X-Road: Message Transport Protocol

Technical Specification

Version: 2.2

14.10.2015

17 pages

Doc. ID: PR-MESSTRANSP

Date	Version	Description	Author
1.07.2015	0.6	Translated to English	Siim Annuk
14.07.2015	0.7	Total restructuring	Ilja Kromonov
11.08.2015	0.8	Comments added	Margus Freudenthal
11.08.2015	0.9	Fixes according to comments	Siim Annuk
27.08.2015	1.0	Minor improvements	Siim Annuk
28.08.2015	1.1	Comments and editorial changes	Margus Freudenthal
28.08.2015	1.2	More fixes according to comments	Siim Annuk
31.08.2015	1.3	Made minor editorial changes	Margus Freudenthal
09.09.2015	2.0	Editorial changes made	Imbi Nõgisto
14.10.2015	2.1	Changes added about HTTP headers and attachments. Ports 5500 and 5577 are default configuration.	Siim Annuk
17.10.2015	2.2	Anchored the figures in place	Margus Freudenthal

Table of Contents

License

1 Introduction

This document describes the communications protocol that is used by service client's and service provider's security servers to exchange messages with each other.

1.1. Overview

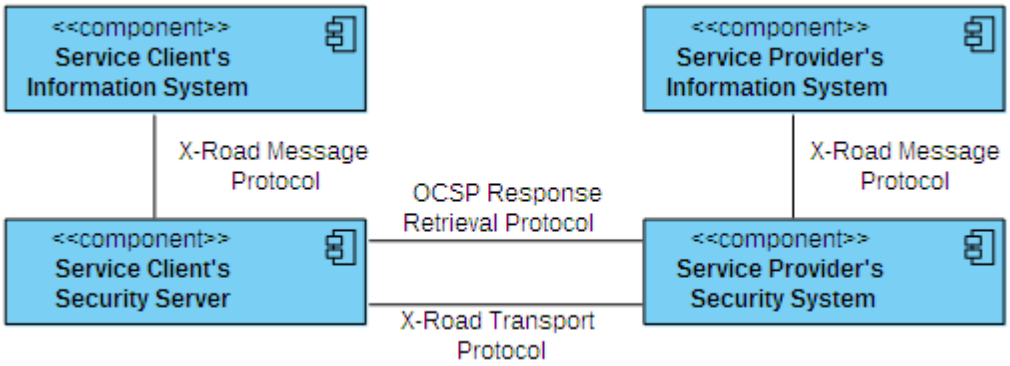

Figure 1. Protocols used in the X-Road system

As can be seen from Figure 1, three protocols are involved when exchanging messages between a service client and a service provider. These include:

- X-Road message protocol – used for communication between an information system and a security server within an organization (see [PR-MESS] for details). X-Road message protocol is a profile of the SOAP protocol[1].
- X-Road message transport protocol – a synchronous secure communication protocol that provides confidentiality and integrity when exchanging messages between two security servers over the public Internet. This protocol is described in the current document.
- OCSP Response Retrieval Protocol – the protocol used in parallel with the X-Road message transport protocol when establishing a secure communications channel between the service client's and the service provider's security servers (see Section 2.2 for details).

The communication protocol is divided into two layers (Figure 1) – the transport layer and the application layer. The transport layer uses HTTP over mutually authenticated TLS; see Section 2 for details on how the TLS session is established. The application layer consists of MIME multipart encoded X-Road transport messages that are exchanged over the transport layer (HTTPS); see Section 3 for the exact format of the message and how it's processed.

The service client's security server encapsulates the request message it receives from the service client into an X-Road transport message and in turn receives an X-Road transport message (message format described in Section 3.1) from the service provider's security server before forwarding the encapsulated response back to the service client (process described in detail in Section 3.2).

---

1 Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

The service provider's security server receives the X-Road transport message from the service client's security server and forwards the encapsulated request message to the service provider. The service provider's security server encapsulates the response from the service provider into an X-Road transport message and sends it to the service client's security server (process described in detail in Section 3.3).

Chapters 2 and 3, as well as the annex of this specification contain normative information. All the other sections are informative in nature. All the references are normative.

This specification does not include option for partially implementing the protocol – the conformant implementation must implement the entire specification.

