OAuth - Password Grant Type

OAuth grants

Grant Type	Typical Use Case	Complex?
No specific resource owner is involved		
Client Credentials	Business system interactions, where resources being operated on are owned by the partner, not a particular user	No
A specific resource owner is involved		
Resource Owner Password Credentials	Resources are owned by a particular user and the requesting application is trusted	A bit
Authorization Code	Resources are owned by a particular user and the requesting application is untrusted	Very
Implicit	Resources are owned by a particular user, and the requesting application is an untrusted browser-based app written in a scripting language such as JavaScript	Very, and potentially insecure as well

OAuth grants

Grant Type	Typical Use Case	Complex?
No specific resource owner is involved		
Client Credentials	Business system interactions, where resources being operated on are owned by the partner, not a particular user	No
A specific resource owner is involved		
Resource Owner Password Credentials	Resources are owned by a particular user and the requesting application is trusted	A bit
Authorization Code	Resources are owned by a particular user and the requesting application is untrusted	Very
Implicit	Resources are owned by a particular user, and the requesting application is an untrusted browser-based app written in a scripting language such as JavaScript	Very, and potentially insecure as well

apigee

Resource Owner Password Credentials - Actors

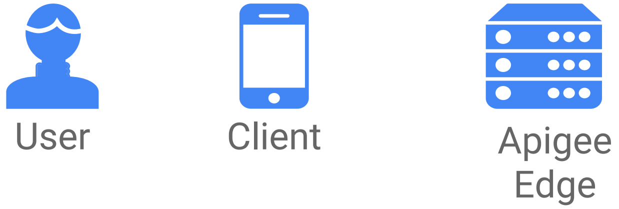
User

Resource Owner Password Credentials - Actors

User

Client

Resource Owner Password Credentials - Actors



