

Arquitectura serverless para una API de actualizaciones remotas de SIMS/eSIMS

Tabla de contenidos

Introducción

- AWS Lambda
- API Gateway
- Application Load Balancer
- ALB Vs API Gateway

Arquitectura serverless

Desarrollo aplicación serverless

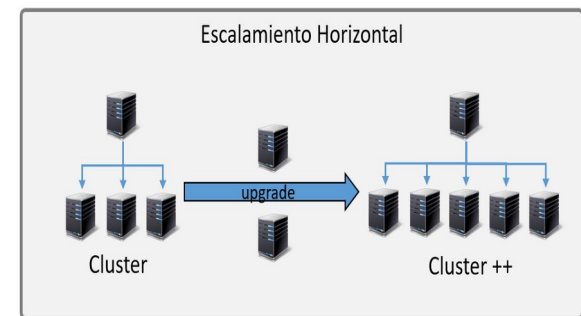
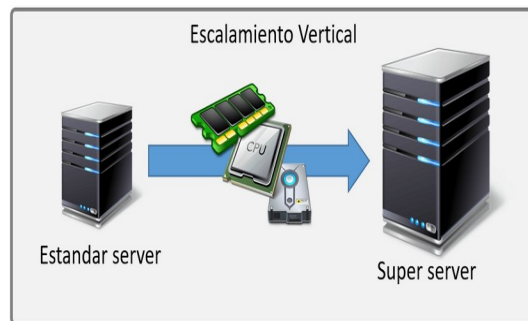
- NodeJS
- Express
- Conclusión

