Anypoint Platform Architecture: Application Networks

Module 5
Enforcing NFRs on the Level of API Invocations Using Anypoint API Manager

Objectives

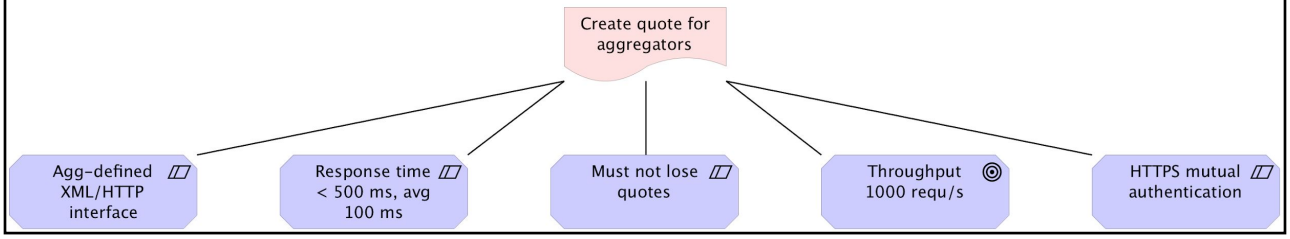
- Describe how **API Manager** controls API invocations
- Use **API policies** to enforce non-functional constraints on API invocations
- Choose between **enforcement of API policies** in an API implementation and an **API proxy**
- Register an **API client** for access to an API version
- Describe when and how to pass **client ID/secret** to an API
- Establish **guidelines for API policies** suitable for System APIs, Process APIs, and Experience APIs

Addressing the NFRs of the "Aggregator Integration" product

NFRs for "Create quote for aggregators"

MuleSoft

- Synchronous creation of up to **5 quotes**:
  - Aggregator-defined **XML**-formatted policy description in HTTP POST request
  - **Up to 5 quotes** in Aggregator-defined XML format in HTTP response
- **Performance**:
  - Throughput: up to **1000 requs/s**
  - Response time: median = **200 ms**, maximum = **500 ms** at 1000 requs/s
- **Security**: **HTTPS mutual authentication**
- **Reliability**: quotes are legally binding and **must not be lost**



Meeting NFRs for "Create quote for aggregators" using Anypoint Platform

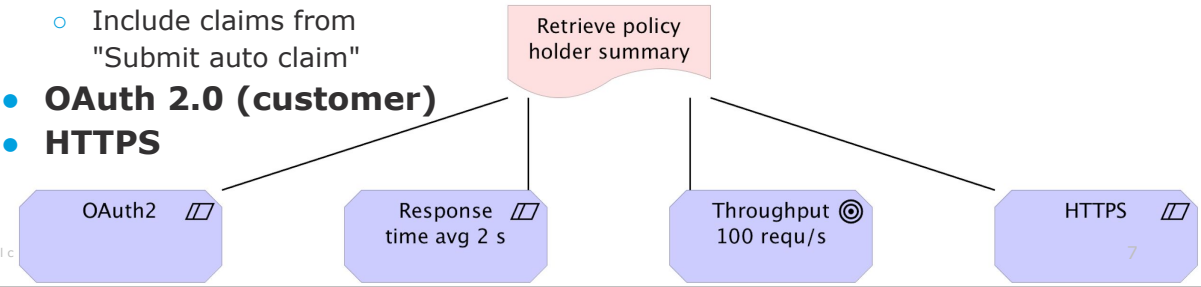
MuleSoft

- **Throughput and response time**:
  - Must be broken-down to APIs in **all tiers**
  - Must be **enforced, monitored and analyzed**
    - API Manager, Anypoint Analytics
  - Anticipate **caching**
  - Highly **performant** runtime plane for API implementations: **CloudHub**
  - Need to carefully manage **load on Policy Admin System**: API Manager
- Must not lose quotes:
  - Synchronous invocations incl. ACID operation on **Policy Admin System**
- HTTPS mutual authentication:
  - **CloudHub Dedicated Load Balancer**
- Should add **client authentication** on top of HTTPS mutual auth

Addressing the NFRs of the "Customer Self-Service App" product

NFRs for "Retrieve policy holder summary"

- Part of "Customer Self-Service App" product
  - Might be opened-up to **external API consumers**
- **Synchronous** HTTP request-response chain
- **Performance**:
  - Ill-defined, aim for **100 requs/s**
  - Aim for avg response time of **2 s** at 100 requs/s
- **Consistency**:
  - Include claims from "Submit auto claim"
- **OAuth 2.0 (customer)**
- **HTTPS**

