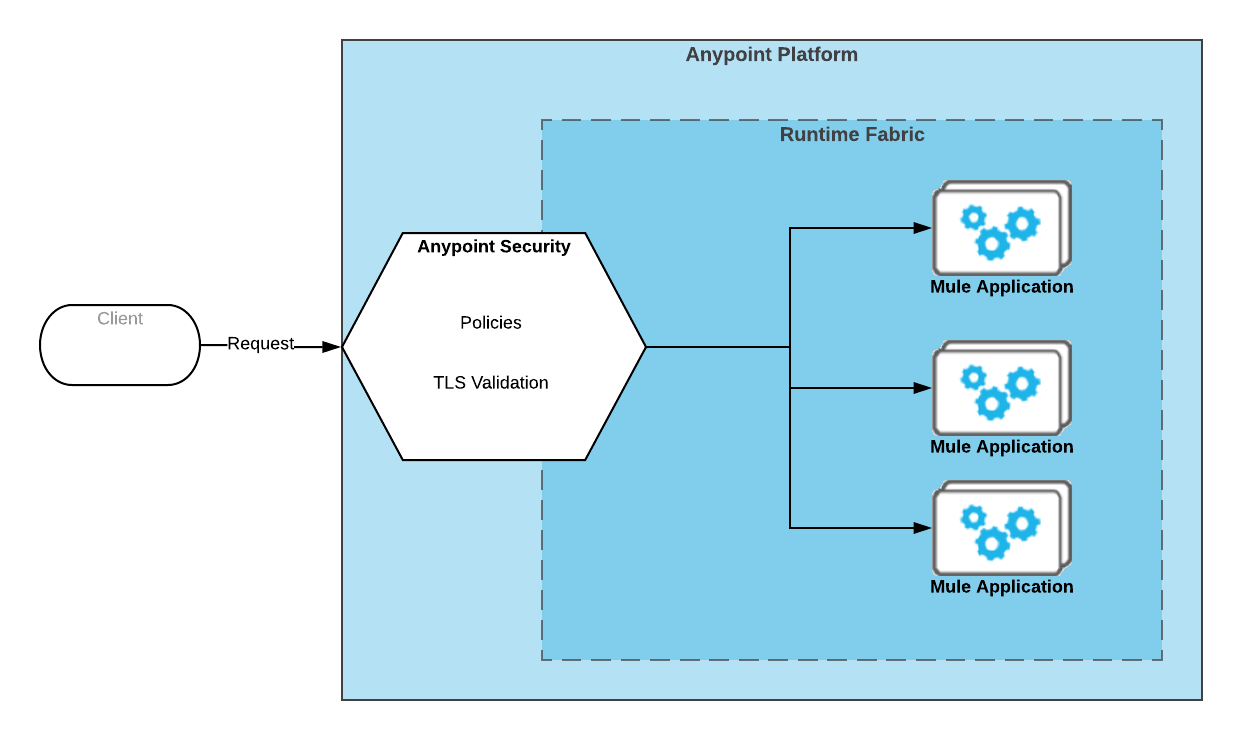
**Anypoint Security**

Anypoint Security provides a layered approach to secure your application network. These layers work together to protect both the application network and the network’s individual nodes by controlling access to APIs, enforcing policies, and proxying all inbound or outbound traffic to mitigate external threats and attacks.

Anypoint Security features a high-performance, reliable, and scalable service to enforce policies that apply to all nodes deployed to our Runtime Fabric.

**Anypoint Security Architecture**

Anypoint Security provides you with a dedicated endpoint to detect attacks and validate traffic without taxing your network implementations.



**Policies**

Anypoint Security features a high-performance, reliable, and scalable service to enforce policies that apply to all nodes deployed to your Runtime Fabric.  
Anypoint Security policies then act as a default firewall/router capability through which all traffic traverses.

**Secrets Manager**

Anypoint Security provides a secure vault for you to store the TLS certificates and keystores used by your deployments.  
Through Secrets Manager, you can store the necessary secrets to set up the TLS context that encrypts inbound traffic to your Runtime Fabric.

**Tokenization Service**

Anypoint Security’s tokenization service protects sensitive data from unwanted exposure by replacing key values, such as a credit card number, with a token.