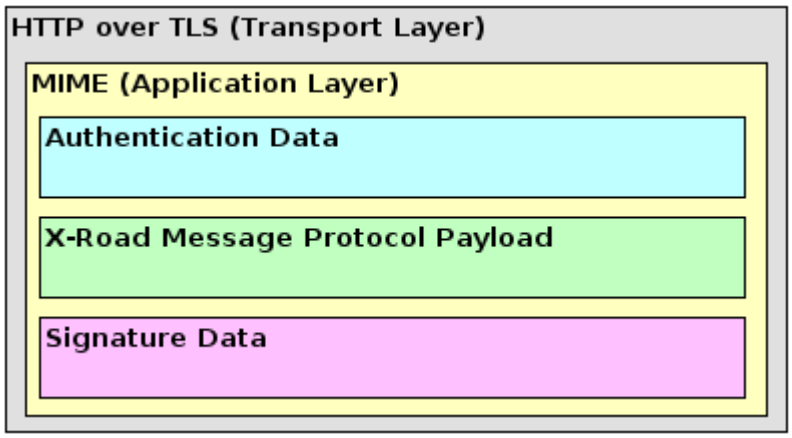

Figure 1. Layers of the X-Road message transport protocol

1.2. Terms and Abbreviations

SOAP	Simple Object Access Protocol
SOAP message	An XML document conforming to the SOAP envelope schema
SOAP message package	Contains a primary SOAP 1.1 message and may contain other entities related to the SOAP message
TLS	Transport Layer Security
OCSP	Online Certificate Status Protocol
MIME	Multipurpose Internet Mail Extensions

1.3. References

[Y-4-23] Freudenthal, Margus. Profile for High-Perfomance Digital Signature. Cybernetica Research Reports, T-4-23, 2015

[Y-4-20] Freudenthal, Margus. Using Batch Hashing for Signing and Time-Stamping. Cybernetica Research Reports, T-4-20, 2013.

[PR-MESS] Cybernetica AS. X-Road: Message Protocol v4.0.

2 Transport Layer

2.1. TLS Authentication

Security servers use authentication certificates to initiate a mutually authenticated message exchange. Each security server's authentication certificate must be registered at the central server. The certification service provider that issued these certificates must be approved by the central server. Therefore, certificate chains constructed when authenticating the connection must include certificates up to the issuing certificate of the trusted certification service provider that is registered at the central server as an approved certification authority.

The process of establishing of a secure communication channel can be described by the following steps.

1. An X-Road request message arrives at the service client's security server.
2. Service client's security server processes the request and determines the target service provider's security server.
3. Service client's security server initiates the TLS handshake with the target service provider's security server on port 5500 (default configuration).
4. Service client's security server receives the authentication certificate chain of the service provider's security server as part of the TLS handshake.
5. Service client's security server checks if the local OCSP cache contains OCSP responses for the received certificates.
6. If the OCSP responses are not cached, the service client's security server must download them from the service provider's security server and cache them locally (see Section 2.2 for details).
7. Service client's security server verifies that the authentication certificate of the service provider's security server was issued by an approved certification service provider and builds the certification chain for the authentication certificate. The certification chain and corresponding OCSP responses are then verified.
8. If verification is successful, the service client's security server forwards the X-Road transport message to the service provider's security server. If verification failed, the service client's security server sends a SOAP Fault message back to the service client's information system.
9. Having received the X-Road transport message, the service provider's security server verifies the service client's authentication certificate chain using the global configuration.

This process is illustrated in the sequence diagram in Figure 1.

