TARGET Instant Payment Settlement

MEPT - Message Exchange Processing for TIPS

Appendix 1 to the TIPS Connectivity - Technical Requirements

v1.~~1~~2

Author	4CB
Version	1.2
Date	16/08/2019

1. Scope

TIPS communicates with different participants (either Central Banks or Payment System Providers) to provide service; communications are in charge of **Network Service Providers (NSP)** chosen by participants and fulfilling the TIPS Connectivity requirements. The variety of NSPs arise the need of a unified messaging processing, where the message content and other technical information are passed as a "single entity".

The following document is directed to NSPs and is part of the TIPS Network Connectivity Requirements. It guides NSPs to connect to the TIPS system and describes the **MEPT** (acronym for **"Message Exchange Processing for TIPS"**), the ruleset that needs to be implemented in order to exchange messages with the TIPS Platform.

The following figure shows where MEPT is used within message exchange flow of TIPS A2A messages. The connection from NSP and TIPS platform is in charge of the NSP, who provides gateways physically hosted in TIPS datacentres. The exchange of data between the Actor and the NSP is based on a protocol defined by the relevant NSP and the NSP offers connectivity services and manages the bi-directional data exchange with TIPS platform according to MEPT.

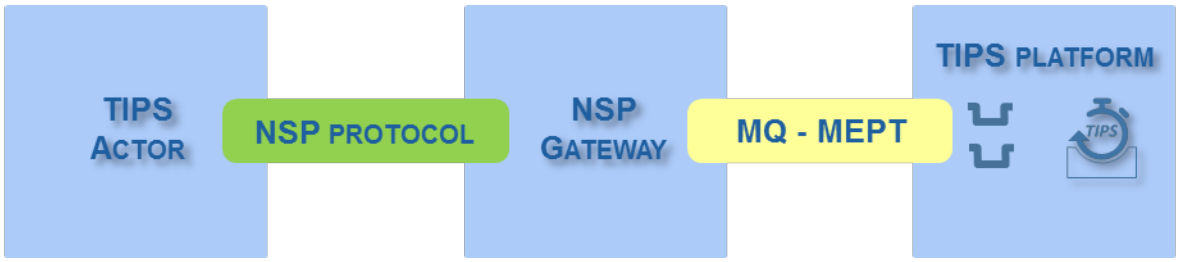


FIGURE 1 – A2A MESSAGES

2. Introduction

Central Banks and TIPS Participants - the latter also on behalf of their Reachable Parties and/or Instructing Parties - can choose their preferred Network Service Provider (NSP), provided that the selected NSP fulfils the TIPS Connectivity Requirements and has passed the Compliance Check done by the TIPS Operator.

Contractual relationship can be established only after the successful completion of the compliance check phase.

The NSP shall provide the following services**:**

- Network connectivity;
- Messaging services in U2A and A2A mode;
- Security services: PKI and Closed Group of User (CGU) management;
- Operational services: Support and incident management.

2.1. NSP gateway

Communication between TIPS and NSPs is ensured by the NSP Gateways. This infrastructure, hosted in the TIPS datacenters, belongs to the perimeter of responsibility of the NSP.

All the functionalities provided at TIPS side are encompassed within the NSP Gateways infrastructure.

Additionally, an application is provided in order to control gateway operation (such as start, stop, login, etc.).

Communication between NSP Gateway and TIPS application is based on sets of queues provided by TIPS, one set for incoming traffic (traffic sent from TIPS Actor and received by TIPS application), one set for outgoing traffic (traffic sent from TIPS application and received by TIPS Actor) and one set dedicated to files.

The channel between gateways and TIPS transporting A2A messages is protected by Transport Layer Security protocol. Additionally, Local Authentication (LAU) based on HMAC method is used.

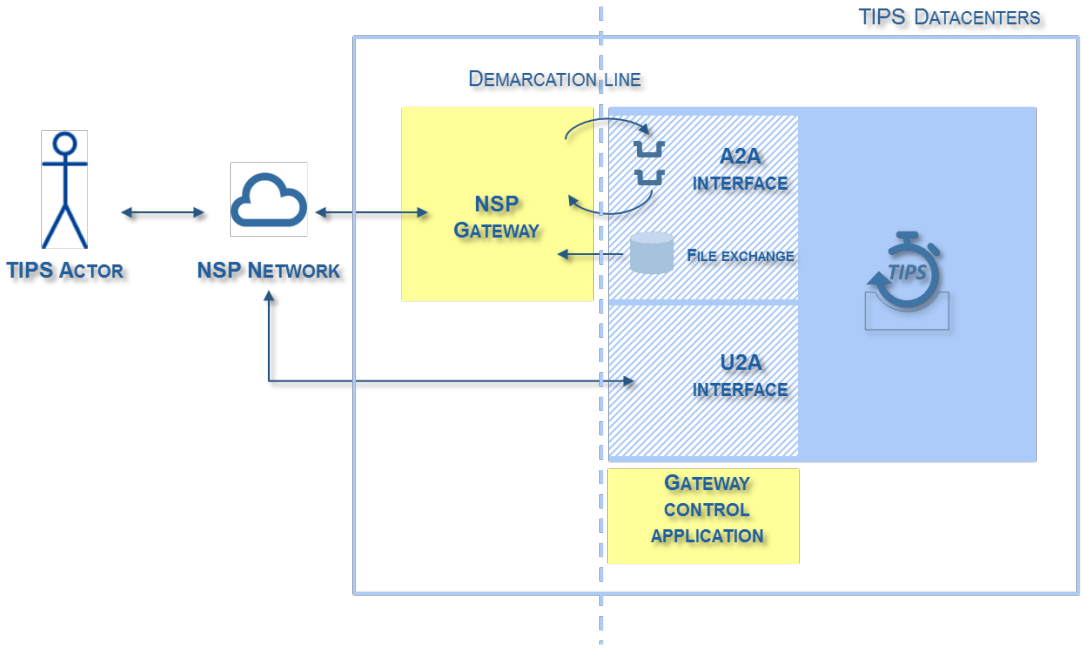


FIGURE 2 – NSP INFRASTRUCTURES IN THE TIPS DATACENTERS

2.2. Gateway control application

The NSP must provide a control application, running on TIPS RHEL server, that allows controlling of NSP gateways via a GUI interface. Such control will allow:

- Starting, stopping a gateway (optional feature);
- Disable/Enable a gateway from processing traffic (optionally, with possibility to disable/enable outgoing traffic towards participants and incoming traffic arriving from participants);
- Login/logout of a gateway to/from the network;
- Displaying gateway status and traffic monitoring;

- Renewal of the LAU symmetric key between the TIPS application and all the gateways.
- Display information about LAU symmetric key (last renewal time, time remaining for next renewal, adoption time of last key on each gateway)

2.3. The communication modes

The TIPS Actor applications and the end-users can communicate with TIPS with two primary modes: **Application to Application (A2A**) or **User to Application (U2A)**. Additionally, TIPS platform **Sends files** to TIPS Participants (currently, only for sending periodical reports). U2A mode doesn't use MEPT and, thus, is not covered in this description.

