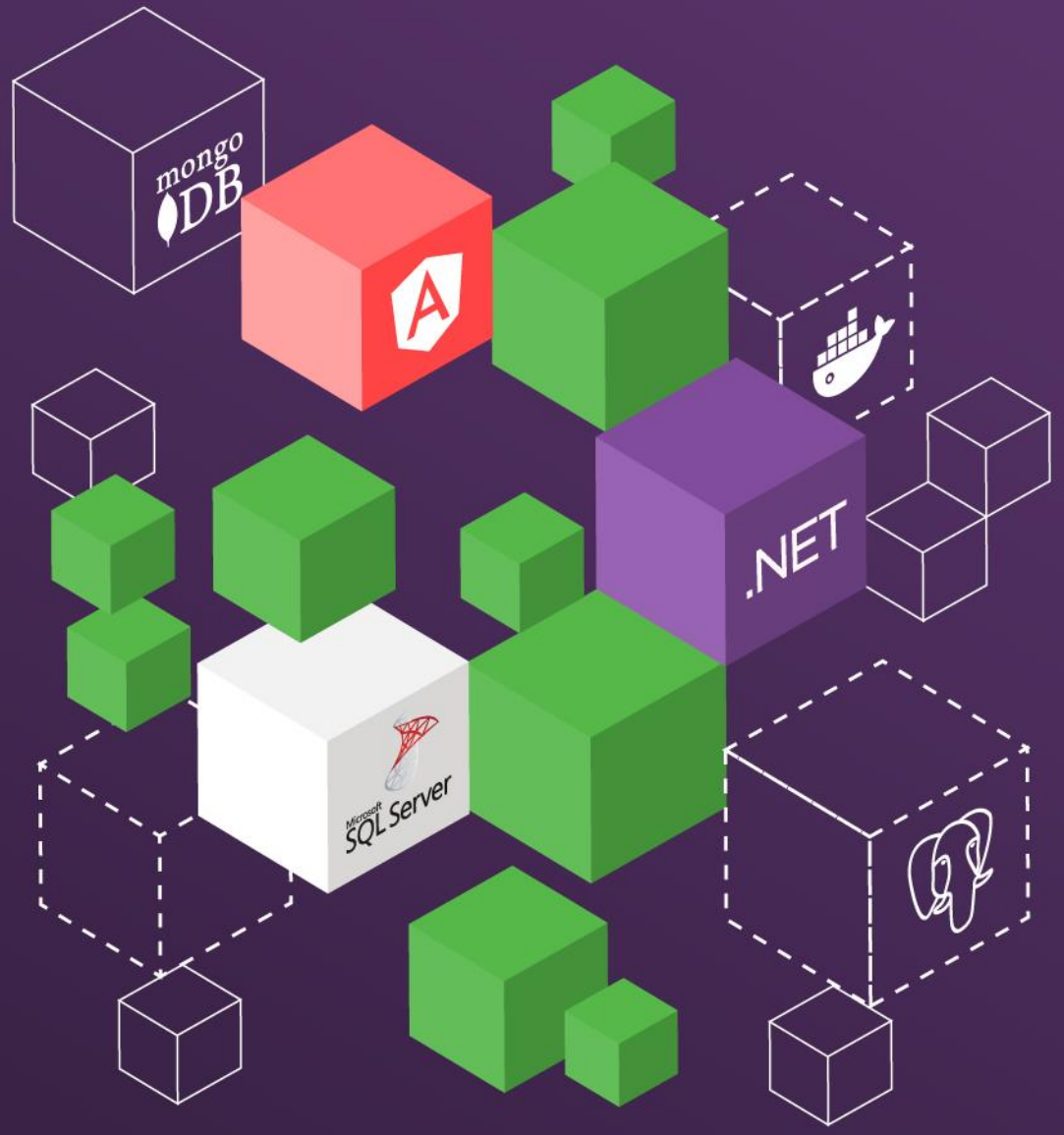


BIENVENIDOS
AL CURSO:

Arquitectura de Microservicios en Net

GALAXY
TRAINING

01 Implementando una interfaz de usuario que consuma directamente los microservicios (Angular 8).

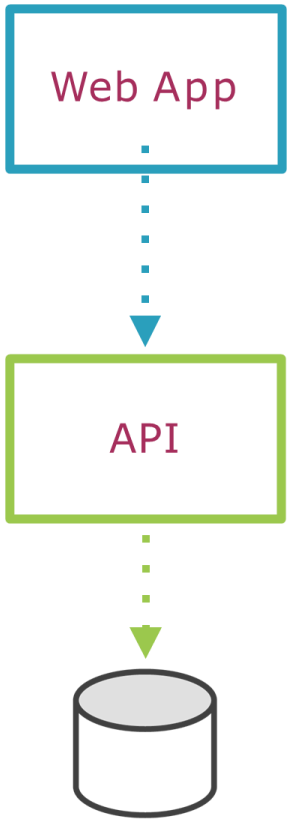
02 Centralización de accesos a los microservicios utilizando el patrón API Gateway.

03 Implementación de un API Gateway con Azure Api Management.

04 Actualización de la interfaz de usuario para el consumo del API Gateway.

ÍNDICE

¿Cómo consumir los Microservicios?



Centralizar el acceso a los microservicios

- API Gateway

¿Por qué?

- Funcionalidad distribuida
- Datos distribuidos
- Seguridad distribuida

Complejidad para las aplicaciones del cliente

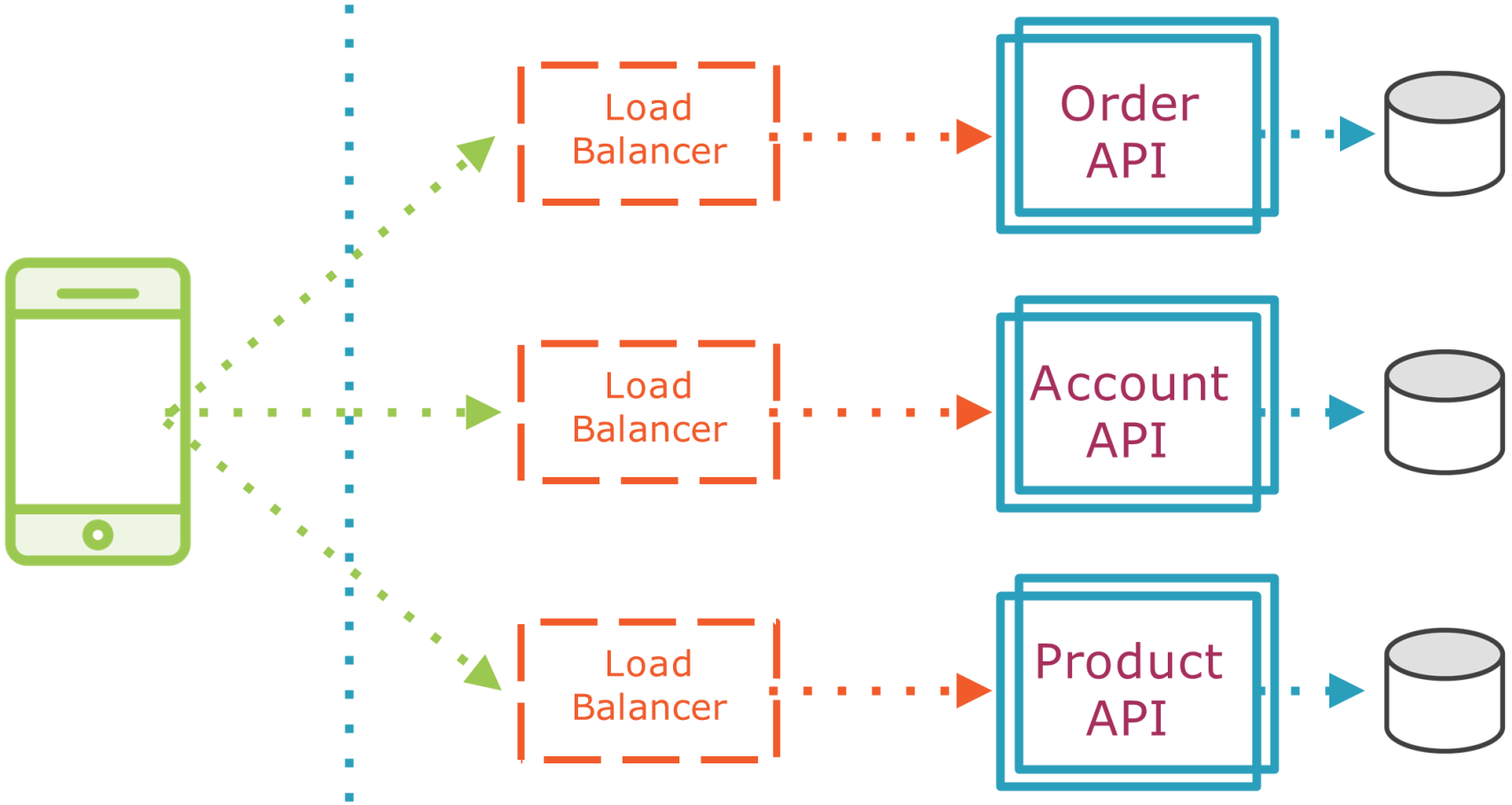
- Consolidar datos de API
- Consolidar las funcionalidades de la API
- Gestionar la seguridad

