IDA Interface Specification

Copyright Coelition 2015 Draft Specification Version V01

Monday, 27 July 2015

# Introduction

The Identity Authority (IDA) provides a service that generates and subsequently validates a digitally signed unique Pseudonymous Key to be used in signup to Data Engine services. The IDA does not require any input to generate the Pseudonymous Key.

Section 2 of this document describes how the IDA API is used by Operators and Data Engines to register Consumers or Devices. Section 3 gives details on the API itself.

# Protocol Overview

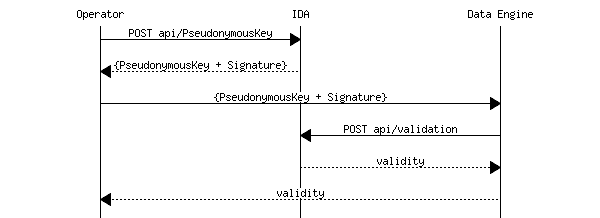


Figure : IDA/Data Engine signup sequence

Figure 1 shows the sequence an Operator must follow in signing up a new consumer: obtain a Pseudonymous Key from IDA and then use it to signup with the Data Engine. The Pseudonymous Key is used in all subsequent communications with the Data Engine such as sending data, requesting reports. For each new consumer, the Operator and Data Engine use a separate Pseudonymous Key. Applications that generate input (Atoms) for the Data Engine also use the Pseudonymous Key.

The signature is used so that the Data Engine can be assured that the Pseudonymous Key is genuine. Rather than using asymmetric key-pairs and distributing a public key and signing algorithm, the IDA provides the means for a receiver of a signed Pseudonymous Key to validate its signature. The Data Engine is not mandated to use this validation mechanism, since the Pseudonymous Key is plain text. The Data Engine may choose to trust the Operator or may pass the signed Pseudonymous Key to the IDA for validation.

It is assumed that this transaction is short - Operators only request Pseudonymous Keys when they need them and register them shortly afterwards (probably within minutes). The Identity Authority needs to be free to alter the means of signature (if for example it believes the mechanism used internally has been revealed). If this change happens during a transaction then validation will fail. This is an unlikely event, but parties in the transaction need to be able to manage it:

* Data Engines receiving a failed validation code must pass the failure back to the Operator
* Operators receiving a failed validation code from the Data Engine must discard the Pseudonymous Key and request a new one from the IDA.
* If the second attempt also fails, the Operator should try once more after a short delay (1-2 seconds) before aborting the attempt to register.

Separate Data Engines will have non-intersecting sets of Pseudonymous Keys. The same consumer will have different Pseudonymous Keys in different Data Engines. There is no possible way to identify the consumer from the Pseudonymous Key. There is no means by which data from the same consumer in several Data Engines can be combined without going back to the holder of the directly identifying personal information (Operator) and finding all of the Pseudonymous Keys that are used for that individual. The Operator would be required to seek specific consent from the Consumer if that was to be done.

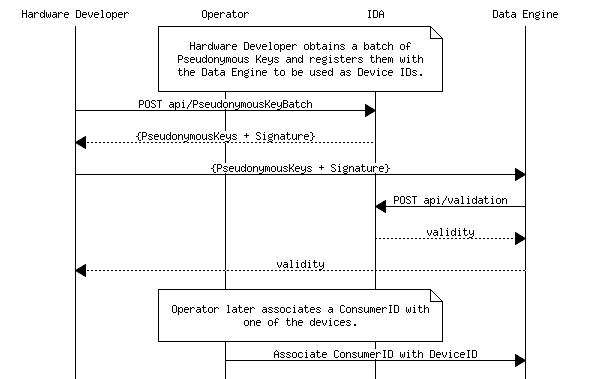


Figure : Hardware Developer registering a batch of DeviceIDs

The IDA also provides a means to generate a batch of up to 1000 Pseudonymous Keys in one request. The batch contains a single signature and the same protocol must be followed for validation: The Hardware Manufacturer passes the batch to the Data Engine which can choose to validate the batch with the IDA. It is expected that the batch transaction will be used to generate Pseudonymous Keys to be used in devices.

Pseudonymous Keys are primarily intended to represent a consumer in the ecosystem. However, Operators and Service Providers also require keys to identify themselves in their machine to machine interactions. IDA generated Pseudonymous Keys may be used for this purpose since they are devoid of DIPI and unique across the ecosystem.

# API Overview

## Introduction

The IDA API provides a means for Operators to generate unique Pseudonymous Keys for Consumers or Devices. A Pseudonymous Key is required when an Operator registers a consumer or device with a Data Engine. Pseudonymous Keys are digitally signed so that Data Engines can validate them to ensure they were generated by IDA and have not been altered.

## Authentication

To access the IDA API, callers need access credentials with two components:

* A userid to identify the caller.
* A password for authentication.