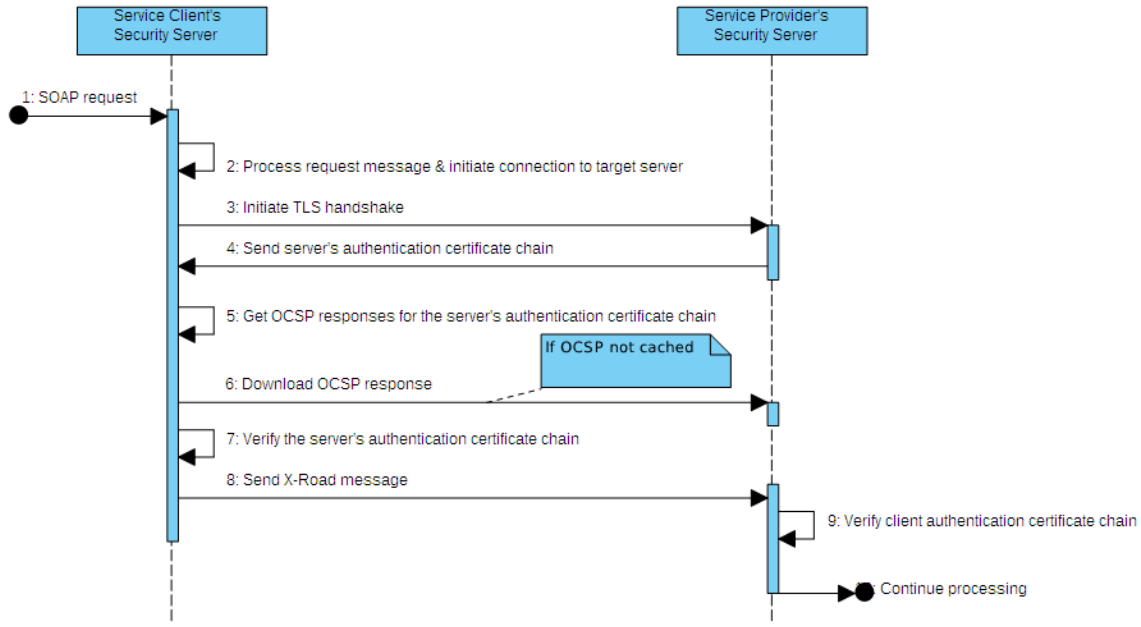

Figure 1. TLS authentication

2.2. Downloading OCSP Responses from Service Provider

Each X-Road security server only interacts with the CA that issued the certificates used by it. For this reason, the OCSP responses for certificates are always transferred together with the certificates themselves. The security servers cache the OCSP responses for their certificates and periodically update this cache.

The service client's security server sends the OCSP responses for authentication certificate as part of the request message. However, before sending the request the client's security server must verify service provider's security server's authentication certificate. Because the OCSP stapling specification is not widely implemented yet, the client's security server downloads the OCSP responses from the service provider's security server using a separate channel (HTTP).

Service provider's security server must respond to HTTP GET requests to port 5577 (default configuration). In the HTTP GET request the client's security server indicates the certificates whose OCSP responses are requested. For this, the client includes *cert* query parameters whose content is hexadecimally encoded SHA-1 hashes of the certificates. For example, the following URL is used to retrieve OCSP responses for two certificates:

```
http://SECURITYSERVER:5577/?cert=a1b2c3d4e5&cert=f6g7h8i9j0
```

where *SECURITYSERVER* is the address of the service provider's security server.

As a response to this request the service responds with a MIME multipart message (*multipart/related*). Each part of this message must contain a requested OCSP responses with content-type *application/ocsp-response*. See Annex 4.1 for an example response.

3 Application Layer

The integrity of transmitted message is ensured by signing the X-Road transport message in the security server. The signature can be either a regular signature or a batch signature. Batch signatures must be created for messages that contain attachments. If a signing key is located on a slow secure signature creation device then batch signatures may be used when signing many messages simultaneously. See [Y-4-23] for more information about how signatures are created.

The X-Road message transport protocol is designed for streaming the message contents (e.g. attachments) between security servers. The signature can be calculated after the previous parts (e.g. attachments) of the transport message have been transferred to the other party. Streaming the message contents puts restrictions on how the signature of the transport message must be verified. The contents of the transport message must be cached in the security server before the signature of the message can be verified, because the verification result determines the validity of the message – the security server must not forward an invalid message to the other party.

