

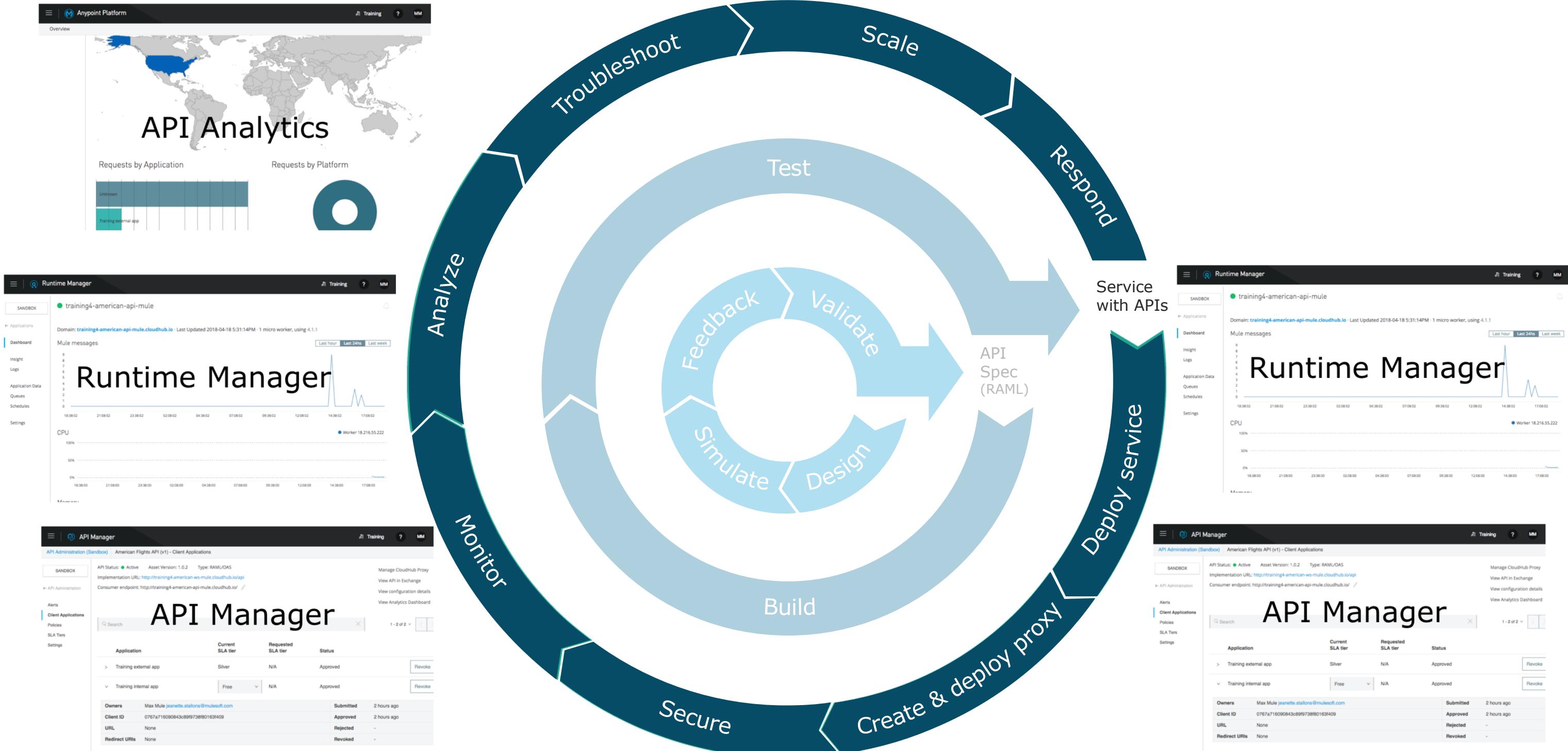


MuleSoft®

Module 5: Deploying and Managing APIs



Goal



At the end of this module, you should be able to



- Describe the options for deploying Mule applications
- Deploy Mule applications to CloudHub
- Use API Manager to create and deploy proxies for APIs
- Use API Manager to restrict access to API proxies

Introducing deployment options



Deploying applications



- During development, applications are deployed to an embedded Mule runtime in Anypoint Studio
- For everything else (testing, Q&A, and production), applications can be deployed to
 - **CloudHub**
 - Platform as a Service (PaaS) component of Anypoint Platform
 - MuleSoft-hosted Mule runtimes on AWS (Amazon Web Services platform)
 - A fully-managed, multi-tenanted, globally available, secure and highly available cloud platform for integrations and APIs
 - **Customer-hosted Mule runtimes**
 - On bare metal or cloud service providers: AWS, Azure, and Pivotal Cloud Foundry



