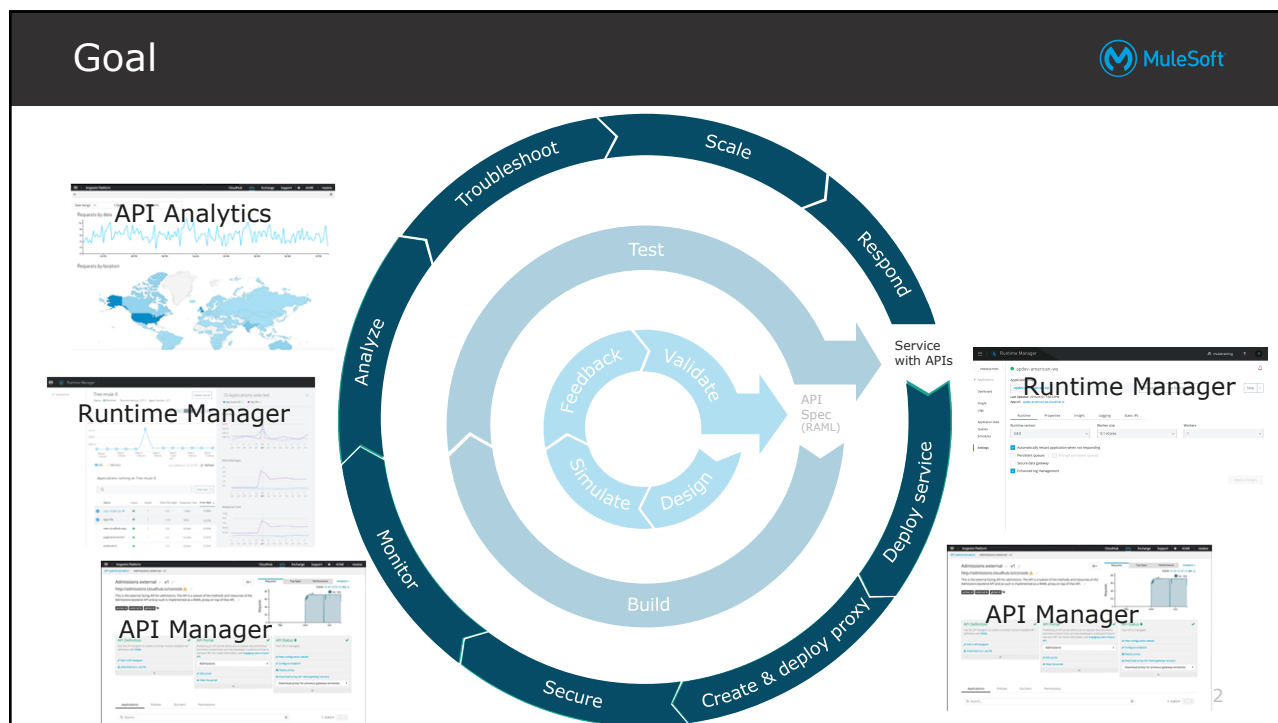




# Module 5: Deploying and Managing APIs



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



- Describe the options for deploying Mule applications
- Use properties in Mule applications so they can easily move between environments
- Deploy Mule applications to CloudHub
- Use API Manager to create and deploy proxies for APIs
- Use API Manager to restrict access to API proxies

All contents © MuleSoft Inc.

3

## 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
  - **Customer-hosted Mule runtimes**
  - **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 runtime



MuleSoft-hosted runtime

All contents © MuleSoft Inc.

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



All contents © MuleSoft Inc.

6

## 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



All contents © MuleSoft Inc.

7

## Preparing an application for deployment



## Before deploying



- Think about anything in your application that might change between development and production...

Global Element Properties

**MySQL Configuration**  
MySQL configuration information.

General Advanced Reconnection Notes

Generic

Name:

General

☒ Database configuration parameters

Host:

Port:

User:

Password:  ☐ Show password

Database:

All contents © MuleSoft Inc.

