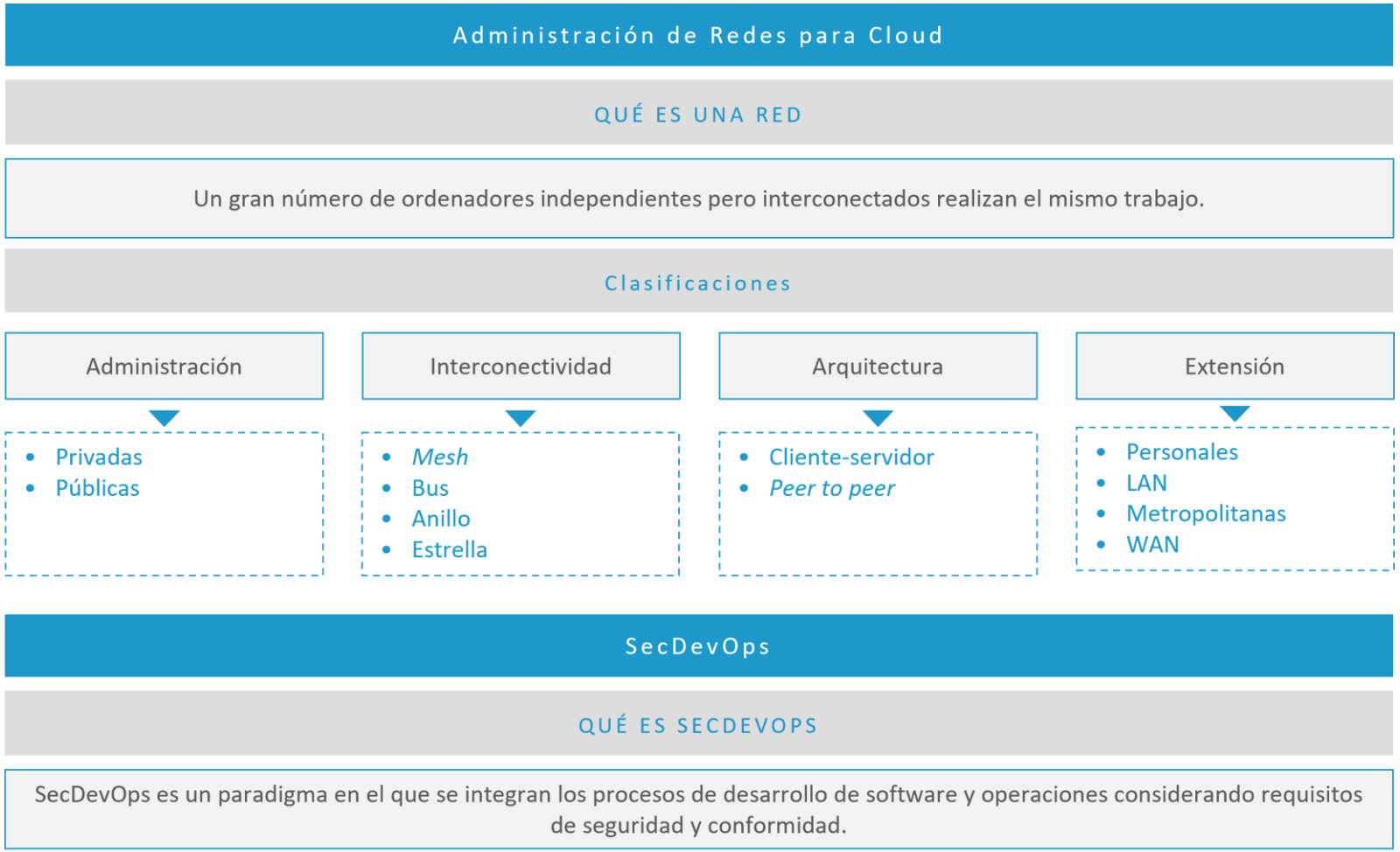
SecDevOps y Administración de Redes para Cloud

Introducción

Índice

Esquema



Ideas clave

1.1. Introducción y objetivos

Abarcaremos dos conceptos muy amplios: **administración de redes en la nube** y **seguridad aplicada a DevOps.**

Tradicionalmente, los roles de **administrador de sistemas** y de **administrador de redes** eran muy diferentes. Las herramientas y las metodologías eran diversas y el grado de especialización era muy alto.

La necesidad de **automatización** en entornos DevOps ha reducido la brecha y es habitual encontrar **roles mixtos.** No obstante, el funcionamiento básico de las redes no ha cambiado. Se explicará este funcionamiento de manera general y se concretará en entornos de nube.

No hay excepciones en la incorporación de **controles de seguridad** a un entorno DevOps. Sí hay diferencias esenciales en cuanto a su aplicación, ya que la velocidad de iteración y las herramientas disponibles son diferentes. Por tanto, cada vez hay más organizaciones que integran la **seguridad** en la **metodología DevOps** desde las primeras etapas.

Los objetivos que se pretenden conseguir en este tema son:

- Comprender el concepto de red.
- Diferenciar varios criterios de clasificación de redes.
- Comprender el alcance del concepto de seguridad y su aplicación a entornos DevOps.

En el siguiente vídeo titulado **«Introducción a SecDevOps y Administración de Redes para Cloud»,** podrás ver un resumen de los principales puntos que se deben tener en cuenta a la hora comenzar a trabajar con estas herramientas.

 Accede al vídeo

1.2. Definición de red

La **referencia principal** en los libros de texto sobre **redes** define una red de ordenadores de la siguiente manera:

«The old model of a single computer serving all of the organization's computational needs has been replaced by one in which a large number of separate but interconnected computers do the job. These systems are called computer networks» (Tanenbaum y Wetherall, 2011). Lo que se traduciría como: «El **modelo tradicional** de un único ordenador para todas las **necesidades de computación** de una organización ha sido reemplazado por un modelo en el que un gran número de ordenadores **independientes pero interconectados** realizan el mismo trabajo. Estos sistemas se denominan redes de ordenadores».

Se observa que la definición no habla de servidores y clientes, ni de centros de datos, ni de proveedores de nube. De hecho, los autores usan para su definición la diferencia principal frente un ordenador monolítico: la interconexión de múltiples ordenadores.

Esto sigue siendo verdad a pesar de la cantidad de **protocolos disponibles** en el mercado. Por ejemplo, la tercera edición de *Computer Networks* de Andrew Tanenbaum, de 1996, fue la primera en incluir referencias al protocolo 802.11, que ha definido y evolucionado el **WiFi** que hay ahora disponible en cualquier bar.

La **primera versión aprobada** fue IEEE 802.11-1997, ahora obsoleta, con una velocidad de hasta 2 Mbps. Nuevas versiones ampliaron la velocidad a lo largo de los años, como el IEEE 802.11g (2003, hasta 54 Mbps) o el IEEE 802.11ac (2013, hasta 500Mbps).

