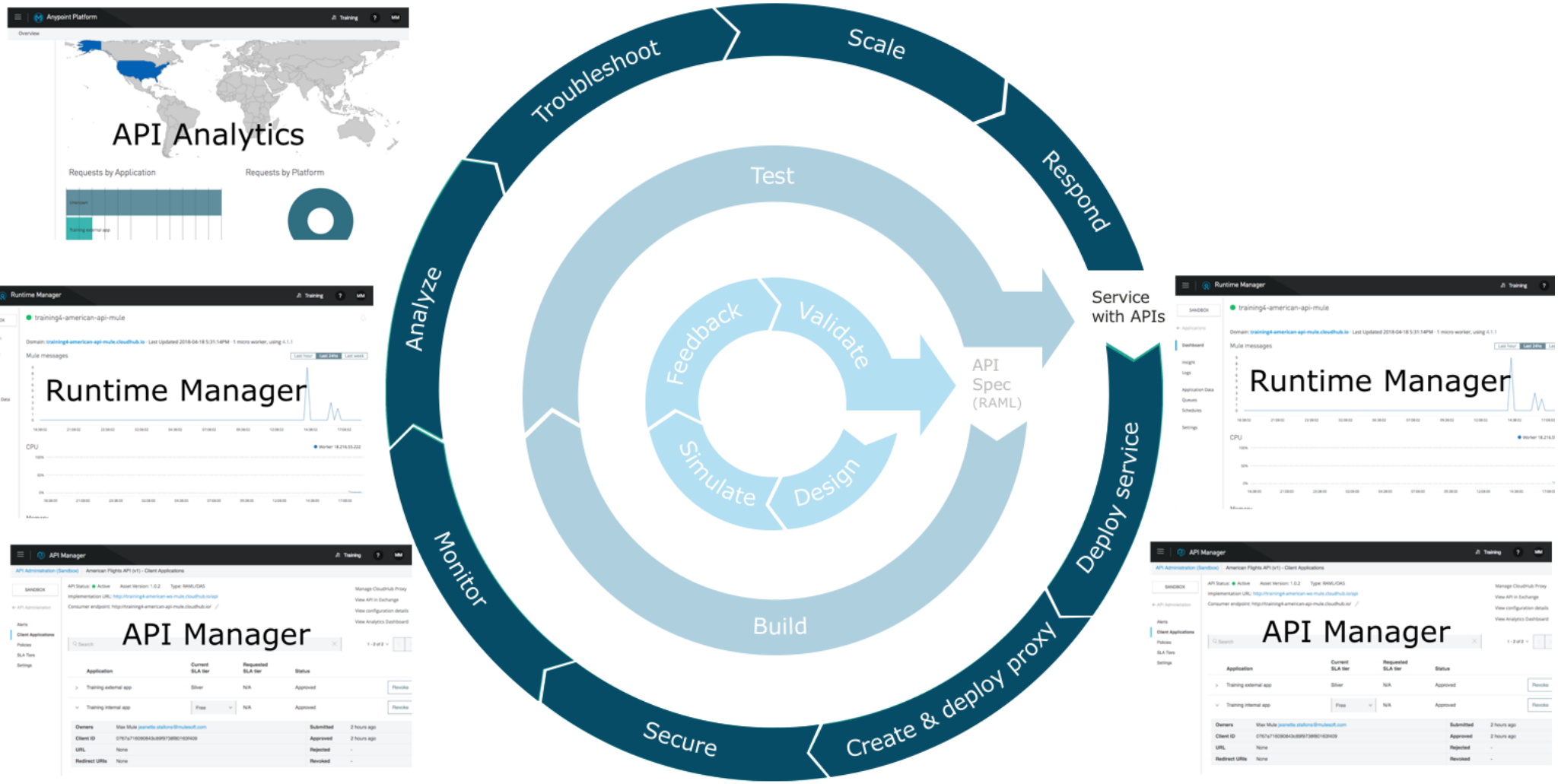
hello

Goal


API Analytics
Requests by Application
Requests by Platform
Runtime Manager
API Manager
Troubleshoot
Scale
Respond
Analyze
Monitor
Secure
Create & deploy proxy
Deploy service
Test
Feedback
Validate
Simulate
Design
Build
API
Spec
(RAML)
Service
with APIs

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 API proxies
- 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

