Customer Integration API

ⓘ **Important**

This API is only available upon request to OX Cloud partners who are NOT using our fully OX branded tenants (*.appsuite.cloud)!

Version 1.3
Date: 12 May 2021

- 1. Revision History
- 2. Introduction
- 3. Accessing The API
    - 3.1 API Versioning
    - 3.2 Security
    - 3.3 URL Encoding
- 4. API concepts
    - 4.1 Status Codes
        - 4.1.1 Success Codes
        - 4.1.2 Client Error Codes
        - 4.1.3 Server Error Codes
    - 4.2 Response Format
        - 4.2.1 Success responses
        - 4.2.2 Error responses
- 5. Resources
        - 5.1.1 General Rules
        - 5.1.2 Field Definitions
    - 5.2 Aliases
        - 5.2.1 Query alias list of a mailbox (ALIASES)
        - 5.2.2 Check availability of an alias (ALIASES_AVAILABLE)
        - 5.2.3 Delete a single alias of a mailbox (ALIASES)
        - 5.2.4 Add an alias to a mailbox (ALIASES)
    - 5.3 Antispam / Block-/Allowlist
        - 5.3.1 Query allowlist filter of a mailbox
        - 5.3.2 Delete a single allowlist filter
        - 5.3.3 Query blocklist filter of a mailbox
        - 5.3.4 Delete a single blocklist filter
    - 5.4 Authentication
        - 5.4.1 Query authentication information for a mailbox (AUTH)
        - 5.4.2 Update password of a mailbox (AUTH)
        - 5.4.3 Set password hash of a mailbox (PASSWORDHASH)
    - 5.5 Filters
        - 5.5.1 Query filter list of a mailbox (FILTERS)
        - 5.5.2 Create a filter of mailbox (FILTERS)
        - 5.5.3 Query redirection filter list of a mailbox (FILTERS_REDIRECT)
        - 5.5.4 Delete a single filter of a mailbox (FILTERS)
    - 5.6 Infostore
        - 5.6.1 Query folder information
    - 5.7 Last Login
        - 5.7.1 Query last login information (LAST_LOGIN)
    - 5.8 Mail Accounts
        - 5.8.1 List mail accounts (MAIL_ACCOUNTS)
        - 5.8.2 Delete a mail account (MAIL_ACCOUNTS)
    - 5.9 Mailboxes
        - 5.9.1 Query an OX account (mailbox) (MAILBOXES)
        - 5.9.2 Query a single mailbox by e-mail address (MAILBOXES_BY_EMAIL)
        - 5.9.3 Check availability of several e-mail addresses (MAILBOXES_BY_EMAIL)
    - 5.10 Mail folders
        - 5.10.1 Query unread/recent message count for mailbox (RECENTMESSAGES)
        - 5.10.2 Query unread/recent messages of a mail folder (RECENTMESSAGES_BY_FOLDER)
    - 5.11 Out-Of-Office Notice
        - 5.11.1 Query the out-of-office notice of a mailbox (FILTERS_OUT_OF_OFFICE)
        - 5.11.2 Update the out-of-office notice of a mailbox (FILTERS_OUT_OF_OFFICE)
    - 5.12 Permissions v1 (deprecated in favor of v2)
        - 5.12.1 Query permissions of an OX account (mailbox) (PERMISSIONS)
        - 5.12.2 Update permissions of an OX account (mailbox) (PERMISSIONS)
    - 5.13 Permissions v2
        - 5.13.1 Query permissions of an OX account (mailbox) (PERMISSIONS)
        - 5.13.2 Update permissions of an OX account (mailbox) (PERMISSIONS)
        - 5.13.3 Query permission change history of an OX account (mailbox)
    - 5.14 Quota
        - 5.14.1 Query quota information of a mailbox (QUOTA)
    - 5.15 Sender Information
        - 5.15.1 Query sender information of a mailbox (SENDER)
    - 5.16 Public key

      - 5.15.1 Get current public key
  - 5.17 Snapshot resource
      - 5.17.1 Snapshot metadata
          - JSON representation
      - 5.17.2 Snapshot status
      - 5.17.3 Create a snapshot of a user's mailbox
      - 5.17.4 List all available snapshots
      - 5.17.5 Get snapshot metadata
      - 5.17.6 Get snapshot status
      - 5.17.7 Get snapshot dump
      - 5.17.8 Delete snapshot of mailbox
  - 5.18 Brand Permission History
      - 5.18.1 Query permissions change history of the current brand (BRAND_PERMISSIONS)

1. Revision History

Version	Changes	Date
1.0	Final complete spec	2016-12-14
1.1	Add endpoint for filter creation	2019-08-12
1.2	Added missing endpoint documentation	2020-10-08
1.3	Add new User Permission API and Brand Permission History API	2021-05-12

2. Introduction

To provide customers the means to integrate their applications, Open-Xchange provides a specific API. This API (called "Customer Integration-API") allows synchroneous access to resources residing within the OX product.
This document specifies the available resources and is intended for a technical audience. Its purpose is to enable software developers to integrate their product with the CI-API.

3. Accessing The API

3.1 API Versioning

The API supports API versioning to assure backwards compatibility in the case of major API changes. This is achieved by including a version identifier in the path used to access the API - e.g.:
/v1/mailboxes/..
The version is incremented for any API change that could break the compatibility of existing client applications - e.g. changed paths or structural changes in JSON requests and responses. The addition of fields to existing JSON responses without structural changes in the JSON document are considered minor changes and may not lead to an incremented API version. Client applications are expected to be implemented in a way that is robust with regards to such minor changes.

3.2 Security

Communication with the API endpoint is secured using HTTPS. API consumers are authenticated using SSL client authentication (client certificates).

3.3 URL Encoding

When accessing the API, special characters need to be encoded using percent-encoding as specified in RFC 3986.