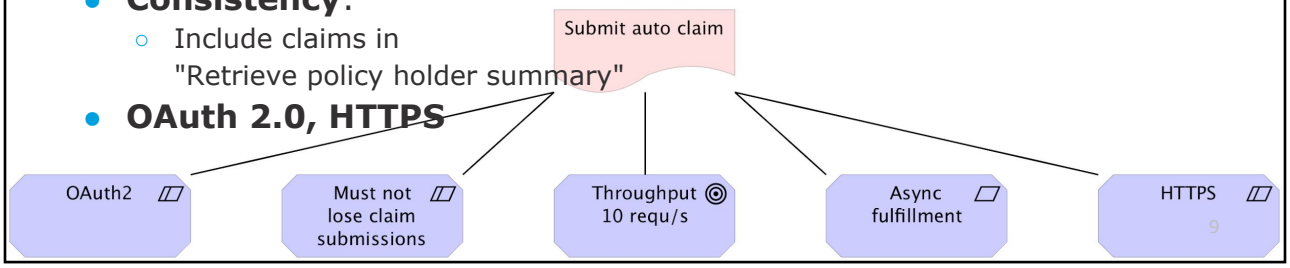


Meeting the NFRs for "Retrieve policy holder summary" using Anypoint Platform

- **Throughput and response time**:
  - **Not challenging**
  - Future use may change that
  - Highly **scalable** runtime plane: **CloudHub**
- **HTTPS**:
  - **Document** in RAML definition
  - Ensure in **API implementation**
- **OAuth 2.0**:
  - Enforce with **API Manager**
  - Requires Identity Provider for **Client Management**
    - PingFederate
- **Consistency**:
  - Through **event notifications**

NFRs for "Submit auto claim"

- Request over HTTP with claim submission and **asynchronous processing** of the submission
  - Processing submission requires lengthy downstream processing steps
- **Performance**:
  - Ill-defined, aim for **10 requs/s**
  - **No response time requirement** because processing is asynchronous
- Reliability: claim submissions **must not be lost**
- **Consistency**:
  - Include claims in "Retrieve policy holder summary"
- **OAuth 2.0, HTTPS**



Meeting the NFRs for "Submit auto claim" using Anypoint Platform

New NFRs for this feature:

- **Async processing** of claim submission and no claim submission loss:
  - **Messaging system**
    - To trigger **async processing without message loss**
    - **Anypoint MQ**
    - Mule runtime **persistent VM queues** as in CloudHub
  - **Persistence mechanism**
    - To store async **correlation** information
    - Mule runtime **Object Store** as in CloudHub
- **Consistency**:
  - Through **event notifications**

Using API Manager and API policies to manage API invocations

Reviewing types of APIs

MuleSoft

- **REST** APIs
  - With API specification as **RAML** definition or **OAS** definition
  - **Without formal API specification**
  - **Hypermedia**-enabled REST APIs
- **Non-REST** APIs
  - GraphQL APIs
  - SOAP web services (APIs)
  - JSON-RPC, gRPC, …

API management on Anypoint Platform

MuleSoft

- Using **API Manager** and **API policies**
- On the level of **HTTP**
- Applicable to **all types of HTTP/1.x APIs**
  - Therefore not to WebSocket APIs or HTTP/2 APIs
- Special support for **RAML-defined APIs**
  - Allow definition of **resource-level** API policies
  - In addition to the **endpoint-level** API policies available for all APIs

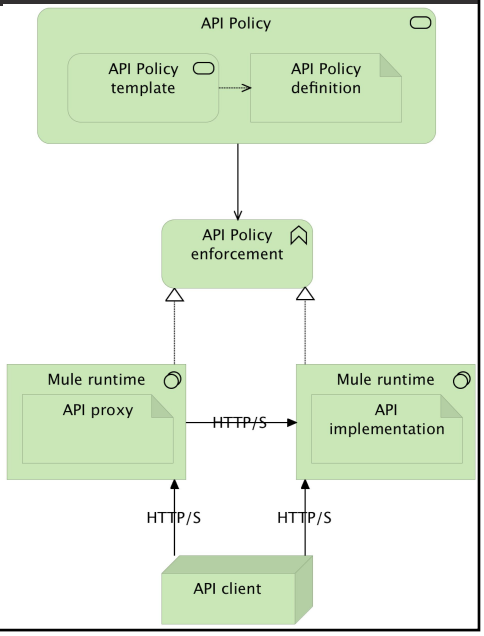
Defining API policy

MuleSoft

- Defines a typically **non-functional requirement**
- Applied to an **API** (instance)
- Injection into **API invocation** between API client and endpoint
  - Without changing API implementation
- Consists of
  - API policy **template** (code and parameter descriptions)
  - API policy **definition** (parameter values)

Enforcement of API policies

- On Anypoint Platform, API policies are always **enforced from within a Mule app**:
  - **API implementation** can **embed** enforcement of API policies
  - **API proxy** deployed infront of the API implementation proper to enforce API policies
- API policies **downloaded at runtime** from API Manager



