[Recursos, amenazas y vulnerabilidades](https://www.coursera.org/learn/assets-threats-and-vulnerabilities/home/welcome)

Este contenido proporciona una comprensión fundamental de los riesgos, amenazas y vulnerabilidades en el ámbito de la seguridad de los recursos. Se presentan definiciones claras de cada término y su importancia en la planificación y gestión de la seguridad en las organizaciones.

- Riesgo: Se refiere a todo aquello que puede afectar la confidencialidad, integridad y disponibilidad de un recurso. La evaluación del riesgo implica calcular la probabilidad de eventos negativos y su impacto potencial.

- Amenaza: Circunstancias o eventos que pueden tener un impacto negativo en los recursos, clasificados en intencionadas (como un ataque cibernético) y no intencionadas (como un descuido de un empleado).

- Vulnerabilidad: Debilidades que pueden ser explotadas por amenazas, divididas en categorías técnicas (software mal configurado) y humanas (errores de empleados).

- El ciclo de gestión de riesgos incluye prevenir eventos perjudiciales, identificar mejoras y priorizar recursos críticos.

- Comprender la relación entre riesgos, amenazas y vulnerabilidades es crucial para desarrollar habilidades como analista de seguridad y ganar credibilidad en el sector.

Conclusiones clave:

- La gestión efectiva de la seguridad depende de una clara comunicación y coordinación en la identificación y respuesta a los riesgos.

- Un enfoque sistemático hacia el análisis de riesgos puede ayudar a las organizaciones a proteger sus recursos valiosos.

# **Requisitos comunes de clasificación**

La **Gestión de activos** es el proceso de seguimiento de los activos y de los riesgos que les afectan.

Sólo se puede proteger lo que se sabe que se tiene.

Las organizaciones protegen una gran variedad de recursos diferentes. Algunos ejemplos pueden ser

* Recursos digitales como los Datos de los clientes o los Registros financieros.
* Sistemas de Información que procesan datos, como redes o software.
* Recursos físicos que pueden incluir instalaciones, equipos o suministros.
* Activos intangibles como la Reputación de marca o la Propiedad intelectual.

La Clasificación **de activos** es la práctica de etiquetar los activos en función de su sensibilidad e importancia para una organización.

Deberemos hacernos las siguientes preguntas:

* Qué tiene
* Dónde está
* A quién pertenece, y
* Qué importancia tiene

Niveles de calcificación de los productos valiosos de una organización.

* **Restringido** es el nivel más alto. Esta categoría se reserva para recursos increíblemente sensibles, como la información que es necesario conocer.
* **Confidencial** se refiere a los recursos cuya divulgación puede tener un impacto negativo significativo en una organización.
* **Sólo** interno describe los recursos que están a disposición de los empleados y socios comerciales.
* **El público** es el nivel más bajo de clasificación. Estos recursos no tienen consecuencias negativas para la organización si se divulgan.

La Responsabilidad es sólo una de las características que hacen que clasificar la Información sea un reto. Otra preocupación es que la información puede tener múltiples valores de clasificación al mismo tiempo.

## **Servicios basados en la nube**

Existen tres categorías principales de servicios basados en la nube:

* Software como servicio (SaaS)
* Plataforma como servicio (PaaS)
* Infraestructura como servicio (IaaS)

### **Software como servicio (SaaS)**

SaaS se refiere a las aplicaciones front-end a las que los usuarios acceden a través de un navegador web. Los proveedores de servicios alojan, gestionan y mantienen todos los sistemas back-end para esas aplicaciones. Ejemplos comunes de servicios SaaS incluyen aplicaciones como el servicio de correo electrónico Gmail™, Slack y el software Zoom.

### **Plataforma como servicio (PaaS)**

PaaS se refiere a las herramientas de desarrollo de aplicaciones back-end a las que los clientes pueden acceder en línea. Los desarrolladores utilizan estos recursos para escribir programación y crear, gestionar e implementar sus propias aplicaciones. Mientras tanto, los proveedores de servicios en la nube alojan y mantienen el hardware y el software de back-end que las aplicaciones utilizan para funcionar. Algunos ejemplos de servicios PaaS son la plataforma Google App Engine™, Heroku® y VMware Cloud Foundry.

### **Infraestructura como servicio (IaaS)**

Los clientes de IaaS disponen de acceso remoto a una serie de sistemas back-end alojados por el proveedor de servicios en nube. Esto incluye servidores de proceso de datos, almacenamiento, recursos de redes y mucho más. Los Recursos se licencian comúnmente según las necesidades, por lo que es una alternativa rentable a la compra y el mantenimiento en las instalaciones.

Los servicios basados en la nube permiten a las empresas conectar con sus clientes, empleados y socios comerciales a través de Internet. Algunas de las mayores organizaciones del mundo ofrecen servicios basados en la nube:

* Google Cloud Platform
* Microsoft Azure

La Seguridad de la nube es un subcampo creciente de la ciberseguridad que se centra específicamente en la protección de los datos, las aplicaciones y la infraestructura en la nube.

En la Seguridad de la nube, este concepto se conoce como Modelo de responsabilidad compartida. Por lo general, los clientes son responsables de la seguridad de todo aquello que esté directamente bajo su control:

* Gestión de identidad y acceso
* Configuración de recursos
* Gestión de datos

## **Retos de seguridad de la nube**

La Desconfiguración es una de las mayores preocupaciones. Los clientes de servicios basados en la nube son responsables de configurar su propio entorno de seguridad.

Es más probable que se produzcan violaciones **en la nube nativa** debido a servicios mal configurados.

**Monitorear el acceso puede ser difícil** dependiendo del cliente y el nivel de servicio.

**Cumplir los Estándares regulatorios** también es una preocupación, particularmente en industrias que están obligadas por ley a seguir requisitos específicos como HIPAA, PCI DSS y GDPR.

## **Componentes del CSF**

## **Núcleo**

El núcleo del CSF es un conjunto de resultados de ciberseguridad deseados que ayudan a las organizaciones a personalizar su plan de seguridad. Consta de seis funciones o partes: Identificar, Proteger, Detectar, Responder, Recuperar y Gobernar.

**Niveles**

Los niveles del CSF son una forma de medir la sofisticación del programa de ciberseguridad de una organización. Los niveles del CSF se miden en una escala de 1 a 4. El nivel 1 es la puntuación más baja, lo que indica que se ha implementado un conjunto limitado de controles de seguridad.

**Perfiles**

Los perfiles CSF son plantillas prefabricadas del NIST CSF desarrolladas por un equipo de expertos del sector. Los perfiles CSF se adaptan para abordar los riesgos específicos de una organización o industria. Se utilizan para ayudar a las organizaciones a desarrollar una línea de base para sus planes de ciberseguridad.

## **Implantación del CSF**

Como recordarás, el cumplimiento es un concepto importante en seguridad. El **cumplimiento** es el proceso de adhesión a normas internas y reglamentos externos. En otras palabras, el cumplimiento es una forma de medir lo bien que una organización está protegiendo sus activos. El **Marco de Ciberseguridad del NIST (CSF)** es un marco voluntario que consta de normas, directrices y mejores prácticas para gestionar los riesgos de ciberseguridad.

**Principales recomendaciones por parte del NIST CSF**

* **Crear un perfil actual** de las operaciones de seguridad y esbozar las necesidades específicas de su empresa.
* **Realice una Evaluación de riesgos** para identificar cuáles de sus operaciones actuales cumplen las normas empresariales y reglamentarias.
* **Analizar y priorizar las deficiencias existentes** en las operaciones de seguridad que ponen en peligro los activos de la empresa.
* **Implemente un plan de acción** para alcanzar las metas y objetivos de su organización.

**Principio de privilegio mínimo**

El **principio de privilegio mínimo** es un concepto de Seguridad en el que a un usuario sólo se le concede el nivel mínimo de acceso y autorización necesario para completar una tarea o función.

Este principio se puede cumplir mediante el uso de estas reglas:

* Limitando el acceso a la Información sensible
* Reduciendo las posibilidades de modificación, manipulación o pérdida accidental de Datos
* Apoyando la supervisión y administración de sistemas

## **Determinar el acceso y la autorización**

Para implementar el mínimo privilegio, primero hay que determinar el acceso y la autorización. Para ello hay que plantearse dos preguntas:

* ¿Quién es el usuario?
* ¿Cuánto acceso necesitan a un recurso específico?

Estos son los tipos más comunes de cuentas de usuario:

* Las cuentas de invitado se proporcionan a usuarios externos que necesitan acceder a una red interna, como clientes, contratistas o socios comerciales.
* Las cuentas de usuario se asignan al personal en función de sus funciones laborales.
* Las cuentas de servicio se conceden a aplicaciones o software que necesitan interactuar con otro software de la red.
* Las cuentas con **privilegios** tienen permisos elevados o acceso administrativo.

## **Auditoría de los privilegios de las cuentas**

Existen tres enfoques comunes para auditar las cuentas de usuario:

* Auditorías de uso
* Auditorías de privilegios
* Auditorías de cambio de cuentas

### **Auditorías de uso**

Al realizar una auditoría de uso, el equipo de seguridad revisará a qué recursos accede cada cuenta y qué hace el usuario con el recurso.

### **Auditorías de privilegios**

Los usuarios tienden a acumular con el tiempo más privilegios de acceso de los que necesitan, un problema conocido como *acumulación de privilegios*.

Los servicios de directorio de cuentas mantienen registros y bitácoras asociados a cada usuario. Los cambios en una cuenta suelen guardarse y pueden utilizarse para auditar el Directorio en busca de actividades sospechosas, como múltiples intentos de cambiar la contraseña de una cuenta.

# **Ciclo de vida de los datos**

El ciclo de vida de los datos es un modelo importante que los equipos de seguridad tienen en cuenta a la hora de proteger la información.

En general, el Ciclo de vida de los datos consta de cinco etapas. Cada una de ellas describe cómo fluyen los datos por una organización desde el momento en que se crean hasta que dejan de ser útiles:

* Recopilar
* Almacenar
* Utilizar
* Archivar
* Destruir

## **Gobierno de datos**

Las empresas manejan cantidades ingentes de Datos cada día. Constantemente se recopila nueva Información procedente de fuentes internas y externas. Un enfoque estructurado de la gestión de todos estos datos es la mejor manera de mantenerlos privados y seguros.

Las políticas de Gobierno de datos suelen categorizar a los individuos en un papel específico:

* **Propietario de los** datos: la persona que decide quién puede acceder, editar, utilizar o destruir su Información.
* **Custodio de datos**: cualquier persona o entidad responsable de la manipulación, el transporte y el almacenamiento seguros de la Información.
* **Administrador** de datos: la persona o grupo que mantiene e implementa las políticas de gobierno de datos establecidas por una organización.

Los datos son muy importantes, no son 1s y 0s solo procesados por un ordenador. Si no unos datos que son pertenecientes a un informe médico importante, nómina de un trabajador, compras etc.

**PII** es cualquier información utilizada para inferir la identidad de un individuo.

**PHI** son las siglas en inglés de información sanitaria protegida.

La **SPII** es un tipo específico de PII que se rige por unas directrices de tratamiento más estrictas. La *S* significa sensible, lo que significa que se trata de un tipo de información de identificación personal.

# **Privacidad de la información: Regulaciones y Cumplimiento normativo**

**Seguridad de la Información vs. Privacidad de la Información**

Aunque a menudo se usan indistintamente, la privacidad de la información se centra en el control del individuo sobre sus datos y cómo se comparten. La seguridad de la información (InfoSec), por su parte, trata de proteger esos datos de accesos no autorizados y amenazas. Por ejemplo, una empresa debe revelar cómo usará los datos de un cliente y darle la opción de no participar, y luego implementar controles de seguridad para proteger esos datos. Ambos son cruciales para mantener la confianza del cliente.

**Importancia de la Privacidad en la Seguridad**

La preocupación por la privacidad de los datos surgió a finales de los años 90, cuando las empresas tecnológicas empezaron a recopilar y usar datos de los usuarios a gran escala. Esto llevó a un debate global sobre el derecho de las organizaciones a manejar datos privados y la creciente vulnerabilidad a los abusos y robos de información. Cuantos más datos se recopilan, más expuestos están a las infracciones.

**Regulaciones de Privacidad Notables**

Las regulaciones son normas que las empresas deben cumplir para operar, especialmente en lo que respecta a la privacidad. Protegen al usuario del uso no consentido de su información y establecen medidas de seguridad. Algunas de las regulaciones más influyentes son:

**Reglamento General de Protección de Datos (RGPD)**: Desarrollado por la Unión Europea, da a los individuos control total sobre su información personal y se aplica a cualquier empresa que maneje datos de ciudadanos o residentes de la UE, sin importar su ubicación.

**Estándar de Seguridad de Datos para la Industria de Tarjetas de Pago (PCI DSS)**: Creado por organizaciones financieras, protege las transacciones con tarjeta de crédito y débito del fraude y el robo de datos.

**Ley de Transferencia y Responsabilidad de los Seguros Médicos (HIPAA)**: Una ley estadounidense que exige la protección de la información sanitaria sensible de los pacientes, prohibiendo su divulgación sin consentimiento.

Es importante saber que estas regulaciones, aunque desarrolladas por naciones específicas, influyen en el manejo de datos a nivel global. El cumplimiento normativo es fundamental para todas las empresas.

**Evaluaciones y Auditorías de Seguridad**

Para asegurar el cumplimiento y demostrar el compromiso con la privacidad de los datos, las empresas realizan un proceso continuo de auditorías y evaluaciones de seguridad:

Una auditoría de seguridad es una revisión de los controles, políticas y procedimientos de seguridad de una organización frente a un conjunto de expectativas.

