# Logging

The Kaa Logging subsystem is responsible for collecting records (logs) of pre-configured structure on the endpoints, periodically transferring these logs from endpoints to Operation servers, and, finally, either persisting them on the server for further processing or submitting them to immediate stream analysis. The Kaa logs structure is determined by the configurable *log schema*.

The Kaa Logging subsystem provides the following features.

- Generation of the logging model and related API calls in the endpoint SDK
- Enforcement of data integrity and validity
- Efficient delivery of logs to Operations servers
- Storing log contents by means of the log appenders configured for the application

The application developer is responsible for designing the log schema and invoking the endpoint logging API from the client application.

## Log schema

The log schema is fully compatible with the Apache Avro schema. There is one log schema defined by default for each Kaa application. This schema supports versioning, therefore, whenever a new log schema is configured on the Kaa server for the application, this new schema gets a new sequence version assigned. The Kaa server maintains compatibility with the older versions of the log schema to ensure proper functioning of the clients that for some reason are not yet upgraded to the latest schema version.

The following examples illustrate basic log schemas.

- The simplest definition of a log record with no data fields (mostly useless)

```
{
    "name": "EmptyLog",
    "namespace": "org.kaaproject.sample",
    "type": "record",
    "fields": []
}
```

- A simple log schema with the log level, tag, and message

```
{
    "name": "LogData",
    "namespace": "org.kaaproject.sample",
    "type": "record",
    "fields": [
        {
            "name": "level",
            "type": {
                "type": "enum",
                "name": "Level",
                "symbols": [
                    "DEBUG",
                    "ERROR",
                    "FATAL",
                    "INFO",
                    "TRACE",
                    "WARN"
                ]
            }
        },
        {
            "name": "tag",
            "type": "string"
        },
        {
            "name": "message",
            "type": "string"
        }
    ]
}
```

## Log appenders

Log appenders are responsible for writing logs received by the Operations server into specific storage systems. For further information, see Log appenders.

## Confirm delivery option

By design every Kaa client stores logs in a log storage before sending them to the Kaa node. By default, upon receiving the logs, the corresponding log appender on the Kaa node sends the delivery confirmation back to the Kaa client. If the delivery was successful, the Kaa client deletes the log copi es from its local storage, otherwise - it receives an appropriate error code and either restarts the operation of log delivery or attempts to deliver the sam e data to a different Kaa node.

The confirm delivery option adds to a log appender the capability to verify whether received logs have been written into actual storage before sending the confirmation to the Kaa client.

To illustrate, let's consider the following scenarios:

1. Log appender A has been configured with the confirm delivery option enabled. The Kaa client will receive a message about successful logs d elivery only after the Kaa node successfully receives and writes the logs into an external storage. If the Kaa node fails to write the logs into the storage, the Kaa client will send the logs again. If there are more than one log appender with the enabled confirm delivery option then the Kaa client will receive a message about successful logs delivery only after each log appender successfully writes its logs into the storage.
2. Log appender A has been configured with the confirm delivery option disabled. The Kaa client will receive a message about successful logs d elivery to the Kaa node but if the Kaa node fails to write the logs into an external storage system, the Kaa client will not be informed about the failure and will not attempt sending the logs again. As a result, the logs will be lost.
3. Log appender A has been configured with the confirm delivery option enabled and log appender B with the disabled one. The Kaa client will receive a message about successful logs delivery as soon as log appender A confirms delivery. Any errors that might happen when log appender B writes its logs into the storage will not be taken into account.

To summarize, the confirm delivery option allows you to have the guaranteed delivery of every log record to the external storage.

Also, it is worth noting that by default an inmemory log storage is used on the Kaa client. This means that you can lose your undelivered data in case of an endpoint reset. To avoid this, use a persistent log storage to store all the data that was not yet confirmed as delivered.

---