SOLUCIÓN LABORATORIO SESIÓN 2

Paso 1: Definir los Términos Clave (Confidencialidad, Integridad y Disponibilidad)

Confidencialidad:
- **Definición:** Explica que la confidencialidad se refiere a la protección de la información para asegurar que solo las personas autorizadas puedan acceder a ella. Es uno de los pilares fundamentales de la seguridad de la información.

- **Conceptos Relacionados:** Incluye el cifrado, controles de acceso, y autenticación como métodos para garantizar la confidencialidad

a. La confidencialidad se refiere a la protección de la información, garantizando que solo personas autorizadas puedan acceder a los datos y esté protegida contra el acceso no autorizado. Esto asegura que la información sensible esté disponible únicamente para quienes tienen los permisos correspondientes.

Cifrado de datos: Los datos se cifran para que solo personas con las claves correctas puedan leer la información. Ejemplo: Si envías un correo con información sensible, el cifrado asegura que solo el receptor autorizado pueda leerlo.

El **cifrado de datos** (o **encriptación de datos**) es un proceso que convierte información legible (texto plano) en un formato ilegible (texto cifrado) utilizando un algoritmo y una clave. Su propósito es **proteger la confidencialidad** de la información, de modo que solo las personas autorizadas con la clave adecuada puedan descifrarla y acceder al contenido original.

Tipos principales de cifrado:

1. **Cifrado simétrico**:
   - Usa la **misma clave** para cifrar y descifrar los datos.
   - Ejemplo: **AES (Advanced Encryption Standard)**.
   - Es rápido y eficiente, pero requiere un método seguro para compartir la clave.
2. **Cifrado asimétrico**:
   - Usa un **par de claves**: una pública para cifrar y una privada para descifrar.
   - Ejemplo: **RSA (Rivest-Shamir-Adleman)**.
   - Es más seguro para intercambiar información, aunque más lento que el simétrico.

Aplicaciones comunes:

- Comunicación segura en internet (HTTPS).
- Protección de archivos en discos duros o servicios en la nube.
- Sistemas de autenticación.
- Mensajería segura (como WhatsApp o Signal).

EJEMPLO CIFRADO SIMETRICO

```
from cryptography.fernet import Fernet
```

```
# 1. Generar una clave
clave = Fernet.generate_key()
fernet = Fernet(clave)

# 2. Texto que quieres cifrar
mensaje_original = "Hola, este es un mensaje confidencial."

# 3. Cifrar el mensaje
mensaje_cifrado = fernet.encrypt(mensaje_original.encode())
print("Mensaje cifrado:", mensaje_cifrado)

# 4. Descifrar el mensaje
mensaje_descifrado = fernet.decrypt(mensaje_cifrado).decode()
print("Mensaje descifrado:", mensaje_descifrado)
```

EJEMPLO ASIMETRICO

```
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import hashes, serialization

# 1. Generar par de claves (privada y pública)
clave_privada = rsa.generate_private_key(
    public_exponent=65537,
    key_size=2048,
)

clave_publica = clave_privada.public_key()

# 2. Texto original
mensaje_original = "Este mensaje será cifrado con RSA"

# 3. Cifrar usando la clave pública
mensaje_cifrado = clave_publica.encrypt(
    mensaje_original,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)
print("Mensaje cifrado:", mensaje_cifrado)

# 4. Descifrar usando la clave privada
mensaje_descifrado = clave_privada.decrypt(
    mensaje_cifrado,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)
```

```
print("Mensaje descifrado:", mensaje_descifrado.decode())
```

El **cifrado asimétrico** funciona usando un **par de claves diferentes pero relacionadas matemáticamente**: una **clave pública** y una **clave privada**. A continuación te explico el funcionamiento paso a paso:

1. Generación de claves

- El usuario (por ejemplo, *Ana*) genera dos claves:
  - **Clave pública**: puede compartirse libremente.
  - **Clave privada**: debe mantenerse en secreto.

2. Cifrado

- Si alguien (por ejemplo, *Juan*) quiere enviarle un mensaje seguro a Ana:
  - Usa **la clave pública de Ana** para **cifrar** el mensaje.
  - El mensaje cifrado **solo podrá ser descifrado con la clave privada de Ana**.