4. API concepts

The API component provides a resource oriented API that follows RESTful principles using a simple UTF-8 encoded JSON format for request and response bodies. Logical resources can be addressed using fixed URLs and accessed using the following HTTP verbs:

- **GET**: Read a resource
- **POST**: Create a resource
- **PUT**: Update a resource
- **DELETE**: Delete a resource

4.1 Status Codes

Standard HTTP status codes are used to denote the outcome of API requests:

4.1.1 Success Codes

200: OK
201: Resource was created
204: Resource was removed

4.1.2 Client Error Codes

These error codes apply to all specified resources. Actual resource specifications may provide specific error code descriptions, if applicable.

400: The request was invalid (e.g. invalid request format, required parameter is missing, invalid parameters)
401: Request was not authorized
403: Resource access was not allowed
404: A requested resource was not found

4.1.3 Server Error Codes

500: A technichal error occurred on the server side

4.2 Response Format

In case of successful requests for writing operations, the returned responses follow some general rules depending on the used HTTP verb.

4.2.1 Success responses

- POST
  - response code 201
  - the response body will contain the JSON representation of the created resource including its ID
  - the HTTP headers will contain a location header containing the URL of the created resource
- PUT
  - response code 200
  - the response body will contain the JSON representation of the updated resource
- DELETE
  - response code 204
  - the response body will contain an empty JSON document

4.2.2 Error responses

In case of a failed requests, the JSON response body will contain detailed error information:

```
{
  "errorCode": "SnapshotService.ValidationError.Snapshot", "errorMessage": "The ssh port is not a number.",
"errorId": "2735af1e-2856-4437-80ef-70b6e988a59c"
}
```

Error Fields

- errorCode: The specific error code for the inidvidual backend service (if it is available)
- errorMessage: The detailed error message
- errorId: The unique error ID of the individual backend service (if it is available)

5. Resources

This chapter specifies the relevant REST resources and their usage.

5.1.1 General Rules

The following general rules apply for all described REST resources:
Unless noted otherwise, all specified parameters are mandatory. Omitting a mandatory parameter will result in an error response with status 400 (invalid request).

5.1.2 Field Definitions

Basic Data Definitions

Name	Type	Length	Description	Mandatory/ Optional
userName	string	128	federation ID of the user	M
password	string	256	password of the user for direct imap/pop3/smtp access	M
displayName	string	320	Display name used for sending mail	M
surname	string	128	Surname of the user	M
givenname	string	128	Givenname of the user	M
primaryEmail	string	256	Uniqe email address of the account	M

5.2 Aliases

This resource represents e-mail aliases associated with mailboxes.

5.2.1 Query alias list of a mailbox (ALIASES)

This resource is used to query the list of email aliases associated with the mailbox of the given user.

Request

```
GET /v1/mailboxes/<userName>/aliases/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "aliases": [
    "address1@example.com",
    "address2@example.com"
  ]
}
```

Fields

- **aliases**: list of aliases

Status

200

Error Codes

See: API Concepts for general error information

- 400: User name is missing

- 404: User does not exist

5.2.2 Check availability of an alias (ALIASES_AVAILABLE)

This resource is used to check the availability of the given e-mail alias.

Request

```
GET /v1/mailboxes/<userName>/aliases/available/<aliasAddress>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **aliasAddress**: the alias to check (String, local part or full email address according to RFC 5322)

Success Response

Body

```
{
  "available": true
}
```

Fields

- **available**: flag indicating, whether the alias is still available

Status

200

Error Codes

See: API Concepts for general error information

- 400: User name or alias address are missing/invalid
- 404: User does not exist

5.2.3 Delete a single alias of a mailbox (ALIASES)

This resource is used to delete an e-mail alias from the mailbox of the given user.

Request

```
DELETE /v1/mailboxes/<userName>/aliases/<aliasAddress>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **aliasAddress**: the alias e-mail address to delete (String, valid email address according to RFC 5322)

Success Response

Status

204

Error Codes

See: API Concepts for general error information

- **400**: User name or alias address are missing/invalid
- **404**: User does not exist

5.2.4 Add an alias to a mailbox (ALIASES)

This resource is used to add a new e-mail alias to the mailbox of the given user.

Request

`POST /v1/mailboxes/<userName>/aliases/`

Body

```
{
  "alias": "username3@domain.com"
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **alias**: the alias to add (String, valid email address according to RFC 5322)

Success Response

Body

```
{
  "aliases": [
    "address1@example.com",
    "address2@example.com",
    ...
  ]
}
```

Fields

- **aliases**: the resulting list of aliases

Status

201

Error Codes

See: API Concepts for general error information

- **400**: User name or alias address are missing/invalid
- **404**: User does not exist

5.3 Antispam / Block-/Allowlist

This resource represents the antispam settings associated with a mailbox.

5.3.1 Query allowlist filter of a mailbox

This resource is used to query the antispam allowlist of the mailbox of a given user.

Request

```
GET /v1/mailboxes/<userName>/antispam/whitelist/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "whitelist": [
    "address1@example.com",
    "address2@example.com",
    ...
  ]
}
```

Fields

- **whitelist**: list of whitelited addresses/domains

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.3.2 Delete a single allowlist filter

This resource is used to delete an email address or domain from the antispam allowlist of the mailbox of a given user.

Request

```
DELETE /v1/mailboxes/<userName>/antispam/whitelist/<emailAdress>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **emailAddress**: the email address to delete (String)

Success Response

Status

204

Error Codes

See: API Concepts for general error information

- **400**: User name or email address is missing/invalid
- **404**: User does not exist

5.3.3 Query blocklist filter of a mailbox

This resource is used to query the antispam blocklist of the mailbox of a given user.

Request

```
GET /v1/mailboxes/<userName>/antispam/blacklist/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "blacklist": [
    "address1@example.com",
    "address2@example.com",
    ...
  ]
}
```

Fields

- **blacklist**: list of whitelited addresses/domains

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.3.4 Delete a single blocklist filter

This resource is used to delete an email address or domain from the antispam blocklist of the mailbox of a given user.

