checkalt

KLIK LOCKBOX LOCKBOX API SPECIFICATIONS

VERSION: 1.4

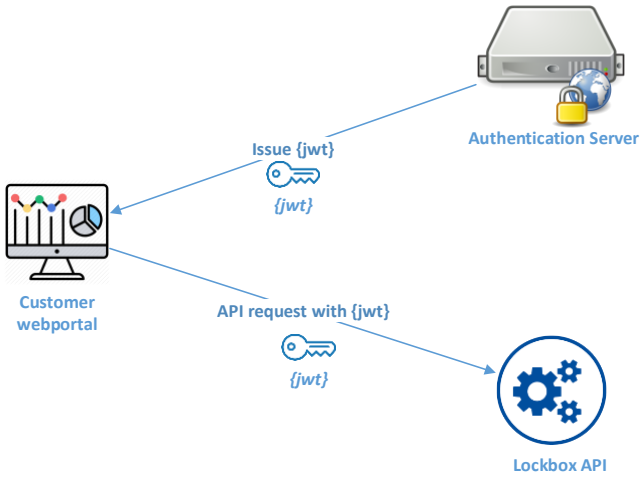
CONTENTS

API OVERVIEW

This document serves as the baseline specifications of the CheckAlt/Klik Open, Extract, Scan, and Re-association Application Programming Interface (LOCKBOX API). The LOCKBOX API is a RESTful based API, implemented for the purposes of extensibility, neutrality, and independence. It uses JSON Information Sets for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol Secure (HTTPs) for message negotiation and transmission.

Physical paper records shall be stored until the client accepts the image file as the official record. Upon acceptance, the paper records can be destroyed. Such acceptance could be in form of an elapsed period, set by client. Klik's lockbox data can be accessed through a set of APIs. The API is designed to be versioned. The initial version is 'v1', accessible through the below URL from the root:

ex: https://api.remithub.io/v1/...



LOCKBOX Lockbox API Architecture Diagram

Authentication Server: An API security service that issues a JWT token for authentication and authorization.

Customer web portal: Customers web portal which is consuming Lockbox API using JWT token for authentication and authorization.

Lockbox API: The LOCKBOX API is a RESTful based API, implemented for the purposes of extensibility, neutrality, and independence. It uses JSON Information Sets for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol Secure (HTTPs) for message negotiation and transmission.

JWT: JSON Web Token.

BASIC ORGANIZATION OF API

The API is organized into two parts: Summary API centers around the summary of items processed on a day per lockbox, Detail API focuses on details of the items that are processed like check, stub details etc… and Security API focus on authentication and authorization.

SUMMARY API

The Summary API is focused on summary, which are exposed as resources according to their types. The routes shown in the following tables are relative to the api root, e.g. the 'dailyitems' resources are accessible from http://<host>/web/v1/dailyitems.

Resource	Description	Route
DailyBatchItems	Daily items grouped by batch	/dailyitems/batch
DailyLockboxItems	Daily items grouped my lockbox	/dailyitems/lockbox
DailyMemberItems	Daily items grouped my member	/dailyitems/member

DETAILS API

The Details API is fine-grained focused on item details that are processed using klik lockbox processing systems.

Resource	Description	Route
Envelope	Envelope details.	/envelope
Image	Base64 encoded image.	/image
Check	Check details.	/Check
Document	Document details.	/Document

SECURITY

A JSON Web Token (JWT) is used for authentication and the API key in the payload details will be used for authorization. JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. We encourage customer to send a JWT token in below format.