Los autores incluso han incorporado un apartado sobre *Radio Frequency Identification* (RFID), el protocolo que hace funcionar las **tarjetas de proximidad,** en la última versión.

Esta asignatura se centrará en el **aspecto práctico de las redes** en entornos de nube, en los que un DevOps realiza su actividad habitual: **conectividad** y **seguridad** a nivel de sistema operativo y redes virtuales en nube pública.

1.3. Criterios de clasificación

Las **redes de ordenadores** pueden **clasificarse** atendiendo a varios criterios. En función del problema a tratar, se usará un criterio u otro.

Administración

Desde el punto de vista de un administrador, una red puede ser una **red privada,** que pertenece a un **único sistema autónomo** y no se puede acceder fuera de su ámbito físico o dominio lógico, o también puede ser **pública,** a la que **todos pueden acceder.**

En este sentido, una **red casera** formada por un dispositivo ADSL o de fibra y los equipos informáticos conectados a ella con o sin cable (ordenadores, teléfonos móviles, televisiones, impresoras, etc.) es una **red privada. La red pública por antonomasia es Internet.**

Interconectividad

Los dispositivos pueden conectarse entre sí de diferentes maneras. Esta conectividad puede ser **lógica, física** o **combinada,** y existir a **diferentes niveles.**

Red *mesh*	Cada dispositivo se puede conectar a cualquier otro en la red, creando una malla. Una red *peer-to-peer* es un caso de red *mesh* a nivel de aplicación; una red WiFi sin punto de acceso es una red mesh a nivel de acceso al medio.
Red tipo bus	Conecta todos los dispositivos a un único medio. Una red local *Ethernet* cableada es un caso de bus a nivel físico.
Red en anillo	Cada dispositivo está conectado a sus pares izquierdo y derecho únicamente, creando estructuras lineales.
Red en estrella	un dispositivo central tiene conexiones a cada uno de los demás dispositivos. Una red Wifi con punto de acceso (el tipo habitual) es de tipo estrella: el tráfico entre dos ordenadores debe atravesar el router WiFi.

Tabla 1. Clasificación de las redes de ordenadores según su conectividad. Fuente: elaboración propia.

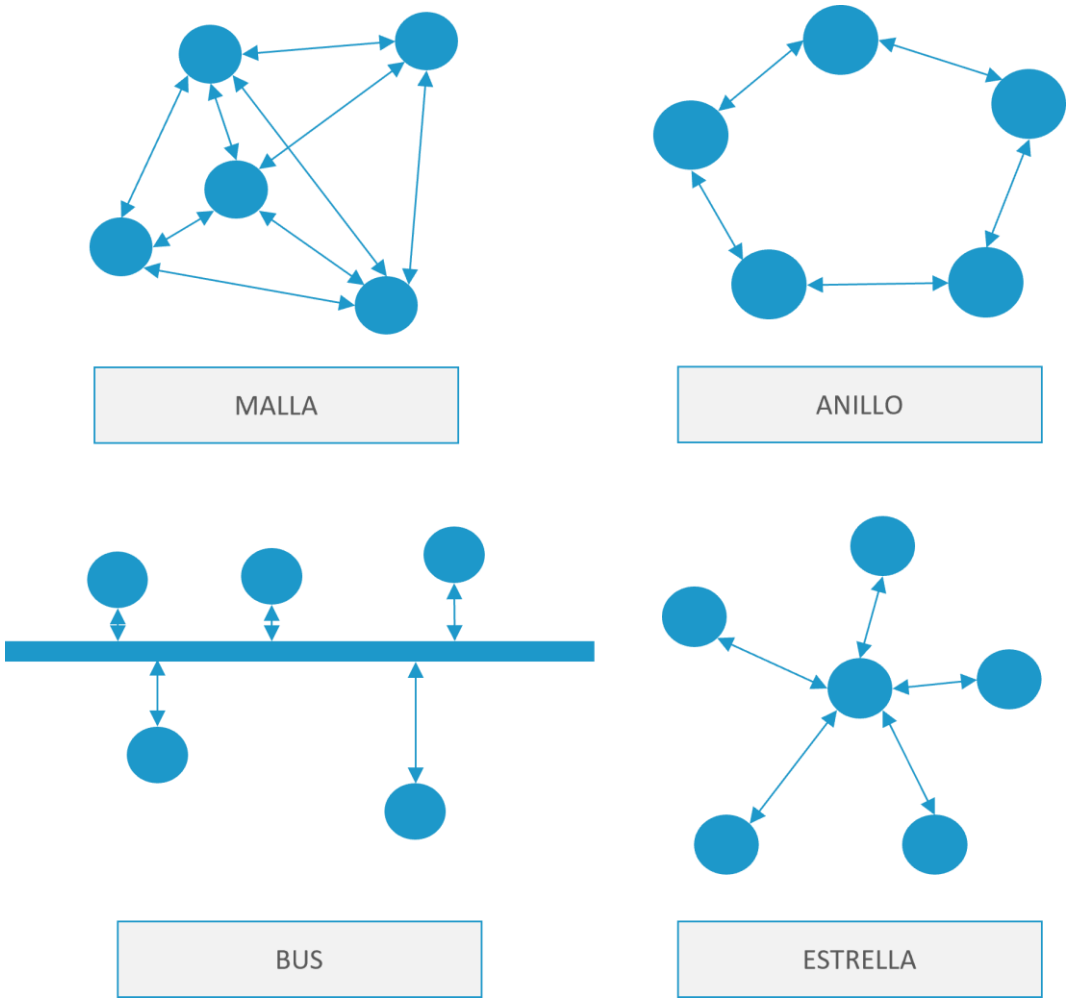


Figura 1. Tipos de redes según la interconexión. Fuente: elaboración propia.

Estos ejemplos sacan a la luz un aspecto importante: **las redes de ordenadores operan a diferentes niveles.** La interconectividad puede variar de un nivel a otro en función del protocolo usado en cada nivel. Por tanto, es crucial **identificar el nivel** en que trabajan las herramientas de un administrador.

Arquitectura

Este criterio se refiere **al rol que cumplen los dispositivos** que pertenecen a la red:

- Puede haber uno o más sistemas que actúen como servidor. Un cliente solicita al servidor que atienda las solicitudes. El servidor toma y procesa la solicitud en nombre de los clientes.
- Dos sistemas pueden estar conectados punto a punto y formar una red *peer-to-peer.* Ambos residen en el mismo nivel, se les denomina pares.