3. Descifrado

- Ana recibe el mensaje cifrado y usa **su clave privada** para **descifrarlo**.
- Nadie más (ni siquiera Juan) puede descifrarlo sin la clave privada de Ana.

Ejemplo con analogía:

Imagina que:

- La **clave pública** es un candado abierto que Ana envía a todo el mundo.
- La **clave privada** es la llave que solo Ana tiene.

Paso a paso:

1. Juan pone su carta dentro de una caja y la cierra con el candado (clave pública de Ana).
2. Solo Ana puede abrir la caja porque solo ella tiene la llave (clave privada).

Ventajas del cifrado asimétrico:

- No necesitas compartir una clave secreta de antemano.
- Muy útil para comunicaciones seguras por internet.
- Se usa también para **firmas digitales** (verificación de identidad).

Controles de acceso: Asignar permisos específicos para diferentes niveles de usuarios, asegurando que solo tengan acceso a lo necesario. (principio de menor privilegio).

Control de acceso físico:

- Limita el acceso a instalaciones, servidores, centros de datos, etc.
- Ejemplos: tarjetas de identificación, cerraduras biométricas, cámaras.

Control de acceso lógico o digital:

- Regula el acceso a sistemas, redes, aplicaciones, archivos y bases de datos.
- Ejemplos: contraseñas, autenticación multifactor (MFA), roles de usuario.

Modelo	Descripción
DAC (Discretionary Access Control)	El propietario del recurso decide quién accede y qué puede hacer.
MAC (Mandatory Access Control)	El sistema impone reglas basadas en niveles de seguridad (por ejemplo, militar).
RBAC (Role-Based Access Control)	Los permisos se asignan según el **rol del usuario** en la organización.
ABAC (Attribute-Based Access Control)	El acceso se decide según **atributos** del usuario, del recurso o del entorno (hora, ubicación, etc.).

Tipo de Control	Ejemplo	Finalidad
Físico	Tarjeta de proximidad	Controlar el ingreso a oficinas o salas de servidores
Físico	Cerradura biométrica (huella dactilar)	Restringir el acceso a áreas restringidas
Físico	Guardia de seguridad	Verificación manual de credenciales en puntos de acceso
Físico	Cámaras de vigilancia (CCTV)	Supervisar intentos de acceso no autorizado
Físico	Torniquetes o puertas con PIN	Permitir ingreso solo a personal autorizado
Lógico	Usuario y contraseña	Validar identidad para acceder a sistemas o aplicaciones
Lógico	Autenticación multifactor (MFA)	Añadir una capa extra de seguridad a los accesos
Lógico	Control de acceso basado en roles (RBAC)	Restringir funcionalidades según el rol del usuario
Lógico	Permisos en archivos o carpetas	Definir quién puede leer, escribir o borrar información
Lógico	Políticas de sesión (timeout, bloqueo)	Evitar accesos prolongados o no supervisados

Técnicas relacionadas:

- **Autenticación**: verificar quién eres.
- **Autorización**: definir qué puedes hacer.
- **Auditoría**: registrar y supervisar los accesos.

Contraseñas seguras y Autenticación: Verificación de la identidad del usuario mediante contraseñas, autenticación multifactor (MFA). Protegen el acceso a sistemas o datos.

Tipo de factor	Ejemplo
Algo que sabes	Contraseña, PIN, respuesta secreta
Algo que tienes	Teléfono móvil, token físico, tarjeta
Algo que eres	Huella dactilar, rostro, iris, voz

Herramientas y tecnologías comunes:

- **Google Authenticator**, **Microsoft Authenticator** (códigos temporales).
- **Tokens físicos** como YubiKey.
- **Notificaciones push** (ej. Duo Security, Okta).
- **Biometría** integrada (Face ID, Windows Hello).

Redes privadas y VPN: Protegen la transmisión de datos entre ubicaciones remotas.

¿Qué es una red privada?

Una **red privada** es una red de computadoras que está restringida a un grupo específico de usuarios u organizaciones. **No es accesible desde internet sin autorización**.

- Usada comúnmente dentro de empresas, hogares o instituciones.
- Utiliza direcciones IP privadas (por ejemplo: 192.168.x.x o 10.x.x.x).
- Proporciona seguridad al mantener los dispositivos fuera del alcance de internet pública.