Experiencia de cliente complicada

- API como producto

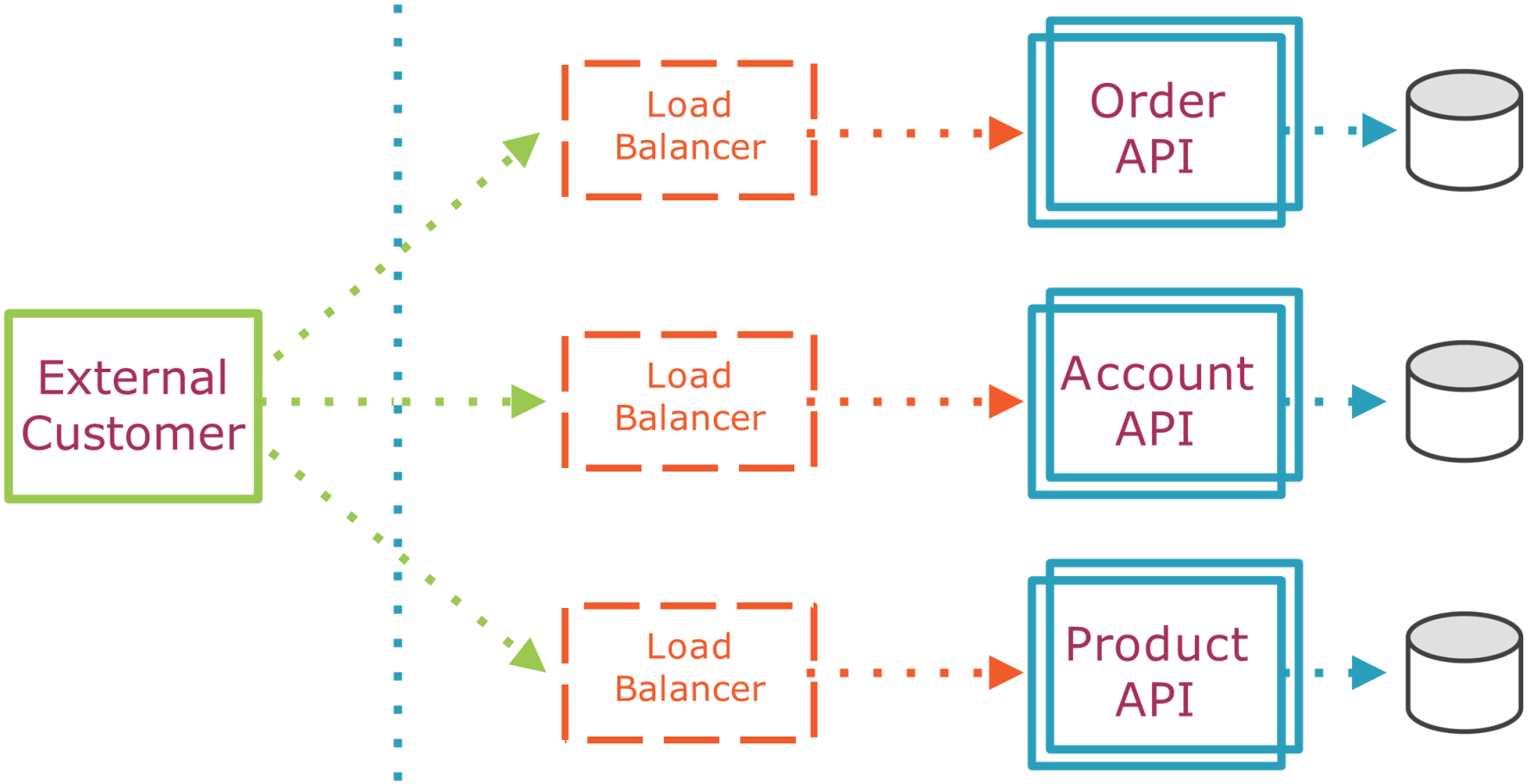
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

¿Cómo consumir los Microservicios?



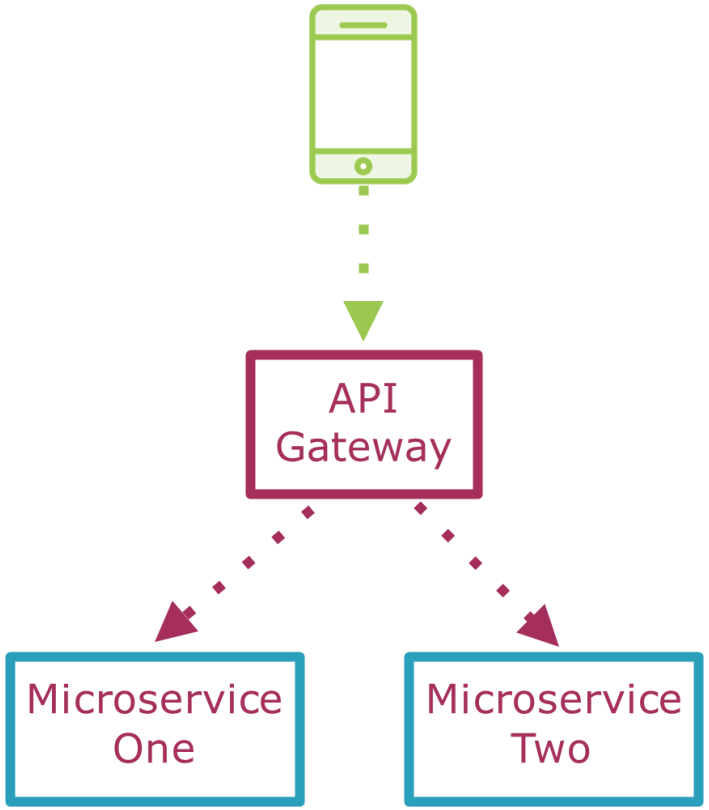
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

¿Cómo consumir los Microservicios?