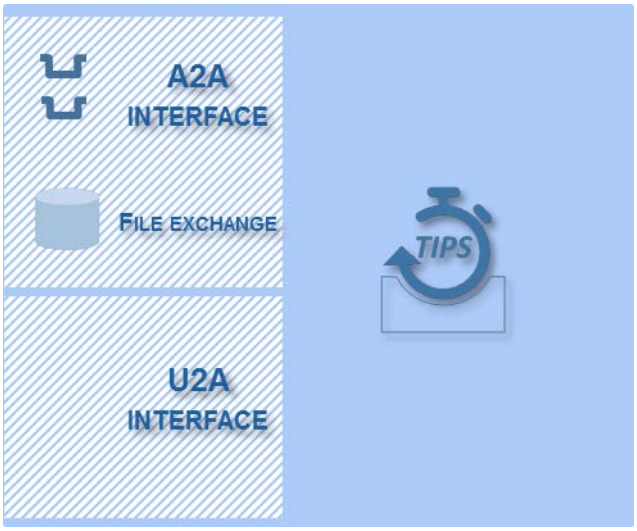


FIGURE 3 – TIPS TYPES OF CONNECTIVITY

2.3.1. A2A Instant messaging

For the A2A instant messaging mode, the TIPS Platform communicates with the participants only using "stateless" messages and with no support of "store-and-forward". This implies that in the case of unavailability of the receiver no retry mechanism is in place.

All communication is in "push" mode, both from TIPS application to the TIPS Actor and from the TIPS Actor to TIPS application. The "push" mode refers to the originator of a message pushing it to the final receiver.

The A2A message exchange between TIPS and the NSP is based on a ruleset named MEPT described hereafter. MEPT relies on XML messages, transported over an MQ connection and containing all the relevant information

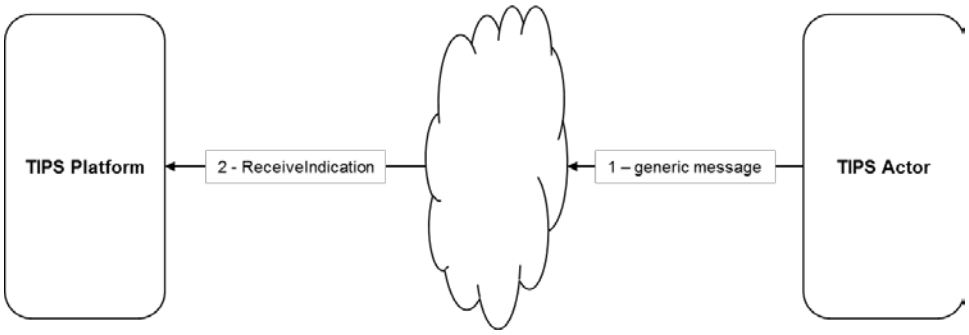
to address and describe messages. The connection from NSP and TIPS is in charge of a NSP gateways physically hosted in TIPS datacenters.

Each NSP offers connectivity services and manages the bi-directional data exchange between his Actors and the TIPS Platform according to the MEPT.

The NSP provides several functionalities: Technical Sender Authentication, CUG, non-repudiation, encryption, NSP protocol transformation into and from MEPT messages.

2.3.1.1. A2A instant messaging - Incoming flow management

The NSP shall manage the instant incoming message pattern as detailed in the following picture:



When the TIPS Platform receives a message from NSP's Network Gateway it will go through the following steps:

1) The TIPS Actor sends the message to the NSP (using a protocol that is not in the scope of this document);
2) The Network Gateway of the TIPS Platform receives the message from the sender TIPS Actor and performs the validation of the received signature. In case of positive processing, the Network Gateway of the TIPS Platform sends a ReceiveIndication primitive to the TIPS Platform. The TIPS Platform receives the message and performs the validation check of the "Local Security" header.

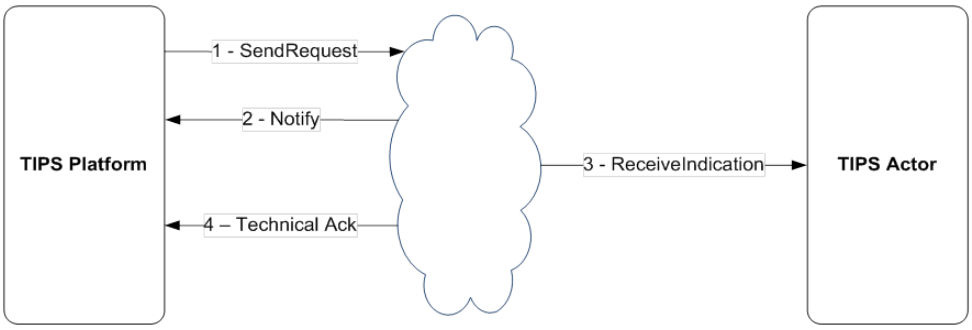
   The message is passed to the TIPS application.

The primitive used in the incoming message processing are:

A ***ReceiveIndication*** primitive is used whenever a message is delivered from NSP to the TIPS system. This type of message provides all information describing the message itself (such as sender, receiver, signature, etc.) and the business message transported.

2.3.1.2. A2A Instant messaging - Outgoing flow management

The NSP shall manage the instant outgoing message pattern as detailed in the following picture:



When the TIPS Platform needs to send a message to TIPS Actor it will go through the following steps:

1) The TIPS Platform sends a “SendRequest” primitive to its Network Gateway
2) The Network Gateway of the TIPS Platform receives the message and performs the validation check of the "Local Security" header. In case of positive validation, the unique network message identifier is generated and the message is signed.

   In case of error, the Network Gateway of the TIPS Platform sends back to TIPS Platform a negative Notify and the flow is completed. In case of positive processing, the Network Gateway of the TIPS Platform sends the message to the TIPS Actor. If the sending to the TIPS Actor is successful, the Network Gateway of the TIPS Platform sends back to TIPS Platform a positive Notify.
3) The TIPS Actor receives the message from the Network Gateway of the TIPS Platform and performs the validation of the received signature.
4) The Network Gateway of the sender TIPS Platform receives the outcome of the processing and sends back a positive/negative Technical Ack to the TIPS Platform depending on:
   - a failure in the validation of the received message performed at TIPS Actor side
   - the outcome (positive or negative) of the delivery of the message to the TIPS Actor

The primitive used in the outgoing message processing are:

A ***SendRequest*** primitive, activated by TIPS when a message has to be delivered to a TIPS Actor. This type of message provides to NSP all information describing the message itself (such as sender, receiver, etc.) and the business message to be transported.

A ***Notify*** primitive is provided for each request to send a message or a file between the TIPS Platform and the NSP Network Gateway to notify the outcome of the initial processing performed by the Network Gateway: local security check, addressing resolution, header validation etc. In case of negative result, a reason code of the error detected is returned. In case of positive result, the unique "network" message / file identifier and the signature of the message are sent back.

A *Technical Ack* primitive is provided for each request to send a message between the TIPS Platform and the NSP Network Gateway to notify the completion of the exchange. In case of negative result, a reason code of the error detected is returned. In case of positive result, the unique "network" message / file identifier, a timestamp of the delivery of the message / file to the TIPS Actor are sent back.

Additionally, for outbound file transfer, the primitive used in the TIPS - NSP communication is called *FileSend* (see after) and its purpose is to convey the information about the delivery of the file (e.g. the receiver) and not the file itself.