Ejemplos:

- La red Wi-Fi de una oficina.
- La red interna de una universidad o fábrica.
- Los servidores y equipos conectados en una sede corporativa.

¿Qué es una VPN (Virtual Private Network)?

Una **VPN** es una tecnología que permite **crear una red privada virtual** a través de una red pública como Internet. Su objetivo es **proteger la conexión y los datos transmitidos** mediante **cifrado**.

Características:

- Establece un **túnel seguro** entre el dispositivo del usuario y la red de destino.
- Oculta la dirección IP real y cifra el tráfico.
- Permite a los usuarios **acceder a recursos internos de forma remota**, como si estuvieran en la red privada local.

- Un empleado accediendo a la red de la empresa desde casa.
- Usuarios que quieren navegar por internet de forma más segura.
- Evitar restricciones geográficas o censura (uso común en VPN comerciales).

Característica	Red privada	VPN
Acceso	Solo desde dentro de la red física	Desde cualquier lugar a través de internet
Seguridad	Requiere firewalls y autenticación	Usa cifrado para proteger datos
Ejemplo de uso	Red LAN de una oficina	Teletrabajo seguro desde casa
Exposición a internet	No expuesta directamente	Usa internet pero de forma segura

Riesgos de la confidencialidad

- Filtración de datos personales (ej. fugas de datos en redes sociales o bancos).
- Robo de propiedad intelectual.
- Violaciones legales o regulatorias (como el RGPD o la Ley de Protección de Datos).

Riesgo	Descripción breve
Accesos no autorizados	Usuarios sin permisos que acceden a sistemas o archivos confidenciales.
Pérdida o robo de dispositivos	Laptops, USB, celulares extraviados con información sensible sin cifrado.
Contraseñas débiles o robadas	Facilitan que atacantes accedan a cuentas o sistemas.
Phishing y ataques de ingeniería social	Engaños para que los usuarios revelen información privada (contraseñas, datos).
Interceptación de comunicaciones	Espionaje de datos en tránsito, por ejemplo, en redes Wi-Fi públicas.
Malware y spyware	Programas maliciosos que capturan datos sin que el usuario lo sepa.
Malas configuraciones de seguridad	Servidores o aplicaciones mal configurados dejan puertas abiertas a intrusos.
Fugas internas (insider threats)	Empleados que filtran o venden información, voluntaria o involuntariamente.
Falta de cifrado	Datos almacenados o transmitidos sin cifrar son vulnerables a ser leídos.
Uso indebido de redes sociales	Publicación accidental de información sensible o privada.

¿Cómo mitigar estos riesgos?

- Implementar **autenticación multifactor (MFA)**.
- Usar **cifrado** en archivos, correos y comunicaciones.
- Establecer **políticas de acceso por roles** (RBAC).
- Capacitar a los usuarios en **seguridad informática**.
- Realizar **auditorías y monitoreo continuo** de accesos.
- Mantener actualizado el **software de seguridad** y sistemas operativos.

b. **Integridad:** La integridad asegura que los datos no hayan sido alterados de manera no autorizada. Garantiza que la información es confiable y exacta. Significa que los **datos son exactos, completos y confiables**, y que **no han sido modificados** desde su creación, transmisión o almacenamiento, a menos que la modificación haya sido autorizada.

Riesgo	Descripción
Malware	Puede modificar archivos, bases de datos o registros.
Errores humanos	Cambios accidentales en configuraciones o información.
Ataques Man-in-the-Middle (MitM)	Alteran datos mientras se transmiten.
Usuarios malintencionados	Empleados o atacantes externos que modifican registros.
Corrupción de datos	Por fallos en el hardware, software o almacenamiento.

Mecanismo	Función
Hashing (funciones hash)	Verifica si los datos han sido modificados (ej. SHA-256).
Firmas digitales	Aseguran que el emisor es auténtico y que el mensaje no fue alterado.
Controles de versiones	Permiten rastrear y revertir cambios no autorizados.
Registros de auditoría (logs)	Detectan alteraciones sospechosas en los sistemas.
Restricciones de escritura	Limitan quién puede modificar qué archivos o sistemas.

Suma de verificación (hashes): Un código único generado a partir de los datos para verificar que no se han modificado, que es una fuente confiable y no han sido manipulados.