**Secrets Manager**

Anypoint Security provides a secure vault for you to store the TLS certificates and keystores used by your deployments.  
Through Secrets Manager, you can store the necessary secrets to set up the TLS context that encrypts inbound traffic to your Runtime Fabric.

**Anypoint Security Policies for Edge**

Anypoint Security features a high-performance, reliable, and scalable service that works with Anypoint Runtime Fabric to enforce security policies on nodes deployed to your Anypoint Runtime Fabric. Anypoint Security provides denial-of-service (DoS), IP whitelist, HTTP limits, and Web Application Firewall (WAF) policies to protect your APIs.

## **DoS Policy**

DoS policies are designed to protect your network nodes against malicious clients trying to flood your network to prevent legitimate traffic to your APIs.

## **IP Whitelist Policy**

Create an IP address whitelist policy to configure an explicit list of IP addresses that can access your deployed endpoints.

## **HTTP Limits Policy**

HTTP limits policies prevent attacks from clients that send large messages that can consume all of your processing bandwidth.

## **Web Application Firewall Security Policy**

The Web Application Firewall (WAF) security policy is available for request and response traffic to provide protection at the Web application level.

Examples:

**SQL Injection**

It is an attack used to take over database servers by dynamically poisoning SQL queries. SQL injection works only on web application that is using the DB.

**CSS (Cross Site Scripting)** is a code injection attack executed on the client-side of a web application.

* Attacker injects malicious script through the web browser
* The malicious script is executed when the victim visits the web page or web server
* Steals cookies, session tokens and other sensitive information.
* Modify the contents of the website

**Tokenization Service**: Tokenization is a highly effective way to protect the sensitive data. When you tokenize data, sensitive data elements are substituted with randomly generated non-sensitive data elements. As a result we can protect the sensitive information even if someone is able to breach APIs

Examples of sensitive information that are suitable for tokenization protection include:

o   Financial Information: Primary Account Number (PAN)

o   Personally Identifiable Information (PII): address, phone number, email

o Protected Health Information (PHI): medical histories, test result, insurance  information

It runs inside the network. There are three sub-components in tokenization services. They are:

1.      Masking

2.      Encryption

3.      Tokenization

## **Masking**

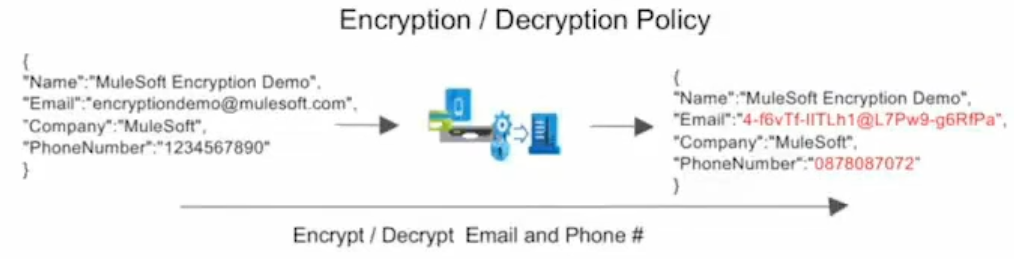
It is a one way process, so here we cannot get original value back. A simple example is the masking an account and phone number



## **Encryption**

It is a format preserving encryption. In this the data is encrypted using some techniques  while the format of the data is maintained. Mule Security offers the

ability to encrypt or decrypt message content in within a Mule flow by utilizing Mule message processors, allowing you to maintain the integrity of your messages



**3. Tokenization:** It is the process of masking a value or piece of information that can be considered sensitive data into a token that can be mapped back to its original value using detokenization concepts. The token is displayed in the same format as the original value, but its actual value is not revealed initially. The token and the original value is stored in the vault. If an API contains sensitive data, the tokenization policy is a highly effective way to protect it.

**Secret Manager:**

The secrets manager is used to write, read, and manage your secrets, keys, and Transport Layer Security (TLS) certificates within a unique source that allows access to other authorized platform services on your behalf.

This is the central and secure repository to manage the secrets.Secrets manager is supported on Runtime Fabric and API Manager only. Secrets manager uses secure vault technology to store and control access to private keys, passwords, certificates, and other secrets.

Anypoint Secrets Manager lets you store these secrets in secret groups, which are vaults associated with your environment and business group.