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

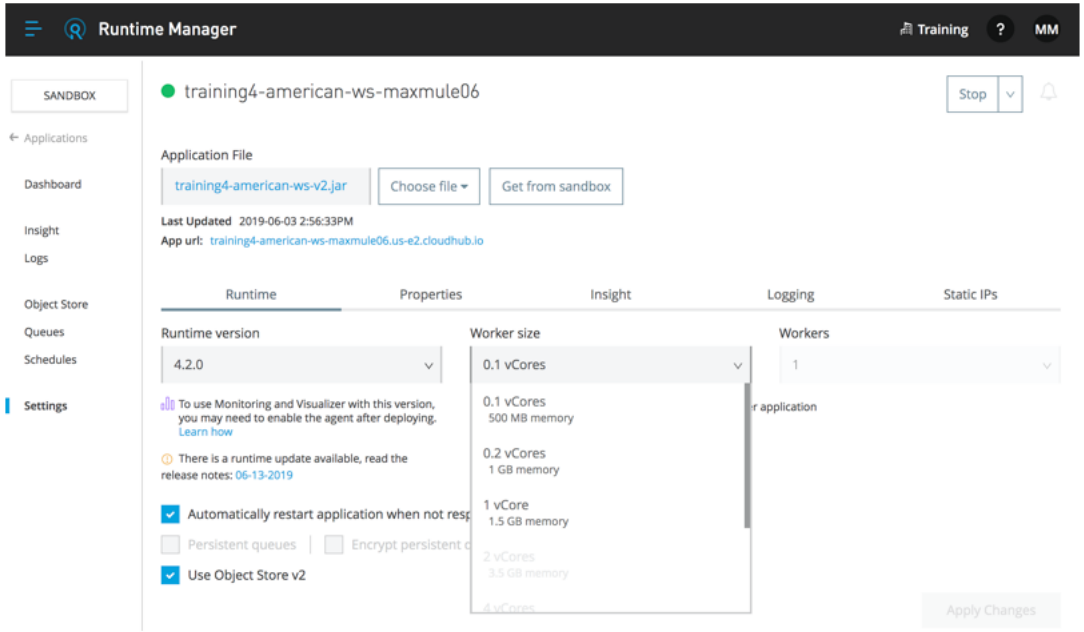
Customer-hosted Mule runtimes

- 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

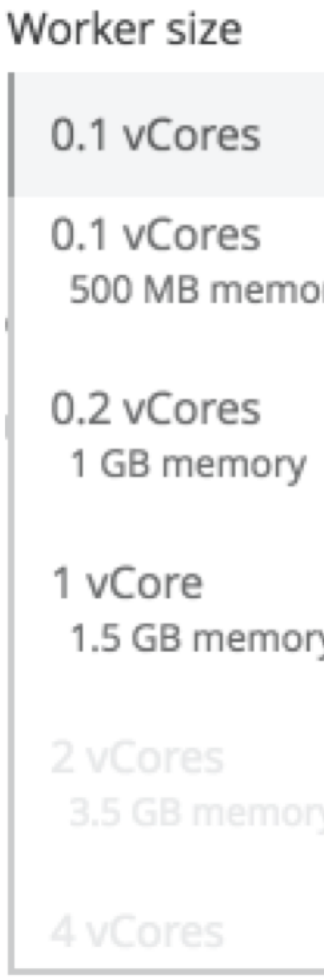
Deploying applications to CloudHub

- 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



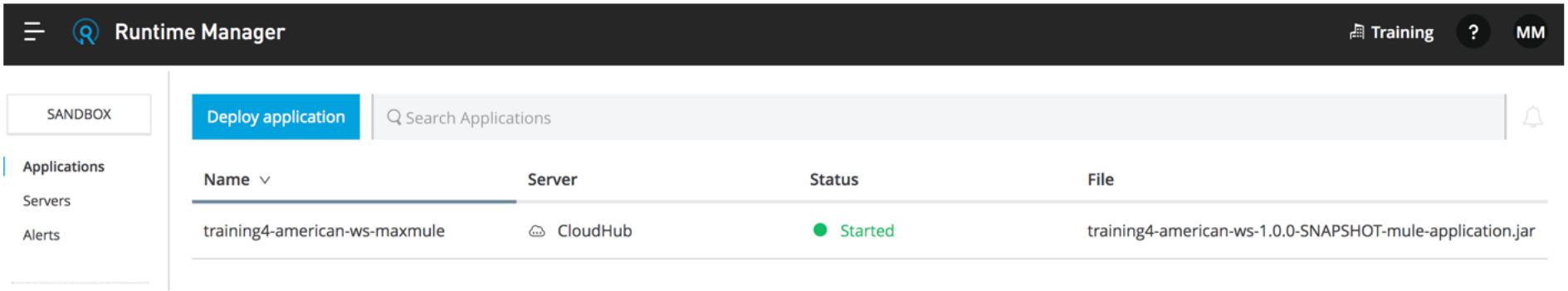
Review: CloudHub workers

- 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



Walkthrough 5-1: Deploy an application to CloudHub

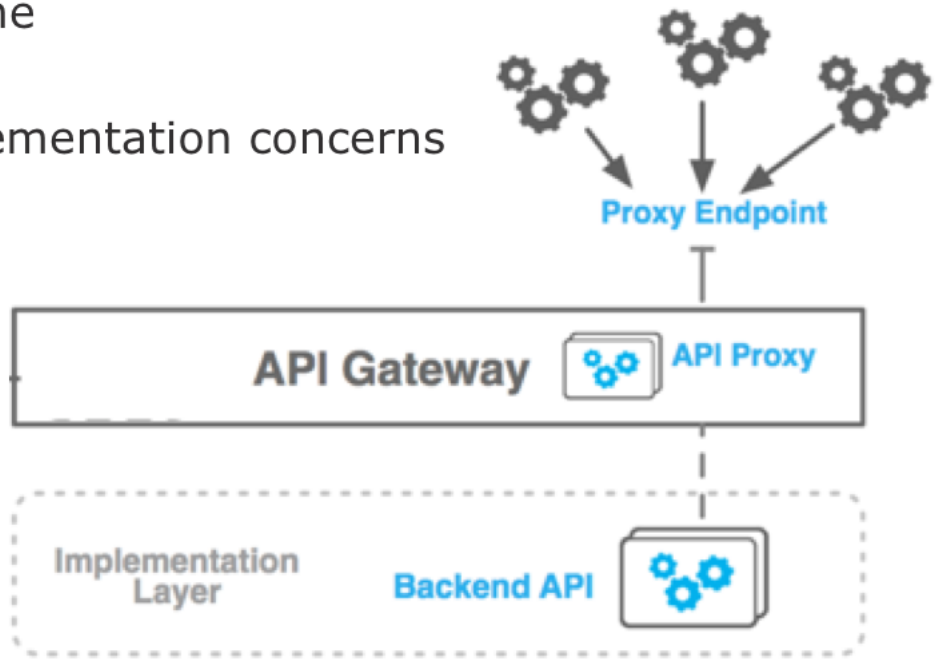
- Deploy an application from Anypoint Studio to CloudHub
- Run the application on its new, hosted domain
- Make calls to the web service
- Update an API implementation deployed to CloudHub