3.1. X-Road Transport Message

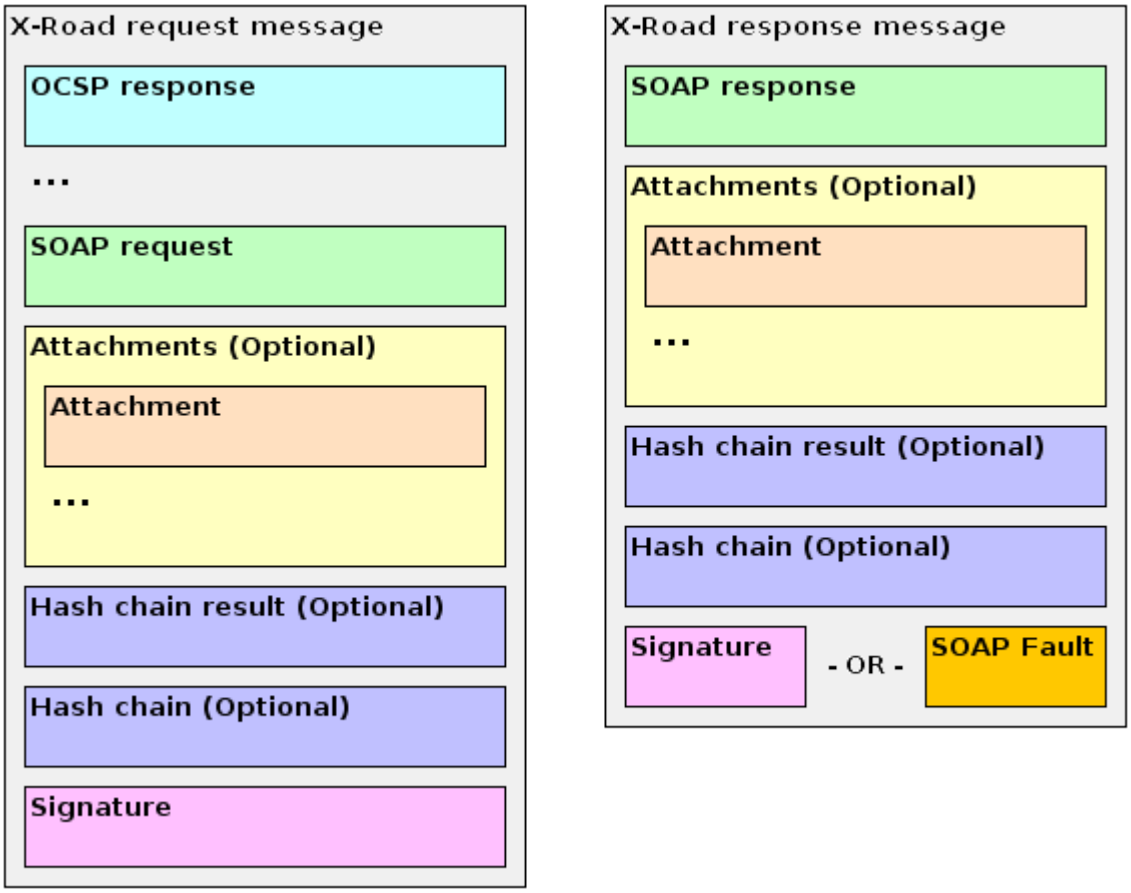

Figure 1. X-Road transport messages

The X-Road transport messages are encoded as MIME multipart messages with content-type *multipart/related.* The content-type of the client request message is sent from the service client's security server to the service provider's security server and vice versa using the “x-original-content-type” HTTP header. The value of the original content type is used to forward the request or response message to the service provider's or service client's information system. All other HTTP headers sent by the service client's security server or service provider's security server are not preserved in the security server. MIME headers in the multipart message are preserved.

The X-Road transport message encapsulates either the SOAP message package that arrives to the security server or a SOAP fault message (uses content-type *text/xml* instead of *multipart/related*). The latter is only sent from the service provider's security server to the service client's security server if an error occurred before processing the request message in the service provider's security server. The normal X-Road request message must consist of the following MIME message parts (see Figure 1. The parts are mandatory unless stated otherwise):

1) byte contents of OCSP responses (content-type *application/ocsp-response*) of the service client's security server authentication certificate chain that was used to authenticate the TLS connection;
2) the SOAP message (content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package);

3) (optional) a nested MIME multipart (content-type *multipart/mixed*) containing all attachments as parts. This part is only present if the original SOAP message package contains attachments;

4) (optional) if the signature is a batch signature, then:

   a) the hash chain result XML (content-type *application/hash-chain-result*) and

   b) the hash chain XML (content-type *application/hash-chain*) of the signature.

