Krestfield PKCloud

RESTful API Specifications

version 2.0

Overview

This document details the REST API (Representational State Transfer Application Interface) calls that can be made to PKCloud in order to process requests (generation and verification of signatures etc.)

Calls to the REST API are web service calls, utilising http POST verbs with payloads formatted as JSON (JavaScript Object Notation)

Responses will return a standard http status code (e.g. 200 – OK, 401 – Unauthorized, 400 – Bad request) and data is returned as a JSON formatted payload

Return Codes

401 – Unauthorized

This will be returned when the account details sent were incorrect (e.g. invalid username or password)

You will also see this error if the authCode provided has expired. In this case the following error message will also be returned "`Auth key has expired. Please re-authenticate`"

400 – Bad Request

The message body (e.g. JSON format) was not correct or not recognised or a field within the JSON request body was not expected

A message may be returned in the body indicating the error e.g.:

```
{
    "authCode": null,
    "authenticatedOk": false,
    "validTo": null,
    "failureReason": "Failed to authenticate. The message body could not be parsed.  Format
is: { \"username\" : \"<username>\", \"password\" : \"<user password>\" }"
}
```

500 – Server Error

An internal error occurred e.g. a server instance not responding or an HSM has gone down etc.

200 - OK

If no error code is returned the usual success code of 200 will be returned, indicating the call completed successfully

Note: Should there be an error in the signature validation process, this code will still be returned (as the call completed successfully) but the response will show success = false and the full failure reason will be provided

API Specification

To use the REST API the Server's Client Interface must be configured as **RESTful** (and not **KrestfieldClient**)

All calls for operations (such as sign data) require that the caller has been authenticated

Authentication must be performed by a user in the **API** group as configured in the Management Console

Calls should be directed to the URL hosting the PKCloud system at the port specified in the server configuration

This specification details version 1.0 of the REST API specification. The URL to direct traffic to will therefore take the following form:

```
https://[Server IP Address]:[Server Port]/v1.0/[Request]
```

In the specifications given below, this part of URL:

```
https://[Server IP Address]:[Server Port]/v1.0
```

will be represented as **[URL]**

The URL will use https only if the **Use TLS** option has been checked in the Server configuration. If this has not been checked use http

`[Server IP Address]`
This will be as configured in the Server configuration for the **Server API Interface**, **IP Address**

`[Server Port]`
This will be as configured in the Server configuration for the **Server API Interface**, **Port**

`[Request]`
This can be any one of the following:

- authenticate
- genrandom
- signdata
- verifyp7signature
- verifysignature
- encrypt
- decrypt

Some sample URLs are given below:

```
https://127.0.0.1:5656/v1.0/authenticate
```

This indicates TLS is being used (https) and the server is local (127.0.0.1), listening on port 5656 and this is a request to authenticate

```
http://pkcloud.company.com/v1.0/signdata
```

In this example, a DNS entry has been configured to point to the EzSign server (pkcloud.company.com) and the server is listening on the default port 80. The is a request to sign data

Authenticate

Authentication to the API is with a username and password of a user in the API group that must have been configured via the Management Console

If verification is successful an `authCode` will be returned (a Base 64 encoded random number) which must then be provided in the `x-pkcloud-auth-code` header for further API calls, to indicate the caller was successfully authenticated and this *session* is still valid

The `authCode` has a validity (the value of this is set as the **API Key Validity** for the user in the Management Console). This value indicates the maximum permitted time (in seconds) between API calls before the API Key will be invalidated. Note that this validity period is refreshed for each API call, so if calls continue to be made the API Key will not expire

The IP address of the client is also linked to the returned `authCode` meaning that each client must authenticate individually and the returned `authCode` cannot be shared. If the same `authCode` is used from another client (that did not successfully authenticate), the call will be rejected

Request

```
POST: [URL]/authenticate
```

e.g.

```
http://127.0.0.1:5656/v1.0/authenticate
```

HEADERS

```
Content-Type=application/json
```

BODY

```
{
      "username" : "username",
      "password" : "password"
}
```

Response

If the provided username and password are correct the following will be returned

Status Code

```
200 : OK
```

BODY

```
{
    "authCode": "[Base64 Auth Code]",
    "authenticatedOk": true,
    "validTo": "[Zulu Timestamp]",
    "failureReason": null
}
```

e.g.

```
{
    "authCode": "KnRWiGm+53ddnKP64iVammQY0z/zG6g73uHztPrAHnM=",
    "authenticatedOk": true,
    "validTo": "2018-11-07T15:29:28.125Z",
    "failureReason": null
}
```

The returned `authCode` must be passed in the `x-pkcloud-auth-code` header for future API calls (see below)

If the details provided do not authenticate, a message such as the following will be returned

Status Code

```
    401 : Unauthorized
```

BODY

```
    {
        "authCode": null,
        "authenticatedOk": false,
        "validTo": null,
        "failureReason": "Failed to authenticate"
    }
```

Generate Random Number

Generates the specified number of random bytes. The channel name must be provided as well as the number of bytes to generate