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



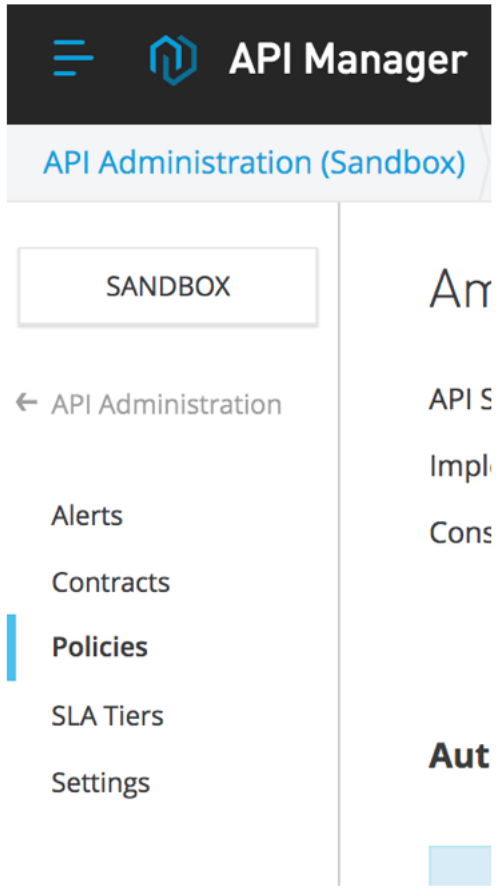
The API Gateway is the point of control

- **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

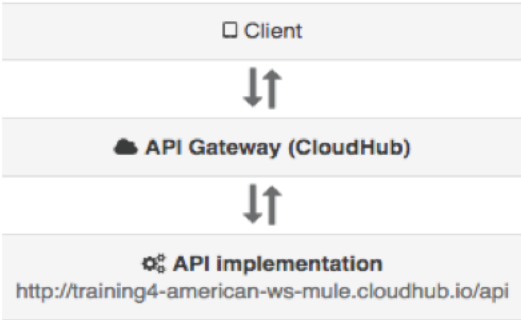
MuleSoft

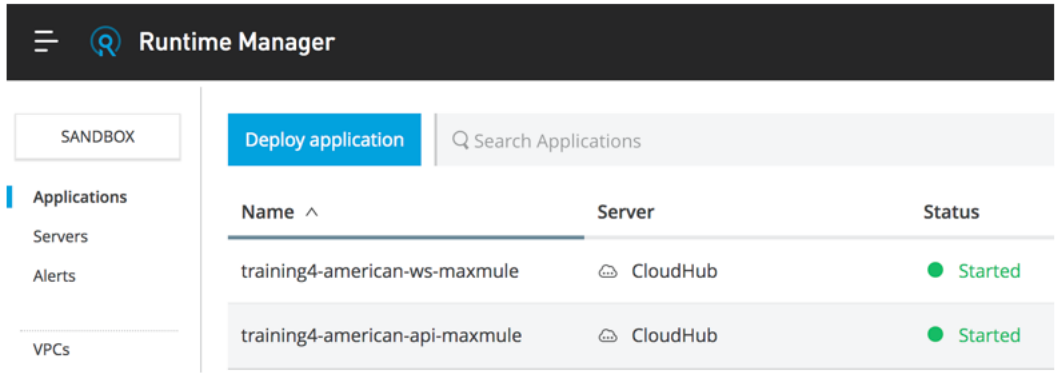
- **Create** proxy applications
- **Deploy** proxies to an API Gateway runtime
  - On CloudHub or a customer-hosted runtime
- Specify throttling, security, and other **policies**
- Specify **tiers** with different types of access
- Approve, reject, or revoke **access** to APIs by clients
- **Promote**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



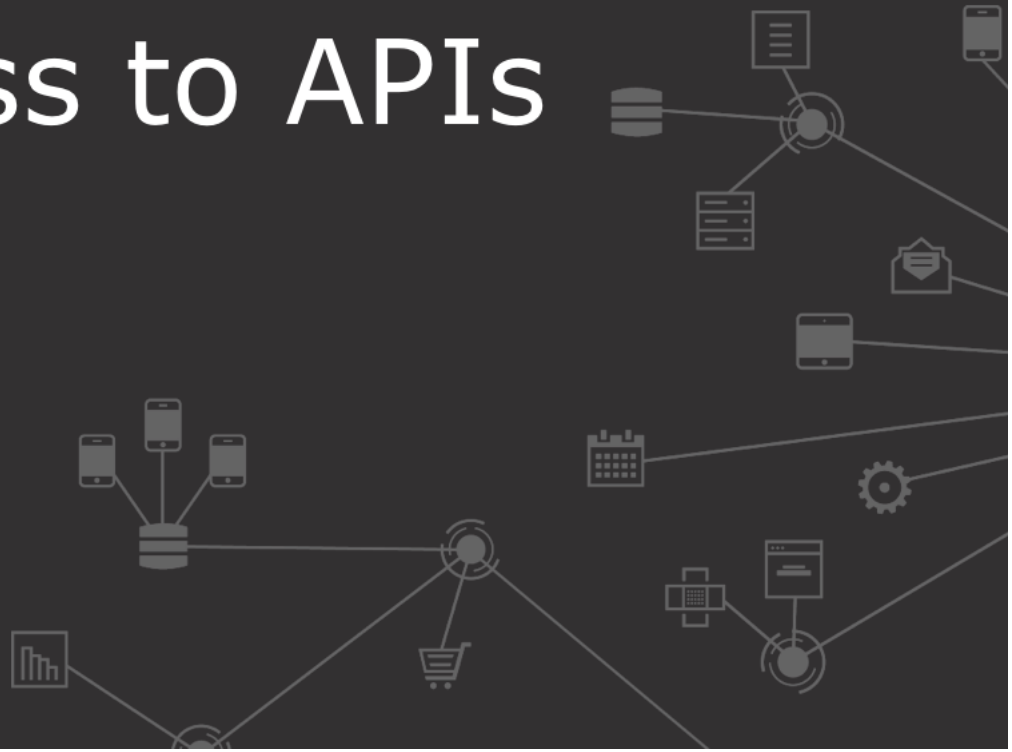
Walkthrough 5-2: Create and deploy an API proxy

- 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 and external developers
- View API request data in API Manager.



