
1

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

- Describe **origins of data** for monitoring, analysis and alerting
- Describe the **metrics** collected on the level of API invocations
- Describe the available **grouping** of API metrics for analysis
- Automatically visualize an application network with **Visualizer**
- Make use of options for performing **API analytics** in/outside of Anypoint Platform
- Define **alerts** for API invocations in all tiers of API-led connectivity
- Use metrics and alerts for **API implementations** to augment those for API invocations
- Recognize **operations teams** as stakeholder in API-related assets

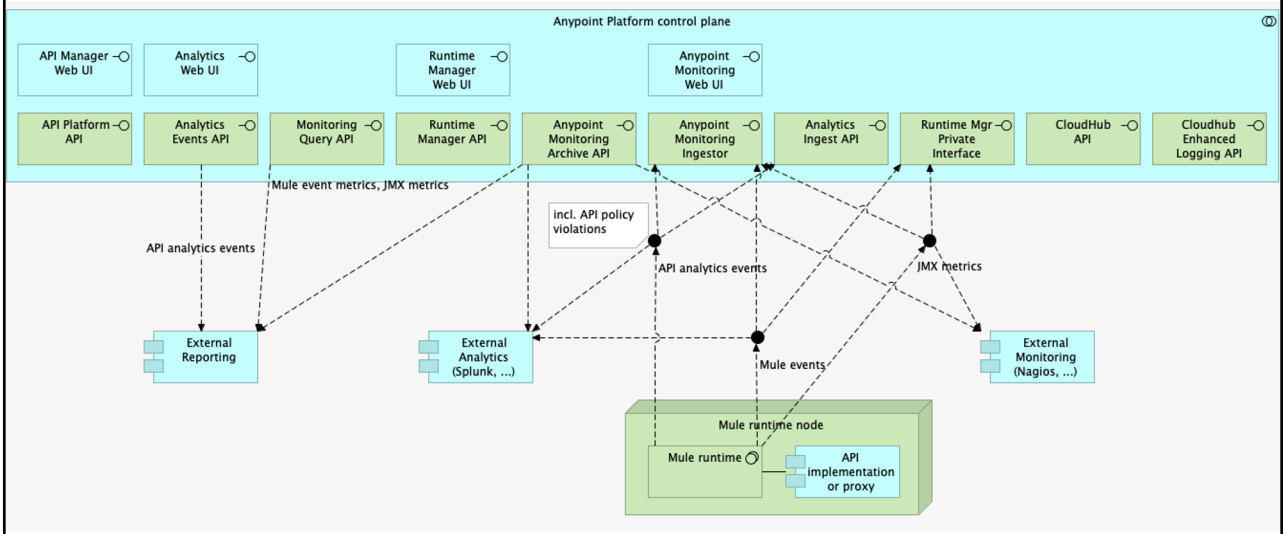



2

# Section 1
# Understanding monitoring data flow in Anypoint Platform

3

## Data flows for Anypoint Platform monitoring, analytics and alerting






4

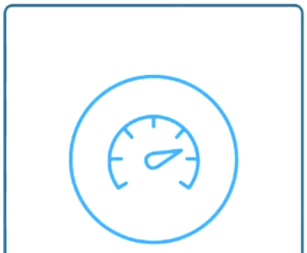
4

# Section 2
# Anypoint Monitoring

5

## What is Anypoint Monitoring?

MuleSoft



Application Performance Monitoring

Log Management

Custom Metrics & Events





8

## How Anypoint Monitoring helps to proactively monitor applications

**Reduce time to resolution of issues**

Anypoint Monitoring

**Ensure high performance and availablility**

API Functional Monitoring





9

## What features are available with Platinum subscription

| Application Performance Monitoring | Log Management | Custom Metrics & Events | Dedicated Monitoring Infrastructure | Enhanced Support |
|---|---|---|---|---|
| Application metrics | Distributed Log Management | Basic custom dashboards | High Frequency Data Collection | Reduced SLA Time |
| API functional monitoring | Codeless Logging | Custom Metrics | Increased Data Storage Capacity | |
| Basic Alerting | Log Warehousing | | Custom Data Retention & Location | |
| Connector Metrics | Advanced Log Search | | | |
| Reports | Log Tokenization | | | |





10

## What features are available with Titanium subscription

MuleSoft

| Application Performance Monitoring | Log Management | Custom Metrics & Events | Dedicated Monitoring Infrastructure | Enhanced Support |
|---|---|---|---|---|
| Application metrics | Distributed log search | Advanced custom dashboards | High frequency data collection | Reduced SLA time |
| Connector metrics | Codeless logging | Custom metrics | Increased data storage capacity | |
| API functional monitoring | Log warehousing | | Custom data retention & location | |
| Advanced alerting | Advanced log search | | | |
| Reports | Log tokenization | | | |

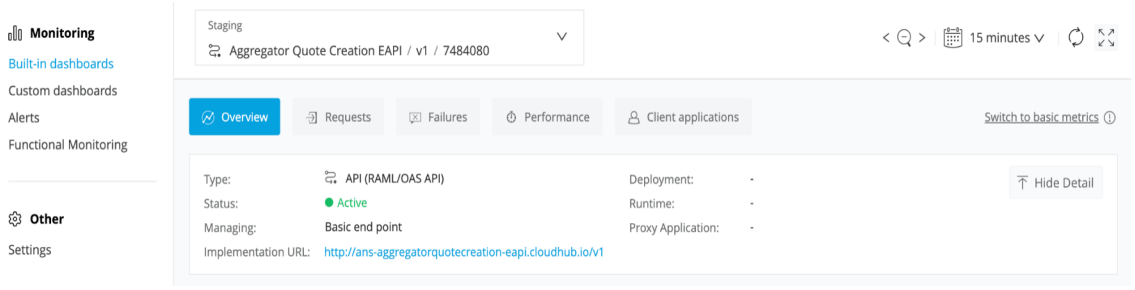


11

## Monitoring applications using the Anypoint Monitoring dashboard

MuleSoft

- Gives insight into application usage and performance
  - https://docs.mulesoft.com/monitoring/dashboards
- Helps to view dashboards (both built-in and custom)
- Needs special permission






12

12

## Monitoring applications using the Anypoint Monitoring **built-in dashboard**

- Default built-in dashboard in Anypoint Monitoring contains a set of time series charts to collect various metrics
  - Inbound and Outbound events
  - performance
  - Infrastructure
  - JVM
  - Failures
- Can also be visualized graphically in the **Application Network** (Covered in a later section)




