1

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

- 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, an API proxy, or Anypoint Service Mesh
- 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**
- Describe how **Anypoint Security** enables **de/tokenization** and additional **Edge policies** in Anypoint Runtime Fabric deployments

2

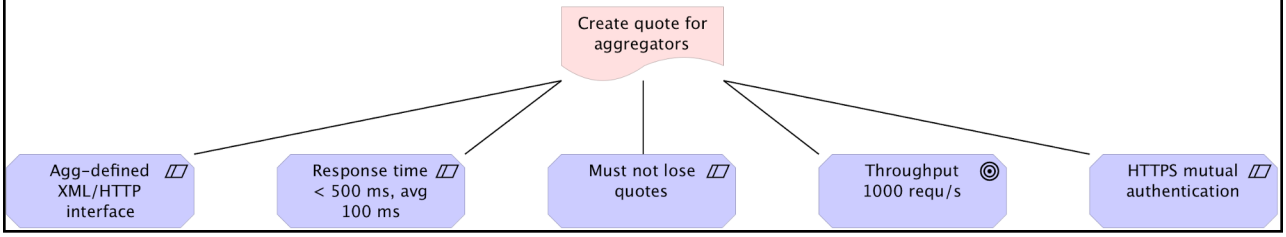
Section 1
Addressing the NFRs of the "Aggregator Integration" product

3

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**



4

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

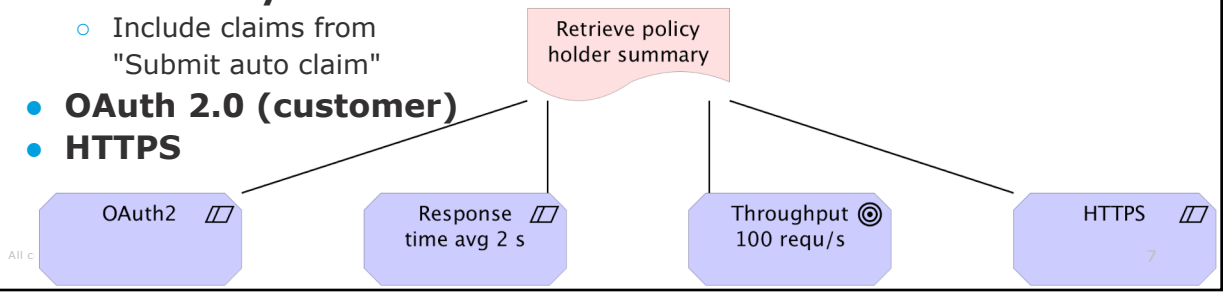
5

Section 2
Addressing the NFRs of the "Customer Self-Service App" product

6

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**



7

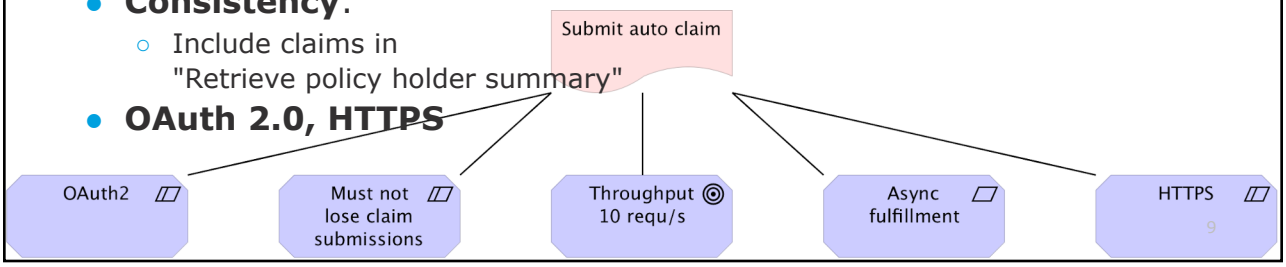
Meeting the NFRs for "Retrieve policy holder summary" using Anypoint Platform

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

8

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**



9

9

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

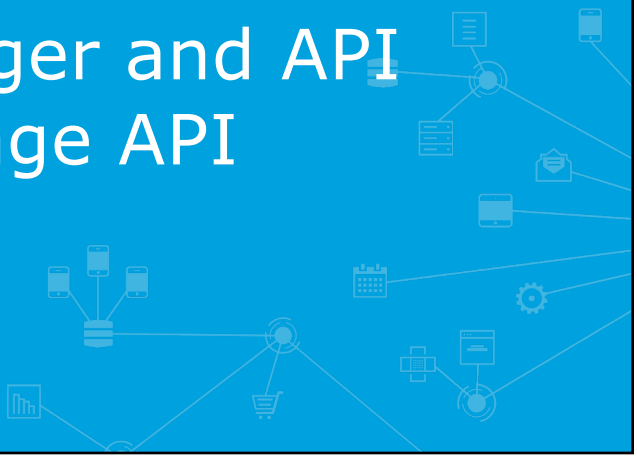
New NFRs for this feature:

- **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**

10

10

Section 3
Using API Manager and API policies to manage API invocations



11

Reviewing types of APIs

MuleSoft

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

12

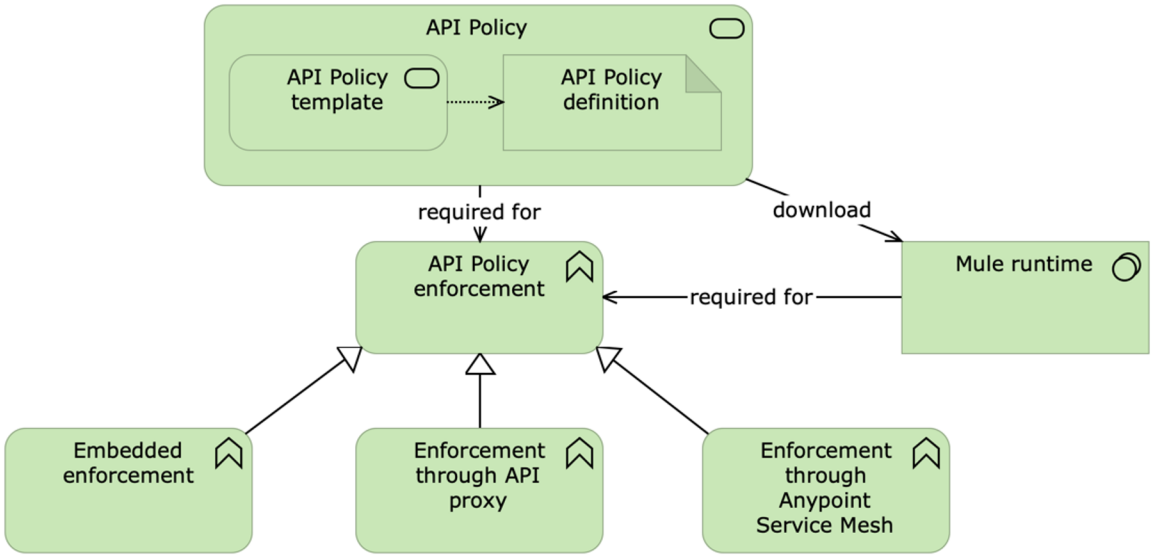
API management on Anypoint Platform

- 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 and OAS-defined APIs**
  - Allow definition of **resource-level** API policies
  - In addition to the **endpoint-level** API policies available for all APIs

Defining API policy

- 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



15

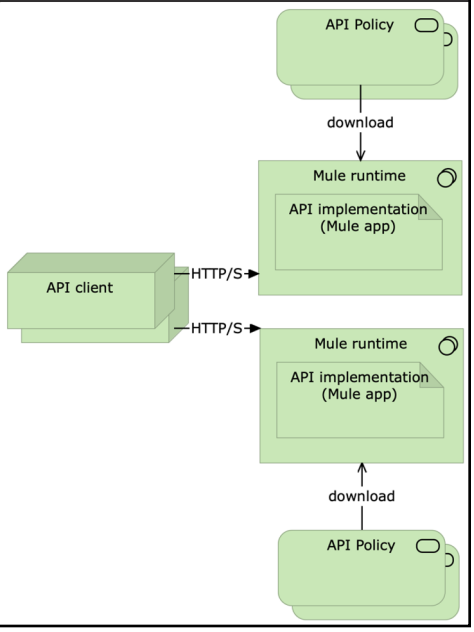
Enforcement of API policies

- 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
  - **Anypoint Service Mesh** for Kubernetes-deployed non-Mule API implementations
- API policies **downloaded at runtime** from API Manager