MuleSoft-hosted runtime

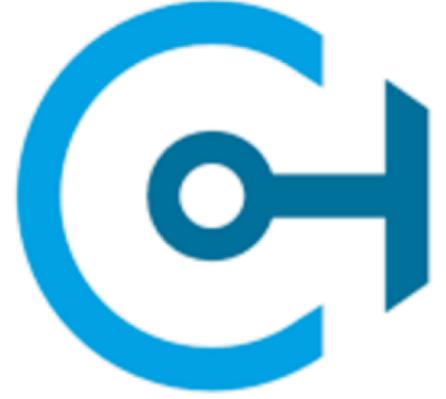


Customer-hosted runtime 5

CloudHub benefits



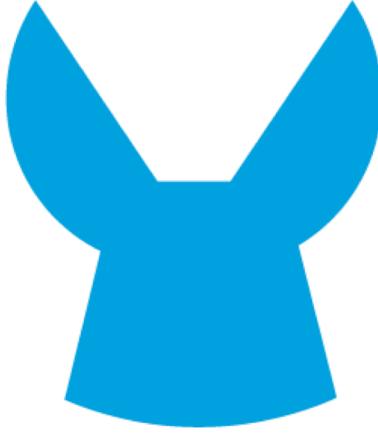
- No hardware to maintain
- Continuous software updates
- Provided infrastructure for DNS and load-balancing
- Built-in elastic scalability for increasing cloud capacity during periods of high demand
- Globally available with data centers around the world
- Highly available with 99.99% uptime SLAs (service level agreements)
<http://status.mulesoft.com/>
- Highly secure
 - PCI, HiTrust, and SSAE-16 certified



Customer-hosted Mule runtimes



- Easy to install
- Requires minimal resources
- Can run multiple applications
- Uses a Java Service Wrapper that controls the JVM from the operating system and starts Mule
- Can be managed by
 - Runtime Manager in MuleSoft-hosted Anypoint Platform
 - Runtime Manager in customer-hosted Anypoint Platform
 - Anypoint Platform Private Cloud Edition



Deploying applications to CloudHub



Deploying applications to CloudHub



- Can deploy from Anypoint Studio or from Anypoint Platform using Runtime Manager
- You must set worker size and number
 - For apps deployed from flow designer, these values were set automatically

The screenshot shows the MuleSoft Runtime Manager interface. At the top, there's a navigation bar with 'Runtime Manager' and other tabs like 'Training' and 'MM'. On the left, a sidebar has 'Sandbox' selected, along with links for 'Applications', 'Dashboard', 'Insight', 'Logs', 'Application Data', 'Queues', 'Schedules', and 'Settings'. The main area displays an application named 'training4-american-ws-mule'. It shows the 'Application File' as 'training4-american-ws-v2.jar', with options to 'Choose file' or 'Get from sandbox' and a 'Stop' button. Below this, it shows 'Last Updated 2018-04-18 1:56:26PM' and 'App url: training4-american-ws-mule.cloudhub.io'. A table at the bottom allows configuration of 'Runtime', 'Properties', 'Insight', 'Logging', and 'Static IPs'. The 'Runtime' tab is active, showing 'Runtime version' as '4.1.1', 'Worker size' as '0.1 vCores', and 'Workers' as '1'. There are dropdown menus for both 'Worker size' and 'Workers'. Under 'Worker size', options include '0.1 vCores 500 MB memory', '0.2 vCores 1 GB memory', and '1 vCore 1.5 GB memory'. Under 'Workers', there's a dropdown menu with '1' selected. At the bottom right, there's a 'Settings' section with checkboxes for 'Automatically restart application when not res' (checked), 'Persistent queues' (unchecked), and 'Encrypt persistent' (unchecked). A large 'Apply Changes' button is at the bottom right.

Review: CloudHub workers



- A worker is a dedicated instance of Mule that runs an app
- Each worker
 - Runs in a separate container from every other application
 - Is deployed and monitored independently
 - Runs in a specific worker cloud in a region of the world
- Workers can have a different memory capacity and processing power
 - Applications can be scaled vertically by changing the worker size
 - Applications can be scaled horizontally by adding multiple workers

| Worker size |
|---------------|
| 0.1 vCores |
| 0.1 vCores |
| 500 MB memory |
| 0.2 vCores |
| 1 GB memory |
| 1 vCore |
| 1.5 GB memory |
| 2 vCores |
| 3.5 GB memory |
| 4 vCores |

Walkthrough 5-1: Deploy an application to CloudHub



- Deploy an application from Anypoint Studio to CloudHub
- Run the application on its new, hosted domain
- Make calls to the web service
- Update an API implementation deployed to CloudHub

A screenshot of the MuleSoft Runtime Manager interface. The top navigation bar includes 'Runtime Manager' with a search icon, 'Training', a help icon, and a user icon. On the left, a sidebar shows 'Sandbox' selected, along with links for 'Applications', 'Servers', and 'Alerts'. The main content area has tabs for 'Deploy application' (which is active and highlighted in blue) and 'Search Applications'. A table lists deployed applications with columns for Name, Server, Status, and File. One application is listed: 'training4-american-ws-mule' is running on 'CloudHub' with a green 'Started' status and the file 'training4-american-ws-v2.jar'.

| Name | Server | Status | File |
|----------------------------|----------|---------|------------------------------|
| training4-american-ws-mule | CloudHub | Started | training4-american-ws-v2.jar |