2.3.1.3. A2A Instant messaging - Message size

The NSP shall support the exchange of messages with maximum length set to 10KiB (1 KiB = 1.024 bytes). The maximum length refers to the business content of the transferred message, without taking into account the communication protocol overheads.

The NSP shall reject as soon as possible any message that is not in the allowed size range.

The NSP shall reject the operation by sending back to the originator a negative acknowledgement message with the explanation of the error (e.g. "Message size out of allowed range.").

2.3.2. File store and forward

The file transfer operates in store-and-forward mode and, as such, enables a sender to transmit files even when a receiver is unavailable. In the case of temporary unavailability of the receiver, the NSP stores files for 14 calendar days (for PROD environment) and delivers them as soon as the receiver becomes available again.

The size is maximum 1 GB.

File transfer mode is used by TIPS Platform only for outgoing exchange, there is no business case for using it for communication from the TIPS actor to the TIPS platform.

When a file has to be sent, TIPS as a first step stores it on an dedicated RHEL server (located in the SSP ESMIG perimeter). Then file transmission starts from TIPS by sending an MQ message to Gateway using the same rules (MEPT) described ahead and specifying the primitive name *FileSend*. The message contains the file name and the file system path to be used to get the file. The NSP Gateway, then, has the responsibility to get the file that has been stored beforehand by TIPS and to send it to the recipient.

The notify primitive is used to indicate to TIPS that the NSP Gateway has read the file and taken responsibility to send the file. This allows TIPS to remove the file.

3. Messaging rules – MEPT (Message Exchange Processing for TIPS)

Toward the TIPS system, the NSP layer uses queues implemented using **IBM WebSphere MQ** off-the-shelf products.

A generic TIPS message is composed by two main sections:

- The **message header**: this section will contain all the information that enrich the message but are not strictly related to the message content (routing, signature, etc..)
- the **message payload**: this section will contain the ISO business message or the payload of the specific TIPS message such as *TechnicalAck* or *Notification*

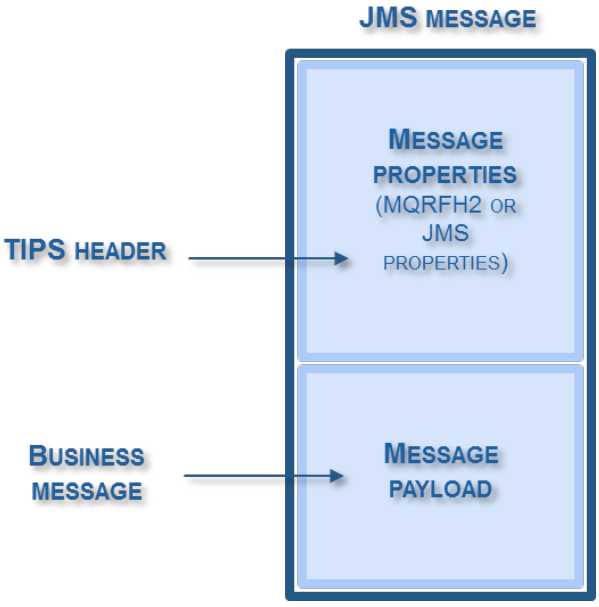


FIGURE 4 – SECTIONS OF A MESSAGE

3.1. Message header

- The message header completes a message with a subset of information strictly related to the message's lifecycle; IBM WMQ as the target technology for message exchange, the message header is composed by JMS properties in the following table. A messages refers to a primitive and all properties apply to all primitive. The following table shows for each property the primitives that apply to it. All properties listed in table are mandatory in the reported primitive, unless stated otherwise ("optional").
- The type of all the JMS properties required by MEPT is String.

- The "Max length", unless stated as "enforced", represents an indication of what should be considered as the maximum reasonable length of each property, based on how that parameter is currently used by the system. For example, the MsgBizIdentifier property has a maximum length of 35 characters because in some circumstances it appears in XML messages inside elements whose maximum allowed length is 35 characters.

Property	Description	Where filled	Used in HMAC calcula-tion	Stored for NRO
HMAC	Hash Message Authentication Code for Local AUtentication between TIPS and NSP gateway. Encoded in base64. Max length: 45	In all types of message	No	No
HMACKeyId	Identifier of the bilateral key to be used for HMAC check	In all types of message	No	No
HMAC2	Second (optional) HMAC for Local AUthentication, to be used when the NSP need to protect communication beyond the link between TIPS and NSP gateway. TIPS adds HMAC2 only when all the following conditions are met: - The message is a *SendRequest* - *SignatureRequired* = yes - The gateway control application already sent to TIPS the values for the HMAC2 Key and HMAC2 KeyId. Encoded in base64. Max length: 45	SendRequest	No	No
HMAC2KeyId	Identifier of the bilateral key to be used for (optional) HMAC2 check	SendRequest	No	No

HMACAlgo	Name of the hashing algorithm used in HMAC and HMAC2 calculation. Optional, when not present the value "SHA-256" is assumed	In all types of message	No	No
MsgSignature	Message signature Max length: 4000 bytes, enforced	*ReceiveIndication, Notify*	No	No
ProtocolVersion	The MEPT version. It must be set to "1" Max length: 1	In all types of message	Yes	No
Service	The service name (e.g. TIPS-PRODUCTION, TIPS-TEST)	In all types of message	Yes	Yes
Sender	The distinguished name of the actor sending the message. For *Notify* and *TechnicalAck*: refers to the original message. Max length: 256	In all types of message	Yes	Yes
Receiver	The distinguished name of the actor receiving the message. For *Notify* and *Technical Ack*: refers to the original message. Max length: 256	In all types of message	Yes	Yes
PrimitiveType	Allowed values: *SendRequest, ReceiveIndication, Notify, TechnicalAck, FileSend.* Max length: 17	In all types of message	Yes	No
MsgType	The ISO message type. Max length: 32	*ReceiveIndication, SendRequest, FileSend*	Yes	No
SendTimestamp	Timestamp the message has been retrieved from the sender's gateway (start of the NSP perimeter). It is exposed as YYYY-MM-DDTHH:MM:SS.SSSZ, where **T** is the delimiter between date and	*ReceiveIndication, Notify, TechnicalAck*	Yes	Yes

	time and **Z** is a zone designator for the zero UTC offset. For *Notify* and *TechnicalAck*: refers to the original *SendRequest*. Max length: 24			
ReceiveTimestamp	Timestamp the message has been put on the receiver's queue (end of NSP perimeter). It is exposed as YYYY-MM-DDTHH:MM:SS.SSSZ, where **T** is the delimiter between date and time and **Z** is a zone designator for the zero UTC offset. For *TechnicalAck*: refers to the original *SendRequest*. Max length: 24	*ReceiveIndication,* *TechnicalAck*	Yes	No
MsgBizIdentifier	Business unique identifier assigned by the sender. For business messages it is a copy of the message identifier transported in the payload. For *Notify* and *TechnicalAck*: it refers to the original *SendRequest message*. Allowed pattern is: [0-9a-zA-Z\-\?:\(\)\.,'\+](/?([0-9a-zA-Z\-\?:\(\)\.,'\+]/?)*[0-9a-zA-Z\-\?:\(\)\.,'\+]+)? Max length: 35	In all types of message	Yes	No
MsgNetworkIdentifier	Message unique identifier assigned by the NSP. For *Notify* and *TechnicalAck*: it refers to the original *SendRequest message*	*ReceiveIndication,* *Notify,* *TechnicalAck*	Yes	Yes
FileName	Full path to the file to be sent to the receiver. Max length: 256	*FileSend*	Yes	No
FileDigest	Digest of file used in HMAC. Encoded in base64. Max length: 45	*FileSend*	Yes	No