16

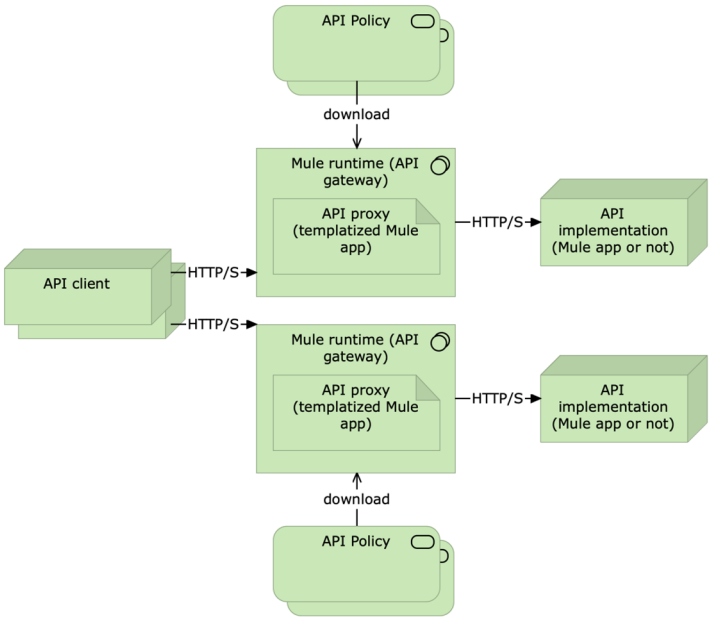
Providing API management for Mule apps

- Always executes in a **Mule runtime**
- Use API **policy enforcement function** of this Mule runtime
  - **Embed** API policy enforcement



17

Providing API management via API proxies



18

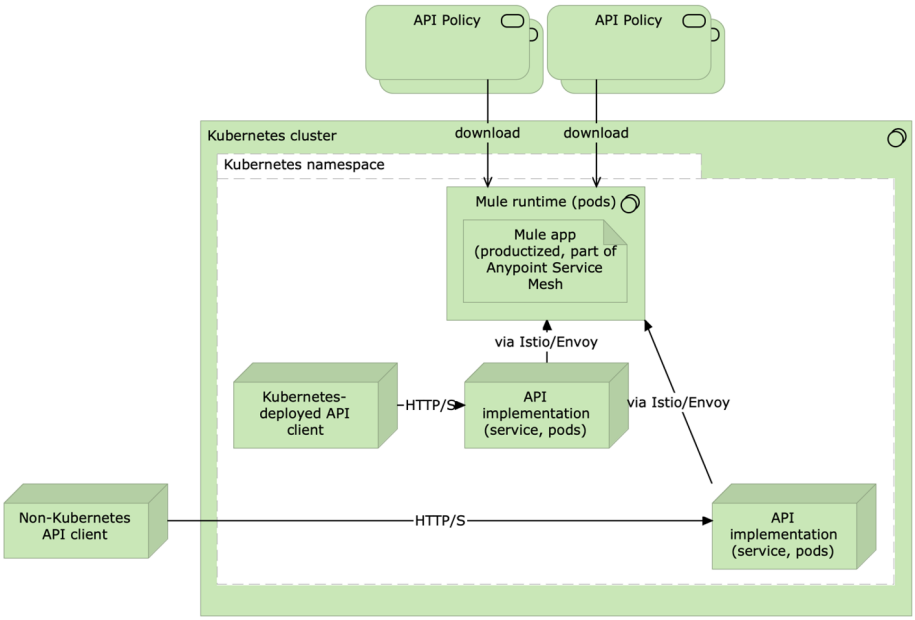
18

Providing API management via API proxies

- Enable policy enforcement for **any API implementation**
  - Must use if **not Mule app** and **not Kubernetes**-deployed
- API proxy is **templated Mule app**
  - **Auto-generated** by API Manager
- Deployed to Mule runtime: **API Gateway**
  - Technical a "normal" Mule runtime
  - On iPaaS (CloudHub): **auto-provision** API Gateway with API proxy
- Exactly **one API implementation** per API proxy
- **API clients** must send API invocations to proxy
- API proxy sends **separate API invocation** to API implementation
- Interface API client->API proxy and API proxy->implementation is **HTTP-based API**
- For **coarsed-grained APIs**: add separate **node**

19

Providing API management for external API implementations deployed to a Kubernetes cluster



20

Providing API management for external API implementations deployed to a Kubernetes cluster

- **Anypoint Service Mesh** for non-Mule app API implementations in Kubernetes (k8s) cluster
  - Typically **fine-grained**: would need too many API proxies
- Install into **customer-hosted k8s** cluster
  - Supported on GKE, EKS, AKS, Red Hat OpenShift
  - Supported versions - k8s: 1.12 through 1.20 or Red Hat OpenShift: 4.x
- Builds on **Istio** (versions 1.7.x - 1.10.x) which uses **Envoy**
- Includes k8s-native managed **Mule app and Mule runtimes** for policy enforcement
  - Replicated pods in k8s namespace
  - Enforces policies **for all API implementations** in namespace
- **API clients** send API invocations to API **implementations**
  - Istio/Envoy intercept and route to Mule runtime/Mule application for policy enforcement

Providing API management for external API implementations deployed to a Kubernetes cluster

Applicable policies for Anypoint Service Mesh:

- **Security**: Basic Authentication: Simple, Basic Authentication: LDAP, JWT Validation, OpenID Connect OAuth 2.0 Token Enforcement
- **QoS**: Rate Limiting, Rate Limiting SLA-Based
- **Compliance**: Client ID Enforcement

Exercise: Pros and cons of policy enforcement sites

Compare the characteristics of the 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**, in an **API proxy**, or through **Anypoint Service Mesh**, respectively

23

Exercise: Pros and cons of policy enforcement sites

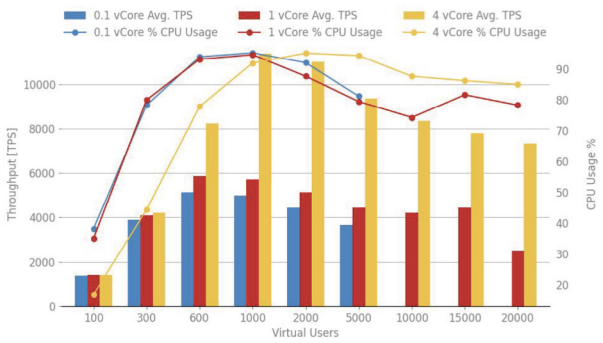
- API implementations are **not Mule apps**
- Deployed to **k8s** cluster or not
- **Resources** must be minized
- **Deployment and CI/CD** must be as simple as possible
- API policies with special **resource requirements** are applied
  - Caching API policy
  - Security API policy requiring HSM
- API policies require **special network configuration**
- **Security sensitive (Experience) APIs**
  - Deployment to **DMZ**
  - **Shield API implementations** from attacks

24

HTTP Proxy Performance

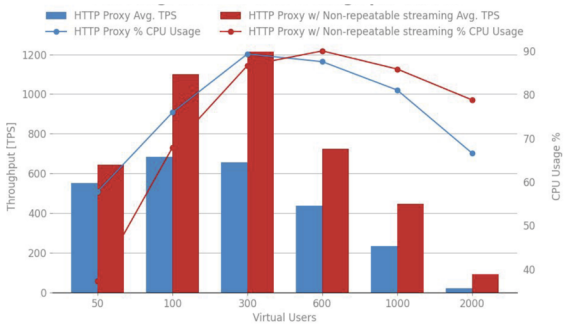
MuleSoft

MULE 4.3.0 - HTTP PROXY 1KB AVG. TPS AND % CPU USAGE PER VCORES



HTTP proxy GET with 1 KB payload for 0.1, 1, and 4 vCores - average throughput and % CPU usage