Exercise: Pros and cons of policy enforcement sites

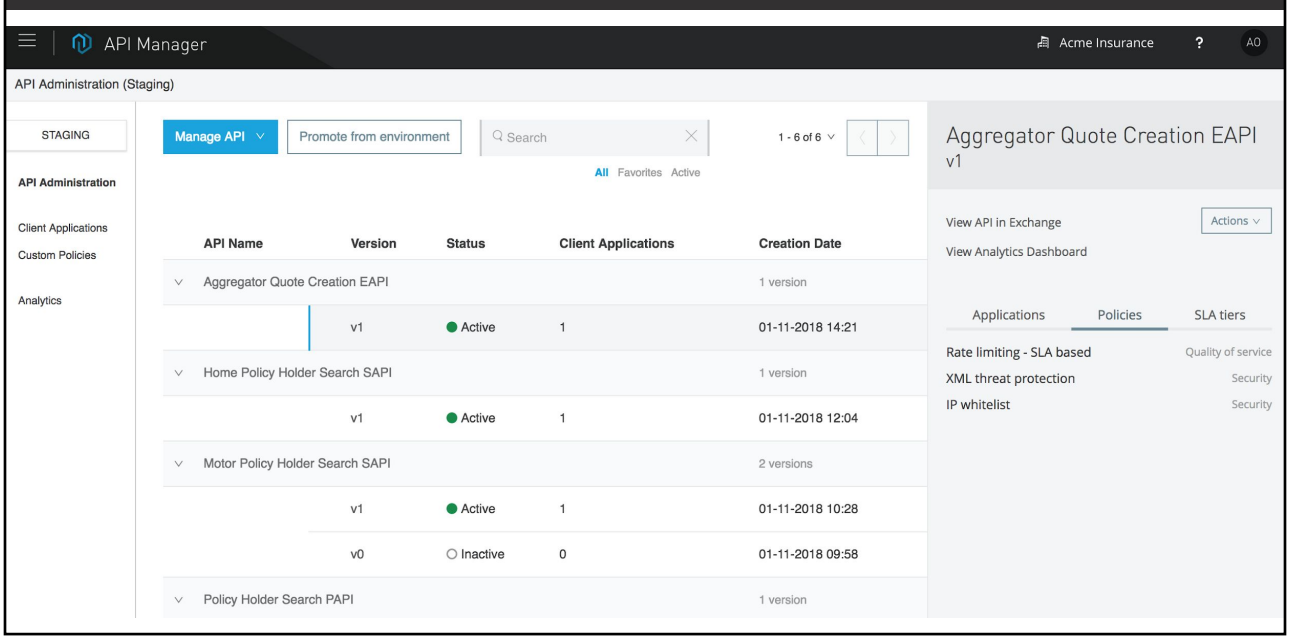
Compare the characteristics of the two sites of API policies enforcement available in Anypoint Platform:

- List scenarios/requirements that would be best addressed by API policy enforcement **embedded in the API implementation** or in an **API proxy**, respectively

Solution: Pros and cons of policy enforcement sites

- API implementations are **not Mule apps**
- **Resources** must be minized
- **Deployment and CI/CD** must be as simple as possible
- API policies with special **resource requirements** are applied
  - Caching API policy
  - Security API policy requiring HSM
- API policies require **special network configuration**
- **Security sensitive (Experience) APIs**
  - Deployment to **DMZ**
  - **Shield API implementations** from attacks

Managing APIs with API Manager



Managing APIs with API Manager

MuleSoft

- Management of APIs using **API instances**
  - **API instance** = endpoint for API with major version in environment
- Configuration of **API policies** for a given API instance
  - Select API policy template and parameterize it with API policy definition
  - **OOTB and custom** API policies
- Contacted from site of API policy enforcement to **download all API policies** that must be enforced
- Definition of **alerts** based on API invocations