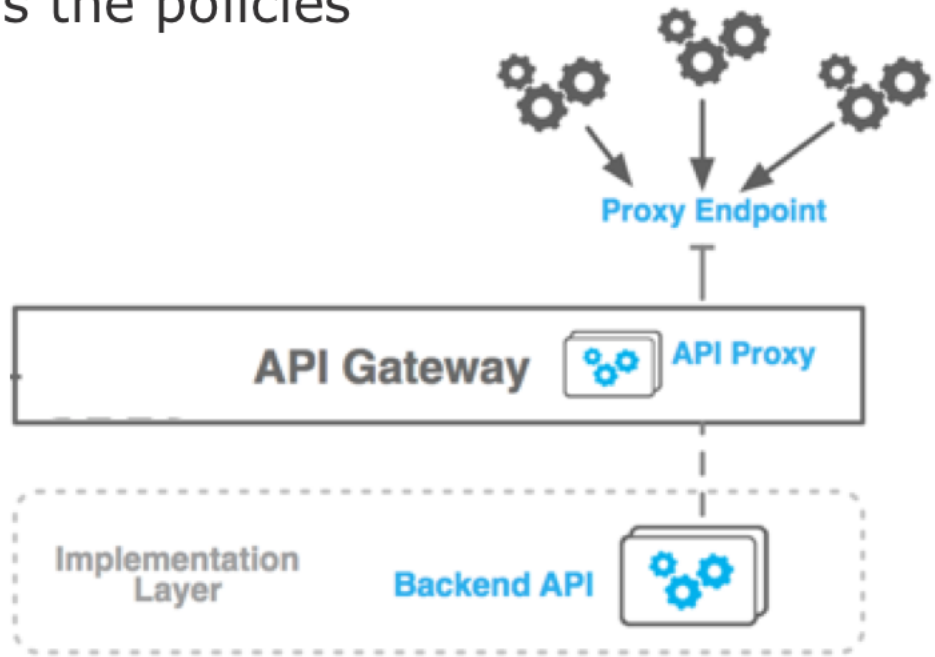


Restricting access to APIs



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

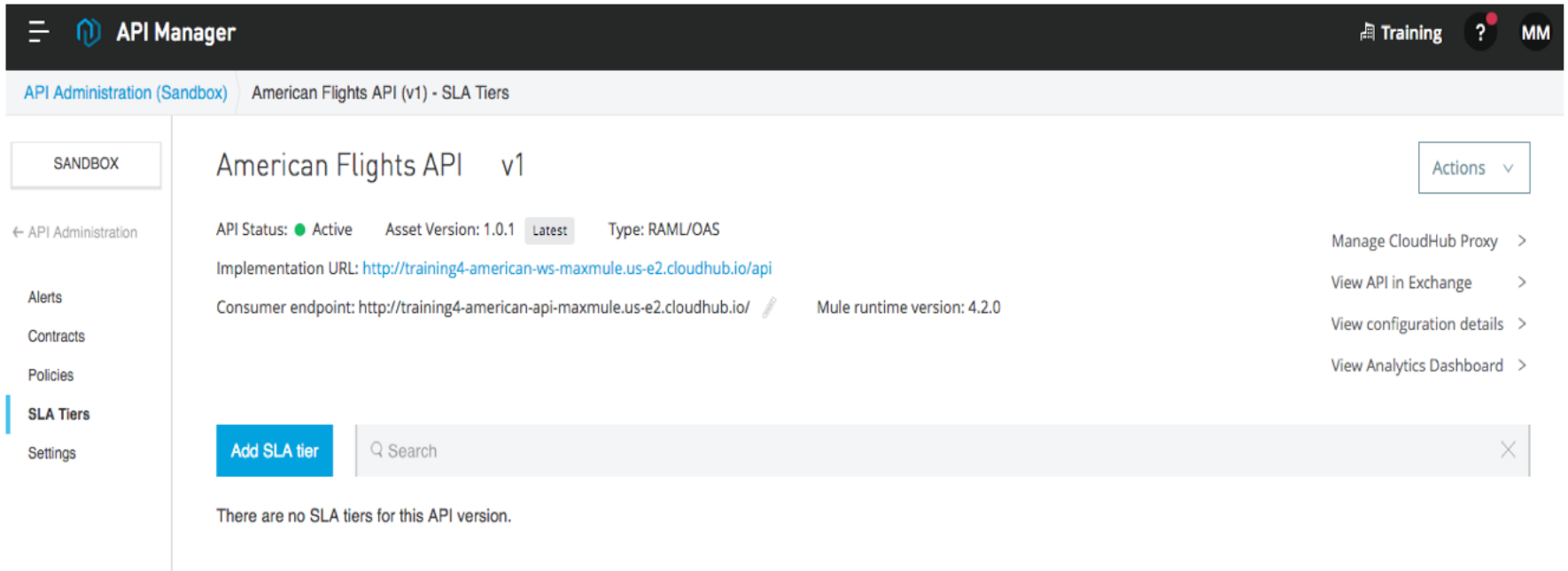
MuleSoft

- 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	