MULE 4.3.0 — HTTP PROXY VS. HTTP PROXY W/ NON-REPEATABLE STREAMING

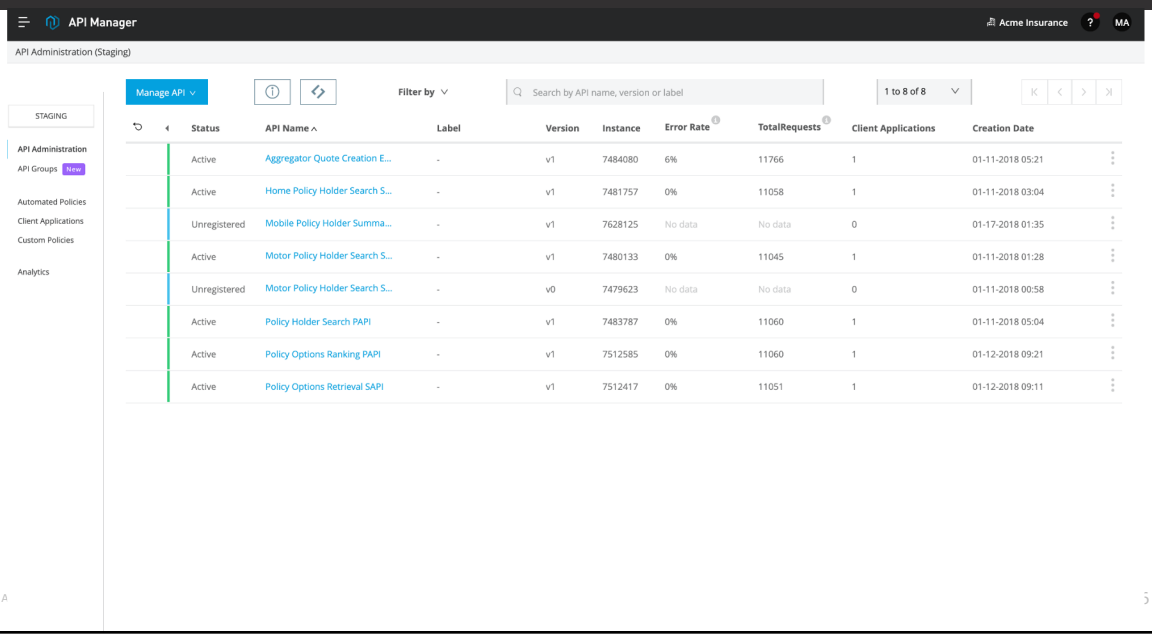


HTTP proxy and HTTP proxy no-repeatable streaming with 1 MB payload performance for 4.3.0 version with 4 vCores average throughput and % CPU usage

25

Managing APIs with API Manager

MuleSoft



26

Managing APIs with API Manager

- 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
- Configuration of **automated policies** for all API instances in an **environment**
- Configuration of **API Groups** that bundle API instances and streamline some API management tasks common to the group

27

Managing APIs with API Manager

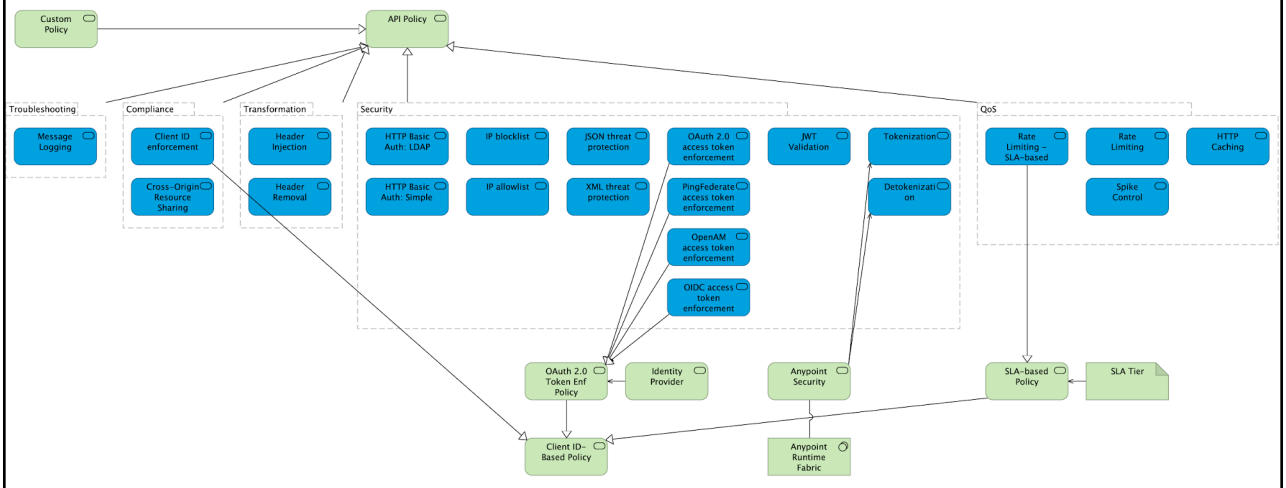
- Contacted from site of API policy enforcement to **download all API policies** that must be enforced
- Definition of **alerts** based on API invocations
- Admin of **API clients** ("Client Applications")
  - API consumers use Exchange to request access
- API consumers use **Exchange to request access** to an API
- Access to Anypoint **Analytics**
- Display total requests and error rates for an API (for the previous 24 hours)

28

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

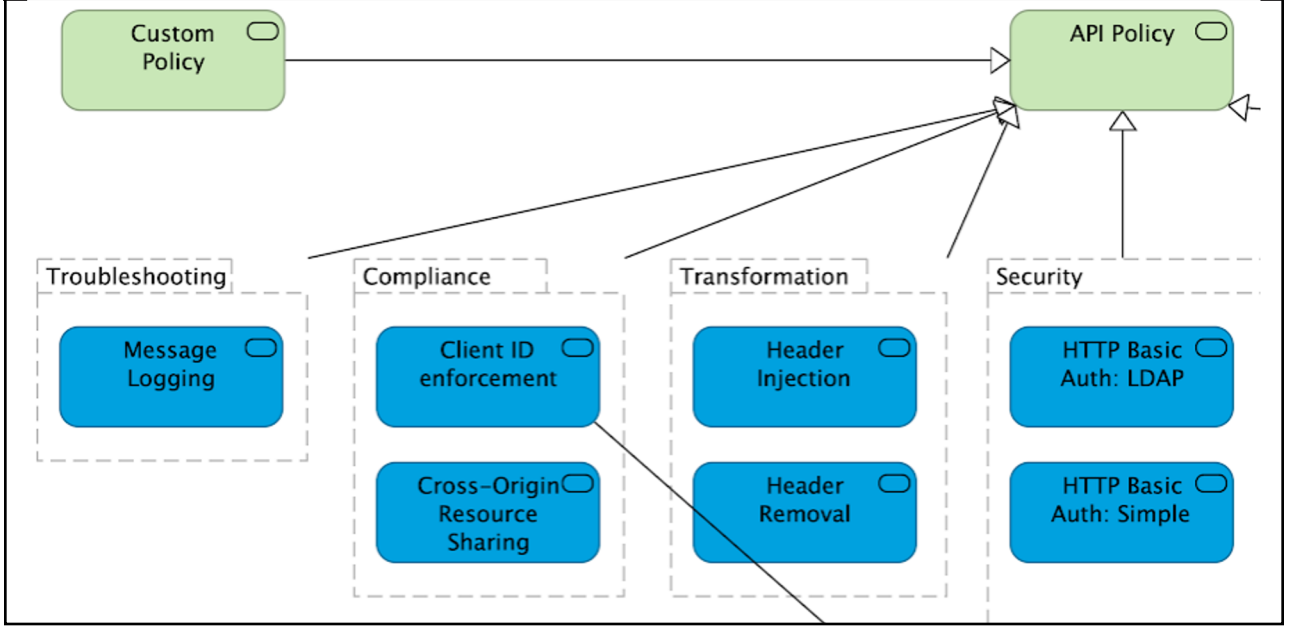
- By default API policies are applied to **entire API endpoint**
  - Represented as API instance in API Manager
- APIs defined with an **API spec** (RAML or OpenAPI definition) can apply API policies also to selected combinations of **API resources and HTTP methods**

API policies



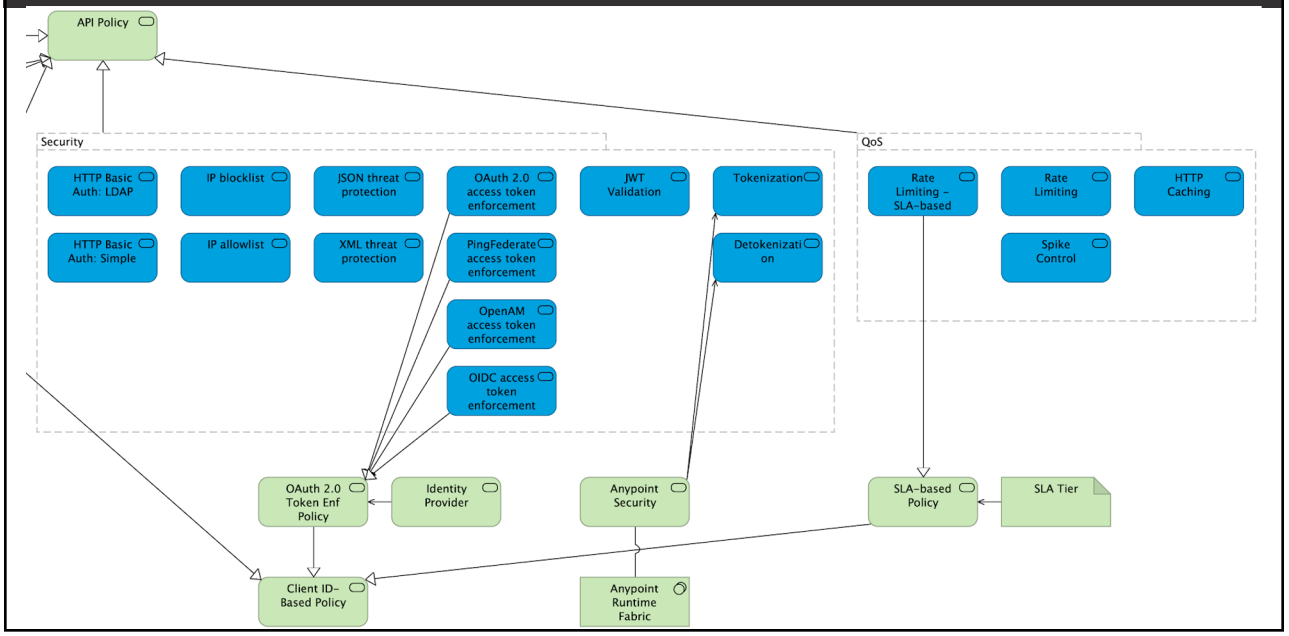
The API policies performing IP blocklisting and allowlisting are officially still called "IP blacklist" and "IP whitelist"

