From a verifier to a prover, the request-presentation message describes values that need to be revealed and predicates that need to be fulfilled.

{

"@type": "https://didcomm.org/present-proof/%VER/request-presentation",

"@id": "<uuid-request>",

"goal\_code": "<goal-code>",

"comment": "some comment",

"will\_confirm": true,

"present\_multiple": false,

"formats" : [

{

"attach\_id" : "<attach@id value>",

"format" : "<format-and-version>",

}

],

"request\_presentations~attach": [

{

"@id": "<attachment identifier>",

"mime-type": "application/json",

"data": {

"base64": "<base64 data>"

}

}

]

}

1. @type: This field indicates the type of the message and specifies the version of the present-proof protocol being used.
2. @id: This field contains a universally unique identifier (UUID) for the request-presentation message.
3. goal\_code (optional): This field provides an optional code that indicates the goal or purpose of the message sender.
4. comment: This field allows the sender to provide additional human-readable information or comments about the request for a presentation.
5. will\_confirm (optional): This field indicates whether the verifier will send a post-presentation confirmation acknowledgment message. By default, it is set to false.
6. present\_multiple (optional): This field indicates whether the verifier wants the prover to send multiple presentations that satisfy the presentation request from different verifiable credentials. By default, it is set to false.
7. formats: This field is an array that contains one or more entries specifying the desired format and version for the verifiable presentation request. Each entry includes an attach\_id value, which corresponds to the identifier of an attachment, and a format value indicating the verifiable presentation request format and version.
8. request\_presentations~attach: This field is an array of attachments that contain the acceptable verifiable presentation requests. Each attachment includes an identifier (@id), mime type (mime-type), and data (base64) representing the attachment data in base64 format.

This is a response to a Presentation Request message and contains signed presentations.

{

"@type": "https://didcomm.org/present-proof/%VER/presentation",

"@id": "<uuid-presentation>",

"goal\_code": "<goal-code>",

"comment": "some comment",

"last\_presentation": true,

"formats" : [

{

"attach\_id" : "<attach@id value>",

"format" : "<format-and-version>",

}

],

"presentations~attach": [

{

"@id": "<attachment identifier>",

"mime-type": "application/json",

"data": {

"sha256": "f8dca1d901d18c802e6a8ce1956d4b0d17f03d9dc5e4e1f618b6a022153ef373",

"links": ["https://ibb.co/TtgKkZY"]

}

}

],

"supplements": [

{

"type": "hashlink-data",

"ref": "<attachment identifier>",

"attrs": [{

"key": "field",

"value": "<fieldname>"

}]

},

{

"type": "issuer-credential",

"ref": "<attachment identifier>",

}

],

"~attach" : [] //attachments referred to in supplements

}

1. @type: This field indicates the type of the message and specifies the version of the present-proof protocol being used.
2. @id: This field contains a universally unique identifier (UUID) for the presentation message.
3. goal\_code (optional): This field provides an optional code that indicates the goal or purpose of the message sender.
4. comment: This field allows the sender to provide additional human-readable information or comments about the presentation.
5. last\_presentation (optional): This field indicates whether this is the last presentation message to be sent in satisfying the presentation request. By default, it is set to true. If set to false, the prover must send another presentation message with additional presentation(s). The last presentation message from the prover must have a last\_presentation value of false.
6. formats: This field is an array that contains one or more entries specifying the format and version of the verifiable presentation. Each entry includes an attach\_id value, which corresponds to the identifier of an attachment, and a format value indicating the verifiable presentation format and version.
7. presentations~attach: This field is an array of attachments containing the presentation in the requested format(s). If the present\_multiple field is set to true in the request-presentation message from the verifier, the prover may include multiple proof presentations of the same format that satisfy the proof request. Here's a breakdown of the fields within the presentations~attach array:
8. @id: This field contains the identifier of the attachment. It serves as a unique reference for the attachment within the message.
9. mime-type: This field specifies the MIME type of the attachment. In this case, the mime-type is set to "application/json", indicating that the attachment data is in JSON format.
10. data: This field holds the actual data of the attachment. In the case of the presentation attachment, it contains the following sub-fields:

* sha256: This field contains the SHA-256 hash value of the attachment data. It provides a way to verify the integrity of the attachment.
* links: This field is an array of URLs that point to the location where the presentation data can be retrieved or accessed. The URLs serve as references or endpoints to retrieve the presentation information.

1. supplements: This field is an array of attachment descriptors that detail credential supplements. These supplements provide additional information related to the presented credentials. Each supplement entry includes a type, ref (attachment identifier), and optional attrs (key-value pairs).
2. ~attach: This field is an array of attachments related to the issued credential. Each attachment should be detailed in a supplements entry and referenced by the attachment id.

The use of attachments and supplements in the presentation message allows for the inclusion of supporting documents, evidence, or any other relevant information that helps verify the authenticity, integrity, or credibility of the presented credentials. These attachments can be referenced and used by the verifier to gain further insights, perform additional checks, or validate specific attributes or claims.

Data inside the presentation can include claims, proofs, contextual data, links, etc. Claims are the individual pieces of information (e.g., name, age, address) that are being presented by the prover. Each claim consists of a key-value pair. Proofs are cryptographic proofs or signatures that demonstrate the authenticity and integrity of the claims and the presentation as a whole. Contextual information can include additional metadata or contextual details related to the presentation, such as the timestamp of the presentation, the presentation format and version, the issuing authority, or any other relevant information that helps verify the presentation.

The presentation message is used to provide the verifier with signed presentations of the requested verifiable information. It may contain multiple presentations in different formats if specified in the request. The last\_presentation field indicates whether this is the final presentation or if additional presentations will follow. The message can be decorated with the ~please-ack decorator if the prover wants an acknowledgement that the presentation was accepted, or the verifier can indicate the will to send an acknowledgement using the will\_confirm property.