Resource Owner Password Credentials - Actors

User

Client

Apigee Edge

Authentication Server

Resource Owner Password Credentials - Actors

User

Client

Apigee Edge

Authentication Server

Resource Server

Resource Owner Password Credentials Grant

- resource owner is involved and the application is trusted

Resource Owner Password Credentials Grant

- resource owner is involved and the application is trusted

- more complex and secure than client credentials grant type

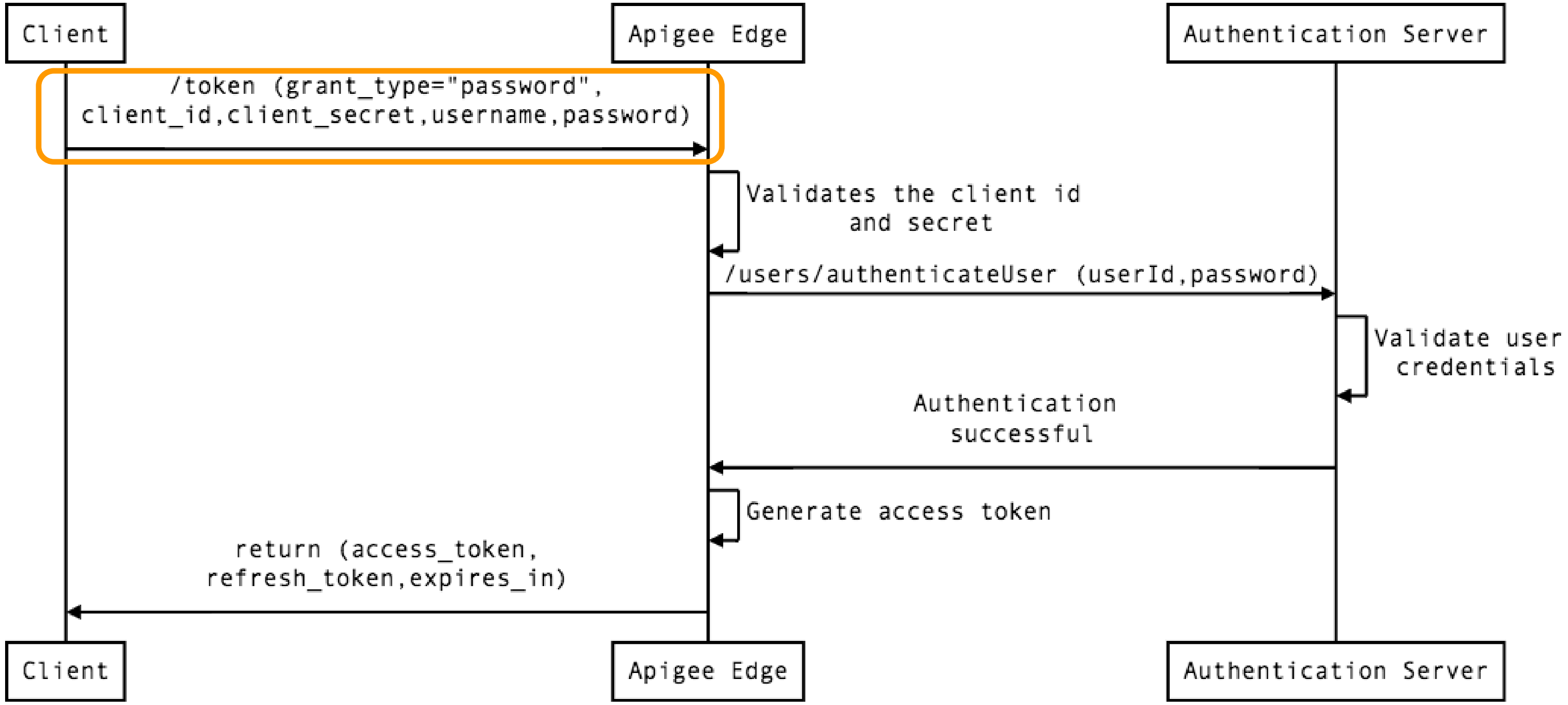
Resource Owner Password Credentials Grant

- resource owner is involved and the application is trusted
- more complex and secure than client credentials grant type
- migrate from basic auth to access tokens

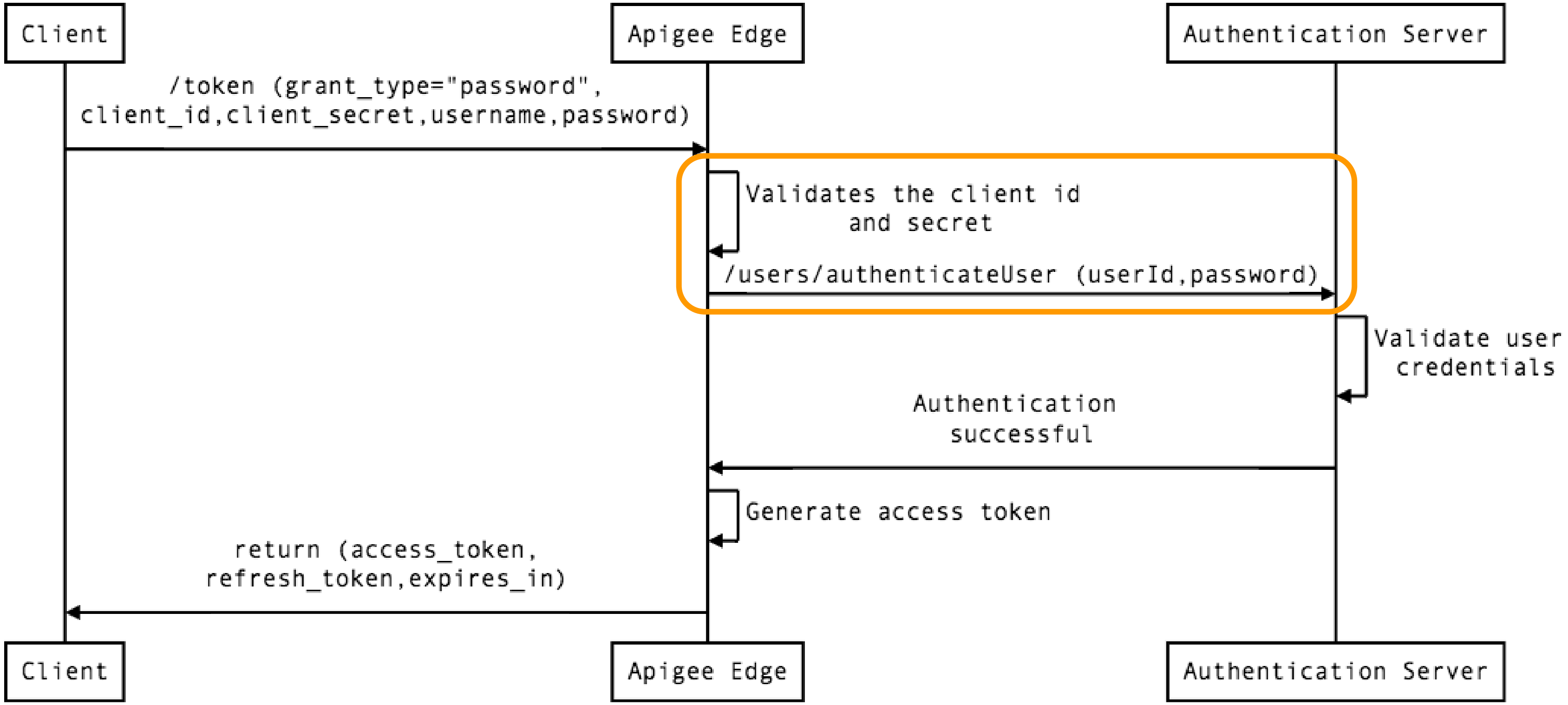
Resource Owner Password Credentials Grant