Una evaluación de la seguridad es una comprobación para determinar la resistencia de las implementaciones de seguridad actuales ante las amenazas.

Por ejemplo, una auditoría podría verificar si la autenticación multifactor (MFA) está habilitada, y una evaluación posterior podría identificar contraseñas débiles, llevando a habilitar MFA en todas las cuentas. Las auditorías se hacen menos frecuentemente (anual), mientras que las evaluaciones son más frecuentes (cada tres a seis meses), a menudo internamente como preparación para una auditoría. Ambas son vitales para proteger la privacidad.

**Métodos de encriptación, cifrado simétrico y asimétrico**

* Cifrado: proceso de conversión de datos de un formato legible a un formato codificado
* Infraestructura de clave pública (PKI): marco de cifrado que protege el intercambio de información en línea
* Cifrado: algoritmo que cifra la información

## **Tipos de cifrado**

Existen dos tipos principales de cifrado:

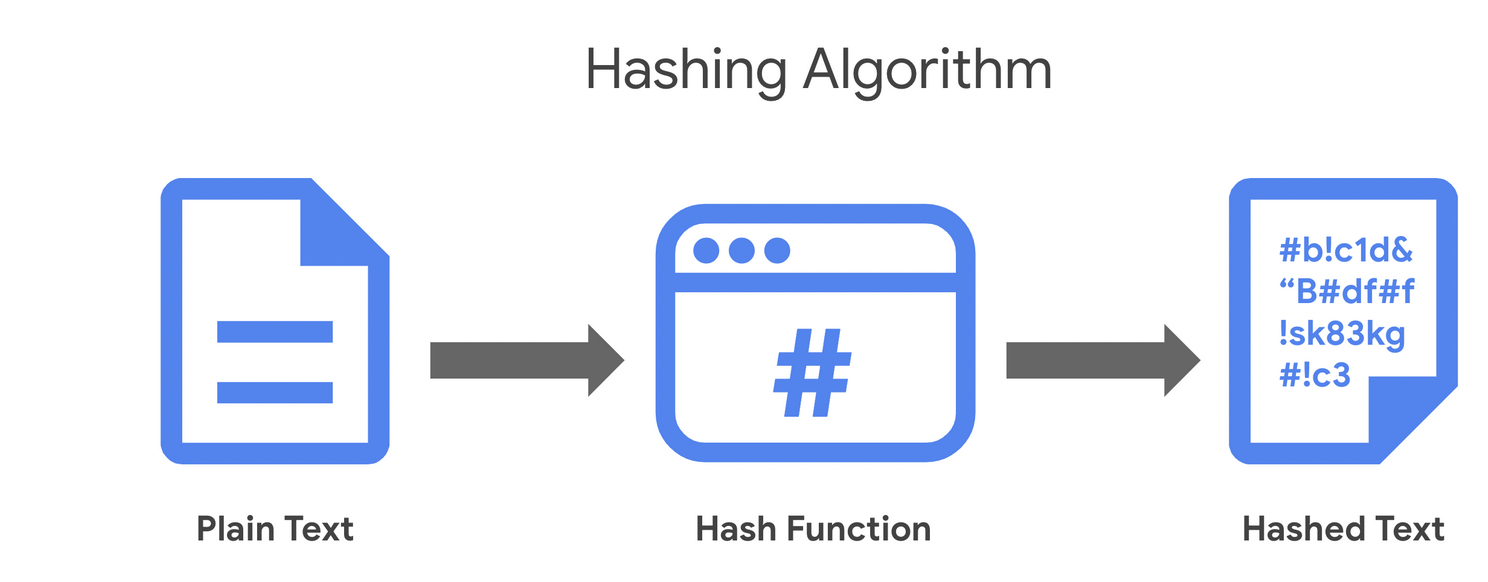
* **Cifrado simétrico**: consiste en utilizar una única clave secreta para intercambiar información. Como utiliza una sola clave para cifrar y descifrar, el emisor y el receptor deben conocer la clave secreta para bloquear o desbloquear el cifrado. Ej: *Triple DES, encriptación estándar*.
* La criptografía **asimétrica** es el uso de un par de claves pública y privada para cifrar y descifrar datos. Utiliza dos claves distintas: una pública y otra privada. La clave pública se utiliza para cifrar los datos y la privada para descifrarlos. La clave privada sólo se entrega a los usuarios con acceso autorizado. Ej: *Rivest Shamir Adleman (RSA*), *Algoritmo de firma digital (DSA).*

Los cifradores son vulnerables a **los ataques de fuerza bruta**, que utilizan un proceso de ensayo y error para descubrir información privada.

En el cifrado moderno, las longitudes de la clave más largas se consideran más seguras.

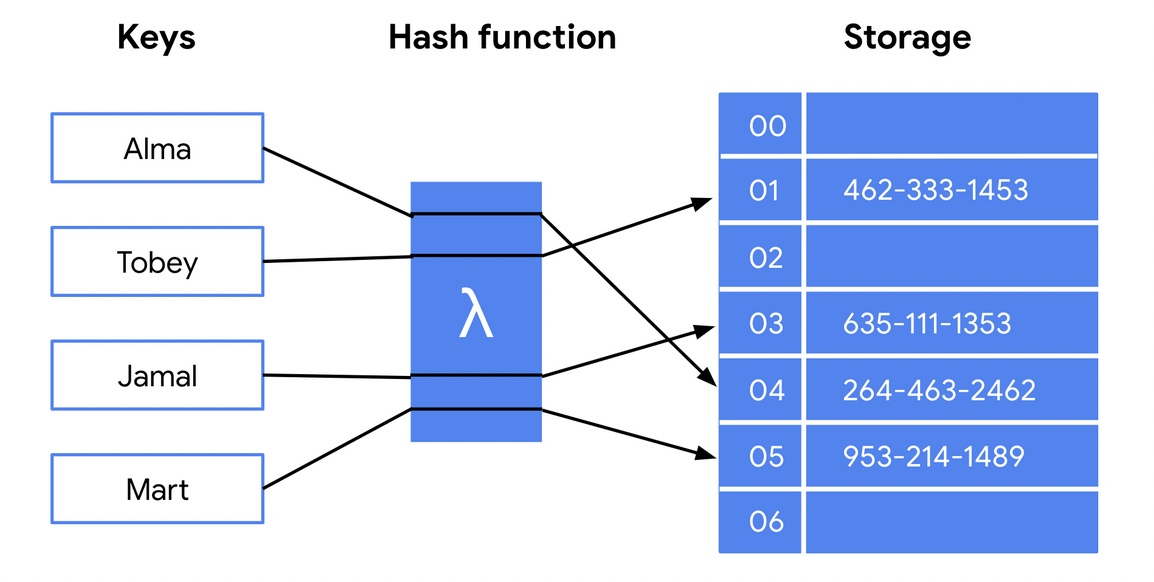
# **La evolución de las funciones hash**

El hash se utiliza ampliamente para la autenticación y el no repudio, el concepto de que no se puede negar la autenticidad de la información.



Utilizando una tabla hash, que es una estructura de datos que se utiliza para almacenar y referenciar valores hash, estos pequeños valores se convirtieron en una forma más segura y eficiente para que las computadoras referenciar datos.

Se desarrolló MD5 a principios de la década de 1990 como una forma de verificar que un archivo enviado a través de una red coincidía con su archivo de origen.



### **Colisiones de Hash: Entendiendo el Problema**

Imaginemos que tenemos una máquina especial que toma cualquier cosa que le demos (un libro, una canción, una foto, una frase corta) y siempre nos da una **etiqueta de identificación** muy específica y de un tamaño fijo. Esta etiqueta está hecha de letras y números. Esa máquina es lo que llamamos una **función hash**.

Ahora, pensemos en la cantidad de cosas diferentes que podemos darle a esa máquina: ¡prácticamente infinitas! Podemos escribir cualquier cosa, desde un "hola" hasta la enciclopedia completa.

Pero la máquina solo puede producir etiquetas de un **tamaño fijo**. Por ejemplo, en el caso de **MD5**, las etiquetas siempre tienen 32 caracteres. Esto significa que hay un número limitado de etiquetas posibles que la máquina puede generar. Es como si tuviéramos un cajón con un número limitado de pegatinas de identificación.

Las deficiencias de MD5 dieron paso a un nuevo grupo de funciones conocidas como algoritmos hash seguros, o SHA.

**Cinco funciones componen la familia de algoritmos SHA:**

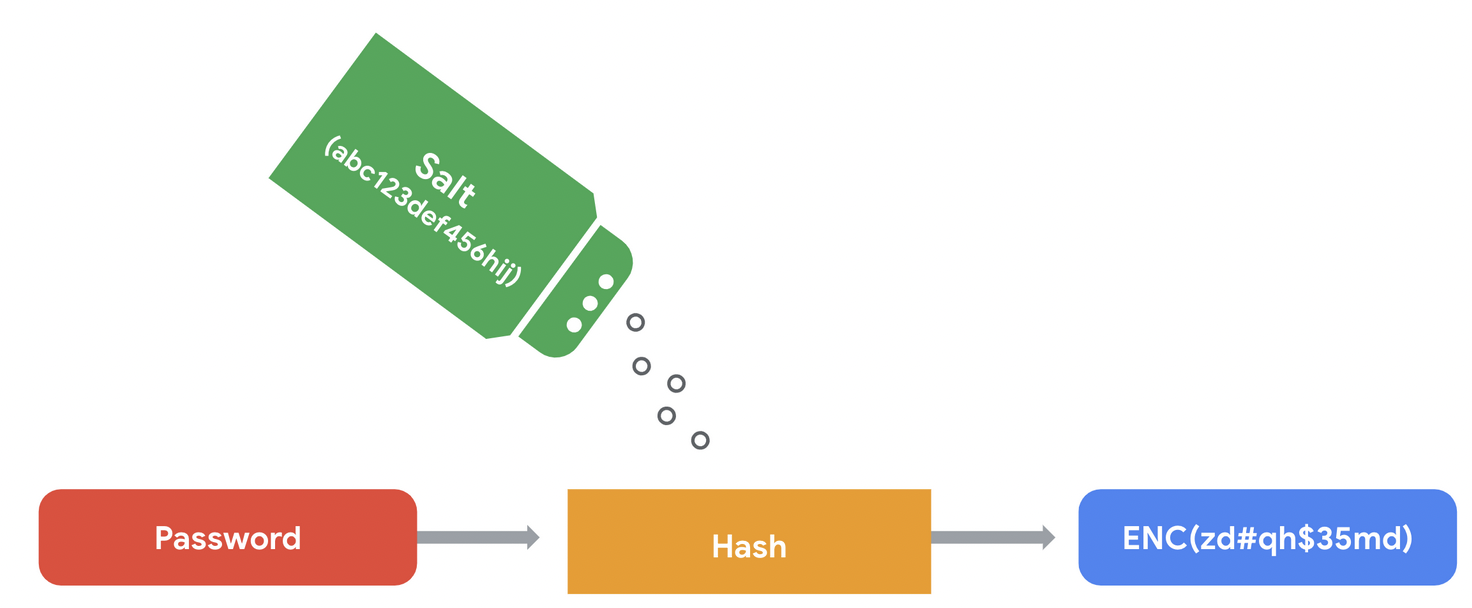
* SHA-1
* SHA-224
* SHA-256
* SHA-384
* SHA-512

Los números son las cantidades de bits que tiene cada método de encriptación.

Si las contraseñas se almacenan en texto plano, un atacante puede robar esa información y utilizarla para acceder a los Recursos de la empresa. El hash añade una capa adicional de Seguridad. Dado que los valores hash no pueden invertirse, un atacante no podría robar las credenciales de inicio de sesión de alguien si consiguiera acceder a la base de datos.

### **Tablas rainbow**

Una **tabla rainbow** es un archivo de valores hash pregenerados y su texto plano asociado. Son como diccionarios de contraseñas débiles.

**El salting** es una salvaguarda adicional que se utiliza para reforzar las funciones hash. Una *sal* es una cadena aleatoria de caracteres que se añade a los Datos antes de hacer el hash. Los caracteres adicionales producen un valor hash más único, lo que hace que los datos salados sean resistentes a los ataques de tabla rainbow.