CompressionAlgo	Algorithm used for compression. "NONE" ⇔ no compression, "ZIP" ⇔ compressed using ZIP algorithm. Allowed values are: "NONE", "ZIP". Optional. Max length: 4	*FileSend*	Yes	No
PDMFlag	Possible duplicate message flag. See note below. ~~-~~Optional for *FileSend.* Allowed values are: 'Y' and 'N'. Max length: 1	*SendRequest, ReceiveIndication.* *FileSend*	Yes	No
SignatureRequired	Flag asking for signature to be added to the message. Allowed values are: 'Y' and 'N'. Max length: 1	*SendRequest*	Yes	No
NotificationRequired	Requires a *Notify*. "A" ⇔ always, "N" ⇔ "Never", "E" ⇔ only in case of errors. Allowed values are: 'N', 'E' and 'A'. Max length: 1	*SendRequest, FileSend*	Yes	No
TechnicalAckRequired	Requires a *TechnicalAck*. "A" ⇔ always, "N" ⇔ "Never", "E" ⇔ only in case of errors. Allowed values are: 'N', 'E' and 'A'. Max length: 1	*SendRequest*	Yes	No
SignatureAddInfo	Additional information included in the signature calculation (optional). These information are only stored by TIPS platform for NRO purposes. ~~(maximum length 300 bytes)~~ Max length: 400, enforced	*ReceiveIndication*	Yes	Yes
AdditionalInfo	Additional information provided by the platform to the actor.	*SendRequest*	Yes	Yes

	It contains multiple key/value pairs, encoded in JSON. The key/value pairs TIPS currently includes are: "seq":n, - where n is the unique sequence number TIPS assigns to each operation, "bd":"YYYY-MM-DD" - where YYYY-MM-DD is the RTGS business day when the settlement took place. Other pairs can be added by TIPS at any moment. Optional. Max length: 300			
PrimitiveReturnCode	Return code of the TIPS primitive The return codes are: OK (if successful) KO (in case of failures). Allowed values are: 'OK' and 'KO' Max length: 2	*TechnicalAck, Notify*	Yes	No
PrimitiveReasonCode	Reason code of the TIPS primitive The following list of reason codes is not exhaustive and some others can be added by NSPs: TIPS.UnknownHMACKeyId TIPS.UnknownHMAC2KeyId TIPS.InvalidHMAC TIPS.MissingProperty.<Property> TIPS.InvalidProperty.<Property> TIPS.FailedDelivery TIPS.FileNotFound NSP errors are prefixed by the NSP id and agreed with TIPS platform.	*TechnicalAck, Notify*	Yes	No

	Optional, it must be necessarily present if *PrimitiveReturnCode* = "KO". Max length: 256			

From a technical point of view, note that for IBM MQ all the properties will be inserted into the RFH2 part of the message.

Note for *PDMFlag* property:

- For the *FileSend* primitive the *PDMFlag* property is provided by the TIPS platform under the following condition:
  TIPS performs retries of *FileSend* for which no Notify has been received within a certain amount of time.
  This flag, in the context of the *FileSend* operation, is used to indicate whether the operation is a retry (PDMFlag=Y) or not (PDMFlag=N).
  In the context of the *SendRequest* the PDMFlag is currently always set to N by the TIPS platform.

3.1.1. HMAC check strings

All messages are subject to **Local AUtentication (LAU)** via message authentication code computed with a symmetric key (HMAC).

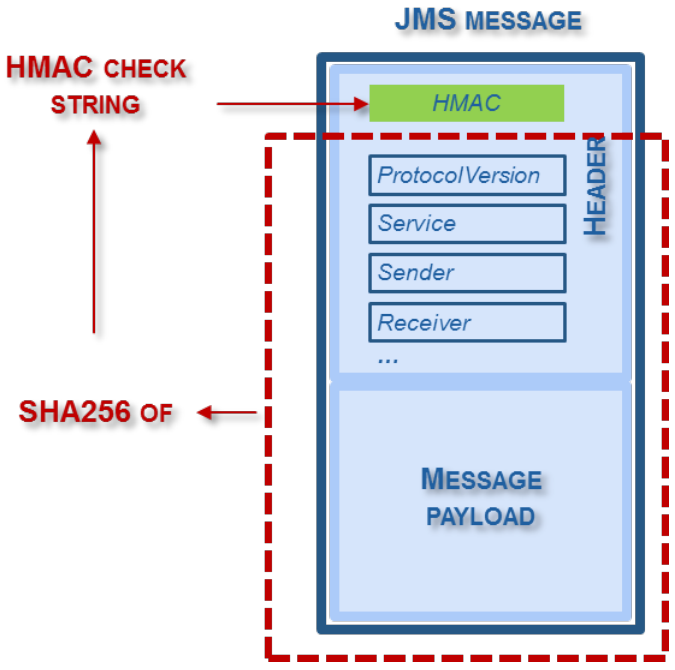


FIGURE 5 – CALCULATION OF HMAC FOR LOCAL AUTHENTICATION

The HMAC code is calculated using the header properties listed as specified in last column in the list in the previous paragraph plus the full payload (plus the HMAC symmetric key as required by HMAC technique).

The properties values, with no names or delimiters, shall be concatenated in the order of the list above, using the actual values and suppressing all trailing blanks.

The HMAC2 code is optional, it is calculated and added to the header only when the TIPS platform is sending a message (*SendRequest* primitive) that must be signed by NSP gateway (*SignatureRequired* = yes). Additionally, the NSP gateway control application must have communicated in advance to the TIPS platform this additional key and its Key Id.

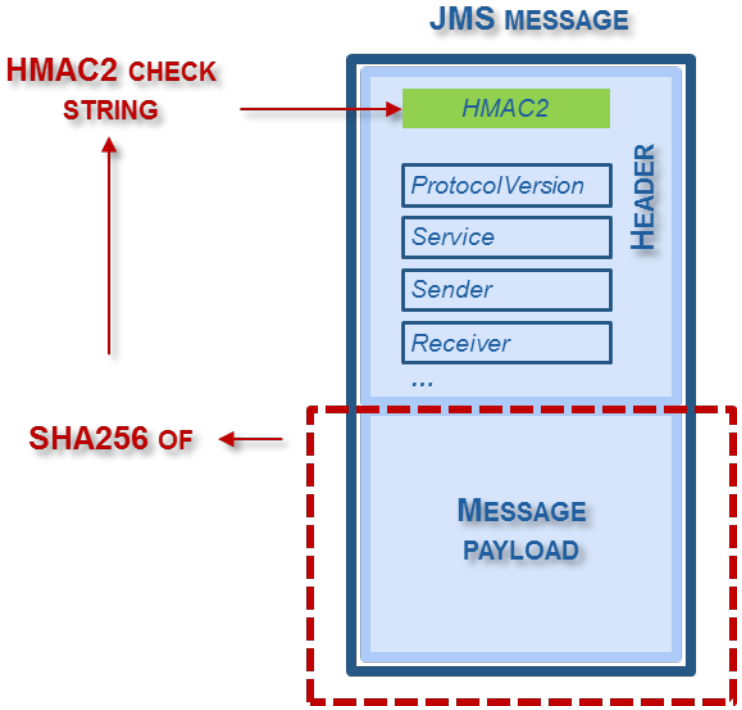


FIGURE 6 – CALCULATION OF HMAC2

The HMAC2 code is calculated using only the message payload (with no header information).