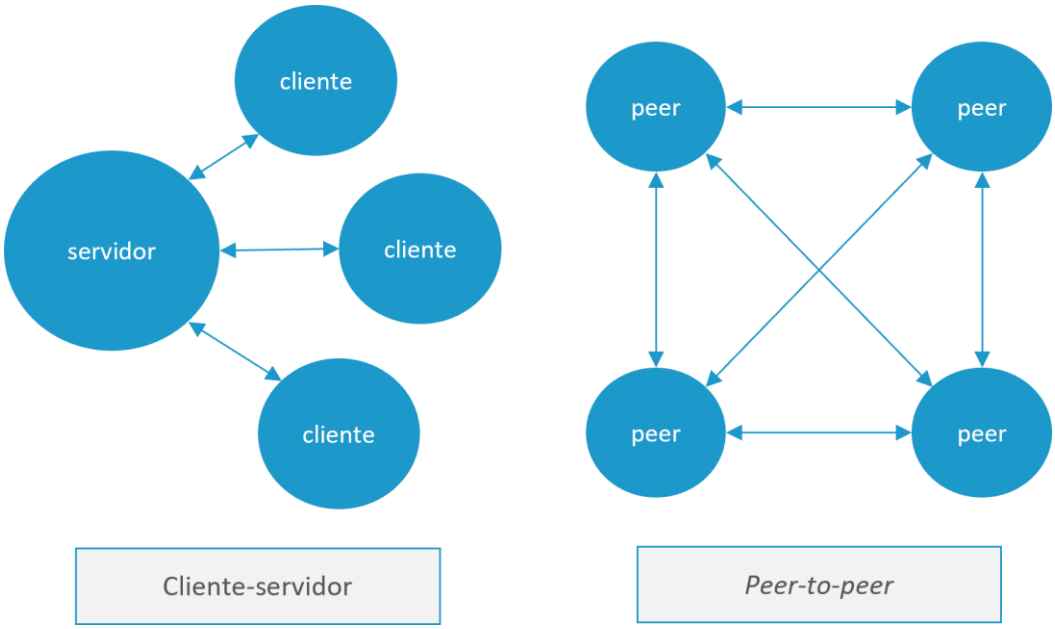


Figura 2. Tipos de redes según su arquitectura. Fuente: elaboración propia.

Extensión geográfica

Atendiendo a la extensión geográfica, las redes se pueden clasificar en:

- **Redes de área personal** o *personal area network* (PAN): aquellas con un alcance de hasta diez metros. Pueden incluir dispositivos habilitados para *bluetooth* o dispositivos habilitados para infrarrojos.

- **Redes de área local** o *local area network* (LAN): aquellas que abarcan una oficina o un edificio y están operadas bajo un único sistema administrativo. Normalmente, una LAN abarca un hogar, una oficina o un edificio, pero puede abarcar una extensión de un campus de universidad. El número de sistemas conectados en LAN puede variar desde al menos dos hasta dieciséis millones. Es la topología habitual para compartir los recursos como impresoras, servidores de archivos o incluso acceso a Internet por parte de dispositivos en una misma ubicación.

- **Redes de área metropolitana** o *metropolitan area network* (MAN): aquellas que cubren una ciudad. Se pueden citar como ejemplos las redes de televisión por cable y las redes WiMAX, aunque solo han tenido aceptación en algunos países.

- **Redes de área amplia** o *wide área network* (WAN): cubren una extensa zona geográfica que puede expandirse a través de provincias o países. Una WAN tiende a interconectar otras redes. Por ejemplo, cada sucursal de una organización puede tener una LAN para compartir el acceso a Internet y las impresoras, mientras que una WAN, mediante la conmutación de etiquetas de protocolo múltiple (MPLS), conecta todas las LAN para permitir el acceso a una aplicación corporativa ubicada en una oficina principal.

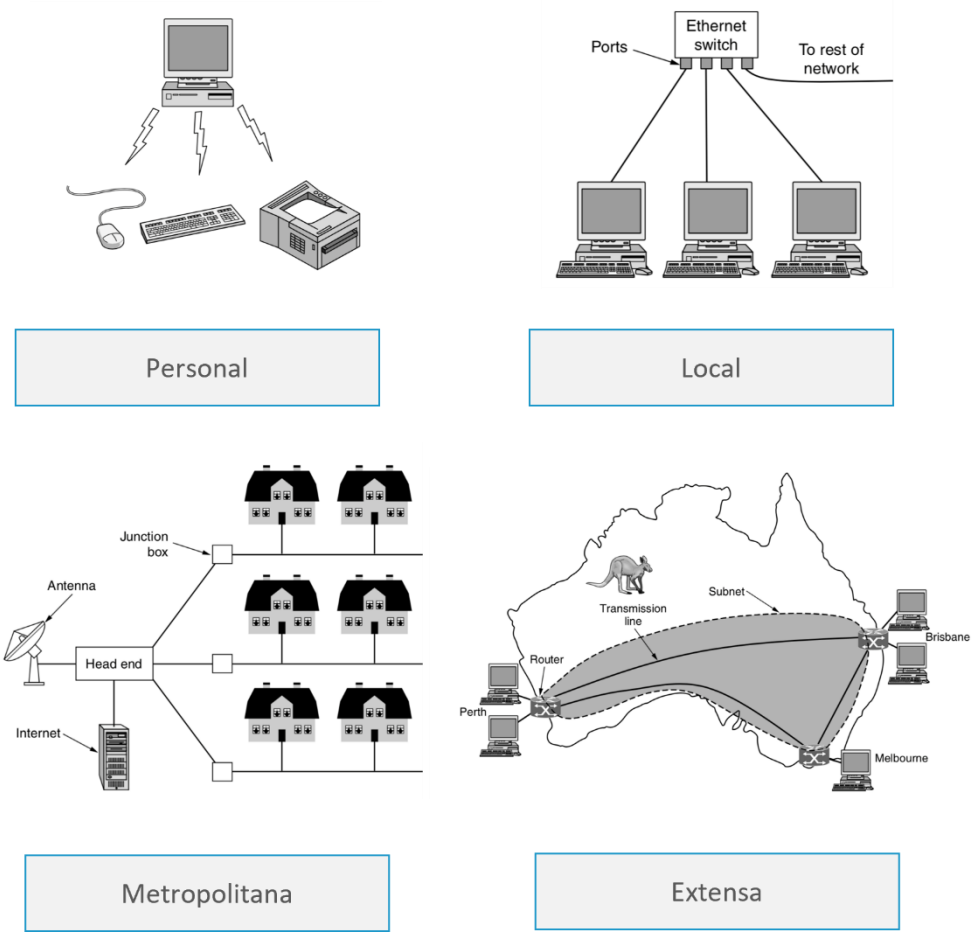


Figura 3. Tipos de redes según su extensión. Fuente: adaptado de Tanenbaum y Wetherall (2011).

1.4. Definición de SecDevOps

Se define DevOps según Farroha y Farroha (2014) como: «DevOps is a term used to describe better communication and collaboration between application development professionals and infrastructure operations professionals», que se traduciría como «DevOps es un término usado para describir **la mejor comunicación y colaboración entre profesionales** de desarrollo de software y operaciones de infraestructura».