5) the signature XML (content-type *signature/bdoc-1.0/ts*) associated with the SOAP message and any attachments of the encapsulated message;

The normal X-Road response message must consist of the following MIME message parts (see Figure 1. The parts are mandatory unless stated otherwise):

1) the SOAP message (content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package);

2) (optional) a nested MIME multipart (content-type *multipart/mixed*) containing all attachments as parts. This part is only present if the original SOAP message package contains attachments;

3) (optional) if the signature is a batch signature, then:

   a) the hash chain result XML (content-type *application/hash-chain-result*) and

   b) the hash chain XML (content-type *application/hash-chain*) of the signature.

4) one of the following:

   a) the signature XML (content-type *signature/bdoc-1.0/ts*) associated with the SOAP message and any attachments of the encapsulated message; or

   b) a SOAP fault XML (content-type *text/xml*), if any errors occurred during the processing of the message (i.e. error when creating signature). Since the previous parts of the message have already been sent to the other party, the SOAP fault must be sent as the last part.

3.2. Message Handling in Service Client's Security Server

The following describes the actions that the service client's security server must take in order to perform a secure message exchange between a service client and a service provider.

1. Receive a SOAP message or a SOAP message package (if attachments are present) from the service client (message format described in [PR-MESS]).
2. Parse the SOAP message to determine the target service provider.
3. Establish TLS connection with it's security server (see Section 2.1).
4. Send an X-Road transport message to the service provider's security server (message format described in Section 3.1) in the following steps:

   a) Add the following HTTP headers to the HTTP headers of the HTTP request:

      - Hash algorithm identifier (x-hash-algorithm). The hash algorithm is used by the other party to calculate the hashes of the message parts to be used during message verification.
      - Original content type (x-original-content-type) of the request message.

   b) Write an OCSP response part to the transport message (content-type

application/ocsp-response) for each OCSP response in the authentication certificate chain used for establishing the TLS connection.

c) Write the service client's request SOAP message (content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package) to the transport message. Calculate the hash of the request SOAP message.

d) If the original request was a SOAP message package, write a nested MIME multipart (content-type *multipart/mixed*) containing all attachments as parts. Copy the MIME headers of each attachment part and calculate the hash of the data.

e) Calculate the signature using the stored message and attachment hashes in accordance with [Y-4-23, Y-4-20]. Write the signature as the last part of the message (content-type *signature/bdoc-1.0/ts*).

5. Start reading a response from the target service provider's security server (message format described in Section 3.1).

6. If the content-type of the response is *multipart/related* then process the message parts as follows:

   a) The first part must be the encapsulated SOAP response message with content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package. The message is not forwarded to the service client until it can be verified.

   b) If the content-type of the next part is *multipart/mixed* then this part is the nested attachments multipart.

   c) If the content-type of the next part is *application/hash-chain-result* then this message contains a batch signature. The hash chain result is stored for message verification.

   d) If the content-type of the next part is *application/hash-chain* then this message contains a batch signature. The hash chain is stored for message verification. The hash chain result must be present if the hash chain is present.

   e) If the content-type of the last part is *signature/bdoc-1.0/ts* then the part contains the signature of the message. If the content-type of the part is *text/xml* then the part contains a SOAP fault indicating that an error occurred during the processing of the message in the service provider's security server and it must be returned to the service client.

   If the content-type of the response is *text/xml* then an error occurred at the service provider's security server and the received SOAP Fault must be returned to the service client. In case of any other content-type, the response is malformed and a corresponding SOAP Fault must be returned to the service client.

7. Verify the response message using the stored message hash, attachment hashes, and signature in accordance with [Y-4-23, Y-4-20].

8. Send the service provider's encapsulated response SOAP message (or a SOAP message package in case the response has attachments) to the service client.

