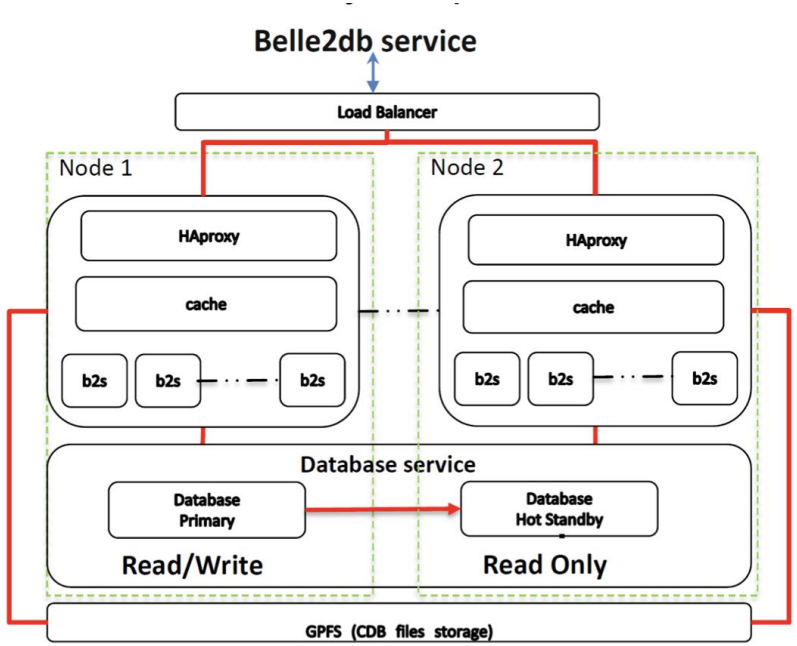
Scalability issues. Cloning huge global tags

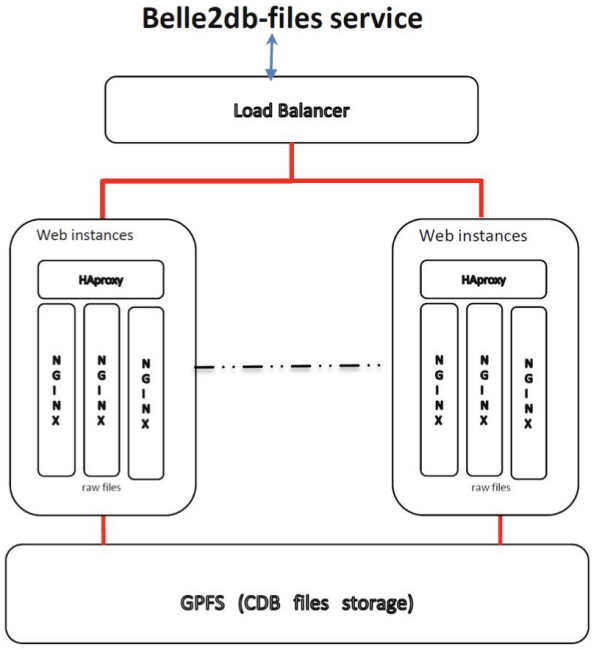
- **Cloning Huge Global Tags**
  - Performance Issues: Cloning and managing large Global Tags are slow, timeouts in extreme cases
  - Ineffective "Clone" API: Requires excessive server-side processing time, leading to timeouts
    - Clone GT object
      - Get GlobalTagPayloads from source
      - Add GlobalTagPayloads to destination
      - Then populate report table
        - Get GlobalTagPayloads
          - Get PayloadIOVs
          - Safe PayloadIOVs to report table

Scalability issues. Cloning huge global tags

- **Cloning Huge Global Tags**
  - Performance Issues: Cloning and managing large Global Tags (~100K Payloads) are slow, timeouts in extreme cases
  - Ineffective "Clone" API: Requires excessive server-side processing time, leading to timeouts
    - Clone GT object
      - Get GlobalTagPayloads from source
      - Add GlobalTagPayloads to destination
      - Then populate report table
        - Get GlobalTagPayloads
          - Get PayloadIOVs
          - Safe PayloadIOVs to report table
  - Client-Side Workaround:
    - Create an empty Global Tag and populate it with PayloadIOVs extracted from the target Global Tag
    - Inefficient and time-consuming due to the need for a new connection for each of the ~100K payloads
    - Results in authentication errors as the JWT expires (default JWT valid for 15 minutes)

Service infrastructure issues





Implementation:

- Docker image
  - Java/Spring Boot application
  - Payara micro server
- Lightweight Kubernetes cluster
  - Puppet deployment scripts for Docker pods (B2S, haproxy, squid, nginx, network routing)

- Problems with Java version upgrade
- Java memory consumption problem
- Problems with Payara version upgrade
- Swagger interface vulnerabilities
- Problems with Kubernetes upgrade

Java memory consumption

- We are consistently facing resource issues with the Java application related to excessive memory allocation and consumption
- Issues are particularly acute during startup and initialization of the Java pods, which take 15-20 minutes per pod, complicating system and service maintenance.

Java Virtual Machine (JVM) Memory Management:

- JVM runs Java applications and manages their memory usage.
- Initial memory allocation reaches upwards of 500GB in total (256GB per node on two nodes).
- Memory requirement scales with the size of the database at an unknown factor close to but not precisely two.
- Uncertainty remains on whether the amount of allocated memory requires adjustment during JVM re-initialization.
- Excessive memory allocation causes service initialization failure, leading to a manual, time-consuming process of testing and tuning.

Partial Fix and Ongoing Issues:

- Partially fixed by disabling data caching during application initialization.
- Despite this fix, the Java application continues to gradually consume a significant amount of memory

Issues with Outdated Versions of Java, Payara, and Kubernetes

- **Java version**
  - **Compilation**: Application is being compiled to be compatible with Java 8 due to the Maven settings
  - **Runtime**: If NetBeans is set to use JDK 20, then your application is running on JDK 20
    - Last Java version is 22
  - Old versions of dependencies
- **Payara-micro version**
  - We're using old version of payara/micro:5.2021.1 (Docker image)
    - Last version is 6.2024.6
  - Expired built-in SSL certificate issue
- **Kubernetes version**
  - **Discontinuation of Library Support**: Google shifted its hosting platform for the Kubernetes infrastructure repository, discontinuing support for the current library versions on which our current deployment relies
- **Swagger version**
  - Swagger interface for the APIs currently redirects to the external site petstore.swagger.io, alongside a JSON file describing the actual APIs. The approach of relying on an external site hosting an outdated version of Swagger has led to vulnerability issues, resulting in the eventual deactivation of the component.

Long-Term Support Model

- A long-term, sustainable support model necessitates changes in the schema design.
- There are critical aspects of the current implementation that we do not fully understand.
- In the HEP community, there is a shortage of Java experts, making the continued operation of the current implementation of this application a potential single point of failure.
- Transitioning to an implementation written in a programming language more universally accepted and known throughout the Belle II computing community, such as Python, could be significantly more beneficial.

Backup

Large global tags

- **Current statistics extraction**
  - There are 153 GTs (out of 2846) with more than 10000 payloads, and 43 GTs (out of 2846) with more than 50000
  - The largest global tag contains 85880 payloads and 163377 payloadIOVs
  - The median size of these global tags is 2394 distinct payloads or 2677 payloadIOVs
  - Currently huge GT are related to the raw reprocessing campaigns
    - Cloning exceeds an hour
- B2BII Global Tags
  - SQLite dumps of the problematic GTs were created and placed at a small number of sites that needed them for Belle experiment (not Belle II) analysis.