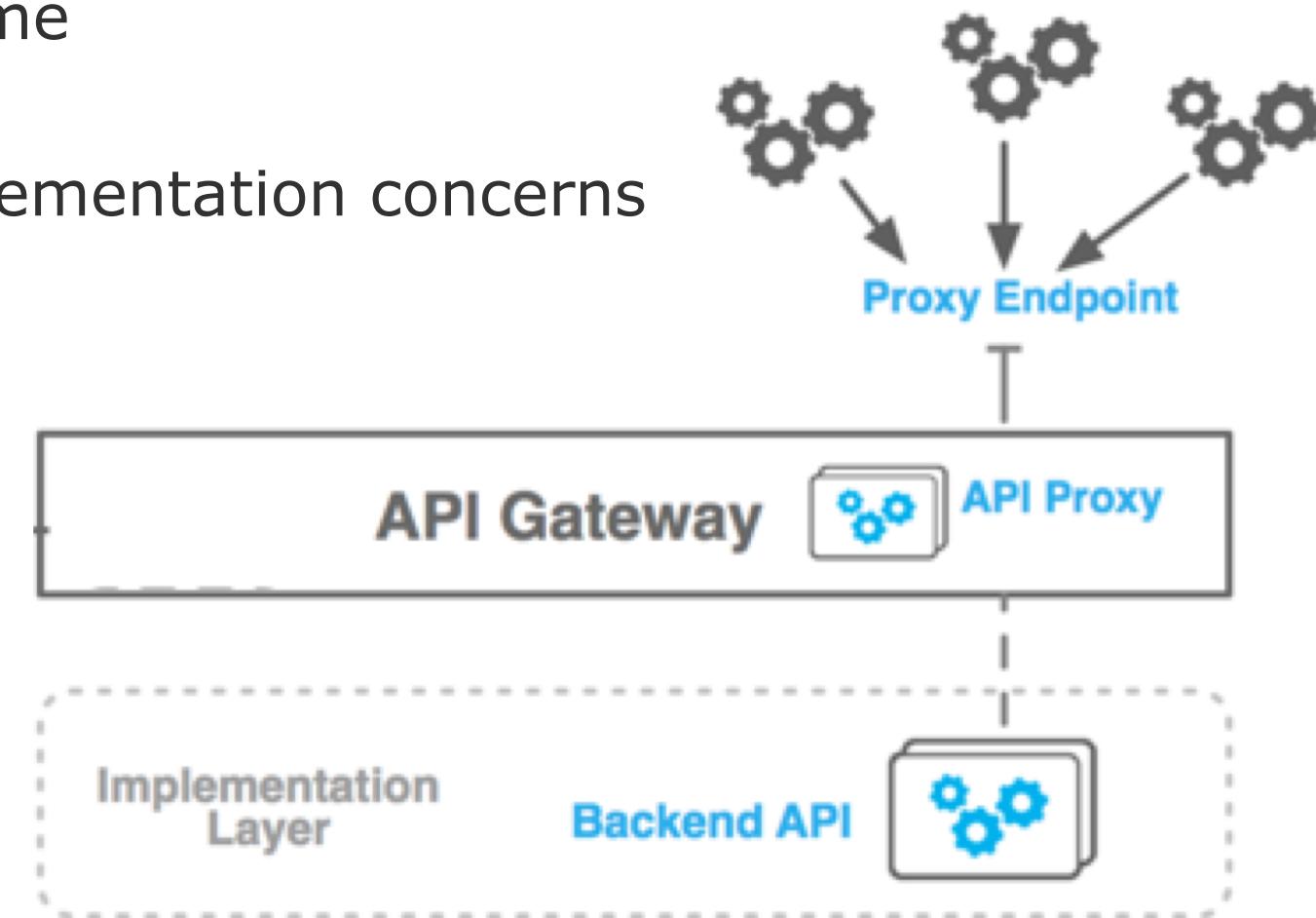
Creating API proxies



Restricting access to APIs



- An **API proxy** is an application that controls access to a web service, restricting access and usage through the use of an API gateway
- The **API Gateway** is a runtime designed and optimized to host an API or to open a connection to an API deployed to another runtime
 - Included as part of the Mule runtime
 - Separate licenses required
 - Separates orchestration from implementation concerns



The API Gateway is the point of control



- **Determines which traffic** is authorized to pass through the API to backend services
- **Meters the traffic** flowing through
- **Logs** all transactions, collecting and tracking analytics data
- Applies runtime policies to **enforce governance** like rate limiting, throttling, and caching

Using API Manager to manage access to APIs



- **Create** proxy applications
- **Deploy** proxies to an API Gateway runtime
 - On CloudHub or a customer-hosted runtime
- Specify throttling, security, and other **policies**
- Specify **tiers** with different types of access
- Approve, reject, or revoke **access** to APIs by clients
- **Promote** managed APIs between environments
- Review **analytics**

A screenshot of the MuleSoft API Manager web interface. The top navigation bar includes the MuleSoft logo, a menu icon, and the text "API Manager". Below the header, a breadcrumb navigation shows "API Administration (Sandbox) > Am". The main content area has a sidebar with links: "Sandbox" (which is highlighted), "← API Administration", "Alerts", "Client Applications", "Policies" (which is selected and highlighted in blue), "SLA Tiers", and "Settings". A large blue button labeled "Apply" is at the bottom right of the sidebar. To the right of the sidebar, there is a vertical column with partially visible text: "Amel", "API Statu", "Impleme", and "Consum".