Es posible incluir el aspecto de la seguridad en dicha definición para obtener SecDevOps, que amplían Mohan, *et al* (2018): «SecDevOps is a paradigm for integrating the software development and operation processes considering security and compliance requirements», que se traduciría como «**SecDevOps es un**

paradigma en el que se integran los procesos de desarrollo de software y operaciones considerando **requisitos de seguridad y conformidad**».

Estas definiciones se pueden considerar un punto de partida, pero no engloban aspectos como la **automatización** o el **código** como **infraestructura,** tan presentes en cualquier entorno DevOps. SecDevOps incorpora ambos al ámbito de la seguridad.

Grupos de trabajo como Devsecops (https://www.devsecops.org/) recopilan herramientas y ejemplos para automatizar análisis de seguridad, para que:

«Cuando la seguridad se convierte en una parte integral de DevOps, los ingenieros de seguridad puedan desarrollar los controles directamente en el producto, en vez de insertarlo a la fuerza una vez construido» (Vehent, 2018).

1.5. Seguridad en DevOps

En un clima tan competitivo como el actual, aquellas **organizaciones** que quieran tener alguna **opción de éxito** deben ser **ágiles** y **asumir los riegos** que empresas menos atrevidas no están dispuestas a asumir.

La protección de los activos está a cargo de los **equipos de seguridad,** que deben colaborar con los equipos de sistemas, desarrollo, producto y dirección. Esto no es algo nuevo y aplica tanto a empresas tradicionales como a aquellas que han adoptado DevOps. Sin embargo, los equipos de seguridad parecen tener **más reticencia a adoptar esta filosofía.**

DevOps y sus predecesores, como el **Manifiesto ágil** (http://agilemanifesto.org/) y los **catorce principios de Deming** (https://deming.org/explore/fourteen-points/), tienen en común el hecho de centrar sus objetivos en ofrecer a los clientes **mejores**

productos, más rápido. Esto se consigue si todos los individuos alinean su objetivo en uno: **satisfacer al cliente.** Por ejemplo, según Vehent (2018):

> Los gerentes de producto *(product managers)* miden las relaciones de compromiso y retención, los desarrolladores miden la ergonomía y la usabilidad, y los operadores miden el tiempo de actividad y los tiempos de respuesta.

Las **métricas de evaluación tradicionales** pasan a un segundo plano, y la **evaluación de los objetivos de la empresa** debe centrarse en la satisfacción del cliente. Muchos equipos, no solo los de seguridad, siguen anclados en **métricas obsoletas,** pero estos parecen ser más **reticentes** que el resto. Por ejemplo, algunos de sus métricas son el porcentaje de cumplimiento de un estándar de seguridad, el número de incidentes de seguridad, la velocidad de resolución y el porcentaje de equipos sin los últimos parches de seguridad.

Si la organización dirige un **enfoque** en un sentido, el del **cliente,** pero uno de los equipos se dirige en otro, el de **asegurar sus propias métricas,** existe una desconexión que impide conseguir el objetivo principal.

Las organizaciones que aplican las **métricas tradicionales** para repartir **bonos anuales** suelen medir el rendimiento de equipos individuales. En estos casos, será difícil encontrar un equipo que intente ir más allá de su **propio interés.**

Se acaban de mencionar objetivos propios de los equipos de seguridad, pero los **equipos de desarrollo y operaciones** también pueden caer en la tentación de, por ejemplo, no hacer caso de las **recomendaciones de seguridad** para poder llegar a tiempo en una entrega, aun a riesgo de no cumplir con la **política de seguridad corporativa.**

Asimismo, si el equipo de seguridad propone **soluciones poco realistas o demasiado genéricas,** no adaptadas a las particularidades de cada aplicación, los equipos de desarrollo y operaciones no verán estos requisitos como algo alcanzable y, por tanto, no los harán suyos. Los argumentos de ambas partes son válidos (la obligatoriedad de cumplir una política y la urgencia en los plazos de entrega), pero ninguno de los dos equipos se adapta a la **motivación** del otro ni comprende sus **objetivos.**

Esto no significa que los equipos de DevOps no se preocupen por la seguridad; al contrario, en su propio interés está que sus **aplicaciones** sean **sólidas y seguras.** Sin embargo, una **mala comunicación y una desconexión de objetivos** los frustra en la interacción con los compañeros de seguridad.

> Es tan perjudicial para la satisfacción del cliente que un equipo de seguridad organice auditorías de seguridad repentinas que retrasen una entrega o que requiera una arquitectura o configuración muy difícil de implementar, como que el equipo de desarrollo ignore las recomendaciones de seguridad y aumente el riesgo y la exposición de la aplicación.

Si en DevOps se intenta acercar los **enfoques de desarrollo** y operaciones para conseguir los objetivos de la organización, lo mismo debería ocurrir cuando se incorpora un **nuevo equipo a la metodología.** Para asegurarse de que un entorno DevOps es **seguro,** se debe comenzar con una **estrecha integración** entre los **desarrolladores, los operadores y los ingenieros de seguridad.**

La seguridad debe servir al cliente como una **función del servicio,** y los objetivos internos de los equipos de seguridad y los equipos de DevOps deben estar alineados.

En aquellos entornos en los que se ha integrado la **seguridad en DevOps,** los ingenieros de seguridad son una pieza más del pipeline*:* pueden **añadir controles** directamente sobre el producto, en vez de **emitir una recomendación** en forma de una documentación que debe llegar a un desarrollador que puede implementarla, o no, de mejor o peor manera.

La idea fundamental es que, tras la adopción de **técnicas ágiles** por parte de desarrolladores y operadores, los ingenieros de seguridad hagan lo mismo y pasen de **defender sus métricas** a **defender a toda la organización.**

Ahora seguimos con una **serie de pasos para integrar DevOps** con seguridad, gradualmente. Los pasos serán controles sencillos y fáciles de probar que se pueden insertar en el **pipeline DevOps** como una característica más. Los últimos objetivos acompañan a la organización a medida que esta madura.

Seguridad DevOps

La seguridad en DevOps se puede organizar en **tres áreas,** cada una enfocada en un aspecto específico de un **pipeline DevOps.** La **retroalimentación del cliente** estimula el crecimiento organizacional y mejora la seguridad continua.