- resource owner is involved and the application is trusted
- more complex and secure than client credentials grant type
- migrate from basic auth to access tokens
- refresh token + access token

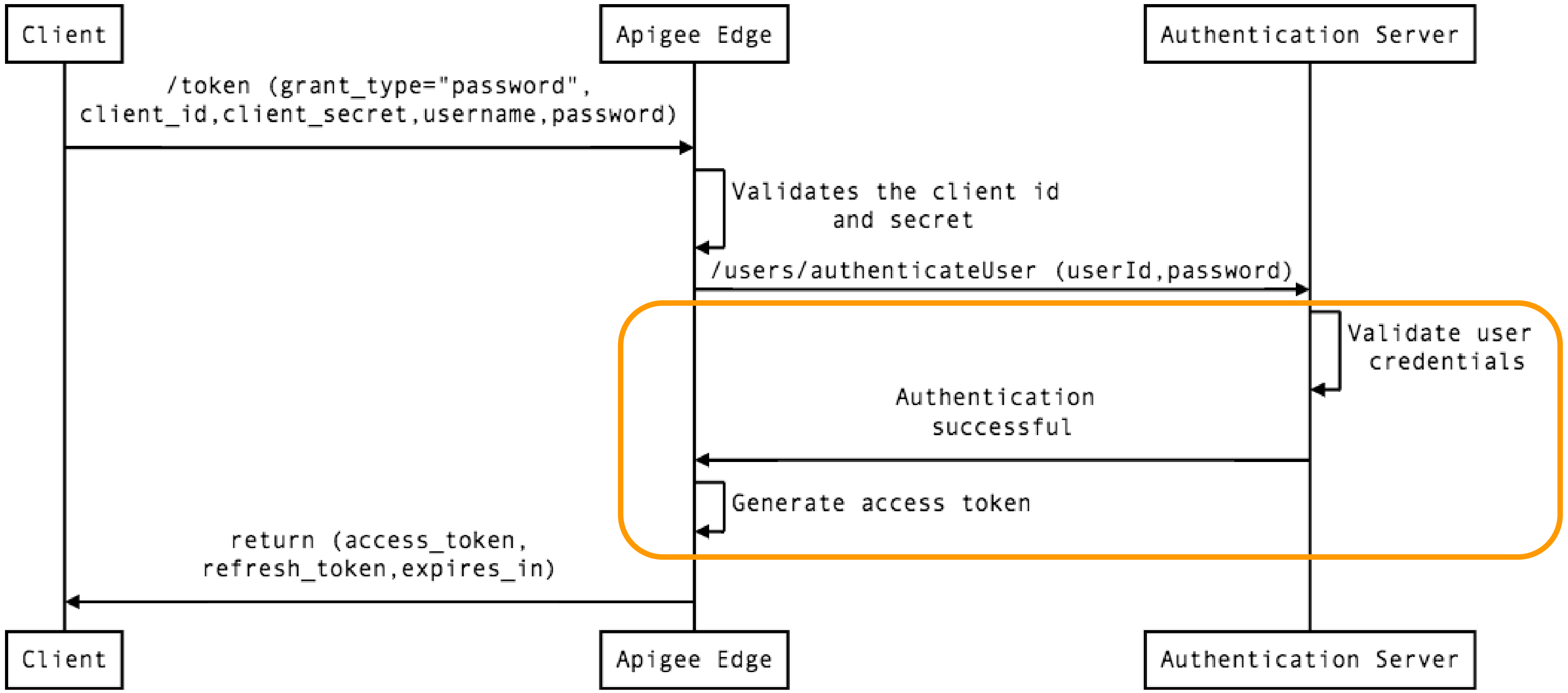
Sequence - Generate Token



Sequence - Generate Token



Sequence - Generate Token



Client
Apigee Edge
Authentication Server
/token (grant_type="password", client_id,client_secret,username,password)
Validates the client id and secret
/users/authenticateUser (userId,password)
Validate user credentials
Authentication successful
Generate access token
return (access_token, refresh_token,expires_in)
Client
Apigee Edge
Authentication Server

Generate Access Token policy

```
<OAuthV2 async="false" continueOnError="false"
enabled="true" name="oauth-generate-token">
  <DisplayName>OAuth Generate Token</DisplayName>
  <Operation>GenerateAccessToken</Operation>
  <ExpiresIn>86400000</ExpiresIn>
  <SupportedGrantTypes>
    <GrantType>password</GrantType>
  </SupportedGrantTypes>
  <GrantType>request.formparam.grant_type</GrantType>
  <UserName>request.formparam.username</UserName>
  <PassWord>request.formparam.password</PassWord>
  <GenerateResponse/>
</OAuthV2>
```