13

## Built-in dashboard **traffic** metrics

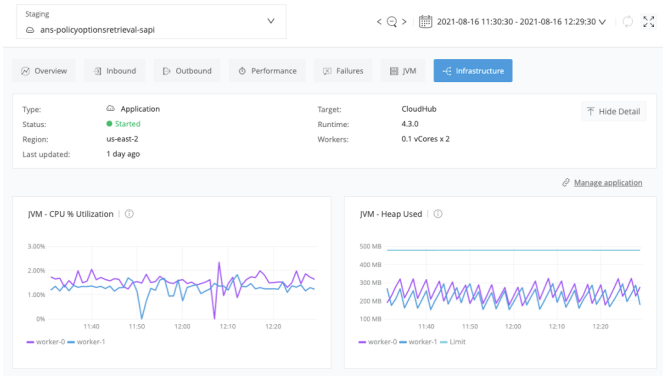
- Inbound
  - Total inbound requests, Avg. response time inbound, Inbound response time by endpoint, Total inbound requests by endpoint, Total inbound requests failed, Inbound protocols, Inbound endpoints
- Outbound
  - Total outbound requests, Avg. response time outbound, Total outbound requests by endpoint, Avg. response time outbound by endpoint, Total outbound requests failed
- Performance
  - Avg. response time inbound, Avg. response time outbound, Avg. response time inbound by endpoint, Avg. response time outbound by endpoint
- Failures
  - Total inbound requests failed, Total outbound requests failed, Total inbound requests failed by endpoint, Total outbound requests failed by endpoint




14

## Built-in dashboard **infrastructure** metrics

- JVM
  - GC count, Memory stats (heap and meta), Thread count
- Infrastructure
  - CPU, Memory utilization, Thread count, Total system memory and total system processor









## Monitoring applications using the Anypoint Monitoring **custom dashboard**

- Data visualization can be customized
- Different applications and API metrics can be combined in a single view
- Add multiple charts (**widgets**)


- Download the data behind a dashboard to a CSV file(Titanium)

## Monitoring applications with alerts

- Alerts can be configured in Anypoint Monitoring
  - Basic alerts
    - Alerts are set up for applications or servers as source
    - Different metric types are available for different sources (Mule applications vs. servers)
  - Advanced alerts
    - Can be created on widgets from custom dashboards
- Can send notifications to email addresses
- These alerts are distinct from API Manager alerts and Runtime Manager alerts




17

## Alert conditions related to Mule applications

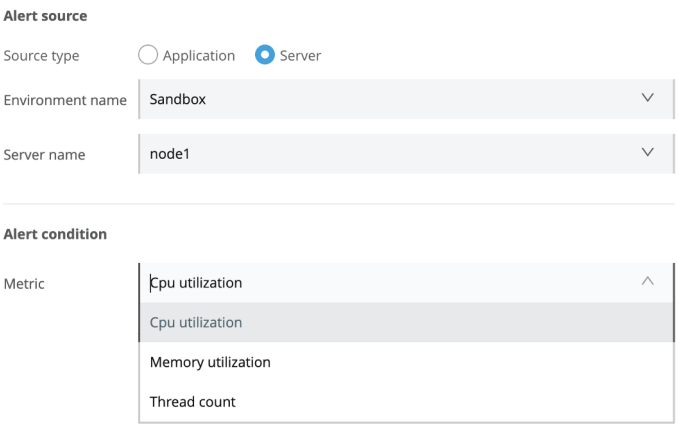
| CloudHub | On-prem |
| --- | --- |
| • Message count | • Message count |
| • Message error count | • Message error count |
| • Message response time | • Message response time |
| • CPU utilization | |
| • Memory utilization | |
| • Thread count | |




18

## Alert conditions related to servers

- Server alerts
  - CPU Utilization
  - Memory Utilization
  - Thread Count



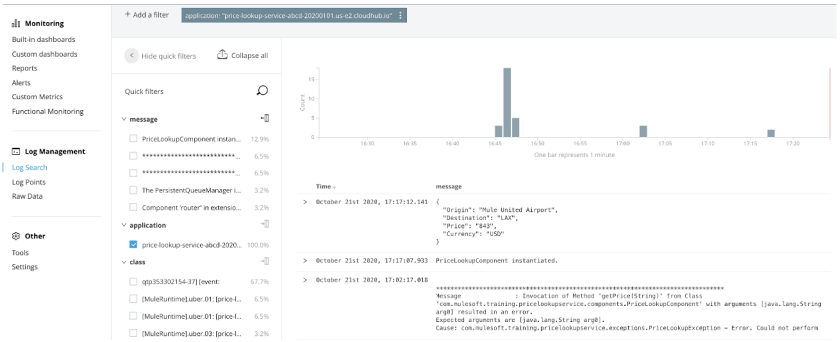






## Log management by Anypoint Monitoring

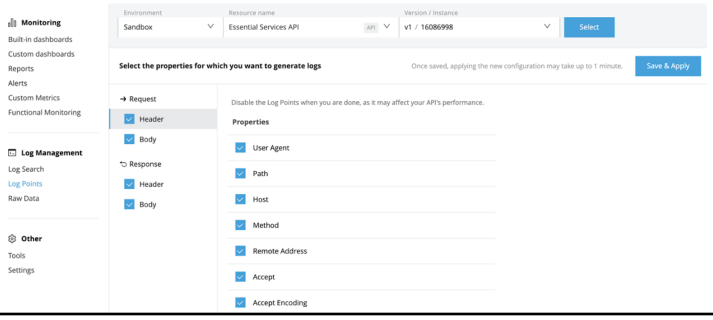
- Require Titanium subscription
- Search and analyze logs of one or more(all) Mule applications
- Reduces the time to resolution of issues with dashboard integration of log management



## How Log Points help Mule applications to log

MuleSoft

- Codeless logging
- Configures log level for supported connectors used within a Mule application
- Configure log message content for API Instances
- As always extensive logging may degrade performance








21

# Section 3
# Functional Monitoring

22

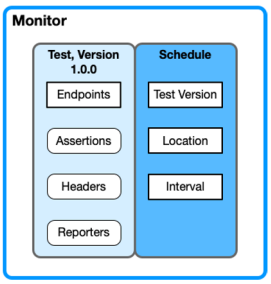
## Introducing API Functional Monitoring

- **API Functional Monitoring** is a capability of Anypoint Monitoring that allows developers and operators to test, monitor, and verify an API's functionality **repetitively** and **automatically** in any "production-like" environment
- Certain **API behaviors** act as leading indicators to API failure, and are not necessarily related to QA tests
  - For example, a downstream API performance/availability failure
- Functional Monitors can be used to watch for these behaviors and respond in real-time
  - They use **behavioral test cases** that use real inputs and expected outputs, and exercise dependencies
- Created via Anypoint Monitoring Web UI, Command-line tool, or the Anypoint Platform API






## Functional Monitors




- Consist of a **versioned test** and a **schedule**
  - Test
    - **Endpoint**
    - Optional **assertions** to verify HTTP Response codes (e.g. 200)
    - Optional **HTTP Headers**
    - Optional **reporting tools** (e.g., SumoLogic, Slack, etc.)
  - Schedule
    - Test **version** (e.g., 1.0.0)
    - **Location** - public or private
      - **Public** locations are shared, regional (e.g., us-east-1) resources and are limited in the number available, duration of test, and frequency – typically used to monitor public APIs
      - **Private** locations run in Anypoint VPC and consume a small amount of vCores and are only limited by available vCores – private APIs typically require private monitor locations
    - **Interval** - frequency of test

