

Lectura 2:

HTTP, DNS, TLS y herramienta de diagnostico (curl, dig, ss, openssl)

- La disponibilidad depende de como resolvemos los nombres (DNS), como transportan y protegen el trafico (TLS) y como se define contratos y políticas de cache (HTTP)

HTTP: contrato, rendimiento y gobernanza operativa

- HTTP es el idioma de la mayoría de microservicios
- Los tiempos de respuesta, códigos de estado y encabezados definen el contrato
- Get y put son idempotentes a diferencia de post, patch, delete.

* Versiones : * HTTP/1.1 : conexiones persistentes y pipelining limitado

* HTTP/2 : multiplexación real sobre una sola conexión, también mejor utilización de ancho de banda y latencia efectiva menor.

* HTTP/3 : mejora ante pérdida de paquetes; útil en redes inestables o móviles

Cache y Control de tráfico : Para APIs, el cache suele focalizarse en GET.

Observabilidad : Logs de acceso (método ruta, código, latencia, tamaño de respuesta)

DNS : la capa de descubrimiento y resiliencia

- DNS mapea nombres a direcciones IP
- Una entrada mal configurada o una TTL inadecuada puede bloquear por completo un despliegue o hacer imposible un rollback rápido

- △ La estrategia TTL condiciona la velocidad de propagación de cambios
- △ TTL cortas agilizan **rollbacks** pero aumentan consultas y latencia
- △ TTL largas ahorran recursos pero endurecen la reversion
- △ TTL Time of live, el proposito es que los datos no circulen indefinidamente en una red.

Registros Clave: **A/AAAA** para direccionamiento, **CNAME** para alias, **TXT** para verificaciones, **SRV** para descubrimiento de servicios, **NS** para delegación.

Seguridad y control **DNSSEC**: añade firmas de registros aunque no cifra el canal.

TLS: confidencialidad, integridad y autenticación

- Rol en DevSecOps es proteger el canal entre clientes, **edges** y microservicios

Rendimiento y riesgos: evitar **fallbacks** inseguros y claves compartidas entre entornos, usar **certificate transparency** Para vigilar emisiones inesperadas

Herramientas clínicas de red: curl, dig, ss y openssl

□ **Curl**: permite observar las respuestas de un endpoint, es útil para:

- ▲ Confirmar códigos de estado
- ▲ Comprobar negociación
- ▲ Auditar
- ▲ Comparar latencias

□ **dig**: Sirve para interrogar servidores autoritativos o resolvers específicos, comparar respuestas en distintas ubicaciones, útil Para:

- ▲ investigar Porqué un canario no recibe tráfico tras un cambio DNS
- ▲ Validar la existencia y firma de registros
- ▲ Medir el tiempo de respuesta

□ **ss**: proporciona visibilidad de puertos abiertos, estados de conexión y métricas de gestión, útil para

- △ Distinguir saturación de backlog de escucha frente a agotamiento de CPU.
- △ Detectar acumulación de conexiones en espera

- **openssl**: Permite examinar cadena de certificados, protocolos, útil para:
 - △ Verificar que el servidor presente la cadena completa
 - △ Inspeccionar OCSP
 - △ Evaluar tiempos de **handshake**

Conexión con Linux/Bash, Make y automatización

- El flujo recomendable es definir contratos observables (HTTP), gobernar el descubrimiento (DNS) y proteger el transporte (TLS) y convertir todo en tareas reproducibles
- Para Dev Sec Ops se añaden verificaciones de política como versiones mínimas de protocolo.