API policies



31

API policies



32

API policies as Aspect-Oriented Programming

MuleSoft

- 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**

33

Understanding custom API policies

MuleSoft

- **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

34

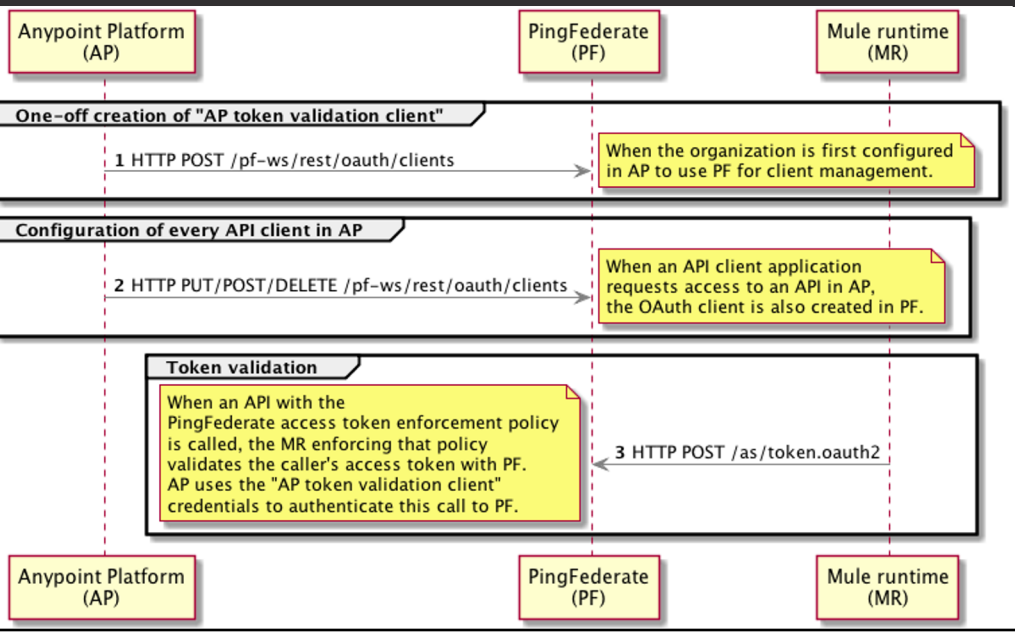
Compliance-related API policies

- **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)
    - Optional "OAuth 2.0 access token enforcement using Mule OAuth provider" requires access to Mule OAuth 2.0 provider configured only in the policy itself
  - **Basic Authentication: LDAP/Simple**
    - Incorporate access to Identity Provider
- **IP-based** access control
  - **blocklisting, allowlisting**
- Payload **threat protection**
  - Guard against attacks sending over-sized HTTP request bodies
  - **Limit size of XML or JSON bodies**
- **De/Tokenization**
  - Only with Anypoint Security on Runtime Fabric

Interactions Anypoint Platform - IdP



37

37

Interactions Anypoint Platform - IdP

- **IdP** issues and validates **OAuth 2.0 access tokens**
  - OAuth 2.0 token enforcement API **policies delegate** to IdP
- Anypoint Platform triggers **creation of API client application** in IdP
  - Client ID and secret stored and **in-sync** in Anypoint Platform and IdP
- Client secret **rotation in IdP**:
  - **Anypoint Platform not updated**
  - **OAuth** 2.0 interactions must use **new secret**
  - **Non-OAuth** 2.0 interactions (Client ID enforcement) must use **old secret**
- Client secret **rotation in Anypoint Platform** with **OIDC DCR** IdP configured for Client Management:
  - Disabled by **default**
  - Can be enabled
    - **Anypoint Platform** supports extension to OIDC DCR to push updates to IdP
    - Requires **custom integration** app to intercept calls and act as adapter to IdP

38

38

JSON Web Tokens (JWTs)

MuleSoft

- Compact **claims** representation format for
  - HTTP **Authorization** headers
  - URI query parameters
- **Claim**:
  - Piece of information asserted about a subject
  - Represented as a **name/value pair** (String/JSON pair)
- **Claims Set**:
  - **JSON object** containing the claims in the JWT
  - May be **digitally signed** or **integrity protected**
    - using JSON Web Signature (**JWS**)
  - May be **encrypted**
    - using JSON Web Encryption (**JWE**)
- **JOSE header** describe **cryptographic operations** applied to the Claims Set
- **Unsecured JWTs**: created without a signature or encryption

Source: IETF RFC 7519

JWT Example 1: JWS using HMAC

MuleSoft

- JOSE Header
  - JWT that is JWS and MACed using the HMAC SHA-256 algorithm:
  - { **"typ"**: "JWT", **"alg"**: "HS256" }
- JWT Claims Set
  - { **"iss"**: "joe", **"exp"**: 1300819380, **"http://org.com/is_root"**: true }
- Complete JWT
  - Above JSON objects are normalized, base64-encoded, MACed,
  - MAC is normalized and base64-encoded
  - All 3 parts concatenated with .
  - **eyJ0<snip>NiJ9.eyJp<snip>VlfQ.dBjf<snip>EjXk**

Source: IETF RFC 7519

JWT Example 2: Unsecured JWT

MuleSoft

- JOSE Header
  - JWT that is JWS and MACed using the HMAC SHA-256 algorithm:
  - { **"alg"**: "none" }
- JWT Claims Set
  - { **"iss"**: "joe", **"exp"**: 1300819380, **"http://org.com/is_root"**: true }
- Complete JWT
  - Above JSON objects are normalized, base64-encoded
  - Both parts concatenated with . plus trailing . for missing signature
  - **eyJh<snip>lIn0.eyJp<snip>VlfQ**.

Source: IETF RFC 7519

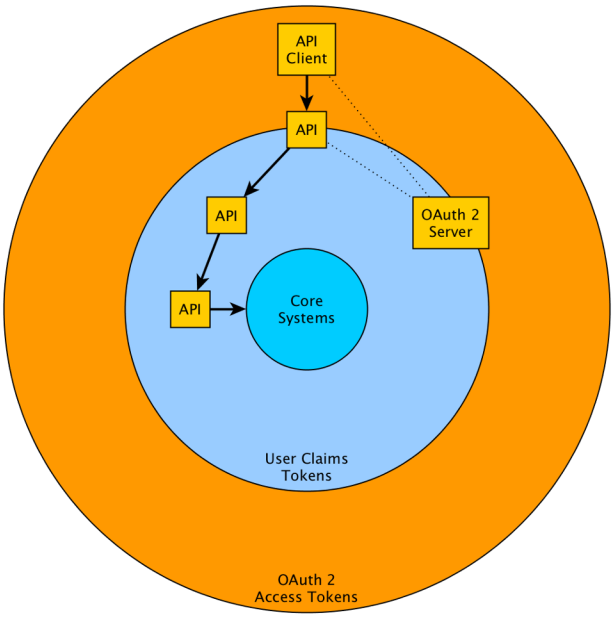
JWT Claims

MuleSoft

- **Registered** Claim Names
  - Registered in the IANA "JSON Web Token Claims" registry:
  - "iss" (Issuer) "sub" (Subject)
  - "aud" (Audience) "exp" (Expiration Time)
  - "nbf" (Not Before) "iat" (Issued At)
  - "jti" (JWT ID)
- **Public** Claim Names
  - Either registered as above
  - or Collision-Resistant Name (**namespaced**)
- **Private** Claim Names
  - **Agreed** between producer and consumer of a JWT