Client credentials vs. Password grant types

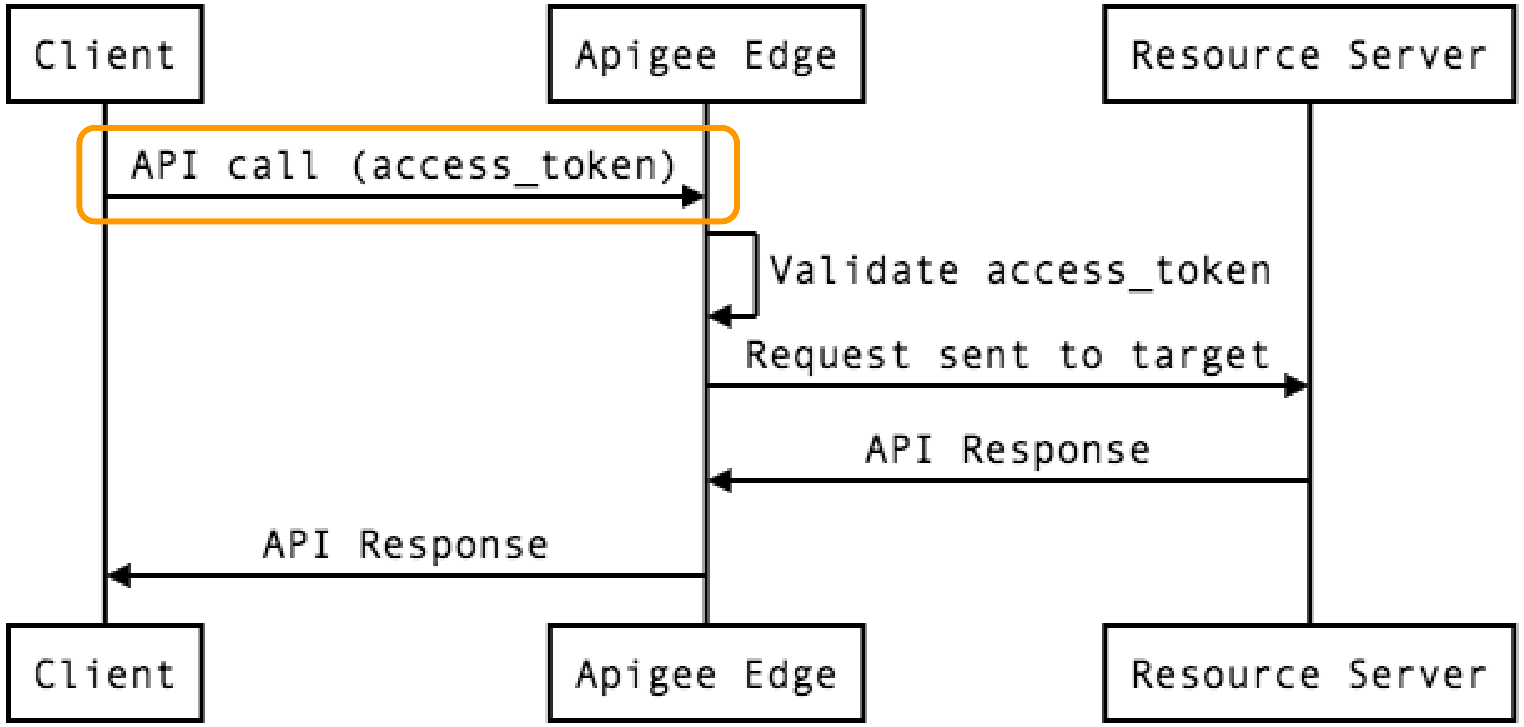
Client Credentials - Response:

```
{
  "issued_at" : "1407513671919",
  "application_name" : "26c855a9-c485-4318-accc",
  "scope" : "",
  "status" : "approved",
  "api_product_list" : "[Product1]",
  "expires_in" : "3599",
  "developer.email" : "xxx@yyy.com",
  "organization_id" : "0",
  "token_type" : "BearerToken",
  "client_id" : "vn0zG4cnSWaWIzdwBZgnREI1NGORDXXz",
  "access_token" : "2CsgxkPqfNtCSAZ5qGEI9x5dGdvV",
  "organization_name" : "demo",
  "refresh_token_expires_in" : "0",
  "refresh_count" : "0"
}
```