HEADER
Algorithm & Token Type

```
{
    "alg" : "RSA256"
    "type" : "JWT"
}
```

+

PAYLOAD
Data

```
{
    "iis" : "issuer.com",
    "exp" : 1450626101",
    "sub": "subject",
    "aud": "APIKey",
 }
```

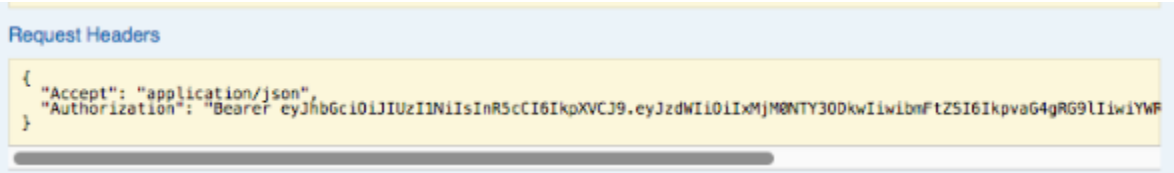
+

Signature Verification

```
Rsa-sha-256(
   base64encode(header) + "." + base64encode(payload), Private Key(RSA)
)
```

During the onboarding process we will share unique "aud" and "APIKey" values which should be included in the payload section of JWT. API call will be rejected if incorrect "aud" and "APIKey" are specified in the request. Whenever customers wants to access a protected route or resource (API), the request shoul send the JWT in the **Authorization** header using the **Bearer** schema. The content of the header should look like the following:

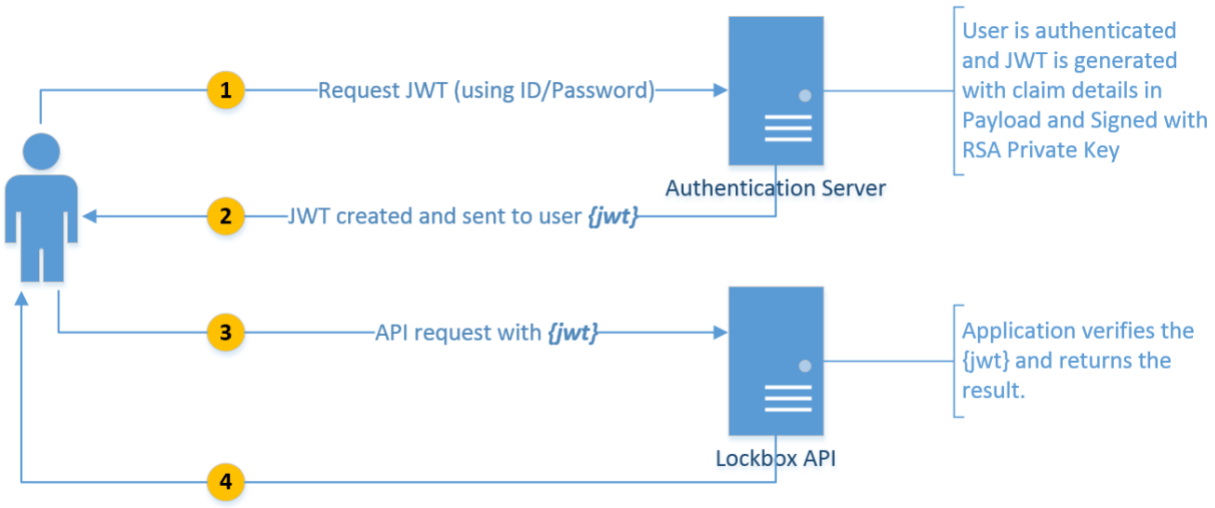
```
Authorization: Bearer <token>
```



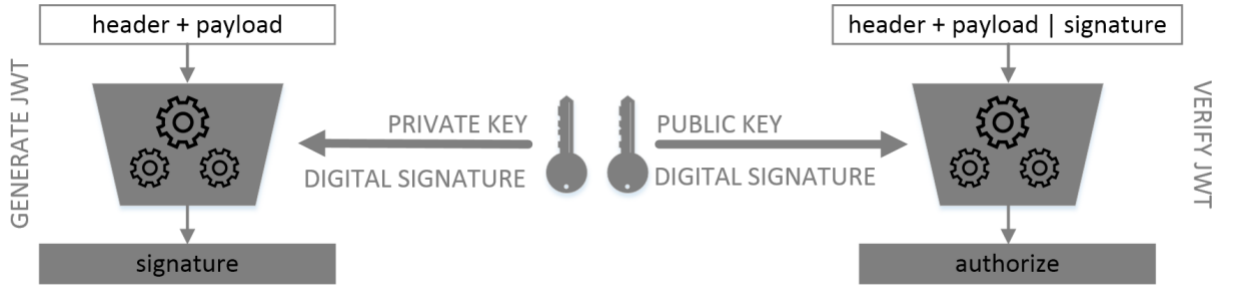
The API request should always be sent over HTTPS connection which prevents unauthorized users from stealing the sent JWT by making it so that the communication is not intercepted. Also, having an expiration in your JWT payload, a short one in particular, is important so that if old JWT ever get compromised, they will be considered invalid and can no longer be used.

JWT AUTHENTICATION FLOW:

The below diagram show the authentication flow for accessing the Lockbox API.



In below figure, the process of generating a signature is shown on the left. The method of verifying the signature is shown on the right.



The issuer of the token uses the private key for signing the JWT. This implies that the private key should be kept in a confidential location, only known to the issuer of the JWT tokens. The public key needs to be shared, so the consumer of the token can verify its integrity. The algorithms that generate such a signature are indicated with the "RS" prefix.

HOW TO REQUEST JWT?

"RequestToken" API call issues a JWT token. The JWT should be used in all LOCKBOX API calls for customer to authenticate and be authored. User Name, Password and API Key will be shared with the customer during onboarding process.

RequestToken Request Example:

C#:

```
var client = new RestClient("https://lockbox-test.checkalt-klik.com/api/Security/RequestToken
    ?UserName=remitapitest1&password=XXXXXX");
var request = new RestRequest(Method.POST);
request.AddHeader("cache-control", "no-cache");
request.AddHeader("Connection", "keep-alive");
request.AddHeader("content-length", "");
request.AddHeader("accept-encoding", "gzip, deflate");
request.AddHeader("Host", "lockbox-test.checkalt-klik.com");
request.AddHeader("Cache-Control", "no-cache");
request.AddHeader("User-Agent", "PostmanRuntime/7.15.0");
request.AddHeader("x-api-key", "2340A472-5F5B-4EB9-A46D-3CA84BC9390D");
request.AddHeader("accept", "application/json");
IRestResponse response = client.Execute(request);
```

cURL:

```
curl -X POST \
  'https://lockbox-test.checkalt-klik.com/api/Security/RequestToken?UserName
      =remitapitest1&password=XXXXXXXX' \
  -H 'Cache-Control: no-cache' \
  -H 'Connection: keep-alive' \
  -H 'Host: lockbox-test.checkalt-klik.com' \
  -H 'User-Agent: PostmanRuntime/7.15.0' \
  -H 'accept: application/json' \
  -H 'accept-encoding: gzip, deflate' \
  -H 'cache-control: no-cache' \
  -H 'content-length: ' \
  -H 'x-api-key: 2340A472-5F5B-4EB9-A46D-3CA84BC9390D'
```