9

## Using application properties



## Application properties



- Are an alternative to hard-coding credentials & resources
- Are injected into the application at runtime
- Provide an easier way to manage credentials, changes, and settings
- Can be encrypted
- Are defined in .properties files
  - Separate property files can host values specific to an environment
    - app-dev.properties and app-prod.properties

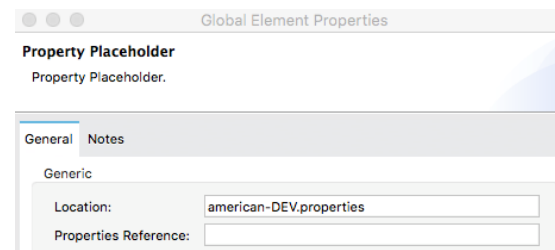
All contents © MuleSoft Inc.

11

## Defining application properties



- Create a properties file in the src/main/resources folder  
`american-DEV.properties`
- Define properties in the properties file  
`db.port = 3306`  
`db.user = mule`
- Create a Properties Placeholder global element
- Use the properties in the application  
`${db.account}`



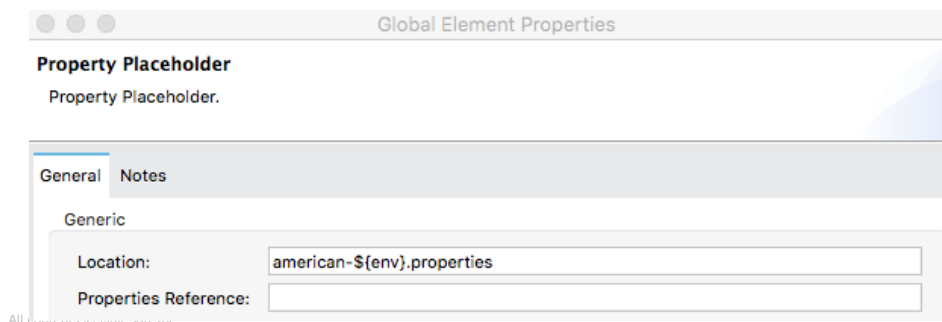
All contents © MuleSoft Inc.

12

## Dynamically specify property files



- Resources and credentials often vary from development to production environments
- You can use environment variables in an application whose values must be set when the application starts
- You can use a dynamic value for the location in the Property Placeholder



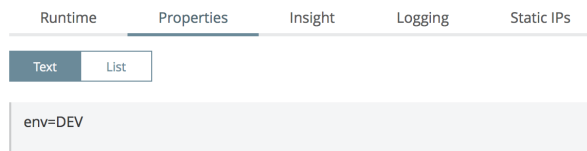
13

## Setting environment variables



- For development, set environment variables in mule-app.properties

`env = DEV`



- For deployment to **CloudHub**
  - Set the values in the Runtime Manager properties tab for the application before you start it
  - If you deploy to CloudHub from Anypoint Studio, these values get automatically added as properties for the application in the Runtime Manager
- For deployment to a **customer-hosted Mule runtime**
  - Variables must be passed to Mule runtime when it starts
  - Set them in wrapper.conf before starting Mule

All contents © MuleSoft Inc.

14

## Do you have to modify HTTP Listener connectors?



- No
- CloudHub routes all requests to your application domain URL on port 80 to an endpoint with the matching path that was configured with a host of 0.0.0.0 and port 8081
- If you use a port other than 8081, you need to set the port in a reserved application property called `http.port` or `https.port`
  - Traffic on port 80 to a CloudHub application domain URL will then be routed to the port set by that property

### URL Configuration

Protocol:	<input checked="" type="radio"/> HTTP (Default) <input type="radio"/> HTTPS
Host:	All Interfaces [0.0.0.0] (Default) <span>▼</span>
Port:	8081
Base Path:	

All contents © MuleSoft Inc.

## Walkthrough 5-1: Prepare an application for deployment using properties



- Create a properties file for an application
- Create a Properties Placeholder element to specify the properties file
- Define and use Database connector properties
- Specify a properties file dynamically

Global Element Properties

**MySQL Configuration**  
MySQL configuration information.

General Advanced Reconnection Notes

Generic

Name: MySQL\_Configuration Interface Implementation american-DEV.properties x

General

Database configuration parameters

Host: \${db.host}

Port: \${db.port}

User: \${db.user}

Password: ••••• ☐ Show password

Database: \${db.database}

1 http.port = 8081  
2  
3 db.host = mddb.mulesoft-training.com  
4 db.port = 3306  
5 db.user = mule  
6 db.password = mule  
7 db.database = training

Test Connection... Cancel OK

All contents © MuleSoft Inc.