EJEMPLO HASHES

```
from google.colab import drive
drive.mount('/content/drive')
```

```
import hashlib

ruta_archivo =
'/content/drive/MyDrive/TalentoTech/Ciberseguridad/Laboratorios/Sesion
2/ejemplo.txt'

# Función para calcular el hash SHA-256 de un archivo
def calcular_hash_archivo(ruta_archivo):
    sha256 = hashlib.sha256()
    with open(ruta_archivo, 'rb') as f:
```

```
        while chunk := f.read(4096):
            sha256.update(chunk)
    return sha256.hexdigest()

# Hash original generado antes de enviar el archivo
hash_original =
calcular_hash_archivo('/content/drive/MyDrive/TalentoTech/Ciberseguridad/Laboratorios/Sesion2/ejemplo.txt')
print(f"Hash original: {hash_original}")
```

```
# Simula la recepción del archivo y verificación del hash
hash_recibido =
calcular_hash_archivo('/content/drive/MyDrive/TalentoTech/Ciberseguridad/Laboratorios/Sesion2/ejemplo2.txt')
print(f"Hash recibido: {hash_recibido}")

if hash_original == hash_recibido:
    print("Integridad verificada: los datos no han sido alterados.")
else:
    print("Integridad comprometida: los datos han sido modificados.")
```

Controles de versiones: Sistemas que permiten rastrear cambios en los datos y revertir a versiones anteriores si es necesario.

- Registrar el historial de cambios: Permite saber quién hizo qué cambio, cuándo y por qué.
- Recuperar versiones anteriores: Puedes volver a un estado anterior del proyecto si algo sale mal.
- Trabajo colaborativo: Varios desarrolladores pueden trabajar en paralelo sin sobrescribir el trabajo de otros.
- Ramas de desarrollo (branches): Facilita el desarrollo de nuevas funciones sin afectar la versión principal del proyecto.

Herramientas comunes de control de versiones:

- Git (la más usada actualmente)
- Subversion (SVN)
- Mercurial
- CVS

c. Disponibilidad: La disponibilidad garantiza que los datos y servicios estén accesibles para los usuarios autorizados cuando los necesiten.

   Conceptos Relacionados:
   Redundancia: Tener múltiples copias de los sistemas para evitar interrupciones en caso de fallos.
   Sistemas de respaldo (backups): Copias de seguridad que permiten recuperar datos en caso de incidentes.

Planificación ante desastres: Preparar estrategias para mantener los servicios en funcionamiento durante incidentes graves. Planes de continuidad del negocio (BCP) y recuperación ante desastres (DRP).
Monitoreo constante y mantenimiento preventivo.
Infraestructura en la nube con alta tolerancia a fallos.

Factores que afectan la disponibilidad

1. Fallos técnicos
- Fallas de hardware: Daños en servidores, discos duros, redes, etc.
- Errores de software: Bugs, cuelgues del sistema operativo, fallas en aplicaciones críticas.
- Obsolescencia tecnológica: Equipos o sistemas que ya no tienen soporte o actualizaciones.
2. Ataques y amenazas
- Ataques DDoS (Denegación de servicio): Saturan un sistema para dejarlo fuera de línea.
- Malware o ransomware: Pueden inutilizar sistemas y bloquear el acceso a los datos.
- Accesos no autorizados: Pueden alterar la configuración o dañar componentes clave.
3. Problemas en la infraestructura
- Cortes de energía eléctrica: Si no hay respaldo (UPS, plantas), puede detener los sistemas.
- Problemas de conectividad: Fallas en redes internas o en la conexión a internet.
- Fallas en el proveedor de servicios en la nube o de hosting.
4. Factores humanos
- Errores de configuración: Cambios mal hechos en el sistema o la red
- Eliminación accidental de archivos o servicios.
- Falta de capacitación del personal técnico.
5. Mantenimiento y actualizaciones
- Actualizaciones mal programadas: Que causan interrupciones en horas críticas.
- Ausencia de mantenimiento preventivo: Lo que aumenta el riesgo de fallos no detectados.
6. Factores externos
- Desastres naturales: Inundaciones, incendios, terremotos, etc.
- Riesgos geopolíticos o sociales: Conflictos, huelgas, bloqueos.