Seguridad basada en pruebas, o *test-driven security*	La fase más temprana consiste en implementar y probar controles de seguridad directamente en el pipeline DevOps. Por ejemplo, una prueba puede verificar que la configuración de un servidor cumple con los requisitos establecidos o que las aplicaciones incluyen las cabeceras de seguridad necesarias. El rendimiento de esta fase es muy alto, medido a partir de la mejora y el valor que aporta respecto a la dificultad de implementación de los controles. Las pruebas de seguridad deben manejarse de la misma manera que todas las pruebas de aplicaciones en los pipelines de CI y CD: automática y continuamente.
Monitorizar y responder a los ataques	Cualquier servicio en línea acabará siendo atacado. En este caso, el equipo de seguridad debe reaccionar a tiempo. Esta fase consiste en preparar a la organización para actuar en estas situaciones, monitorizando continuamente y respondiendo a las amenazas. En esta fase se consideran la detección de fraudes e intrusos, el análisis forense digital y la respuesta a incidentes.
Evaluar riesgos y madurar la seguridad	Una estrategia de seguridad exitosa no puede tener éxito cuando solo se enfoca en cuestiones técnicas. La tercera fase de la seguridad continua se centra en la operativa de alto nivel, en la que la tecnología no es el aspecto más importante. Entran en juego la gestión de riesgos y las pruebas de seguridad, tanto internas como externas, para ayudar a las organizaciones a reenfocar sus esfuerzos de seguridad e invertir sus recursos de manera más eficiente.

Tabla 2. Áreas de la seguridad en DevOps. Fuente: elaboración propia.

Las organizaciones maduras confían en sus **políticas de seguridad** y trabajan junto a sus **equipos de seguridad.** Alcanzar un punto de confianza en dichas políticas requiere **concentración, experiencia** y **sentido común** para saber cuándo tomar riesgos y cuándo negarse a hacerlo. Una estrategia integral de seguridad combina tecnología y personas, para identificar **áreas de mejora** y asignar **recursos de manera apropiada.**

Seguridad basada en pruebas

El mito de los jáqueres que penetran en *firewalls* gubernamentales o **decodifican la encriptación con sus teléfonos inteligentes** es propio de excelentes películas, pero pobres ejemplos del mundo real. Los casos reales son más mundanos**: los atacantes buscan objetivos fáciles,** como vulnerabilidades de seguridad conocidas, equipos sin parchear, aplicaciones expuestas con las contraseñas predefinidas de instalación o con credenciales expuestas en código fuente.

El **primer objetivo** al implementar una estrategia de seguridad continua debe ser **cuidar el nivel básico:** aplicar conjuntos de controles elementales en la aplicación y la infraestructura de la organización y probarlos continuamente. Estos controles tienen un alcance reducido y pueden ser muy concretos. Vehent (2018) cita varios ejemplos de controles:

- El inicio de sesión con el usuario *root* debe estar deshabilitado en todos los sistemas.
- Los sistemas y las aplicaciones deben ser parcheados a la última versión disponible dentro de los treinta días de su lanzamiento.
- El uso de HTTP debe limitarse a una redirección automática a la dirección HTTPS equivalente.

- Los secretos y las credenciales no deben almacenarse con el código de la aplicación. Como alternativa, se pueden gestionar en una caja de seguridad virtual accesible solo para operadores.
- Los puntos de acceso administrativos o privilegiados de las aplicaciones deben estar protegidos, preferentemente detrás de una VPN.

Esta **lista de controles** no tiene por qué establecerse unilateralmente por el equipo de seguridad. Debería existir un **consenso** en el que tanto **seguridad como desarrollo y operaciones** entienden el valor de cada control y están de acuerdo en implementarlo.

Seguridad a nivel de aplicación

El *Open Web Application Security Project* (OWASP) clasifica los diez **ataques más comunes** en un inventario publicado cada tres años: una lista de *scripting* cruzado *(cross-site scripting),* inyecciones de SQL, peticiones cruzadas *(cross-site request forgery),* ataques de fuerza bruta, etc., aparentemente sin fin. Estas vulnerabilidades se pueden subsanar con un parcheado y una configuración correctos.

Puedes acceder a la lista completa de *OWASP Top Ten* a través del aula virtual o desde la siguiente dirección web: https://owasp.org/www-project-top-ten/

Seguridad a nivel de infraestructura

Confiar en **proveedores de nube públicos** o en una **nube privada** para ejecutar software no exime a un equipo de DevOps de preocuparse por la **seguridad de la infraestructura.**

> *Amazon Web Services* (AWS)
>
> Un ejemplo de este tipo de seguridad lo presenta *Amazon Web Service* (AWS) que menciona su **modelo de responsabilidad compartida** en cualquier curso de formación y a lo largo y ancho de la documentación. Según este modelo, el proveedor se compromete a ofrecer un sistema protegido, y es responsabilidad suya mantenerlo, y el cliente es responsable de hacer buen uso de esos sistemas y de aplicar las configuraciones correctas.

El hecho de que un proveedor de nube ofrezca un servicio de ***firewall* virtual** no significa que el tráfico no deseado vaya a rechazarse sin intervención del cliente. Este deberá asegurarse de que la configuración del *firewall* es la correcta para **exponer las aplicaciones públicas** y **limitar el acceso por *secure shell* (SSH) o *remote desktop protocol* (RDP)** a donde sea exclusivamente necesario.

Seguridad del pipeline

La forma en que los equipos DevOps **despliegan sus productos** a través de la automatización es muy diferente de las operaciones tradicionales. Los **pipelines de CI/CD** son una pieza muy sensible, ya que un atacante podría conseguir control absoluto del código que se ejecuta en producción.

Los equipos de seguridad no suelen estar acostumbrados a **proteger estas soluciones,** pero hay controles adaptados a ellas, como los **controles de integridad de los *commits*** o el **acceso basado en roles** típico de herramientas de CI/CD.

Pruebas continuas

Los **controles de la capa de aplicación, de infraestructura y del pipeline CI/CD** son relativamente sencillos de implementar de forma aislada. Las pruebas de integración continua, sin embargo, no son tan sencillas. Una forma de afrontar esta tarea es seguir los **principios del desarrollo basado en pruebas** o *test-driven development*.

En este paradigma, el **desarrollador** escribe primero las pruebas y a continuación implementa la solución. Si las pruebas definen correctamente el comportamiento de la aplicación, una vez se superan correctamente, se puede considerar que **la aplicación funciona bien.**

El concepto se puede extender a los **controles de seguridad:** se pueden diseñar y escribir pruebas que **comprueben el estado deseado,** e implementar los controles a continuación. Estas pruebas se pueden integrar fácilmente en todas las fases de un pipeline, tal como muestra la Figura 4.