## Applying API Functional Monitoring at Acme Insurance

- Endpoints:
  - Healthcheck endpoints: API implementations expose endpoints for Kubernetes-style "probes":
    - For a liveness probe at /alive, returning 200 if alive or 500 if not
    - For a readiness probe at /ready, returning 200 if ready or 500 if not
      - Readiness requires the API implementation to verify that all its dependencies are alive. For API dependencies this means invoking /alive.
  - Monitors must only use idempotent HTTP methods
  - Use the same HTTP Listener configuration as the main API endpoints




25

## Applying API Functional Monitoring at Acme Insurance

- Functional Monitors:
  - Implemented in code and checked-in with the code for the API Implementation in each environment
  - Public APIs will use public monitor locations but may use private monitors when required
  - Private APIs will use private monitor locations
  - Monitors will run at intervals appropriate to detect behavioral anomalies and may vary across API Implementations
  - Failures and exceptions will be reported to a centrally-monitored, operator email address



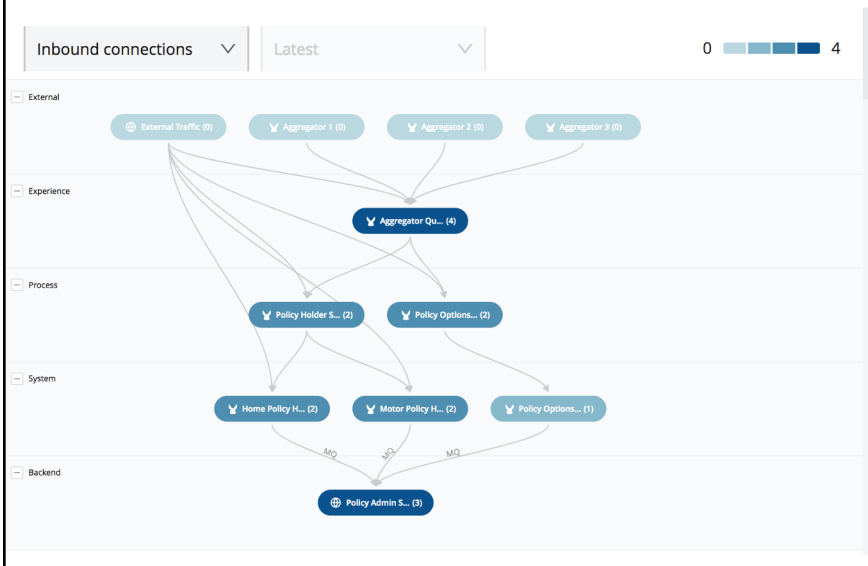

26

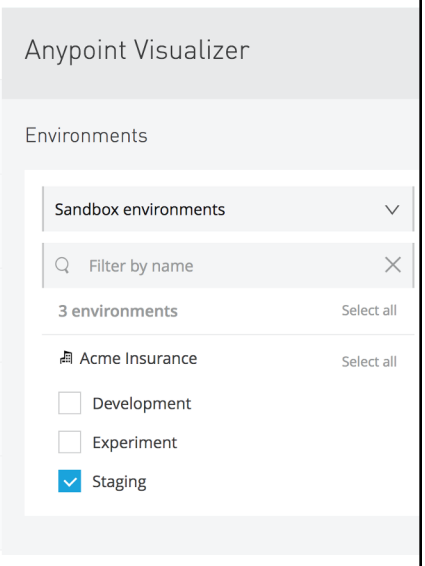
# Section 4
# Visualize the application network with Anypoint Visualizer

27

## Introducing Anypoint Visualizer

MuleSoft






28

28

## Introducing Anypoint Visualizer

- Visual **application network explorer**
- Integrated into MuleSoft-hosted Anypoint Platform control plane
- Shows automatically rendered graph of application network
- **Nodes: application components**
  - Special support for Mule apps deployed to CloudHub
  - All others categorized as "external"
  - Arbitrary layers: System, Process and Experience pre-defined
  - Colored by inbound connections, response time, throughput, failures, ...
- **Edges: request-response interactions** detected at runtime
  - Direction from originator to target
  - Includes all API invocations
  - Analyzes IPs and URLs
  - Data collection through Mule runtime and Connectors




29

## Introducing Anypoint Visualizer

- Nodes and edges **dynamically updated to reflect actual traffic**
- Just select one or more **environments** to show
- Assign arbitrary **labels**
- Currently **no concept of APIs** and API instances as in API Manager




30

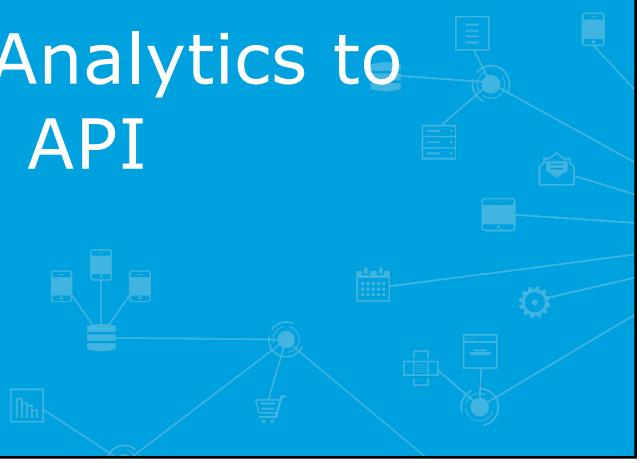
## Use-cases for Anypoint Visualizer

- **Discover** baseline application architecture
- Identify duplicate interactions as **preparation for consolidation**
- **Architecture governance**
  - Identifying violations of API-led connectivity
- **Impact analysis** for proposed changes
- **Comparing environments**
- As substitute for proper **documentation**
- Visualizer shows traffic detected at runtime
  - Not suitable for detecting potential interactions