Managing APIs with API Manager

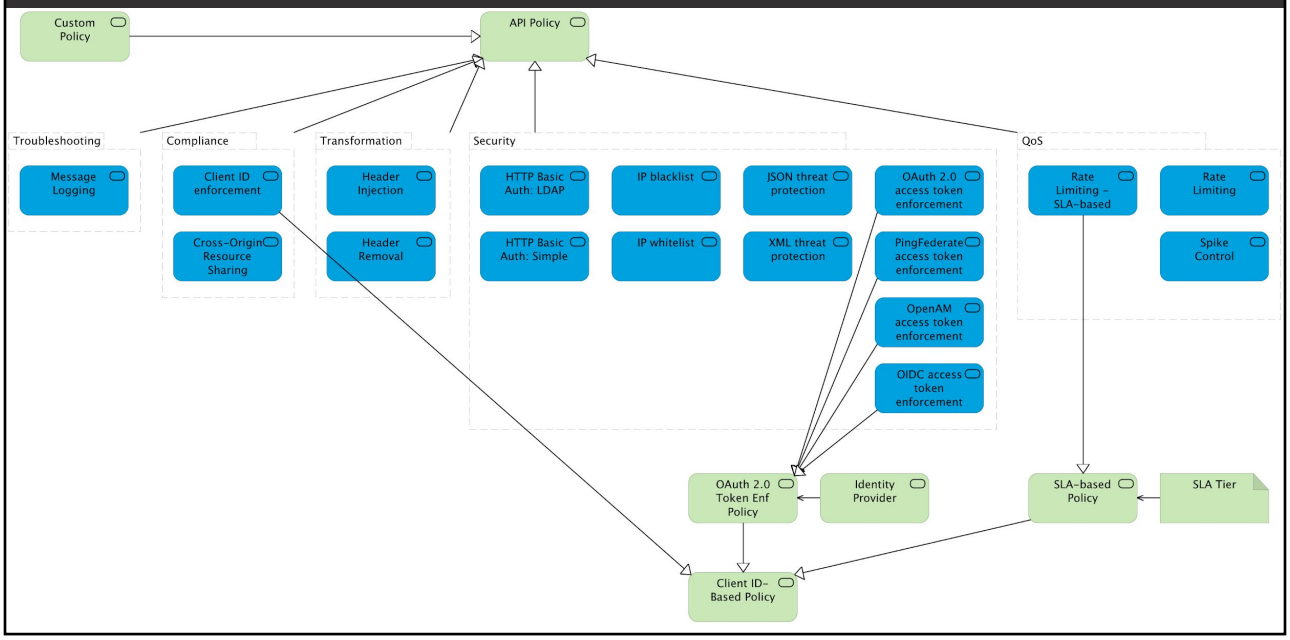
MuleSoft

- Admin of **API clients** ("Client Applications")
  - API consumers use Exchange to request access
- API consumers use **Exchange to request access** to an API
- Access to Anypoint **Analytics**

Selectively applying an API policy to some resources and methods of an API

- By default API policies are applied to **entire API endpoint**
  - Represented as API instance in API Manager
- APIs defined with a RAML definition can apply API policies also to selected combinations of **API resources and HTTP methods**
- **OpenAPI** documents can be converted to **RAML definitions**

API policies



API policies as Aspect-Oriented Programming

- API policies are **AOP** applied to API invocations:
  - **Ordered**, API implementation/proxy as last element
  - **Incoming HTTP request** passed down this chain, returning **HTTP response** passed up
  - API policies implement "**around advice**":
    - Execute code **before/after** handing control to the **next element** in the chain
    - **Change HTTP request/response** if desired
  - In Mule 4: also applied to **outgoing HTTP requests**

Understanding custom API policies

- **Implementing and applying** custom API policies:
  - Very similar to **Mule apps**
  - Packaged and deployed to **Exchange**
    - Contains both **policy template** (code and parameter descriptions)
  - **API Manager** retrieves policy from Exchange and shows **configuration UI** to enter the definition (parameter values)
  - Policy template and definition **downloaded to any Mule runtime** that registers as that API instance

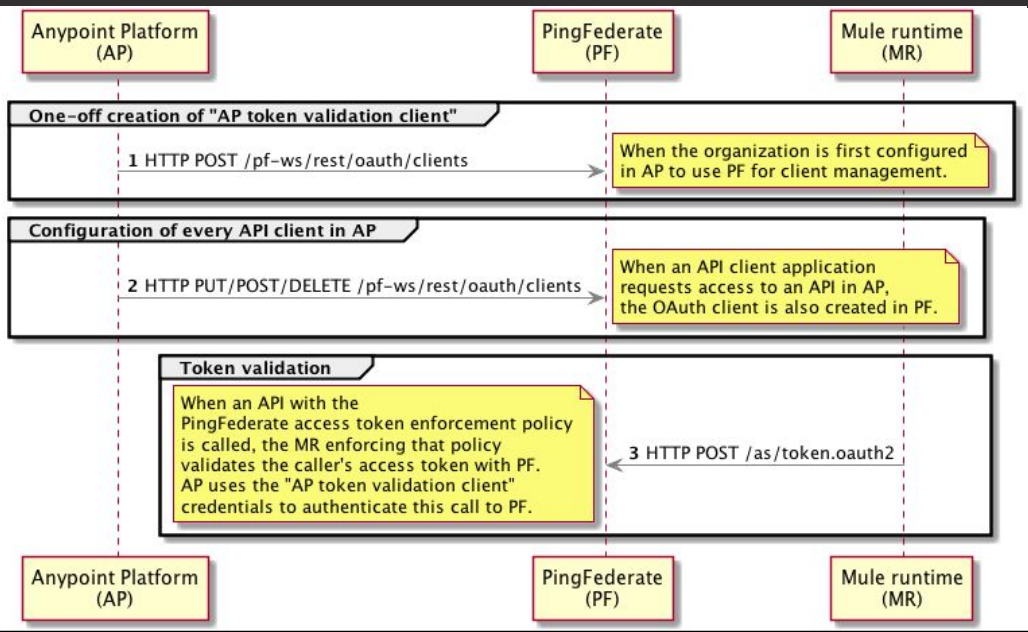
Compliance-related API policies

- **Client ID enforcement**
- **CORS control**
  - Interacts with API clients for **Cross-Origin Resource Sharing**:
    - Rejects HTTP requests whose **Origin** request header does not match configured origin domains
    - Sets **Access-Control-*** HTTP response headers to match configured cross-origins, usage of credentials, etc.
    - Responds to CORS pre-flight **HTTP OPTIONS requests**
  - Can be important for Experience APIs invoked from a **browser**