**Utilice el comando *sha256sum* para generar el hash del archivo*file1.txt***

sha256sum file1.txt

**Utilice el comando cmppara resaltar las diferencias en los archivos *file1hash* y *file2hash***

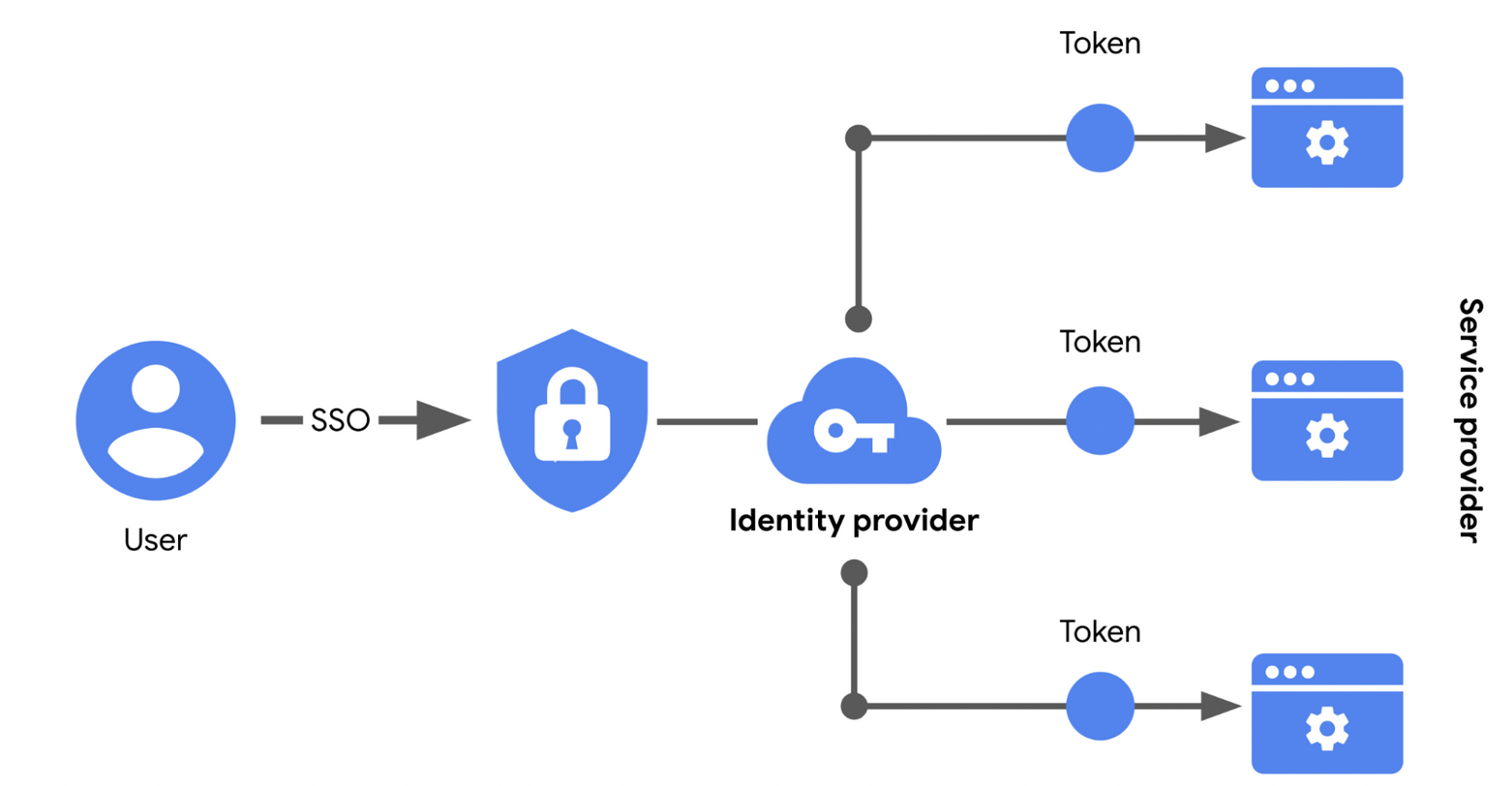
*cmp file1hash file2hash*

# **El auge de SSO y MFA**

**Un mejor enfoque de la autenticación**

El Inicio de sesión único (SSO) es una tecnología que combina varios inicios de sesión diferentes en uno solo.

1. El SSO **mejora la experiencia del usuario** al eliminar el número de nombres de usuario y contraseñas que la gente tiene que recordar.
2. **Las empresas pueden reducir costes** racionalizando la forma en que gestionan los servicios conectados.
3. El **SSO mejora la Seguridad general** al reducir el número de puntos de acceso a los que pueden dirigirse los atacantes.

Las implementaciones de SSO suelen basarse en dos protocolos de autenticación diferentes: LDAP y SAML. LDAP, que significa Protocolo ligero de acceso a directorios, se utiliza sobre todo para transmitir información dentro de las instalaciones; SAML, que significa Lenguaje de marcado de aserción de seguridad, se utiliza sobre todo para transmitir información fuera de las instalaciones, como en la nube.

Con este método nos exponemos a perder nuestra contraseña y que el ciberdelincuente acceda a diferentes servicios. Afortunadamente tenemos solución ante ese problema.

## **MFA al rescate**

La autenticación **de múltiples factores** (MFA) requiere que un usuario verifique su identidad de dos o más formas para acceder a un sistema o red.

## **Refuerzo de la autenticación**

* Algo que un usuario **sabe** : normalmente un nombre de usuario y una contraseña
* Algo que un usuario tiene**:** normalmente recibido de un proveedor de servicios, como un código de acceso de un solo uso (OTP) enviado por SMS
* Algo que un usuario **es** : se refiere a las características físicas de un usuario, como sus huellas dactilares o escáneres faciales

## **Gestión de identificación y accesos**

* El **principio de privilegio mínimo** en el que a un usuario sólo se le concede el nivel mínimo de acceso y autorización requerido para completar una tarea o función.
* **Separación** de funciones, que es el principio según el cual no se debe conceder a los usuarios niveles de autorización que les permitan hacer un uso indebido de un sistema.

En otras palabras, el privilegio mínimo *limita* *el acceso* que recibe un individuo, mientras que la separación de funciones *divide las responsabilidades* entre varias personas para evitar que una sola tenga demasiado control.

## **Gestión de identidad y acceso (IAM)**

La **gestión de identidad y** acceso (IAM) es un conjunto de procesos y tecnologías que ayuda a las organizaciones a gestionar las identidades digitales en su entorno. Tanto los sistemas AAA como los IAM están diseñados para autenticar a los usuarios, determinar sus privilegios de acceso y realizar un seguimiento de sus actividades dentro de un sistema.

## **Autenticación de usuarios**

* **Conocimientos**, o algo que el usuario sabe
* **Responsabilidad**, o algo que el usuario posee
* **Característica**, o algo que el usuario es

### **Aprovisionamiento de usuarios**

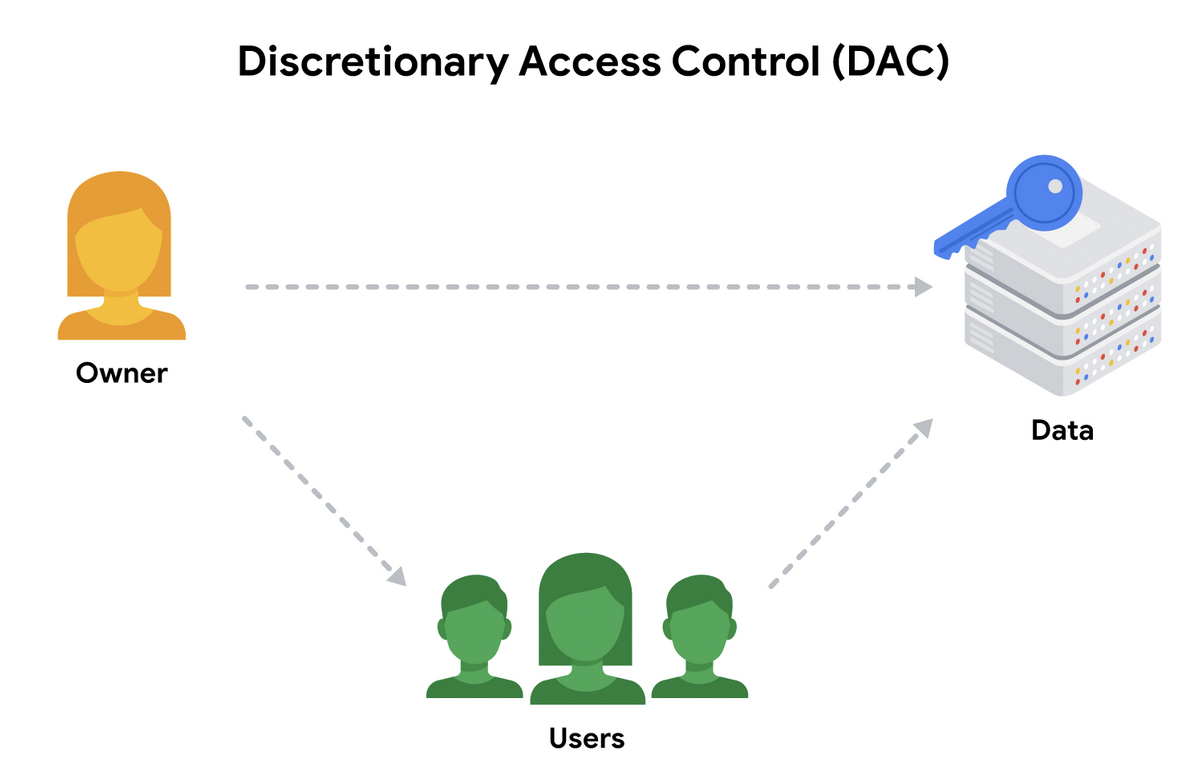
El **Aprovisionamiento** de usuarios es el proceso de creación y mantenimiento de la identidad digital de un usuario.

Otra función que tienen los analistas en IAM es la de desaprovisionar usuarios. Se trata de una práctica importante que elimina los derechos de acceso de un usuario cuando ya no debería tenerlos.

### **Conceder autorización**

## 

### **Control de acceso obligatorio (MAC)**

MAC es el más estricto de los tres framework. La autorización en este Modelo se basa en una estricta necesidad de conocer. El acceso a la información debe ser concedido manualmente por una autoridad central o por el administrador del sistema. 

### **Control de acceso discrecional (DAC)**

El DAC se aplica normalmente cuando el Propietario de los datos decide los niveles apropiados de accesibilidad.

Cuando el propietario de una carpeta de Google Drive comparte el acceso de editor, visualizador o comentarista con otra persona.

### 

### **Control de acceso basado en roles (RBAC)**

El RBAC se utiliza cuando la autorización viene determinada por la función de un usuario dentro de una organización.

Por ejemplo, un usuario del departamento de marketing puede tener acceso a los análisis de datos de los usuarios pero no a la administración de redes.

## **¿Qué es CI/CD y por qué es importante?**

CI/CD automatiza todo el proceso de publicación de software, desde la creación del código hasta su despliegue.

### **Integración continua (IC): Construir una base sólida**

La **integración continua (IC)** consiste en fusionar con frecuencia los cambios de código de distintos desarrolladores en una ubicación central. Esto activa procesos automatizados como la creación de software y la ejecución de pruebas.

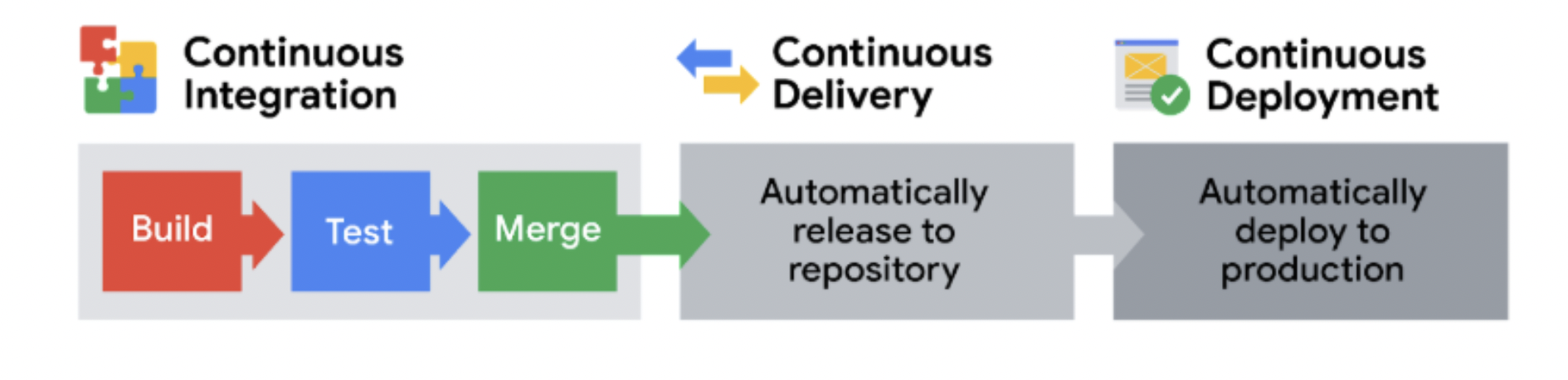
La integración continua detecta los problemas mediante un proceso automatizado: cada vez que se integra el código, el sistema lo construye y lo prueba automáticamente.

### **Entrega continua (CD): Listo para Publicar**

La **entrega continua** significa que el código está siempre listo para ser distribuido a los usuarios. Una vez superadas las pruebas automatizadas, el código se despliega automáticamente en un entorno de pruebas.

### **Despliegue continuo (CD): Lanzamientos totalmente automatizados**

La **implantación continua** automatiza todo el proceso de publicación. Los cambios que superan todas las comprobaciones automatizadas se despliegan automáticamente en el entorno de producción, sin aprobación manual. Se trata de velocidad y eficacia.



## **Por qué es innegociable contar con canalizaciones de CI/CD seguras**

**Automatización segura:** CI/CD automatiza las tareas repetitivas: construcción, pruebas, despliegue. Cuando la automatización se implementa de forma segura, se reducen los errores del trabajo manual, se agilizan los procesos y, lo que es más importante, se reducen los errores humanos que crean vulnerabilidades.

**Mejora de la calidad del código mediante comprobaciones de seguridad:** Las pruebas automatizadas en CI/CD comprueban rigurosamente el código antes de su publicación.

**Menor tiempo de comercialización de las actualizaciones de seguridad:** CI/CD acelera las versiones. Esto permite una entrega más rápida de nuevas funciones, correcciones de errores *y actualizaciones de seguridad*, mejorando el tiempo de respuesta tanto a las necesidades de los usuarios como a las amenazas de seguridad.