Response:


```
{
   "accessToken": "bearer  eyJhbGciOiJodHRwOi8vd3d3LnczLm9yZy8yMDAxLzA0L3httbGVuYyNyc2Etb2F1cCIsImVuYyI6IkExMjhDQkMtSFMyNTYiLCJraWQiOiI0N0RCMTlGMzI0MzUyNzg0MjY3NkQ2MjEzNTgx
      QzhCNzlFMUU0NEZDIiwidHlwIjoiSldUIn0.CrlK8BIEU_2it4zDrOyX6NR11id5hiULXd6hi1pF5Nb38bPZDLKqg6GbRCm9cNMv4RSqmUP9AIk3WBcMPWns9FOhnNfW61jCWKK2pF2et_OIUjpmxvIIyEmXI9Me0v_a
      FtHmjbgUVTIUVwk4B1WDqcgeBxDmuZjgexxV4IevEkjW8WrXLN5ftay8LcOdw_f1ldCr4_F_F2yOd3GVNowZ4RZq2YCYj91VoDKWwIO-uZCI-Dh8UPd1Q6Sgqusz4vNgl4Isjn_bWSxE4ri96J9vZqaO8MHBCiIi
      -8f8IDYdTzr0iQBYaNiJuP22eLxhBL8dQipgGXqHgp_yo12I53xO5Q.RK9eZNvLgPiQA75YjmVYkw.2GUz_97uYgTSrwkDX4ESXER5HLRO0BPtHkhRyriIFpOJcooVZV
      -Ft4TwZPR38y2T_HAI0rQoA09Hb5Zu8Xnjp3vCyfQkF1wK3jay71I62WE347Wg5q39Emmy7o15Rh1cC7e2ygT1SCXPEFeGT15ci6vjmu2JehKDZq7
      -Q8Q8mzg9k2zmHnMuF2isNCk1BgM6Q7jFnm5Z0aLBzu5R1A5oewqFVfZWM2SwBRbysbjQz-thF2GzRpMJDFrqWRkDU9ktkv6wyg2Uqk0UrIFiNqhx92sVS9WRRPh7a3Wz2Of4l2Y5qP1cjXv9XTOdA-O5hdJkqRjx
      -2R1qJ_DLpLHutLK8vH4X1HTgU9xCXGjvQi-_I.X5XFwSGaIQiOyQj46iC1vA",
   "expiry": "2019-06-24T17:45:49.48992Z"
}
```


HOW TO ACCESS LOCKBOX API?

All LOCKBOX api calls required JWT and API Key in the header for authentication and authorization.

DailyItems/Batch Request example:

C#:

```
var client = new RestClient("https://lockbox-test.checkalt-klik.com/api/v1/DailyItems/Batch");
var request = new RestRequest(Method.GET);
request.AddHeader("cache-control", "no-cache");
request.AddHeader("Connection", "keep-alive");
request.AddHeader("accept-encoding", "gzip, deflate");
request.AddHeader("Host", "lockbox-test.checkalt-klik.com");
request.AddHeader("Cache-Control", "no-cache");
request.AddHeader("Accept", "*/*");
request.AddHeader("User-Agent", "PostmanRuntime/7.15.0");
request.AddHeader("Authorization", "Bearer
    eyJhbGciOiJodHRwOi8vd3d3LnczLm9yZy8yMDAxLzA0L3htbGVuYyNyc2Etb2FlcCIsImVuYyI6IkExMjhDQkMtSF
    MyNTYiLCJraWQiOiI0N0RCMTlGMzI0MzUyNzg0MjY3NkQ2MjEzNTgxQzhCNzlFMUU0NEZDIiwidHlwIjoiSldUIn0
    .e_uErKcrOjPHz4eNDKadxdnrVGLtAUJxf1BNgXWn8j29451fAtQsu07eURWlkrZFz3FqQ9R8z6Sw8R_pS
    -srorugVKF5gkfpWkDjz5oqxaX9A3ne_jIYmhB9pqgBpesBL0Tp33OQVtJli8BgD8rtVbka7LwBVRn2ca_UjtYCCjf
    IQle2PHza5XYiPWz5oR7cC0xRiYgKi75EjSJKDPS0J1NZy9RGTGW5C2rWlsJFRl3eC2dQ9nEEJONhk3mBCagTuIHdG
    8xlQXx4Xs9qIkpdyPq5XSDsgOYBElCL0YE4SLDe-oSQF7x3fYuyO2ZJZjmqAlY3TNvDswWBCaKxisdX8A
    .CreYl0swjAXh7AlnoS1UIw
    .hgWrOBJSS6ALY15yZCb8Ci_4puoShqTWoOS2PYadQgwJMbvypXN360gEY3lzQwblnM9xB3GAp1o3wGP-ID1xx
    -43ya_QJEghOeN2C4opTkOL3bMs128KaOj8_2xYPT8nDQDQFnzPwlVlop54qfCaSIHrfB_gUQ94m75KBvX8FylOCSO
    CE8vWtU3Z1NYvM2Ae0PzTT
    -nzYka_iDiQxbkZYQKIlersnXbSubkUjLImGRdgKOTF3ZynHlljRhVKs46c4Ziz03rDQBJjuQSiY111Jpa
    -zKaDJTBN13ymvDZvJdRotqMP5r9Wx5NXL85wLz0o5Unw5La--2TRa-F1b3kpVidQVaqpDte38PhUglLixIY
    .snCd7FYYn8fiaab6-CY2mg");
request.AddHeader("x-api-key", "2340A472-5F5B-4EB9-A46D-3CA84BC9390D");
IRestResponse response = client.Execute(request);
```