Security-related API policies

- Authentication/Authorization
  - **OAuth 2.0 token enforcement** API policies
    - Require matching Identity Provider configured for **Client Management**
      - OpenAM, PingFederate or OIDC DCR compatible (Okta)
    - Discouraged "OAuth 2.0 access token enforcement using external provider" requires access to Mule OAuth 2.0 provider or other configured in the policy
  - **Basic Authentication: LDAP/Simple**
    - Incorporate access to Identity Provider
- **IP-based** access control
  - **blacklisting, whitelisting**
- Payload **threat protection**
  - Guard against attacks sending over-sized HTTP request bodies
  - **Limit size of XML or JSON bodies**

Interactions with OAuth 2.0 Client Management



QoS-related API policies

- Quality of Service (QoS) related API policies on Anypoint Platform enforce **throughput limit** in # of API invocations per unit of time:
  - **Rate Limiting**: rejects requests above limit
  - **Spike Control**: queues requests above limit
- Two different ways to define the throughput limit:
  - **Non-SLA-based** (Rate Limiting and Spike Control)
    - Limit defined on API policy definition
    - Enforced for that API instance **across all API clients**
  - **SLA-based** (Rate Limiting)
    - Limit defined in an **SLA tier**
      - **API clients must register** with the API instance at a particular SLA tier
    - **Enforced separately** for each registered API client
      - API client must identify itself with **client ID**
- **X-RateLimit-*** HTTP response headers optionally inform API client of remaining capacity

Anypoint Platform SLA tiers for APIs

- SLA tiers
  - Enable **different API clients** to receive **different QoS**
  - Define one or more **throughput limits**
    - Per API client and API instance
- API instance with SLA tiers requires every **API client to register** for access with exactly one SLA tier
  - Manual or automatic **approval**
  - API clients must send **client ID/client secret** in API invocations
  - API client is promised the QoS offered by that SLA tier
- Enforcement by **SLA-based Rate Limiting** API policy
- Violation of SLA **monitored, reported and alerted-on**

Registering API clients with an Anypoint Platform-managed API

- API clients must **register to invoke** API instance with Client ID-based API Policies
  - Called "application" or "client application"
- Request access **through Exchange** entry for that API
  - Directly from Exchange or via Public (Developer) Portal
- Access **approval** is automatic or manual
- API consumer receives **client ID and client secret**
  - Must be supplied by that API client in all API invocations to that API version in that environment

Registering API clients with an Anypoint Platform-managed API

MuleSoft

Aggregator Quote Creation EAPI v1

Share Download Edit Request access

- Requires HTTPS mutual a

Request API access

Create a new application

Application	Aggregator
API Instance	Staging - v1:7484080
SLA tier	Standard

# of Reqs	Time period	Time Unit
1000	1	Second

Cancel | Request API access

Overview

Type	REST API
Created By	AnySurance Owner
Published On	Jan 11, 2018
Visibility	Private

Asset versions for v1

Version	Instances
1.0.1	Mocking Service Staging - v1:7484080

Registering API clients with an Anypoint Platform-managed API

MuleSoft

API Manager | Acme Insurance | ? | AO

API Administration (Staging) | Aggregator Quote Creation EAPI (v1) - Client Applications

STAGING

- API Administration
  - Alerts
  - **Client Applications**
  - Policies
  - SLA Tiers
  - Settings

Aggregator Quote Creation EAPI v1

Actions

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

Implementation URL: http://ans-aggregatorquotecreation-eapi.cloudhub.io/v1

Consumer endpoint: http://ans-aggregatorquotecreation-eapi.cloudhub.io/v1

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Search

1 - 1 of 1

Application	Current SLA tier	Requested SLA tier	Status	
Aggregator	Standard	N/A	Approved	Revoke

Owners	AnySurance Owner anysurance+owner@googlegroups.com	**Submitted**	5 days ago
Client ID	552f92bfd0a94500b007c165fde8dbd2	**Approved**	5 days ago
URL	None	**Rejected**	-
Redirect URIs	None	**Revoked**	-

