

Mongo Logback Appender System Logs

Saturday, March 25, 2023 7:07 PM

These design notes specify how the system log data for the eligibility and settlement microservices should be sent to a central mongodb collection

Example code: https://github.com/btparker-work/logback_mongodb_appender.git

The system log data for the eligibility and settlement microservices should be sent to mongodb and tagged by:

- Service: name of service when running in AKS
- Instance: name of service instance in AKS
- Timestamp: time of log entry EST
- Level: log level, with default by environment as
 - INFO - dev, test (TRACE and DEBUG as needed)
 - WARN - qa, prod
- logger, thread as they come
- Message: format to give at least first 1000 characters but that is maximum

```
// customize the document
private Document getDocument(LoggingEvent event) {
    Document doc = new Document();
    // append the fields fetched from the event
    doc.append("service", "JobScheduler");
    doc.append("instance", "JobScheduler_WSEDSS1232");
    doc.append("timestamp", new Date(event.getTimeStamp()));
    doc.append("level", event.getLevel().toString());
    doc.append("logger", event.getLoggerName());
    doc.append("thread", event.getThreadName());
    doc.append("message", event.getFormattedMessage());
    return doc;
}
```

The screenshot shows the MongoDB Compass web interface. At the top, the title is 'system_log.SystemLog'. Below the title are tabs for 'Documents', 'Aggregations', 'Schema', 'Explain Plan', 'Indexes', and 'Validation'. The 'Documents' tab is selected. Below the tabs is a search bar with a 'Filter' button and a dropdown arrow. To the right of the search bar is a text input field containing the query '{ field: 'value' }'. Below the search bar are two buttons: 'ADD DATA' and 'EXPORT COLLECTION'. Below these buttons is a list of documents. The first document is expanded, showing the following fields and values: 'instance: "JobScheduler_WSEDSS1232"', 'timestamp: 2023-03-26T00:03:50.821+00:00', 'level: "INFO"', 'logger: "com.yanbo.logbackmongo.LogbackMongoExampleApplication\$\$EnhancerBySprin..."', 'thread: "main"', and 'message: "This is the INFO log"'. At the bottom of the document list, the 'id' field is shown as 'id: ObjectId('641f8be64c42320f6f667b22')'.

```
message: "This is the INFO log"

_id: ObjectId('641f8be64c42320f6f667b22')
service: "JobScheduler"
instance: "JobScheduler_WSEDSS1232"
timestamp: 2023-03-26T00:03:50.648+00:00
level: "INFO"
logger: "org.mongodb.driver.cluster"
thread: "cluster-ClusterId{value='641f8baa4c42320f6f667b1c', description='null'...}"
message: "Monitor thread successfully connected to server with description Serve..."

_id: ObjectId('641f8c054c42320f6f667b24')
service: "JobScheduler"
instance: "JobScheduler_WSEDSS1232"
timestamp: 2023-03-26T00:04:21.062+00:00
level: "WARN"
logger: "com.yanbo.logbackmongo.LogbackMongoExampleApplication$$EnhancerBySprin..."
thread: "main"
message: "This is the WARN log"
```

A custom logback appender is required to bind logback to mongodb. This is done by creating an appender extended from `UnsynchronizedAppenderBase<LoggingEvent>`.