The assignment and renewal for the symmetric keys is described in paragraph dedicated to LAU. The hash function used for HMAC calculation is SHA256. The HMAC code is computed by TIPS when sending a message and passed to the NSP gateway in the HMAC and HMAC2 fields of the message header. The NSP is responsible for checking these hashes and to ensure they refer to the message transported and to last two symmetric keys previously exchanged (in order to smoothly manage the renewal of the symmetric key).

For incoming messages, gateway is responsible for computing and adding HMAC field and TIPS checks this hash to ensure it refers to the message payload transported and to one of the last two symmetric keys previously exchanged.

The HMAC an HMAC2 fields are encoded as a base64 value.

3.2. Gateway-backend channel security (LAU)

The Local authentication between TIPS and NSP gateways provides both message integrity and authentication. Optionally, the communication beyond the NSP gateway can be protected by a further HMAC code (HMAC2).

HMAC must be calculated and provided to the other side. Calculation of the HMAC starts from the message payload, part of the header and the symmetric keys stored on both sides.

HMAC2, when used, must be calculated and provided from TIPS platform to the NSP. Calculation of the HMAC2 starts from the message payload and the dedicated symmetric keys stored on both sides.

The receiver repeats the calculation using the key already owned and checks that the HMAC provided in the header is equal to the value just calculated.

The symmetric keys specifically identified within the message header must be used for HMAC/HMAC2 verification. During the asynchronous renewal of a key both the Gateway and the application will update the Key-Id to store the reference to the currently valid symmetric key.

Calculation and renewal of the symmetric key(s) is done by the **gateway control application** and communication to TIPS and to gateway must use a secure technique to protect key exchange and to prevent disclosure.

Symmetric key length must be 160 bits minimum.

The key renewal process starts with gateway control application calculating an integer value for the key and assigning to this key a Key-Id. Then the process invokes synchronously a specific TIPS component (Java method invocation) to pass the values for the key and Key-Id. Finally, the Gateway and the TIPS application starts using the new key by using the new Key-Id on each message sent to the counterparty.

The steps of the key renewal process are summarized in the following list:

1. The gateway control application triggers the key renewal functions
2. The gateway control application calculates the new key and assigns a new id
3. The gateway control application communicates the new key and its id to the gateway(s) in a secure way; The NSP gateway stores the new key internally but it doesn't start using it yet
4. The gateway invokes the TIPS component to pass the new key and the id
5. From this moment (just after TIPS component has been successfully invoked) the NSP gateway and TIPS can use the new key when sending message. The old key is still valid and it can be used when sending messages and it must be considered as valid when receiving messages.

3.2.1. Signature

Signature is expected on all business messages incoming to TIPS and in some outgoing messages sent by TIPS.

The following figures shows the messages signed in case of SCT-Inst payment[1]:

a. Signature by Originator participant of the input pacs.008 (number 2 below) payment orders sent to TIPS;

b. Signature by TIPS of the output pacs.008 (message 3 below) – message forwarded to the beneficiary participant;

c. Signature by the Beneficiary participant of the input authorization by the Beneficiary participant pacs.002 (number 4 below);

d. No signature on confirmation messages sent by TIPS (pacs.002 number 5 and 7) sent to the originator participant and beneficiary participant as final confirmation.

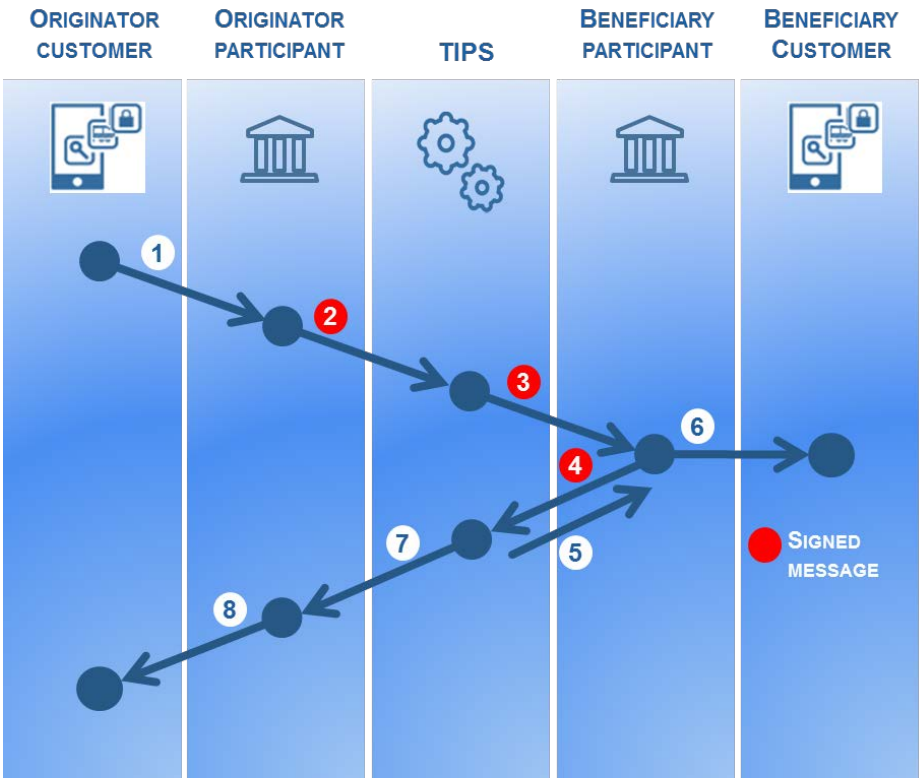


FIGURE 7 – SIGNATURES

[1] In any case, as far as MEPT is concerned, the decision to sign a message sent by TIPS platform is instructed by the SignatureRequired field in the header.

The message payload and, optionally, some of the header properties are signed and the signature is included in the message header.

The header properties that the NSP can use when calculates the signature are those that TIPS platform stores for Non-Repudiation of Origin (NRO) specified in the header description (cfr 3.1). The NSP can include these values into the signature calculation or it can use only some of them or decide to sign the payload only, in any case TIPS platform stores all these signature-eligible properties for in order to make possible to perform, on demand, a re-calculation of the signature.

In case the NSP signature requires additional fields not included in the header, NSP can provide the needed values to TIPS using the *SignatureAddInfo* header property; this optional field is stored by TIPS for NRO purposes.