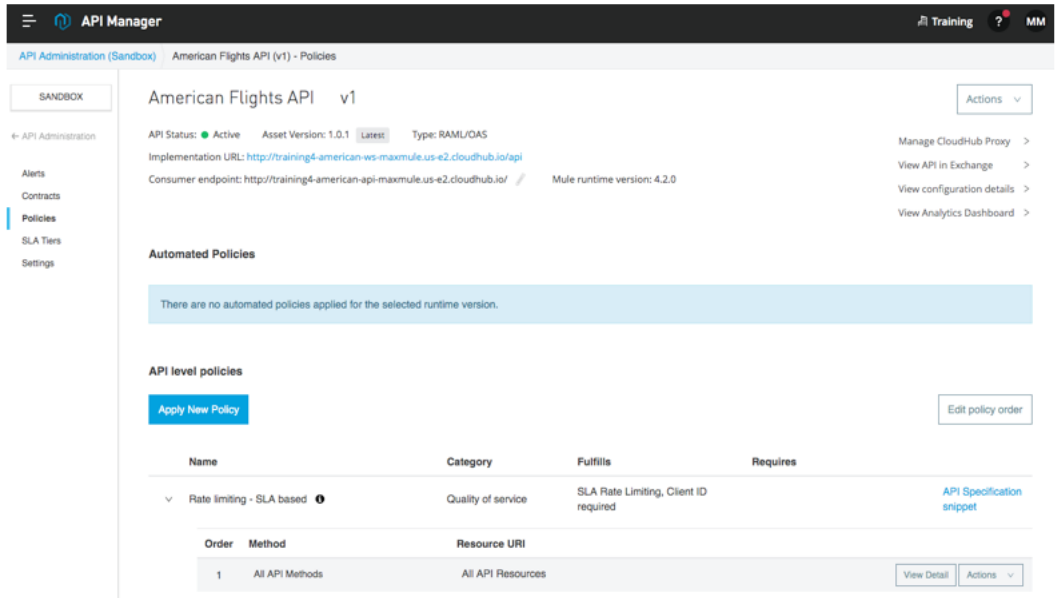
Using SLA tiers to restrict access by client ID

- A **S**ervice **L**evel **A**greement 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 $)