16



# 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 for an application named 'apdev-american-ws'. The interface includes a sidebar with navigation options like Applications, Dashboard, Insight, Logs, and Settings. The main content area displays the application's details, including the application file 'apdev-american-ws.zip', the last updated time, and the app URI. Below this, there are tabs for Runtime, Properties, Insight, Logging, and Static IPs. The 'Runtime' tab is active, showing the runtime version (3.8.0), worker size (0.1 vCores, 500 MB memory), and the number of workers (1). There are also checkboxes for 'Automatically restart application when not responding', 'Persistent queues', 'Secure data gateway', and 'Enhanced log management'.

18

## Recall: 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

All contents © MuleSoft Inc.

## Walkthrough 5-2: 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 the API implementation deployed to CloudHub

Name	Server	Status	File
training-american-ws	CloudHub	Started	training-american-ws.zip

All contents © MuleSoft Inc.

20

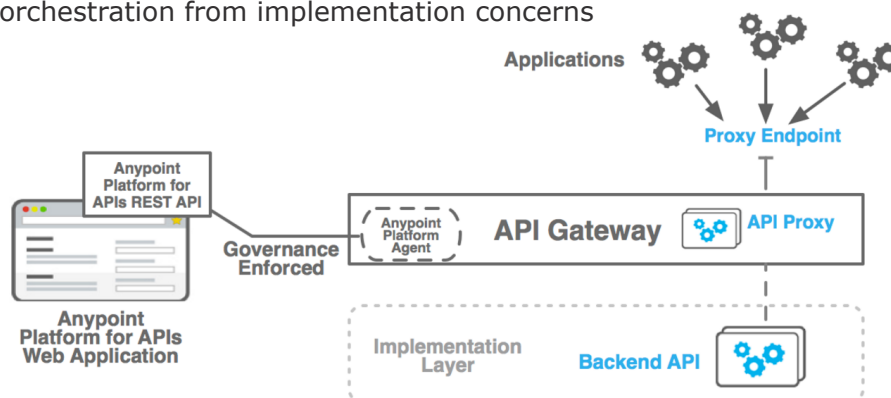
# Creating API proxies



## 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
  - Separates orchestration from implementation concerns



All contents © MuleSoft Inc.

22

## The API Gateway is the point of control

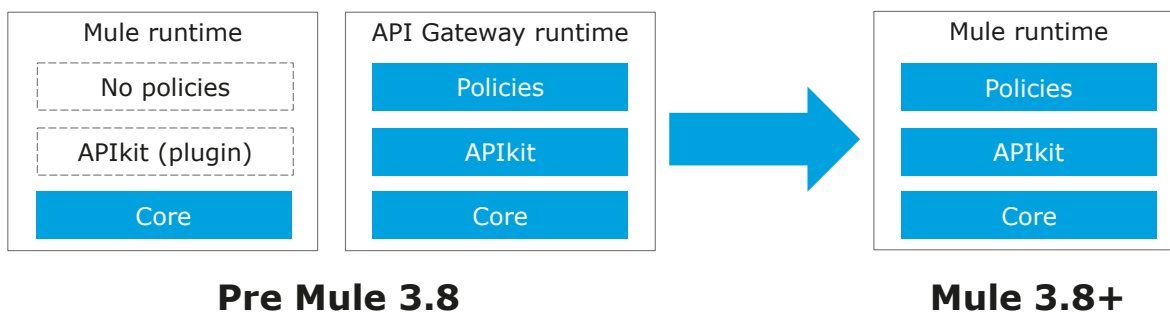


- **Determines which traffic** is authorized to pass through the API to backend services
- **Meters the traffic** flowing through
- **Logs** all transactions, collecting and tracking analytics data
- Applies runtime policies to **enforce governance** like rate limiting, throttling, and caching

All contents © MuleSoft Inc.

23

## Mule runtime includes API Gateway runtime



\* Mule runtime and API Gateway runtime require separate licenses

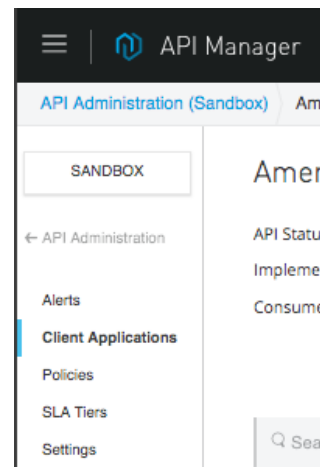
All contents © MuleSoft Inc.