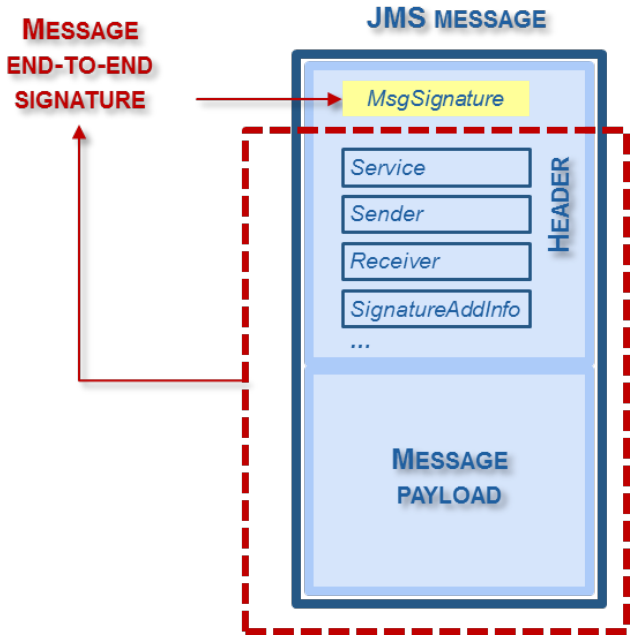


FIGURE 8 – MESSAGE SIGNATURE

In outgoing communication, the signature is added by the NSP gateway on behalf of TIPS, using the TIPS private key.

In incoming communication, the signature has to be added by the Actor using a NSP certificate and the check of its validity is performed, at TIPS side, by the NSP gateway on behalf of TIPS.

The NSP will put in place all the necessary activities related to the digital signature, e.g. signing, verification of signature, check against directory services (such as CRL and/or CSL).

The certificates used are issued by the NSP PKI for both outgoing and incoming cases and belong to a specific certificate class with a strong level of authentication and non-repudiation. The validity period of these certificates is 24 months.

3.3. Message payload

The message payload transported by the MQ message contains information that depends on the MEPT primitive the message refers to. The following table specifies the information transported by the message payload.
In case the primitive refers to a business message (ISO message), no further information shall be included in such a payload other than the business message itself.

TIPS primitive	**Payload content**
SendRequest	The ISO message to be sent by TIPS, e.g. pacs.008, pacs.002.
ReceiveIndication	The ISO message received from TIPS, e.g. pacs.008, pacs.002.
TechnicalAck	Empty payload. Other relevant information for *TechnicalAck* are stored in the header (e.g. message identifiers, PrimitiveReasonCode)
Notify	Empty payload. Other relevant information for *Notify* are stored in the header (e.g. MsgSignature, message identifiers, PrimitiveReasonCode)
FileSend	Empty payload

3.4. MQ MD descriptor usage

The following table shows fields used in the MQ MD:

MQ MD field	**SendRequest / ReceiveIndication / FileSend**	**Notify / Technical Ack**
MQMD.MsgType	DATAGRAM	REPORT
MQMD.Format	MQFMT_RF_HEADER_2~~MQFMT_NONE~~	MQFMT_RF_HEADER_2~~MQFMT_NONE~~
MQMD.MsgId	Present Used as CorrelId for Notify/TechnicalAck	Present

MQ MD field	SendRequest / ReceiveIndication / FileSend	Notify / Technical Ack
MQMD.CorrelId	Absent	Equal to MQMD.MsgId of the message it corresponds to (only for Notify)
MQMD.ReportOption	0	MQRO_PAN for positive Notify/TechnicalAck MQRO_NAN for negative Notify/TechnicalAck
MQMD.Feedback	MQFB_NONE	0 for MQRO_PAN 1000 for MQRO_NAN 2000 for MQRO_NAN when garbage was received
MQMD.CodedCharSetID	1208 (UTF-8)	1208 (UTF-8)
MQMD.Expiry	MQEI_UNLIMITED	MQEI_UNLIMITED at the beginning. It will be possible to set the property assigning a value that will prevent the TIPS Platform from being overwhelmed by “old” messages that will be timed out.
MQMD.ApplIdentityData	Not used – ignored	Not used -ignored
MQMD.Encoding	MQENC_NATIVE	MQENC_NATIVE
MQMD.ReplyToQ	Empty	Empty
MQMD.ReplyToQMgr	Empty	Empty
MQMD.AccountingToken	MQACT_NONE	MQACT_NONE
MQMD.Persistence	MQPER_NOT_PERSISTENT	MQPER_NOT_PERSISTENT

The use of the MQMD.CorrelId is only needed in case it is not possible to generate a *Notify* or *Technical Ack* that contains the RFH2 field MsgBizIdentifier to be used for reconciliation of the *Notify* or *Technical Ack*.

In this case the only way to reconcile is to use the MQMD.CorrelId. To indicate that this is the case the MQMD.Feedback has the value 2000.

The fields in the RFH2 for such Notify or Technical Ack would not include any unknown fields but would indicate what fields are filtered out in the ReasonCode.

3.4.1. MQ MD Descriptor and Payload Format

As a general advice, the format of the message payload should preferably be binary (i.e. BytesMessage, in JMS terminology) instead of text (i.e.TextMessage, in JMS terminology), for performance reasons. Even if the TIPS platform is able to handle both formats, binary is preferred, because it allows to avoid a few data conversion operations and is more memory efficient.

The value of **MQMD.MsgId** is used as CorrelId only for Notify.

The **MQMD.CodedCharSetID** is the CCSID of the RFH2 header. CCSSID of message payload (in MQ message data after the RFH2 section) is defined by RFH2.CodedCharSetId and must be 1208 (UTF-8).

The value of **MQMD.ApplIdentityData**, **MQMD.AccountingToken** can be set only if authority SET_IDENTITY_CONTEXT is set: no. Currently those fields cannot be set.

The value of fields **MQMD.ReplyToQ and MQMD.ReplyToQMg** is currently unused.

3.5. MQ queues, MQ channels and affinity

There is a set of queues containing SendRequest, a set of queues containing ReceiveIndication, a set of queues containing SendFile and a set of queues containing Notify and TechnicalAck.
It is possible to configure the same queue used for ReceiveIndication to be used for Notify and TechnicalAck.
The set of queues can be over multiple Queue Managers running on different hosts. Each NSP gateway will establish MQI connections to each of the Queue Managers.

There is no affinity between SendRequest queues and queues containing Notify and TechnicalAck. It is possible that a Notify is put on a queue of a different Queue Manager than the Queue Manager of the queue from which the request was taken.
The Queue Manager is dedicated for a given Service.