Source: IETF RFC 7519

Propagating user claims in JWTs within an application network



43

JWT signing and signature validation

- According to **JWS**
- Either: Message Authentication Code (**MAC**)
  - **HMAC** algorithm
  - **Shared secret** for signing and signature validation
  - **Integrity checks** JWT Claims Set
- Or: **digital signatures**
  - **RSA** or **ECDSA**
  - **Public/private key pair**
    - Private key for signing
    - Public key for signature validation
  - **Integrity checks** JWT Claims Set
  - **Identifies originator** (= is in possession of private key)

Source: IETF RFCs 7515 and 7518

44

JWT signing and signature validation

- JWT **Claims Set is readable** by third parties
  - Not a form of encryption - see JWE
- **Signature validation** by the JWT recipient
  - Requires **shared secret or public key**
    - matching shared secret or private key used for signing the JWT
  - Typically retrieved from a JSON Web Key Set (**JWKS**) **server** at a well-known URL

Source: IETF RFCs 7515 and 7518

JWT validation API policy

- Validates JWT in **incoming HTTP request**
  - By default: from HTTP Authorization header as **Bearer** token
- Validates and propagates the JWT's **Claims Set**
- **Signature validation**
  - Rejects HTTP request if signature not valid
  - No support for JWE (encrypted) JWTs
  - Supports **JWS** (signed) JWTs and validates the signature
    - Only HMAC and RSA
  - Shared secret or public key
    - Either supplied in **policy definition**
    - or retrieved from **JWKS server**
  - Supports **unsecured** (unsigned, unencrypted) JWTs
  - Can also ignore signature even if present

JWT validation API policy

- **Claims Set validation**
  - Rejects HTTP request if the JWT Claims Set does not match config
  - Supports all types of JWT Claims (registered, public, private)
- **Claims Set propagation**
  - Claims Set passed **to Mule app** that enforces JWT validation API policy
  - Local, **in-process** propagation in variable

47

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

48

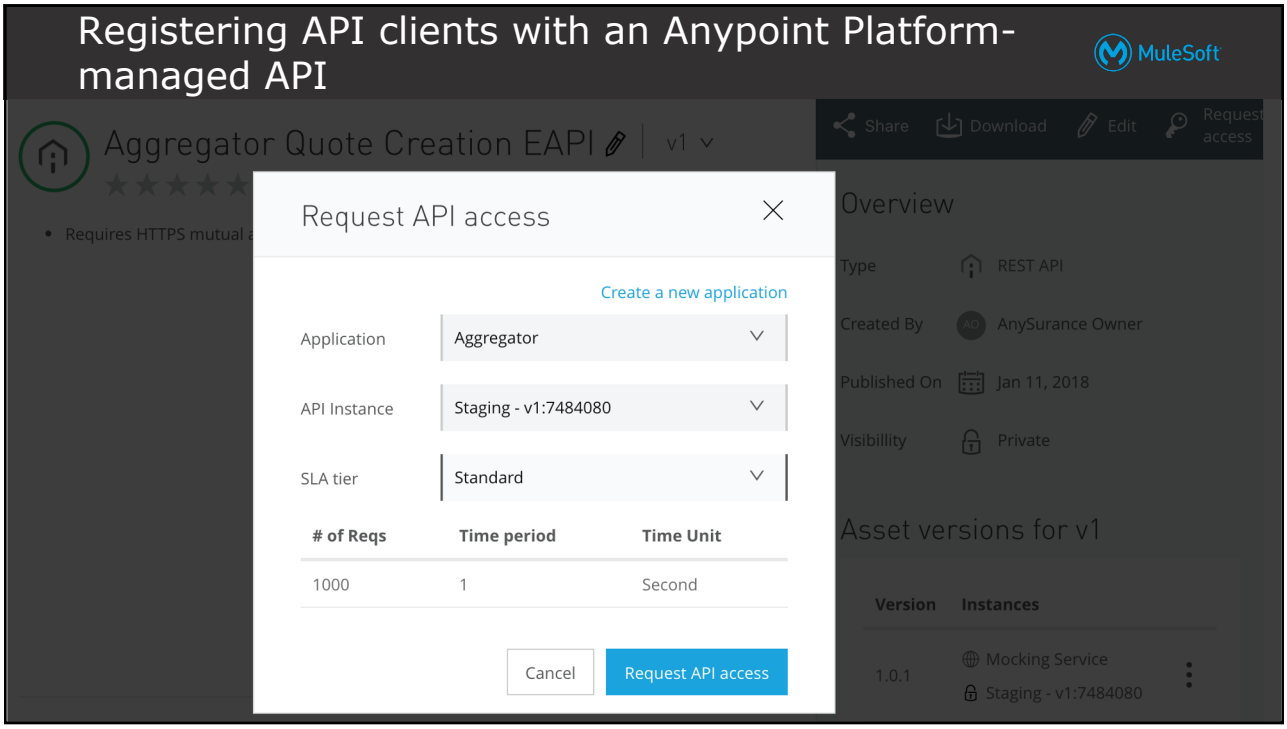
Anypoint Platform SLA tiers for APIs

- SLA tiers
  - Enable **different API clients** to receive **different QoS**
  - Define one or more **throughput limits**
    - Per **API client** and **API instance**
  - Can also be assigned to an **API Group instance** and thereby to all API instances in that group
- 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

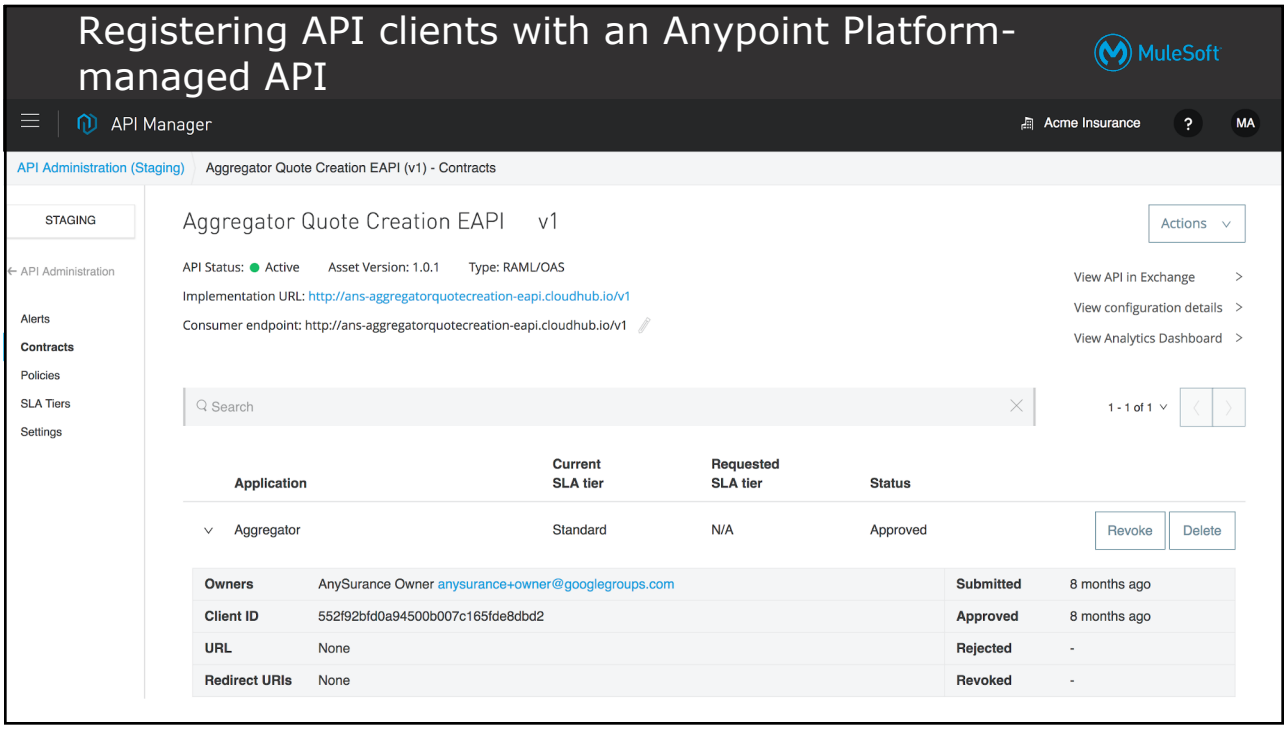
- API clients must **register to invoke** API instance with Client ID-based API Policies
  - Called "application" or "client application"
  - API-API client relationship: "contract" in API Manager
- Request access **through Exchange** entry for that API
  - Directly from Exchange or via Public (Developer) 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
- Similarly for **API Groups**:
  - Published to Exchange from API Manager
  - Request access to API Group to gain access to **all API instances in group**
  - Establishes group-level contract instead of API instance-level contract