31

# Section 5
# Using Anypoint Analytics to gain insight into API invocations



32

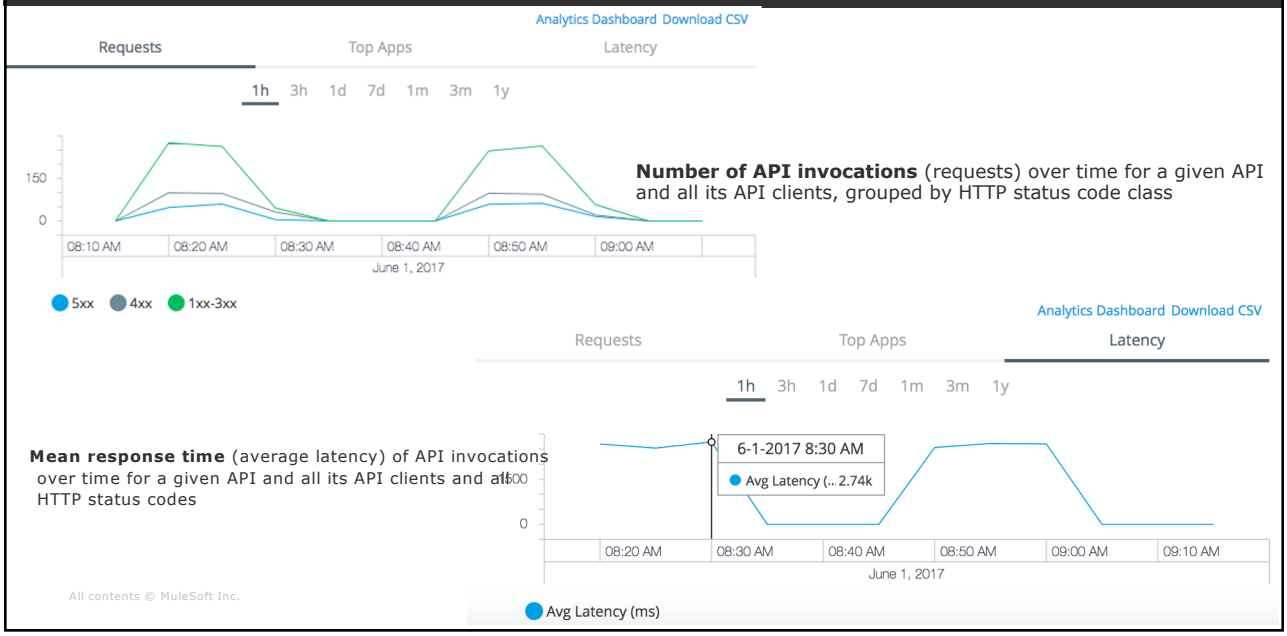
## Metrics in Anypoint Analytics

- **Number** of API invocations (requests)
  - Successful: **[100, 400)**
  - Unsuccessful due to a client error: **[400, 500)**
  - Unsuccessful due to a server error: **[500, 600)**
- Mean **response time** (average latency)
- Request and response **payload size**
- Properties of the **API client**:
  - Client ID (if registered), geographical location, OS platform, ...
- Properties of the **API invocation**:
  - resource path, HTTP method, ...
- Metrics can be **grouped** and displayed along various dimensions:
  - for one/all API(s) and one/all API client(s)
  - custom




33

## Metrics in Anypoint Analytics

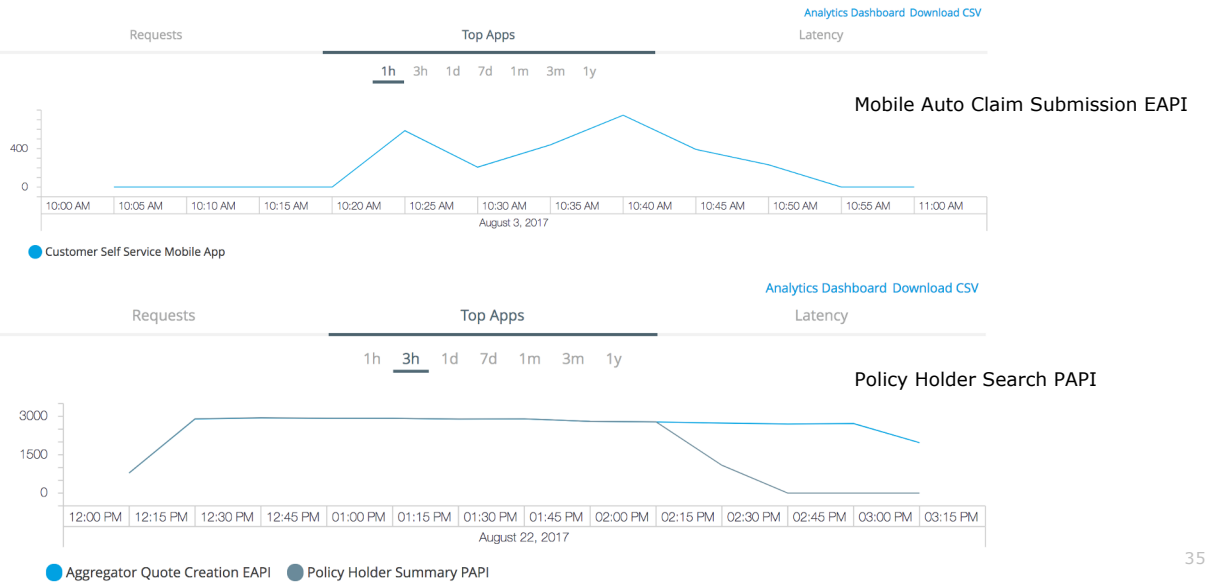





34

# Analyzing API invocations

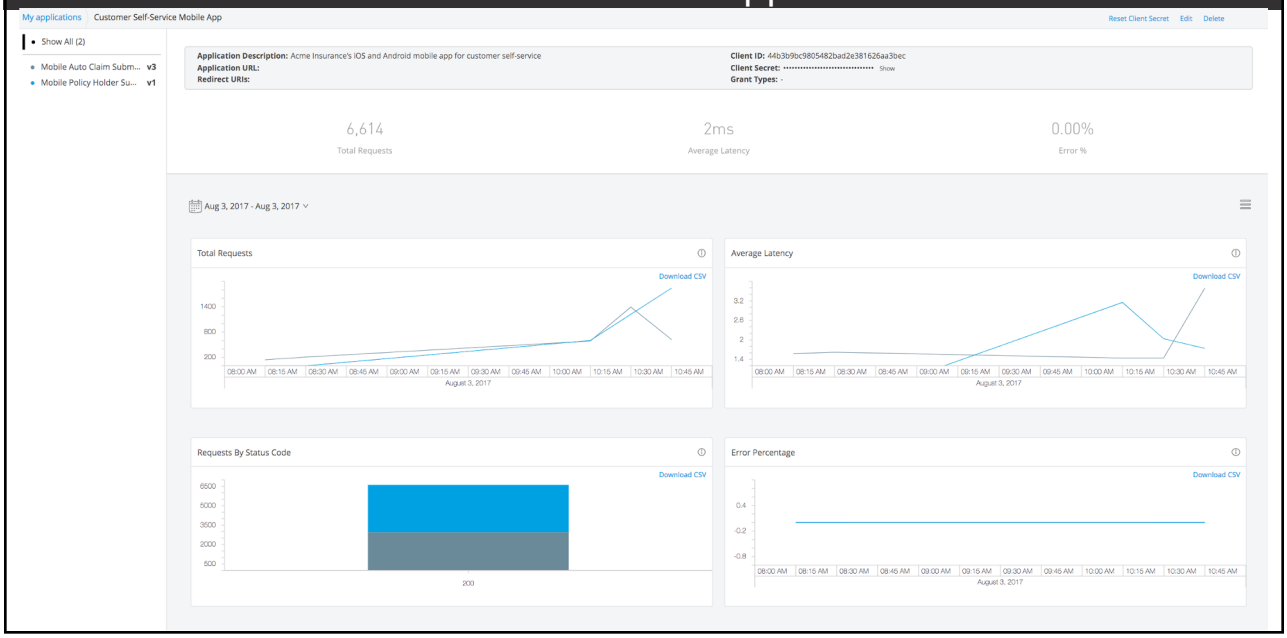
**Number of API invocations** (requests) over time grouped by each of its top 5 API clients