Client ID-based API policies

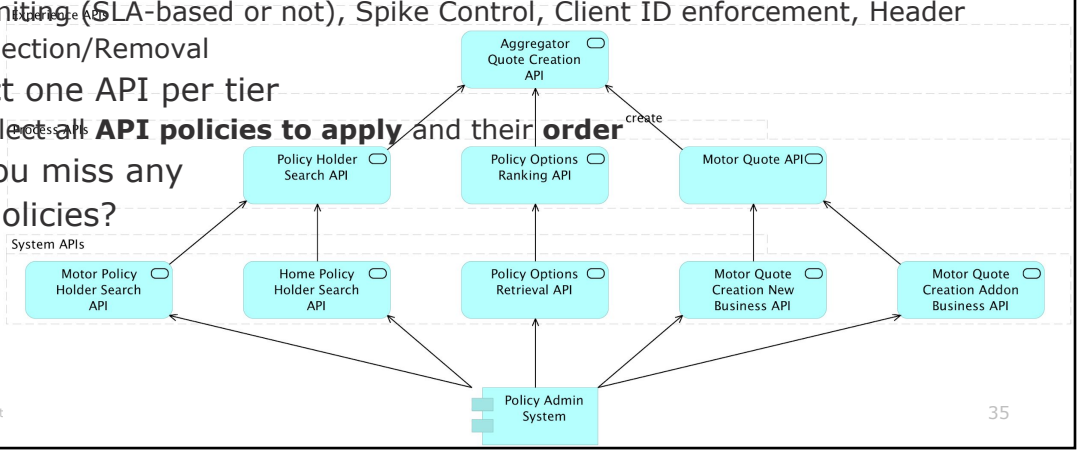
- API policies that require **API clients to identify** themselves:
  - **Client ID enforcement**
  - Rate Limiting - **SLA-based**
    - Retrieve SLA tier by client ID
      - Also enforce presence and validity of **client ID** and secret (optional)
  - **OAuth 2.0** access token enforcement
    - Token implicitly carries client ID
    - Policy **exchanges token for client ID** and passes it to SLA-based API policy
- **Client ID and client secret** passed in API invocations as defined by the API policy
  - **Query parameters**
  - Custom request **headers**
  - Standard **Authorization header** as in HTTP Basic Authentication

Transformation API policies

- To manipulate **HTTP headers** in requests and responses:
  - **Header Injection**
    - Values are **expressions** and hence dynamically evaluated
  - **Header Removal**
- For instance, to propagate transcation IDs as HTTP headers along chains of API invocations

Exercise: Select API policies for all tiers in Acme Insurance's application network

1. Using OOTB API policies
   - CORS, HTTP Basic Auth Simple/LDAP, IP black/whitelist, JSON/XML threat protection, PingFederate/OpenAM/OIDC access token enforcement, Rate Limiting (SLA-based or not), Spike Control, Client ID enforcement, Header Injection/Removal
2. Select one API per tier
   - Select all **API policies to apply** and their **order**
3. Do you miss any API policies?



Choosing appropriate API policies for System APIs

Policy Options Retrieval SAPI v1

Actions

API Status: Active Asset Version: 1.0.0 Type: RAML/OAS

Implementation URL: http://ans-policyoptionsretrieval-sapi.cloudhub.io/v1

Consumer endpoint: http://ans-policyoptionsretrieval-sapi.cloudhub.io/v1

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Apply New Policy

Edit policy order

	Name	Category	Fulfills	Requires	
>	IP whitelist	Security	IP filtered		
>	Rate limiting - SLA based	Quality of service	SLA Rate Limiting, Client ID required		RAML snippet
>	Spike Control	Quality of service	Baseline Rate Limiting		

Choosing appropriate API policies for Process APIs

MuleSoft

Policy Holder Search PAPI v1

Actions

API Status: Active Asset Version: 1.0.3 Type: RAML/OAS

Implementation URL: http://ans-policyholdersearch-papi.cloudhub.io/v1

Consumer endpoint: http://ans-policyholdersearch-papi.cloudhub.io/v1

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Apply New Policy

Edit policy order

	Name	Category	Fulfills	Requires	
>	IP whitelist	Security	IP filtered		
>	Client ID enforcement	Compliance	Client ID required		RAML snippet
>	Spike Control	Quality of service	Baseline Rate Limiting		

Choosing appropriate API policies for Experience APIs

MuleSoft

Aggregator Quote Creation EAPI v1

Actions

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

Implementation URL: http://ans-aggregatorquotecreation-eapi.cloudhub.io/v1

Consumer endpoint: http://ans-aggregatorquotecreation-eapi.cloudhub.io/v1

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Apply New Policy

Edit policy order

	Name	Category	Fulfills	Requires	
>	IP whitelist	Security	IP filtered		
>	XML threat protection	Security	XML threat protected		
>	Rate limiting - SLA based	Quality of service	SLA Rate Limiting, Client ID required		RAML snippet