Centralización de accesos a los microservicios utilizando el patrón API Gateway.

API Gateway



Componente para acceder a las API Una API en sí

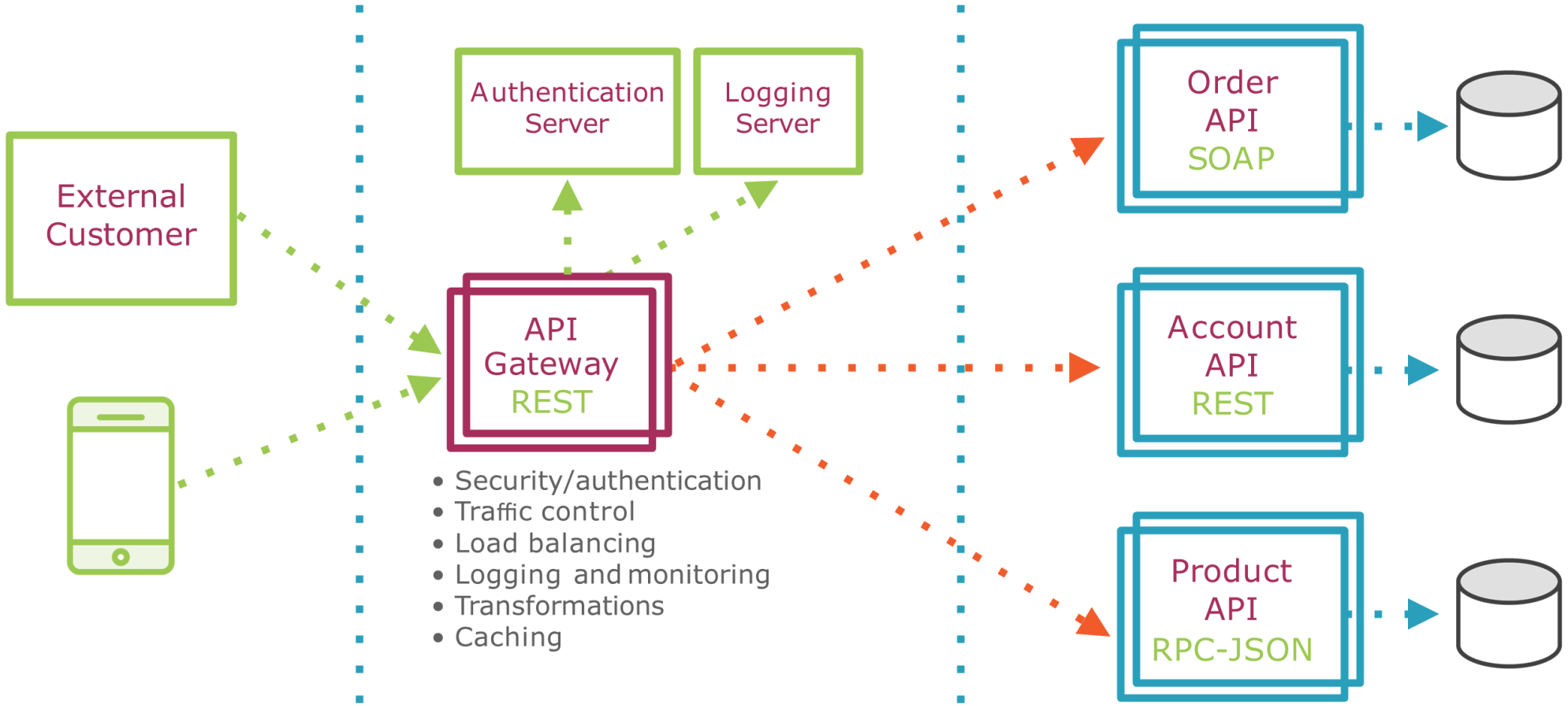
- API RESTFul
- Endpoints

Más que centralizar el acceso

- Seguridad / autenticación
- Control de trafico
- Balanceo de carga
- Registro y seguimiento
- Transformaciones
- Circuit breaker
- Caching

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

API Gateway



Centralización de accesos a los microservicios utilizando el patrón API Gateway.

¿Qué es un API Gateway?

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Puerta de enlace o puerta de acceso: es un punto de entrada por el que debe pasar si desea acceder a un lugar protegido.

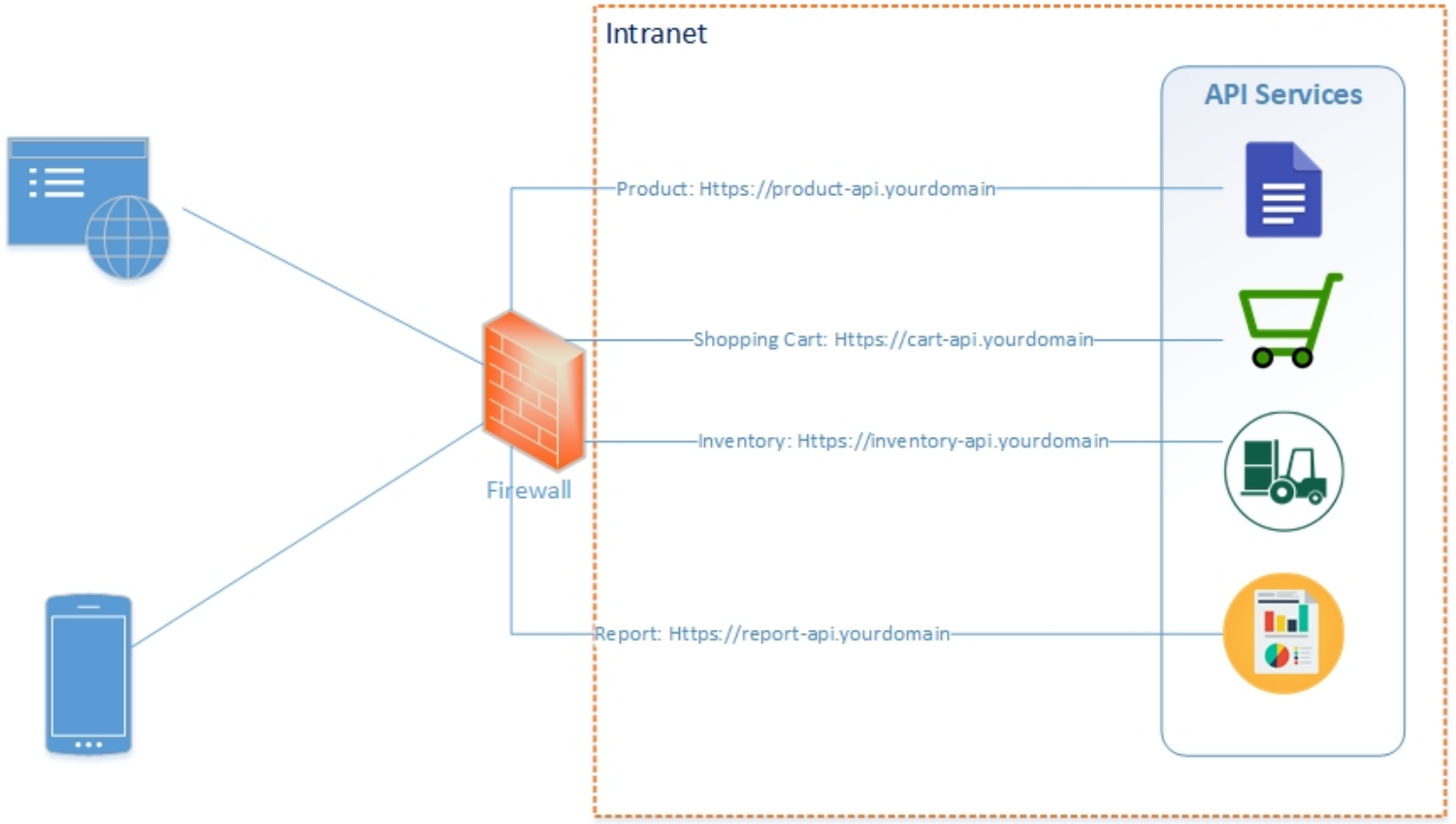
Imaginemos que desea visitar Machu Picchu, un famoso lugar en Cuzco. En primer lugar, tendrá que comprar un boleto y luego pasar por la puerta de peaje (puerta de enlace). En la puerta de peaje, el guardia de seguridad verificará su boleto, si es válido, puede acceder, si no, debe regresar o tener que comprar un boleto válido.