SANDBOX

← API Administration

Alerts

Client Applications

Policies

SLA Tiers

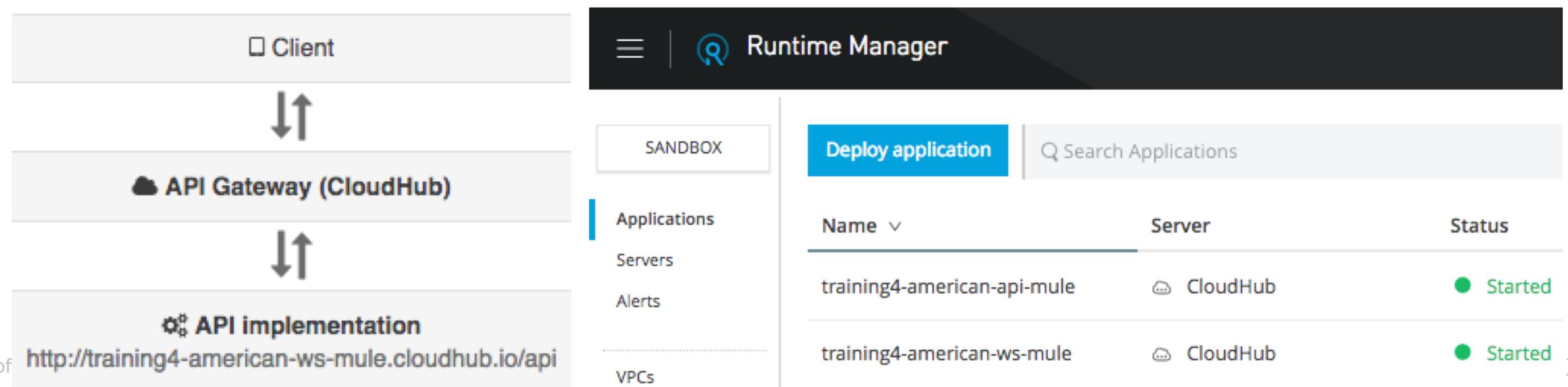
Settings

Apply

Walkthrough 5-2: Create and deploy an API proxy



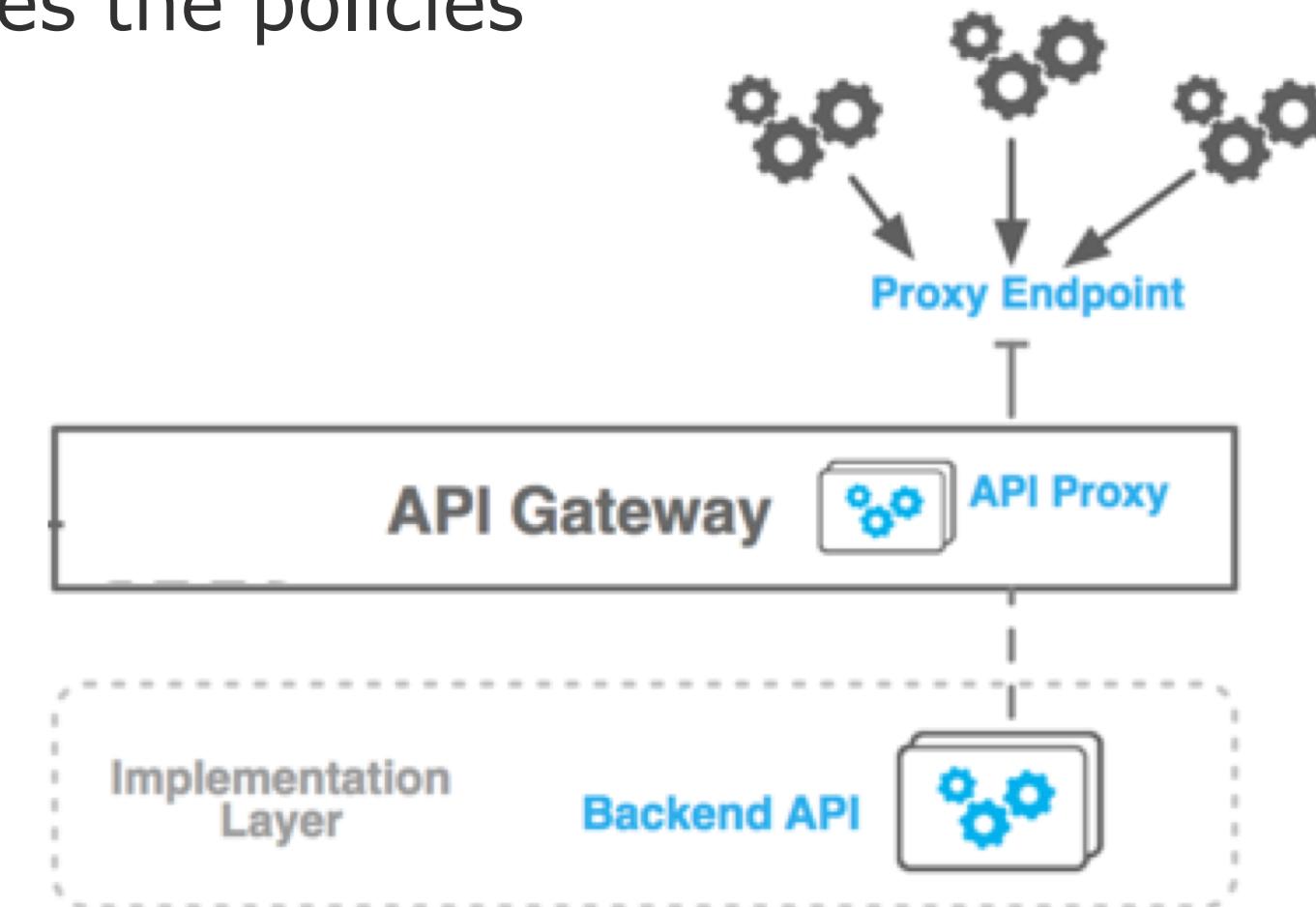
- Add an API to API Manager
- Use API Manager to create and deploy an API proxy application
- Set a proxy consumer endpoint so requests can be made to it from Exchange
- Make calls to an API proxy from API portals for internal & external users
- View API request data in API Manager.