Choosing appropriate API policies for Experience APIs

MuleSoft

Mobile Policy Holder Summary ... v1

Actions

API Status: Unregistered Asset Version: 1.0.0 Type: RAML/OAS

Implementation URL: http://acmeins-mobilepolicyholdersummary-eapi.cloudhub.io/v1

Consumer endpoint: http://acmeins-mobilepolicyholdersummary-eapi.cloudhub.io/v1

View API in Exchange >

View configuration details >

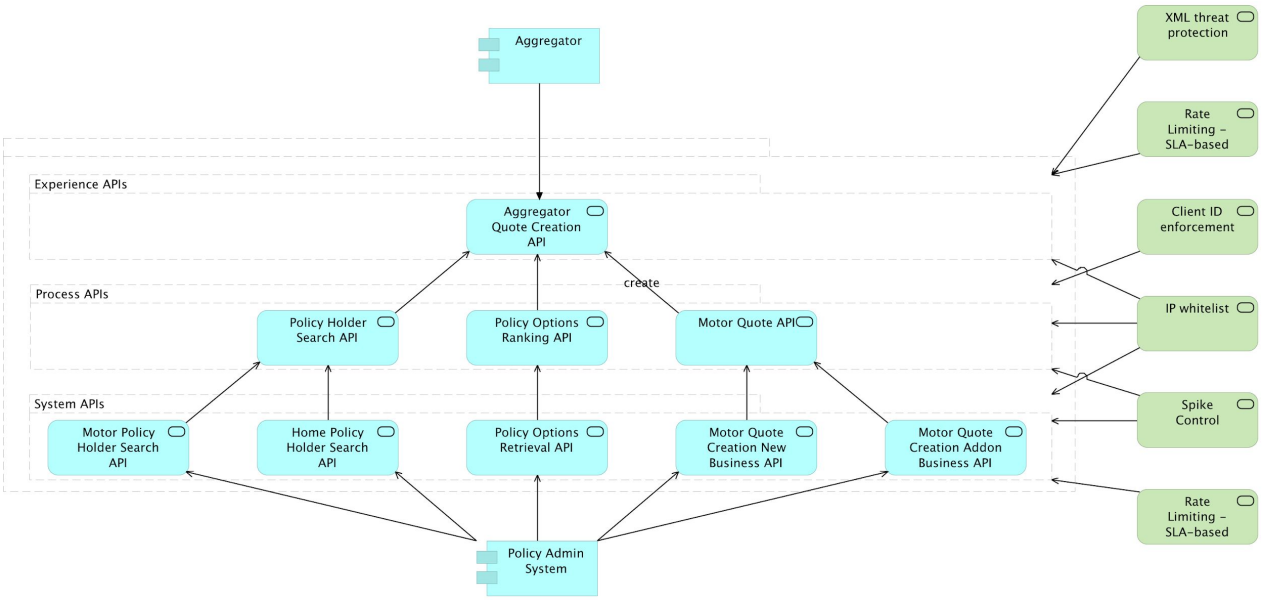
Apply New Policy

Edit policy order

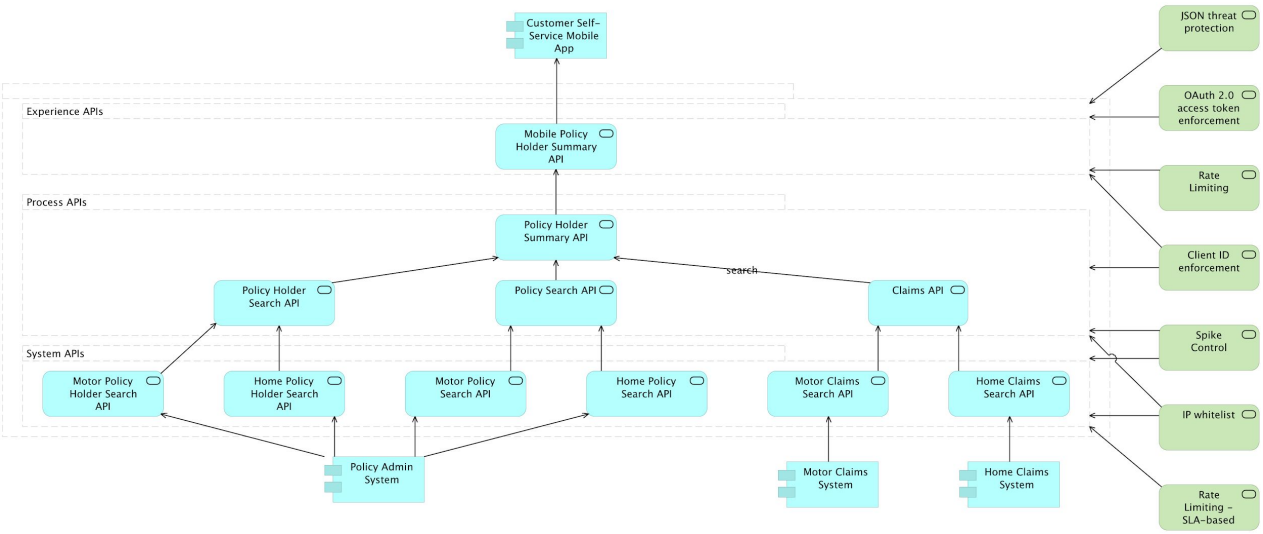
	Name	Category	Fulfills	Requires	
>	JSON threat protection	Security	JSON threat protected		
>	OAuth 2.0 access token enforcement using external provider	Security	OAuth 2.0 protected		RAML snippet
>	Client ID enforcement	Compliance	Client ID required		RAML snippet
>	Rate limiting	Quality of service	Baseline Rate Limiting		