Puerta de enlace o puerta de acceso: es un punto de entrada por el que debe pasar si desea acceder a un lugar protegido.

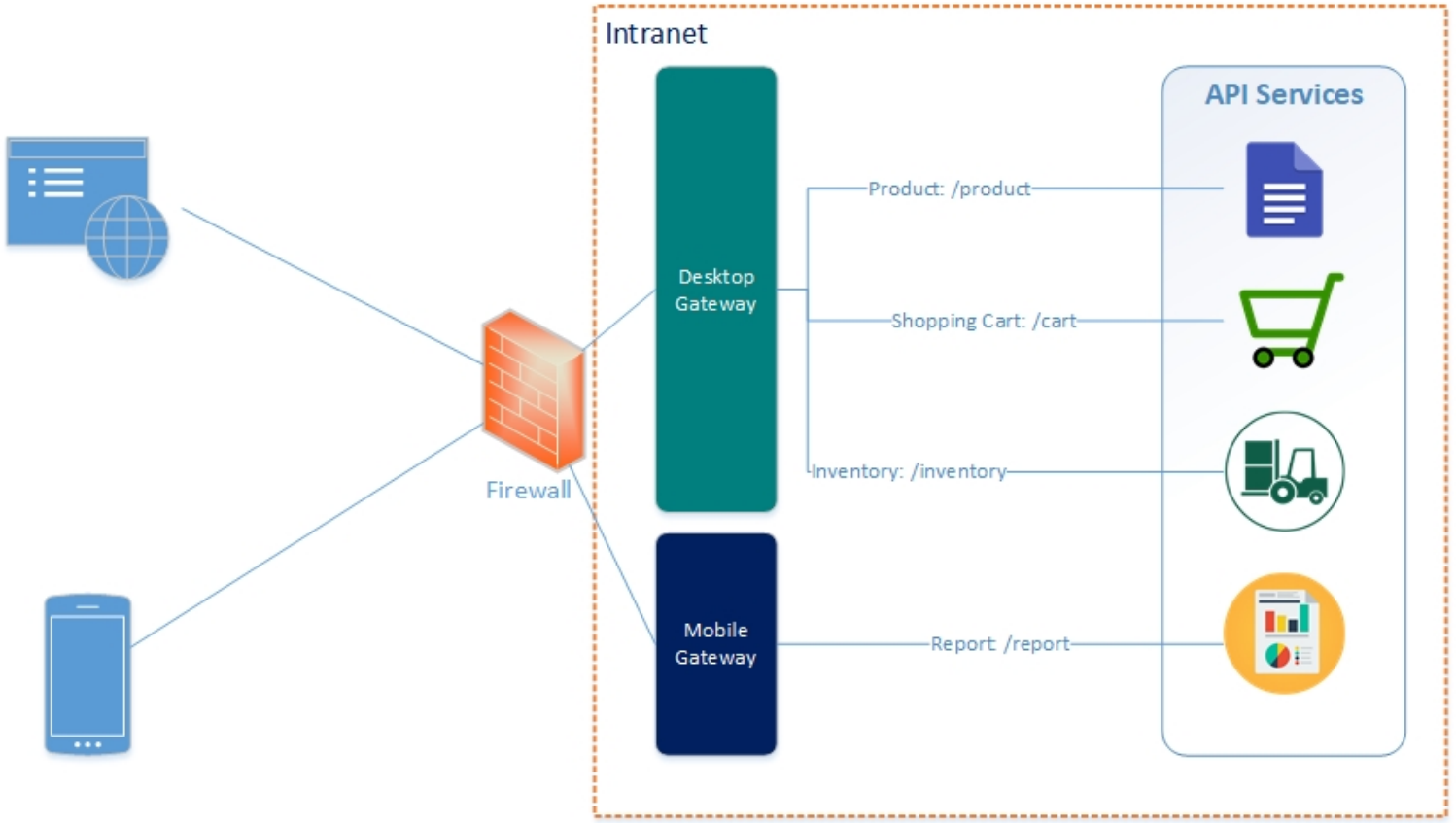
Imaginemos que desea visitar Machu Picchu, un famoso lugar en Cuzco. En primer lugar, tendrá que comprar un boleto y luego pasar por la puerta de peaje (puerta de enlace). En la puerta de peaje, el guardia de seguridad verificará su boleto, si es válido, puede acceder, si no, debe regresar o tener que comprar un boleto válido.

Del mismo modo, un API Gateway es un componente que se encuentra frente a sus API y dentro de una intranet o firewall. La implementación de API Gateway ayudará a garantizar que cada solicitud desde el exterior (internet) tenga que pasar por ella antes de llegar a sus API.

Nota: Podemos tener varias API Gateways en una sola aplicación. Por ejemplo, API Gateway para aplicaciones móviles y API Gateway para aplicaciones de escritorio.



Centralización de accesos a los microservicios utilizando el patrón API Gateway.



Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Ventajas de un API Gateway

El mayor beneficio de un API Gateway es que los clientes solo tienen que hablar con la puerta de enlace en lugar de llamar a una API específica. También simplifica los códigos del cliente.

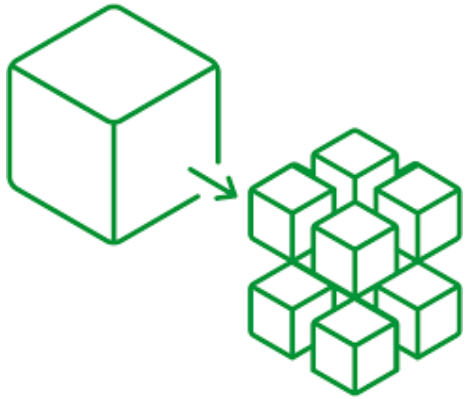
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

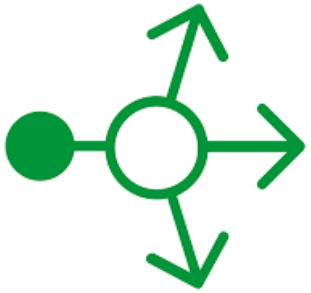
Desventajas de un API Gateway

API Gateway también se convierte en un cuello de botella de desarrollo. Debido a que los desarrolladores tienen que actualizar la API Gateway si desean exponer sus API a externos. Es por eso que el proceso para actualizar API Gateway debe ser lo más liviano posible. De lo contrario, el desarrollador tendrá que esperar en línea para actualizar API Gateway.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Caracterìsticas de un API Gateway