Registering API clients with an Anypoint Platform-managed API



51

Registering API clients with an Anypoint Platform-managed API



52

Client ID-based API policies

MuleSoft

- 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

HTTP Caching API policy

MuleSoft

- **Server-side** caching
- Caches **entire HTTP responses**
  - status code, headers, body
  - Size limit of 1MB
- Only if
  - **HTTP request expression** is true:
    - Default: HTTP method is GET or HEAD
  - **HTTP response expression** is true
    - Default: status code is in restricted set of 2xx, 3xx, 4xx or 5xx
- May honor many **caching directives** (HTTP headers)
- **Cache invalidation** via HTTP request header

HTTP Caching API policy - caching parameters

- Key
  - Default: request path
- Number of entries
- Time-to-live
- Distributed
- Persistent

55

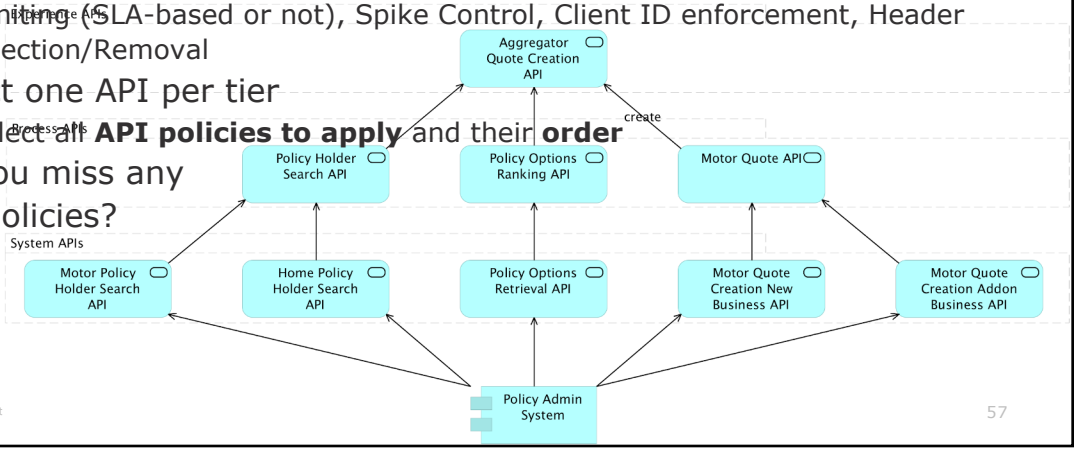
Transformation API policies

- 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

56

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

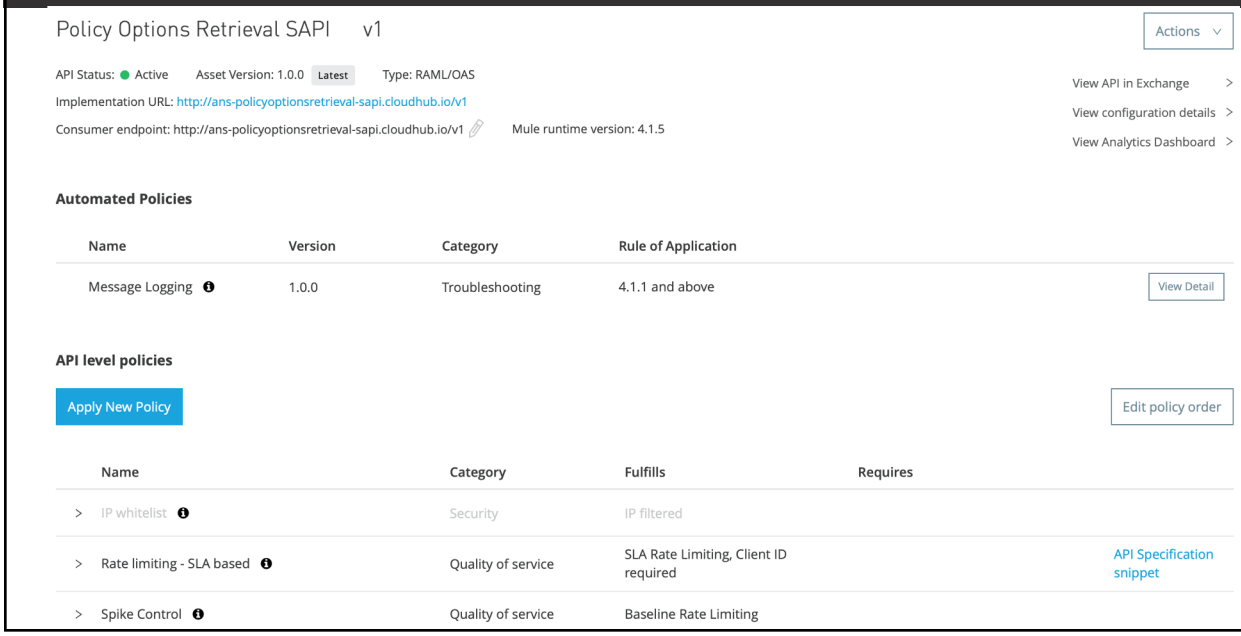
1. Using OOTB API policies
   - CORS, HTTP Basic Auth Simple/LDAP, IP block/allowlist, JSON/XML threat protection, PingFederate/OpenAM/OIDC access token enforcement, Rate Limiting (SLA-based or not), Spike Control, Client ID enforcement, Header Injection/Removal
2. Select one API per tier
   - Select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



57

57

Choosing appropriate API policies for System APIs



58

Choosing appropriate API policies for Process APIs

MuleSoft

Policy Holder Search PAPI v1

Actions

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

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

Consumer endpoint: http://ans-policyholdersearch-papi.cloudhub.io/v1 Mule runtime version: 4.1.5

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Automated Policies

Name	Version	Category	Rule of Application	
Message Logging	1.0.0	Troubleshooting	4.1.1 and above	View Detail

API level policies

Apply New Policy

Edit policy order

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

59

Choosing appropriate API policies for Experience APIs

MuleSoft

Aggregator Quote Creation EAPI v1

Actions

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

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

Consumer endpoint: http://ans-aggregatorquotecreation-eapi.cloudhub.io/v1 Mule runtime version: 4.1.5

View API in Exchange >

View configuration details >

View Analytics Dashboard >

Automated Policies

Name	Version	Category	Rule of Application	
Message Logging	1.0.0	Troubleshooting	4.1.1 and above	View Detail

API level policies

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		API Specification snippet

60

Choosing appropriate API policies for Experience APIs

MuleSoft

Mobile Policy Holder Summary EAPI v1

Actions

API Status: Unregistered Asset Version: 1.0.0 Latest 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 >

Automated Policies

There are Automated Policies configured for this environment. Once an API is deployed, depending on its runtime version, Automated Policies may override API level policies. View Automated Policies

API level policies

Apply New Policy

Edit policy order

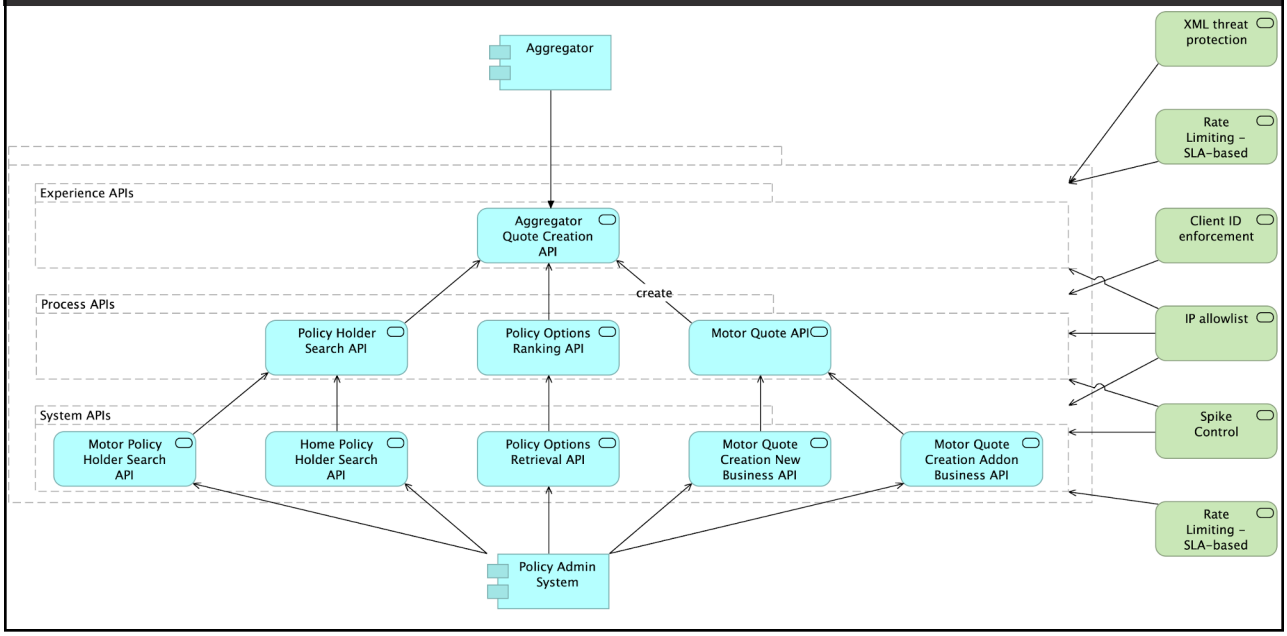
Name	Category	Fulfills	Requires	
> JSON threat protection	Security	JSON threat protected		
> OAuth 2.0 access token enforcement using external pr...	Security	OAuth 2.0 protected		API Specification snippet
> Rate limiting	Quality of service	Baseline Rate Limiting		