Restricting access to APIs



- Use **API Manager** to manage access to API proxies
 - Define SLA tiers
 - Apply runtime policies
- The **API Gateway** enforces the policies



Applying policies to restrict access



- There are **out-of-the box** policies for many common use cases
 - Rate limiting
 - Spike control
 - Security
- You can define **custom** policies (using XML and YAML)
- You can apply **multiple** policies and set the order

| | |
|------------------------------------|-----------------------------|
| Client ID enforcement | JSON threat protection |
| Cross-Origin resource sharing | Basic Authentication - LDAP |
| OAuth 2.0 access token enforcement | Message Logging |
| Header Injection | Rate limiting |
| Header Removal | Rate limiting - SLA based |
| Basic authentication - Simple | Spike Control |
| IP blacklist | XML threat protection |
| IP whitelist | |

Using SLA tiers to restrict access by client ID



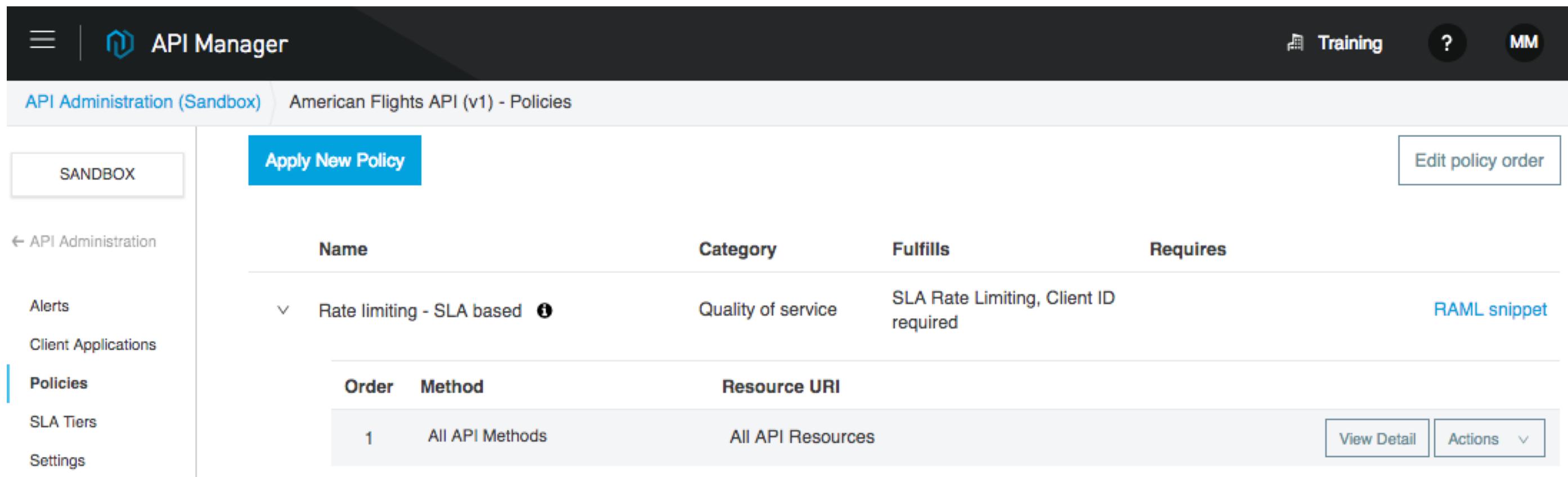
- A **Service Level Agreement** tier defines the # of requests that can be made per time frame to an API
 - Request approval can be set to automatic (free) or manual (for tiers that cost \$)