24

## 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**



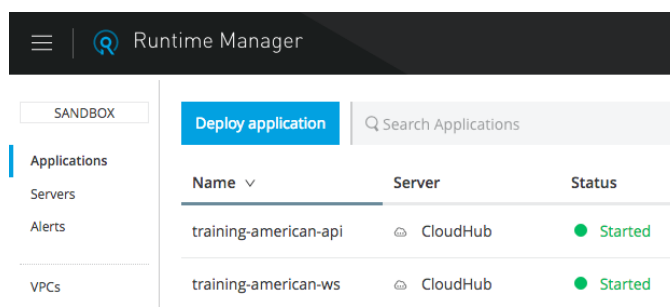
All contents © MuleSoft Inc.

25

## Walkthrough 5-3: Create and deploy an API proxy

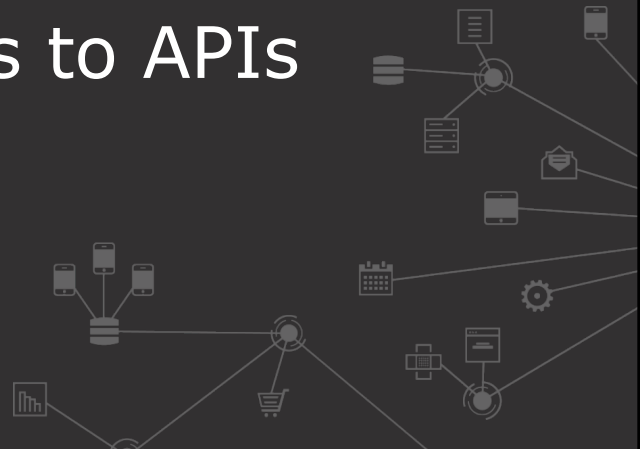


- Add an API to API Manager
- Use API Manager to automatically create and deploy an API proxy app
- Set a label and consumer endpoint for a proxy so requests can be made to it from Exchange
- Make calls to the API proxy from API portals for both internal and external developers
- View API request data in API Manager



All contents © MuleSoft Inc.

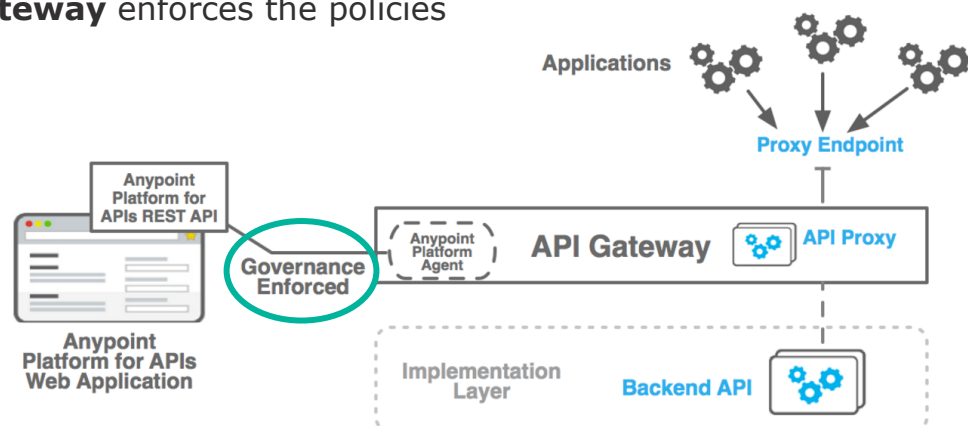
# Managing access to APIs



## Managing access to APIs



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



All contents © MuleSoft Inc.