**Colaboración y retroalimentación mejoradas con enfoque de seguridad:** CI/CD fomenta la colaboración entre los equipos de desarrollo, seguridad, pruebas y operaciones.

**Riesgo reducido:** las versiones frecuentes y más pequeñas, resultado de CI/CD, son menos arriesgadas que las versiones grandes y poco frecuentes.

En esencia, CI/CD es el motor del desarrollo ágil de software moderno. Permite una entrega de software fiable, eficiente y con capacidad de respuesta.

### **Dependencias inseguras: Riesgos del código de terceros**

Los procesos de CI/CD suelen utilizar muchas bibliotecas y componentes de terceros. Si estos componentes tienen vulnerabilidades conocidas (Vulnerabilidades y exposiciones comunes, o CVE), esas vulnerabilidades pueden añadirse sin saberlo a su aplicación durante el proceso de compilación automatizado.

**Paso a seguir:** Analice y actualice regularmente sus dependencias.

### **Permisos mal configurados: Control de acceso**

Acceso no autorizado puede permitir a los atacantes modificar código, configuraciones de canalización, o inyectar contenido malicioso.

**Paso a seguir:** Implantar una sólida gestión de accesos mediante el control de acceso basado en roles (RBAC).

Aquí tiene un resumen de lo que nos ha compartido sobre la seguridad en CI/CD:

## **Fortalecer la seguridad en CI/CD: Un enfoque esencial**

Nos parece fundamental la integración de la **seguridad** en el proceso de **Integración Continua/Despliegue Continuo (CI/CD)** para proteger las aplicaciones y la infraestructura. No abordar las vulnerabilidades comunes en esta etapa puede llevar a la publicación de software con fallos y a mayores costos de reparación.

### **Vulnerabilidades clave y soluciones**

Hemos identificado tres áreas críticas que a menudo se pasan por alto:

* **Falta de pruebas de seguridad automatizadas:** No incluir herramientas como **SAST (Static Application Security Testing)** y **DAST (Dynamic Application Security Testing)** en el proceso CI/CD es una omisión grave. Esto casi garantiza la publicación de software con vulnerabilidades que solo se detectarán post-lanzamiento, lo que implica un mayor costo y esfuerzo para solucionarlas.
  + **Paso a seguir:** Integrar las pruebas de seguridad automatizadas (SAST y DAST) directamente en el proceso CI/CD, ya que son una parte esencial de una estrategia CI/CD segura.
* **Secretos expuestos:** Codificar datos sensibles como claves de API, contraseñas y tokens directamente en el código o en la configuración del canal es un riesgo significativo. Si estos secretos se exponen, pueden dar lugar a graves brechas de seguridad.
  + **Paso a seguir:** Nunca codificar secretos. Utilizar bóvedas seguras o herramientas de gestión de secretos específicas para almacenar y gestionar información confidencial.
* **Entornos de compilación inseguros:** El propio entorno de CI/CD debe ser seguro. Si este entorno es vulnerable, los atacantes pueden comprometerlo para alterar compilaciones, inyectar código malicioso o robar datos sensibles.
  + **Paso a seguir:** Reforzar los entornos de compilación utilizando contenedores o máquinas virtuales seguras para minimizar el riesgo de compromiso.

### **Creación de un canal de CI/CD seguro: Defensa en profundidad**

Para abordar estas vulnerabilidades de forma proactiva, es crucial adoptar un enfoque de seguridad por capas. A continuación, presentamos las mejores prácticas esenciales para una estrategia de seguridad CI/CD robusta:

* **Integrar la seguridad desde el principio (DevSecOps):** Adoptar una mentalidad DevSecOps implica integrar la seguridad en todas las fases del desarrollo, desde la planificación hasta la implementación.
* **Controles de acceso estrictos:** Utilizar políticas de permisos estrictas basadas en el principio de privilegio mínimo. Conceder solo el acceso necesario al código, los ajustes de la canalización y las configuraciones de implementación. Herramientas como la **Autenticación de Múltiples Factores (MFA)** y el **Control de Acceso Basado en Roles (RBAC)** son clave para proteger el entorno de CI/CD.
* **Automatizar las pruebas de seguridad en todas partes:** Convertir los análisis y pruebas de seguridad automatizados (SAST, **SCA (Software Composition Analysis)** y DAST) en una parte fundamental del proceso de creación y despliegue.
* **Mantener actualizadas las dependencias:** Mantener un inventario actualizado de todas las dependencias, bibliotecas y complementos de CI/CD de terceros. Actualizar periódicamente estos componentes para corregir vulnerabilidades. Herramientas como Dependabot y Snyk pueden automatizar la gestión de dependencias.
* **Gestión segura de secretos:** Nunca codificar información sensible en el código o las configuraciones de la canalización. Exigir el uso de herramientas dedicadas de gestión de secretos como HashiCorp Vault o AWS Secrets Manager para almacenar, acceder y rotar los secretos de forma segura.

### **Conclusión: CI/CD seguro - Software seguro**

Creemos que al abordar proactivamente estas vulnerabilidades e implementar las mejores prácticas de seguridad en el canal de CI/CD, los equipos de software pueden crear y publicar aplicaciones con una postura de seguridad significativamente más fuerte. Una base segura de CI/CD es crucial para minimizar los riesgos de seguridad y construir una estrategia de seguridad general más resistente para las aplicaciones y la infraestructura.

La esencia de la seguridad de su canal de CI/CD es proporcionar una seguridad sólida a su proceso de publicación de software, permitiendo a los ingenieros desarrollar, probar e implementar código con confianza y resiliencia frente a las amenazas.

**OWASP Top 10**

OWASP es una plataforma abierta que los profesionales de la seguridad de todo el mundo utilizan para compartir información, herramientas y eventos centrados en la seguridad de la Web.

## **El OWASP Top 10**

La organización lleva publicando esta lista desde 2003 como una forma de dar a conocer las vulnerabilidades más específicas de la Web. El Top 10 se aplica principalmente a software nuevo o hecho a medida.

### **Control de acceso roto**

Controles de acceso limitan lo que los usuarios pueden hacer en una aplicación web.

También pueden dar a alguien acceso no autorizado a otras aplicaciones empresariales.

### **Fallos criptográficos**

Pueden producirse vulnerabilidades cuando las empresas no encriptan elementos como la información de identificación personal (PII). Por ejemplo, si una aplicación web utiliza un algoritmo de hash débil, como MD5, corre más riesgo de sufrir una violación de datos.

### **Inyección**

### La inyección se produce cuando se inserta un código malicioso en una aplicación vulnerable. Aunque la aplicación parece funcionar con normalidad, hace cosas que no estaba previsto que hiciera.

Los ataques de inyección pueden proporcionar a los agentes de amenaza una puerta trasera al sistema de información de una organización. Un objetivo común es el formulario de inicio de sesión de un sitio web.

### **Diseño inseguro**

Las aplicaciones deben diseñarse de forma que sean resistentes a los ataques. Cuando no lo están, son mucho más vulnerables a amenazas como los ataques de inyección o las infecciones por software malicioso.

### **Desconfiguración de la seguridad**

Los errores de configuración se producen cuando los ajustes de Seguridad no se establecen o mantienen adecuadamente. Las empresas utilizan una gran variedad de sistemas diferentes interconectados. A menudo se producen errores cuando esos sistemas no se configuran o auditan adecuadamente.

### **Componentes vulnerables y obsoletos**

Los componentes vulnerables y obsoletos es una categoría que se refiere principalmente al desarrollo de aplicaciones. En lugar de codificarlo todo desde cero, la mayoría de los desarrolladores utilizan bibliotecas de código abierto para completar sus proyectos de forma más rápida y sencilla.

### **Fallos de identificación y autenticación**

Identificación es la palabra clave en esta categoría de vulnerabilidad. Cuando las aplicaciones fallan a la hora de reconocer quién debe tener acceso y qué está autorizado a hacer, pueden surgir problemas graves.

### **Fallos de software e integridad de datos**

Los fallos de integridad del software y de los datos son instancias en las que las actualizaciones o los parches no se revisan adecuadamente antes de implementarlos. Los atacantes podrían explotar estos puntos débiles para distribuir software malicioso.

### **Fallos en el registro y el monitoreo de la Seguridad**

En Seguridad, es importante poder registrar y rastrear los eventos. Disponer de un registro A de eventos como los intentos de inicio de sesión de los usuarios es fundamental para encontrar y solucionar problemas. Monitorear y responder ante incidentes es igualmente importante.

### **Falsificación de peticiones del lado del servidor**

Las falsificaciones de peticiones del lado del servidor (SSRF) se producen cuando los atacantes manipulan las operaciones normales de un servidor para leer o actualizar otros recursos en ese servidor.

Son posibles cuando una aplicación en el servidor es vulnerable. La aplicación vulnerable puede transportar código malicioso al servidor anfitrión que obtendrá datos no autorizados.

# **Inteligencia de fuentes abiertas**

*Información* se refiere a la recopilación de datos brutos o hechos sobre un sujeto específico. *La Inteligencia*, en cambio, se refiere al análisis de la Información para producir Conocimientos o estadísticas que puedan utilizarse para apoyar la toma de decisiones.

## **La inteligencia mejora la toma de decisiones**

OSINT desempeña una función importante en la **seguridad de la información (InfoSec)**, que es la práctica de mantener los datos en todos los estados alejados de usuarios no autorizados.

Estas son algunas de las formas en que se puede utilizar la OSINT para generar inteligencia:

* Para proporcionar estadísticas sobre ciberataques
* Para detectar posibles exposiciones de Datos
* Para evaluar las defensas existentes
* Para identificar vulnerabilidades desconocidas

# **Enfoques para la exploración de vulnerabilidades**

Un **escáner de vulnerabilidades** es un software que compara automáticamente las vulnerabilidades y exposiciones conocidas con las tecnologías de la red.

Las herramientas de escaneo de vulnerabilidades, buscan las mismas en las siguientes capas que se muestran:

1. Capa de perímetro,como los sistemas de autenticación que validan la accesibilidad de los usuarios
2. **Capa de** red,que se compone de tecnologías como firewalls de red y otras
3. Capa De **punto** final,que describe los dispositivos de una red, como ordenadores portátiles, de sobremesa o servidores
4. **Capa de** aplicación,que implica el software con el que interactúan los usuarios
5. Capa de datos, que incluye cualquier información almacenada, en tránsito o en uso

En su lugar, simplemente escanean una superficie y le alertan de cualquier puerta potencialmente desbloqueada en sus sistemas.

Los escaneos externos prueban la capa perimetral fuera de la red interna.

Los escaneos*internos* parten del extremo opuesto, examinando los sistemas internos de una organización.

### **Autenticación frente a no autenticación**

Los escaneos autenticados y no autenticadossimulan si un usuario tiene o no acceso a un sistema.

Los escaneos autenticados pueden probar un sistema registrándose con una cuenta de usuario real o incluso con una cuenta de administrador.

Los escaneos no autenticados simulan agentes de amenaza externos que no tienen acceso a los recursos de su empresa.

### **Limitado frente a exhaustivo**

Los escaneos limitados y exhaustivos se centran en dispositivos concretos a los que acceden usuarios internos y externos.

Los escaneos *limitados* analizan dispositivos concretos de una red, como la búsqueda de errores de configuración en un firewall.

Los escaneados *exhaustivos* analizan todos los dispositivos conectados a una red. Esto incluye sistemas operativos, bases de datos de usuarios, etc.

## **Parchear las brechas de seguridad**

Una **actualización de parche** es una actualización de software y del sistema operativo que aborda las vulnerabilidades de seguridad dentro de un programa o producto.

Idealmente, los parches abordan las vulnerabilidades y exposiciones comunes antes de que los hackers maliciosos las encuentren. Sin embargo, los parches se desarrollan a veces como resultado de un **Día cero**, que es un exploit desconocido hasta entonces.

Existen dos tipos de actualizaciones, automáticas y manuales.

Hemos analizado la importancia de mantener el software actualizado y las diferencias entre las **actualizaciones manuales y automáticas**, así como los riesgos del **software al final de su vida útil (EOL)**.

### **Estrategias de actualización: Manual vs. Automática**

Comprendemos que existen dos enfoques principales para la implementación de actualizaciones:

* **Actualizaciones manuales:**
  + **Funcionamiento:** Requieren que los departamentos de TI o los usuarios busquen, descarguen e instalen las actualizaciones. En entornos empresariales, a menudo se gestiona con herramientas de administración de configuraciones.
  + **Ventaja:** Ofrecen **control**, lo cual es útil si las actualizaciones no han sido probadas a fondo por los desarrolladores y podrían causar inestabilidad.
  + **Desventaja:** Las actualizaciones críticas pueden olvidarse o ignorarse, dejando los sistemas vulnerables.
* **Actualizaciones automáticas:**
  + **Funcionamiento:** El sistema o la aplicación se encargan de buscar, descargar e instalar las actualizaciones por sí solos. La Agencia de Ciberseguridad y Seguridad de las Infraestructuras (CISA) recomienda usar esta opción siempre que sea posible. Requiere la habilitación de ciertos permisos.
  + **Ventaja:** Simplifican el proceso de implementación y mantienen los sistemas y el software al día con los últimos parches de seguridad.
  + **Desventaja:** Pueden generar problemas de inestabilidad si los parches no han sido probados a fondo por el proveedor.