35

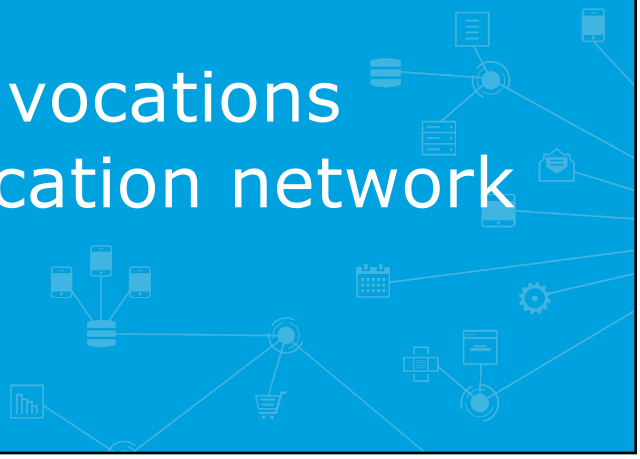
35

# Analyzing API invocations from the perspective of the Customer Self-Service Mobile App




36

# Section 6
# Analyzing API invocations across the application network



37

## Introducing application network-wide API analytics

MuleSoft

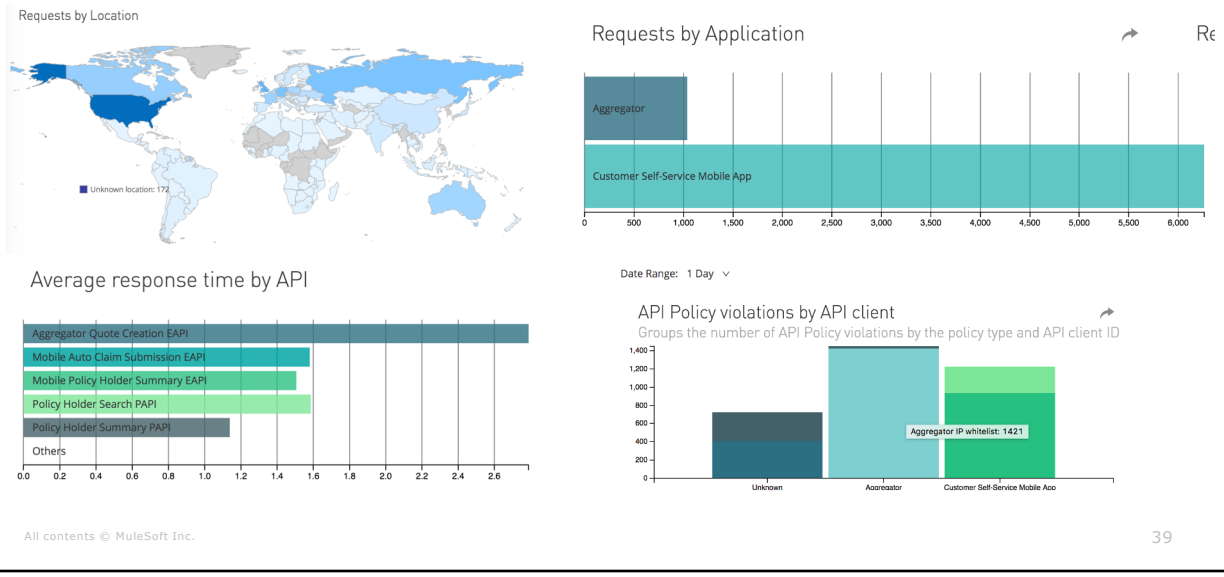
- **Anypoint Analytics** can perform standard and custom analyses across all API invocations in an application network:
  - **Interactive** exploration through drill-down
  - Definition of **custom charts and dashboards**
  - Definition of **custom reports**
  - Exporting all data underlying a graph to **CSV** files
  - Access to all data via Anypoint **Platform APIs**





38

## API analysis examples










# Section 7
# Defining alerts for exceptional occurrences in an application network

## Introducing alerts at the level of API invocations

MuleSoft

- **Alerts based on these metrics** of API invocations:
  - Number of **violations** of a given policy
  - Request count (**number of API invocations**)
  - **Response code** in given set of HTTP response status codes
  - **Response time** exceeding given threshold in milliseconds
- Alert is triggered **when** the metric
  - Falls above/below a **threshold**
  - For a given number of **time periods** of a given duration
- **C4E guideline**: alerts should at least cover:
  - All **violations of API policies**
  - All **violations of QoS guarantees** not explicitly captured in API policies




45

## Defining alerts for "Policy Options Retrieval SAPI"

MuleSoft

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 >

Add alert | Search

| | Name | Type | Date modified | Date created | Enabled | |
|---|---|---|---|---|---|---|
| > | Client not in Process API subnet for "Policy Options Retrieval SAPI" | Policy | 1/17/18 6:32 PM | 1/17/18 6:32 PM | Yes | Edit Delete |
| > | SLA tier exhausted for "Policy Options Retrieval SAPI" | Policy | 1/17/18 6:31 PM | 1/17/18 6:31 PM | Yes | Edit Delete |




46

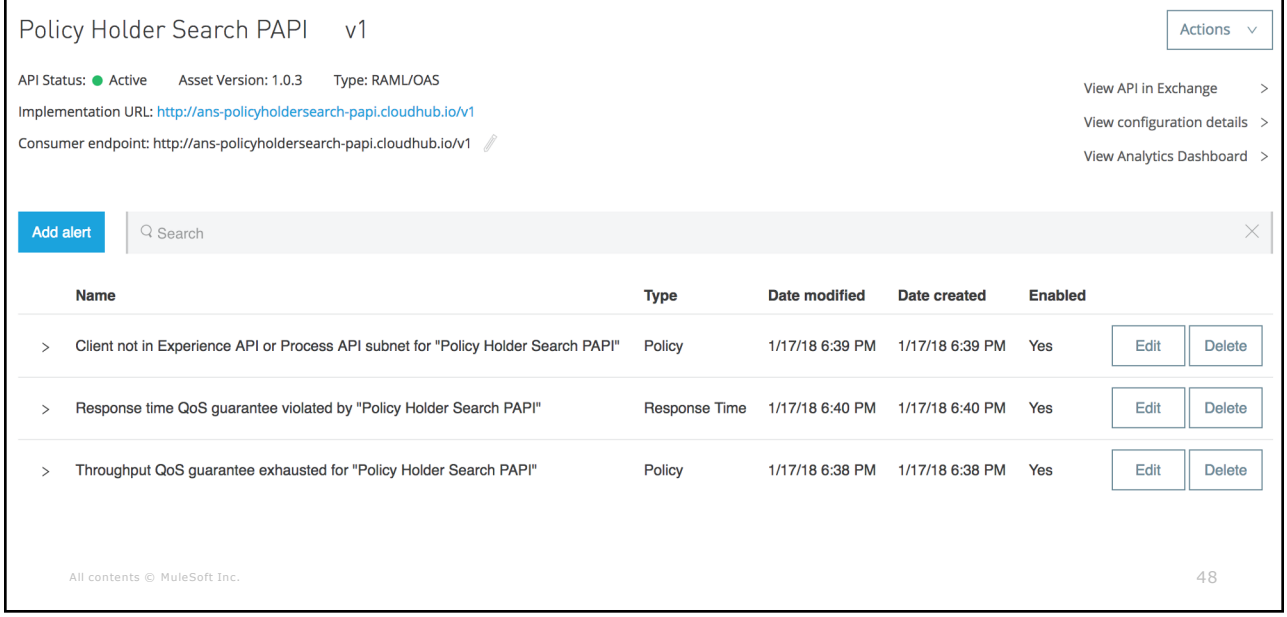
## Defining alerts for "Policy Options Retrieval SAPI"