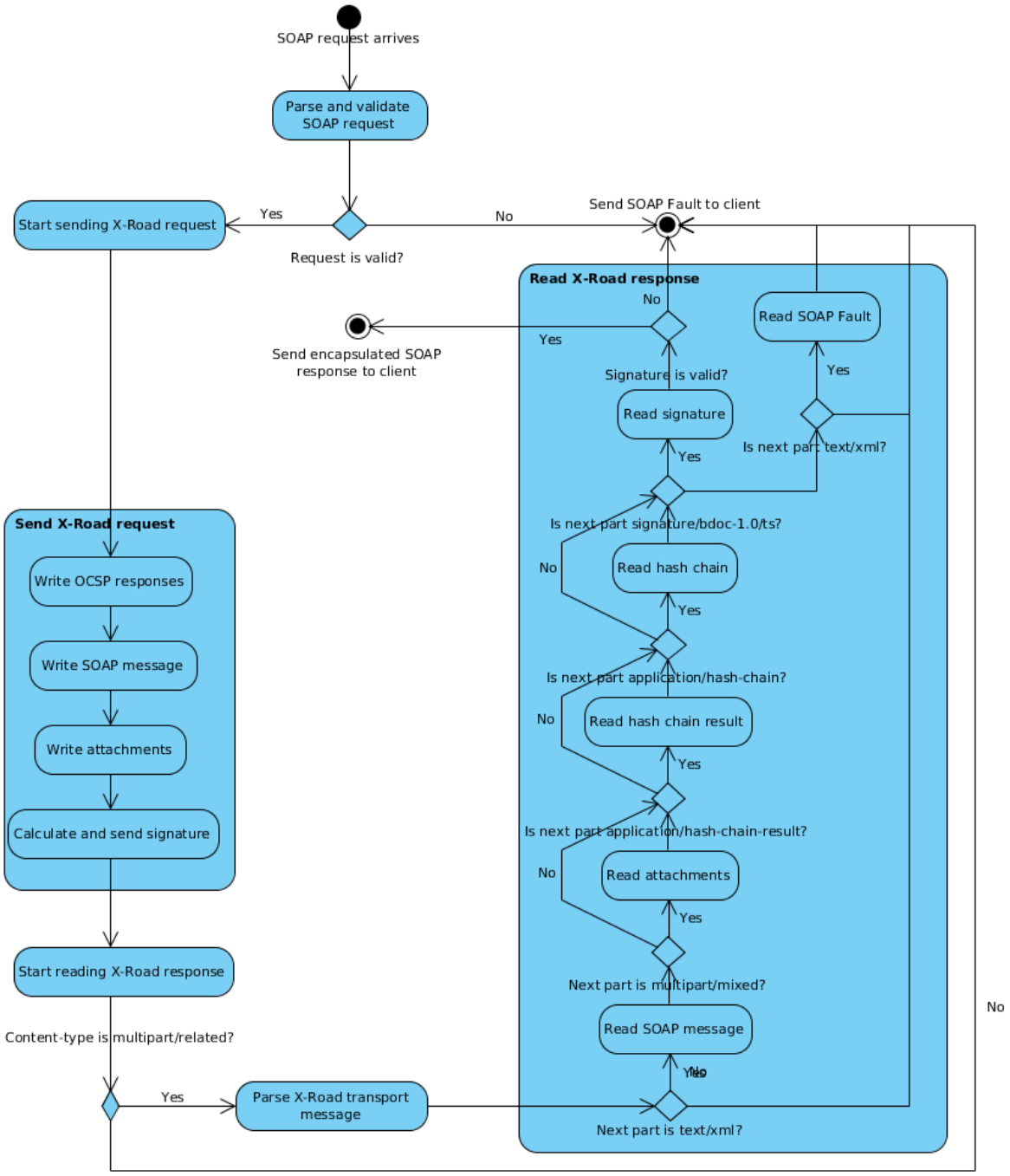

Figure 1. Message processing on service client's side

3.3. Message Handling in Service Provider's Security Server

The following describes the actions that the service provider's security server must take in order to perform a secure message exchange between a service client and a service provider.

