PassBy[ME] API Documentation

Document id:	PBM_01
Document Version:	1.1.12
Author:	Microsec Ltd.
Date:	2015.09.13.
API Version	**1**

Table of contents

1 Introduction

The PassBy[ME] message delivery service lets the administrator of a registered organization to send arbitrary messages –using the same infrastructure as the authentication service- to *their* users. Similarly to the authentication, this sends a notification to the user's PassByME ready smartphone, then the recipients receive the message in exchange to an Electronic Proof of Delivery evidence.

A message consists of the following parts: recipients, subject, body and an expiration date.

The message delivery service can be used either via with the web based administration interface or with the message delivery API. Sending a message using the API requires an Application PFX.

2 Terms

The document uses the following PassBy[ME] specific terms:

Administrator: A human, or machine with access to the Administration interface or the Management API, being able to administer the Organization.

Alias: In the PassBy[ME] system a user can have multiple PassBy[ME] IDs. These IDs are all unique identifiers of the user within the Organization serving as his aliases.

Administration interface: Web based client for administrative operations.

Authentication API: Programmable webservice interface for authentication related operations.

Application: An external application that uses PassBy[ME] as a second factor.

Organization: The registration unit of the PassBy[ME] system. Encapsulates everything related to a registered parties PassBy[ME] operations, like Administrators, Applications, Users etc.

PassBy[ME] App: The PassBy[ME] mobile application available for iOS and Android in the application stores.

PassBy[ME] ID: A string which is unique to the Organization and identifies a user.

PassBy[ME] ready: A device is PassBy[ME] ready if it has the PassBy[ME] App installed and went through the Enrollment process successfully.

User: Someone who is registered within the Organization willing to use PassBy[ME] as a second factor for authentication while accessing an Application.

UserId: In the context of a User: see Alias. In the context of an Administrator, the UserId is the unique identifier of the Administrator within the Organization. Its value is the email address the Administrator is registered with.

3 PassBy[ME] message delivery API

The REST (Representational State Transfer) API provides programmatic access to the PassBy[ME] message delivery service. The REST API identifies accounts and applications by their certificates. Each application has a certificate-key pair for authentication.

You'll need a PassBy[ME] account and an application certificate to integrate your application into PassBy[ME]. To get this account data do the following steps:

1. Sign up for a PassBy[ME] test account using the web based administration interface, and register a new organization if you haven't registered before. If you have already registered into the PassBy[ME] service, please log in as an administrator with your previously created credential data (username/password and your PassBy[ME] ready mobile device).
2. Create a new Application under this organization by selecting "Register new Application" under the "Application" navigation menu. This will generate an authentication certificate for your application. Download and save the authentication certificate and the key by clicking on the suitcase icon ("Key Details") next to your application then "Download PFX" under the "Application" navigation menu. Save the PFX's password after clicking the „Show PFX password" button.
3. Create at least one user by selecting "Register new User" under the "Users" navigation menu.

Client certificate authentication is enforced by the server. In order to use the API you are required to use your application's authentication certificate during all API calls.

3.1 Access to the PassBy[ME] service

Web based administration interface: https://admin.passbyme.com

Authentication and Message Delivery API webservice: https://auth-sp.passbyme.com/frontend

Management API webservice: https://api.passbyme.com/register

3.2 Response format

Responses are available in JSON (JavaScript Object Notation) format.

- On success the system returns Objects represented in JSON format.
- PassBy[ME] returns HTTP 420 error code on any error. Error responses are objects represented in JSON format. Error responses contain a *code* that is always present and an optional *message*.

  General error object:

  ```
  { "code" : <error_code>, "message" : <error_message> }
  ```

3.2.1 Examples

A successful result.

```
HTTP/1.0 200 OK
{
   "messageId" : "YzX95zUA1et2ijQ",
   "expirationDate" : "2015-06-11T13:06:12.658+02:00",
   "recipients" : {
       "pbmId1" : "PENDING"
   }
}
```

Error result:

```
HTTP/1.0 420
{ "code" : "MALFORMED_INPUT", "message" : "Missing recipients" }
```

3.3 Authentication

To access the PassBy[ME] service URL you should have a **valid authentication certificate and key (PFX file):** The PassBy[ME] authentication service URL can be used only with a valid authentication certificate, which you can acquire after registering your application under your organization. To download the certificate do the following steps:

1. Go to the 'Applications' menu.
2. Click on the suitcase icon ("Key Details") next to your application.
3. Choose the 'Download PFX' option.

This certificate has a software based private key inside the downloaded PFX. The PFX file is protected with a passphrase, which can be printed on the administration website.

Important: The authentication certificate identifies the registered application under the organization.in the PassBy[ME] service.