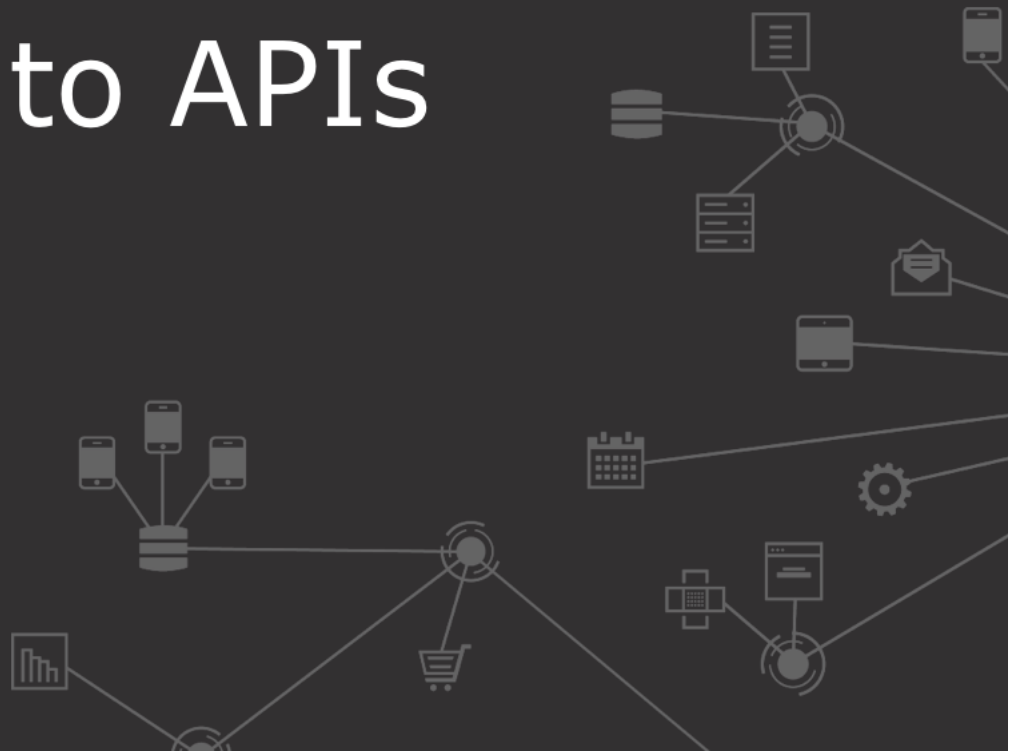


Walkthrough 5-3: Restrict API access with policies and SLAs

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

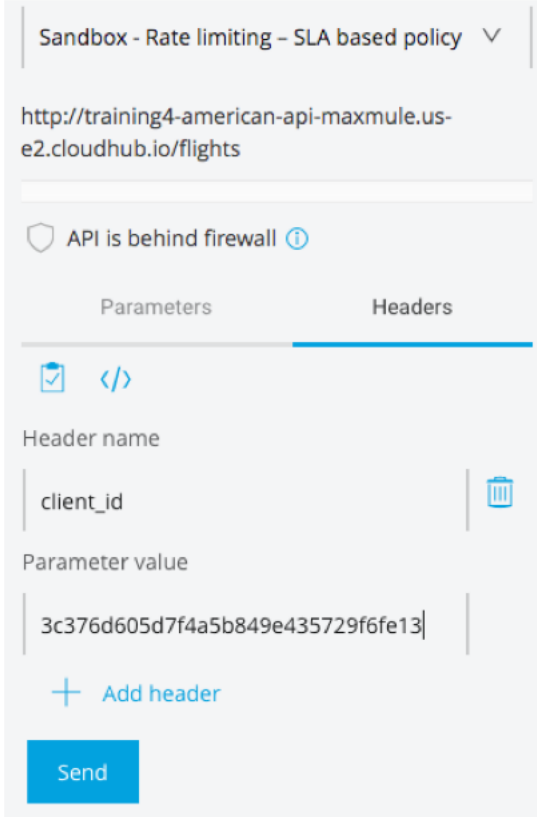


Granting access to APIs



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