1. Establish TLS connection with the service client's security server (see Section 2.1).

2. Start reading the X-Road transport message from the service client's security server (message format described in Section 3.1).
3. The content-type of the request message must be *multipart/related.* The security server must process the message parts as follows:
    a) Read all the parts with content-type *application/ocsp-response*. These parts contain OCSP responses that must be used in when verifying the authentication certificate chain of the service client's security server.
    b) The part that comes after OCSP responses must have the content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package, this part is the encapsulated SOAP request message. The message is not forwarded to the service provider until it can be verified.
    c) If the content-type of the next part is *multipart/mixed* then this part is the nested attachments multipart.
    d) If the content-type of the next part is *application/hash-chain-result* then this message contains a batch signature. The hash chain result is stored for message verification.
    e) If the content-type of the next part is *application/hash-chain* then this message contains a batch signature. The hash chain is stored for message verification.
    f) If the content-type of the last part is *signature/bdoc-1.0/ts* then the part contains the signature of the message. If the content-type of the last part is *text/xml* then the part contains a SOAP fault indicating that an error occurred during the processing of the message in the service client's security server.
4. Verify the request message using the stored message hash, attachment hashes, and signature in accordance with [Y-4-23, Y-4-20].
5. Send the encapsulated SOAP message and any attachments to the target service provider.
6. Start reading a response from the target service provider (message format described in [PR-MESS]).
7. Send an X-Road transport message to the service client's security server (message format described in Section 3.1) in the following steps:
    a) Add the following HTTP headers to the HTTP headers of the HTTP request:
        - Hash algorithm identifier (x-hash-algorithm). The hash algorithm is used by the other party to calculate the hashes of the message parts to be used during message verification.
        - Original content type (x-original-content-type) of the request message.
    b) Write the service provider's response SOAP message (content-type *text/xml* or *application/xop+xml* in case the original message is a MTOM-encoded SOAP message package). Calculate the hash of the response SOAP message to be used when creating the signature.
    c) If the response from the service provider was a SOAP message package, write a nested MIME multipart (multipart*/mixed*) containing all attachments as parts. For each part, calculate the hash of the data to be used when creating the signature.
    d) Calculate the signature using the stored message and attachment hashes in accordance with [Y-4-23, Y-4-20]. Write the signature as the last part of the message (content-type *signature/bdoc-1.0/ts*).