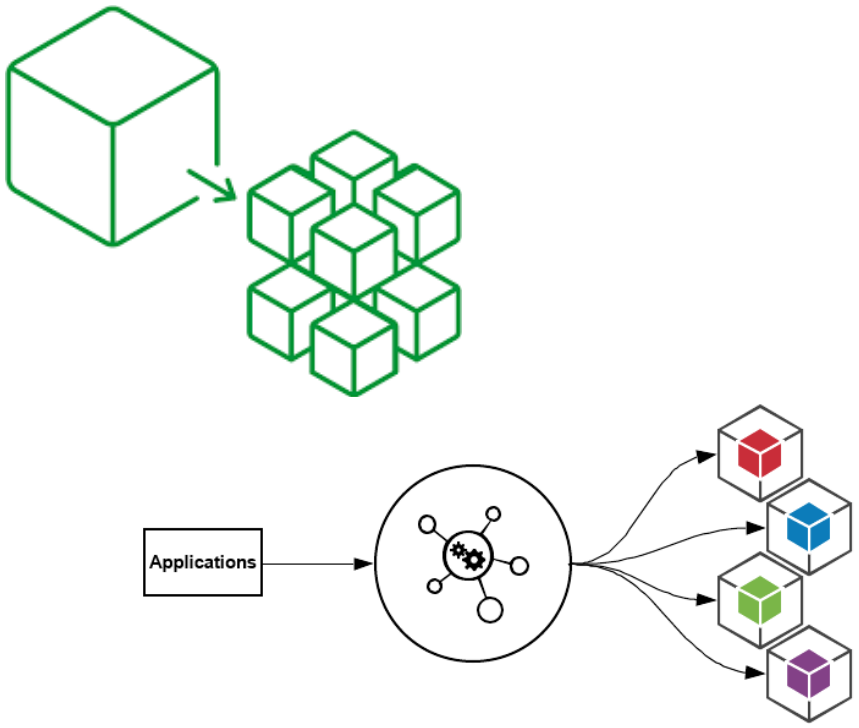


Enrutamiento : Esta es una de las funciones clave de un API Gateway. Básicamente, cuando un API Gateway recibe una solicitud del exterior, reenviará la solicitud a la API interna por correspondencia en función del mapa de enrutamiento.

Esta función es idéntica a la función de proxy inverso proporcionada por servidores web como NGINX.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Características de un API Gateway

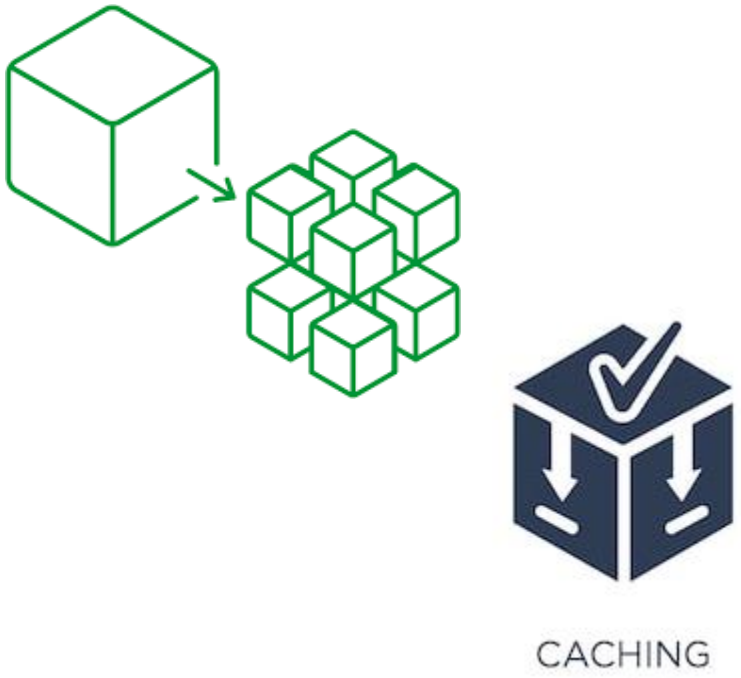


Composición de API : Un API Gateway no solo actúa como proxy inverso, sino que también implementa algunas operaciones de API mediante la composición de API.

Por ejemplo, su aplicación proporciona una función de "Informes", esta función requiere la participación de 3 API's (GetProducts, GetBill y GetInventory) que pertenecen a 3 servicios diferentes: Product, Payment e Inventory. Sin un API Gateway, su cliente tendrá que realizar 3 llamadas a las 3 API's. En su lugar, solo tenemos que implementar una API Gateway utilizando la función de composición de API para exponer solo un endpoint (Informes) que realice las 3 llamadas. Esto ayudará al cliente a recuperar los datos de manera eficiente mediante una sola solicitud al API Gateway.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

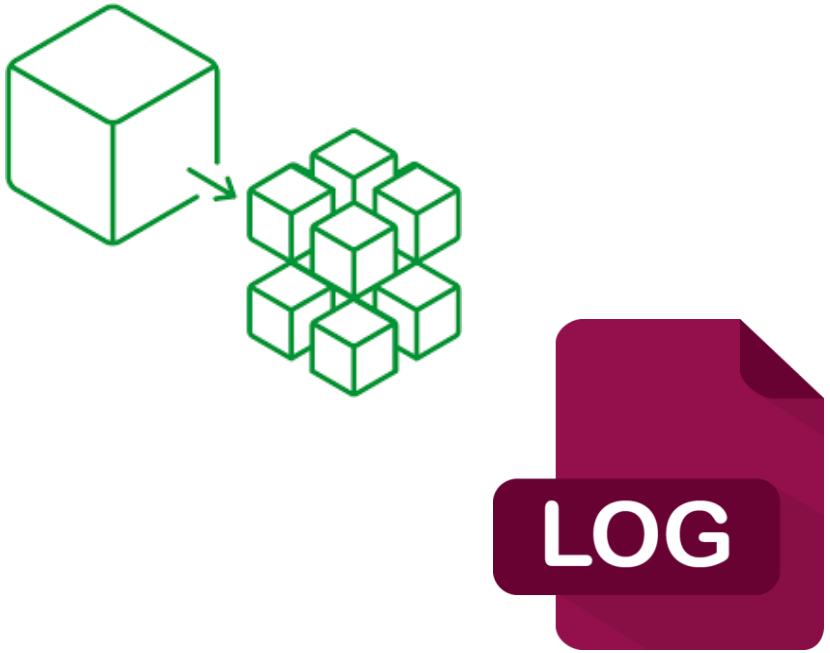
Características de un API Gateway



Almacenamiento en caché : Respuestas de caché para reducir las solicitudes a los servicios.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

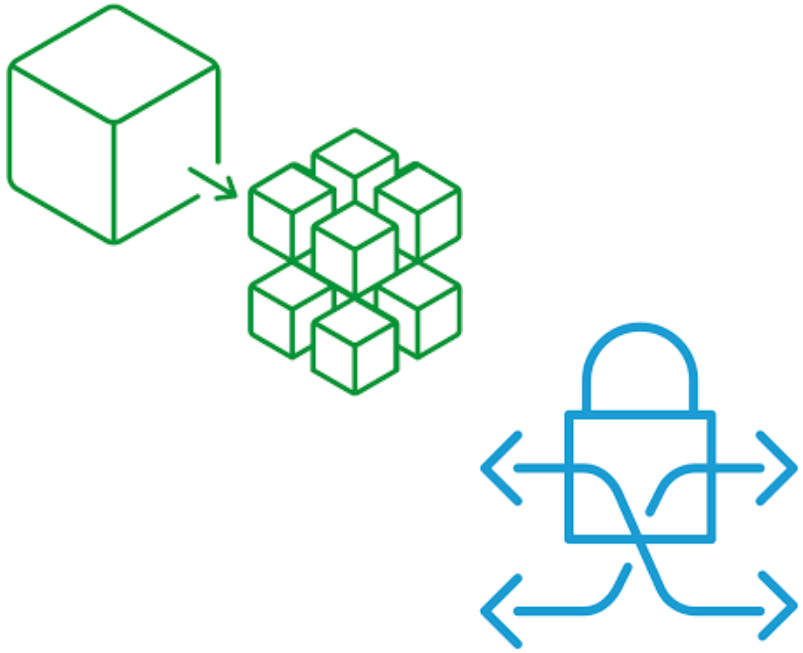
Características de un API Gateway



Logging : la función de logging es necesaria para el seguimiento y la depuración.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