3.4 Parameters in URL path

Some input parameters of the PassBy[ME] REST interface are included in the request path. These parameters are distinguished from other path values by a preceding colon. For example:

```
PUT /rest/method/with/a/:parameter/and/an/:otherparameter
```

contains two parameters: "parameter" and "otherparameter".

3.5 API Version

The PassBy[ME] service requires the client to declare its supported API version. To do that, the client has to include the **X-PBM-API-VERSION** custom HTTP header into **every request** with the value of its current API version. The current API version this document is compatible with can be found on the cover of this document.

Example:

```
X-PBM-API-VERSION: 1
```

4 Message delivery API

4.1 Sending message

METHOD
POST /messages

This sends a message to the smartphones of the given users. The request returns the created message including its messageId. This messageId must be used when calling the GET /messages/:messageId (**Error! Reference source ot found.**) method.

4.1.1 Input parameters

The input must be POSTed in JSON format as a request body. The JSON object consists of the following keys.

JSON key	**Description**	
recipients	A JSON Array of PassBy[ME] IDs	*Required*
subject	A JSON String that contains the subject. Maximum size is 254 characters.	*Required*
body	A JSON String that contains the message body. Maximum size is 4094 characters.	*Required*
availability	A JSON integer that denotes the availability of the message in seconds.	*Required*
type	Message type: • "authorization" in case of user authorization request • "message" in case of general message • "esign" in case of esign request (since device API version 3)	*Required*

4.1.2 Output

Outputs the created message, or one of the following errors:

- CERTIFICATE_REVOKED - client certificate has been revoked.
- FORBIDDEN - if your account is not authorized for this operation.
- SUBSCRIPTION_EXPIRED – subscription has been expired.
- MALFORMED_INPUT – the request is not well formed

4.1.3 Example request

```
POST:  /messages
{
  "recipients" : [ "pbmID1", "pbmID2", "pbmID3" ],
  "subject" : "example message subject",
  "body" : "example message body"
  "availability" : 300,
  "type" : "message"
}
```

4.1.4 Example Response

```
HTTP/1.0 200 OK
{
   "messageId" : " YzX95zUA1et2ijQ",
   "expirationDate" : "2015-06-11T13:06:12.658+02:00",
   "recipients" : [
       { "userId" : "pbmId1", "status" : "PENDING" },
       { "userId" : "pbmId2", "status" : "PENDING" },
       { "userId" : "pbmId3", "status" : "PENDING" }
   ]
}
```

or

```
HTTP/1.0 420
{ "code" : "MALFORMED_INPUT", "message" : "Missing recipients" }
```

4.2 Tracking the message

METHOD
GET /messages/:messageId

This returns the status of a message, identified by the given messageId. The messageId is obtained by calling the POST /message (**Error! Reference ource not found.**) method.

4.2.1 Input parameters

The input must be the path parameter of the GET request.

- messageId - The ID of the message to be tracked

4.2.2 Output

On success it returns the status of the message transaction. Possible values are:

