# Identifying Account on Requests to Service API

Incoming requests may be intended by the client to access global resources or resources associated with a specific account. Examples of global resources are requests to the API root to obtain version information or to the version-specific root to obtain the description file in WADL format.

For requests coming in to the Service API targeted at a specific account or resource within an account, the request will contain the account string as part of the URI. The URI should consist of three components; the API version string, the account string, and the resource locator.

**Example 1. An API Request**

GET /v1.0/A1000/widgets/

In this example, a request is being made to retrieve a list of available resources (widgets in this case) under account A1000 following the version 1.0 contract for the API.

The management of accounts is not exposed in the Service API by default as that is considered an administrative task, and should be exposed on the Admin API only. Therefore, requests to create new accounts or access non-existing ones should fail.

**Example 2. Attempt to create an Account from the Service API**

PUT /v1.0/B2000

**Sample XML Error Response**

<unauthorized xmlns="http://api.openstackoperator.com/v1.0/B2000/"

code="401">

<message>Not Authorized</message>

</unauthorized>

**Sample JSON Response**

{

"unauthorized" : {

"code" : 401,

"message" : "Not Authorized"

}

}

**Example 3. Attempt to access a non-existent Account or an Account belonging to another tenant from Service API**

GET /v1.0/C1000

**Sample XML Error Response**

<itemNotFound xmlns="http://api.openstackoperator.com/v1.0/C1000/"

code="404">

<message>Not Found</message>

</itemNotFound>

**Sample JSON Response**

{

"itemNotFound" : {

"code" : 404,

"message" : "Not Found"

}

}

In the case of an attempt to access an account without the correct authorization, it is recommended to return an item not found (404) error instead of an unauthorized (401) error to not reveal to the caller that an account does in fact exist with that name.

## Use Case: Account lifecycle from Service Developers’ Perspective

1 – A REST request to create an account is received from the Admin API and/or from a client authorized to create accounts. The request contains the account name, which is an arbitrary string. Any necessary operations to provision the account are performed.

2 – A REST request from a client comes in from the Service API identifying the account in the URI. The validity of the account string and the authorization of the requestor to perform the requested operation are confirmed before performing the operation.

3 – Relevant operations and usage metrics are logged along with the account string using the logging tools of the developer’s choice.

## Use Case: Account lifecycle from Operator’s Perspective

1 – A request to provision an OpenStack cloud resource is received and contains the relevant information such as the resource type (and relevant attributes), the identity of the user(s) which includes the information about which tenant they belong to (this could be an ISP’s customer, an employee’s cost center, a hosting provider’s reseller, etc…).

2 – The operator generates or assigns a unique string to identify the account under which to create the resources.

3 – The operator creates the account using the Admin API of the deployed OpenStack service.

4 – The users are provided their OpenStack account identifier (potentially only as part of a URL).

5 – The operator collects usage logs from the service, aggregates and calculates necessary usage metrics, and charges back usage to the tenant.