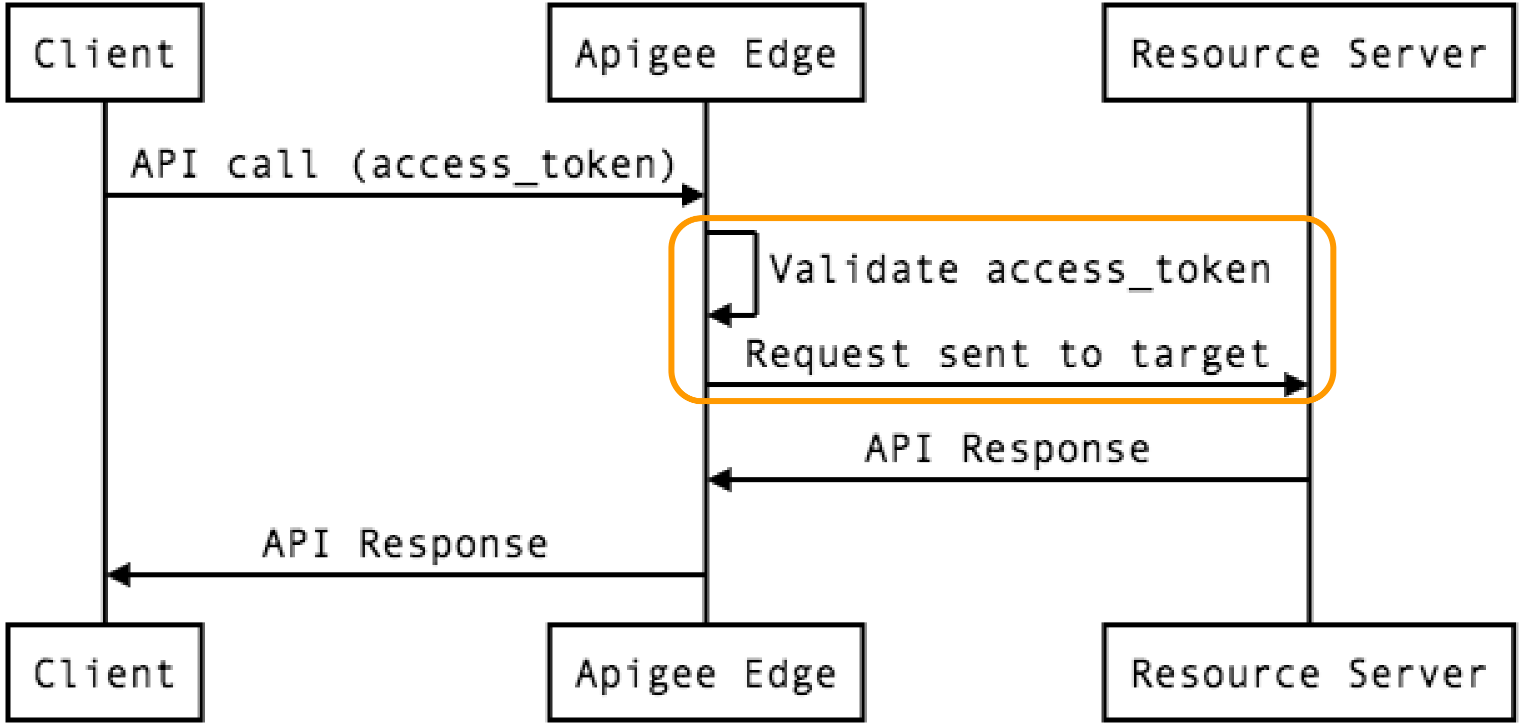
Password - Response:

```
{
  "issued_at" : "1407513709051",
  "scope" : "",
  "application_name" : "26c855a9-c485-4318-accc",
  "refresh_token_issued_at" : "1407513709051",
  "status" : "approved",
  "refresh_token_status" : "approved",
  "api_product_list" : "[Product1]",
  "expires_in" : "3599",
  "developer.email" : "xxx@yyy.com",
  "organization_id" : "0",
  "token_type" : "BearerToken",
  "refresh_token" : "HsnXmyIQqmJJQrFVdevmVztGGASUfBfz",
  "client_id" : "vn0zG4cnSWaWIzdwBZgnREI1NGORDXXz",
  "access_token" : "GRQAJcgSFZcklbIUxfoUaYFW2ROd",
  "organization_name" : "demo",
  "refresh_token_expires_in" : "0",
  "refresh_count" : "0"
}
```