## Applying policies to restrict access



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

Policies	Policies
<input type="radio"/> Cross-Origin resource sharing ⓘ	<input type="radio"/> IP blacklist ⓘ
<input type="radio"/> Throttling ⓘ	<input type="radio"/> IP whitelist ⓘ
<input type="radio"/> Throttling - SLA based ⓘ	<input type="radio"/> JSON threat protection ⓘ
<input type="radio"/> Rate limiting ⓘ	<input type="radio"/> LDAP security manager ⓘ
<input type="radio"/> Rate limiting - SLA based ⓘ	<input type="radio"/> OAuth 2.0 access token enforcement ⓘ
<input type="radio"/> Client ID enforcement ⓘ	<input type="radio"/> Simple security manager ⓘ
<input type="radio"/> HTTP basic authentication ⓘ	<input type="radio"/> XML threat protection ⓘ

All contents © MuleSoft Inc.

29

## 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 \$)

API Manager

API Administration (Sandbox) American Flights API (v1) - SLA Tiers

SANDBOX

← API Administration

Alerts

Client Applications

Policies

SLA Tiers

Settings

Add SLA tier

Search

1 - 2 of 2

Name	Limits	Applications	Status	Approval	
Free	1	1	Active	Auto	<a href="#">Edit</a> <a href="#">Deprecate</a>
Silver	1	1	Active	Manual	<a href="#">Edit</a> <a href="#">Deprecate</a>

View Analytics Dashboard >

All contents © MuleSoft Inc.

30

## Enforcing access to APIs using SLA tiers



- To enforce, apply an **SLA based** rate limiting or throttling 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** to calls made to the API

The screenshot shows the 'API Manager' interface for 'American Flights API (v1) - Policies'. The left sidebar includes 'API Administration (Sandbox)', 'Alerts', 'Client Applications', 'Policies', 'SLA Tiers', and 'Settings'. The main area has a table with columns: Name, Category, Fulfills, and Requires. A policy named 'Rate limiting - SLA based' is listed under the 'Quality of service' category. Below this, a table shows the policy details: Order (1), Method (All API Methods), and Resource URI (All API Resources). Buttons for 'Apply New Policy', 'Edit policy order', 'View Detail', and 'Actions' are visible.

## Walkthrough 5-4: 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 policy
- Request application access to SLA tiers from private & public API portals
- Approve application requests to SLA tiers in API Manager

The screenshot shows the 'API Manager' interface for 'American Flights API (v1) - Client Applications'. The left sidebar includes 'API Administration (Sandbox)', 'Alerts', 'Client Applications', 'Policies', 'SLA Tiers', and 'Settings'. The main area has a table with columns: Application, Current SLA tier, Requested SLA tier, and Status. A client application named 'Training external app' is listed with a 'Silver' current tier and 'N/A' requested tier, with a status of 'Approved'. Below this, a table shows the application details: Owners (Max Mule, jeanette.stallons@mulesoft.com), Client ID (2f38b1ca94814b4aad02aab4e0c995a1), URL (None), and Redirect URIs (None). Buttons for 'Revoke' and 'Submitted' are visible.

32



# Making calls to APIs with client ID enforcement policies from API portals

## Walkthrough 5-5: Make calls to an API with a client ID based policy from API portals



- Add authentication query parameters to an API specification
- Update a managed API to use a new version of an API specification
- Call a governed API with client credentials from API portals

The screenshot displays the MuleSoft API portal interface. On the left, the 'Parameters' tab is active, showing query parameters for a GET request. The 'client\_id' is set to 'e708026bb0cf4c3e8594ca39138b9a00' and the 'client\_secret' is 'e10A1faF382D47DEA6A90cA8d4FC3891'. A 'Send' button is at the bottom. On the right, the response is shown as a 200 OK status with a response time of 1486.61 ms. The response body is a JSON array containing two objects, each representing a flight with details like ID, code, price, departure date, origin, destination, empty seats, plane type, and total seats.

```

[Array[1]]
-0: {
  "ID": 1,
  "code": "rree0001",
  "price": 541,
  "departureDate": "2016-01-20T00:00:00",
  "origin": "MUA",
  "destination": "LAX",
  "emptySeats": 0,
  "plane": {
    "type": "Boeing 787",
    "totalSeats": 200
  }
},
-1: {
  "ID": 2,
  "code": "eefd0123",
  "price": 541,
  "departureDate": "2016-01-20T00:00:00",
  "origin": "MUA",
  "destination": "LAX",
  "emptySeats": 0,
  "plane": {
    "type": "Boeing 787",
    "totalSeats": 200
  }
}

```

All contents © MuleSoft Inc.