Paso 2: Proporcionar y Analizar Ejemplos Prácticos (25 minutos)

2.1. Ejemplo de Confidencialidad:
Una empresa de salud maneja los registros médicos de los pacientes.
• Aplicación Práctica:
- La empresa utiliza cifrado para proteger los registros de los pacientes.
- Solo el personal médico autorizado puede acceder a los datos utilizando autenticación multifactor o biometría.
- Se aplican roles y permisos definidos por perfil de usuario.

- Sistemas de registro de accesos (logs) para auditar quién accedió a qué información y cuándo.
- Todos los empleados deben firmar acuerdos de confidencialidad.
- Se establecen protocolos claros para el manejo, almacenamiento, impresión o eliminación de historias clínicas.
- Cuando los datos se usan para estudios o reportes estadísticos, se eliminan o codifican los datos personales identificables.

• Discusión: Pregunta a los participantes qué otros métodos de seguridad podrían mejorar la confidencialidad en este contexto (ej., monitoreo de accesos).

Para mejorar la confidencialidad en este contexto se puede:

-Monitoreo de Accesos y Alertas: Implementar sistemas de monitoreo en tiempo real para detectar patrones de acceso inusuales o sospechosos (por ejemplo, accesos fuera del horario laboral, un gran número de registros accedidos por un solo usuario en un corto período de tiempo, intentos fallidos de acceso repetidos). Esto podría generar alertas automáticas para una investigación más profunda.

- Minimización de Datos: Adoptar el principio de solo recopilar y retener la cantidad mínima de datos personales necesarios para un propósito específico. Esto reduce la superficie de ataque y el riesgo en caso de una brecha.

- Capacitación y Concientización Continua: Implementar programas de capacitación regulares para todo el personal sobre la importancia de la confidencialidad, las políticas de seguridad de la empresa y las mejores prácticas para proteger la información del paciente. El factor humano es crucial en la seguridad.

2.2. Ejemplo de Integridad: Una empresa de software distribuye actualizaciones de productos a sus clientes.
• Aplicación Práctica:
- La empresa utiliza hashes para asegurar que los archivos no han sido modificados durante la descarga. Si un cliente descarga una actualización, la suma de verificación del archivo se compara con la suma original.
- La empresa firma digitalmente las actualizaciones usando una clave privada. El cliente puede verificar la firma usando la clave pública de la empresa. Esto asegura integridad y autenticidad (que realmente proviene del proveedor oficial).
- Uso de canales seguros como HTTPS o SFTP para evitar la manipulación en tránsito.
- Algunos sistemas usan redes de distribución de contenido (CDN) confiables y autenticadas.
- Sistemas automáticos validan la integridad antes de instalar una actualización.
- Se usan checksums en el código para asegurar que los componentes no hayan sido alterados.
- Se integran herramientas de DevSecOps en el ciclo de distribución.
- En caso de errores de descarga o corrupción, se usan mecanismos de recuperación automática o verificación por repetición de descarga.

- Se recomienda mantener versiones anteriores como respaldo.

• Discusión: ¿Qué ocurriría si la integridad no se mantuviera? (ej., la instalación de software malicioso sin que los usuarios lo detecten).

¿Qué ocurriría?: si la integridad no se mantuviera, las consecuencias podrían ser bastante graves y afectar tanto a la empresa como a sus clientes de diversas maneras.

Estos solo serian algunos de los problemas que tendrian al no mantener la integridad:

- Instalación de Software Malicioso: Este es quizás el riesgo más directo y peligroso. Si un atacante logra manipular una actualización, podría inyectar código malicioso (malware, spyware, ransomware, etc.). Los usuarios, confiando en que están instalando una actualización legítima, ejecutarían este software malicioso en sus sistemas sin darse cuenta. Esto podría llevar al robo de datos sensibles (contraseñas, información financiera, datos personales), cifrado de archivos para extorsión (ransomware), o el uso de los sistemas comprometidos para lanzar ataques a otros (botnets).

- Compromiso de la Seguridad del Sistema: Incluso si la alteración no introduce malware directamente, podría crear vulnerabilidades de seguridad en el software. Un atacante podría explotar estas debilidades para obtener acceso no autorizado a los sistemas de los usuarios, comprometiendo la confidencialidad, integridad y disponibilidad de sus datos y recursos.