- **SLA tier exhausted**:
  - Violation of **SLA-based Rate Limiting**, severity **Info**, > 60 violations for at least 3 consecutive 10-minute periods
  - Alerts when approx. **10%** of 1-second intervals are **above limit defined by SLA tier**
  - Also alerts on **invalid client ID/secret** supplied
- **Client not in Process API subnet**:
  - Violation of **IP allowlist**, severity **Critical**, > 1 violation for at least 3 consecutive 1-minute periods
- Could add alert for violations of **Spike Control**




47

## Defining alerts for "Policy Holder Search PAPI"







48

## Defining alerts for "Policy Holder Search PAPI"

MuleSoft

- **Throughput QoS guarantee exhausted**:
  - Violation of **Spike Control**, severity **Info**, > 60 violations for at least 3 consecutive 10-minute periods
  - Alerts when approx. **10%** of 1-second intervals are **above limit**
- **Client not in Experience API or Process API subnet**:
  - Violation of **IP allowlist**, severity **Critical**, > 1 violation for at least 3 consecutive 1-minute periods



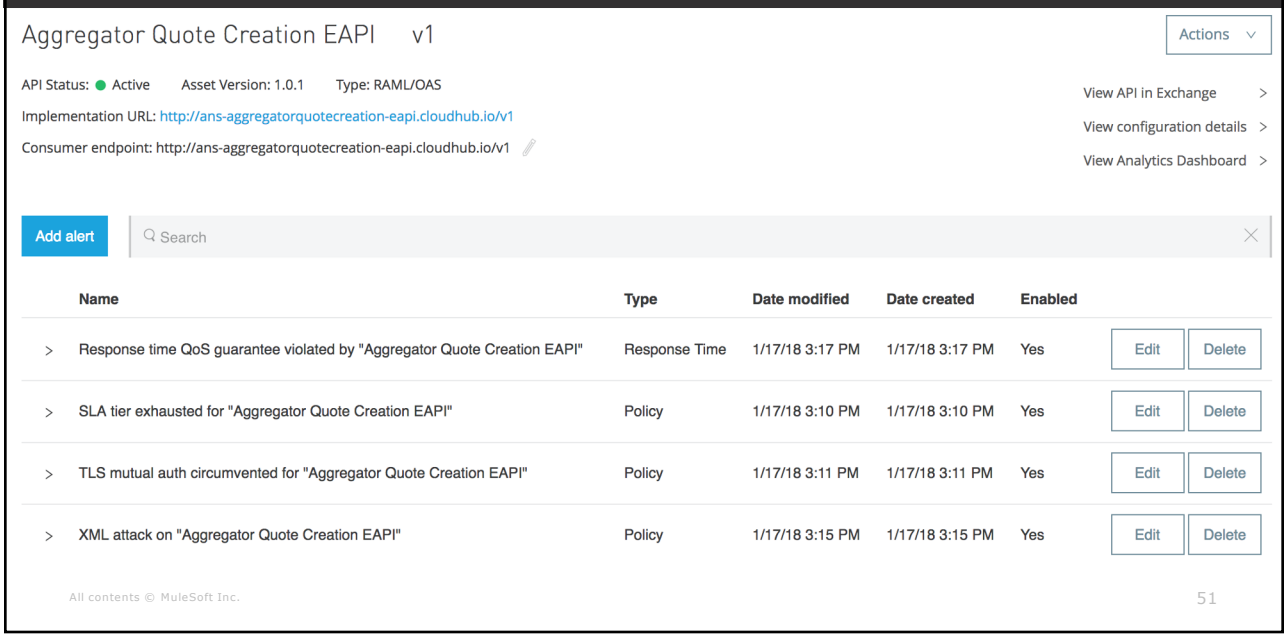



## Defining alerts for "Policy Holder Search PAPI"

MuleSoft

- **Response time QoS guarantee violated**:
  - Severity **Warning**, > 6600 requests whose response time > 100 ms for at least 3 consecutive 10-minute periods
  - Alerts when approx. **1%** of API invocations (1% of 1100*60*10 = 6600) are **above limit** of 100 ms (twice the target median of 50 ms)
  - Note that **exact QoS guarantee** cannot be expressed in alert
    - median = 50 ms, maximum = 150 ms
- Should add alert for violations of **Client ID enforcement**

## Defining alerts for "Aggregator Quote Creation EAPI"






51

51

## Defining alerts for "Aggregator Quote Creation EAPI"

- **SLA tier exhausted**:
  - Violation of **SLA-based Rate Limiting**, severity **Info**, > 60 violations for at least 3 consecutive 10-minute periods
  - Alerts when approx. **10%** of 1-second intervals are **above limit defined by SLA tier**
- **TLS mutual auth circumvented**:
  - Violation of **IP allowlist**, severity **Critical**, > 1 violation for at least 3 consecutive 1-minute periods



52

52

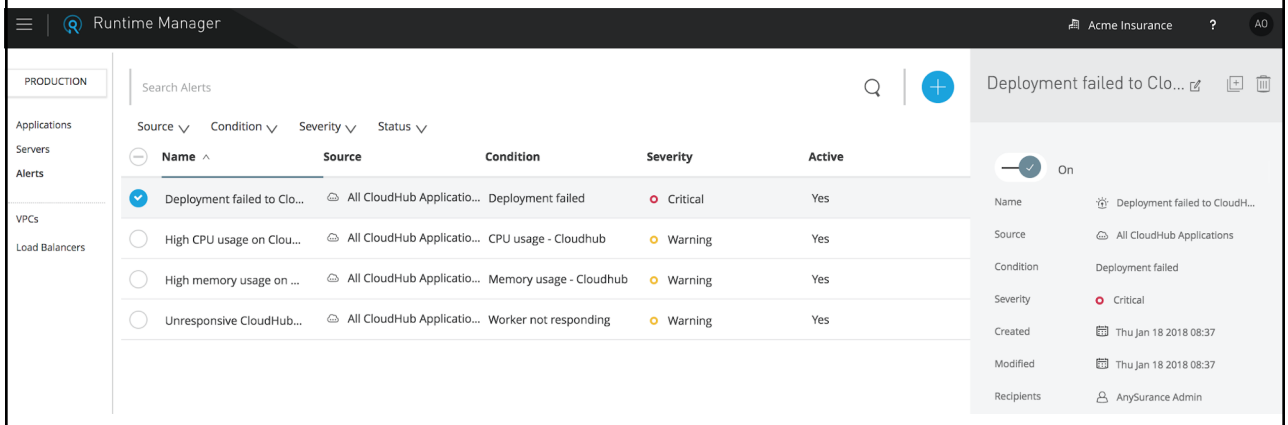
## Defining alerts for "Aggregator Quote Creation EAPI"