RSP API Proxy



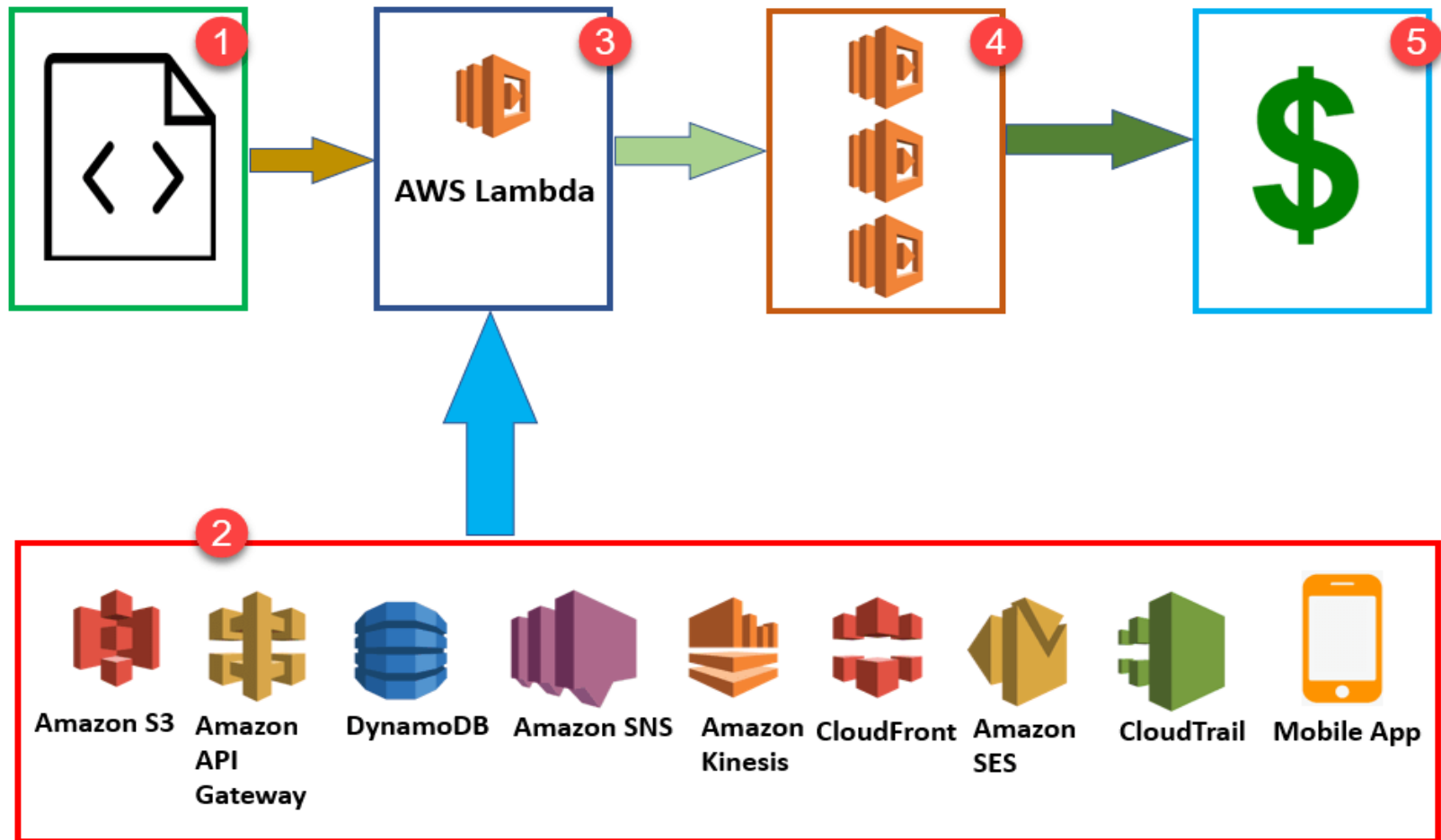
crononauta

Introducción



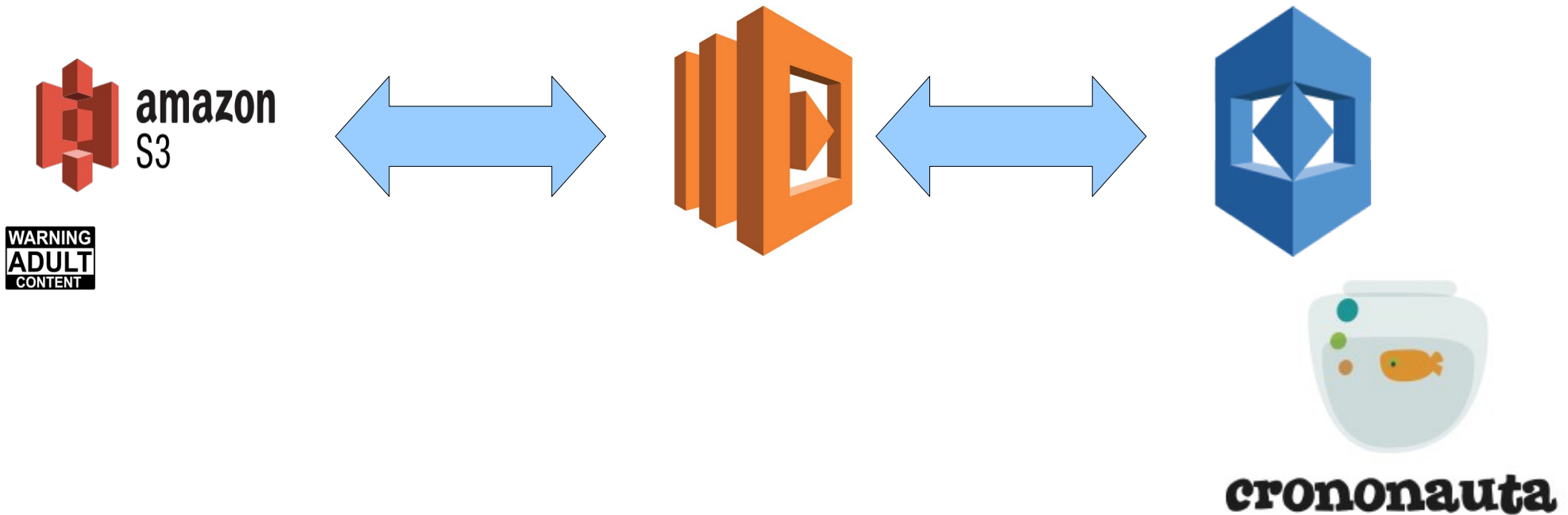
crononauta

Introducción: AWS Lambda

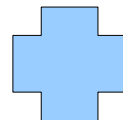
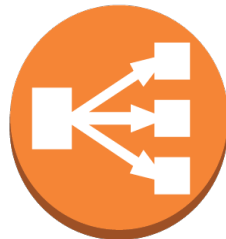
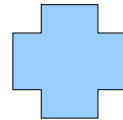


crononauta

Introducción: AWS Lambda: Ejemplo S3



Introducción: AWS Lambda: Ejemplo Backend

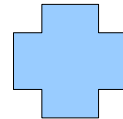
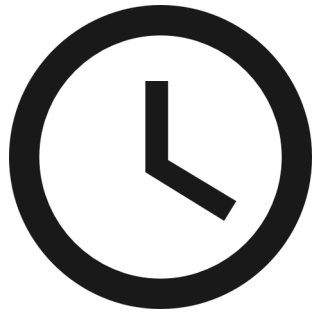


