FSC - Logging 1.1.0

Logius Standard
Definitive version April 21, 2026

This version:
https://gitdocumentatie.logius.nl/publicatie/fsc/logging/1.1.0/

Latest published version:
https://gitdocumentatie.logius.nl/publicatie/fsc/logging/

Latest editor's draft:
https://logius-standaarden.github.io/fsc-logging/

Previous version:
https://gitdocumentatie.logius.nl/publicatie/fsc/logging/1.0.0/

Editor:
Logius (Logius)

Authors:
Eelco Hotting (Hotting IT), Email
Ronald Koster (PhillyShell), Email
Henk van Maanen (AceWorks), Email
Niels Dequeker (ND Software), Email
Edward van Gelderen (vanG IT), Email
Pim Gaemers (Apily), Email

Participate:
GitHub Logius-standaarden/fsc-logging
File an issue
Commit history
Pull requests

This document is also available in these non-normative format: PDF

Status of This Document

This is the definitive version of this document. Edits resulting from consultations have been applied.

Table of Contents

§ Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words *MAY*, *MUST*, *RECOMMENDED*, and *REQUIRED* in this document are to be interpreted as described in BCP 14 [*RFC2119*] [*RFC8174*] when, and only when, they appear in all capitals, as shown here.

Abstract

Logging is an extension on the Federated Service Connectivity (FSC) standard FSC Core. It can only be used in addition to the FSC Core specification and if applicable the Delegation extension. The purpose of the Logging extension is to provide insights into transactions performed with services inside a FSC Group. The extension ensures Log records are uniformly described, stored and can be exchanged between Peers. The Logging extensions contains descriptions of the contents of Log Records, as well as a manner for other Peers to request Log records of transactions they have participated in as a Peer.

In addition, this extension also described the format of uniquely identifying transactions occurring within an FSC Group. Which also can be used to establish an audit trail.

§ 1. Introduction

The Logging specification is an extension of the Federated Service Connectivity (FSC) Core specification. This extension describes how Peers should log requests made to Services and how Peers should provide log records to other Peers.

§ 1.1 Purpose

Organizations should handle data in a transparent and responsible manner, partly this means that each organization should keep a log of data handled by the organization and provide insight into this log to relevant parties. FSC aims to uniform the logging of API requests to lay the groundwork for more extensive logging requirements like GDRP. As it is impossible to create a logging standard that will satisfy the requirements of each individual organization, FSC will focus on

logging the properties of an API request of which FSC can guarantee its authenticity e.g. the Peer making the request, the Peer receiving the request, the Service that is being called etc. Organizations can use this as a foundation to create a log that will satisfy all their needs.

§ 1.2 Overall Operation of Logging

A client makes a request to a Service of the Group. The Outway will receive this request and write a log record with a unique ID before proxying the request to the Inway offering the Service. The Outway will include the unique ID in the request made to the Inway. The Inway also writes a log record containing the same unique ID before proxying the request to the Service.

Peers can request log records from other Peers. Peers provide only log records in which the requesting Peer is an active party.

Log records between Peers can be matched using the unique ID.

§ 1.3 Terminology

This section lists terms (#header) and abbreviations that are used in this document. This document assumes that the reader is familiar with the Terminology of FSC Core.

Transaction:

A request made by a Peer to a Service.

TransactionLog:

A Peers log of Transactions. This log can contain both incoming and outgoing requests.

TransactionID:

A unique identifier which can be used to trace a Transaction between Peers.

§ 1.4 Profiles

When using the Logging Extension the following additions ***MUST*** be made to the FSC Profile:

1. Determine the expiration date for log records

In addition, the mandatory decisions a Profile ***MAY*** also contain additional agreements or restrictions within the Group. These are not technically required for the operation of FSC Logging extension, but can become mandatory within a Group. For example an additional set of rules to comply with local legislation. Below are a few examples listed of these additional decisions for inspirational purposes:

1. formatting restrictions on the TransactionID, for example UUIDv7

2. Architecture

2.1 Writing to the TransactionLog

A Peer makes an HTTP request to a Service. The Outway will generate a unique ID for the Transaction and write a record to the TransactionLog **before** proxying the request to the Inway. The Inway will parse the unique ID from the request and also write a record containing the unique ID to its own TransactionLog **before** proxying the request to the Service.

> The storage of the log record *MAY* be implemented both synchronously or asynchronously. For both implementations it is *REQUIRED* to receive confirmation that the log record is persisted in order to continue. For example, you can introduce a message broker to improve performance. The message broker will ensure the records are persisted later on.