- **XML attack**:
  - Violation of **XML threat protection**, severity **Warning**, > 30000 violations for at least 3 consecutive 10-minute periods
  - Alerts when approx. **5%** of requests (5% of 1000*60*10 = 30000) are identified as XML threats
- **Response time QoS guarantee violated**:
  - Severity **Warning**, > 6000 requests whose response time > 400 ms for at least 3 consecutive 10-minute periods
  - Alerts when approx. **1%** of API invocations (1% of 1000*60*10 = 6000) are **above limit** of 400 ms (twice the target median of 200 ms)
  - Note that **exact QoS guarantee** cannot be expressed in alert
    - median = 200 ms, maximum = 500 ms






## Alerts on API implementations augment alerts for API invocations



## Alerts on API implementations augment alerts for API invocations

- Alerts for **API invocations and API implementations** complement each other
- If API implementation **crashes** but no API client invokes that API then no alert on the level of API invocations will be raised
  - But remember **auto-restart**
- Consistently high **CPU usage**
- **Deployment failures** in production and staging environments




55

# Section 8
# Organizing discoverable documentation for operations

56

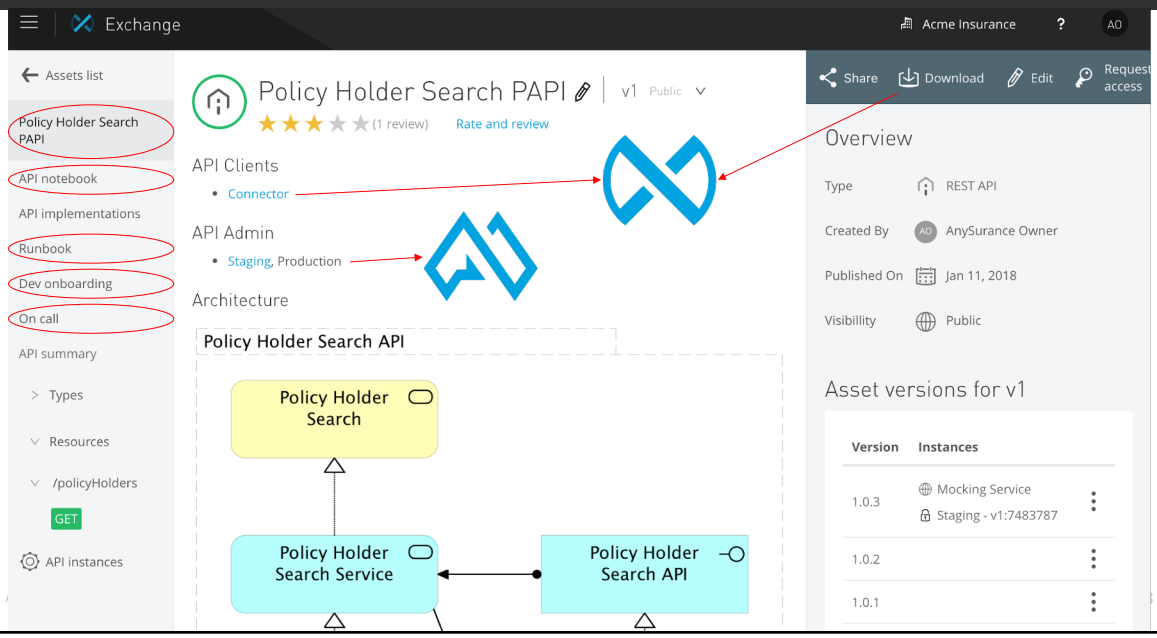
## Operations teams as a stakeholder in APIs

- **Development teams** may also operate the APIs and API implementations they implement
  - Thereby become **operations teams**
- Operations teams **need**
  - **Dashboards and alerts**
    - **Anypoint Monitoring**
    - **Runtime Manager**, **API Manager**, Anypoint **Analytics**
  - Custom-written **documentation**:
    - **Runbooks**: how to address **alerts**
    - **On-call registers**: **who** to contact
- Should be discoverable through **Exchange**