Request

```
POST: [URL]/genrandom
```

e.g.

```
http://127.0.0.1:5656/v1.0/genrandom
```

HEADERS

```
Content-Type=application/json
x-pkcloud-auth-code=[authCode Returned from authenticate]
```

e.g.

```
x-pkcloud-auth-code=KnRWiGm+53ddnKP64iVammQY0z/zG6g73uHztPrAHnM=
```

BODY

```
{
  "channel" : "[Channel Name]",
  "numBytes" : [Num Bytes]
}
```

e.g.

```
{
  "channel" : "CHANNEL1",
  "numBytes" : 20
}
```

Response

BODY

```
{
    "success": [true | false],
    "failureReason": [null | error details],
    "randomBytes": "[Random Bytes]"
}
```

e.g.

```
{
    "success": true,
    "failureReason": null,
    "randomBytes": "AQIDBAUGBwgJAAECAwQFBgcICQA="
}
```

Sign Data

Generates a signature over the provided data in the format as configured on the server (i.e. PKCS#7 or PKCS#1)

Request

```
POST: [URL]/signdata
```

e.g.

```
http://127.0.0.1:5656/v1.0/signdata
```

HEADERS

```
Content-Type=application/json
x-pkcloud-auth-code=[authCode Returned from authenticate]
```

BODY

```
{
  "channel" : "[Channel Name]",
  "dataToSign" : "[Base64 data]",
  "isDigest" : [true | false]
}
```

e.g.

```
{
  "channel" : "CHANNEL1",
  "dataToSign" : "AQIDBAUGBwgJAQECAwQFBgcICQE=",
  "isDigest" : true
}
```

`[Base64 data]` is the data to be signed, encoded as Base64
`isDigest` must be true if the `[Base64 data]` has already been hashed

Note: It is recommended to hash data before sending to reduce the amount of traffic that must be transmitted over the network

Response

BODY

```
{
  "success" : [true | false],
  "failureReason" : [null | error details],
  "signature" : "[Base64 Signature]"
}
```

e.g.

```
{
  "success" : true,
  "failureReason" : null,
  "signature" : "MIIM5QYJKo... 4hpEkPU="
}
```

If `success = false`, the `failureReason` field will contain the error details

Verify PKCS7 Signature

Verifies a PKCS#7 signature performing path building and revocation checking as configured at the server

Request

```
        POST: [URL]/verifyp7signature
```

e.g.

```
        http://127.0.0.1:5656/v1.0/verifyp7signature
```

HEADERS

```
        Content-Type=application/json
        x-pkcloud-auth-code=[authCode Returned from authenticate]
```

BODY

```
        {
          "channel" : "[Channel Name]",
          "signature" : "[Base64 Signature]",
          "content" : "[Base64 data]",
          "isDigest" : [true | false],
          "byPassRevocationCheck" : [true | false],
          "byPassPathBuild" : [true | false],
        }
```

e.g.

```
        {
          "channel" : "CHANNEL1",
          "signature" : "MIIM5QYJKo... 4hpEkPU=",
          "content" : "AQIDBAUGBwgJAQECAwQFBgcICQE=",
          "isDigest" : true,
          "byPassRevocationCheck" : false,
          "byPassPathBuild" : false
        }
```

`[Base64 data]` is the data (content) to be verified, encoded as Base64
`isDigest` must be true if the `[Base64 data]` has already been hashed

Note: It is recommended to hash data before sending to reduce the amount of traffic that must be transmitted over the network

`byPassRevocationCheck` and `byPassPathBuild` are optional fields

If `byPassRevocationCheck = true`, no revocation checking will be performed, even if enabled at the server

If `byPassPathBuild = true`, no path building will be performed

Note:

- If path building is not performed only the integrity of the signature will be checked, the signature will not be verified as from a trusted source
- If revocation checking is not performed there is a risk the signing certificate (and others in the chain) may have been revoked and the signature will therefore not be fully verified

Response

BODY

```
{
  "success" : [true | false],
  "failureReason" : [null | error details]
}
```

e.g.

```
{
    "success" : true,
    "failureReason" : null
}
```

If `success = false`, the `failureReason` field will contain the error details for example:

```
"Verification Exception: The supplied digest did not match the signature
digest data"

"Path Exception: The signer certificate CN=Test,O=Dev,C=GB does not have
the digital signature key usage set.  Set relaxAllCertExtensionChecks=true
to allow this"
```

Verify Signature

Verifies a raw (or PKCS#1 if RSA) signature. A raw signature does not contain any certificates, therefore to be able to verify the signature, at least the signer certificate must be provided. Other certificates in the chain may also be required if they are not already installed in the channel

Request

```
POST: [URL]/verifysignature
```

e.g.

```
http://127.0.0.1:5656/v1.0/verifysignature
```

HEADERS

```
Content-Type=application/json
x-pkcloud-auth-code=[authCode Returned from authenticate]
```