crononauta

Introducción: AWS Lambda: Crons



CRON JOB



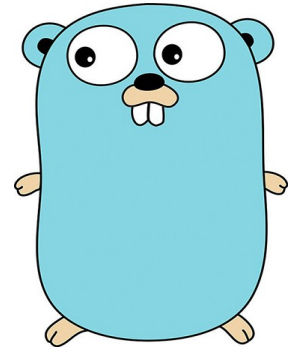
crononauta

Introducción: AWS Lambda: Métricas en Cloudwatch



crononauta

Introducción: AWS Lambda: Lenguajes compatibles



crononauta

Introducción: API Gateway

- Desplegar
- Administrar
- Proteger
- Monitorizar
- Escalar



crononauta

Introducción: Application Load Balancer

- Desplegar un balanceador de carga
- Monitorizar la API
- Escalar la API



crononauta

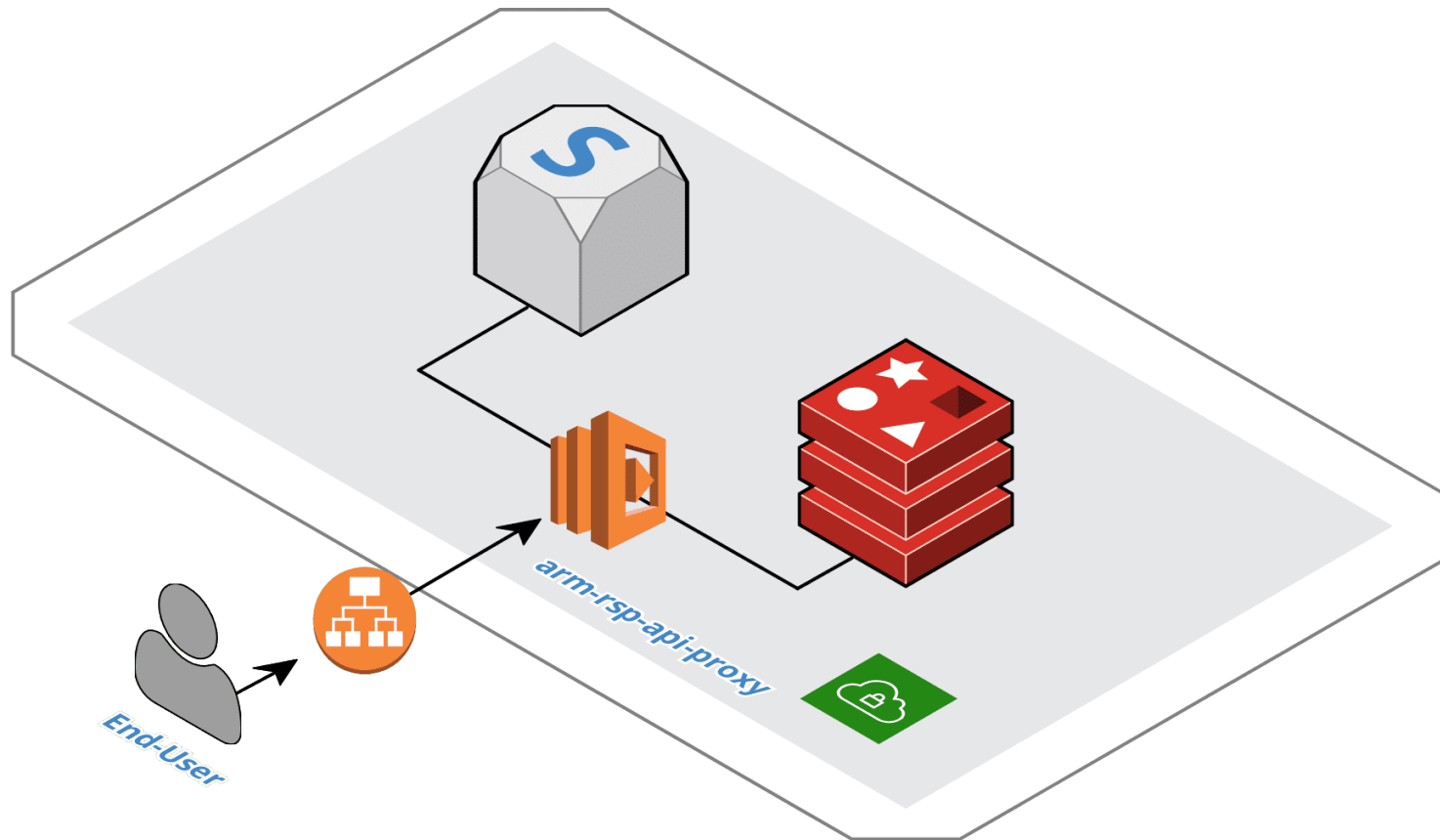
Introducción: ALB Vs. API Gateway

- ALB ofrece un **precio más competitivo.**
- API Gateway ofrece opciones de seguridad.
- ALB permite utilizar certificados SSL propios.
- ALB permite enrutar tráfico hacia un servidor.