### **El riesgo del software al final de su vida útil (EOL)**

Sabemos que todo software tiene un ciclo de vida. Cuando una versión más reciente se lanza, la anterior se convierte en **software EOL**, lo que significa que el fabricante deja de darle soporte.

* **Riesgos:** CISA recomienda encarecidamente dejar de usar el software EOL, ya que presenta un riesgo irreparable para los sistemas, puesto que no recibe más parches de seguridad. Estos riesgos aumentan con la proliferación de dispositivos conectados, como los de Internet de las Cosas (IoT). Un solo dispositivo sin parches puede ser una puerta de entrada para un atacante.
* **Desafío:** Reemplazar el software EOL puede ser costoso tanto para empresas como para usuarios individuales, lo que a menudo lleva a que esta recomendación no se siga.

### **Conclusión: La importancia de las actualizaciones**

Podemos confirmar que **actualizar el software y parchear las vulnerabilidades** es una práctica de seguridad fundamental que, lamentablemente, no siempre se cumple. Muchos de los ciberataques más grandes del mundo, como el de **WannaCry en 2017**, que causó daños estimados en 4.000 millones de dólares y afectó a computadoras en más de 150 países, podrían haberse evitado si los sistemas hubieran estado actualizados con parches de seguridad disponibles meses antes.

Entendemos que mantener el software actualizado requiere esfuerzo, pero los beneficios en términos de seguridad y prevención de incidentes hacen que valga la pena.

Hemos recopilado información sobre cómo los equipos de seguridad evalúan la efectividad de sus defensas mediante las **pruebas de penetración**.

### **Pruebas de Penetración: Simulando ataques para fortalecer defensas**

Sabemos que un plan de seguridad eficaz se basa en pruebas periódicas para identificar las debilidades de una organización. Mientras que las **evaluaciones de vulnerabilidad** revisan internamente los sistemas de seguridad, las **pruebas de penetración** (o "pen tests") van un paso más allá. Son ataques simulados autorizados que imitan las herramientas y técnicas de los actores maliciosos para identificar y explotar vulnerabilidades en sistemas, redes, sitios web, aplicaciones y procesos.

A diferencia de una evaluación de vulnerabilidades que solo encuentra puntos débiles, una prueba de penetración los explota para determinar las posibles consecuencias si el sistema fuera comprometido por un atacante real. Por ejemplo, un equipo de ciberseguridad en una empresa financiera podría simular un ataque a su aplicación bancaria para encontrar errores de configuración que permitirían el robo de información o transferencias ilegales. Es importante destacar que las organizaciones reguladas por normativas como **PCI DSS, HIPAA o GDPR** deben realizar pruebas de penetración de forma rutinaria para mantener los estándares de cumplimiento.

### **Perspectivas y estrategias en las pruebas de penetración**

Entendemos que estas pruebas autorizadas son realizadas por "pen testers" expertos en programación y arquitectura de redes. Dependiendo de sus objetivos, las organizaciones pueden utilizar diferentes enfoques:

* **Pruebas del Equipo Rojo:** Simulan ataques para identificar vulnerabilidades.
* **Pruebas del Equipo Azul:** Se enfocan en la defensa y la respuesta ante incidentes para validar los sistemas de seguridad existentes.
* **Pruebas del Equipo Púrpura:** Son colaborativas, combinando elementos de los ejercicios de los equipos rojo y azul para mejorar la postura de seguridad.

Los expertos en pruebas de penetración deben decidir cuánto acceso e información necesitan antes de simular un ataque. Para ello, existen tres estrategias comunes:

* **Pruebas de caja abierta (o caja blanca/conocimiento pleno):** El evaluador tiene acceso privilegiado a información interna, similar a un desarrollador.
* **Pruebas de caja cerrada (o caja negra/conocimiento cero):** El probador tiene poco o ningún acceso a los sistemas internos, imitando a un hacker malicioso. Estas pruebas suelen producir las simulaciones más precisas de un ataque real.
* **Pruebas de conocimiento parcial (o caja gris):** El probador tiene acceso y conocimientos limitados del sistema interno, como un representante de atención al cliente.

Cada estrategia ofrece resultados valiosos al demostrar cómo un atacante podría infiltrarse en un sistema y a qué información podría acceder.

### **Convertirse en un probador de penetración y programas de recompensas**

Hemos identificado que los "pen testers" están en alta demanda en el campo de la ciberseguridad. Las habilidades esenciales para esta carrera incluyen:

* Seguridad de redes y aplicaciones.
* Experiencia con sistemas operativos (ej. Linux).
* Análisis de vulnerabilidad y modelado de amenazas.
* Herramientas de detección y respuesta.
* Lenguajes de programación (ej. Python y BASH).
* Habilidades de comunicación.

Los conocimientos de programación son particularmente útiles. Con práctica y dedicación, los profesionales de ciberseguridad de cualquier nivel pueden desarrollar las habilidades necesarias para ser un "pen tester".

Además, las organizaciones a menudo ofrecen **programas de recompensas por errores** ("bug bounty programs"), que recompensan económicamente a los "pen testers" independientes por encontrar y reportar vulnerabilidades en sus productos. Estos programas son una excelente oportunidad para que los profesionales de la seguridad aficionados participen y mejoren sus habilidades. Podemos mencionar plataformas como **HackerOne** como una comunidad donde encontrar este tipo de oportunidades.

### **Conclusión**

Podemos decir que un riesgo significativo para las organizaciones son los hackers maliciosos que irrumpen en sus sistemas. Las pruebas de penetración son una herramienta crucial que permite a los equipos de seguridad obtener una comprensión clara de los puntos débiles de sus defensas. Existe una necesidad creciente de profesionales especializados en seguridad en este campo. Incluso si se empieza ayudando en estas actividades, hay muchas oportunidades para crecer y aprender las habilidades necesarias para convertirse en un experto en pruebas de penetración.

# **Enfoque la ciberseguridad con mentalidad de atacante**

**Aplicando una mentalidad de atacante para simular amenazas.**

Los equipos de seguridad a menudo realizan simulaciones como parte de su estrategia de gestión de vulnerabilidades. Una forma efectiva de hacerlo es aplicando una mentalidad de atacante a los puntos débiles detectados.

Adoptar esta mentalidad ofrece una perspectiva diferente sobre los desafíos de seguridad y genera estadísticas valiosas para establecer o modificar un plan de seguridad.

**Simulaciones proactivas (ejercicios de equipo rojo):** Asumen el rol de un atacante que explota vulnerabilidades y traspasa defensas.

**Simulaciones reactivas (ejercicios de equipo azul):** Asumen el rol de un defensor que responde a un ataque.

**Escaneado en busca de problemas y soluciones innovadoras.**

**Un ejemplo de un ejercicio de simulación reactiva podría seguir los pasos de una evaluación de vulnerabilidad:**

1. Identificación: Marcar un servidor vulnerable por ejecutar un sistema operativo obsoleto.
2. Análisis de la vulnerabilidad: Investigar el sistema operativo anticuado y sus vulnerabilidades.
3. Evaluación de riesgos: Puntuación de la gravedad de cada vulnerabilidad y evaluación del impacto de no solucionarla.
4. Remediación: Utilizar la información recopilada para solucionar el problema.

Muchos controles de seguridad actuales fueron creados como respuesta reactiva a riesgos, ya que los delincuentes buscan continuamente formas de eludir las defensas existentes.

Recursos como la **Base de Datos Nacional sobre Vulnerabilidades (NVD) del NIST** pueden ser de gran ayuda para mantenerse actualizados sobre las vulnerabilidades más comunes.

**Tipos de amenazas**

**Agentes de amenaza**

Los Agentes de amenaza se dividen normalmente en cinco categorías en función de sus motivaciones:

* Los competidores se refieren a empresas rivales que suponen una amenaza porque podrían beneficiarse de la filtración de información.
* Los actores estatales son las agencias de inteligencia gubernamentales.
* Los Sindicatos criminales se refieren a grupos organizados de personas que ganan dinero con actividades delictivas.
* Amenaza interna puede ser cualquier individuo que tenga o haya tenido autorización para acceder a los Recursos de una organización. Esto incluye a empleados que accidentalmente comprometen recursos o a individuos que los ponen en peligro a propósito para su propio beneficio.
* La TI en la sombra se refiere a los individuos que utilizan tecnologías que carecen de Gobierno de TI. Un ejemplo común es cuando un empleado utiliza su correo electrónico personal para enviar comunicaciones relacionadas con el trabajo.

### **Tipos de hackers**

Dado que la definición formal de hacker es amplia, el término puede resultar un poco ambiguo. En Seguridad, se aplica a tres tipos de individuos en función de su intención:

1. Hackers no autorizados
2. Hackers autorizados, o éticos
3. Hackers semi autorizados

Un hacker no autorizado, o hacker poco ético, es un individuo que utiliza sus habilidades de programación para cometer crímenes. Los hackers no autorizados también son conocidos como hackers maliciosos.

Los hackers autorizados o éticos se refieren a individuos que utilizan sus habilidades de programación para mejorar la seguridad general de una organización.

Los hackers semiautorizados suelen referirse a individuos que podrían violar las normas éticas, pero no se consideran maliciosos. Por ejemplo: Hacktivista

Una amenaza persistente avanzada se refiere a los casos en que un actor de amenaza mantiene el acceso no autorizado a un sistema durante un período prolongado.

**Access Points**

**Acceso directo**, refiriéndonos a las instancias en las que tienen acceso físico a un sistema.

**Soportes extraíbles**, que incluyen hardware portátil, como las unidades flash USB.

**Plataformas de redes sociales** que se utilizan para la comunicación y el intercambio de contenido.

**Correo electrónico**, incluyendo tanto cuentas personales como de empresa.

**Redes inalámbricas en las instalaciones.** Servicios en la nube generalmente proporcionados por organizaciones de terceros.

**Cadenas de suministro** como los proveedores externos que pueden presentar una puerta trasera a los sistemas.

Herramientas usadas por los hackers:

**Ataques de fuerza bruta**

Los Ataques *de fuerza bruta simples* son un método en el que los atacantes adivinan las credenciales de inicio de sesión de un usuario.

*Los ataques de diccionario* son una técnica similar, salvo que en estos casos los atacantes utilizan una lista de credenciales de uso común para acceder a un sistema.

Los Ataques *de fuerza bruta inversa* son similares a los ataques de diccionario, salvo que comienzan con una única credencial y la prueban en varios sistemas hasta encontrar una coincidencia.

El relleno de credenciales es una táctica en la que los atacantes utilizan credenciales de inicio de sesión robadas de anteriores violaciones de datos para acceder a cuentas de usuarios en otra organización.

**Herramientas para romper contraseñas:**

* **Aircrack-ng**
* **Hashcat**
* **John el Destripador**
* **Ophcrack**
* **THC Hydra**

## **Medidas de prevención**

* **Hashing y salting**
* **Autenticación de múltiples factores (MFA)**
* **CAPTCHA**
* **Políticas de contraseñas**

### **Hashing y salting**

El hash convierte la información en un valor único que puede utilizarse para determinar su integridad. El salting es una salvaguarda adicional que se utiliza para reforzar las funciones hash.

### **Autenticación de múltiples factores (MFA)**

La autenticación **de múltiples** factores (MFA) es una medida de seguridad que requiere que un usuario verifique su identidad de dos o más formas para acceder a un sistema o red.

### **CAPTCHA**

CAPTCHA son las siglas de Completely Automated Public Turing test to tell Computers and Humans Apart (Prueba de Turing Pública Completamente Automatizada para distinguir ordenadores y humanos).

### **Política de contraseñas**

Las organizaciones utilizan estos controles de gestión para estandarizar las buenas prácticas de contraseñas en toda su empresa.

Los ataques de fuerza bruta son formas sencillas pero fiables de obtener acceso no autorizado a los sistemas.

# **Tácticas de ingeniería social**

La **ingeniería social** es una técnica de manipulación que explota el error humano para obtener información privada, acceso u objetos de valor.

## **Señales de un ataque**

**Cebo** es una táctica de ingeniería social que tienta a las personas para que comprometan su Seguridad. Un ejemplo común es el USB baiting que se basa en que alguien encuentre una unidad USB infectada y la conecte a su dispositivo.

**Phishing** es el uso de las comunicaciones digitales para engañar a la gente para que revele Datos sensibles o implemente software malicioso. Es una de las formas más comunes de ingeniería social, que suele realizarse a través del correo electrónico.

**Quid pro** quo es un tipo de cebo utilizado para engañar a alguien haciéndole creer que será recompensado a cambio de compartir accesibilidad, Información o dinero. Por ejemplo, un atacante puede hacerse pasar por un agente de préstamos de un banco y llamar a los clientes ofreciéndoles un tipo de interés más bajo en su tarjeta de crédito. Les dirá a los clientes que sólo tienen que proporcionar los datos de su cuenta para reclamar el trato.

**El tailgating** es una táctica de ingeniería social en la que personas no autorizadas siguen a una persona autorizada hasta una zona restringida. Esta técnica también se conoce a veces como piggybacking.

Ataque de **"agujero de** agua" es un tipo de ataque en el que un agente de amenaza compromete un sitio web visitado con frecuencia por un grupo específico de usuarios.

## **Fomentar la precaución**

**Manténgase** alerta ante comunicaciones sospechosas y personas desconocidas, especialmente cuando se trate de correo electrónico.

**Sea precavido** a la hora de compartir Información.

**Controle la curiosidad** cuando algo parezca demasiado bueno para ser cierto.

Lo ideal es que el Entrenamiento en Seguridad se extienda más allá de los empleados. Educar a los clientes sobre las amenazas de la ingeniería social también es una clave para mitigarlas. Basta un acto de amabilidad o un lapsus momentáneo de juicio para que un ataque funcione.