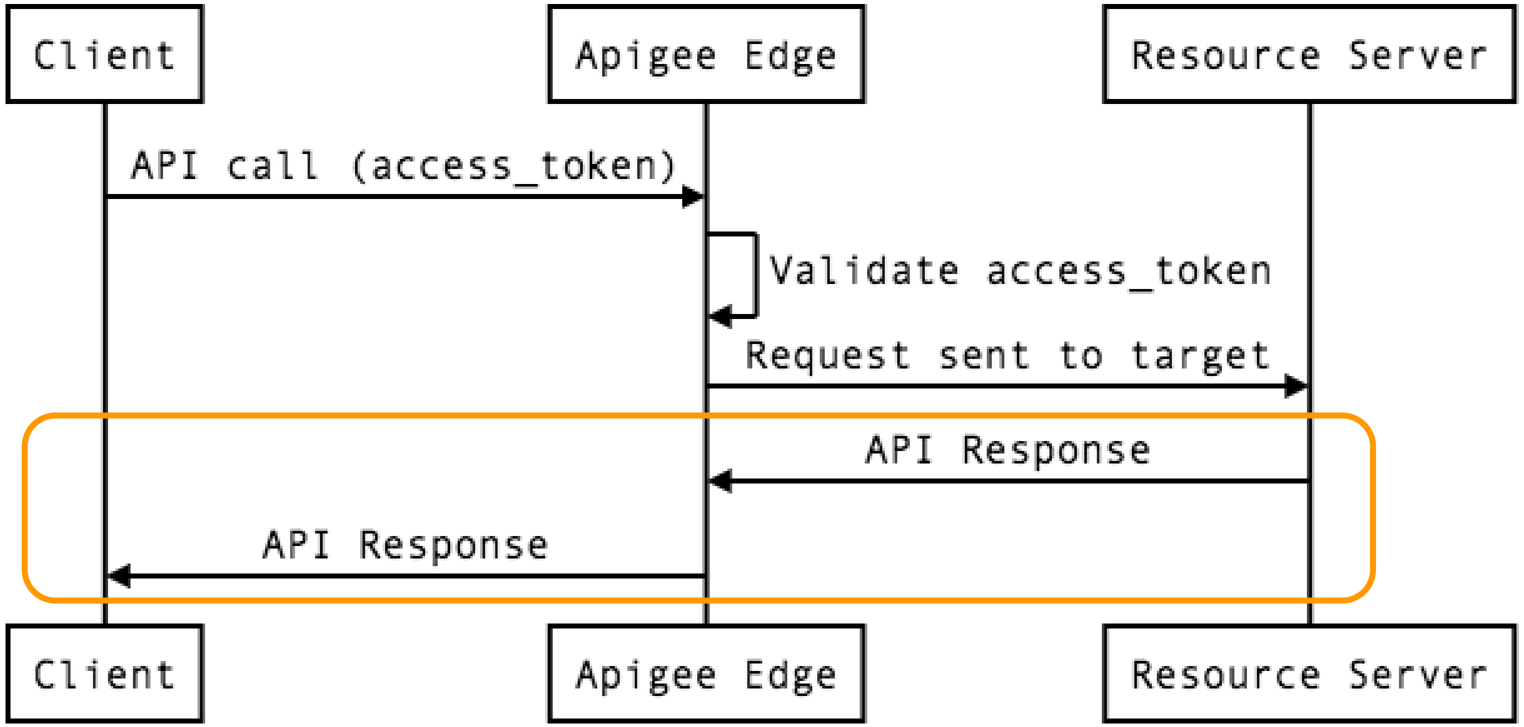
Sequence - Protected API



Sequence - Protected API



Sequence - Protected API



Verify OAuth token policy

- Set the access token as the bearer token in the authorization header of the http request.

```
curl -H "Authorization: Bearer {access_token}"
https://myorg-test.apigee.net/v1/customers/1234
```

Verify OAuth token policy

- Set the access token as the bearer token in the authorization header of the http request.