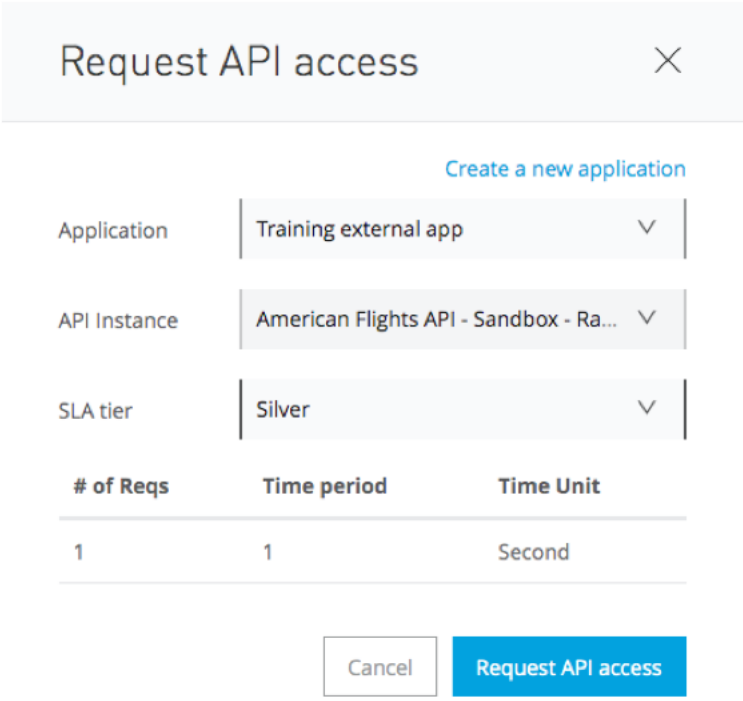


Requesting access to SLA tiers

- If an API has an SLA-based policy, a Request API access button appears in API portal

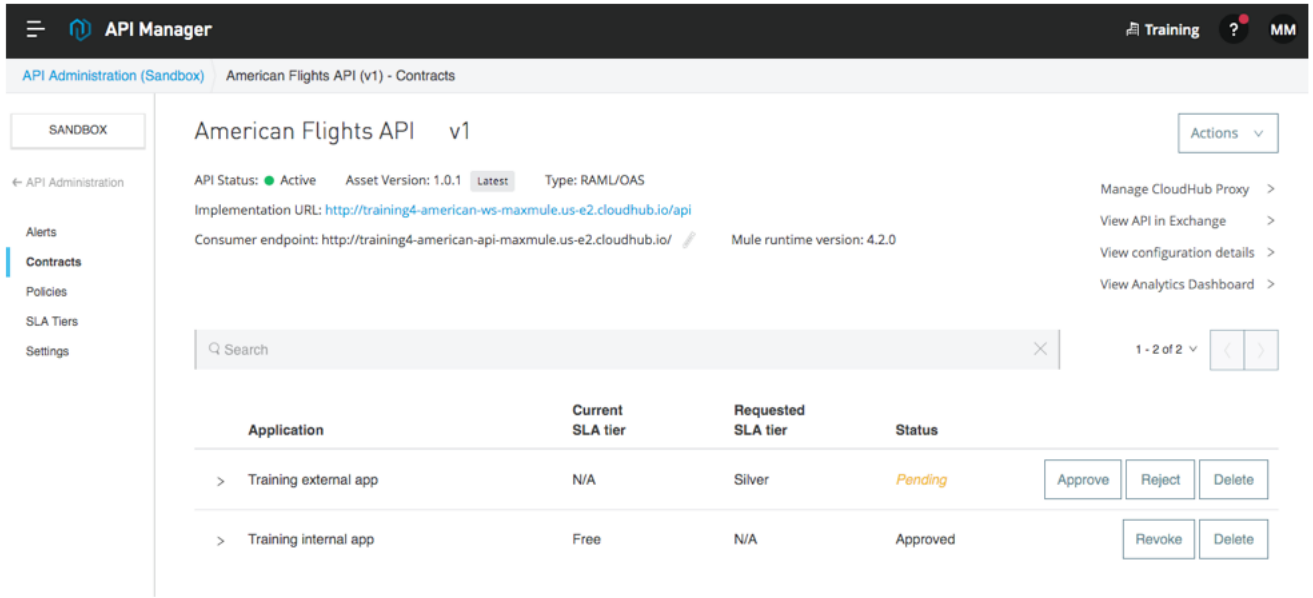
- To request access, developer must belong to the Anypoint Platform organization and be logged in