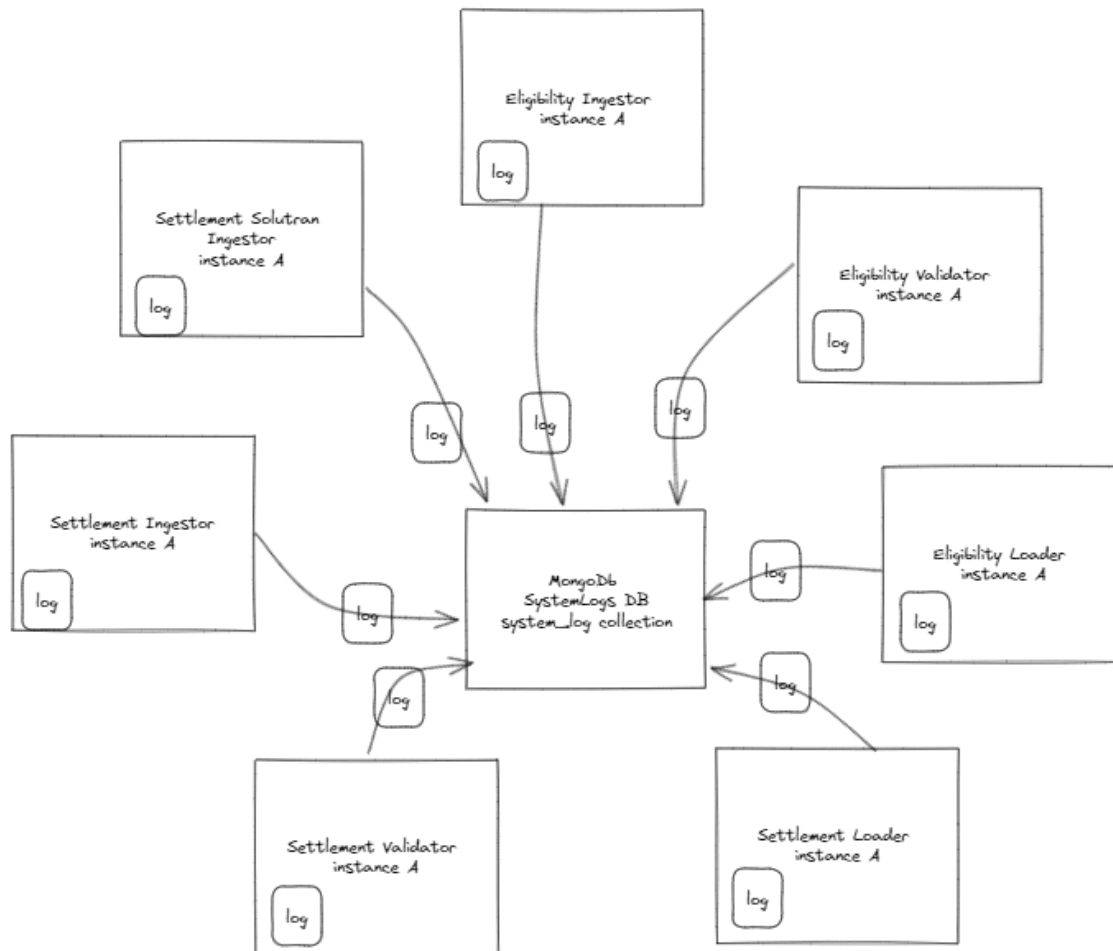
```
public class MongoAppender extends UnsynchronizedAppenderBase<LoggingEvent> {
    private MongoClient mongoClient;
    private MongoCollection<Document> collection;
    private String databaseName;
    private String collectionName;
    private String connectionUri;

    public MongoClient getMongoClient() {
        return this.mongoClient;
    }
}
```

The logback-spring.xml file should get it's mongodb connection values via `<springProperty>` entries or similar.

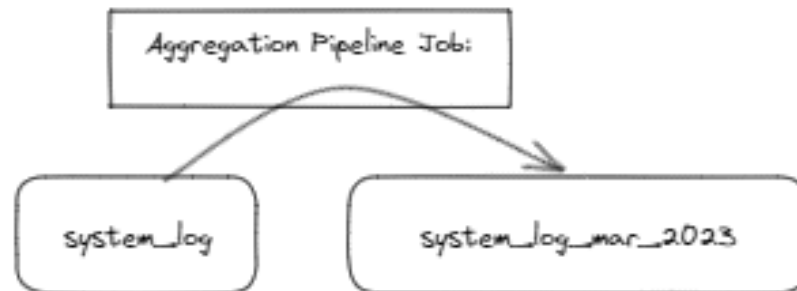
Make sure to log locally to a file as well as to mongodb for some redundancy.

Eligibility Settlement Services Centralized and Local logging



Make sure there is a job (probably done with mongodb pipeline aggregation) that will copy log files to a backup collection.

MongoDB
SystemLogs DB
system_log collection



once per month:
copy system_log to new system_log_<last month>_<year> collection
truncate system_log