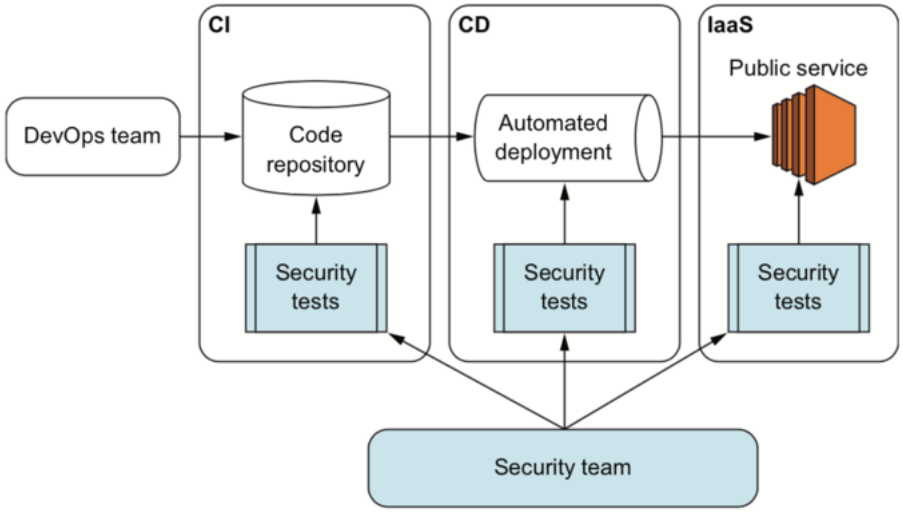


Figura 4. Integración de seguridad en un pipeline CI/CD. Fuente: Vehent, 2018.

Las **ventajas** de este modelo son las siguientes:

- Cuando un ingeniero de seguridad escribe las pruebas, debe **involucrarse** en el desarrollo del producto desde el principio, en vez de emitir unas recomendaciones a *posteriori.*

- Los **controles** deben ser **concretos y específicos.** Se evitan **requisitos vagos** como «encriptar la comunicación de red». En su lugar, la prueba se define como «hacer cumplir HTTPS explícito con cifrados X, Y y Z en todo el tráfico», que establece claramente lo que se espera.

- La **reutilización** de las pruebas entre productos es alta, ya que la mayoría de los productos y servicios comparten la **misma infraestructura base.**

- Al estar involucrado desde el principio, el equipo de seguridad puede **detectar la falta de controles críticos** en etapas más tempranas del desarrollo. En caso de aplicar elementos de seguridad a *posteriori*, existe el riesgo de que el producto llegue a manos del cliente sin ellos, poniéndolos en riesgo.

Las pruebas fallarán inicialmente, como ocurre en el desarrollo basado en pruebas tradicional. Se espera que esto **verifique su corrección** ya que, una vez se implemente la característica de seguridad, las pruebas pasarán con éxito. Al igual que en el desarrollo de software, **la experiencia es un valor fundamental** y los ingenieros de seguridad podrán aportar la suya en las implementaciones iniciales de los controles. Los equipos DevOps podrían acabar encargándose de las pruebas por sí mismos.

La idea detrás de establecer estar pruebas es que la seguridad debe ser una característica más del producto. Si la seguridad se tiene en cuenta como un añadido y los controles se aplican fuera de las aplicaciones, los equipos DevOps no los tomarán como algo propio.

La **relación entre equipos** no será de confianza y la **seguridad del producto** se verá afectada. Los equipos de desarrollo, operaciones y seguridad deben ser copropietarios de la estrategia de seguridad o esta no sobrevivirá.

En resumen, las pruebas continuas adaptan las estrategias de seguridad a la cultura DevOps, **aumenta la comunicación** entre equipos, implica a los ingenieros de seguridad en la **definición de controles** y en su **implementación** y evita que se establezcan controles al margen del desarrollo del producto.

Monitorización y respuesta a ataques

Operar un servicio popular en Internet es, en esencia, esperar a que sea **atacado.** En algún momento, un escáner lo detectará e intentará entrar. La **variedad de ataques** posibles es muy amplia. Un atacante podría:

- Centrarse en atacar usuarios específicos e intentar adivinar sus contraseñas.
- Interrumpir el servicio, impidiendo que los usuarios puedan acceder a él y, por tanto, interrumpir el negocio.
- Pedir un rescate si consigue comprometer el negocio.
- Explotar una vulnerabilidad en la infraestructura para llegar a la capa de datos y extraer información.

Las **organizaciones modernas** son lo suficientemente complejas como para **protegerse de todos los vectores de ataque** con un coste razonable, por lo que es necesario establecer prioridades. La monitorización y la respuesta a los ataques se enfoca en **tres áreas:** registro y detección de fraude, detección de intrusiones y respuesta a incidentes.

Registro y detección de fraude

Generar, almacenar y analizar registros son **acciones útiles** para cualquier sección de la organización:

- Los desarrolladores y operadores necesitan *logs* para resolver problemas y analizar la salud de los servicios.

- Los *product managers* usan estos mismos *logs* (cuando están correctamente definidos) para medir la popularidad de las funciones o la retención de usuarios.

Con respecto a la seguridad, hay **dos necesidades** específicas:

- Detección de anomalías de seguridad.
- Análisis forense en la investigación de incidentes.

Aunque es esencial, la **recopilación y el análisis de registros** rara vez es posible al nivel deseado una vez el evento bajo análisis ha ocurrido. La gran cantidad de datos hace que almacenarlos no sea práctico. El concepto de una ***logging* pipeline** para procesar y centralizar eventos de registro de varias fuentes es valioso porque **proporciona un solo túnel** donde realizar la detección de anomalías.

Es un **modelo más simple** que pedirle a cada grupo que realice la detección por sí mismo, pero puede ser **difícil de implementar** en un entorno grande. Este pipeline se muestra a alto nivel en la Figura 5.

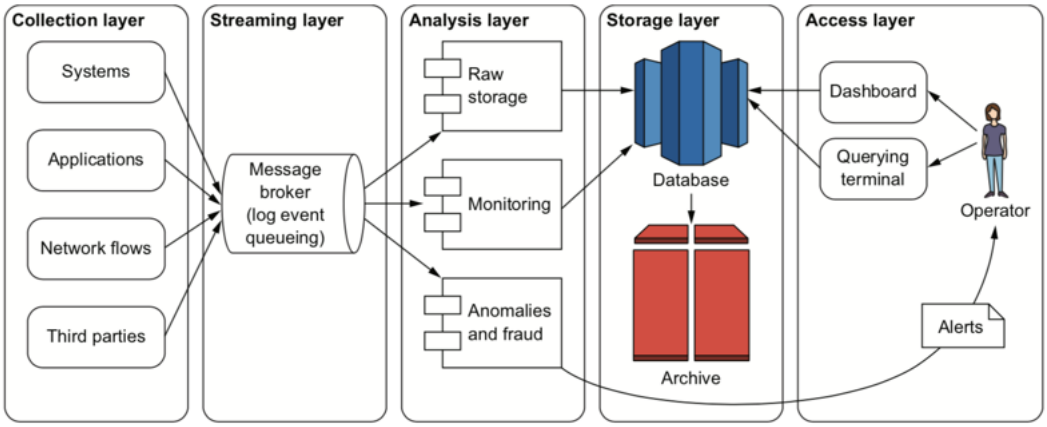


Figura 5. *Logging pipeline*. Fuente: Vehent, 2018.

Las **fases** del pipeline son las siguientes:

- Una **capa de recopilación** en el que los componentes de la infraestructura redirigen sus *logs* a una cola central.