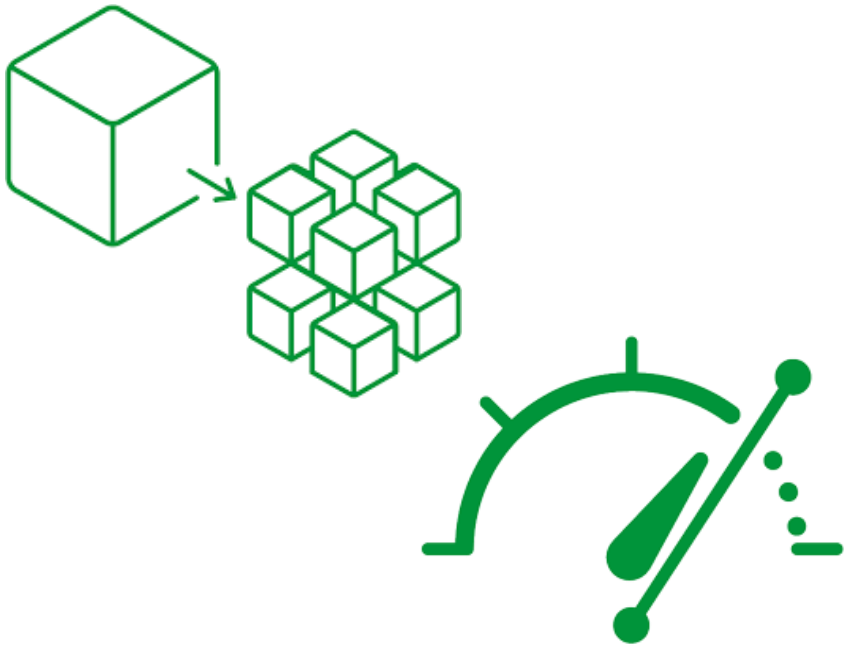
Características de un API Gateway



Integración con proveedores de identidad: En lugar de integrar con Identity Provider para la autenticación y autorización en cada servicio, podemos centralizar esto dentro de un API Gateway.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Características de un API Gateway



Limitación de velocidad : limitar cuántas solicitudes por segundo se permiten en un cliente específico o en todos los clientes.

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Seguridad

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Información sensible

Catalog Service

información no senisble

Ordering Service

Información altamente sensible Necesita cifrado

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Cifrado de datos

Cifrado en tránsito

Utiliza algoritmos estándar para certificados SSL, Transport Layer Security (TLS)

Gestión de certificados

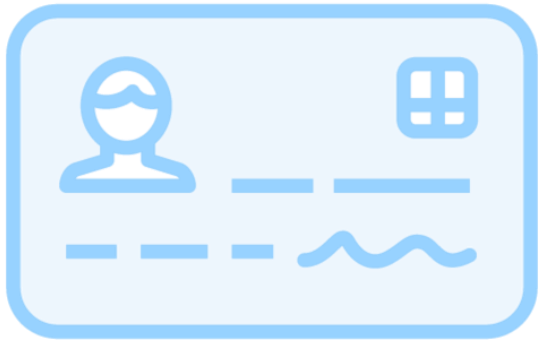
Cifrado en reposo

Cifrado de disco

Gestión declaves

Cifrar copias de seguridad

Centralización de accesos a los microservicios utilizando el patrón API Gateway.



Autenticación

- Necesitamos saber quién llama a nuestro servicio.

Opciones de autorización HTTP

Username & password

"Basic authentication"

Acceso de clientes

Requiere almacenamiento de contraseña

API key

Key per client

Key management

Client certificate

Criptografía de clave pública

Gestión compleja

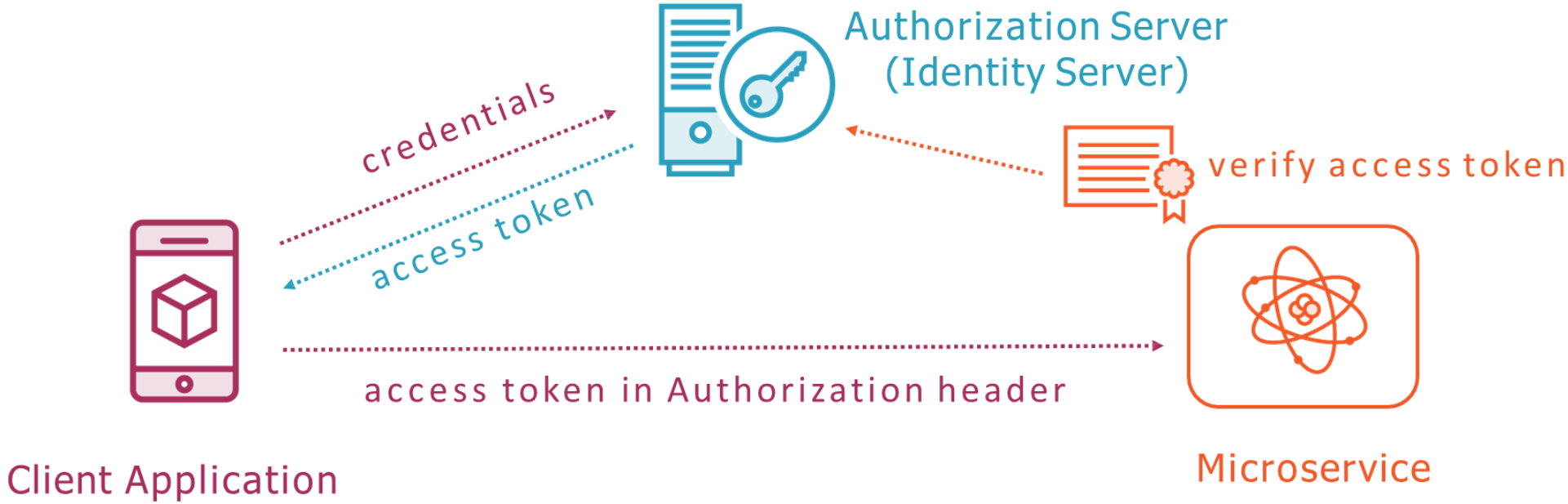
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Usar un servidor de identidad

Use industry-standard protocols:

OAuth 2.0 & OpenID Connect

IdentityServer4



Centralización de accesos a los microservicios utilizando el patrón API Gateway.

OAuth

OAuth trata sobre la autorización y no con la autenticación. La autorización es pedir permiso para hacer cosas. La autenticación consiste en demostrar que eres la persona correcta porque sabes cosas. OAuth no pasa datos de autenticación entre consumidores y proveedores de servicios, sino que actúa como un token de autorización.

Ejemplo; OAuth es la llave de valet para su coche. La llave del servicio de aparcacoches permite que el aparcacoches arranque y mueva el coche, pero no les da acceso al maletero ni a la guantera.

Un token de OAuth es como esa clave de valet. Como usuario, puede indicar a los consumidores qué pueden usar y qué no pueden usar de cada proveedor de servicios. Puede dar a cada consumidor una llave de valet diferente. Nunca tienen la clave completa ni ninguno de los datos privados que les da acceso a la clave completa.