# **Tipos de phishing**

El Phishing **por correo electrónico** es un tipo de ataque enviado a través del correo electrónico en el que los agentes de amenaza envían mensajes haciéndose pasar por una persona o entidad de confianza.

El Smishinges un tipo de phishing que utiliza el Servicio de Mensajes Cortos (SMS), una tecnología que permite enviar mensajes de texto. El smishing abarca todas las formas de servicios de mensajería de texto, incluidos iMessages de Apple, WhatsApp y otros medios de chat en los teléfonos.

El Vhishingse refiere al uso de llamadas o mensajes de voz para engañar a los objetivos para que proporcionen información personal por teléfono.

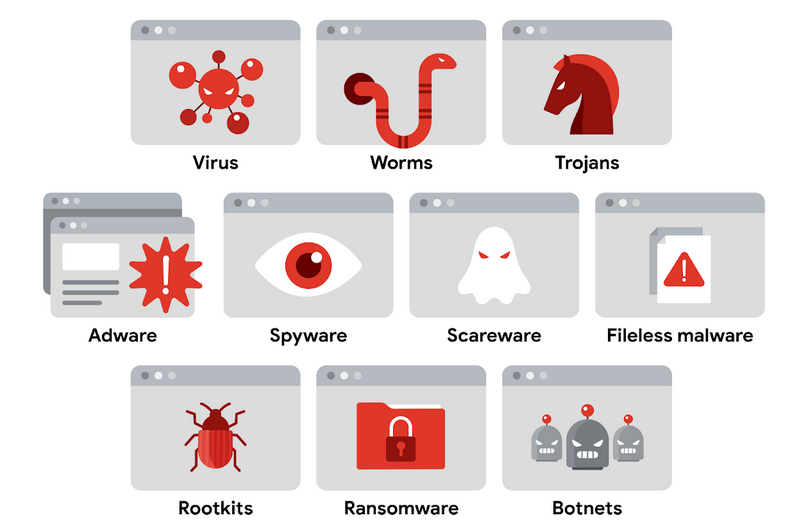
El Spear **phishing** es un subconjunto del phishing por correo electrónico en el que se ataca a propósito a personas concretas, como los contables de una pequeña empresa.

**Whaling** se refiere a una categoría de intentos de phishing dirigidos a altos ejecutivos de una organización.

El **angler phishing** es una técnica en la que los atacantes se hacen pasar por representantes de atención al cliente en las redes sociales.

Introducción al Malware

Las **PUA** son **Aplicaciones Potencialmente No Deseadas**



## **Virus**

Un **virus** es un código malicioso escrito para interferir en el funcionamiento de las computadoras y causar daños en los datos y el software. Este tipo de software malicioso debe ser instalado por el usuario objetivo antes de que pueda propagarse y causar daños.

## **Gusano**

Un **gusano** es un software malicioso que puede replicarse y propagarse por los sistemas por sí solo.

## **Troyano**

Un troyano, también llamado **Caballo de Troya**, es un software malicioso que parece un archivo o programa legítimo.

Al igual que los virus, los atacantes distribuyen este tipo de software malicioso oculto en descargas de archivos y aplicaciones.

## **Adware**

El software con anuncios, o **adware**, es un tipo de software legítimo que a veces se utiliza para mostrar anuncios digitales en las aplicaciones.

El adware malicioso pertenece a una subcategoría de software malicioso conocida como **aplicación potencialmente no deseada (PUA)**. Una PUA es un tipo de software no deseado que se incluye junto con programas legítimos y que puede mostrar anuncios, ralentizar el dispositivo o instalar otro software.

## **Spyware**

El Software **espía** es un software malicioso que se utiliza para recopilar y vender Información sin consentimiento. También se considera una PUA. El spyware suele estar oculto en *bundle ware*, software adicional que a veces se empaqueta con otras aplicaciones.

## **Scareware**

Otro tipo de PUA es el **scareware**. Este tipo de software malicioso emplea tácticas para asustar a los usuarios y conseguir que infecten su propio dispositivo. El scareware engaña a los usuarios mostrándoles advertencias falsas que parecen proceder de empresas legítimas.

## **Malware sin archivos**

El**malware sin** archivos no necesita ser instalado por el usuario porque utiliza programas legítimos que ya están instalados para infectar una computadora. Este tipo de infección reside en la memoria, donde el software malicioso nunca toca el disco duro. Esto es diferente a los otros tipos de software malicioso, que se almacenan dentro de un archivo en el disco.

## **Rootkits**

Un **rootkit** es un software malicioso que proporciona acceso remoto y administrativo a una computadora. La mayoría de los atacantes utilizan los rootkits para abrir una puerta trasera a los sistemas. Vienen acompañados de un dropper.

Un **dropper** es un tipo de software malicioso que viene empaquetado con código malicioso que se entrega e instala en un sistema objetivo.

Si el usuario abre el programa dropper, se ejecuta su código malicioso y se oculta en el sistema de destino. Luego tenemos los cargadores.

Un **cargador** es un tipo de software malicioso que descarga cepas de código malicioso de una fuente externa y las instala en un sistema objetivo. Los atacantes pueden utilizar los cargadores para distintos fines, como para instalar otro tipo de software malicioso: una botnet.

## **Botnet**

Una **botnet**, abreviatura de "red de robots", es una colección de computadoras infectadas por software malicioso que están bajo el control de un único agente de amenaza, conocido como "bot-herder" A menudo se utilizan virus, gusanos y troyanos para propagar la infección inicial y convertir los dispositivos en un bot para el bot-herder. A continuación, el atacante utiliza protocolos de intercambio de archivos, correo electrónico o aplicaciones de Redes sociales para crear nuevos bots y hacer crecer la botnet.

## **Ransomware**

El ransomware describe un ataque malicioso en el que los actores de la amenaza encriptan los datos de una organización y exigen un pago para restaurar la accesibilidad.

# **Prevenir ataques de inyección**

## **Consultas SQL**

Una **base** de datos es una colección organizada de información o datos en un solo lugar.

## **Categorías de inyección de SQL**

Existen tres categorías principales de inyección de SQL:

* Dentro de banda
* Fuera de banda
* Inferencial

### **Inyección de SQL en banda**

La inyección de SQL en banda, o clásica, es el tipo más común. Una inyección en banda es aquella que utiliza el *mismo canal de comunicación* para lanzar el ataque y recoger los resultados.

Por ejemplo, esto podría ocurrir en el cuadro de búsqueda de la página web de un minorista que permite a los clientes encontrar productos para comprar.

### **Inyección de SQL fuera de banda**

Una inyección fuera de banda es aquella que utiliza un *canal de comunicación diferente* para lanzar el ataque y recoger los resultados.

Por ejemplo, un atacante podría utilizar una consulta maliciosa para crear una conexión entre un sitio web vulnerable y una base de datos que controle.

### **Inyección de SQL inferencial**

La inyección de SQL inferencial se produce cuando un atacante no puede ver directamente los resultados de su ataque. En su lugar, pueden interpretar los resultados analizando el *comportamiento* del sistema.

Por ejemplo, un atacante puede realizar un ataque de inyección de SQL en el formulario de inicio de sesión de un sitio web que haga que el sistema responda con un mensaje de error.

## **Prevención de la inyección**

Las consultas SQL se programan a menudo dando por sentado que los usuarios sólo introducirán información relevante.

Una clave para prevenir los ataques de inyección de SQL es *escapar* de las entradas del usuario *, impidiendo*que alguien inserte cualquier código que un programa no esté esperando.

* **Sentencias preparadas**: una técnica de programación que ejecuta sentencias SQL antes de pasarlas a una base de datos
* **Saneamiento** de entradas: programación que elimina las entradas del usuario que podrían interpretarse como código.
* **Validación de** entrada: programación que garantiza que la entrada del usuario cumple las expectativas de un sistema.

# **Rasgos de un modelo de amenaza eficaz**

El Modelado **de amenazas** es el proceso de identificación de los recursos, sus vulnerabilidades y el modo en que cada uno de ellos está expuesto a las amenazas.

### **La importancia de la seguridad en las aplicaciones**

En la actualidad, las aplicaciones son fundamentales para el éxito de muchas organizaciones. Las **aplicaciones web** facilitan la conexión global entre empresas, socios y clientes, mientras que las **aplicaciones móviles** son la principal vía para el intercambio de datos entre usuarios y empresas.

Debido al gran volumen de información que estas aplicaciones procesan, su seguridad es esencial para mitigar los riesgos para todos los involucrados. Un ejemplo claro es la vulnerabilidad **Log4Shell (CVE-2021-44228)** en bibliotecas de registro basadas en Java. Sin un parche adecuado, esta vulnerabilidad permitiría la **ejecución remota de código**, dando a los atacantes control total sobre un sistema desde cualquier lugar del mundo. La explotación de una vulnerabilidad crítica como esta podría comprometer millones de dispositivos.

En resumen, la seguridad de las aplicaciones no es solo una buena práctica, sino una necesidad imperativa para proteger los datos y la infraestructura digital de las organizaciones y sus usuarios.

## **Defensa de la capa de aplicación**

Un proceso típico de Modelado de amenazas se realiza en un ciclo:

* Definir el Alcance
* Identificar las amenazas
* Caracterizar el entorno
* Analizar las amenazas
* Mitigar riesgos
* Evaluar los resultados

## **Marcos comunes**

A la hora de realizar el Modelado de amenazas, existen múltiples métodos que se pueden utilizar, como:

* STRIDE
* PASTA
* Trike
* VAST

Las organizaciones pueden utilizar cualquiera de ellos para recopilar información y tomar decisiones para mejorar su postura de Seguridad.

### **STRIDE**

STRIDE es un framework de Modelado de amenazas desarrollado por Microsoft.

El acrónimo representa cada uno de estos vectores: suplantación de identidad, manipulación, repudio, revelación de información, denegación de servicio y elevación de privilegios.

### **PASTA**

El **Proceso de Simulación de Ataques y Análisis de Amenazas** (PASTA) es un proceso de Modelado de Amenazas centrado en el Riesgo desarrollado por dos líderes de OWASP y apoyado por una empresa de ciberseguridad llamada VerSprite.

### **Trike**

Trike es una metodología y una herramienta de código abierto que adopta un enfoque centrado en la Seguridad para el modelado de amenazas.

Se suele utilizar para centrarse en los permisos de seguridad, los casos de uso de las aplicaciones, los modelos de privilegios y otros elementos que sustentan un entorno seguro.

### **VAST**

El framework de Modelado de Amenazas Visual, Ágil y Sencillo (VAST) forma parte de una plataforma automatizada de modelado de amenazas llamada ThreatModeler®. Muchos equipos de Seguridad optan por utilizar VAST como forma de automatizar y agilizar sus evaluaciones de modelado de amenazas.

## **Participar en el Modelado de amenazas**

Una de las claves del Modelado de amenazas es formular las preguntas adecuadas:

* ¿En qué estamos trabajando?
* ¿Qué tipo de cosas pueden salir mal?
* ¿Qué estamos haciendo al respecto?
* ¿Lo hemos abordado todo?
* ¿Hemos hecho un buen trabajo?

# **Funciones en la respuesta**

### Comando, Control y Comunicación en la Respuesta a Incidentes

Para que la respuesta a incidentes de seguridad sea efectiva, es fundamental que haya un \*\*mando, control y comunicación\*\* claros.

\* \*\*Comando:\*\* Se refiere al \*\*liderazgo\*\* y la dirección para supervisar la respuesta.

\* \*\*Control:\*\* Implica la capacidad de \*\*gestionar los aspectos técnicos\*\*, como coordinar recursos y asignar tareas.

\* \*\*Comunicación:\*\* Consiste en \*\*mantener informadas a todas las partes interesadas\*\*.

Todo esto lo gestiona un \*\*Equipo de Respuesta ante Incidentes de Seguridad Informática (CSIRT)\*\*, un grupo de profesionales especializados en manejar y responder a estos eventos. Una estructura organizativa clara dentro del CSIRT es vital para una respuesta eficiente.

---

### Funciones en los CSIRT

Los CSIRT varían según la organización y pueden ser equipos dedicados o grupos de trabajo. Incluyen tanto profesionales de seguridad como de otras áreas (RRHH, legal, TI, etc.) que aportan su experiencia. Las tres funciones clave de seguridad en un CSIRT suelen ser:

\* \*\*Analista de seguridad:\*\* Monitoriza el entorno, analiza y clasifica alertas, investiga la causa raíz y escala amenazas críticas.

\* \*\*Líder técnico:\*\* Gestiona todos los aspectos técnicos de la respuesta, determina la causa raíz, e implementa estrategias de contención, erradicación y recuperación. Colabora con otros equipos para alinear las prioridades de seguridad con las del negocio.