- Una **capa de transmisión** para capturar y enrutar los eventos de registro a los consumidores.

- Una **capa de análisis** para inspeccionar el contenido de los registros, detectar fraudes y generar alertas. Cada tarea puede ser llevada a cabo por un componente diferente, pero todos son alimentados con el mismo origen de datos.

- Una **capa de almacenamiento** para archivar registros.

- Una **capa de acceso** para permitir a los operadores y desarrolladores acceder a los registros en modo interactivo, consola y con alertas proactivas.

Un pipeline que siga esta estructura proporciona al equipo de seguridad las **funcionalidades básicas** que necesita para vigilar la infraestructura.

Detección de intrusiones

Cuando un atacante consigue **introducirse en una infraestructura,** suele seguir estos pasos:

- **Deposita una carga útil *(payload)* en los servidores de destino.** Típicamente, será un *script* autocontenido lo suficientemente pequeño como para descargarlo y ejecutarlo sin llamar la atención.

- **El *script* despliega una puerta trasera** que contacta con un servidor remoto. En esta conexión establece un canal de comando y control (C2, *command-and-control*). Los canales C2 pueden consistir en una página HTML con palabras clave ocultas, un registro DNS de tipo TXT que contenga comandos o en una conexión de IRC.

- **La puerta trasera aplica las instrucciones** que recibe en el canal C2. Intentará replicarse a otros nodos de la red con el objetivo de encontrar datos valiosos.

- Si se da el caso, **usará un segundo canal** para hacer llegar los datos al atacante.

El objetivo será **observar y analizar** el tráfico de la red y los eventos del sistema para detectar estos pasos. Algunas **herramientas** que lo permiten son las siguientes:

- **Sistema de detección de intrusos o *intrusion detection system* (IDS):** los IDS analizan el tráfico de red con el objetivo de detectar actividad fraudulenta. Mientras que un *firewall* bloquea tráfico en base a unos parámetros fijos de las capas de red, transporte y aplicación, los IDS buscan patrones específicos que identifiquen canales C2 y actividades sospechosas. La Figura 6 muestra un diagrama general de un IDS.

- **Auditoría de conexión:** además de la búsqueda de amenazas en tiempo real, es fundamental mantener una auditoría de conexiones para llevar a cabo análisis forenses tras un incidente.

- **Auditoría del sistema:** no solo es necesario auditar las conexiones de red, sino el acceso a los sistemas. Un atacante puede ser capaz de ganar acceso a un sistema, pero necesitará, además, ocultar sus pasos para pasar realmente desapercibido. El pipeline de registros podría usar estos datos para detectar intrusiones.

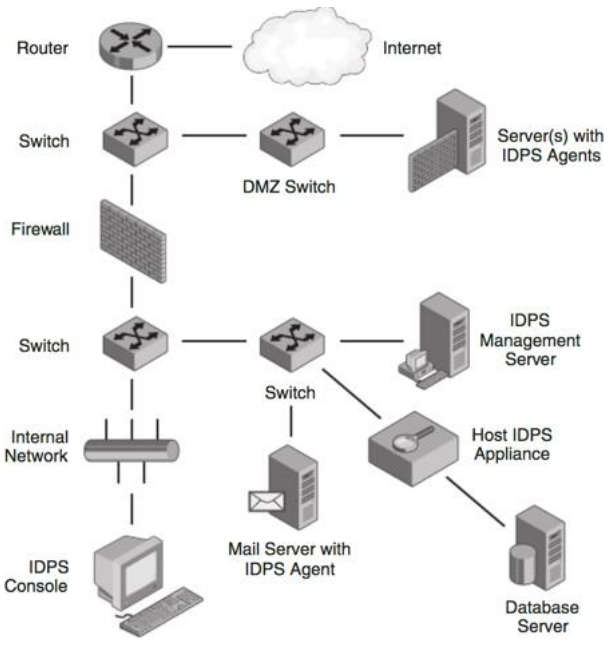


Figura 6. Esquema general de un IDS. Fuente: Newman, 2010.

Esta tarea no es fácil y requiere de **comunicación y colaboración** continua entre los equipos de seguridad y operaciones. Es habitual llegar a situaciones en las que se añade una **sobrecarga excesiva,** tanto a la infraestructura, como al personal, sin conseguir un beneficio claro.

Respuesta a incidentes

Nadie quiere verse en la situación de enfrentarse a un problema de seguridad. Las **violaciones de seguridad** son **situaciones caóticas,** llenas de incertidumbre, que pueden afectar tanto al negocio de manera inmediata como a la imagen de la empresa a largo plazo. Los equipos técnicos deben **recuperar el funcionamiento normal de las aplicaciones de negocio,** controlar los daños y volver a la normalidad cuanto antes.

La **reacción** ante un incidente de seguridad debería seguir **seis fases:**

- **Preparación:** garantizar que la organización ha definido los procesos mínimos necesarios para lidiar con un incidente.
- **Identificación:** decidir rápidamente si una anomalía es un incidente de seguridad.
- **Contención:** impedir que la violación se extienda.
- **Erradicación:** eliminar las amenazas de la organización.
- **Recuperación:** devolver la organización a las operaciones normales.
- **Lecciones aprendidas:** analizar el incidente a *posteriori* para aprender de él.

Evaluación de riesgos y maduración de la seguridad

La **integración de pruebas continuas** y la detección y respuesta a incidentes no son los únicos pilares de una estrategia completa de seguridad. La **gestión de riesgos** y el lado humano de los problemas complementan los aspectos técnicos de esta estrategia.

Evaluación de riesgos

La **gestión de riesgos** no consiste, solamente, en **hojas de cálculo con métricas codificadas con colores.** Esto ocurre, desafortunadamente, con demasiada frecuencia y ha llevado a muchas organizaciones a evitar la gestión de riesgos.

La gestión del riesgo consiste en **identificar y priorizar los problemas** que amenazan el negocio de la compañía y sus objetivos de crecimiento. Algunos de los **objetivos** de un buen enfoque de gestión de riesgos deberían ser:

- **Analizar los riesgos al mismo ritmo que las actividades DevOps:** en pequeños incrementos, pero muy a menudo. Si el software cambia continuamente, el análisis de riesgos debería poder acompañar a la organización a la misma velocidad; de lo contrario, se convertirá en un cuello de botella.

- **Automatizar:** la integración en un entorno DevOps significa que las tareas manuales no deberían ocurrir, o, al menos, no habitualmente.

- **Involucrar a todos los equipos en el análisis.**

El análisis de riesgos no debería ser una **tarea para después,** sino **parte integral de la arquitectura de seguridad,** para proveer de un valor real a todos los equipos. De lo contrario, los equipos no lo asumirán como propio y solo verán en ello un gasto de tiempo en reuniones semanales.