- Problemas de Estabilidad y Funcionalidad: Una actualización manipulada podría introducir errores o incompatibilidades en el software, causando fallos, inestabilidad del sistema, o la pérdida de funcionalidades importantes. Esto afectaría la productividad de los usuarios y generaría frustración, incluso si no hay intenciones maliciosas detrás de la alteración.

2.3. Ejemplo de Disponibilidad: Un banco gestiona sus servicios en línea, como banca digital.
• Aplicación Práctica: El banco implementa servidores redundantes y sistemas de respaldo para asegurar que los clientes puedan acceder a los servicios incluso durante una interrupción o fallo del hardware.

1. Alta disponibilidad (High Availability – HA)

- Uso de clústeres redundantes de servidores: Si un servidor falla, otro asume automáticamente la carga.
- Balanceadores de carga que distribuyen las solicitudes entre varios servidores activos para evitar sobrecargas.

2. Infraestructura redundante

- Componentes duplicados: servidores, fuentes de energía, sistemas de red, bases de datos.
- Centros de datos en configuración activa-activa o activa-pasiva, en distintas ubicaciones físicas.

3. Computación en la nube y servicios híbridos

- Algunos bancos usan soluciones en la nube (como AWS, Azure o Google Cloud) con alta tolerancia a fallos y escalabilidad automática.
- Integración de modelos híbridos: una parte en la nube y otra en centros de datos propios (on-premise), para mayor control y redundancia.

4. Replicación de datos en tiempo real

- Uso de bases de datos replicadas (ej., PostgreSQL con streaming replication, Oracle Data Guard).
- Los datos se sincronizan en tiempo real o casi en tiempo real en múltiples ubicaciones.

5. Planes de continuidad del negocio (BCP) y recuperación ante desastres (DRP)

- Bancos tienen protocolos establecidos para restaurar operaciones rápidamente en caso de desastre.
- Se realizan pruebas periódicas de recuperación simulada.

6. Actualizaciones con tolerancia a fallos (rolling updates)

- Los sistemas se actualizan por partes (nodos, servicios) sin afectar al usuario final.
- Se emplea arquitectura de microservicios con contenedores (ej., Docker, Kubernetes) para aislar fallos.

7. Monitoreo 24/7 y respuesta automatizada

- Monitoreo continuo de infraestructura, rendimiento y disponibilidad.
- Sistemas automáticos de respuesta ante fallos (self-healing systems), que reinician servicios o redirigen tráfico.

8. Sistemas de energía ininterrumpida (UPS) y plantas eléctricas

- Protegen contra cortes de energía, garantizando continuidad operativa.

Ejemplo práctico:
Un banco tiene un sistema de banca en línea que corre en dos centros de datos (uno en Bogotá y otro en Medellín). Si el centro en Bogotá falla por corte eléctrico, el tráfico se redirige automáticamente a Medellín, gracias a:

9. **DNS dinámico:**
   El Sistema de Nombres de Dominio (DNS) es como la libreta de direcciones de Internet, traduciendo nombres de dominio (como www.mibanco.com) a direcciones IP. Un DNS dinámico permite que esta traducción se actualice en tiempo real. En el caso de un fallo en el centro de datos de Bogotá, el DNS dinámico se actualiza automáticamente para dirigir todo el tráfico a la dirección IP del centro de datos de Medellín. Esto es transparente para el usuario, que sigue accediendo a la banca en línea sin necesidad de realizar ninguna acción.

10. **Replicación activa-activa de bases de datos:**
    En una configuración activa-activa, ambas bases de datos (en Bogotá y Medellín) están operativas simultáneamente y reciben las escrituras y lecturas de datos. Esto asegura que, en el momento en que falla el centro de datos de Bogotá, la base de datos en Medellín ya tiene la información más actualizada y puede continuar sirviendo las solicitudes sin pérdida de datos. Esta es una diferencia importante con una configuración activa-pasiva, donde la base de datos secundaria solo se activa en caso de fallo de la primaria.

11. **Monitoreo y conmutación por error automática (failover):**
    Un sistema de monitoreo continuo está vigilando la salud y el rendimiento de la infraestructura en ambos centros de datos. Si detecta un fallo en Bogotá (como el corte de energía), activa automáticamente el proceso de conmutación por error (failover). Este proceso implica redirigir el tráfico (a través del DNS dinámico) y asegurar que los sistemas en Medellín asuman la carga de trabajo sin intervención manual.