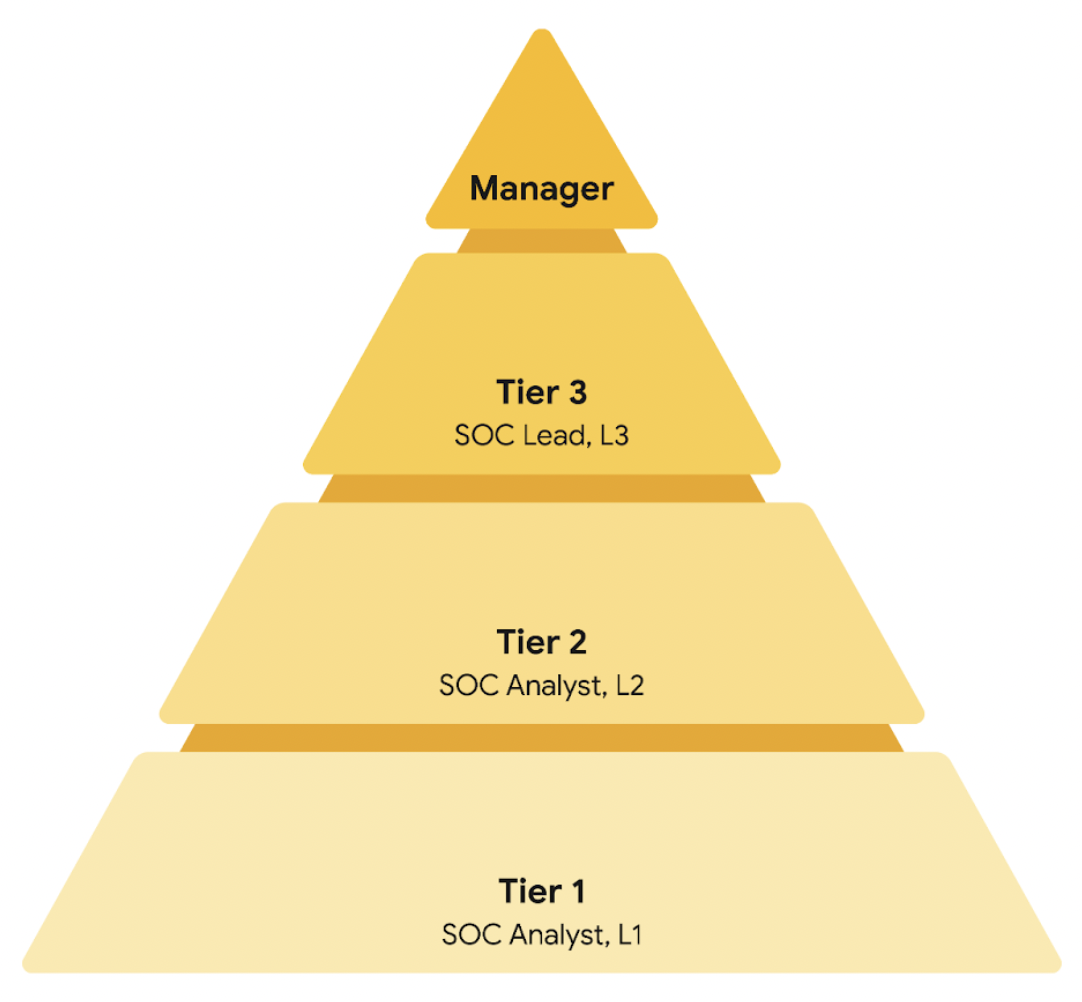
\* \*\*Coordinador de incidentes:\*\* Se encarga de la coordinación con los departamentos no relacionados con la seguridad (legales, comunicaciones, etc.) para asegurar líneas de comunicación claras y mantener a todo el personal informado.

Otros roles específicos pueden existir según la organización.

---

### Centro de Operaciones de Seguridad (SOC)

Un \*\*Centro de Operaciones de Seguridad (SOC)\*\* es una unidad dedicada a \*\*monitorizar redes, sistemas y dispositivos en busca de amenazas\*\*. Los profesionales de un SOC son a menudo conocidos como el \*\*Equipo Azul\*\*, responsables de la defensa contra ataques. El SOC se encarga del monitoreo, análisis y respuesta a incidentes.

La organización típica de un SOC incluye:

\* \*\*Analistas SOC de Nivel 1 (L1):\*\* Son los menos experimentados. Monitorizan, revisan y priorizan alertas, abren/cierran tickets y escalan a niveles superiores.

\* \*\*Analistas SOC de Nivel 2 (L2):\*\* Más experimentados. Investigan a fondo los tickets escalados, configuran y mejoran herramientas de seguridad, e informan al Líder del SOC.

\* \*\*Líder del SOC de Nivel 3 (L3):\*\* Profesionales altamente experimentados. Gestionan las operaciones del equipo, exploran métodos de detección avanzados (análisis de malware, forense) y reportan al Gerente del SOC.

\* \*\*Gerente del SOC:\*\* En la cima de la jerarquía. Responsable de contratar, capacitar y evaluar al equipo, establecer métricas de rendimiento y elaborar informes para la dirección ejecutiva.

También pueden existir otras funciones especializadas en un SOC, como \*\*investigadores forenses\*\* (L2/L3 que analizan pruebas digitales) y \*\*cazadores de amenazas\*\* (L3 que detectan amenazas nuevas y avanzadas usando inteligencia de amenazas). Al igual que los CSIRT, la estructura de un SOC puede variar entre organizaciones.

# **Visión general de las herramientas de Detección**

## **Herramientas de Detección**

## 

## **Visión general de las herramientas IDS**

Un **Sistema de detección de intrusiones (IDS**) es una aplicación que monitorea la actividad del sistema y alerta sobre posibles intrusiones.

Un IDS proporciona un monitoreo continuo de los eventos de red para ayudar a proteger contra amenazas o ataques a la seguridad.

Por ejemplo, un IDS puede enviar una alerta cuando identifica un registro de usuario sospechoso, como una dirección IP desconocida que inicia sesión en una aplicación o un dispositivo a una hora inusual. Pero, un IDS no detendrá ni impedirá ninguna acción posterior, como bloquear el inicio de sesión del usuario sospechoso.

### **Categorías de Detección**

Como analista de Seguridad, investigará las alertas que genera un IDS. Hay cuatro tipos de categorías de Detección con las que debe estar familiarizado:

1. **Un Positivo** verdadero es una alerta que detecta correctamente la presencia de un ataque.
2. Un Negativo verdadero es un estado en el que no se detecta actividad maliciosa. Es cuando no existe actividad maliciosa y no se activa ninguna alerta.
3. Un Falso positivo es una alerta que detecta incorrectamente la presencia de una amenaza. Es cuando un IDS identifica una actividad como maliciosa, pero no lo es. Los falsos positivos son un inconveniente para los Equipos de Seguridad porque gastan tiempo y recursos en investigar una alerta ilegítima.
4. **Un** falso negativo es un estado en el que no se detecta la presencia de una amenaza. Es cuando se produce una actividad maliciosa pero un IDS no logra detectarla. Los falsos negativos son peligrosos porque los Equipos de Seguridad se quedan sin saber de ataques legítimos a los que pueden ser vulnerables.

## **Visión general de las herramientas IPS**

Un **sistema de prevención de intrusiones (IPS**) es una aplicación que monitoriza la actividad del sistema en busca de actividad intrusiva y toma medidas para detenerla. Un IPS funciona de forma similar a un IDS. Pero, el IPS monitoriza la actividad del sistema para detectar y alertar sobre intrusiones, *y* también toma medidas para *prevenir* la actividad y minimizar sus efectos.

## **Visión general de las herramientas EDR**

**Detección y respuesta en el punto final (EDR**) es una aplicación que monitorea un punto final en busca de actividad maliciosa. Las herramientas EDR se instalan en los puntos finales.

# **Visión general de la tecnología SIEM**

## **Ventajas de SIEM**

**Accesibilidad a los Datos de Eventos:** Las herramientas SIEM proporcionan acceso a los datos de eventos y actividades que se producen en una red, incluida la actividad en tiempo real.

**Monitorear, Detectar y Alertar:** Las herramientas SIEM supervisan continuamente los sistemas y las redes en tiempo real.

**Almacenamiento de registros:** Las herramientas SIEM pueden actuar como un sistema de retención de datos, que puede proporcionar accesibilidad a los datos históricos. Los Datos pueden conservarse o borrarse después de un periodo dependiendo de los Requisitos de una organización.

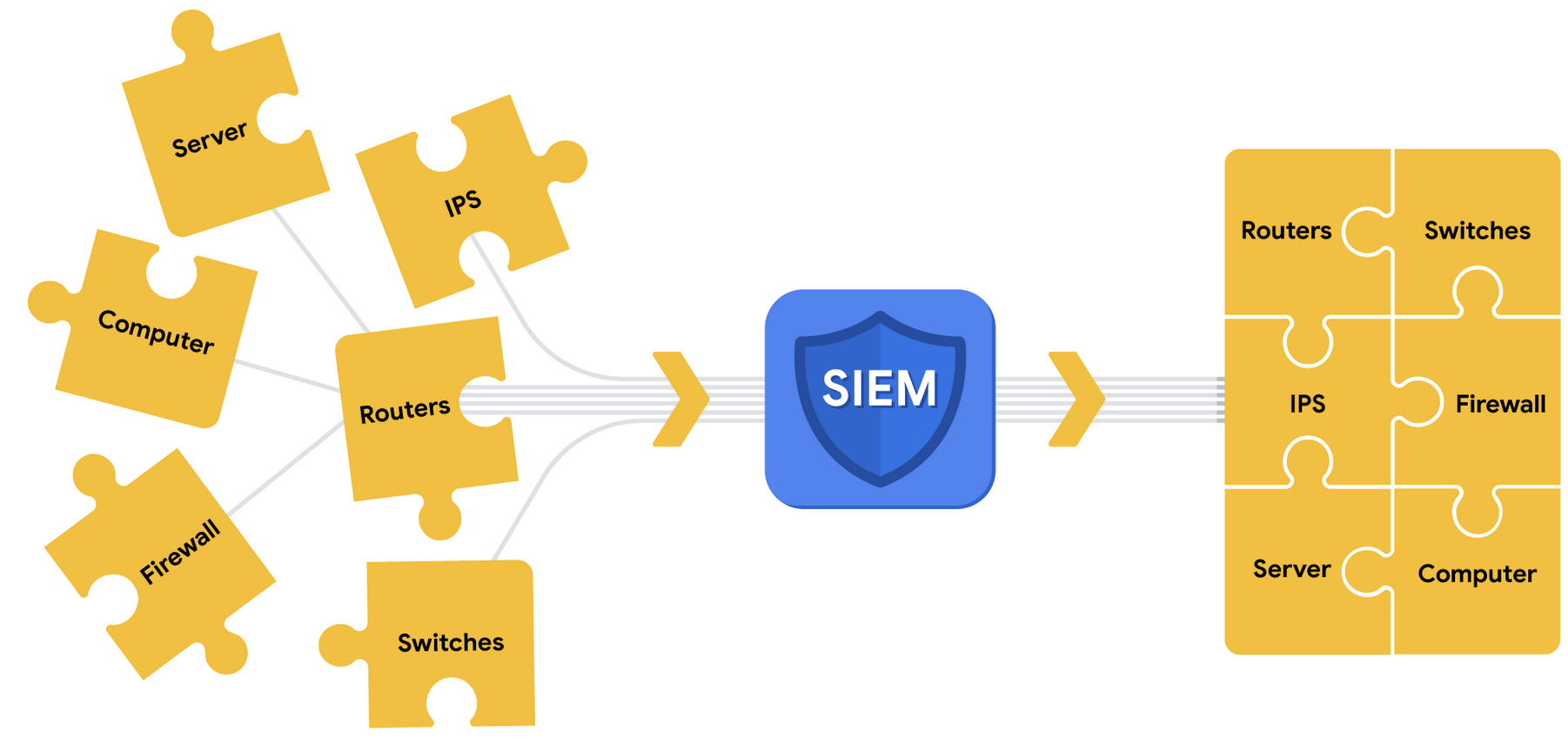
## **El proceso SIEM**

El proceso SIEM consta de tres pasos críticos:

1. **Recopilar y agregar Datos**
2. **Normalización de datos**
3. **Analizar los datos**

En un primer momento, el SIEM **recopila datos de eventos**, también conocidos como **registros**, de diversas fuentes, como firewalls, servidores y routers. Estos registros contienen detalles importantes como marcas de tiempo y direcciones IP.

Una vez que se han recopilado todos estos registros, se **agregan** o consolidan en una ubicación centralizada. Este proceso de recopilación y agregación elimina la necesidad de revisar y analizar manualmente los datos de eventos de fuentes individuales, ya que todos los datos de eventos están disponibles en un solo lugar: el SIEM.



*La normalización* convierte los datos en un formato estándar y estructurado que se puede buscar fácilmente.

El Análisis puede realizarse con algún tipo de lógica de Detección, como un conjunto de reglas y condiciones. A continuación, las herramientas SIEM aplican estas reglas a los datos y, si alguna actividad del registro coincide con una regla, se envían alertas a los equipos de ciberseguridad.

# **Mantenga la concienciación con la supervisión de la red**

## **Conozca su red**

En seguridad, las líneas de base ayudan a establecer un estándar de comportamiento esperado o normal para sistemas, dispositivos y redes. Esencialmente, al conocer la línea de base del comportamiento *normal* de la red, podrá identificar mejor el comportamiento *anormal* de la red.

## **Monitoree su red**

Una vez que haya determinado una línea de base, puede monitorear una red para identificar cualquier desviación de esa línea de base.

**Componentes de red para detectar actividades maliciosas**

### **Análisis de flujo**

Fluir se refiere al movimiento de las comunicaciones de red e incluye información relacionada con paquetes, protocolos y puertos. Los paquetes pueden viajar a puertos, que reciben y transmiten comunicaciones.

Sin embargo, los actores maliciosos pueden utilizar protocolos y puertos que no están comúnmente asociados para mantener comunicaciones entre el sistema comprometido y su propia máquina.