cURL:

```
curl -X GET \
  https://lockbox-test.checkalt-klik.com/api/v1/DailyItems/Batch \
  -H 'Accept: */*' \
  -H 'Authorization: Bearer
      eyJhbGciOiJodHRwOi8vd3d3LnczLm9yZy8yMDAxLzA0L3htbGVuYyNyc2Etb2FlcCIsImVuYyI6IkExMjhDQkMt
      SFMyNTYiLCJraWQiOiI0N0RCMTlGMzI0MzUyNzg0MjY3NkQ2MjEzNTgxQzhCNzlFMUU0NEZDIiwidHlwIjoiSldU
      In0.e_uErKcrOjPHz4eNDKadxdnrVGLtAUJxf1BNgXWn8j29451fAtQsu07eURWlkrZFz3FqQ9R8z6Sw8R_pS
      -srorugVKF5gkfpWkDjz5oqxaX9A3ne_jIYmhB9pqgBpesBL0Tp33OQVtJli8BgD8rtVbka7LwBVRn2ca_UjtYCC
      jfIQle2PHza5XYiPWz5oR7cC0xRiYgKi75EjSJKDPS0J1NZy9RGTGW5C2rWlsJFRl3eC2dQ9nEEJONhk3mBCagTu
      IHdG8xlQXx4Xs9qIkpdyPq5XSDsgOYBElCL0YE4SLDe-oSQF7x3fYuyO2ZJZjmqAlY3TNvDswWBCaKxisdX8A
      .CreYl0swjAXh7AlnoS1UIw
      .hgWrOBJSS6ALY15yZCb8Ci_4puoShqTWoOS2PYadQgwJMbvypXN360gEY3lzQwblnM9xB3GAp1o3wGP-ID1xx
      -43ya_QJEghOeN2C4opTkOL3bMs128KaOj8_2xYPT8nDQDQFnzPwlVlop54qfCaSIHrfB_gUQ94m75KBvX8FylOC
      SOCE8vWtU3Z1NYvM2Ae0PzTT
      -nzYka_iDiQxbkZYQKIlersnXbSubkUjLImGRdgKOTF3ZynHlljRhVKs46c4Ziz03rDQBJjuQSiY111Jpa
      -zKaDJTBN13ymvDZvJdRotqMP5r9Wx5NXL85wLz0o5Unw5La--2TRa-F1b3kpVidQVaqpDte38PhUglLixIY
      .snCd7FYYn8fiaab6-CY2mg' \
  -H 'Cache-Control: no-cache' \
  -H 'Connection: keep-alive' \
  -H 'Host: lockbox-test.checkalt-klik.com' \
  -H 'accept-encoding: gzip, deflate' \
  -H 'cache-control: no-cache' \
  -H 'x-api-key: 2340A472-5F5B-4EB9-A46D-3CA84BC9390D'
```