BODY

```
{
  "channel" : "[Channel Name]",
  "signature" : "[Base64 Signature]",
  "content" : "[Base64 data]",
  "isDigest" : [true | false],
  "signerCert" : "[Base64 Certificate Data]",
  "otherCerts" : ["[Base64 Certificate Data]", "[Base64 Certificate Data]"],
  "byPassRevocationCheck" : [true | false],
  "byPassPathBuild" : [true | false]
}
```

e.g.

```
{
  "channel" : "CHANNEL1",
  "signature" : "MIIM5QYJKo...4hpEkPU=",
  "content" : "AQIDBAUGBwgJAQECAwQFBgcICQE=",
  "isDigest" : true,
  "signerCert" : "MIIFsjCCA...OomYw==",
  "otherCerts" : ["MIIE7TCC...vZz9po="],
  "byPassRevocationCheck" : false,
  "byPassPathBuild" : false
}
```

`[Base64 data]` is the data (content) to be verified, encoded as Base64
`isDigest` must be true if the `[Base64 data]` has already been hashed

Note: It is recommended to hash data before sending to reduce the amount of traffic that must be transmitted over the network

`signerCert` must be provided as the Base64 certificate data (not including the `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` header and footer). This field is required

`otherCerts` may be provided. If other certificates in the chain (intermediate CA certificates) are already held within the channel's key store, this field will not be required to build the complete path.
If provided it must be specified as a String array, each string being the Base64 certificate data (not including the certificate headers and footers mentioned above)

`byPassRevocationCheck` and `byPassPathBuild` are optional fields

If `byPassRevocationCheck = true`, no revocation checking will be performed even if enabled at the server

If `byPassPathBuild = true`, no path building will be performed

Note:
- If path building is not performed only the integrity of the signature will be checked, the signature will not be verified as from a trusted source
- If revocation checking is not performed there is a risk the signing certificate (and others in the chain) may have been revoked and the signature will therefore not be fully verified

Response

BODY

```
{
  "success" : [true | false],
  "failureReason" : [null | error details]
}
```

e.g.

```
{
    "success" : true,
    "failureReason" : null
}
```

If `success = false`, the `failureReason` field will contain the error details for example:

```
“Verification Exception: The supplied digest did not match the signature
digest data”

“Path Exception: The signer certificate CN=Test,O=Dev,C=GB does not have
the digital signature key usage set.  Set relaxAllCertExtensionChecks=true
to allow this”
```

etc.

Encrypt

Encrypts data with the key specified. Data must be provided as a Base64 encoded string. The channel must be an Encryption channel (this call will fail on a PKI channel). Keys must have been generated on the channel

Request

```
        POST: [URL]/encrypt
e.g.
        http://127.0.0.1:5656/v1.0/encrypt
```

HEADERS

```
        Content-Type=application/json
        x-pkcloud-auth-code=[authCode Returned from authenticate]
```

BODY

```
        {
          "channel" : "[Channel Name]",
          "dataToEncrypt" : "[Base64 data]",
          "keyLabel" : "[Key Label]"
        }
e.g.
        {
          "channel" : "CHANNEL1",
          "dataToEncrypt" : "SGVsbG8=",
          "keyLabel" : "key1"

        }
```

`[Base64 data]` is the data to be encrypted, encoded as Base64

If `keyLabel` is omitted and a default key has been set on the server, the default key will be used. If no default key has been configured this field is required and the call will fail if not provided

Response

BODY

```
        {
          "success" : [true | false],
          "failureReason" : [null | error details],
          "encryptedData": "[Base64 encrypted data]"

        }
e.g.
        {
            "success": true,
            "failureReason": null,
            "encryptedData": "df6OASBSUZq5+...Lf28sjjPSZ"
        }
```

`[Base64 encrypted data]` is the provided `dataToEncrypt` encrypted under the key referenced by `keyLabel`

Decrypt

Decrypts data with the key specified. Encrypted data must be provided as a Base64 encoded string. The channel must be an Encryption channel (this call will fail on a PKI channel). Keys must have been generated on the channel

Request

```
        POST: [URL]/decrypt
```

e.g.

```
        http://127.0.0.1:5656/v1.0/decrypt
```

HEADERS

```
        Content-Type=application/json
        x-pkcloud-auth-code=[authCode Returned from authenticate]
```

BODY

```
        {
          "channel" : "[Channel Name]",
          "encryptedData" : "[Base64 encrypted data]",
          "keyLabel" : "[Key Label]"
        }
```

e.g.

```
        {
          "channel" : "CHANNEL1",
          "encryptedData" : "df6OASBSUZq5+...Lf28sjjPSZ",
          "keyLabel" : "key1"

        }
```

`[Base64 encrypted data]` is the Base64 encoded encrypted data

If `keyLabel` is omitted and a default key has been set on the server, the default key will be used. If no default key has been configured this field is required and the call will fail if not provided

Response

BODY

```
        {
          "success" : [true | false],
          "failureReason" : [null | error details],
          "clearData": "[Base64 clear data]"

        }
```

e.g.

```
        {
            "success": true,
            "failureReason": null,
            "ClearData": "SGVsbG8="
        }
```

`[Base64 clear data]` is the provided `encryptedData` decrypted under the key referenced by `keyLabel`