The screenshot shows the API Manager interface for the American Flights API (v1) - SLA Tiers. The left sidebar has a 'Sandbox' button and links to Alerts, Client Applications, Policies, SLA Tiers (which is selected and highlighted in blue), and Settings. The main content area has tabs for API Administration (Sandbox) and American Flights API (v1) - SLA Tiers, with a 'View Analytics Dashboard' link. A search bar and a navigation bar with '1 - 2 of 2' and arrows are also present. The table lists two SLA tiers: 'Free' and 'Silver'. The 'Free' tier has 1 limit, 1 application, is active, and has auto approval. The 'Silver' tier also has 1 limit, 1 application, is active, and has manual approval. Each tier has 'Edit' and 'Deprecate' buttons.

| Name | Limits | Applications | Status | Approval | |
|--------|--------|--------------|--------|----------|--|
| Free | 1 | 1 | Active | Auto | Edit Deprecate |
| Silver | 1 | 1 | Active | Manual | Edit Deprecate |

Walkthrough 5-3: Restrict API access with policies and SLAs

- Add and test a rate limiting policy
- Add SLA tiers, one with manual approval required
- Add and test a rate limiting SLA based policy

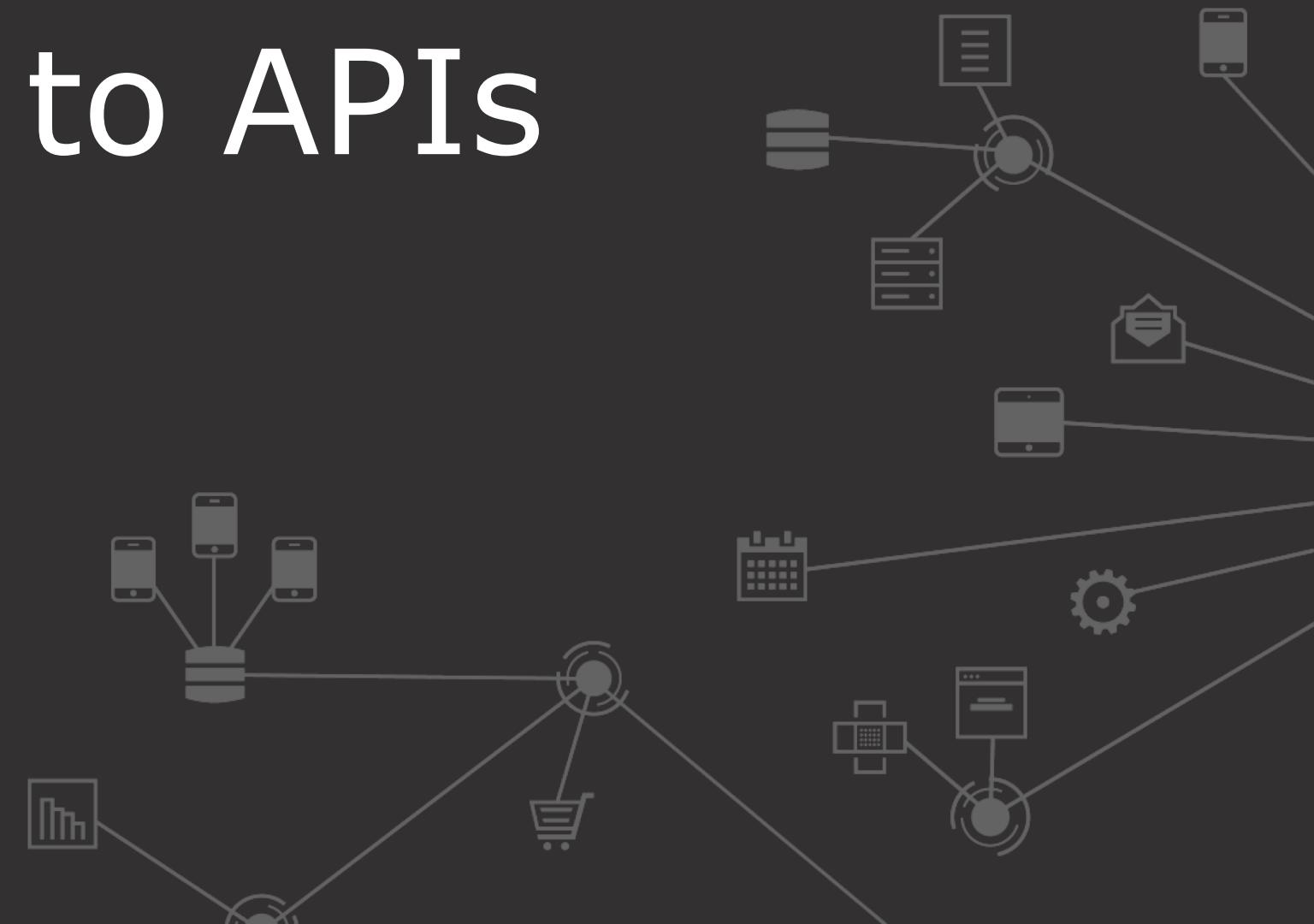


The screenshot shows the MuleSoft API Manager interface. The top navigation bar includes 'API Manager', 'Training', a help icon, and a user icon. The left sidebar has links for 'Sandbox', 'API Administration', 'Alerts', 'Client Applications', 'Policies' (which is selected), 'SLA Tiers', and 'Settings'. The main content area displays the 'American Flights API (v1) - Policies' page. A blue button labeled 'Apply New Policy' is visible. The table lists policies with columns for Name, Category, Fulfils, Requires, Order, Method, and Resource URI. One policy, 'Rate limiting - SLA based', is expanded to show its details: Category is 'Quality of service', Fulfillments are 'SLA Rate Limiting, Client ID required', and it uses a 'RAML snippet'. The 'Edit policy order' button is highlighted with a red box.