Response:


```
{
    "batchItems": [
        {
            "details": {
                "batchId": 7222612,
                "totalEenvelopesCount": 9,
                "totalAmount": 9891.33,
                "processDate": "2019-06-24T00:00:00",
                "acceptsCount": 7,
                "rejectsCount": 0,
                "exeptionsCount": 2
            },
            "related": {
                "envelopeIds": [
                    "7222612-1",
                    "7222612-2",
                    "7222612-3",
                    "7222612-4",
                    "7222612-5",
                    "7222612-6",
                    "7222612-7",
                    "7222612-8",
                    "7222612-9"
                ],
                "lockboxIds": [
                    "19444"
                ],
                "memberIds": [
                    "354"
                ]
            }
        },
        {
            "details": {
                "batchId": 10991691,
```

API STANDARD RESPONSE

Every method returns the following standard response:

```
{
        { ...the actual result(s) ... },
        { ...any related entities ids that are referred within 'results'... }
}
```

Note that the results will be returned as an array for ‘many’ requests, and as a scalar for ‘individual’ requests (such as when <id> is in URL). In the documentation of each method, generally only the ‘results’ part of the response structure are described.

RELATED SECTION

The ‘related’ section contains the array of the unique identifiers that are related to the entity. The purpose of this is to retrieve the fine-grained data of the related entities. From the below example, the related section contains all the envelope identifiers that are processed for a Lockbox.

Ex:


```
"batchItems": [
    {
        "details": {
            "batchId": 1001,
            "lockboxId": 1,
            "memberId": 11,
            "totalEenvelopesCount": 284,
            "totalAmount": 2478,
            "processDate": "2019-05-30T14:25:26.4932735-04:00",
            "acceptsCount": 48,
            "rejectsCount": 6,
            "exeptionsCount": 73
        },
        "related": {
            "envelopeIds": [
                "20190503E00011",
                "20190503E00012",
                "20190503E00013",
                "20190503E00014",
                "20190503E00015",
                "20190503E00016",
                "20190503E00017",
                "20190503E00018",
                "20190503E00019"
            ]
        }
    },
    {
        "details": {
            "batchId": 1002,
            "lockboxId": 2,
            "memberId": 12,
            "totalEenvelopesCount": 204,
            "totalAmount": 1564,
            "processDate": "2019-05-30T14:25:26.4932735-04:00",
            "acceptsCount": 168,
            "rejectsCount": 9,
            "exeptionsCount": 151
        },
        "related": {
            "envelopeIds": [
                "20190503E00021",
                "20190503E00022",
                "20190503E00023",
                "20190503E00024",
                "20190503E00025",
                "20190503E00026",
                "20190503E00027",
```

ENTITY RELATIONSHIP

API RESPONSE CODES

Below are the standard HTTP Response Codes.

Code	Status	Description
200	OK	The request was successfully completed.
201	Created	A new resource was successfully created.
204	No Content	Indicates that the request was accepted but that there was nothing to return.
400	Bad Request	One of the below: • The request was invalid. • The authentication information was not provided in the correct format. Verify the value of Authorization header. • The value provided for one of the HTTP headers was not in the correct format. • The HTTP verb specified was not recognized by the server. • One of the request inputs is not valid. • An invalid value was specified for one of the query parameters in the request URI. • The range specified is invalid for the current size of the resource. • The requested URI does not represent any resource on the server. • A required query parameter was not specified for this request. • A required HTTP header was not specified. • A query parameter specified in the request URI is outside the permissible range.
401	Unauthorized	The request did not include an authentication token or the authentication token was expired.
403	Forbidden	The client did not have permission to access the requested resource.
404	Not Found	The requested resource was not found.
405	Method Not Allowed	The HTTP method in the request was not supported by the resource. For example, the DELETE method cannot be used with the Agent API.
409	Conflict	The request could not be completed due to a conflict.
412	Precondition Failed	Precondition failed.
429	Too Many Requests	Too many request for rate limiting (future use)
500	Internal Server Error	The request was not completed due to an internal error on the server side.
503	Service Unavailable	The server was unavailable.