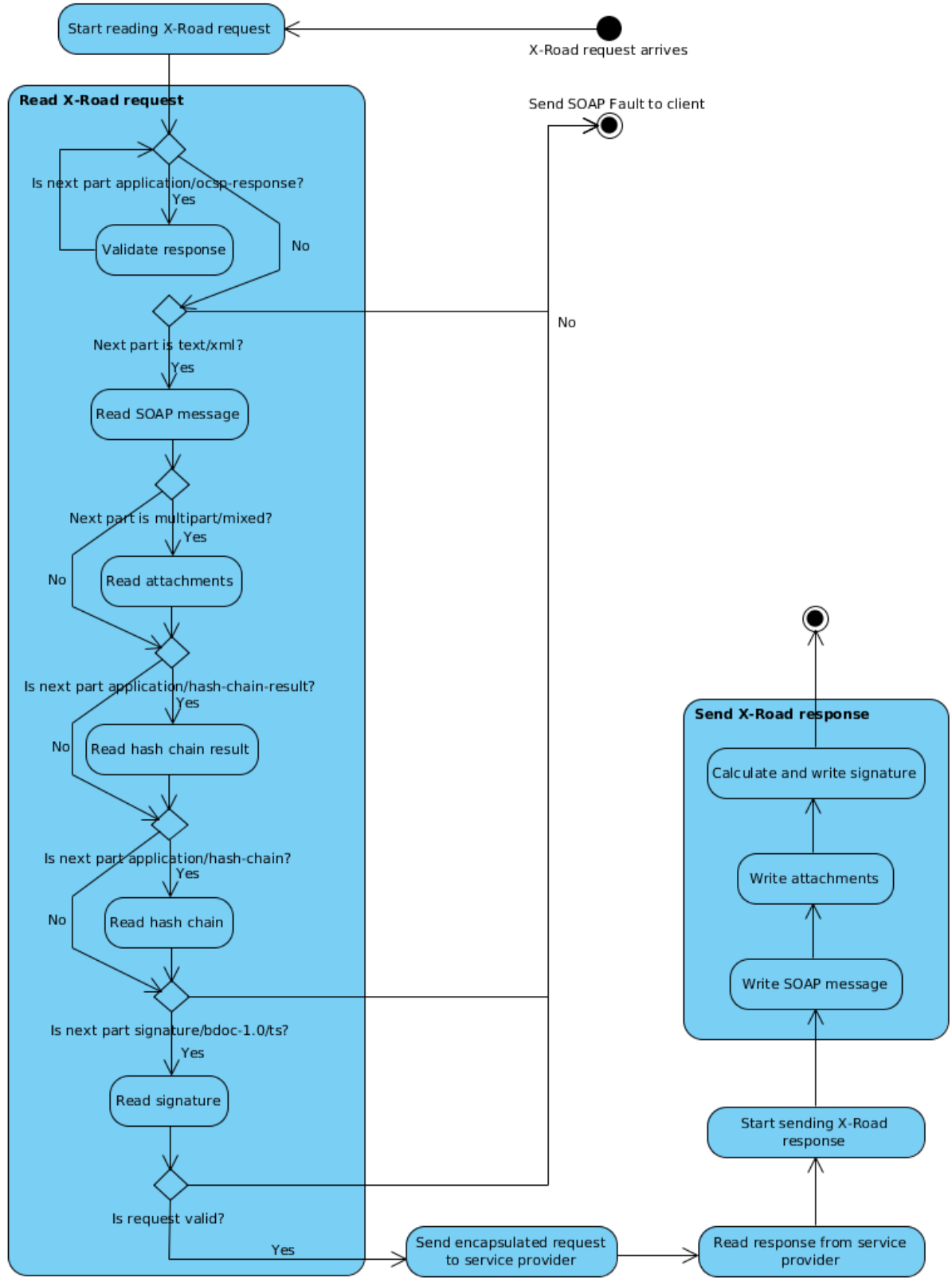

Figure 1. Message processing on the service providers side

4 Annex: Example Messages

All the example messages only contain the relevant HTTP headers (content-type) and the headers of the MIME parts. The contents of the MIME parts are omitted for brevity.

4.1. Response to OCSP Downloading Request

```
Content-Type: multipart/related; charset=UTF-8; boundary=jetty625909216ic7gfi1u

--jetty625909216ic7gfi1u
Content-Type: application/ocsp-response

...bytes of the ASN.1-encoded OCSP response...
--jetty625909216ic7gfi1u--
```

4.2. Simple Request

```
Content-Type: multipart/related; charset=UTF-8; boundary=xtop1357783211hcn1yiro

--xtop1357783211hcn1yiro
Content-Type: application/ocsp-response

...ocsp response...
--xtop1357783211hcn1yiro
Content-Type: text/xml ; charset=UTF-8

...request SOAP...
--xtop1357783211hcn1yiro
Content-Type: signature/bdoc-1.0/ts

...signature XML...
--xtop1357783211hcn1yiro--
```

4.3. Simple Response

```
Content-Type: multipart/related; charset=UTF-8; boundary=xatt569125687hcu8vfma

--xatt569125687hcu8vfma
Content-Type: text/xml ; charset=UTF-8

...response SOAP...
--xatt569125687hcu8vfma
Content-Type: signature/bdoc-1.0/ts

...signature XML...
--xatt569125687hcu8vfma--
```

4.4. Request with Attachments

```
Content-Type: multipart/related; charset=UTF-8; boundary=xtop1357783211hcn1yiro

--xtop1357783211hcn1yiro
Content-Type: application/ocsp-response

...ocsp response...
```

```
--xtop1357783211hcn1yiro
Content-Type: text/xml; charset=UTF-8

...request SOAP...
--xtop1357783211hcn1yiro
Content-Type: multipart/mixed; charset=UTF-8; boundary=xtop569125687h3h10du0

--xtop569125687h3h10du0
Content-Type: text/plain; charset=UTF-8
Content-Location: attachment.txt

...attachment data...
--xtop569125687h3h10du0
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64
Content-Id: <data.bin>

...attachment data...
--xtop569125687h3h10du0 --

--xtop1357783211hcn1yiro
Content-Type: application/hash-chain-result

...hash chain result XML...
--xtop1357783211hcn1yiro
Content-Type: application/hash-chain

...hash chain XML...
--xtop1357783211hcn1yiro
Content-Type: signature/bdoc-1.0/ts

...signature XML...
--xtop1357783211hcn1yiro--
```

4.5. Response with Fault as Last Part

```
Content-Type: multipart/related; charset=UTF-8; boundary=xatt569125687hcu8vfma

--xatt569125687hcu8vfma
Content-Type: text/xml; charset=UTF-8

...response SOAP...
--xatt569125687hcu8vfma
Content-Type: text/xml; charset=UTF-8

...SOAP fault...
--xatt569125687hcu8vfma--
```