Request

```
DELETE /v1/mailboxes/<userName>/antispam/blacklist/<emailAdress>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **emailAddress**: the email address to delete (String)

Success Response

Status

204

Error Codes

See: API Concepts for general error information

- **400**: User name or email address is missing/invalid
- **404**: User does not exist

5.4 Authentication

This resource represents the authentication information associated with a mailbox.

Authentication Fields

active	Flag indicating, whether the password is active
passwordMisentries	Number of password misentries since last successful login
passwordLastChanged	Timestamp of last password change as java timestamp (miliseconds since epoch) and UTC

5.4.1 Query authentication information for a mailbox (AUTH)

This resource is used to query authentication information of the mailbox of a given user.

Request

```
GET /v1/mailboxes/<userName>/auth/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

```
{
  "active": true, "passwordMisentries": 0,
  "passwordLastChanged": 1465552800000
}
```

Fields

- **active**: Flag indicating, whether the password is active
- **passwordMisentries**: Number of password misentries since last successful login
- **passwordLastChanged**: Timestamp of last password change as Java timestamp (miliseconds since epoch) and UTC If no data is found in the backend system the fields will be set to 'null'.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing/invalid
- **404**: User does not exist

5.4.2 Update password of a mailbox (AUTH)

This resource is used to update the password of the mailbox of a given user.

Request

`PUT /v1/mailboxes/<userName>/auth/`

Body

```
{
  "password": "topsecret"
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **password**: the password to set (String)

Success Response

Body

```
{
  "active": true, "passwordMisentries": 0,
  "passwordLastChanged": 1465552800000
}
```

Fields

- **active**: Flag indicating, whether the password is active
- **passwordMisentries**: Number of password misentries since last successful login
- **passwordLastChanged**: Timestamp of last password change as java timestamp (miliseconds since epoch) and UTC If no data is found in the backend system the fields will be set to 'null'.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or request body are missing/invalid
- **404**: User does not exist

5.4.3 Set password hash of a mailbox (PASSWORDHASH)

This resource is used to set the password hash of the mailbox of a given user.

Request

`PUT /v1/mailboxes/<userName>/auth/hash`

Body

```
{
  "passwordHash": "{HASH_METHOD}<valid_password_hash>"
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **passwordHash**: the password hash to set (has formatted string)
  - **HASH_METHOD:** supported values are MD5, SHA, CRYPT, SMD5, SSHA, SSHA384, SSHA512

Success Response

Body

```
{
  "active": true, "passwordMisentries": 0,
  "passwordLastChanged": 1465552800000
}
```

Fields

- **active**: Flag indicating, whether the password is active
- **passwordMisentries**: Number of password misentries since last successful login
- **passwordLastChanged**: Timestamp of last password change as java timestamp (miliseconds since epoch) and UTC If no data is found in the backend system the fields will be set to 'null'.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or request body are missing/invalid
- **404**: User does not exist

5.5 Filters

This resource represents the e-mail filters associated with a mailbox.

5.5.1 Query filter list of a mailbox (FILTERS)

This resource is used to query the list of e-mail filters of the mailbox of a given user. The returned filters follow the format defined in the OX HTTP API.

Request

```
GET /v1/mailboxes/<userName>/filters/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "filters": [
  {
    "id": 1,
    "rulename": "my filter", "active": true,
    "test": {
      "id": "true"
    },
    "actioncmds": [{
      "id": "size", "comparison": "over", size": 100000 }
    ]
  },
  {
    "id": 2,
    ...
  }
  ]
}
```

Fields

- **filters**: List of mail filters with the following fields:
  - **id**: ID of the filter
  - **rulename**: Display name of the filter rule
  - **active**: Status of the rule
  - **test**: List of test objects as defined in Tests
  - **actioncmds**: List of action commands as defined in Actions

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.5.2 Create a filter of mailbox (FILTERS)

This resource is used to add a new filter to the mailbox of a given user.
The filter format follows the format defined in the OX HTTP API

Request

```
POST /v1/mailboxes/<userName>/filters/
```

Fields
All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "filters": [
  {
    "id": 1,
    "position": 0,
    "rulename": "my filter",
    "active": true,
    "test": {
      "id": "true"
    },
    "actioncmds": [{
    "id": "size",
    "comparison": "over",
    "size": 100000}
  ]
  },
  {
  "id": 2,
  ...
  }
  ]
}
```

Fields

- **filters**: List of mail filters with the following fields:
  - **id**: ID of the filter
  - **position:** The position in the filter list
  - **rulename:** Display name of the filter rule
  - **active:** Status of the rule
  - **flags:** List of optional flags of the rule
  - **test:** List of test objects as defined in Tests
  - **actioncmds:** List of action commands as defined in Actions

Status

201

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.5.3 Query redirection filter list of a mailbox (FILTERS_REDIRECT)

This resource is used to query the list of redirection filters of the mailbox of a given user. The resulting filters will only contain redirection filters - i.e. filters that contain an action command of type *redirect*. All action commands that are not of type *redirect* will be omitted in the resulting filters.
The returned filters follow the format defined in the OX HTTP API.

Request

```
GET /v1/mailboxes/<userName>/filters/redirect/
```

Fields
All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "filters": [
  {
    "actioncmds": [
    {
      "id": "redirect",
      "to": "test@domain.com"
    }
    ],
    "active": true,
    "id": 2,
    "rulename": "autoforward",
    "test": {
      "id": "true"
    }
  },

  { filter 2 }
  ]
}
```

Fields

- **filters**: List of mail filters with the following fields:
  - **id**: ID of the filter
  - **rulename**: Display name of the filter rule
  - **active**: Status of the rule
  - **test**: List of test objects as defined in Tests
  - **actioncmds**: List of action commands as defined in Actions

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.5.4 Delete a single filter of a mailbox (FILTERS)

This resource is used to delete a single e-mail filter from the mailbox of a given user.