Pruebas de seguridad

De nada sirve un programa de seguridad si no se puede saber si funciona bien o no. La estrategia de seguridad deberá **incorporar pruebas regulares** más allá de los controles integrados en el pipeline de CI/CD, por ejemplo:

- **Evaluar la seguridad** de las aplicaciones y la infraestructura internamente, utilizando técnicas de seguridad como escaneo de vulnerabilidades, *fuzzing,* análisis de código estático o auditoría de configuración.
- **Usar consultoras externas** para auditar la seguridad de los servicios centrales. Cuando se ejecutan adecuadamente, las auditorías de seguridad aportan mucho valor a una organización y ayudan a aportar nuevas ideas y perspectivas a un programa de seguridad.
- **Establecer un programa de recompensas de errores.** Los investigadores de seguridad independientes pueden aprovechar el hecho de que muchas compañías publican sus productos como código abierto para, a cambio de una recompensa, realizar pruebas de sus aplicaciones.

Conclusión

Los equipos de seguridad deben **mantener una relación estrecha** con sus colegas de desarrollo y operaciones. Solo así podrán adoptar la cultura DevOps y actualizar y adaptar las técnicas tradicionales, como la **detección de intrusiones** o el **escaneo de vulnerabilidades,** en el pipeline de CI/CD.

Un programa de seguridad no se define de un día para otro: debe formar parte integral de la **estrategia de la compañía** y madurar con ella. Solo cuando existe colaboración entre equipos **surgirá la confianza necesaria** para trabajar juntos con un objetivo común. En resumen, la manera de garantizar la seguridad de los productos en un entorno DevOps es **integrar** a todos los equipos bajo este paradigma.

1.6. Referencias bibliográficas

Farroha, B., y Farroha, D. (2014). A framework for managing mission needs, compliance, and trust in the DevOps environment. *IEEE Military Communications Conference.*

Mohan, V., Ben Othmane, L. y Kres, A. (2018). *BP: security concerns and best practices for automation of software deployment processes: an industrial case study.* Iowa State University. https://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=1070&context=ece_conf

Newman, R. (2010). Com*puter Security: Protecting Digital Resources.* Jones and Bartlett Publishers.

Tanenbaum, A. y Wetherall, D. (2011). *Computer Networks.* (5ª ed.) Pearson New International.

Vehent, J. (2018). *Securing DevOps.* Manning Publications.

A fondo

Computer Networks

Tanenbaum, A. y Wetherall, D. (2011). *Computer Networks.* (5ª ed.) Pearson New International.

El «Tanenbaum», como es conocido en múltiples facultades, es la biblia de las redes de ordenadores. Su introducción pone al lector en contexto con el concepto de red con ejemplos de la vida diaria.

DevSecOps Manifiesto

DevSecOps. (S. f.). *Manifesto*. https://www.devsecops.org/

A semejanza del Agile Manifesto (https://agilemanifesto.org/). Los autores plasman su filosofía de trabajo: cómo acercar las operaciones de seguridad al producto y al usuario final, y no al revés.

Estándares del IEEE

Standards Association. (https://standards.ieee.org/)

En este recurso puedes encontrar estándares del IEEE como el *IEEE 802.11-1997 - IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,* o el *IEEE 802.11ac-2013 - IEEE Standard for Information technology--Telecommunications and information exchange between systems—Local and metropolitan area networks--Specific requirements--Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications--Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz.*

Test

1. ¿Cómo se clasifican las redes desde el punto de vista de un administrador?

A. Como cliente-servidor, *peer-to-peer* o híbridas.

B. Como *network mesh* (malla), estructura de bus, estructura lineal, estructura de tipo estrella o estructura híbrida.

C. Como privadas o públicas.

D. Como LAN y WAN.

2. ¿Cuál de las siguientes afirmaciones sobre PAN es correcta?

A. Es una red de ordenadores extendida dentro de un edificio y operado bajo un único sistema administrativo.

B. Por lo general, las redes de telecomunicaciones son una PAN.

C. Utiliza el modelo TCP/IP y utiliza IP como su protocolo de direccionamiento.

D. Ninguna de las anteriores.

3. ¿Cuál de las siguientes afirmaciones sobre LAN es correcta?

A. Es una red de ordenadores extendida dentro de un edificio y operado bajo un único sistema administrativo.

B. Ethernet es un protocolo habitual de LAN.

C. Varias LAN pueden estar interconectadas mediante una MAN o una WAN.

D. Todas las anteriores son correctas.

4. ¿Cuál de las siguientes afirmaciones sobre WAN es correcta?

A. Es una red de ordenadores extendida dentro de un edificio y operado bajo un único sistema administrativo.

B. Es una red pública.

C. Abarca una extensión geográfica amplia, del orden de magnitud de un país o superior.

D. Es de tipo *peer-to-peer.*

5. ¿Cómo se clasifican las redes desde el punto de vista de la interconectividad?

A. Como cliente-servidor, *peer-to-peer* o híbridas.

B. Como *network mesh* (malla), estructura de bus, estructura lineal, estructura de tipo estrella o estructura híbrida.

C. Como privadas o públicas.

D. Ninguna de las anteriores.

6. ¿Cómo se clasifican las redes desde el punto de vista de la extensión geográfica?

A. Como cliente-servidor, *peer-to-peer* o híbridas.

B. Como *network mesh* (malla), estructura de bus, estructura lineal, estructura de tipo estrella o estructura híbrida.

C. Como privadas o públicas.

D. PAN, LAN, MAN y WAN.

7. ¿Cuál de las siguientes afirmaciones sobre una red cliente-servidor es correcta?

A. Puede operar en una LAN, MAN y WAN, pero no en una PAN.

B. Hay dos tipos de nodos con roles diferentes.

C. No se usan para compartir archivos.

D. Al contrario que las *peer-to-peer,* pueden funcionar sobre TCP/IP.

8. Relaciona el tipo de red con el protocolo correspondiente.

PAN	1
LAN	2
MAN	3
WAN	4

A	Ethernet
B	WiMAX
C	MPLS
D	Bluetooth

9. ¿Qué relación hay entre SevDevOps y DevOps?

A. SecDevOps es una ampliación del paradigma DevOps.

B. SecDevOps es un modelo diferente de DevOps y aplica procedimientos diferentes.

C. Los ingenieros de seguridad tienen el rol de SecDevOps, mientras que los administradores de sistemas tienen el rol de DevOps.

D. Ninguna de las anteriores.

10. ¿Qué diferencia hay entre la administración de redes tradicionales y de redes en la nube?

A. Principalmente los protocolos, ya que han hecho falta protocolos totalmente nuevos para trabajar en la nube.

B. Principalmente las herramientas, ya que el funcionamiento básico de los protocolos no tiene por qué cambiar.

C. No tienen nada que ver y deben ser roles separados dentro de una compañía.

D. Ninguna, los procedimientos, herramientas y protocolos son iguales.