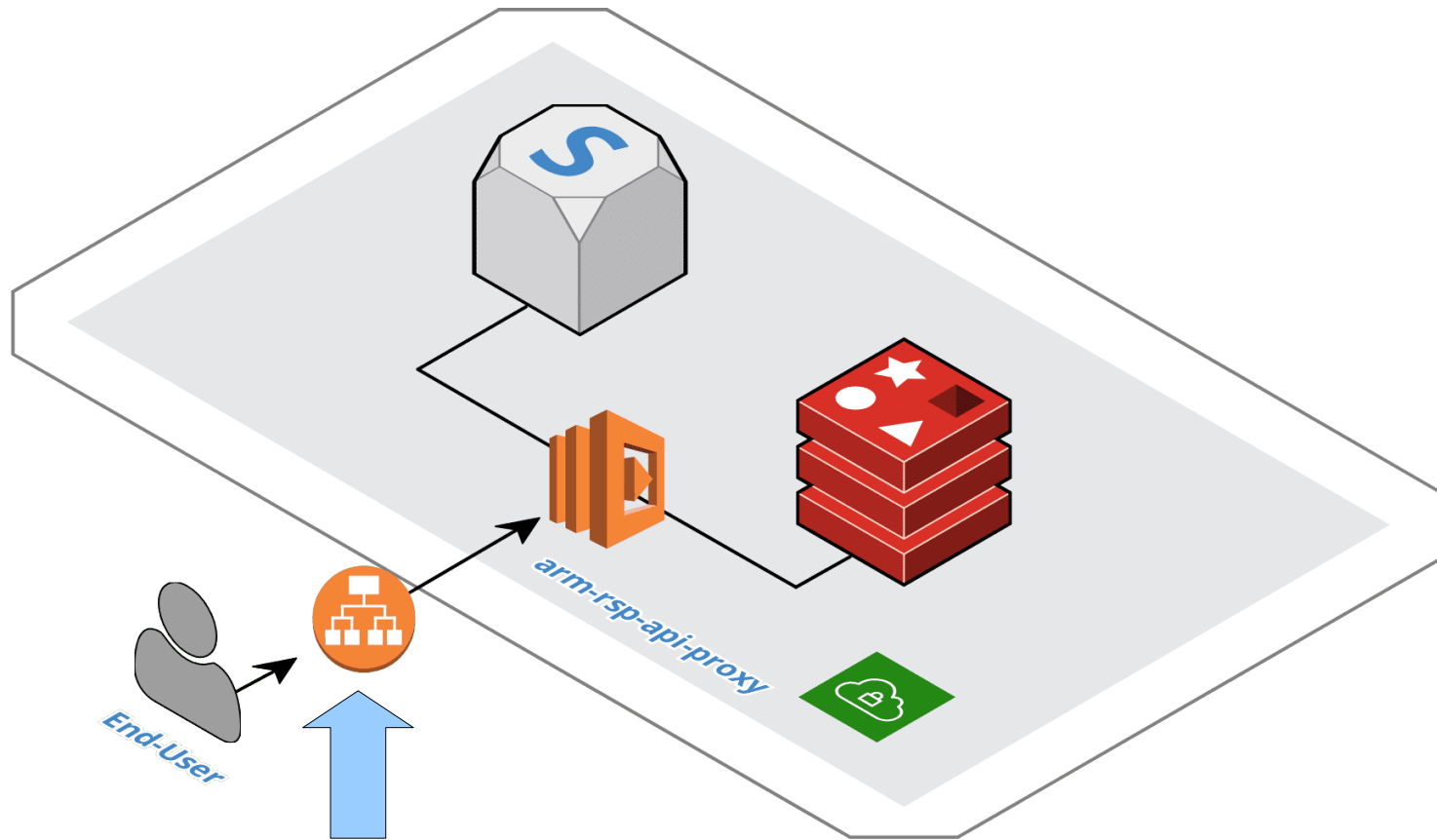
crononauta

RSP API: Arquitectura serverless



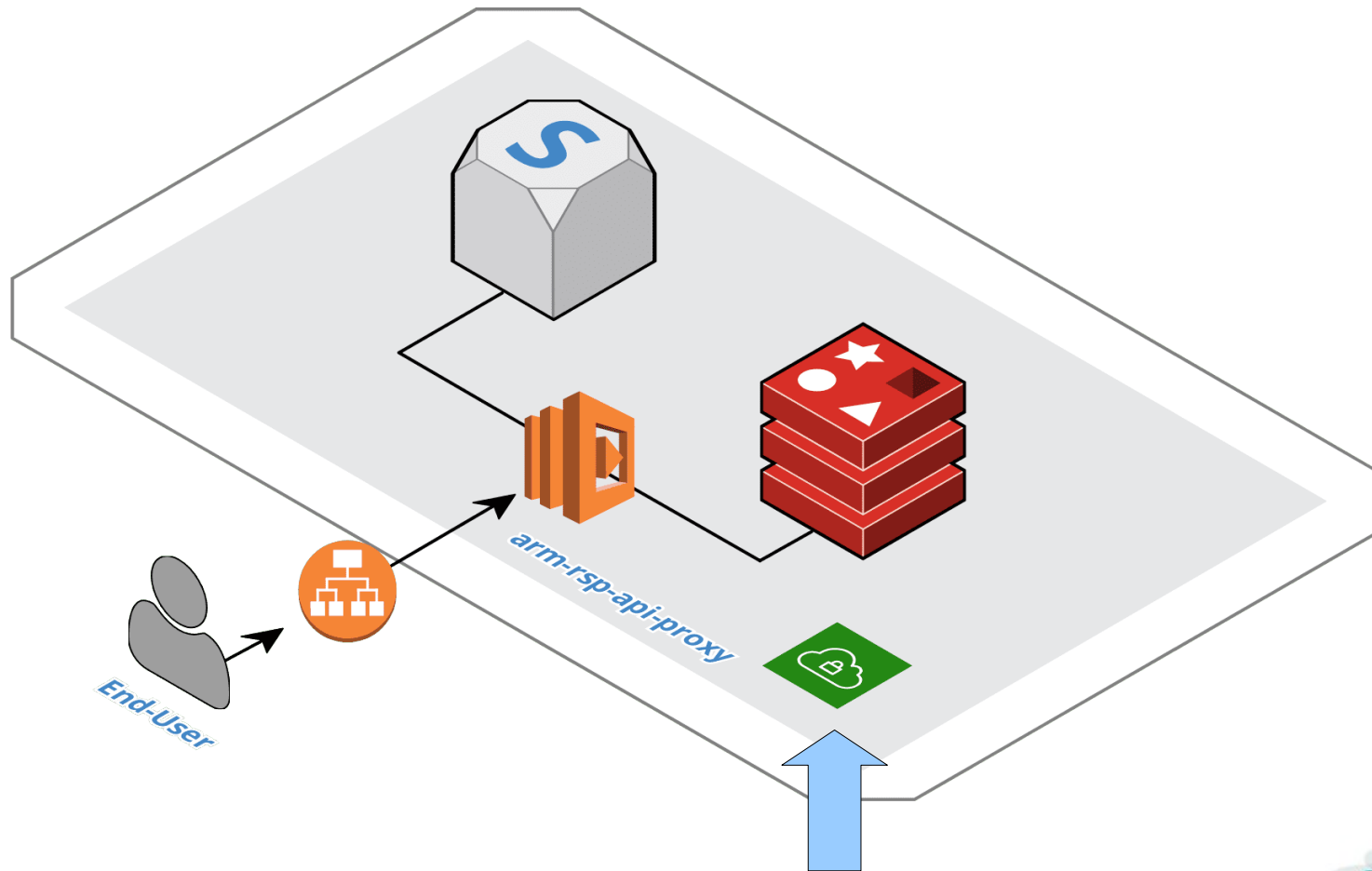
crononauta

RSP API: Arquitectura serverless: ALB



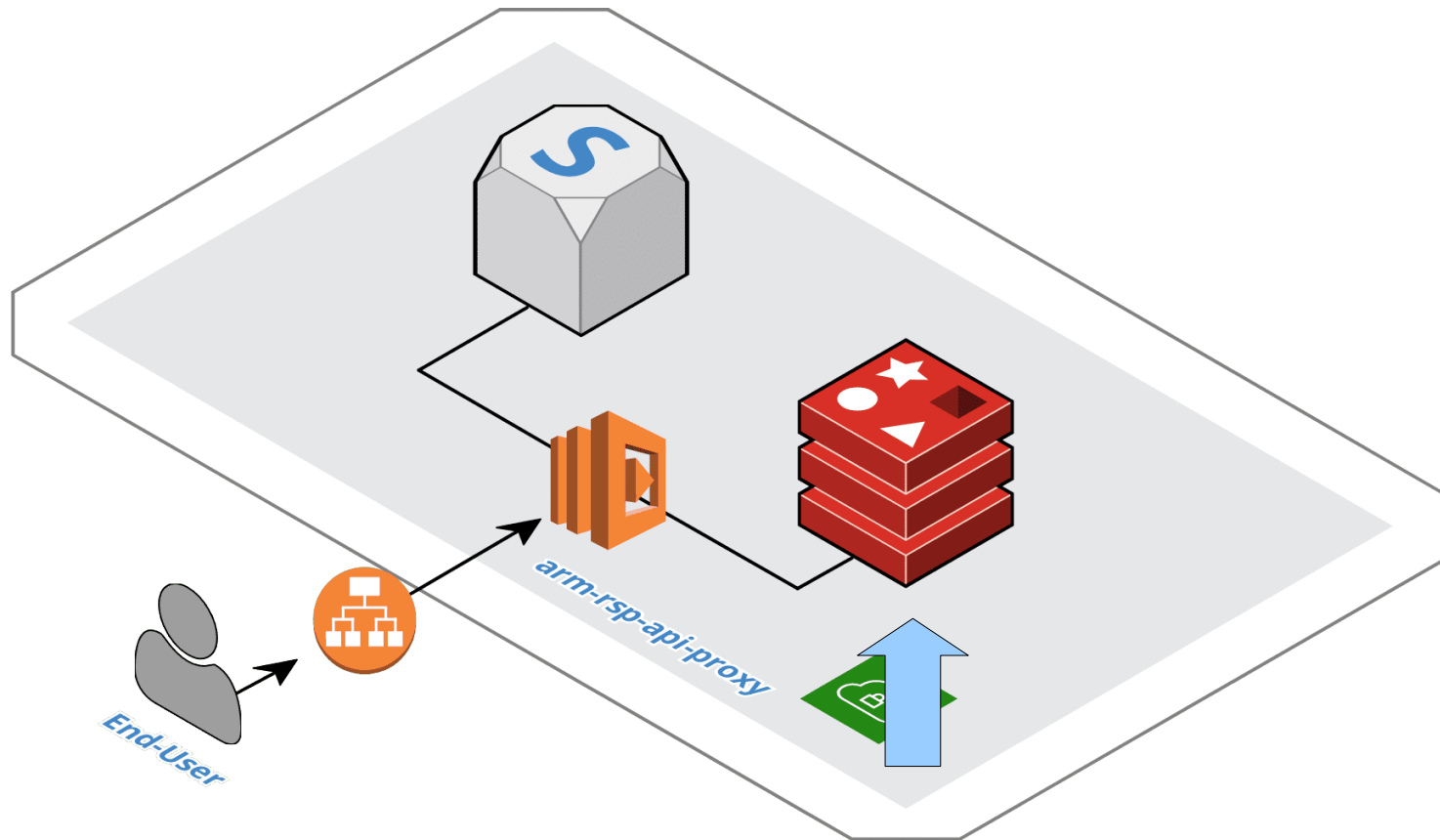
crononauta

RSP API: Arquitectura serverless: VPC



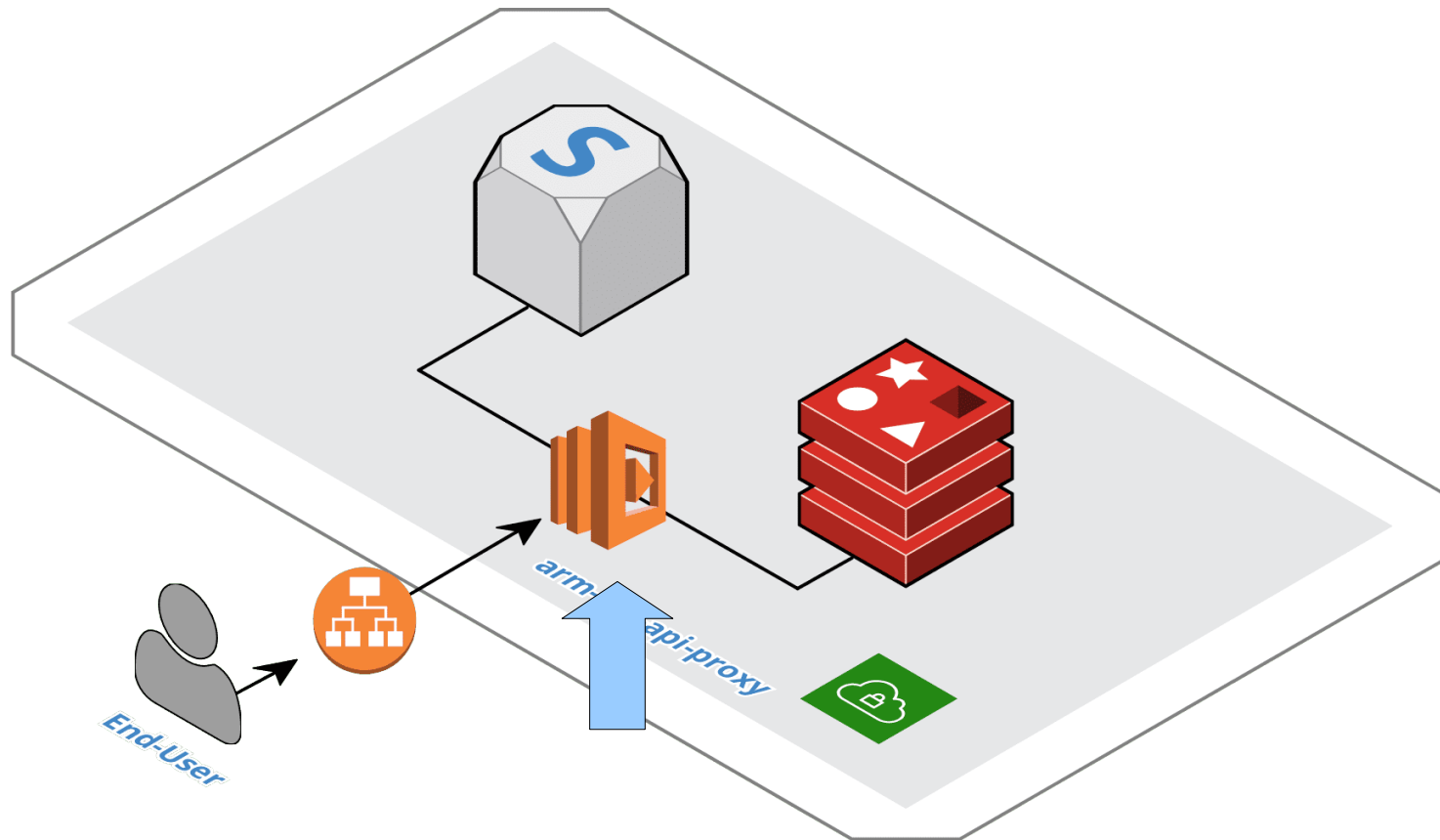
crononauta

RSP API: Arquitectura serverless: Elaticache



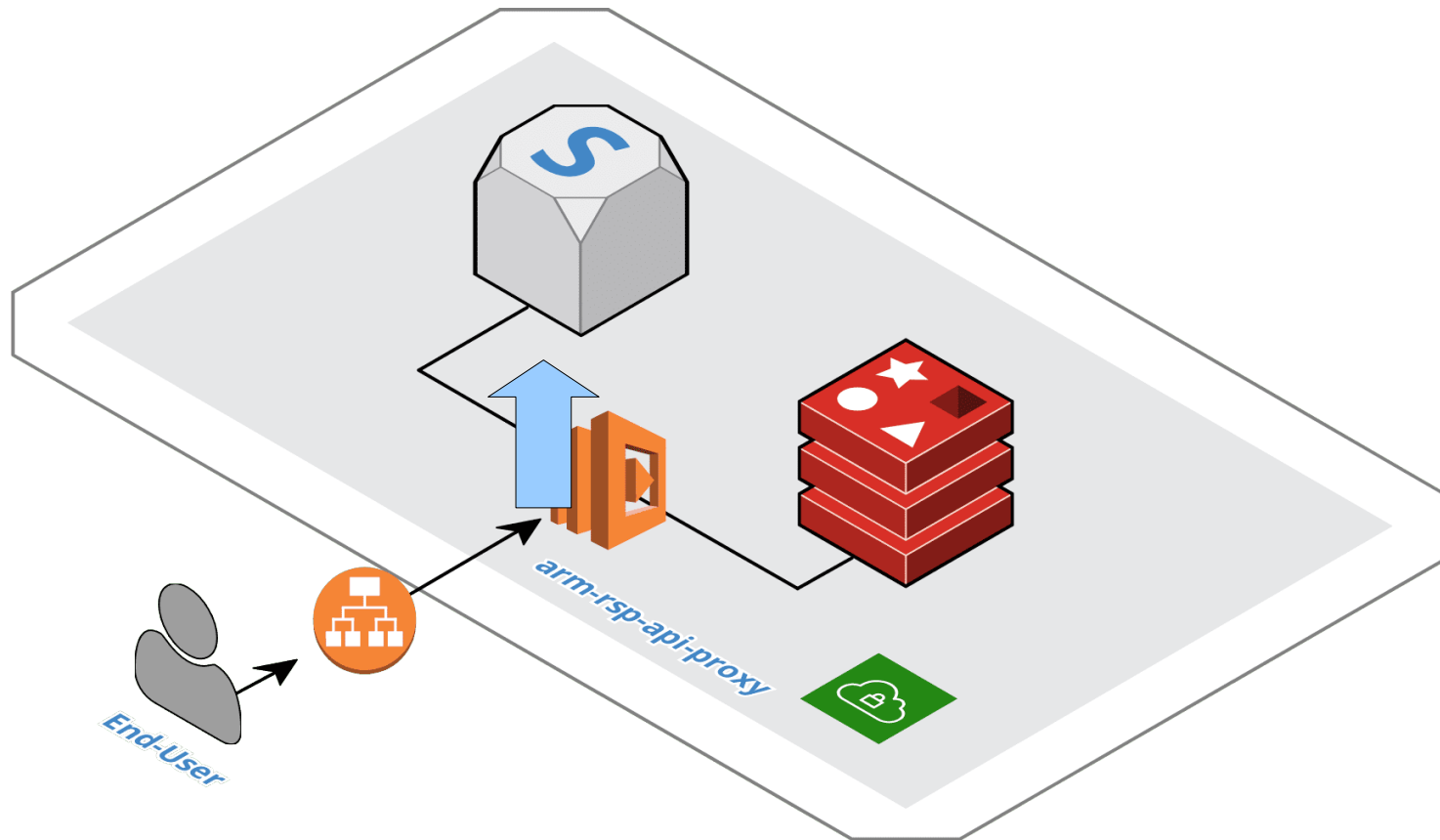
crononauta

Arquitectura serverless: Lambda



crononauta

Arquitectura serverless: RDS



crononauta

Desarrollo de una aplicación serverless

