|  |
| --- |
| NCBI |
| Pubseq Gateway Server (PSG) |
| Overview and the Protocol Specification |
|  |
| **Sergey Satskiy** |
| **1/24/2018** |

|  |
| --- |
| Document version: 1.3 |

Changes history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | What changes |
| 1.0 | Jan 24, 2018 | Sergey Satskiy | Initial revision |
| 1.1 | Mar 08, 2018 | Sergey Satskiy | Added format #2 for blob retrieval; GetBlob diagram |
| 1.2 | Mar 09, 2018 | Sergey Satskiy | Changes in the resolution request URL format |
| 1.3 | Mar 23, 2018 | Sergey Satskiy | Adding ADMIN/config request |

Table of Contents

[Pubseq Gateway Server (PSG) 3](#_Toc509579364)

[Requirements 3](#_Toc509579365)

[Overview 4](#_Toc509579366)

[Communication Protocol 5](#_Toc509579367)

[Files Architecture 5](#_Toc509579368)

[Requests 5](#_Toc509579369)

[Accession Resolver 5](#_Toc509579370)

[Blob Retrieval 6](#_Toc509579371)

[Configuration 7](#_Toc509579372)

[Cassandra Database 8](#_Toc509579373)

[entity 8](#_Toc509579374)

[largeentity 8](#_Toc509579375)

[Monitoring and Maintenance 9](#_Toc509579376)

[Command Line Arguments 10](#_Toc509579377)

[Configuration Parameters 10](#_Toc509579378)

[[LMDB\_CACHE] section 10](#_Toc509579379)

[[SERVER] section 10](#_Toc509579380)

[[CASSANDRA\_DB] section 11](#_Toc509579381)

[Appendix 13](#_Toc509579382)

[GetBlob Diagram 13](#_Toc509579383)

# Pubseq Gateway Server (PSG)

This document provides an overview of the Pubseq Gateway server functionality and requirements to the various aspects of the server lifecycle. Basically the server provides two types of services:

* accession resolution
* blobs retrieval with its accsessions

Accessions are string identifiers.

# Requirements

Below is a list of major requirements to the Pubseq Gateway server.

* The server must operate as a Linux operating system daemon.
* The server must read all the settings from a configuration file only.
* The server must be able to work in a cloud environment.
* The server must serve many clients simultaneously.
* The logging facilities must be provided via (TBD)
* The server must provide an interface for monitoring.
* The protocol to communicate to the clients is HTTP/2.
* The server must support alerts
* The server must not deal neither with authentication nor with authorization. These features, if necessary, must be implemented outside of the server.
* The server must not keep any information about the clients it serves.

# Overview

Basically, the Pubseq Gateway server is stateless and operates in request – response mode.

The diagram below shows the main actors and entities involved into a typical Pubseq Gateway application.



The clients establish TCP/IP connections using HTTP/2 with the Pubseq Gateway server via an API (psg\_client library), and they send requests over the established connection.

The Cassandra DB stores two major types of objects: resolutions for accessors and BLOBs. So the clients receive the BLOBs through the pubseq server while the resolutions are used by the server internally.

To speed up the resolution process there is a local copy of them stored in an LMDB file. The file is populated by a synchronization utility showen on the diagram in green.

# Communication Protocol

# Files Architecture

The diagram below shows the files used by Pubseq Gateway server.



Pubseq Gateway reads its configuration file (usually named pubseq\_gateway.ini) and configures data access and internal structures correspondingly.

Due to a predicted high load on the server there will be no extensive logging on every event. The only warnings and errors are going to be logged.

# Requests

The server accepts HTTP requests. The section describes the requests and the server responces.

## Accession Resolver

The format of the request:

http://<host:port>/ID/resolve?accession=<accession>

where

|  |  |
| --- | --- |
| Parameter | Description |
| host:port | Host and port where the server accepts requests, e.g. iebdev12:2180 |
| accession | Accession which needs to be resolved, e.g. P43208.1 |

Response:

The HTTP header Content-Length is set to the accession resolution binary data size.

The content is formed as follows:

* accession resolution binary data

## Blob Retrieval

There are two formats supported for the blob retrieval.

Format #1 of the request:

http://<host:port>/ID/getblob?sat=<sat>&sat\_key=<sat key>

Format #2 of the request:

http://<host:port>/ID/getblob?accession=<accession>

where

|  |  |
| --- | --- |
| Parameter | Description |
| host:port | Host and port where the server accepts requests, e.g. iebdev12:2180 |
| sat | Satellite of the blob  e.g. 4 |
| sat key | Blob key within the satellite  e.g. 42 |
| accession | Accession of the blob  e.g. P43208.1 |

Response to format #1:

The HTTP header Content-Length is set to the blob size.

The content is formed as follows:

* blob binary data

Response to format #2:

The HTTP header Content-Length is set to: 4 + accession resolution data size + blob size.