| | Name | Category | Fulfils | Requires |
|--------|-----------------------------|--------------------|---------------------------------------|--------------|
| Alerts | Rate limiting - SLA based ⓘ | Quality of service | SLA Rate Limiting, Client ID required | RAML snippet |

| Order | Method | Resource URI | View Detail | Actions |
|-------|-----------------|-------------------|-------------|---------|
| 1 | All API Methods | All API Resources | | |

Granting access to APIs



Enforcing access to APIs using SLA tiers



- To enforce, apply an **SLA based** rate limiting policy
- SLA based policies require all applications that consume the API to
 - **Register** for access to a specific tier
 - From an API portal in private or public Exchange
 - **Pass their client credentials** in calls made to the API

GET

Sandbox - Rate limiting - SLA based policy

http://training4-american-api-mule.cloudhub.io/flights

Parameters Headers

| | |
|---|------------------|
| <input checked="" type="checkbox"/> </> | |
| Header name | Value |
| client_id | 7623dbcc2e1949d7 |
| Header name | Value |
| client_secret | 52505680a6FB4d5; |

+ Add custom header

Send

200 OK 1042.50 ms Details ▾

⌂ </> [Array[11]]

```
-0: {  
  "ID": 1,  
  "code": "rree0001",  
  "price": 541}
```

Requesting access to SLA tiers



- If an API has an SLA-based policy, a Request API access button appears in API portal
- To request access, developer must belong to the Anypoint Platform organization and be logged in
- When developers request access, they must
 - Register/add an app to their Anypoint Platform account
 - Select a tier

The screenshot shows the MuleSoft API portal interface. At the top, there's a navigation bar with the MuleSoft logo, 'MuleSoft // Training', 'Home', and 'My applications'. Below the navigation, the 'Assets list' shows an item named 'American Flights API | v1'. This item has a green house icon, a 5-star rating with '(0 reviews)', and links for 'Rate and review', 'Download', and 'Request access'. A modal dialog box titled 'Request API access' is open in the foreground. It contains fields for 'Application' (set to 'Training external app'), 'API Instance' (set to 'Sandbox - Rate limiting - SLA based ...'), 'SLA tier' (set to 'Silver'), and a table for specifying the request parameters: '# of Reqs' (1), 'Time period' (1), and 'Time Unit' (Second). At the bottom of the dialog are 'Cancel' and 'Request API access' buttons.

| # of Reqs | Time period | Time Unit |
|-----------|-------------|-----------|
| 1 | 1 | Second |

Approving SLA tier access requests



- For tiers with manual approval, emails are sent to the Organization Administrators when developers request access for applications
- Organization Administrators can review the applications in API Manager and approve, reject, or revoke requests

The screenshot shows the API Manager interface for managing client applications. The left sidebar has links for API Administration, Alerts, Client Applications (which is selected), Policies, SLA Tiers, and Settings. The main content area shows a table of client applications with the following data:

| Application | Current SLA tier | Requested SLA tier | Status | Action |
|-----------------------|------------------|--------------------|----------|---------------------------|
| Training external app | N/A | Silver | Pending | Approve Reject Delete |
| Training internal app | Free | N/A | Approved | Revoke |

Walkthrough 5-4: Request and grant access to a managed API



- Request application access to SLA tiers from private and public API portals
- Approve application requests to SLA tiers in API Manager

The screenshot shows the API Manager interface for the American Flights API (v1) - Client Applications. The left sidebar has tabs for Alerts, Client Applications (which is selected), Policies, SLA Tiers, and Settings. The main content area shows a table of client applications:

| Application | Current SLA tier | Requested SLA tier | Status | |
|-----------------------|------------------|--------------------|----------|------------------------|
| Training external app | Silver | N/A | Approved | Revoke |
| Training internal app | Free | N/A | Approved | Revoke |

Below the table, there is a detailed view for the first application:

| | | | |
|---------------|----------------------------------|-----------|-------------------|
| Owners | Max Mule | Submitted | 2 minutes ago |
| Client ID | 7623dbcc2e1949d7a861160fe4a3a1e6 | Approved | a few seconds ago |
| URL | None | Rejected | - |
| Redirect URIs | None | Revoked | - |

Adding client ID enforcement to API specifications



Adding client ID enforcement to API specifications



- You need to add client ID enforcement to the API spec for the
 - REST connector that is created for the API to enforce the authentication
 - Required headers to automatically show up in the API console so you don't have to manually add them for every call
- Instructions are in the RAML snippet for a policy in API Manager

| Name | Category | Fulfils | Requires |
|--|--------------------|---------------------------------------|------------------------------|
| Rate limiting - SLA based i | Quality of service | SLA Rate Limiting, Client ID required | RAML snippet |