OAuth 1.0 vs. OAuth 2.0

OAuth 2.0 es un rediseño completo de OAuth 1.0, y los dos no son compatibles.

Si crea una nueva aplicación hoy, utilice OAuth 2.0. , ya que OAuth 1.0 está en desuso.

OAuth 2.0 es más rápido y fácil de implementar. OAuth 1.0 utilizó requisitos criptográficos complicados, solo admitía tres flujos y no escalaba.

OAuth 2.0, por otro lado, tiene seis flujos para diferentes tipos de aplicaciones y requisitos y habilita los secretos firmados a través de HTTPS.

Los tokens de OAuth ya no necesitan cifrarse en los puntos de conexión en 2.0, ya que se cifran en tránsito.

OpenID Connect

OAuth y los diferentes flujos que tiene no es perfecto y que tiene carencias frente a algunas necesidades. Por ejemplo:

- Solo es un framework de autorización
- No es capaz de identificar a los usuarios

OpenID Connect está pensado para la autenticación y OAuth para la autorización. Este añade las siguientes funcionalidades que complementan a OAuth:

- Un ID token que nos permite saber quién es el usuario.
- Un nuevo endpoint, **UserInfo**, que nos permite recuperar más información del usuario.
- Un conjunto de scopes estándar.
- Un conjunto de claims que nos permite obtener datos del sujeto.

Autorización

Autenticación: ¿quién llama?

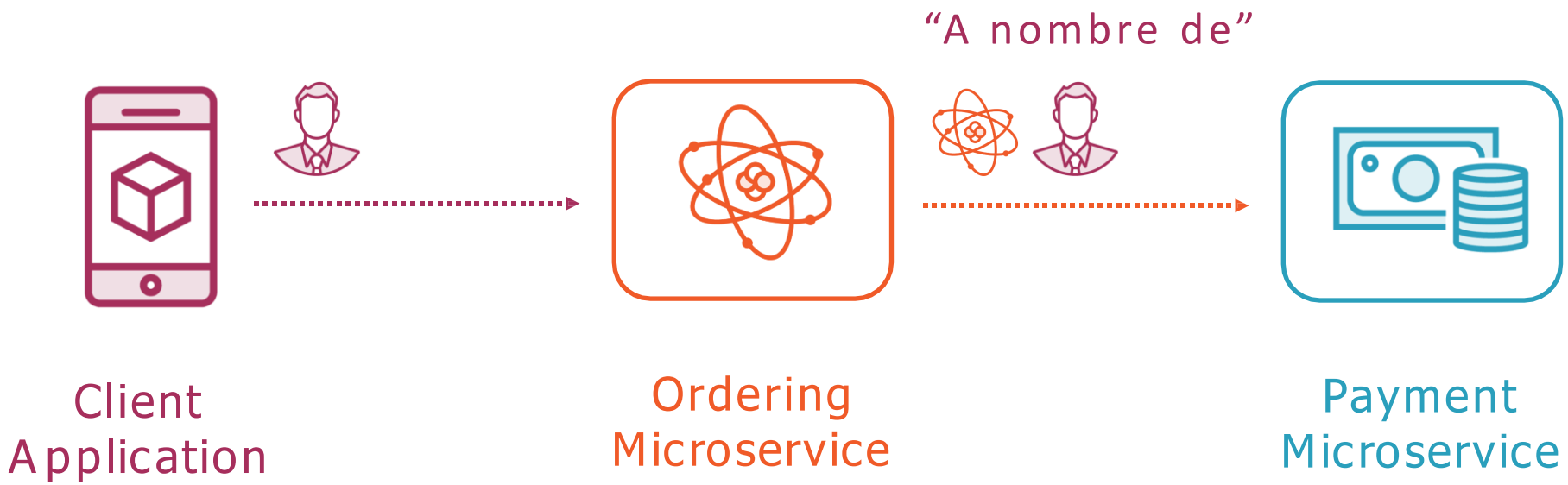
Autorización: ¿qué pueden hacer?

- p.ej. Puedo ver mis ordenes
- No se me debería permitir ver tu pedidos

Frameworks de autorización

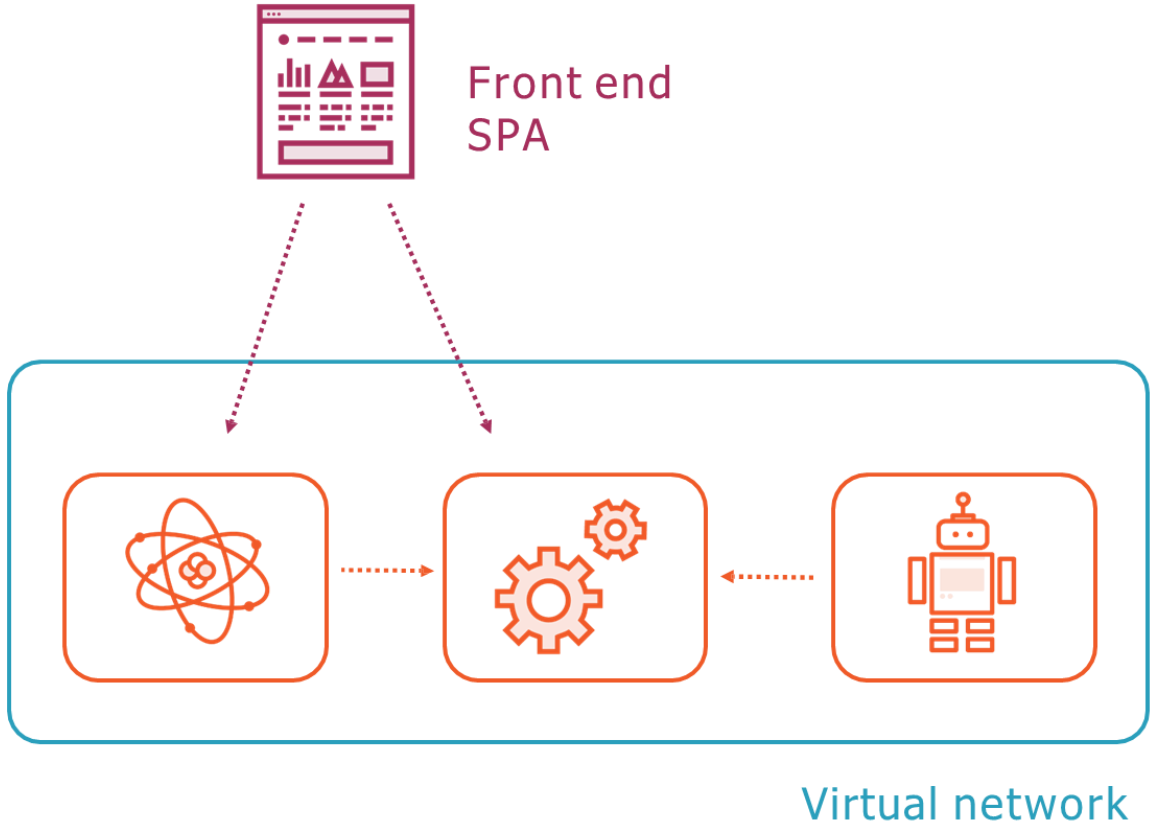
- Puede tomar decisiones basadas en "roles"
- Considere detenidamente qué se debe hacer.

