Riskhandler API

Consumer Documentation

Table of Contents

Introduction

The Riskhandler API is a RESTful API that has been developed using the Microsoft ASP.NET Web API framework.

Authentication is implemented using the Basic Authentication which requires an authentication token to be included in the headers of each request sent to the server.

All requests must be made over a secure HTTPS connection using a minimum of TLS 1.2. Requests over HTTP or using older version of TLS will not be accepted.

Pagination is implemented on some requests to reduce server load and improve performance for requests that return large amounts of data.

This document explains the authentication process, how pagination works and details the features that is currently available to consumers of the API.

Useful Resources

ASP.NET Web API - Microsoft

REST API - Mozilla Web Glossary

RESTful Web API Design – Microsoft

HTTP Authentication – Mozilla Web Glossary

Basic Authentication Scheme - RFC 7617

HTTP Request Methods – Mozilla Web Glossary

Transport Security Layer (TLS) - Wikipedia

Authentication

The Riskhandler API uses Basic Authentication which means the consumer needs to provide a valid API Key and a Client Secret to access the API endpoints. Access is also restricted using an IP Address allowlist. Each API Key has its own unique list of allowed IP addresses.

We have provided a simple method for managing your API Keys and Client secrets in the Riskhandler application.

API Keys

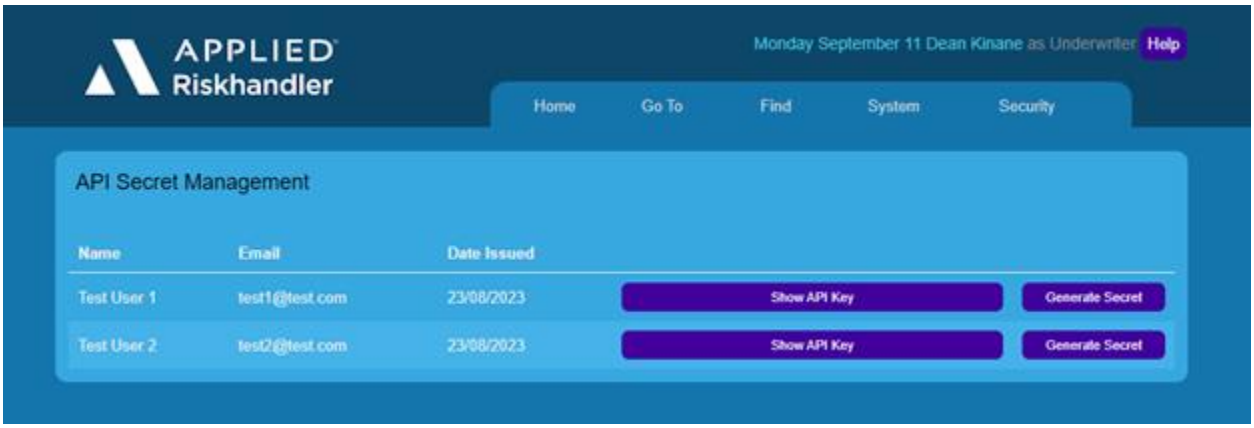
An API Key is akin to a username. When a request is sent to the API, the API Key identifies the system that sent the request.

New API Keys can only be created by the Riskhandler team and must be requested by sending an email to the support system at support@riskhandler.com.

Once the API Key has been created, a Riskhandler administrator user for the MGA can access the key and create a new Client Secret from the Riskhandler application, as described below.

Client Secrets

To make use of your API Key, you must also create a Client Secret. Your Client Secret is like a password and is only known by you. You can create one via the Riskhandler application by accessing the **Manage API Secrets** screen under the **Security** menu.



On this screen, you will see a list of active API Keys for your system. You can reveal the API Key at any time by clicking the **Show API Key** button.

To generate a Client Secret, click the **Generate Secret** button.

The Client Secret can only be viewed once.
It is very important to record your Client Secret in a secure location once it is created.

As soon as you close the window it can never be recovered. The Riskhandler team has no access to the Client Secret and cannot recover it for you.

If the secret is lost, a new one must be generated. Generating a new Client Secret will cause the old one to be immediately deactivated which means any further attempts to access the API using the old secret will fail.

Accessing the API

The Riskhandler API uses Basic Authentication to secure API access. This requires the **Authorization header** to be correctly set on the HTTP Request for every call to the API.

The header contains the authentication token which identifies the caller and, if accepted, grants access to the API resources. The header is a combination of the API Key and Client Secret, separated by a colon, which has been Base64 encoded.

The header must be set to the following:

Authorization: Basic <token>

Where <token> is:

Base64Encode(<APIKey>:<ClientSecret>)

For example, given the following credentials:

API Key: abc123
Client Secret: xyz789

The unencoded token would be:
abc123:xyz789

The resulting complete header would be:
Authorization: Basic YWJjMTIzOnh5ejc4OQ==

If this header is correctly set on each API call, from an authorised IP address, you will be able to successfully access the API resources.

Pagination

In order to reduce server load and improve performance, pagination has been implemented on many of the API endpoints where large amounts of data can be requested.