Request

```
DELETE /v1/mailboxes/<userName>/filters/<filterId>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **filterId**: the ID of the filter (Integer)

Success Response

Status

204

Error Codes

See: API Concepts for general error information

- **400**: User name or filter are missing/invalid
- **404**: User or filter does not exist

5.6 Infostore

This resource represents the content of the OX infostore.

5.6.1 Query folder information

This resource is used to query information about the contents of a folder in the OX infostore. If no folder ID is provided, the content of the root folder of the OX infostore will be returned.

Request

- root folder (INFOSTORE):

  ```
  GET /v1/mailboxes/<userName>/infostore/
  ```

- or specific folder (INFOSTORES):

  ```
  GET /v1/mailboxes/<userName>/infostore/<folderId>
  ```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **folderId**: the ID of the folder (Integer)

Success Response

Body

```
{
  "name": "Images",
  "id": "111"
  "parent": "110",
  "files": [
  {
    "name": "Image1.jpg",
    "mime-type": "image/jpeg",
    "size": 145882
  },
  {
    "name": "Image.jpg",
    "mime-type": "image/jpeg",
    "size": 195876
  },
  ],
  "subFolders": [
  {
    "name": "Archive",
    "id": "112"
  },
  {
    "name": "Trash",
    "id": "113"
  },
  ],
}
```

Fields

- **name**: the folder name

- **id**: the folder ID
- **parent**: the parent folder ID
- **files**: list of files with the following fields:
    - **name**: the file name
    - **mime-type**: the file MIME type
    - **size**: the file size
    - **subFolders**: list of sub folders with the following fields:
        - **name**: the folder name
        - **id**: the folder ID

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or folder ID are missing/invalid
- **404**: User does not exist

5.7 Last Login

Last Login Fields

Name	Description
protocol	The protocol used to access the mailbox. The result list will contain one object for each protocol available on the queried mailbox
lastAccess	The last access time is returned as java timestamp (miliseconds since epoch) and UTC
lastAccessIP	The IP address of the last access might be *null* if it is unknown

This resource represents mailbox access information of the available protocols.

5.7.1 Query last login information (LAST_LOGIN)

This resource is used to query last access information of a given user.

Request

```
GET /v1/mailboxes/<userName>/last_login/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "access": [
  {
    "protocol": "POP3",
    "lastAccess": 1465552800000
    "lastAccessIP": "127.0.0.1"
  },
  {
    "protocol": "IMAP",
    "lastAccess": 1465552800000
    "lastAccessIP": "127.0.0.1"
  },
  {
    "protocol": "HTTP",
    "lastAccess": 1465552800000
    "lastAccessIP": "127.0.0.1"
  },
  ]
}
```

Fields

- **protocol**: The protocol used to access the mailbox. The result list will contain one object for each protocol available on the queried mailbox
-

  lastAccess: The last access time is returned as java timestamp (miliseconds since epoch) and UTC
- **lastAccessIP**: The IP address of the last access might be null if it is unknown

If no data is found in the backendsystem, the list will be empty.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.8 Mail Accounts

This resource represents the e-mail accounts associated with users.

Mail Account Fields

Name	**Description**
ID	ID of the account. The primary OX account is identified by ID 0.
login	login name of the account
password	password of the account
email	E-mail address of the account
incomingUrl	URL of the mail server used to receive mails
incomingSecure	whether to establish a secure connection to the incoming mail server
outgoingUrl	URL of the mail server used to send mails
outgoingSecure	whether to establish a secure connection to the outgoing mail server

5.8.1 List mail accounts (MAIL_ACCOUNTS)

This resource is used to query the list of e-mail accounts associated with a given user.

Request

```
GET /v1/mailboxes/<userName>/mail_accounts/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "mailAccounts": [
  {
    "id": 111,
    "login": "username",
    "email": "user@domain.com",
    "incomingUrl": "imap://mail.domain.com",
    "incomingSecure": true,
    "outgoingUrl": "smtp://mail.domain.com",
    "outgoingSecure": true,
  },
  {
    "id": 112,
    "login": "username",
    "email": "user2@domain.com",
    "incomingUrl": "imap://mail.domain.com",
    "incomingSecure": true,
    "outgoingUrl": "smtp://mail.domain.com",
    "outgoingSecure": true,
  },
  ]
}
```

Fields

- **mailAccounts**: The list of external mail accounts with the following fields:
    - **id**: the ID of the account. The primary OX account is identified by ID 0.
    - **login**: the login name of the account
    - **email**: the e-mail address of the account
    - **incomingUrl**: the URL of the mail server used to receive mails
    - **incomingSecure**: whether to establish a secure connection to the incoming mail server outgoingUrl: the URL of the mail server used to send mails
    - **outgoingSecure**: whether to establish a secure connection to the outgoing mail server

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.8.2 Delete a mail account (MAIL_ACCOUNTS)

This resource is used to delete the settings of a single e-mail account associated with a given user.

Request

```
DELETE /v1/mailboxes/<userName>/mail_accounts/<id>
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **id**: the ID of the mail account (Integer)

Success Response

Status

204

Error Codes

See: API Concepts for general error information

- **400**: User name or account ID are missing
- **404**: User does not exist

5.9 Mailboxes

This resource represents mailboxes of users (i.e. OX accounts).

5.9.1 Query an OX account (mailbox) (MAILBOXES)

This resource is used to query information about the mailbox of a given username or email.

Request

```
GET /v1/mailboxes/<username>
GET /v1/mailboxes?username=<username>
GET /v1/mailboxes?email=<email>
```

Fields

All listed parameters are mandatory.

- **username**: the unique ID of the user (String)
  or
- **email**: the email of the user (String)

Success Response

Body

```
{
  "userName" : "unique ID",
  "displayName" : "Joe Smith",
  "surname" : "Smith",
  "givenName" : "Joe",
  "primaryEmail" : "joe.smith@example.com",
  "classOfService" : "premium"
}
```

Fields

- **userName**: the unique ID of the user
- **displayName**: the display name used for sending email
- **surname**: the surname of the user
- **givenName**: the given name of the user
- **primaryEmail**: the unique primary email address of the account
- **classOfService**: the class of service (String) - might be *null* if not set

Status

200

Error Codes

See: Chapter 4 - API Concepts for general error information

- **400**: User name or email is missing
- **404**: User does not exist

5.9.2 Query a single mailbox by e-mail address (MAILBOXES_BY_EMAIL)

This resource is used to search a single mailbox using the primary e-mail address.

Request

```
GET /v1/mailboxes/by_email/<emailAddress>
```

Fields

All listed parameters are mandatory.