API policies for "Create quote for aggregators"

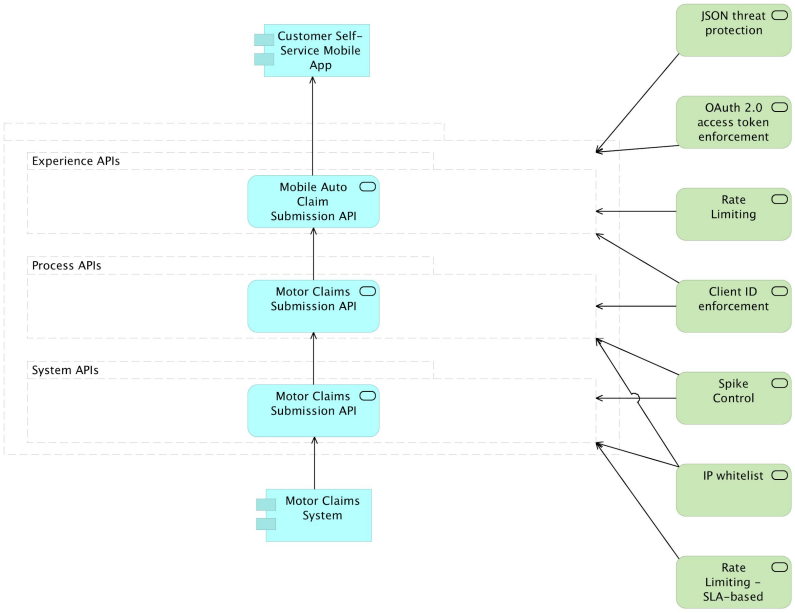
MuleSoft



API policies for "Retrieve policy holder summary"



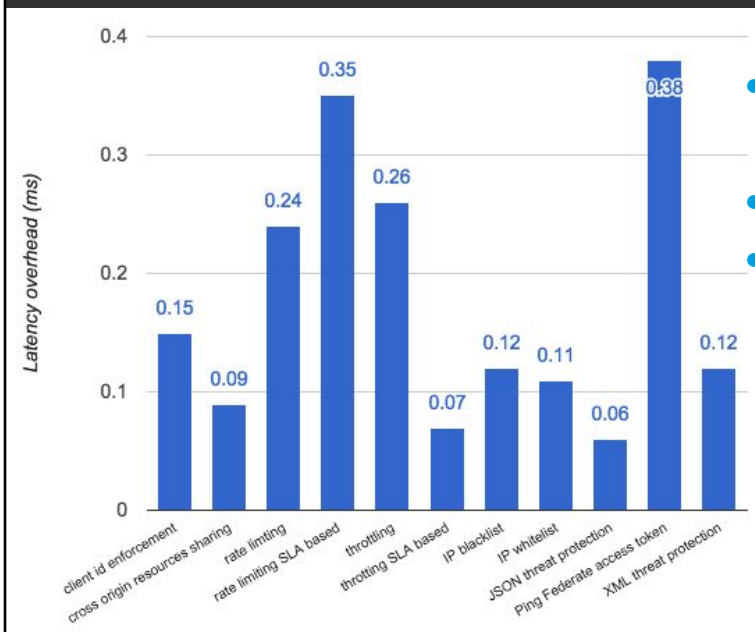
API policies for "Submit auto claim"



Reflecting the application of API policies in the RAML definition of an API

- Many API policies **change HTTP request/response**:
  - Require certain HTTP request headers: **Authorization**
  - Require certain query parameters: **client_id**
  - Add HTTP response headers: **X-RateLimit-Limit**
- **Change contract** between API client and API implementation
- Must be reflected in **RAML definition** of API
  - RAML has specific support for **securitySchemes** such as OAuth 2.0
  - In other cases define **RAML traits**
- **C4E** owns definition of reusable **RAML fragments**
  - Publish to **Exchange** to encourage consumption and reuse.

Latency overhead of applying API policies



- Increase in HTTP request-response latency
- through API policies
- enforced embedded in API implementation

Summary

Summary

MuleSoft

- **NFRs** for products are constraints on throughput, response time, security and reliability
- **API Manager and API policies** control invocations of APIs and impose non-functional constraints
- Compliance, Security, QoS, Transformation
- API policies **enforced**
  - Directly in an **API implementation** that is a Mule app
  - In an **API proxy**
- **Client ID**-based API policies require registered API clients
  - Must pass client ID/secret with every API invocation
- **C4E** defines guidelines for API policies and publishes matching reusable RAML fragments to Exchange