Desafío Técnico: Arquitectura y Estrategia DevSecOps

1. Contexto del Escenario

La empresa "CloudCorp" está lanzando una nueva iniciativa tecnológica y requiere aprovisionar una infraestructura de nube segura, escalable y automatizada desde cero.

La organización tiene dos Unidades de Negocio (OUs) distintas que deben operar con independencia pero bajo estándares de gobernanza compartidos:

1. **Unidad SM (Sales & Marketing)**
2. **Unidad AY (Analytics & Yield)**

Cada unidad requiere tres entornos aislados:

- **Desarrollo (Dev)**
- **Pruebas de Aceptación de Usuario (UAT)**
- **Producción (Prod)**

2. El Reto

Su misión es diseñar, aprovisionar y documentar la estrategia DevSecOps para soportar esta estructura. Buscamos un enfoque de "Shift-Left Security", donde la seguridad es parte intrínseca de la infraestructura y el despliegue.

Requisitos Técnicos Mandatorios

A. Infraestructura como Código (IaC)

- Debe utilizar una herramienta de IaC (Terraform, OpenTofu, AWS CDK o Pulumi).
- El código debe ser modular y reutilizable (ej. uso de módulos para VPCs, Clusters, RDS).
- **Reto de Arquitectura:** Debe proponer una estrategia de aislamiento. ¿Usará una cuenta de AWS por entorno? ¿VPCs separadas? ¿AWS Organizations? Justifique su decisión basándose en seguridad vs. costos.

B. Pipeline CI/CD & DevSecOps

Debe crear un pipeline de despliegue (puede ser un "Skeleton" o "Hello World") que incluya las siguientes etapas de seguridad explícitas:

1. **SCA (Software Composition Analysis):** Detección de vulnerabilidades en librerías.
2. **SAST (Static Application Security Testing):** Análisis estático de código.
3. **Secret Scanning:** Detección de credenciales hardcodeadas.
4. **IaC Scanning:** Análisis de seguridad sobre el código de infraestructura (ej. detectar buckets S3 públicos o Security Groups abiertos al mundo).

C. Observabilidad y Gobierno

- Explique cómo centralizará los logs y métricas de ambas unidades (SM y AY).
- Proponga una estrategia de acceso (IAM) para que los desarrolladores de "SM" no puedan tocar recursos de "AY".

3. Cláusula de Flexibilidad (Agnosticismo de Nube)

Sabemos que el talento DevSecOps a veces se especializa en una nube específica (Azure, GCP) o prefiere entornos locales para pruebas rápidas.

- **Preferencia:** AWS.
- **Opción Flexible:** Si usted no es experto en AWS, puede realizar la implementación técnica en Azure, GCP o utilizando LocalStack/Contenedores.
- **Condición Obligatoria:** Si elige NO usar AWS para el código, **debe incluir una sección en su documentación titulada "Mapeo a AWS"**, donde explique conceptualmente qué servicios nativos de AWS utilizaría para reemplazar los que implementó en la otra nube (Ej: *"Usé GitHub Actions, en AWS usaría CodePipeline/CodeBuild"* o *"Usé Azure Monitor, en AWS usaría CloudWatch"*).

4. Entregables

Entregable 1: Repositorio de Código (GitHub/GitLab/Bitbucket)

Debe proveernos acceso a un repositorio que contenga:

- Código de Infraestructura (IaC).
- Archivos de configuración del Pipeline (YAML/Jenkinsfile).
- Un README.md claro sobre cómo ejecutar el proyecto.
- *Nota:* No es necesario desplegar una aplicación compleja, basta con un contenedor Nginx básico o una función Serverless "Hello World" para demostrar el flujo.

Entregable 2: Documento de Diseño Técnico (PDF o Markdown)

Un documento de máximo 3-4 páginas que incluya:

- **Diagrama de Arquitectura:** Mostrando la red, subredes, flujo de datos y herramientas de seguridad.
- **Estrategia de Ramas (Branching Strategy):** Cómo el código fluye desde el commit del desarrollador hasta Producción.
- **Matriz de Herramientas:** ¿Qué herramientas eligió y por qué? (Ej. *"Elegí Trivy sobre Clair porque..."*).

Entregable 3: Video de Sustentación (Loom/YouTube/Drive)

Un video corto (máximo 10 minutos) donde usted:

1. Muestre el diagrama y explique su diseño.
2. Navegue brevemente por la estructura de su código (muestre la modularidad).

3. Explique cómo su solución mitiga riesgos de seguridad comunes.
4. (Si aplica) Explique el mapeo conceptual a AWS.

5. Criterios de Evaluación

Evaluaremos su prueba bajo los siguientes pilares:

Pilar	Peso	Qué buscamos
Seguridad (Sec)	30%	Implementación correcta de escaneos, IAM Least Privilege, cifrado en reposo/tránsito.
Arquitectura (Ops)	30%	Diseño de red, Alta Disponibilidad (HA), separación lógica de las unidades SM y AY.
Automatización (Dev)	20%	Calidad del IaC (DRY), limpieza del pipeline, facilidad de replicación.
Comunicación	20%	Capacidad para "vender" y defender sus decisiones técnicas en el video y documento.

Puntos Extra (Bonus)

- Implementación de **Cost Management** (Alertas de presupuesto o FinOps).
- Uso de **GitOps** (ArgoCD o Flux).
- Estrategia de **Disaster Recovery**.

¡Mucho éxito! Esperamos ver su propuesta.