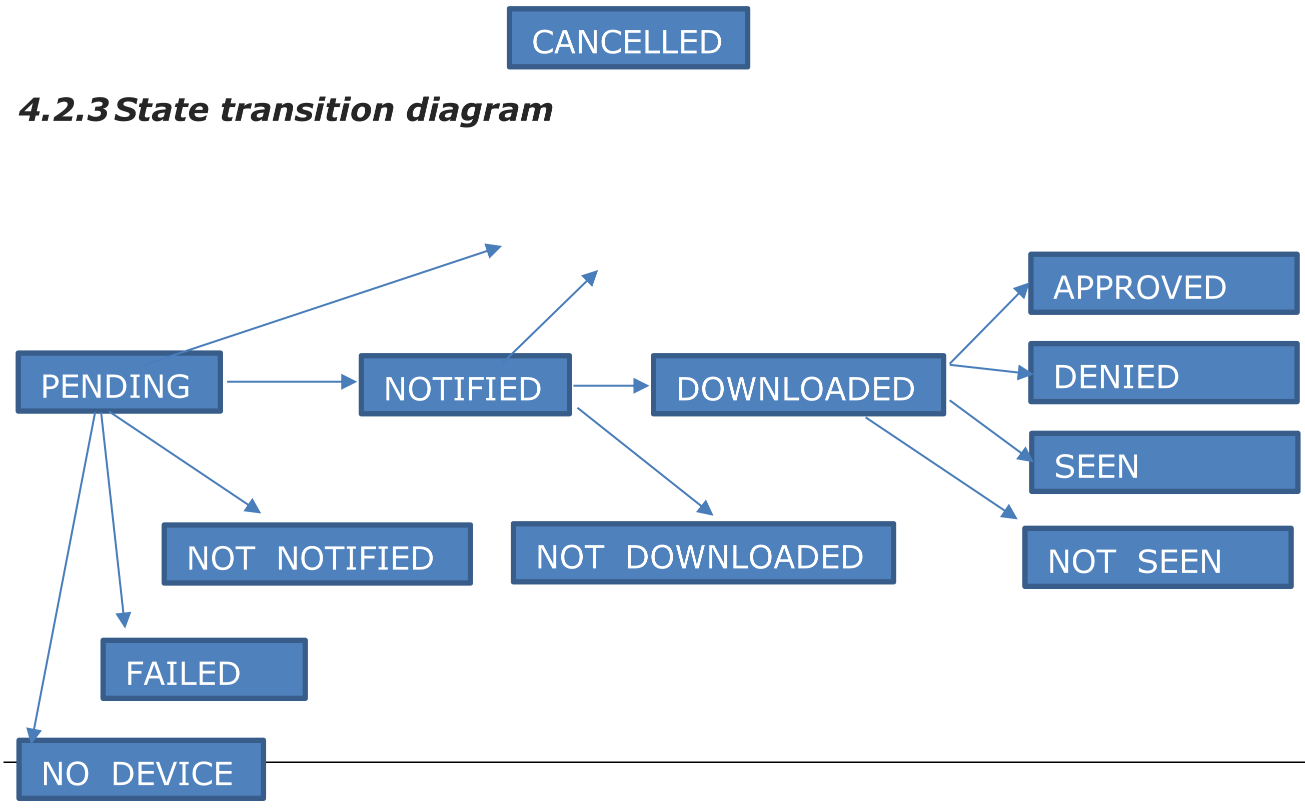
JSON key	Description	
recipients	A JSON Array with status of the recipients	*Required*
messageId	The ID of the message	*Required*
expirationDate	Expiration date of the message. ISO8601 format.	*Required*

Available statuses

Status	
PENDING	Initial status of the message.
NOTIFIED	The recipient has been notified about a new message.
DOWNLOADED	The recipient has downloaded the message, but has not uploaded the evidence yet.
SEEN	The recipient has seen the message and uploaded the evidence.

NOT_SEEN	The recipient has not seen the message.
NOT_NOTIFIED	The recipient has not received the notification.
NOT_DOWNLOADED	The recipient received the notification about the message but has not downloaded the message
NO_DEVICE	The message could not be sent because the recipient had no PassBy[ME] ready device that supports messaging.
FAILED	The message could not be sent because of an error.
DISABLED	The message could not be sent because the recipient is disabled.
CANCELLED	The message was cancelled by the sender.
APPROVED	Authentication has finished successfully.
DENIED	The user has cancelled the authentication.

4.2.3 State transition diagram



On error it returns one of the following error codes:

- FORBIDDEN - if your account is not authorized for this operation.
- MALFORMED_INPUT - if the input is not well formed, or invalid
- NOT_FOUND - if the given message was not found.
- CERTIFICATE_REVOKED - client certificate has been revoked.

4.2.4 Example Request

```
GET: /messages/YzX95zUA1et2ijQ
```

4.2.5 Example Response

```
HTTP/1.0 200 OK
{
 "messageId" : " YzX95zUA1et2ijQ",
 "expirationDate" : "2015-06-11T13:06:12.658+02:00",
 "recipients" : [
   { "userId" : "pbmId1", "status" : "PENDING" },
   { "userId" : "pbmId2", "status" : "NOTIFIED" },
   { "userId" : "pbmId3", "status" : "SEEN" }
 ]
}
```

or

```
HTTP/1.0 420
{ "code": "NOT_FOUND", "message" : "No such message" }
```

4.3 Cancelling the message

METHOD
DELETE /messages/:messageId

This cancels a message, identified by the given messageId. The messageId is obtained by calling the POST /messages (4.1) method.

4.3.1 Input parameters

The input must be the path parameter of the GET request.

- messageId - The ID of the message to be cancelled

4.3.2 Output

See 4.2.2

4.3.3 State transition diagram

See 4.2.3

4.3.4 Example Request

```
DELETE: /messages/YzX95zUA1et2ijQ
```

4.3.5 Example Response

```
HTTP/1.0 200 OK
{
 "messageId" : " YzX95zUA1et2ijQ",
 "expirationDate" : "2015-06-11T13:06:12.658+02:00",
 "recipients" : [
   { "userId" : "pbmId1", "status" : "PENDING" },
   { "userId" : "pbmId2", "status" : "NOTIFIED" },
   { "userId" : "pbmId3", "status" : "SEEN" }
 ]
}
```

or

```
HTTP/1.0 420
{ "code": "NOT_FOUND", "message" : "No such message" }
```