Paso 3: Reflexión y Comparación de Conceptos (15 minutos)
Análisis Comparativo:
Actividad: Pide a los participantes que comparen los tres conceptos y cómo se complementan entre sí en un sistema de seguridad completo. Por ejemplo:

Comparacion de conceptos:

-Disponibilidad: Se refiere a la garantía de que los usuarios autorizados puedan acceder a la información y a los recursos del sistema cuando los necesiten. El objetivo principal es asegurar que los servicios estén operativos y funcionales en todo momento.

- Confidencialidad: Se centra en prevenir la divulgación de información sensible a personas o entidades no autorizadas. El objetivo es asegurar que solo aquellos con los permisos adecuados puedan acceder a la informac

- Integridad: Busca asegurar que la información sea precisa, completa y no haya sido alterada sin autorización. El objetivo es mantener la fiabilidad y la consistencia de los datos.

Como se complementan lo terminos:

los tres conceptos son interdependientes y deben implementarse de manera equilibrada para lograr un sistema de seguridad integral

La confidencialidad sin disponibilidad es inútil: Si la información está tan bien protegida que nadie puede acceder a ella, ni siquiera los usuarios autorizados, no cumple su propósito. Un sistema debe ser seguro, pero también utilizable.

La disponibilidad sin confidencialidad es peligrosa: Si la información está accesible para todos, incluyendo personas no autorizadas, se compromete la privacidad y la seguridad de los datos sensibles.

La confidencialidad sin integridad no es confiable: Si la información puede ser modificada sin control, aunque solo la vean personas autorizadas, su valor y fiabilidad se ven comprometidos. La información podría ser incorrecta o manipulada
.

La integridad sin disponibilidad limita la utilidad: Si la información es precisa y protegida contra alteraciones, pero los usuarios no pueden acceder a ella cuando la necesitan, el sistema no cumple su función principal.

Preguntas de Reflexión:

• Pregunta 1: ¿Qué concepto consideras más crítico en una empresa de salud? ¿Y en una empresa de comercio electrónico?

R:
En una empresa de salud considero más crítico la confidencialidad por las siguientes razones:

- **Información sensible**: Las empresas de salud manejan información personal y médica extremadamente sensible de sus pacientes. La divulgación no autorizada de estos datos puede tener consecuencias graves, incluyendo daños a la reputación, discriminación, angustia emocional e incluso riesgos para la seguridad del paciente.

- **Obligaciones legales y éticas:** Existen estrictas regulaciones (como HIPAA en Estados Unidos o leyes de protección de datos en otras regiones) que exigen la protección de la confidencialidad de la información del paciente. Además, existe una fuerte obligación ética por parte de los profesionales de la salud de mantener la privacidad de sus pacientes.

En cuanto a una empresa de comercio electronico considero la integridad como lo más crítico debido a que:

- Transacciones financieras: Las plataformas de comercio electrónico gestionan transacciones financieras y datos de pago de los clientes. Garantizar la integridad de estos datos (que no sean alterados durante la transmisión o el almacenamiento) es fundamental para prevenir fraudes, pérdidas económicas tanto para la empresa como para los clientes, y disputas.

- Información de productos y precios: La integridad de la información sobre los productos (descripciones, especificaciones, disponibilidad) y los precios es esencial para mantener la confianza del cliente y evitar malentendidos o reclamaciones. Si la información se manipula, puede generar una mala experiencia de compra y dañar la reputación de la empresa.

• Pregunta 2: ¿Cómo podrías priorizar la implementación de estos conceptos en una organización con recursos limitados?

Acciones concretas con bajo costo

- Confidencialidad:

Control de accesos (RBAC).
Uso de cifrado básico (datos sensibles en disco y transmisión).
Políticas de contraseñas y concienciación.

- Integridad:

Backups frecuentes y verificados.
Hashes/verificación de archivos críticos.

Registros de auditoría (logs).

- Disponibilidad:

Monitoreo básico de servicios (como UptimeRobot).
Backups redundantes (en nube o local).
UPS o alimentación protegida para servidores críticos.