Pagination, or paging, helps to reduce server load by breaking up requests for large amounts of data into smaller more manageable pieces referred to as pages.

For paged requests, a parameter called **page** is passed along with the HTTP request, starting from page 1 for the initial request. If the page parameter is omitted, page 1 will be returned by default.

Every successful response from a paginated request will include an **X-Pagination** header which includes details of the pagination state. This is in the form of a JSON object with the following properties:

TotalCount: Total number of records for the request.
PageSize: The size of each page of results.
TotalPages: The total number of pages for the request.
HasNextPage: True is there is another page to be retrieved, false if not.

Sample X-Pagination Header Value:

```
{
  "TotalCount": 490,
  "PageSize": 50,
  "TotalPages": 10,
  "HasNext": true
}
```

API Features

Get Live Products

Returns a list of all active products in the system.

URL
/api/products/live

HTTP Request Method
GET

Request Parameters
None

Response Codes
200 – Request Succeeded
401 – Unauthorised (invalid credentials)

Sample JSON Response

```
[
  {
    "ProductID": "00000000-0000-0000-0000-000000000000",
    "ProductName": "string",
    "Version": 0,
    "CaptureID": "00000000-0000-0000-0000-000000000000"
  }
]
```

Get Transactions for a Product by Date Range

Returns a list of all transactions for the specified product in the given date range. This is a paginated request, as described in the Pagination section earlier in this document.

URL
/api/products/*<product-id>*/transactions

Where *<product-id>* is the ID of a product, which can be retrieved from the Get Live Products method.

HTTP Request Method
GET

Request Parameters

Parameter Name	Description	Format	Example
fromDate	Start of date range	ISO 8601	2023-01-31
toDate	End of date range	ISO 8601	2023-01-31
page	Page being requested	Unsigned integer	1

Sample Request URL
api/products/1264e2f9-8a18-4b28-8d97-91b8836e3ae8/transactions?fromDate=2023-03-01&toDate=2023-09-10&page=1

Response Codes
200 – Request Succeeded
401 – Unauthorised (invalid credentials)
400 – Bad Request (malformed request, parameter format likely incorrect)

Sample JSON Response


```
[
  {
    "PolicyNumber": "string",
    "TransactionType": "string",
    "DateTransacted": "2023-09-11T08:59:24.816Z",
    "IsVoid": true,
    "AccountsTransaction": {
      "SequenceNumber": 0,
      "PaymentStatus": "string",
      "DateEffective": "2023-09-11T08:59:24.816Z",
      "DateStatement": "2023-09-11T08:59:24.816Z",
      "TotalClientDebit": 0,
      "BrokerFee": 0,
      "MgaFee": 0,
      "InsurerFee": 0,
      "GrossIncLevy": 0,
      "Levy": 0,
      "GrossExLevy": 0,
      "BrokerCommission": 0,
      "MgaCommission": 0,
      "NettToMga": 0,
      "NettToInsurer": 0
    },
    "RiskData": [
      {
        "TableName": "string",
        "Properties": {
           "Property1Name": "string",
           "Property2Name": 0
        }
      }
    ],
    "Endorsements": [
      {
        "Code": "string",
        "Wording": "string",
        "Summary": "string"
      }
    ]
  }
]
```


Get Latest Transactions for a Product

Returns a list of all transactions for the specified product that have not previously been retrieved for the API Key making the request.

This follows the same principle as a paginated request, but the page need not be specified as the next page of data that has not previously been retrieved will always be provided by default. The X-Pagination header is still provided to give context on how many records and pages remain to be accessed.

URL
/api/products/<*product-id*>/transactions/latest

Where <*product-id*> is the ID of a product, which can be retrieved from the Get Live Products method.

HTTP Request Method
GET

Request Parameters
None

Sample Request URL
api/products/1264e2f9-8a18-4b28-8d97-91b8836e3ae8/transactions/latest

Response Codes
200 – Request Succeeded
401 – Unauthorised (invalid credentials)
400 – Bad Request (malformed request, parameter format likely incorrect)

Sample JSON Response


```
[
  {
    "PolicyNumber": "string",
    "TransactionType": "string",
    "DateTransacted": "2023-09-11T08:59:24.816Z",
    "IsVoid": true,
    "AccountsTransaction": {
      "SequenceNumber": 0,
      "PaymentStatus": "string",
      "DateEffective": "2023-09-11T08:59:24.816Z",
      "DateStatement": "2023-09-11T08:59:24.816Z",
      "TotalClientDebit": 0,
      "BrokerFee": 0,
      "MgaFee": 0,
      "InsurerFee": 0,
      "GrossIncLevy": 0,
      "Levy": 0,
      "GrossExLevy": 0,
      "BrokerCommission": 0,
      "MgaCommission": 0,
      "NettToMga": 0,
      "NettToInsurer": 0
    },
    "RiskData": [
      {
        "TableName": "string",
        "Properties": {
           "Property1Name": "string",
           "Property2Name": 0
        }
      }
    ],
    "Endorsements": [
      {
        "Code": "string",
        "Wording": "string",
        "Summary": "string"
      }
    ]
  }
]
```