35

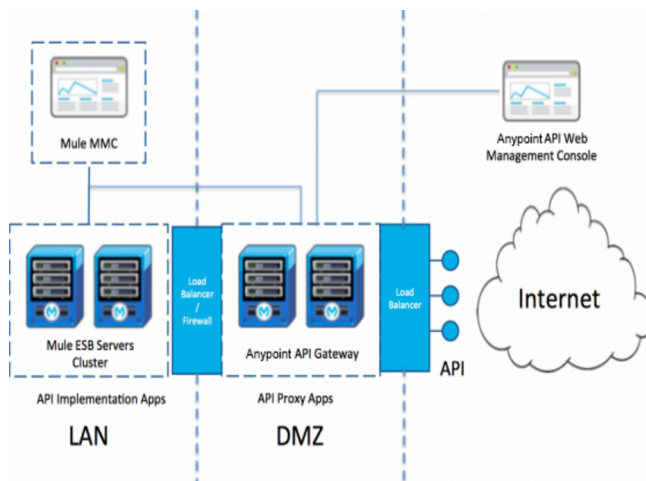
# Restricting access to API implementations



## On-prem API implementations and API proxies



- Set up an API Gateway cluster inside a DMZ to run the API proxy applications



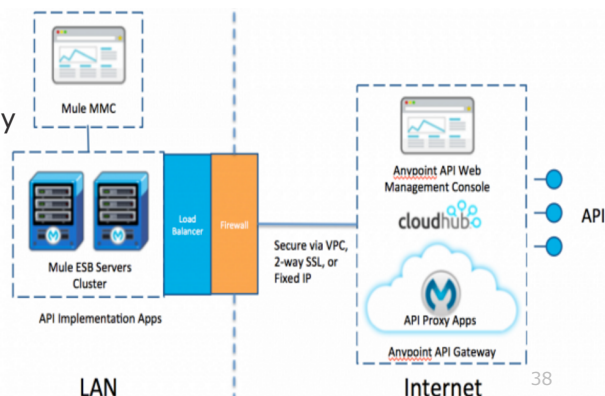
All contents © MuleSoft Inc.

37

## On-prem API implementations and cloud API proxies



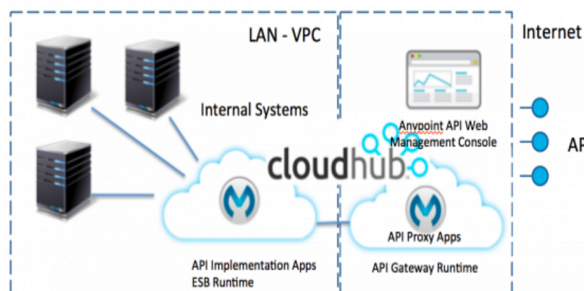
- Set up secure communication between the proxy applications and the internal on-prem runtimes using a Virtual Private Cloud (VPC)
  - A VPC is a private and isolated network of your CloudHub workers
- Connect this network to other VPCs or data centers via a VPN connection
  - This allows CloudHub workers to access resources behind a corporate firewall
  - You can use an IPSec gateway or AWS Direct Connect for VPN connectivity
- See here for setting up VPC  
<https://docs.mulesoft.com/runtime-manager/virtual-private-cloud>



## Cloud API implementations and API proxies



- Option 1
  - Do not use separate API proxy applications and instead specify policies for the service API implementation applications
- Option 2: Use VPC
  - Leave the workers running API proxy applications outside the VPC and put the workers running API implementations inside the VPC
  - Use ports 8091 or 8092 in your API implementations



# Summary



## Summary: Deployment



- 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
- Use **application properties** to avoid hard-coding endpoint properties, credentials, and resources
  - Define them in a .properties file whose location is specified in a Properties Placeholder global element
  - Dynamically specify a properties file when the application starts by parameterizing its name and setting the variable
    - As an application property with the Runtime Manager
    - As an argument in on-prem Mule runtime wrapper.conf file

## Summary: Access control



- 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

All contents © MuleSoft Inc.

42

## Anypoint Platform Operations training courses



- This module was just an introduction to deploying and managing applications and APIs
- Anypoint Platform Operations:
  - CloudHub (1 day in-person or 2 days online)
  - Customer-Hosted Runtimes (2 days)
  - API Management (1 day)



All contents © MuleSoft Inc.

43