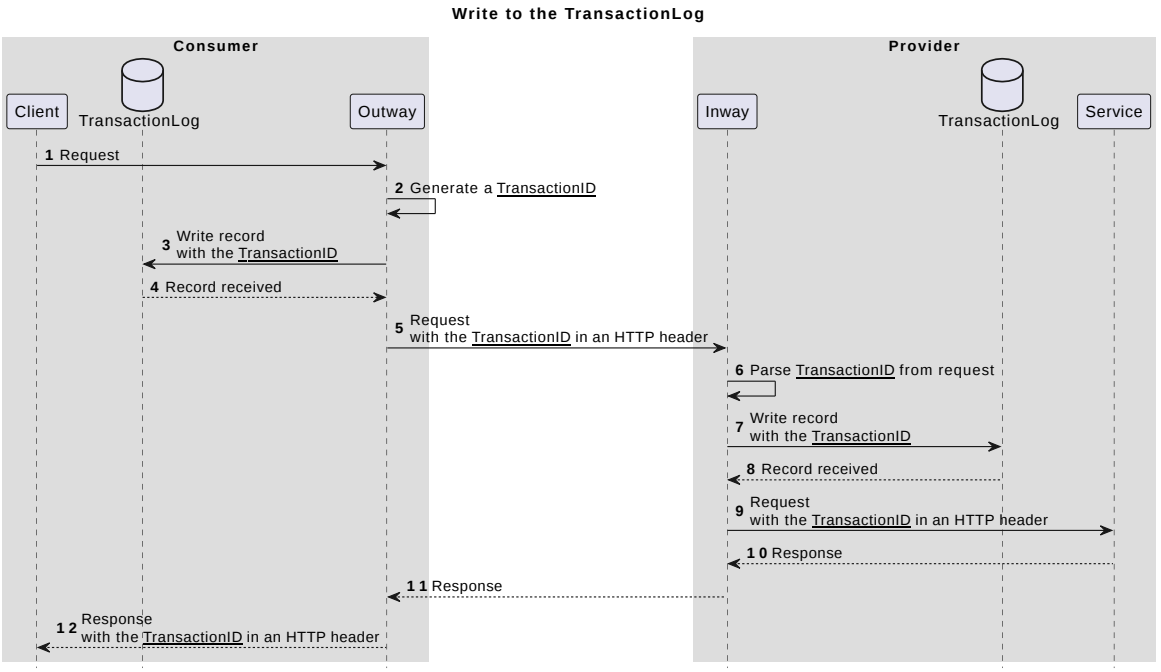

Figure 1 Write to the TransactionLog

1. The client sends a request to the Outway.
2. The Outway generates a unique ID to be used as the TransactionID.
3. The Outway writes the record for transaction in the TransactionLog.
4. The TransactionLog confirms to the Outway that the record has been received.
5. The Outway proxies the request to the Inway and includes the TransactionID in the HTTP header `Fsc-Transaction-Id`.
6. The Inway reads the unique ID from the request.
7. The Inway writes the record for transaction in the TransactionLog.
8. The TransactionLog confirms to the Inway that the record has been received.
9. The Inway proxies the request to the Service and includes the TransactionID in an HTTP header.
10. The Service returns the response to the Inway.
11. The Inway return the response to the Outway.
12. The Outway returns the response to the client. The response includes the TransactionID in an HTTP header.

§ 2.2 Providing the TransactionLog

A Peer provides the TransactionLog to other Peers. A Peer can request the records of the TransactionLog through the Manager of a Peer. The Manager returns only logs records that involve the Peer requesting the log records.

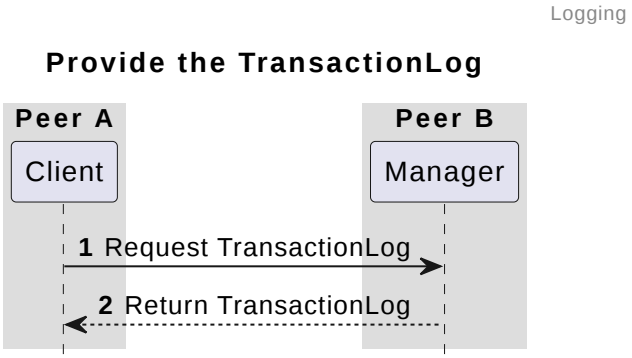

Figure 2 Provide the TransactionLog

1. Peer A requests the TransactionLog from Peer B.
2. Peer B returns the TransactionLog records that contain Peer B.

§ 2.3 Connecting log records

Each log record will have a TransactionID which is the unique ID for the Transaction. This ID is used to link the log records of a Transaction made across multiple Peers. It is ***RECOMMENDED*** to also add the TransactionID to logs created by other applications involved with the Transaction. E.g. the client making the request or the API offered as Service. This will enable Peers to provide a detailed audit trail of a request.