The content is formed as follows:

* 4 bytes size of the accession resolution data (network byte order)
* accession resolution binary data
* blob binary data

## Configuration

The format of the request:

http://<host:port>/ADMIN/config

where

|  |  |
| --- | --- |
| Parameter | Description |
| host:port | Host and port where the server accepts requests, e.g. iebdev12:2180 |

Response:

The HTTP header Content-Type is set to “application/json”

The HTTP header Content-Length is set approprietely

The content is formed as a JSON dictionary with the following items:

|  |  |  |
| --- | --- | --- |
| Key | Value Type | Description |
| ConfigurationFilePath | String | Full path on the server local file system to the configuration file |
| Configuration | String | The full content of the configuration file the server started with |

# Cassandra Database

For each keyspace two tables are defined.

## entity

The table stores a general blob information. If a blob is small then the blob body is saved right in this table. If a blob is large, the table stores the information of how many chunks the blob is split into and the blob body goes into the largeentity table as a sequence of chunks.

|  |  |  |
| --- | --- | --- |
| Column | Data type | Description |
| ent | int | blob identifier  it shared between the entity and largeentity tables |
| data | blob | If a blob is less than a certain size then its body is written here.  Otherwise the blob is split into chunks and the chunks are saved in the largeentity table (this field value gets null) |
| flags | bigint | A bitwise value.  1 – complete  2 – packed  4 – check failed |
| large\_parts | int | The number of chunks in a large blob.  If a blob is small and stored in this table (see the ‘data’ field) then 0 is saved here. |
| modified | timestamp | Blob modification date. |
| size | bigint | Total blob size |

## largeentity

The table stores large blob chunks.

|  |  |  |
| --- | --- | --- |
| Column | Data type | Description |
| ent | int | blob identifier |
| local\_id | int | Consequent (starting from 0) number of a chunk |
| data | blob | Blob chunk data |

# Monitoring and Maintenance

Logging, events, alerts, counters.

# Command Line Arguments

The table below describes the server command line arguments.

|  |  |
| --- | --- |
| Argument | Description |
| -help | Prints help message and exits. |
| -nodaemon | If given then the server does not daemonize. |
| -version | Prints the server version and exits. |
| -version-full | Prints the server version, the storage version and the protocol version and then exits. |
| -logfile | The file to which the server log should be redirected. |
| -conffile | The file from which the server should read the configuration. |

# Configuration Parameters

Pubseq Gateway reads the configuration from a file. The default name of the server is pubseq\_gateway so (if the –conffile command line argument is not provided) the default configuration file name will be pubseq\_gateway.ini.

The configuration file uses the NCBI standard ini file format with sections and values within sections. The sections below describe each section of the configuration file separately.

## [LMDB\_CACHE] section

|  |  |
| --- | --- |
| Value | Description |
| dbfile | Path to the file where an LMDB with a resolutions cache is located. |

## [SERVER] section

|  |  |
| --- | --- |
| Value | Description |
| port | HTTP port (1...65534)  No default. If port is not specified or is out of range the server will not start |
| workers | Number of HTTP workers (1...100). If an out of range value is provided then a default value will be used.  Default: 32 |
| backlog | Listener backlog (5...2048). If an out of range value is provided then a default value will be used.  Default: 256 |
| maxconn | Max number of connections (5...65000). If an out of range value is provided then a default value will be used.  Default: 4096 |
| optimeout | Operation timeout in milliseconds  Default: 30000 |
| countfile | A file where various event counters are printed into  If not provided (or cannot be opened for writing) and it is a non-daemon run then the counters are printed on stdout  Default: not provided |

## [CASSANDRA\_DB] section

|  |  |
| --- | --- |
| Value | Description |
| ctimeout | Connection timeout in ms  Default: 30000 |
| qtimeout | Query timeout in ms  Default: 5000 |
| namespace | Data namespace  Default: empty string |
| fallbackrdconsistency | Fallback read consistency  Default: false |
| fallbackwriteconsistency | Lower down consistency of BD write operations if local quorum can't be achieved.  0 - default cassandra driver behavior  Default: 0 |
| loadbalancing | Load balancing policy. Accepted values are: DCAware, RoundRobin  Default: DCAware |
| tokenaware | Enables TokenAware routing  Default: true |
| latencyaware | Enables LatencyAware routing  Default: true |
| numthreadsio | Number of io threads to async processing (1...32)  Default: 4 |
| numconnperhost | Number of connections per node (1...8)  Default: 2 |
| maxconnperhost | Maximum count of connections per node (1...8)  Default: 4 |
| keepalive | TCP keep-alive the initial delay in seconds  Default: 0 |
| drvlog | Location of a cassandra driver log file. If not provided then there will be no driver logs produced.  Default: empty string |
| password\_file | Cassandra password file and a section where credentials are stored. If a password\_file is not provided then password\_section value is ignored.  Default: empty, i.e. no user/password combination is used. |
| password\_section |
| service | The value is a load balancer name or a list of host[:port] items (‘,’ or ‘ ‘ separated).  If the value has neither ‘ ‘, nor ‘,’ nor ‘:’ then it is treated as a load balancer name. The load balancer resolved host ports are are sorted in accordance with their rates.  The list of host[:port] items, regardless where it came from – directly from a parameter value or from a load balancer – is analyzed further. The analizis checks that if ports are provided then they are the same. If the port is provided then it is used for the Cassandra cluster. If no ports are provided then the Cassandra driver uses its default one. |

# Appendix

## GetBlob Diagram