- **emailAddress**: the email address to search (String, valid email address according to RFC 5322)

Success Response

Body

```
{
  "userName": "unique ID"
}
```

Fields

- **userName**: the unique ID of the user

Status

200

Error Codes

See: Chapter 4 - API Concepts for general error information

- **400**: Email address is missing/invalid
- **404**: User does not exist

5.9.3 Check availability of several e-mail addresses (MAILBOXES_BY_EMAIL)

This resource is used to check the availability of a list of e-mail addresses.

Request

```
GET /v1/mailboxes/by_email?available=<email-addresses>
```

Fields

- **available**: the list of e-mail addresses to query (comma-separated list of valid email addresses according to RFC 5322, must be URL encoded)

Success Response

Body

```
{
  "mail1@example.com" : true,
  "mail2@example.com" : false,
}
```

Status

200

Error Codes

See: Chapter 4 - API Concepts for general error information

- **400**: e-mail address list is missing/invalid

5.10 Mail folders

This resource represents the mail folders associated with mailboxes.

5.10.1 Query unread/recent message count for mailbox (RECENTMESSAGES)

This resource is used to query the total number of unread (recent) messages of all mail folders excluding trash and spam.

Request

```
GET /v1/mailboxes/<userName>/recentmessages
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "recent": 0
}
```

Fields

- **recent**: The total number of unread messages in all folders excluding trash and spam

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.10.2 Query unread/recent messages of a mail folder (RECENTMESSAGES_BY_FOLDER)

This resource is used to query the total number of unread messages of a given mail folder.

Request

```
GET /mailboxes/<userName>/<folderName>/unread
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **folderName**: the name of the folder to query - e.g. *INBOX* to query the inbox of the user (String)

Success Response

Body

```
{
  "unread": 0
}
```

Fields

- **unread**: The total number of unread messages in the queried folder

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or folder name are missing/invalid
- **404**: User does not exist

5.11 Out-Of-Office Notice

This resource represents the out-of-office notice of a mailbox.

Out-Of-Office Fields

Name	Description
message	the message body of the out-of-office notice
subject	the message subject of the out-of-office notice
startDate	timestamp of the start date as java timestamp (miliseconds since epoch) and UTC
endDate	timestamp of the end date as java timestamp (miliseconds since epoch) and UTC
active	flag indicating, whether the out-of-office notice is active

5.11.1 Query the out-of-office notice of a mailbox (FILTERS_OUT_OF_OFFICE)

This resource is used to query the out-of-office notice of a given user.

Request

```
GET /v1/mailboxes/<userName>/filters/out_of_office/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "message": "I'm not in the office currently.",
  "subject": "Out-of-office notice",
  "startDate": 1465552800000 ,
  "endDate": 1465552800000,
  "active": true
}
```

Fields

- **message**: the message body of the out-of-office notice
- **startDate**: timestamp of the start date as java timestamp (miliseconds since epoch) and UTC
- **endDate**: timestamp of the end date as java timestamp (miliseconds since epoch) and UTC
- **active**: flag indicating, whether the out-of-office notice is active

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.11.2 Update the out-of-office notice of a mailbox (FILTERS_OUT_OF_OFFICE)

This resource is used to update the out-of-office notice of a given user.

Request

```
PUT /v1/mailboxes/<userName>/filters/out_of_office/
```

Body

```
{
  "message": "I'm not in the office currently.",
  "subject": "Out-of-office notice",
  "startDate": 1465552800000 ,
  "endDate": 1465552800000,
  "active": true
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **message**: the message body of the out-of-office notice (String)
- **startDate**: timestamp of the start date as Java timestamp (miliseconds since epoch) and UTC
- **endDate**: timestamp of the end date as Java timestamp (miliseconds since epoch) and UTC
- **active**: flag indicating, whether the out-of-office notice is active (Boolean)

Success Response

Body

```
{
  "message": "I'm not in the office currently.",
  "subject": "Out-of-office notice",
  "startDate": 1465552800000,
  "endDate": 1465552800000,
  "active": true
}
```

Fields

- **message**: the message body of the out-of-office notice
- **startDate**: timestamp of the start date as Java timestamp (miliseconds since epoch) and UTC
- **endDate**: timestamp of the end date as java timestamp (miliseconds since epoch) and UTC
- **active**: flag indicating, whether the out-of-office notice is active

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or request body are missing/invalid
- **404**: User does not exist

5.12 Permissions v1 (deprecated in favor of v2)

This resource represents the access permissions of a mailbox.

Permission Fields

Name	Description
SEND	user is allowed to send mail
RECEIVE	user is allowed to receive mail
MAILLOGIN	user can login using IMAP/PO3
WEBLOGIN	user can login to OX webmail

5.12.1 Query permissions of an OX account (mailbox) (PERMISSIONS)

This resource is used to query the access permissions of the mailbox of a given user.

Request

```
GET /v1/mailboxes/<userName>/permissions/
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "permissions": [
    "SEND",
    "RECEIVE"
  ]
}
```

Fields

- **permissions**: The list of permissions of a mailbox.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.12.2 Update permissions of an OX account (mailbox) (PERMISSIONS)

This resource is used to update the access permissions of the mailbox of a given user.