3.6. Examples

Instant payment request from the originator

```
<rfh2>
   <HMAC>dGhpcyBpcyBub3QgYSBzaWduYXR1cmUK…</HMAC>
   <HMACKeyId>1234</HMACKeyId>
   <MsgSignature>
```

```
      <Signature ...
      </Signature>
   </MsgSignature>
   <ProtocolVersion>1</ProtocolVersion>
   <Service>TIPS-TEST</Service>
   <Sender>cn=originator-dn,ou=...,o=...</Sender>
   <Receiver>cn=tips-dn,ou=...,o=...</Receiver>
   <PrimitiveType>ReceiveIndication</PrimitiveType>
   <MsgType>pacs.008.001.02</MsgType>
   <SendTimestamp>2016-12-19T12:00:01.222Z</SendTimestamp>
   <ReceiveTimestamp>2016-12-19T12:00:01.777Z</ReceiveTimestamp>
   <MsgBizIdentifier>MSG001</MsgBizIdentifier>
   <MsgNetworkIdentifier>NWX000001</MsgNetworkIdentifier>
</rfh2>
<Document                                    xmlns="urn:iso:std:iso:20022:tech:xsd:pacs.008.001.02"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:iso:std:iso:20022:tech:xsd:pacs.008.001.02">
<FIToFICstmrCdtTrf>
                <GrpHdr>
                        <MsgId>MSG001</MsgId>
                        <CreDtTm>2016-12-19T12:00:01.222Z</CreDtTm>
                        <NbOfTxs>1</NbOfTxs>
                        <TtlIntrBkSttlmAmt Ccy="EUR">123.45</TtlIntrBkSttlmAmt>
                        <IntrBkSttlmDt>2016-12-19</IntrBkSttlmDt>
                        <SttlmInf>
                                <SttlmMtd>INDA</SttlmMtd>
                        </SttlmInf>
                        <PmtTpInf>
                                <SvcLvl>
                                        <Cd>SEPA</Cd>
                                </SvcLvl>
                                <LclInstrm>
                                        <Cd>INST</Cd>
                                </LclInstrm>
                        </PmtTpInf>
                </GrpHdr>
                <CdtTrfTxInf>
                        <PmtId>
                                <EndToEndId>ENDTOEND001</EndToEndId>
                                <TxId>TRX001</TxId>
                        </PmtId>
                        <IntrBkSttlmAmt Ccy="EUR">123.45</IntrBkSttlmAmt>
                        <AccptncDtTm>2016-12-19T12:00:01.222Z</AccptncDtTm>
                        <ChrgBr>SLEV</ChrgBr>
                        <Dbtr>
                                <Nm>BANK NAME DEBTOR</Nm>
                        </Dbtr>
                        <DbtrAcct>
                                <Id>
                                        <IBAN>IT039283584360549</IBAN>
                                </Id>
                        </DbtrAcct>
                        <DbtrAgt>
```

```
                        <FinInstnId>
                            <BIC>BANKAABBXXX</BIC>
                        </FinInstnId>
                    </DbtrAgt>
                    <CdtrAgt>
                        <FinInstnId>
                            <BIC>BANKBBBBXXX</BIC>
                        </FinInstnId>
                    </CdtrAgt>
                    <Cdtr>
                        <Nm>BANK NAME CREDITOR</Nm>
                    </Cdtr>
                    <CdtrAcct>
                        <Id>
                            <IBAN>DE045984568540</IBAN>
                        </Id>
                    </CdtrAcct>
                </CdtTrfTxInf>
            </FIToFICstmrCdtTrf>
</Document>
```

Instant payment request validated and to be forwarded to the beneficiary

```
<rfh2>
   <HMAC>dGhpcyBpcyBub3QgYSBzaWduYXR1cmUK...</HMAC>
   <HMACKeyId>1234</HMACKeyId>
   <ProtocolVersion>1</ProtocolVersion>
   <Service>TIPS-TEST</Service>
   <Sender>cn=tips-dn,ou=...,o=...</Sender>
   <Receiver>cn=beneficiary-dn,ou=...,o=...</Receiver>
   <PrimitiveType>SendRequest</PrimitiveType>
   <MsgType>pacs.008.001.02</MsgType>
   <MsgBizIdentifier>MSG001</MsgBizIdentifier>
   <SignatureRequired>Y</SignatureRequired>
   <NotificationRequired>E</NotificationRequired>
   <TechnicalAckRequired>E</TechnicalAckRequired>

</rfh2>
<Document                                    xmlns="urn:iso:std:iso:20022:tech:xsd:pacs.008.001.02"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:iso:std:iso:20022:tech:xsd:pacs.008.001.02">
<FIToFICstmrCdtTrf>
                <GrpHdr>
                    <MsgId>MSG001</MsgId>
                    <CreDtTm>2016-12-19T12:00:01.222Z</CreDtTm>
                    <NbOfTxs>1</NbOfTxs>
                    <TtlIntrBkSttlmAmt Ccy="EUR">123.45</TtlIntrBkSttlmAmt>
                    <IntrBkSttlmDt>2016-12-19</IntrBkSttlmDt>
                    <SttlmInf>
                        <SttlmMtd>INDA</SttlmMtd>
                    </SttlmInf>
                    <PmtTpInf>
                        <SvcLvl>
                            <Cd>SEPA</Cd>
```

```
                        </SvcLvl>
                        <LclInstrm>
                                <Cd>INST</Cd>
                        </LclInstrm>
                </PmtTpInf>
        </GrpHdr>
        <CdtTrfTxInf>
                <PmtId>
                        <EndToEndId>ENDTOEND001</EndToEndId>
                        <TxId>TRX001</TxId>
                </PmtId>
                <IntrBkSttlmAmt Ccy="EUR">123.45</IntrBkSttlmAmt>
                <AccptncDtTm>2016-12-19T12:00:01.222Z</AccptncDtTm>
                <ChrgBr>SLEV</ChrgBr>
                <Dbtr>
                        <Nm>BANK NAME DEBTOR</Nm>
                </Dbtr>
                <DbtrAcct>
                        <Id>
                                <IBAN>IT039283584360549</IBAN>
                        </Id>
                </DbtrAcct>
                <DbtrAgt>
                        <FinInstnId>
                                <BIC>BANKAABBXXX</BIC>
                        </FinInstnId>
                </DbtrAgt>
                <CdtrAgt>
                        <FinInstnId>
                                <BIC>BANKBBBBXXX</BIC>
                        </FinInstnId>
                </CdtrAgt>
                <Cdtr>
                        <Nm>BANK NAME CREDITOR</Nm>
                </Cdtr>
                <CdtrAcct>
                        <Id>
                                <IBAN>DE045984568540</IBAN>
                        </Id>
                </CdtrAcct>
        </CdtTrfTxInf>
    </FIToFICstmrCdtTrf>
</Document>
```

Positive response from the beneficiary

```
<rfh2>
   <HMAC>dGhpcyBpcyBub3QgYSBzaWduYXR1cmUK...</HMAC>
   <HMACKeyId>1234</HMACKeyId>
   <MsgSignature>
      <Signature ...
      </Signature>
   </MsgSignature>
```

```
    <ProtocolVersion>1</ProtocolVersion>
    <Service>TIPS-TEST</Service>
    <Sender>cn=originator-dn,ou=tips,o=...</Sender>
    <Receiver>cn=tips-dn,ou=tips,o=...</Receiver>
    <PrimitiveType>ReceiveIndication</PrimitiveType>
    <MsgType>pacs.002.001.03</MsgType>
    <SendTimestamp>2016-12-19T12:00:01.222Z</SendTimestamp >
    <ReceiveTimestamp>2016-12-19T12:00:01.777Z</ReceiveTimestamp >
    <MsgBizIdentifier>MSG002</MsgBizIdentifier>
    <MsgNetworkIdentifier>NWX000002</MsgNetworkIdentifier>
</rfh2>
<Document                                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:iso:std:iso:20022:tech:xsd:pacs.002.001.03">
<FIToFIPmtStsRpt>
                <GrpHdr>
                        <MsgId>MSG002</MsgId>
                        <CreDtTm>2016-12-19T12:00:01.222Z</CreDtTm>
                </GrpHdr>
                <OrgnlGrpInfAndSts>
                        <OrgnlMsgId>MSG001</OrgnlMsgId>
                        <OrgnlMsgNmId>pacs.008.001.02</OrgnlMsgNmId>
                        <GrpSts>ACCP</GrpSts>
                </OrgnlGrpInfAndSts>
                <TxInfAndSts>
                        <StsId>MSG001</StsId>
                        <OrgnlEndToEndId>ENDTOEND001</OrgnlEndToEndId>
                        <OrgnlTxId>TRX001</OrgnlTxId>
                        <AccptncDtTm>2016-12-19T12:00:01.222Z</AccptncDtTm>
                        <OrgnlTxRef>
                                <IntrBkSttlmAmt Ccy="EUR">123.45</IntrBkSttlmAmt>
                                <DbtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKAABBXXX</BIC>
                                        </FinInstnId>
                                </DbtrAgt>
                                <CdtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKBBBBXXX</BIC>
                                        </FinInstnId>
                                </CdtrAgt>
                        </OrgnlTxRef>
                </TxInfAndSts>
        </FIToFIPmtStsRpt></Document>
```