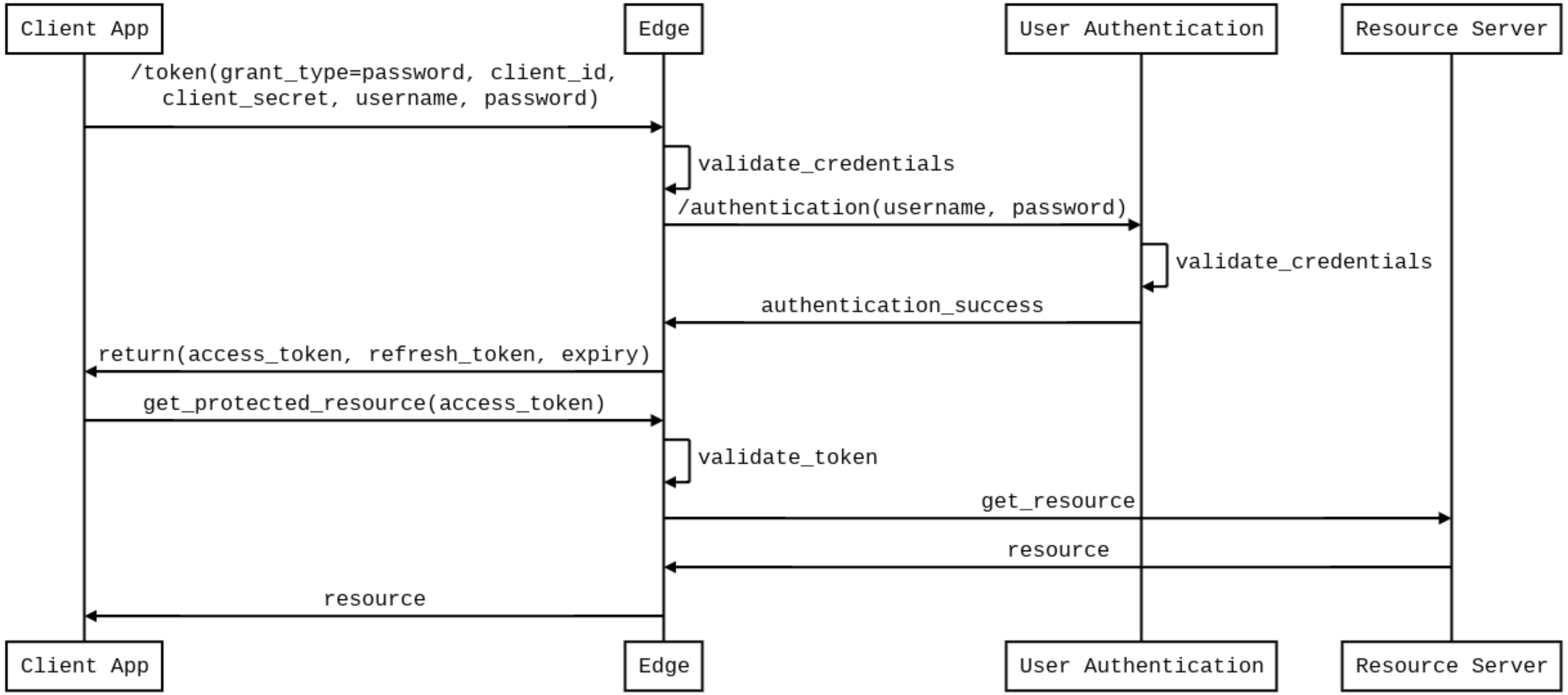
```
curl -H "Authorization: Bearer {access_token}"
https://myorg-test.apigee.net/v1/customers/1234
```

- *VerifyAccessToken* operation will validate the access token

```
<OAuthV2 async="false" continueOnError="false"
enabled="true" name="VerifyOAuthToken">
  <DisplayName>OAuth Verify Token</DisplayName>
  <Operation>VerifyAccessToken</Operation>
</OAuthV2>
```

Thank You

Sequence


Client App
Edge
User Authentication
Resource Server
/token(grant_type=password, client_id, client_secret, username, password)
validate_credentials
/authentication(username, password)
validate_credentials
authentication_success
return(access_token, refresh_token, expiry)
get_protected_resource(access_token)
validate_token
get_resource
resource
resource
Client App
Edge
User Authentication
Resource Server