Request

```
PUT /v1/mailboxes/<userName>/permissions/
```

Body

```
{
  "permissions": [
    "SEND",
    "RECEIVE"
  ],
  "reason": "account unlocked"
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **permissions**: the list of permissions
- **reason**: the reason for the change

Success Response

Body

```
{
  "permissions": [
    "SEND",
    "RECEIVE"
  ]
}
```

Fields

- **permissions**: The list of permissions.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or request body are missing/invalid
- **404**: User does not exist

5.13 Permissions v2

This resource represents the access permissions of a mailbox via API v2.

Permission Fields

Name	**Description**
SEND	User is allowed to send mail
RECEIVE	User is allowed to receive mail
MAILLOGIN	User can login using IMAP/PO3
WEBLOGIN	User can login to OX App Suite

5.13.1 Query permissions of an OX account (mailbox) (PERMISSIONS)

This resource is used to query the access permissions of the mailbox of a given user.

Request

`GET /v2/mailboxes/<userName>/permissions/`

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "enabled": [
    "MAILLOGIN",
    "SEND",
    "RECEIVE"
  ],
  "disabled": [
    "WEBLOGIN"
  ]
}
```

Fields

- **enabled**: The current enabled permissions of the user
- **disabled:** The current disabled permissions of the user

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.13.2 Update permissions of an OX account (mailbox) (PERMISSIONS)

This resource is used to update the access permissions of the mailbox of a given user via API v2

Request

`PUT /v2/mailboxes/<userName>/permissions/`

Body - Enable permissions

```
{
  "enable": [
    "SEND",
    "RECEIVE"
  ],
  "reason": "permissions enabled",
  "clientUser": "unique user id"
}
```

Body - Disable permissions

```
{
  "disable": [
    "WEBLOGIN",
    "MAILLOGIN"
  ],
  "reason": "permissions disabled due to misuse",
  "clientUser": "unique user id"
}
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)
- **enable**: the list of permissions that should be enabled
- **disable**: the list of permissions that should be disabled
- **reason**: the reason for the change

Success Response

Body - Enable permissions with change

```
{
  "change": {
    "enabled": ["SEND"]
  },
  "permissions": {
    "enabled": ["SEND", "RECEIVE", "MAILLOGIN", "WEBLOGIN"],
    "disabled" : []
  }
}
```

Body - Disable permissions with change

```
{
  "change": {
    "disabled": ["MAILLOGIN", "WEBLOGIN"]
  },
  "permissions": {
    "enabled": ["SEND", "RECEIVE"],
    "disabled" : ["MAILLOGIN", "WEBLOGIN"]
  }
}
```

Body - without change

```
{
  "permissions": {
    "enabled": [
      "SEND",
      "RECEIVE",
      "MAILLOGIN"
    ],
    "disabled" : [
      "WEBLOGIN"
    ]
  }
}
```

Fields

- **change:** Object is optional and describes the actual performed change (if any) of enable or disable specific permissions for the user.
- **permissions**: The list of permissions.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name or request body are missing/invalid
- **404**: User does not exist

5.13.3 Query permission change history of an OX account (mailbox)

This resource is used to query the change history of permissions associated with the mailbox of the given user via API v2.

Request

```
GET /v2/mailboxes/<userName>/permissions/history
```

Path-Parameters

Parameter	Format & Range of Values	Mandatory	Description
userName	String	Yes	The unique ID of the user

Query-Parameters

A Request of the Permission History can be optionally extended with one or more Query-Parameters

Parameter	Format & Range of Values	Mandatory	Description
`before`	ISO-8601 Date-Time	No	Only return permission changes prior to before.
`after`	IOS-8601 Date-Time	No	Only return permissio nchanges that occurred after after.
`order`	`asc, desc`	No	Only when using the flat format, order of the entries by the time attribute ▪ ascending: oldest permission changes first ▪ descending: most recent permission changes first
`limit`	Integer of 0,...	No	Limits the number of objects that are returned, reduction varies if object or array format is requested: ▪ flat format: the maximum amount of JSON objects returned in the JSON array ▪ map format: the maximum amount permission change JSON objects that are returned for each oxid/user

Success Response

A JSON object that contains an array of changes.

Body

```
{
  "changes": [
    {
      "time": "2021-03-26T12:55:32.193Z",
      "authUser": "brand1",
      "clientIp": "192.168.0.1",
      "clientUser": "subsubadmin1",
      "disabled": [],
      "enabled": [
        "WEBLOGIN",
        "MAILLOGIN",
        "SEND",
        "RECEIVE"
      ],
      "ipAddress": "10.10.150.233",
      "reason": "re-enabled upon customer request"
    },
    {
      "time": "2021-03-23T04:23:11.191Z",
      "authUser": "brand1",
      "clientIp": "192.168.0.1",
      "clientUser": "subsubadmin1",
      "disabled": [
        "WEBLOGIN",
        "MAILLOGIN",
        "SEND",
        "RECEIVE"
      ],
      "enabled": [],
      "ipAddress": "10.10.150.233",
      "reason": "disabled due to abuse detection"
    }
  ]
}
```

Fields

- **time:** An ISO-8601 formatted UTC string represents the point in time when the permission change was performed
- **authUser**: The username used in the successful authentication when performing the permission change
- **clientIp**: IP address of the client that invoked the software that performed the permission change (optional)
- **clientUser**: Username of the credentials that authenticated against the software that performed the permission change (optional)
- **disabled**: An array of strings that hold the permissions that were effectively disabled
- **enabled**: An array of strings that hold the permissions that were effectively enabled
- **ipAddress**: The IP address of the client that performed the API operation to change permissions
- **reason**: The reason for the permission change, as indicated by the client that performed the API operation to change permissions (except when using the SOAP API)

Status

200

Error Codes

See: API Concepts for general error information

- **400:** Bad query request, most likely wrong query parameter

5.14 Quota

This resource represents the quotas associated with a mailbox:

- **mail quota**: e-mail quota, if unified quota is not available - otherwise empty
- **cloud quota**: cloud storage quota, if unified quota is not available - otherwise empty
- **unified quota**: total quota, if available

5.14.1 Query quota information of a mailbox (QUOTA)

Request

`GET /v1/mailboxes/<userName>/quota/`

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "mailQuota": {
    "maxQuota": 2048,
    "usedQuota": 578
  },
  "cloudQuota": {
    "maxQuota": 2048,
    "usedQuota": 237
  }
}
```

Fields

- **mailQuota**: mail quota information containing the following fields:
  - **maxQuota**: maximum quota in megabytes
  - **usedQuota**: used quota in megabytes
- **cloudQuota**: cloud quota information containing the following fields:
  - **maxQuota**: maximum quota in megabytes
  - **usedQuota**: used quota in megabytes
- **unifiedQuota**: unified quota information containing the following fields:
  - **maxQuota**: maximum quota in megabytes
  - **usedQuota**: used quota in megabytes

Missing quota values will be returned as null values.