3. Specification

3.1 Log record

The fields that a log record ***MUST*** contain are described in the OpenAPI Specification

3.1.1 Access token

Data from the access token ***MUST*** be used to fill the following fields of the log record:

`accessToken.gth` --> `logRecord.grant_hash` `accessToken.sub` --> `logRecord.source.outway_peer_id` `accessToken.iss` --> `logRecord.destination.service_peer_id` `accessToken.svc` --> `logRecord.service_name`

in case of a Peer making a request on behalf of another Peer an additional field ***MUST*** be set:

`accessToken.cdi` --> `logRecord.source.delegator_peer_id`

in case of a request made to a Service offered on behalf of another Peer and additional field ***MUST*** be set:

`accessToken.pdi` --> `logRecord.destination.delegator_peer_id`

3.2 Manager

The FSC Log specification requires the Manager described in Core to be implemented.

§ 3.2.1 Behavior

§ *3.2.1.1 Providing TransactionLog records*

The Manager ***MUST*** be able to provide log records to other Peers.

The Manager ***MUST*** only return log records which match any of the following criteria:

- The Peer ID of the X.509 certificate used by the Peer requesting the log records matches the value of the field `logRecord.source.outway_peer_id`
- The Peer ID of the X.509 certificate used by the Peer requesting the log records matches the value of the field `logRecord.destination.service_peer_id`
- The Peer ID of the X.509 certificate used by the Peer requesting the log records matches the value of the field `logRecord.source.delegator_peer_id`
- The Peer ID of the X.509 certificate used by the Peer requesting the TransactionLog records matches the value of the field `logRecord.destination.delegator_peer_id`

§ 3.2.2 Interface

The Manager ***MUST*** implement the interface described in the OpenAPI Specification

§ 3.3 Inway

The FSC Log specification requires the Inway described in Core to be implemented.

§ 3.3.1 Behavior

§ *3.3.1.1 Writing to the TransactionLog*

The Inway ***MUST*** write a record to the TransactionLog for each received request for a Service.

The Inway ***MUST*** use the TransactionID provided by the Outway in the HTTP header `Fsc-Transaction-Id`.

The Inway ***MUST*** add the TransactionID to the request sent to the Service using the HTTP header `Fsc-Transaction-Id`.

The TransactionLog record ***MUST*** contain the fields described in the log record section

The Inway ***MUST*** deny the request if the record to the TransactionLog could not be written.

§ *3.3.1.2 Delegation*

When the requesting Peer is making the request on behalf of another Peer the source of a log record ***MUST*** contain a sourceDelegated object as described in the OpenAPI Specification.

When the Service is published on behalf of another Peer the destination of a log record ***MUST*** contain a destinationDelegated as described in the OpenAPI Specification.

§ *3.3.1.3 Error response*

This extension introduces a new error code for the Inway:

Error code	HTTP status code	Description
TRANSACTION_LOG_WRITE_ERROR	500	The TransactionLog record could not be created
INVALID_LOG_RECORD_ID	400	The format of the `Fsc-Transaction-Id` header is not

Error code	HTTP status code	Description
		valid
MISSING_LOG_RECORD_ID	400	The the `Fsc-Transaction-Id` header is missing

3.4 Outway

The FSC Log specification requires the Outway described in Core to be implemented.

3.4.1 Behavior

3.4.1.1 Writing to the TransactionLog

The Outway ***MUST*** write a record to the TransactionLog for each request that will be sent to the Inway.

The Outway ***MUST*** create a TransactionID which ***MUST*** be unique for the transaction, the format is determined in a FSC Profile.

The Outway ***MUST*** add the TransactionID to the request sent to the Inway using the HTTP header `Fsc-Transaction-Id`.

The TransactionLog record ***MUST*** contain the fields described in the TransactionLog record section

The Outway ***MUST*** deny the request if the record to the TransactionLog could not be written.

The Outway ***MUST*** add the TransactionID to the response sent to the Client using the HTTP header `Fsc-Transaction-Id`.

3.4.1.2 Delegation

When the requesting Peer is making the request on behalf of another Peer the source of a log record ***MUST*** contain a sourceDelegated object as described in the OpenAPI Specification.

When the Service is published on behalf of another Peer the destination of a log record ***MUST*** contain a destinationDelegated as described in the OpenAPI Specification.

§ *3.4.1.3 Error response*

This extension introduces a new error code for the Outway:

Error code	HTTP status code	Description
TRANSACTION_LOG_WRITE_ERROR	500	The TransactionLog record could not be created
INVALID_LOG_RECORD_ID	400	The format of the `Fsc-Transaction-Id` header is not valid
MISSING_LOG_RECORD_ID	400	The the `Fsc-Transaction-Id` header is missing

§ 4. List of Figures

§ A. References

§ A.1 Normative references

[RFC2119]
Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc2119

[RFC8174]
Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc8174