Confused Deputy



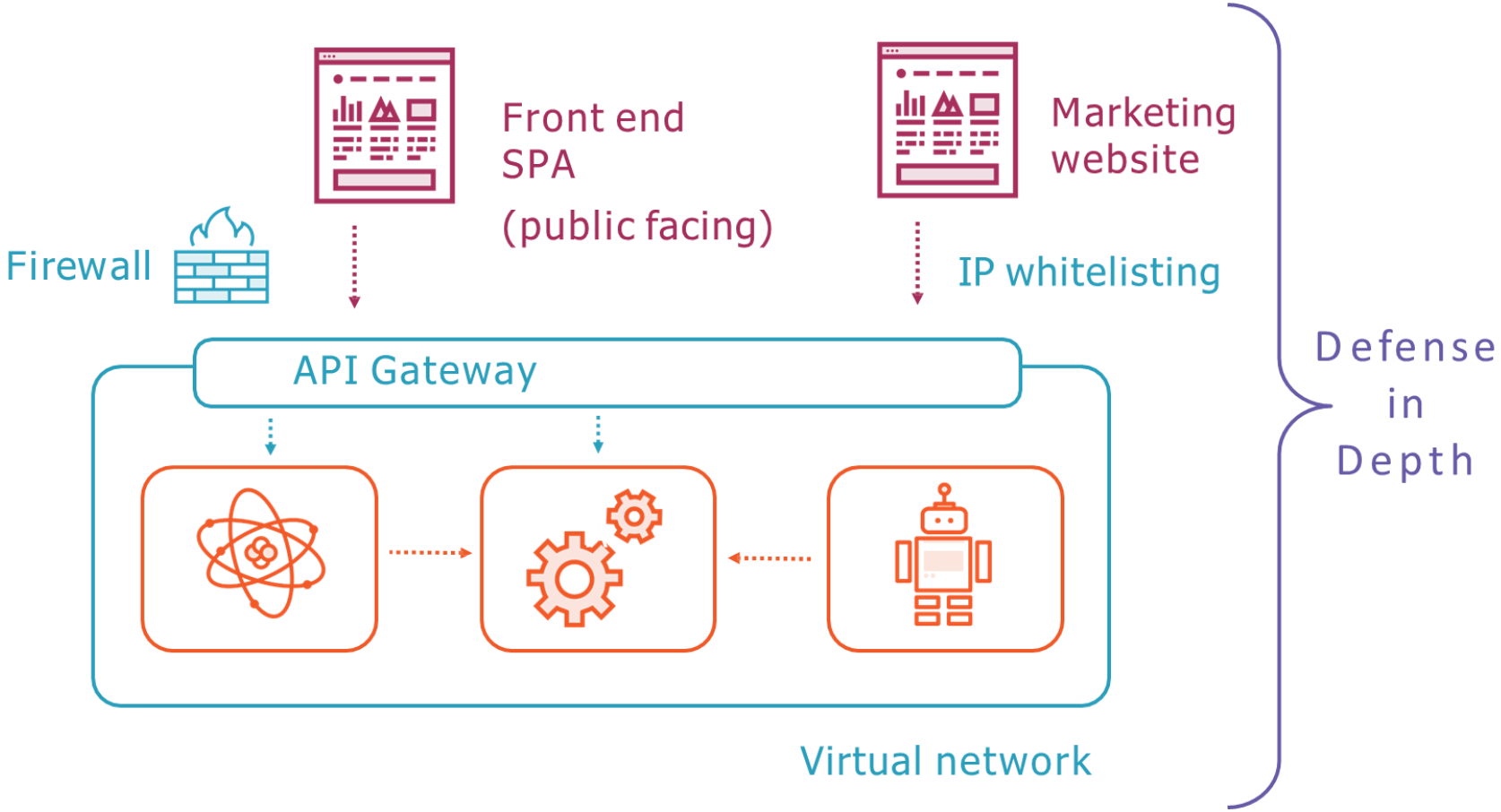
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Seguridad de la Red



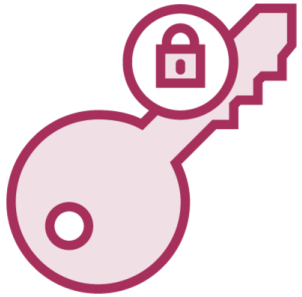
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Seguridad de la Red



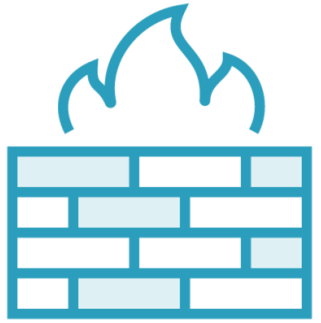
Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Defense in Depth



Cifrado en transito

Token de acceso



Seguridad de red

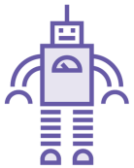
No confíe en una sola técnica

Aplique múltiples capas de protección

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

Medidas defensivas adicionales

Pruebas de penetración ... obtener ayuda de los expertos

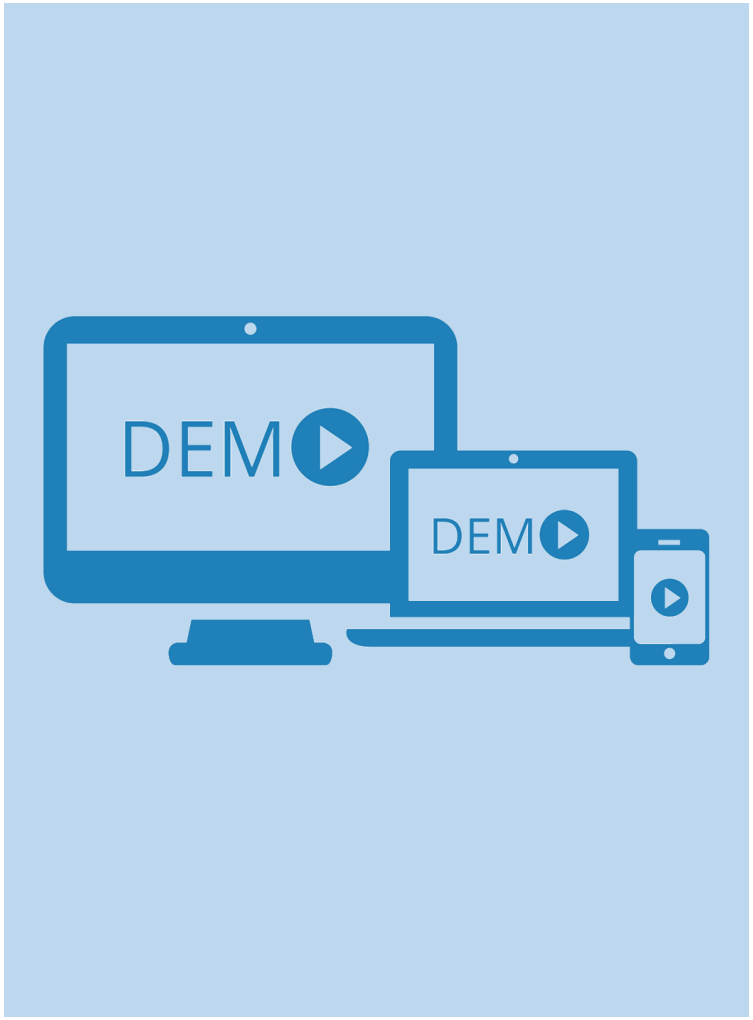


Pruebas de seguridad automatizadas ... demuestre que sus API rechazan a las llamadas no autorizadas

Detección de ataques ... reaccionar rápidamente cuando se está bajo ataque

Auditoría ... sepa exactamente quién hizo qué y cuándo

Centralización de accesos a los microservicios utilizando el patrón API Gateway.



Construyendo .NET Core API Gateway con Ocelot

Centralización de accesos a los microservicios utilizando el patrón API Gateway.

GRACIAS

POR SU PREFERENCIA

GALAXY
TRAINING