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.15 Sender Information

This resource represents the sender information of a mailbox.

5.15.1 Query sender information of a mailbox (SENDER)

This resource is used to query sender information of a given user.

Request

```
GET /v1/mailboxes/<userName>/sender
```

Fields

All listed parameters are mandatory.

- **userName**: the unique ID of the user (String)

Success Response

Body

```
{
  "senderName": "Example User",
  "senderAddress": "test@example.com"
}
```

Fields

- **senderName**: the sender display name
- **senderAddress**: the sender email address

Status

200

Error Codes

See: API Concepts for general error information

- **400**: User name is missing
- **404**: User does not exist

5.16 Public key

5.15.1 Get current public key

This resource provides the current public key that is used for uploading data through scp/sftp.

Request

```
GET /intercept/publickey
```

Success response

Header

Content-Type: "text/plain"

Body

```
{
   "ssh-rsa": "AAAAB3NzaC1yc2EAAAADAQABAAACAQ..."
}
```

Status

200

5.17 Snapshot resource

A Snapshot Resource is defined by the base path: /intercept/snapshots

The representations of a snapshot are meta data (for creation and management) and the binary data of the snapshot itself.

The creation of the snapshot dump is an asynchronous operation and the current status can be queried with corresponding snapshot ID.

5.17.1 Snapshot metadata

Snapshot metadata is represented by JSON, the represntation can differ between the different REST calls.

JSON representation

Field	Description	Mandatory
{	-	-
"snapshotId": <string>,	The unique id that qualified a snapshot, will be a UUID.	Yes
"username": <string>,	The name of the user for which the snapshot is created.	Yes
"type": <string>,	The type of the snapshot, one of: MAILBOX, CALENDAR, CONTACTS.	Yes
"action": <string>,	The type of upload action, one of: SCP, SFTP, DOWNLOAD.	Yes
"status": <string>,	The current snapshot status, see status table below.	Yes
"lastChanged": <timestamp>,	The timestamp when the last status change happened.	No. Service managed.
"ssh": {	The ssh connection object.	Yes if "action" is "SCP" or "SFTP"
"fingerprint": "75:5c:f9:c8:b6:f0:18:aa:23:ec:d9:67:56:ee:5f:c1",	The public key of remote server for upload in hexdecimal notation.	Yes if "action" is "SCP" or "SFTP"
"host": "upload.customer.net",	The FQDN or IP of the upload destination.	Yes if "action" is "SCP" or "SFTP"
"port": 22,	The network port of the upload destination.	Yes if "action" is "SCP" or "SFTP"
"path": "/upload/dir",	The path on remote server to store the upload file.	Yes if "action" is "SCP" or "SFTP"
"username": "uploaduser"	The username used when connecting to remote server.	Yes if "action" is "SCP" or "SFTP"
},	-	-
"start": "2016-08-01T14:00:00Z",	The start date (interval) of the messages to be exported.	Yes if "type" is "MAILBOX"
"end": "2016-08-14T15:00:00Z",	The end date (interval) of the messages to be exported.	Yes if "type" is "MAILBOX"
"error": {	The error object, set if status is 'failed'.	No, Service managed.
"errorCode": <string>,	The error code of the failure.	No, Service managed.
"errorMessage": <string>	The error message for the failure.	No, Service managed.
}	-	-
}	-	-

5.17.2 Snapshot status

A Snapshot status is based on it's current state of processing.

Status	Description
pending	After creation is pending and waits for execution.
dumping	The dump is in progress.
dumped	The dump was generated and waits for upload.
uploading	The upload is in progress.
available	Indicates, that the upload was successful and snapshot is available.
failed	Indicates, that an error happens on dumping or uploading operation.
deleted	The snapshot has been deleted by an operator

5.17.3 Create a snapshot of a user's mailbox

This resource is used to create a snapshot of the data of a given user.

Request

```
POST /intercept/snapshots
```

Body

Mailbox snapshot	Calendar snapshot	Contacts snapshot

{
"username" : "someone",
"type": "mailbox",
"action": "scp",
"ssh": {
"fingerprint": "75:5c:f9:c8:b6:f0:18:aa:23:ec:d9:67:56:ee:5f:c1.",
"host": "upload.customer.net",
"port": 22,
"path": "/upload/dir",
"username": "uploaduser"
},
"start": "2016-08-01T14:00:00Z",
"end": "2016-08-14T15:00:00Z"
}
	{
	"username" : "someone",
	"type": "calendar",
	"action": "scp",
	"ssh": {
	"fingerprint": "75:5c:f9:c8:b6:f0:18:aa:23:ec:d9:67:56:ee:5f:c1.",
	"host": "upload.customer.net",
	"port": 22,
	"path": "/upload/dir",
	"username": "uploaduser"
	}
	}
		{
		"username" : "someone",
		"type": "contacts",
		"action": "scp",
		"ssh": {
		"fingerprint": "75:5c:f9:c8:b6:f0:18:aa:23:ec:d9:67:56:ee:5f:c1.",
		"host": "upload.customer.net",
		"port": 22,
		"path": "/upload/dir",
		"username": "uploaduser"
		}
		}

Sucess response

Header

```
Location=/intercept/snapshots/{1a2b3c-..}
```

Body

```
{
  "snapshotId": "1a2bCc-..",
  "username" : "someone",
  "type": "calendar",
  "action": "scp",
  "status": "pending"
  "lastChanged": 123456..
}
```

Status

200

5.17.4 List all available snapshots

This resource is used to query the list of all available snapshots.

Request

```
GET /intercept/snapshots?status=available
```

Sucess response

Body

```
[
  {
    "snapshotId": <string>,
    "username": <string>,
    "type": <string>,
    "action": <string>,
    "status": "available"
    "lastChanged": <timestamp>
  },
  {
    "snapshotId": <string>,
    "username": <string>,
    ...
  }
]
```

Status

200

5.17.5 Get snapshot metadata

This resource is used to query the metadata of a specific snapshot.