RAML snippet for Rate limiting - SLA based X

[RAML 0.8](#) [RAML 1.0](#)

Client ID based policies by default expect to obtain the client ID and secret as query parameters. To enforce this in the API definition a trait can be defined in RAML as shown below.

```
traits:  
  client-id-required:  
    headers:  
      client_id:  
        type: string  
      client_secret:  
        type: string
```

This trait must then be applied to the resource or methods using the `is` RAML attribute.

```
/products:  
  get:  
    is: [client-id-required]  
    description: Gets a list of all the inventory products.
```

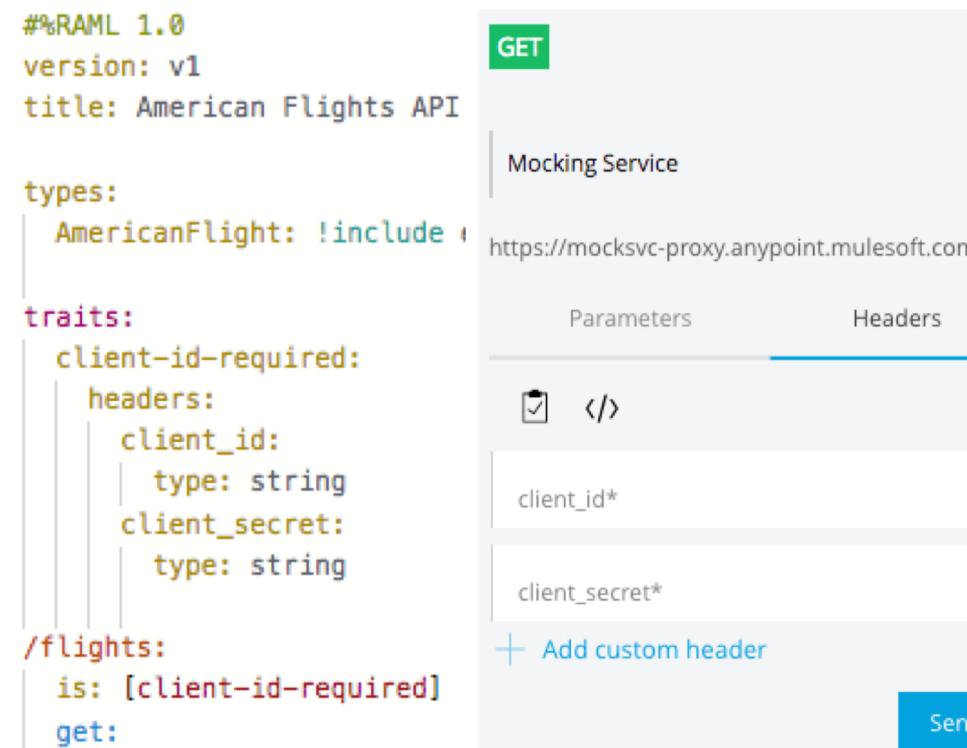
Please read [Applying Resource Types and Traits](#) section on RAML documentation for more information.

[Close](#)

Walkthrough 5-5: (Optional) Add client ID enforcement to an API specification

- Modify an API specification to require client id and client secret headers with requests
- Update a managed API to use a new version of an API specification
- Call a governed API with client credentials from API portals

Note: If you do not complete this exercise for Fundamentals, the REST connector that is created for the API and that you use later in the course will not have client_id authentication



The image shows a split-screen view. On the left is a code editor displaying a RAML 1.0 API specification. The specification includes a title, types, traits (with a 'client-id-required' trait), and a /flights endpoint marked as 'client-id-required'. On the right is a screenshot of the Anypoint Platform Mocking Service interface, showing a GET request configuration. The URL is set to https://mocksvc-proxy.anypoint.mulesoft.com/exc. Under the 'Headers' tab, there are two fields: 'client_id*' and 'client_secret*', both marked with red asterisks indicating they are required. A blue 'Send' button is at the bottom right.

```
%%RAML 1.0
version: v1
title: American Flights API

types:
  AmericanFlight: !include !
```

```
traits:
  client-id-required:
    headers:
      client_id:
        type: string
      client_secret:
        type: string

/flights:
  is: [client-id-required]
  get:
```

Summary



- Deploy applications to MuleSoft-hosted or customer-hosted Mule runtimes
- **CloudHub** is the Platform as a Service (PaaS) component of Anypoint Platform
 - Hosted Mule runtimes (workers) on AWS
- An **API proxy** is an application that controls access to a web service, restricting access and usage through the use of an API gateway
- The **API Gateway runtime** controls access to APIs by enforcing policies
 - Is part of the Mule runtime but requires a separate license

- Use **API Manager** to
 - Create and deploy API proxies
 - Define SLA tiers and apply runtime policies
 - Anypoint Platform has out-of-the box policies for rate-limiting, throttling, security enforcement, and more
 - SLA tiers defines # of requests that can be made per time to an API
 - Approve, reject, or revoke access to APIs by clients
 - Promote managed APIs between environments
 - Review API analytics

- This module was just an introduction to deploying and managing applications and APIs
- Anypoint Platform Operations:
 - CloudHub
 - Customer-Hosted Runtimes
 - API Management