61

61

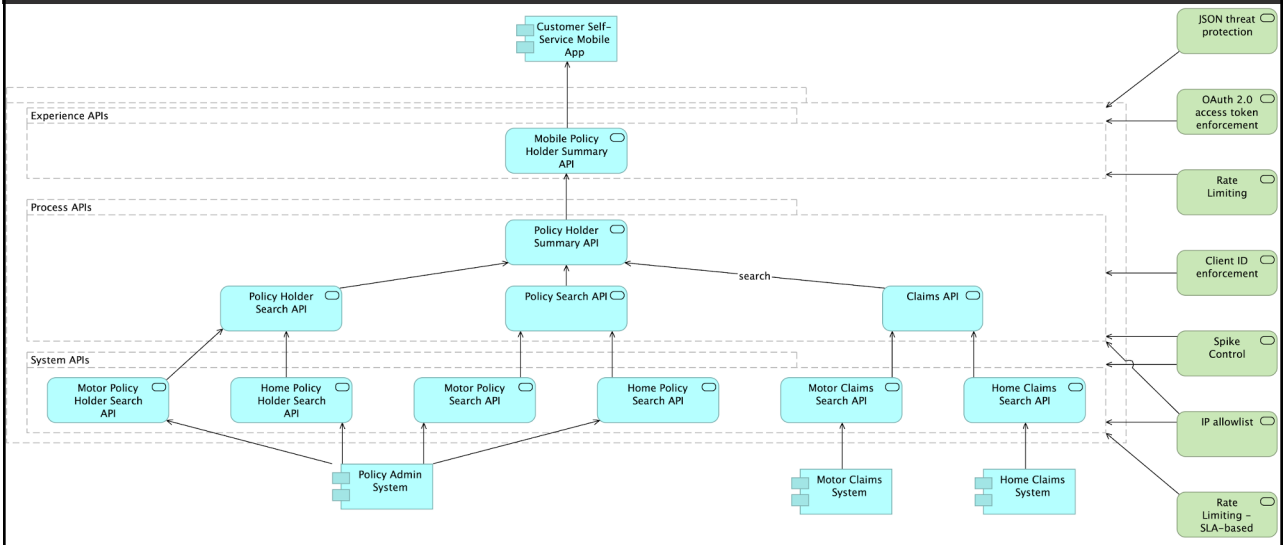
API policies for "Create quote for aggregators"

MuleSoft



62

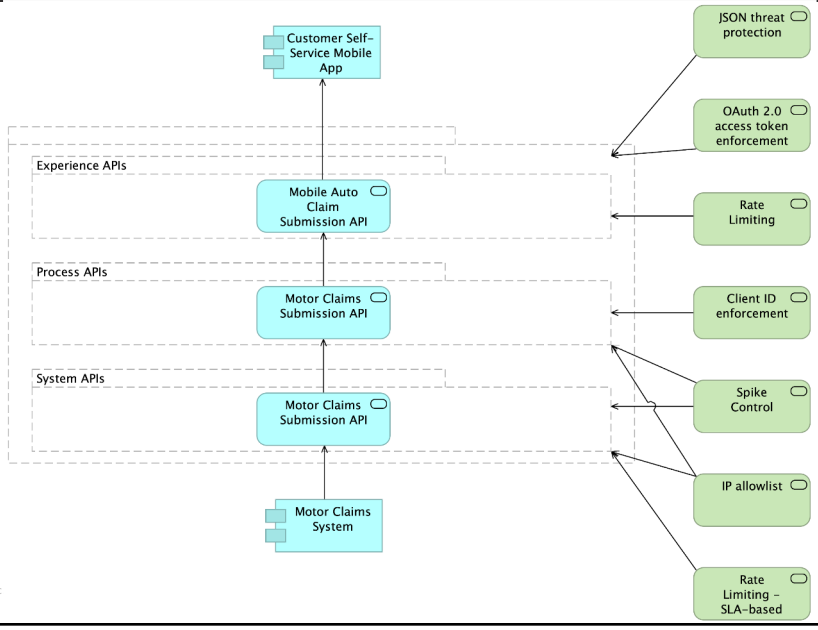
API policies for "Retrieve policy holder summary"



63

63

API policies for "Submit auto claim"



64

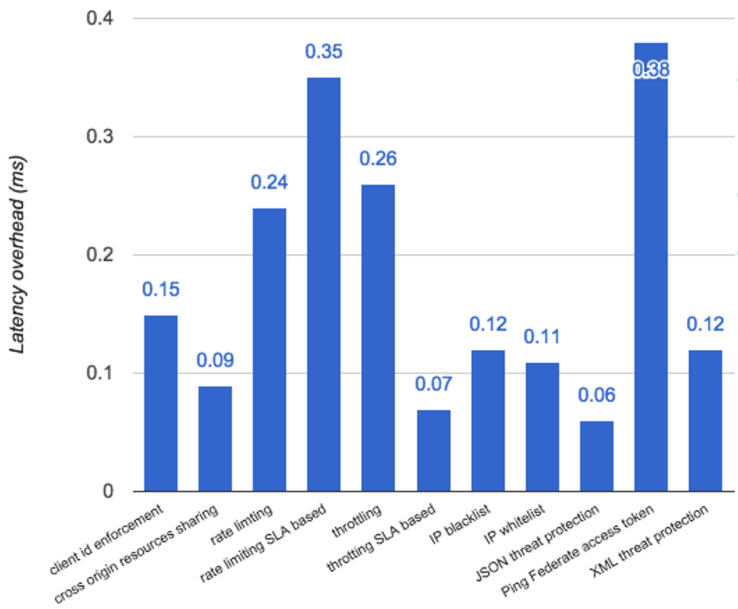
64

Reflecting the application of API policies in the API spec 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 **API spec** of the 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.

65

Latency overhead of applying API policies



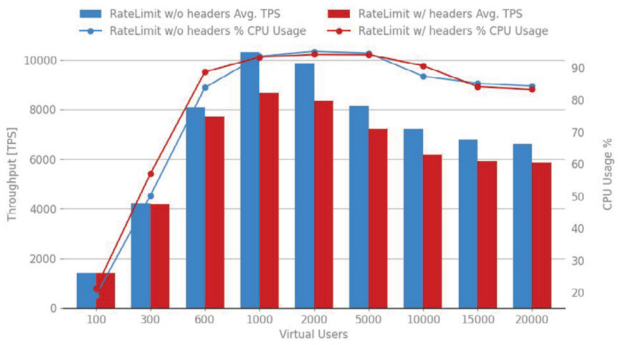
- Increase in HTTP request-response latency
- through API policies
- enforced embedded in API implementation

66

Latency overhead of applying API policies

MuleSoft

MULE 4.3.0 — RATELIMIT W/ HEADERS VS RATELIMIT W/O HEADERS



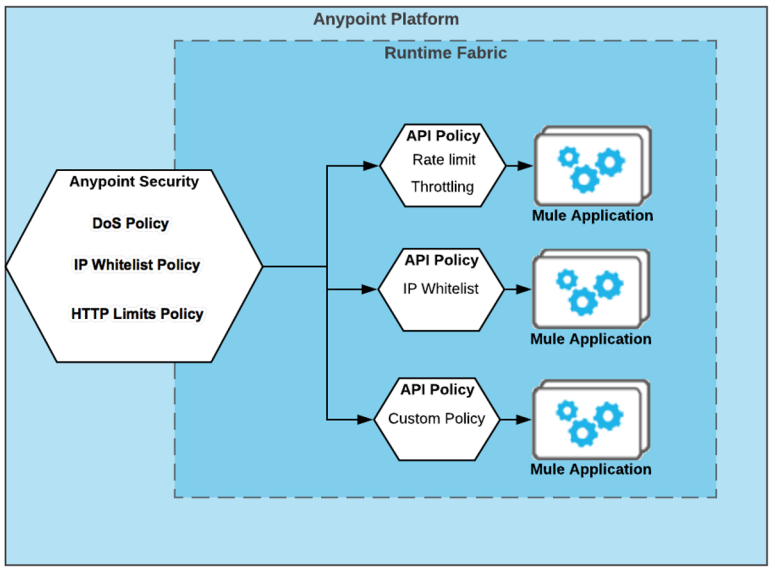
- Comparison of Rate Limit policy when the headers are enabled vs disabled.