Request

```
GET /intercept/snapshots/{snapshotId}
```

Sucess response

Body

```
{
  "snapshotId": <string>,
  "username": <string>,
  "type": <string>,
  "action": <string>,
  "status": <string>,
  "lastChanged": <timestamp>,
  "error": <object>
}
```

Status

200

5.17.6 Get snapshot status

This resource is used to query the current status of a given snapshot.

Request

`GET /intercept/snapshots/{snapshotId}/status`

Sucess response

Body

```
{
  "status": <string>
}
```

Status

200

5.17.7 Get snapshot dump

This resource provides the binary dump of the snapshot if snapshot was created with action "DOWNLOAD".

Request

`GET /intercept/snapshots/{snapshotId}/dump`

Success response

Header

Content-Type: application/octet-stream

Body

byte[]

Status

200

5.17.8 Delete snapshot of mailbox

This resource is used to mark a snapshot as deleted. If the snapshot was of type 'Download', the actual dump is deleted too.

Request

`DELETE /intercept/snapshots/{snapshotId}`

Sucess response

Status

204

5.18 Brand Permission History

This resource represents the access to permission history of all (sub) brand's mailboxes via API v1.

5.18.1 Query permissions change history of the current brand (BRAND_PERMISSIONS)

This resource is used to query the history of permission changes of the current brand.

Request

```
GET /v1/permissions/history
```

Query parameters

A Request of the Permission History can be optionally extended with one or more Parameters

Parameter	Format & Range of Values	Mandatory	Description
`before`	ISO-8601 Date-Time	No	Only return permission changes prior to before.
`after`	ISO-8601 Date-Time	No	Only return permission changes that occurred after `after`.
`flatten`	`true / false`	No	Use JSON object or array format, see response examples.
`order`	`asc / desc`	No	Only when using the flat format, order of the entries by the time attribute. ▪ ascending: oldest permission changes first ▪ descending: most recent permission changes first
`limit`	Integer of 0, ...	No	Limits the number of objects that are returned, reduction varies if object or array format is requested: ▪ flat format: the maximum amount of JSON objects returned in the JSON array ▪ map format: the maximum amount permission change JSON objects that are returned for each oxid /user

Success Response - Map Format

A JSON object with its keys being "OX identifiers" (userid@contextid), and each value being a JSON array of permission change JSON objects.

Body

```
{
  "3@100": [
    {
      "time": "2021-03-30T13:33:17.643Z",
      "authUser": "brand1",
      "disabled": [
        "RECEIVE",
        "SEND"
      ],
      "ipAddress": "10.10.1.10",
      "reason": "sendmail flooding detection"
    },
    {
      "time": "2021-03-30T13:33:17.660Z",
      "authUser": "brand1",
      "disabled": [
        "MAILLOGIN",
        "WEBLOGIN"
      ],
      "ipAddress": "10.20.32.193",
      "reason": "DDoS abuse detection"
    },
    {
      "time": "2021-03-30T13:33:17.671Z",
      "authUser": "brand1",
      "enabled": [
        "RECEIVE",
        "SEND",
        "MAILLOGIN",
        "WEBLOGIN"
      ],
      "ipAddress": "192.168.38.199",
      "reason": "Resolved by customer support",
      "clientIp": "82.193.93.112",
      "clientUser": "carla"
    }
  ],
  "3@329": [ ... ]
}
```

Fields

- **time:** An ISO-8601 formatted UTC string represents the point in time when the permission change was performed
- **authUser**: The username used in the successful authentication when performing the permission change
- **clientIp**: IP address of the client that invoked the software that performed the permission change (optional)
- **clientUser**: Username of the credentials that authenticated against the software that performed the permission change (optional)
- **disabled**: An array of strings that hold the permissions that were effectively disabled
- **enabled**: An array of strings that hold the permissions that were effectively enabled
- **ipAddress**: The IP address of the client that performed the API operation to change permissions
- **reason**: The reason for the permission change, as indicated by the client that performed the API operation to change permissions (except when using the SOAP API)

Success Response - Flatten Format

A flat JSON array with each value being a JSON permission change object, augmented with an oxid and a brand attribute.

Body

```
[
  {
    "time": "2021-03-30T13:33:17.643Z",
    "authUser": "brand1",
    "disabled": [
      "RECEIVE",
      "SEND"
    ],
    "ipAddress": "10.10.1.10",
    "reason": "sendmail flooding detection",
    "oxid": "3@100",
    "brand": "brand1"
  },
  {
    "time": "2021-03-30T13:33:17.660Z",
    "authUser": "brand1",
    "disabled": [
      "MAILLOGIN",
      "WEBLOGIN"
    ],
    "ipAddress": "10.20.32.193",
    "reason": "DDoS abuse detection",
    "oxid": "3@100",
    "brand": "brand1"
  },
  {
    "time": "2021-03-30T13:33:17.671Z",
    "authUser": "brand1",
    "enabled": [
      "RECEIVE",
      "SEND",
      "MAILLOGIN",
      "WEBLOGIN"
    ],
    "ipAddress": "192.168.38.199",
    "reason": "Resolved by customer support",
    "clientIp": "82.193.93.112",
    "clientUser": "carla",
    "oxid": "3@100",
    "brand": "brand1"
  }
]
```

Fields

- **time**: An ISO-8601 formatted UTC string represents the point in time when the permission change was performed
- **oxid**: Unique identifier of the user, in the form <userId>@<contextId> (e.g. 3@21839)
- **brand**: The brand the user resides in
- **authUser**: The username used in the successful authentication when performing the permission change
- **clientIp**: IP address of the client that invoked the software that performed the permission change (optional)
- **clientUser**: Username of the credentials that authenticated against the software that performed the permission change (optional)
- **disabled**: An array of strings that hold the permissions that were effectively disabled
- **enabled**: An array of strings that hold the permissions that were effectively enabled
- **ipAddress**: The IP address of the client that performed the API operation to change permissions
- **reason**: The reason for the permission change, as indicated by the client that performed the API operation to change permissions (except when using the SOAP API)

Status

200

Error Codes

See: API Concepts for general error information

- **400**: Bad query request, most likely wrong query parameters