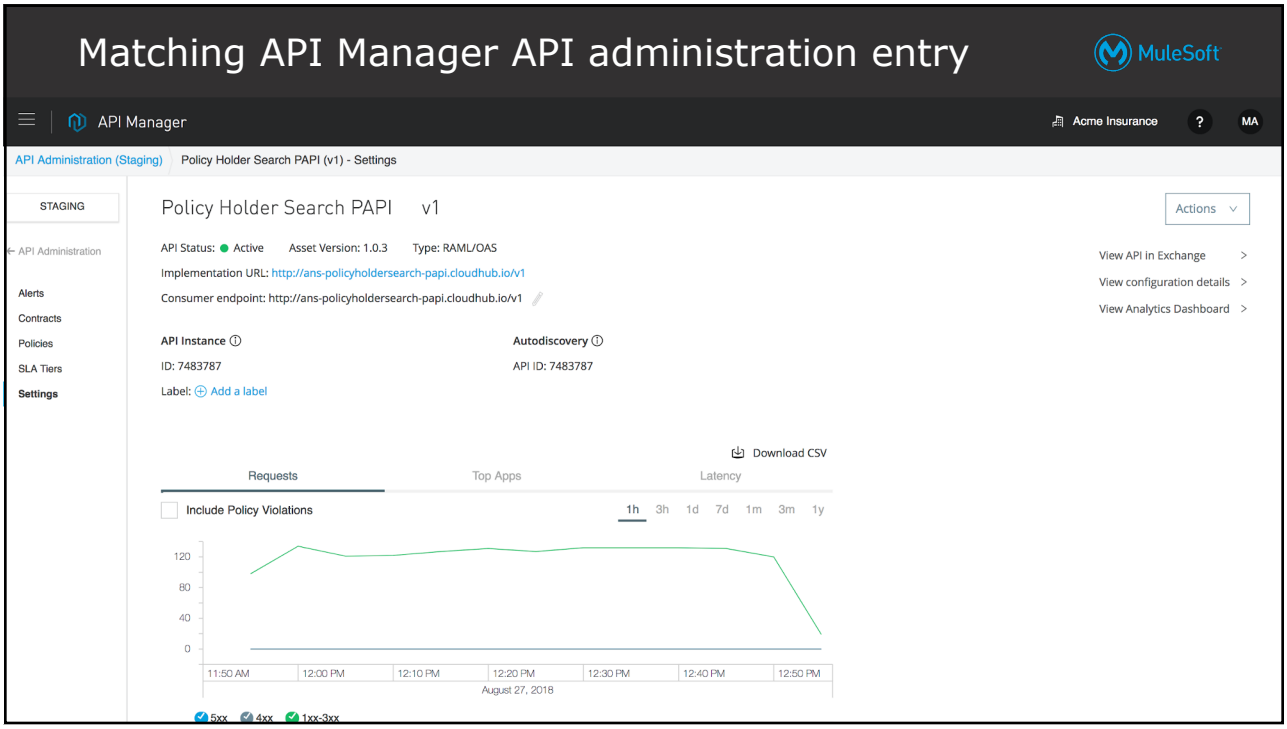

57

## Exchange entry is portal to API's documentation and assets



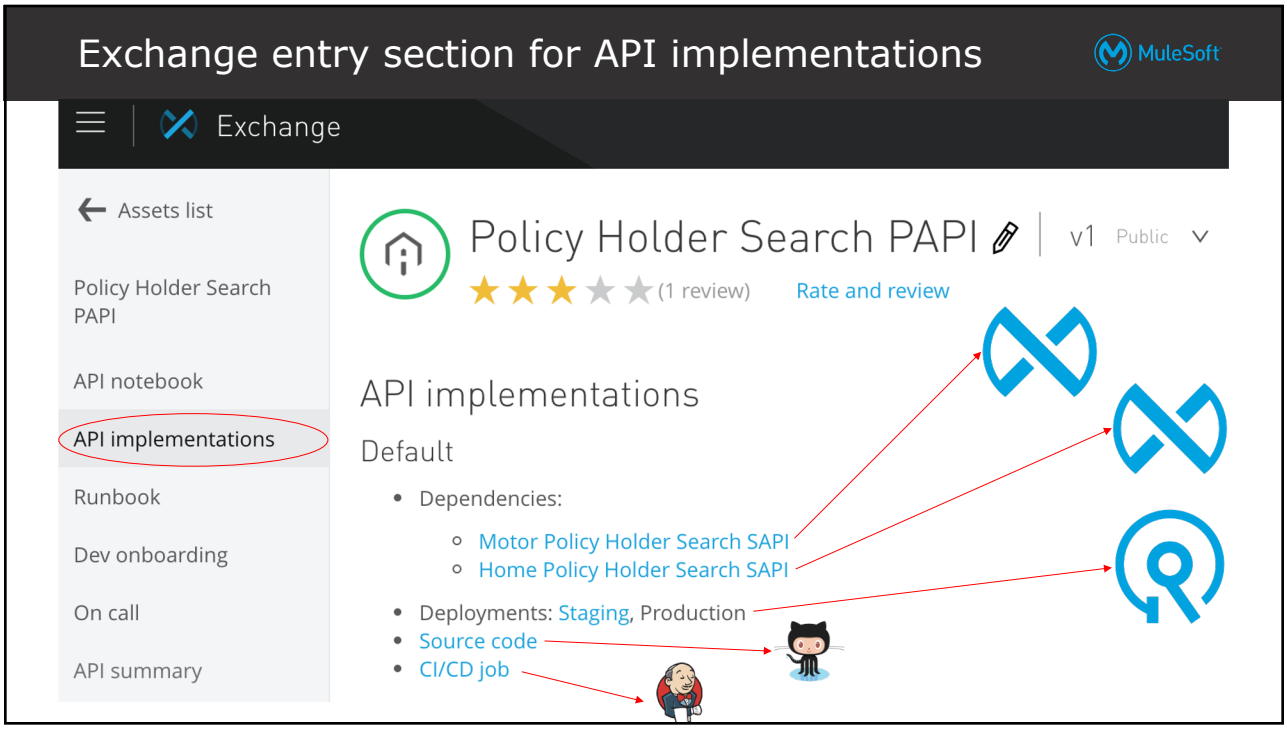

58

# Matching API Manager API administration entry



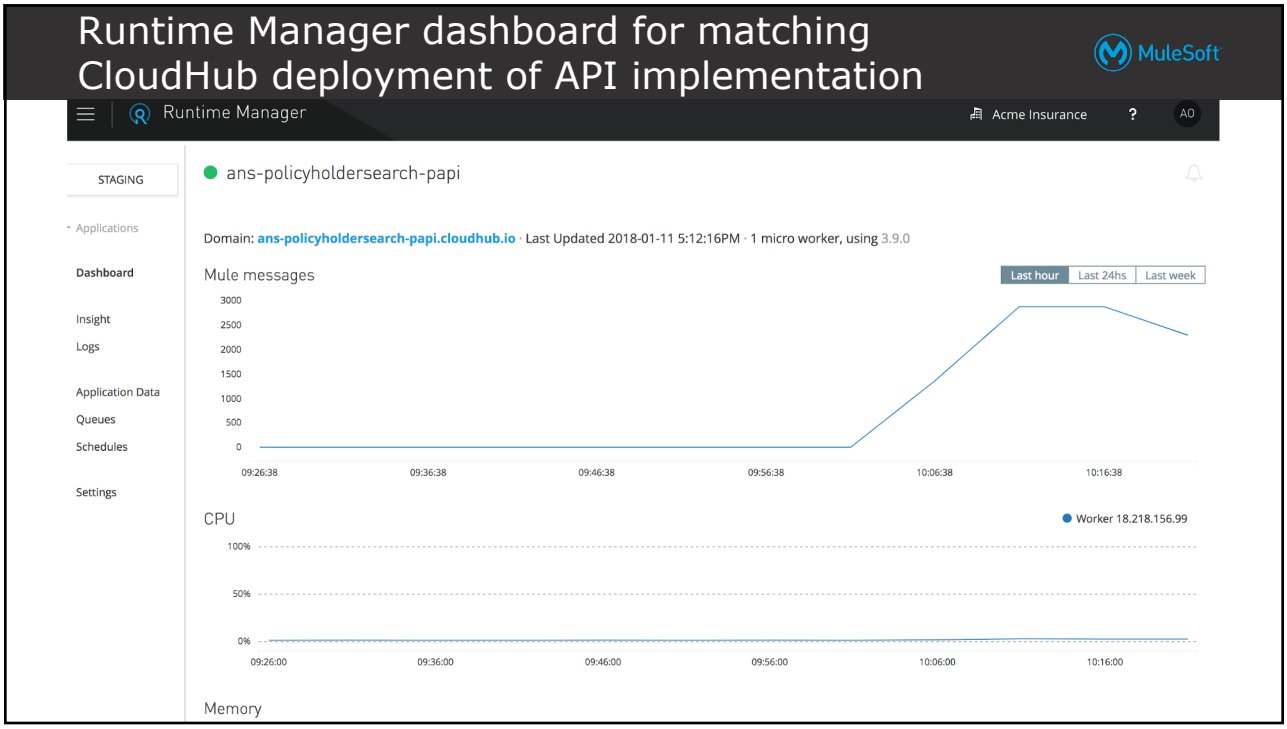

59

# Exchange entry section for API implementations




60

# Runtime Manager dashboard for matching CloudHub deployment of API implementation




61

# Summary

62

## Summary

- **Data** used in monitoring, analysis and alerting flows from Mule runtimes to external monitoring/analytics systems and/or Anypoint Platform
  - **Available via APIs** for external reporting
- Anypoint Platform collects numerous **metrics** for API invocations:
  - Response time, payload size, client location, ...
- Metrics can be **grouped** by API, API client or any of the other metrics
- Analyses targeted specifically at **API consumers** and clients






## Summary

- Anypoint **Analytics** supports
  - Interactive analyses, custom charts and reports
  - Data download in CSV files and/or retrieval through Anypoint Platform APIs
- **Alerts** defined based on API invocation metrics:
  - Request count and time, response status code
  - Number of API policy violations
- Metrics and alerts for **API implementations** defined in Runtime Manager augment API invocations metrics and alerts
- **Operations teams** are important stakeholder in API-related assets: structure and link assets to support them