Each userid is assigned one of two roles:

* *Generator* Allowing the userid to generate Pseudonymous Keys
* *Validator* Allowing the userid to validate Pseudonymous Keys

Access credentials are generated by the Identity Authority.

HTTP basic authentication is used to authenticate calls to the API. Passwords are 64 bytes in length and supplied as a base 64 encoding string. This should be converted to ASCII and prefixed with the userid followed by a colon to form the token passed in the HTTP Authorisation Header.

Example:

“9abf5386-2ac6-4e61-abc4-6b809a85d6cb:J1dOeWJJOkd3akhnSn4ma007M  
DtUMVAxISgyOn9jI2U9NHNdRi4hfiw9c2I8PURcVltNMWQkamsrfGR4T24vKA==”

If the userid is unrecognised, or the wrong password is supplied a HTTP status code *401 Invalid username or password* is returned.

If a request is made with a userid that assigned a role that is not authorised to perform that action then the HTTP status code *403 Unauthorised* is returned.

## PseudonymousKey endpoint

The PseudonymousKey end-point provides a means to generate Pseudonymous Keys. It is only accessible to users whose access credentials have the Generator role.

| **API** | **Description** |
| --- | --- |
| POST  api/PseudonymousKey | Generate a new signed Pseudonymous Key for a consumer. The mechanism used to sign the response is periodically changed, so the response should be passed to the Data Engine shortly after generation or validation may fail. |

### Response

The response is a packet containing three entities: The Pseudonymous Key; the timestamp at which it was generated; and a signature that can be used for validation.

**Media type:**

application/json, text/json

**Sample:**

{  
 "PseudonymousKey": "00000000-0000-0000-0000-000000000000",  
 "TimeStamp": "2011-02-14T00:00:00",  
 "Signature":  
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="  
}

## PseudonymousKeyBatch endpoint

The PseudonymousKeyBatch end point provides a means to generate a batch of Pseudonymous Keys in one response packet. It is only accessible to users whose access credentials have the Generator role.

| **API** | **Description** |
| --- | --- |
| [POST api/PseudonymousKey](http://demogift.tessella.co.uk/Help/Api/POST-api-Identifier)Batch | Generate a new signed batch of Pseudonymous Keys. The mechanism used to sign the response is periodically changed, so the response should be passed to the Data Engine shortly after generation or validation may fail. |

### Request

| **Parameter Name** | **Description** | **Additional information** |
| --- | --- | --- |
| **batchAttributes** | Packet defining the attributes of the batch. If this is invalid then a status code of 400 will be returned. Batches up to 1000 are supported. | Define this parameter in the request **body**. |

**Media type:**

application/json, text/json

**Sample:**

{"Size": 1000}

### Response

A packet containing: an array of Pseudonymous Keys; the timestamp at which the batch was generated; and a signature that can be used for validation.

**Media type:**

application/json, text/json

**Sample:**

{  
 "PseudonymousKeys": [  
 "00000000-0000-0000-0000-000000000000",  
 "00000000-0000-0000-0000-000000000001",  
 "00000000-0000-0000-0000-000000000002"]  
 "TimeStamp": "2011-02-14T00:00:00",  
 "Signature":  
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="  
}

## Validation endpoint

The Validation end point provides a means to validate Pseudonymous Keys to ensure they were generated by IDA. The same endpoint is used to validate a single Pseudonymous Key or a batch. It is only accessible to users whose access credentials have the Validator role.

| **API** | **Description** |
| --- | --- |
| [POST api/Validation](http://demogift.tessella.co.uk/Help/Api/POST-api-Validation) | Validates a single signed Pseudonymous Key or a signed batch of Pseudonymous Keys to ensure that they were generated by IDA. The mechanism used for signing is periodically changed. If validation fails, the caller should request that the Operator requests a new Pseudonymous Key. |

### Request

| **Parameter Name** | **Description** | **Additional information** |
| --- | --- | --- |
| **signedPseudonymousKey** | The packet originally passed by the Operator. If this is invalid then a status code of 400 will be returned. | Define this parameter in the request **body**. |

**Media type:**

application/json, text/json

**Sample 1 (Single input):**

{  
 "PseudonymousKey": "00000000-0000-0000-0000-000000000000",  
 "TimeStamp": "2011-02-14T00:00:00",  
 "Signature": "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="  
}

**Sample 2 (Batch input):**

{  
 "PseudonymousKeys": [  
 "00000000-0000-0000-0000-000000000000",  
 "00000000-0000-0000-0000-000000000001",  
 "00000000-0000-0000-0000-000000000002"]  
 "TimeStamp": "2011-02-14T00:00:00",  
 "Signature":  
"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="  
}

### Response

Returns true if the Pseudonymous Key or batch matches its signature, false otherwise.

**Media type:**

application/json, text/json

**Sample:**

true