67

Section 4
Using Anypoint Security and Edge policies in addition to API Manager and API policies

68

Anypoint Security



69

Anypoint Security

- To implement **perimeter defence** in customer-hosted deployments of Anypoint Platform runtime plane (only) on **Anypoint Runtime Fabric** (RTF)
- Serves as Kubernetes **Ingress** and enforces **Edge policies**
  - Ingress provides **load-balancing** and **SSL termination** for **external API clients** of API implementations deployed to Kubernetes cluster
- Includes **Secrets Manager** for storing **certificates** needed for enabling TLS traffic, optionally with **mututal auth**, to the Ingress
- Anypoint Security and Edge policies **independent** of API Manager and API policies
  - enforce core set of similar NFRs
- Edge policies enforced **once for all APIs** exposed from RTF
  - API policies enforced separately for each API implementation
- **API policies** typically enforced as **2nd line of defence**

70

Edge policies supported by Anypoint Security

MuleSoft

At least the following Edge policies:

- Content Attack Prevention (**CAP**) by **limiting HTTP request** properties
  - HTTP methods, header size, body size, URL path length, …
- **Allowlisting** of API client IP addresses
  - similar to IP-based access control in API policies
- Web Application Firewall (**WAF**) security policy enforcing the **OWASP Core Rule Set**:
  - SQL injection, cross-site scripting, local file inclusion, HTTPoxy, Shellshock, session fixation, …

71

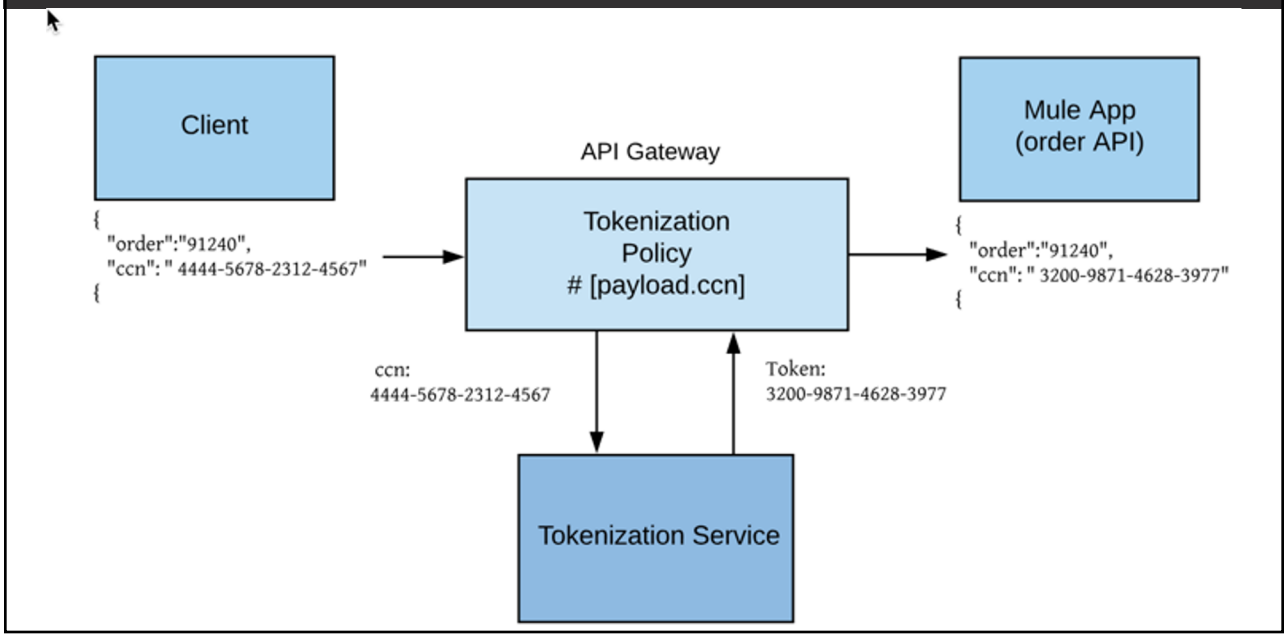
Edge policies supported by Anypoint Security

MuleSoft

- **DoS attack prevention** through monitoring of API clients' HTTP requests
  - Rate limiting or blocking client IP address upon detection of DoS attack
  - Other Edge policies and API policies can escalate policy violations to DoS policy to contribute to detection of DoS attack
  - Defined by rules in DoS policy

72

Tokenizing sensitive information in API invocations



73

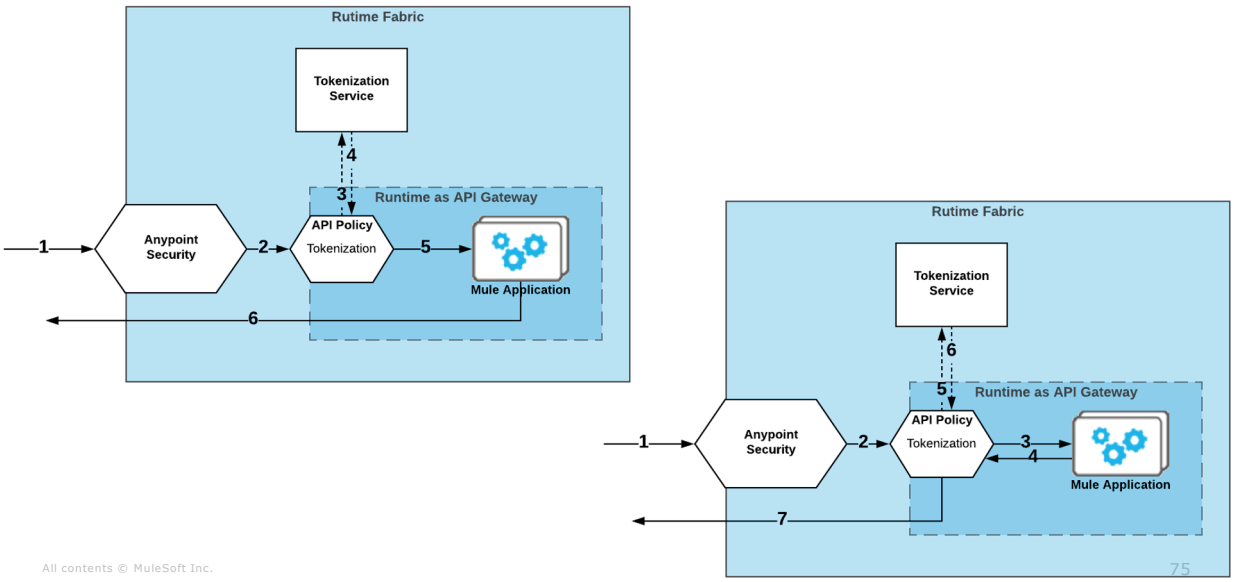
Tokenizing sensitive information in API invocations

A security feature of **Anypoint Security** and is enabled by a **Tokenization Service** and corresponding **API policies** (not Edge policies):

- Tokenization **replaces sensitive information** (credit card number, SSN, account number, any regex, …) with a **reversable token**
- **Detokenization** restores the original sensitive information
- Typically **format-preserving** such that downstream systems' validation rules are not violated
  - **1234-5678-9012-3456 -> 9264-1956-3442-3456** (tokenization of credit card number, configured to preserve format and keep last 4 digits)

74

Tokenizing sensitive information in API invocations



75

Tokenizing sensitive information in API invocations

- Applied to **HTTP requests** or **responses** sent to/from individual APIs by configuring the **Tokenization and Detokenization API policies** via Anypoint API Manager on the corresponding API instances
- Tokenization **Service** is deployed to RTF and these API policies delegate to it the actual de/tokenization
- Anypoint Security implements **vaultless tokenization**
  - There is no database that stores the original, clear-text values
  - Tokens are not amenable to brute-force attempts of detokenization

76

Summary

77

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**
  - Via **Anypoint Service Mesh**

78

Summary

MuleSoft

- **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
- **Anypoint Security** can enforce Edge policies to implement perimeter defence in customer-hosted deployments of Mule runtimes on **Anypoint Runtime Fabric**
- **De/Tokenization** can be applied to API invocation content by API policies that require Anypoint Security