¿Qué lenguaje de programación elijo?

¿Cómo desarrollo eficientemente?

¿Estoy creando una dependencia tecnológica?

¿Va a ser mi aplicación segura?

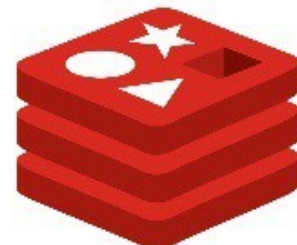
¿Va a estar mi aplicación disponible 24x7?



crononauta

NodeJS

node JSTM
express

 redis
node JSTM

node JSTM

MySQL[®]


crononauta

NodeJS: Express

Infraestructura web rápida, **minimalista** y **flexible** para Node.js.

Facilita la **creación de APIs** (SOAP o REST) a través de NodeJS.

aws-serverless-express: Permite **migrar una aplicación a lambda sin** necesidad de **cambios en el código** *no serverless*



crononauta

NodeJS: Conclusión

Express nos permite trabajar directamente desde local (O un entorno de desarrollo), sin tener que pasar por AWS Lambda

Al desarrollar nuestra API con express y **aws-serverless-express** estamos exentos de cualquier dependencia tecnológica



crononauta

RSP API Proxy

- Es una API Soap que se encarga de administrar los perfiles instalados en las eSIM
- Implementa el estándar de la GSM
- Permite desactivar, borrar, activar, descargar y auditar una eSIM conectada a la red