Instant Payment completed

Response to be sent to the originator

```
<rfh2>
    <HMAC>dGhpcyBpcyBub3QgYSBzaWduYXR1cmUK...</HMAC>
    <HMACKeyId>1234</HMACKeyId>
    <ProtocolVersion>1</ProtocolVersion>
    <Service>TIPS-TEST</Service>
    <Sender>cn=tips-dn,ou=tips,o=...</Sender>
```

```
    <Receiver>cn=originator-dn,ou=tips,o=...</Receiver>
    <PrimitiveType>SendRequest</PrimitiveType>
    <MsgType>pacs.002.001.03</MsgType>
    <MsgBizIdentifier>MSG001@00001@dbtr</MsgBizIdentifier>
    <SignatureRequired>N</SignatureRequired>
    <NotificationRequired>E</NotificationRequired>
    <TechnicalAckRequired>E</TechnicalAckRequired>
    <AdditionalInfo>{"seq":2287345,"bd":"2019-07-11"}{"RTGSDate":"2019-07-11"}</AdditionalInfo>
</rfh2>
<Document                                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:iso:std:iso:20022:tech:xsd:pacs.002.001.03">
<FIToFIPmtStsRpt>
                <GrpHdr>
                        <MsgId>MSG002</MsgId>
                        <CreDtTm>2016-12-19T12:00:01.222Z</CreDtTm>
                </GrpHdr>
                <OrgnlGrpInfAndSts>
                        <OrgnlMsgId>MSG001</OrgnlMsgId>
                        <OrgnlMsgNmId>pacs.008.001.02</OrgnlMsgNmId>
                        <GrpSts>ACCP</GrpSts>
                </OrgnlGrpInfAndSts>
                <TxInfAndSts>
                        <StsId>MSG001</StsId>
                        <OrgnlEndToEndId>ENDTOEND001</OrgnlEndToEndId>
                        <OrgnlTxId>TRX001</OrgnlTxId>
                        <AccptncDtTm>2016-12-19T12:00:01.222Z</AccptncDtTm>
                        <OrgnlTxRef>
                                <IntrBkSttlmAmt Ccy="EUR">123.45</IntrBkSttlmAmt>
                                <DbtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKAABBXXX</BIC>
                                        </FinInstnId>
                                </DbtrAgt>
                                <CdtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKBBBBXXX</BIC>
                                        </FinInstnId>
                                </CdtrAgt>
                        </OrgnlTxRef>
                </TxInfAndSts>
        </FIToFIPmtStsRpt></Document>
```

Response to be sent to the beneficiary

```
<rfh2>
    <HMAC>dGhpcyBpcyBub3QgYSBzaWduYXR1cmUK...</HMAC>
    <HMACKeyId>1234</HMACKeyId>
    <ProtocolVersion>1</ProtocolVersion>
    <Service>TIPS-TEST</Service>
    <Sender>cn=tips-dn,ou=tips,o=...</Sender>
    <Receiver>cn=beneficiary-dn,ou=tips,o=...</Receiver>
    <PrimitiveType>SendRequest</PrimitiveType>
    <MsgType>pacs.002.001.03</MsgType>
```

```
    <MsgBizIdentifier>MSG001@00001@cdtr</MsgBizIdentifier>
    <SignatureRequired>N</SignatureRequired>
    <NotificationRequired>E</NotificationRequired>
    <TechnicalAckRequired>E</TechnicalAckRequired>
    <AdditionalInfo>{"seq":2287345,"bd":"2019-07-11"}{"RTGSDate":"2019-07-11"}</AdditionalInfo>
</rfh2>
<Document                                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:iso:std:iso:20022:tech:xsd:pacs.002.001.03">
    <FIToFIPmtStsRpt>
                <GrpHdr>
                        <MsgId>MSG002</MsgId>
                        <CreDtTm>2016-12-19T12:00:01.222Z</CreDtTm>
                </GrpHdr>
                <OrgnlGrpInfAndSts>
                        <OrgnlMsgId>MSG001</OrgnlMsgId>
                        <OrgnlMsgNmId>pacs.008.001.02</OrgnlMsgNmId>
                        <GrpSts>ACCP</GrpSts>
                </OrgnlGrpInfAndSts>
                <TxInfAndSts>
                        <StsId>MSG001</StsId>
                        <OrgnlEndToEndId>ENDTOEND001</OrgnlEndToEndId>
                        <OrgnlTxId>TRX001</OrgnlTxId>
                        <AccptncDtTm>2016-12-19T12:00:01.222Z</AccptncDtTm>
                        <OrgnlTxRef>
                                <IntrBkSttlmAmt Ccy="EUR">123.45</IntrBkSttlmAmt>
                                <DbtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKAABBXXX</BIC>
                                        </FinInstnId>
                                </DbtrAgt>
                                <CdtrAgt>
                                        <FinInstnId>
                                                <BIC>BANKBBBBXXX</BIC>
                                        </FinInstnId>
                                </CdtrAgt>
                        </OrgnlTxRef>
                </TxInfAndSts>
        </FIToFIPmtStsRpt></Document>
```

Technical Ack received on Response sent to Beneficiary

```
<rfh2>
    <HMAC>odinf90sUSKDoUSLLio5S4d5VdWpoad4...</HMAC>
    <HMACKeyId>1234</HMACKeyId>
    <ProtocolVersion>1</ProtocolVersion>
    <Service>TIPS-TEST</Service>
    <Sender>cn=tips-dn,ou=tips,o=...</Sender>
    <Receiver>cn=beneficiary-dn,ou=tips,o=...</Receiver>
    <PrimitiveType>TechnicalAck</PrimitiveType>
    <SendTimestamp>2016-12-19T12:00:01.222Z</SendTimestamp>
    <MsgBizIdentifier>MSG001@00001@cdtr</MsgBizIdentifier>
    <MsgNetworkIdentifier>NWX000005</MsgNetworkIdentifier>
    <PrimitiveReturnCode>KO</PrimitiveReturnCode>
    <PrimitiveReasonCode>TIPS.FailedDelivery</PrimitiveReasonCode>
```

```
</rfh2>
```