- When developers request access, they must
  - Register/add an app to their Anypoint Platform account
  - Select a tier



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



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

API Manager | Training | ? | MM

API Administration (Sandbox) | American Flights API (v1) - Contracts

SANDBOX

← API Administration

Alerts
Contracts
Policies
SLA Tiers
Settings

American Flights API v1

Actions

API Status: Active | Asset Version: 1.0.1 Latest | Type: RAML/OAS

Implementation URL: http://training4-american-ws-maxmule.us-e2.cloudhub.io/api

Consumer endpoint: http://training4-american-api-maxmule.us-e2.cloudhub.io/ | Mule runtime version: 4.2.0

Manage CloudHub Proxy >
View API in Exchange >
View configuration details >
View Analytics Dashboard >

Search

1 - 2 of 2

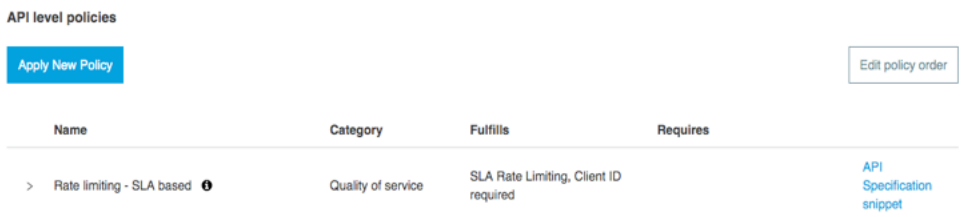
Application	Current SLA tier	Requested SLA tier	Status	
> Training external app	N/A	Silver	Pending	Approve Reject Delete
> Training internal app	Free	N/A	Approved	Revoke Delete

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



RAML 0.8 RAML 1.0 OAS 2.0

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

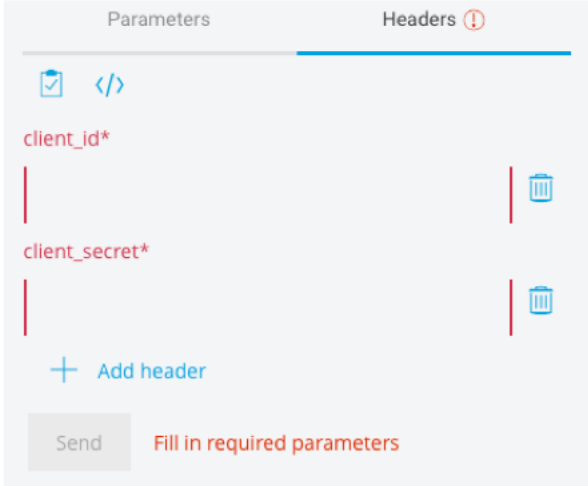
```
traits:
  client-id-required:
    headers:
      client_id:
        type: string
      client_secret:
        type: string
    responses:
      401:
        description: Unauthorized, The client_id or client_secret are not valid or the client does not
have access.
      429:
        description: The client used all of it's request quota for the current period.
      500:
        description: An error ocurred, see the specific message (Only if it is a WSDL enpoint).
      503:
        description: Contracts Information Unreachable.
```

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

```
1   #%RAML 1.0
2   version: v1
3   title: American Flights API
4
5   types:
6     AmericanFlight: !include
    exchange_modules/68ef9520-24e9-4cf2-b2f5-620025690
7
8   traits:
9     client-id-required:
10      headers:
11        client_id:
12          type: string
13        client_secret:
14          type: string
15      responses:
16        401:
17          description: Unauthorized, The client_id o
18        429:
19          description: The client used all of it's r
20        500:
21          description: An error ocurred, see the spe
22        503:
```



Summary

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

Summary

- 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

Anypoint Platform Operations training courses

- This module was just an introduction to deploying and managing applications and APIs

- Anypoint Platform Operations:
  - CloudHub
  - Customer-Hosted Runtimes
  - API Management