- Es un servicio que **puede ser llamado en cualquier momento, pero con poca frecuencia.**
- Puede recibir **muchas llamadas simultaneamente** cuando se activa un lote de tarjetas.
- La aplicación debe de **poder escalar.**
- Centrado en la seguridad.
- Sólo el ALB es accesible desde fuera del VPC.



RSP API Proxy: Seguridad

- Protección de fuerza bruta a nivel de nombre de usuario al iniciar sesión.
- Protección frente *timing attack* al iniciar sesión.
- Limitación de recursos, tarjetas, perfiles... por usuario.
- Servicios restringidos al VPC.



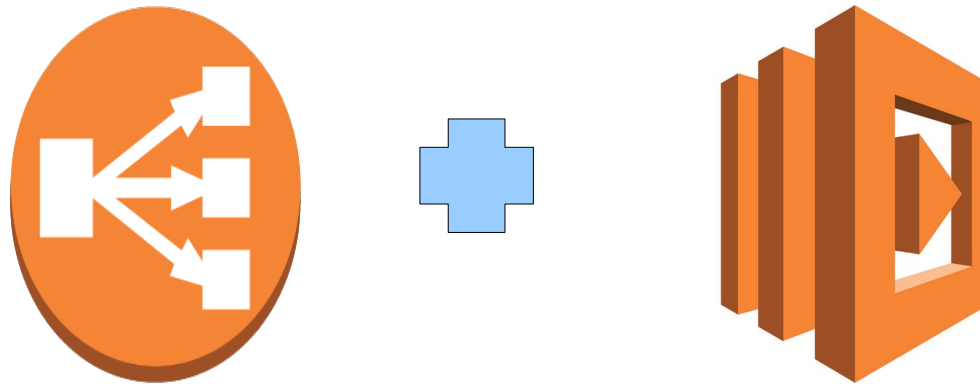
crononauta

RSP API Proxy: Fácil desarrollo



crononauta

RSP API Proxy: Escalabilidad & Disponibilidad



crononauta

RSP API Proxy: Monitorización



crononauta

¿Dudas? ¿Preguntas?

<https://crononauta.com/>
<https://www.podgroup.com/>

Javier Carranza

javier.carranza@crononauta.com

twitter: @trunks

José Carlos García

jose.garcia@crononauta.com

twitter: @josegarciaor

José Félix Ontañón

felix.ontanon@podgroup.com

twitter: @fontanon



crononauta