Estas comunicaciones son lo que se conoce como **comando y control (C2)**,que son las técnicas utilizadas por los actores maliciosos para mantener las comunicaciones con los sistemas comprometidos.

### **Información sobre la carga útil de los paquetes**

En los paquetes los datos están encriptados y requieren desencriptación para que sean legibles. Las organizaciones pueden monitorizar la información de la carga útil de los paquetes para descubrir actividades inusuales, como datos sensibles que se transmiten fuera de la red, lo que podría indicar un posible ataque de exfiltración de datos.

### **Patrones temporales**

Los paquetes de red contienen información relativa al tiempo. Esta Información es útil para comprender los patrones temporales.

## **Proteja su red**

Las organizaciones pueden implementar un centro de operaciones de **red** (**NOC**), que es una unidad organizativa que supervisa el rendimiento de una red y responde a cualquier interrupción de la red, como un corte de red.

Los analistas de seguridad monitorean las redes para identificar cualquier signo de posibles incidentes de seguridad conocidos como **Indicadores de compromiso (IoC**).

### **Herramientas de Monitoreo de red**

**Sistemas de detección de intrusiones (IDS**) Monitorean la actividad del sistema y alertan sobre posibles intrusiones. Un IDS detectará y alertará sobre las desviaciones que usted le haya configurado para detectar.

**Los analizadores de protocolos de red**, también conocidos como rastreadores de paquetes, son herramientas diseñadas para capturar y analizar el tráfico de datos dentro de una red. Pueden utilizarse para analizar manualmente las comunicaciones de red en detalle.

# **Más información sobre la captura de paquetes**

## **Paquetes**

Un paquete **de datos** es una unidad básica de información que viaja de un dispositivo a otro dentro de una red.

Los paquetes contienen tres componentes: la cabecera, la carga útil y el pie de página. He aquí una descripción de cada uno de estos componentes.

### **Cabecera**

### **Carga útil**

El componente de carga útil desigue directamente a la cabecera y contiene los datos reales que se envían. Piensa en el ejemplo de subir una imagen a un sitio web; la carga útil de este paquete sería la propia imagen.

### **Pie de página**

El pie de página, también conocido como trailer, se encuentra al final del paquete. El protocolo Ethernet utiliza los pies de página para proporcionar información de comprobación de errores para determinar si los datos se han corrompido. Además, es posible que los paquetes de red Ethernet que se analizan no muestren información de pie de página debido a las configuraciones de red.

## **Analizadores de protocolos de red**

**Los analizadores de protocolos** de red **(packet sniffers)** son herramientas diseñadas para capturar y analizar el tráfico de datos dentro de una red. Algunos ejemplos de analizadores de protocolos de red son tcpdump, Wireshark-

### **Cómo funcionan los analizadores de protocolos de red**

En primer lugar, los paquetes deben recogerse de la red a través de la **Tarjeta de interfaz de red (NIC)**. Configurar antena de red en modo monitor para acceder a todos los tipos de paquetes de la red que estamos analizando.

El analizador de protocolos de red recopila el tráfico de red en formato binario sin procesar y este mismo lo procesa a formato legible.

Activar el modo promiscuo puede exponer nuestra tarjeta de red a atacantes y captar información como datos sensibles o contraseñas.

### **Captura de paquetes**

El sniffing de paquetes es la práctica de capturar e inspeccionar paquetes de datos a través de una red. Una captura de paquetes (p-cap) es un archivo que contiene paquetes de datos interceptados desde una interfaz o red. Podemos filtrar por la información más importante que queramos analizar, por ejemplo paquetes enviados desde una IP en específico.

El uso de analizadores de protocolos de red para interceptar y examinar comunicaciones de red privadas sin permiso se considera ilegal en muchos lugares.

Libcap: Linux, MacOS

WinPcap: Windows

Npcap: Herramienta de Windows

PCAPng: es un formato de archivo moderno que puede capturar paquetes y almacenar datos simultáneamente. Su capacidad para hacer ambas cosas explica la "ng", que significa "próxima generación"

# **Investigar los detalles del Paquete**

## **Protocolo de Internet (IP)**

El **Protocolo de Internet** (IP) incluye un conjunto de estándares utilizados para enrutar y direccionar los paquetes de datos a medida que viajan entre los dispositivos de una red. IP funciona como la base de todas las comunicaciones a través de Internet.

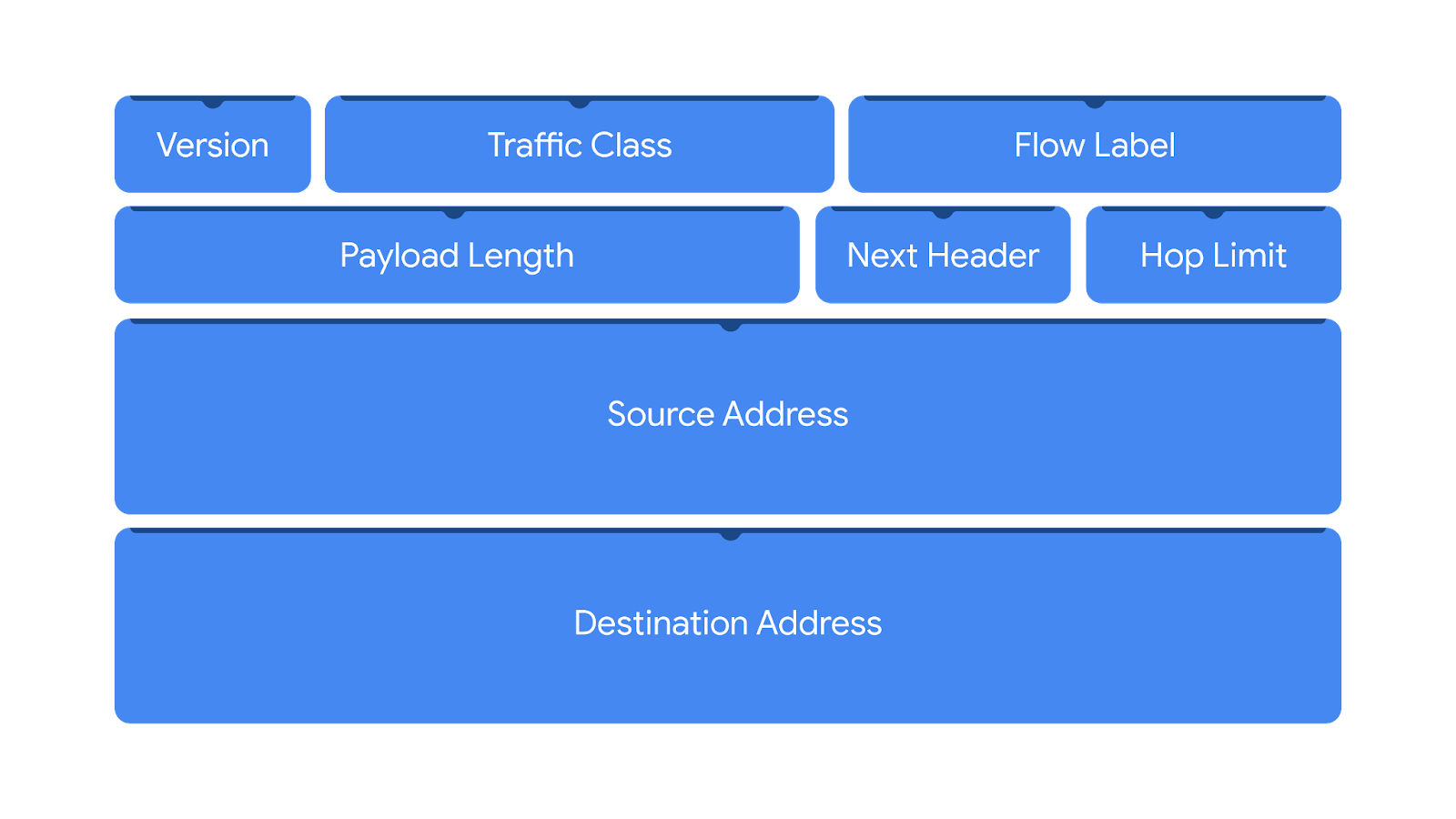
### **IPv4**

IPv4 es la versión de IP más utilizada. Hay trece campos en el encabezado:

* **Versión**: Campo que indica la versión de IP. Para un encabezado IPv4, se utiliza IPv4.
* **Longitud del Encabezado de Internet (IHL**): Este campo especifica la longitud del Encabezado IPv4 incluyendo cualquier Opción.
* **Tipo de Servicio (ToS)**: Este Campo proporciona Información sobre la prioridad de entrega del Paquete.
* **Longitud** Total: Este campo especifica la longitud total de todo el paquete IP incluyendo el Encabezado y los Datos.
* **Identificación**: Los Paquetes que son demasiado grandes para enviarlos se fragmentan en trozos más pequeños. Este Campo especifica un identificador Único para los fragmentos de un Paquete IP original para que puedan ser reensamblados una vez que alcancen su destino.
* **Banderas**: Este Campo proporciona Información sobre la fragmentación del Paquete incluyendo si el Paquete original ha sido fragmentado y si hay más fragmentos en tránsito.
* **Desplazamiento de fragmentación**: Este Campo se utiliza para identificar la secuencia correcta de los fragmentos.
* **Tiempo de vida (TTL**): Este campo limita el tiempo que un paquete puede circular por una red, evitando que los routers reenvíen los paquetes indefinidamente.
* **Protocolo**: Este campo especifica el protocolo utilizado para la parte de datos del paquete.
* Suma de comprobación**del Encabezado**: Este campo especifica un valor de suma de comprobación que se utiliza para la comprobación de errores del Encabezado.
* **Dirección** de origen: Este Campo especifica la dirección de origen del remitente.
* Dirección de**destino**: Este Campo especifica la dirección de destino del receptor.
* **Opciones**: Este Campo es opcional y puede utilizarse para aplicar opciones de Seguridad a un Paquete.

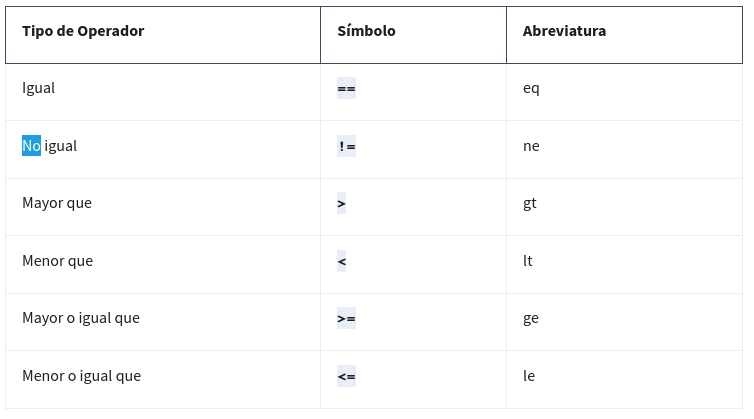
### **IPv6**

La Adopción de IPv6 ha ido en aumento debido a su gran espacio de direcciones. Hay ocho campos en el Encabezado:

* **Versión**: Campo que indica la versión de IP. Para un encabezado IPv6, se utiliza IPv6.
* Clase de**Tráfico**: Este Campo es similar al Campo de Tipo de Servicio IPv4. El Campo de clase de tráfico proporciona información sobre la prioridad o clase del Paquete para ayudar en la entrega del mismo.
* **Etiqueta de Flujo**: Este Campo identifica los paquetes de un Flujo. Un flujo es la secuencia de paquetes enviados desde una fuente específica.
* **Longitud de la carga útil**: Este campo especifica la Longitud de la porción de Datos del paquete.
* **Encabezado siguiente**: Este Campo indica el tipo de Encabezado que sigue al Encabezado IPv6 como TCP.
* **Límite** de salto: Este campo es similar al campo de tiempo de actividad de IPv4. El Límite de Salto limita cuánto tiempo puede viajar un paquete en una red antes de ser descartado.
* **Dirección** de origen: Este Campo especifica la dirección de origen del remitente.
* Dirección de**destino**: Este Campo especifica la dirección de destino del receptor.

**Filtros para Wireshark**

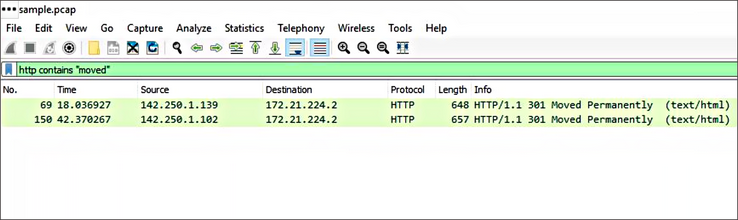
**Operadores**

****

**Puede combinar operadores de comparación con operadores lógicos booleanos como *and* y *or* para crear filtros de visualización complejos.**

### **Operador de contención**

**El operador *contains* se utiliza para filtrar los paquetes que contienen una coincidencia exacta de una cadena de texto.**

****

### **Operador de coincidencias**

**El operador *matches* se utiliza para filtrar paquetes basándose en la expresión regular (regex) especificada.**

**Filtrado de protocolos**

* **dns**
* **http**
* **ftp**
* **ssh**
* **arp**
* **telnet**
* **icmp**

**Filtrar por IP**

Para una ip en específico:

***ip.addr == 172.21.224.2***

IP de origen específica, puede utilizar el filtro *ip.src*.

***ip.src == 10.10.10.10***

IP de destino específica, puede utilizar el filtro *ip.dst*.

***ip.dst == 4.4.4.4***

### **Filtrado para una dirección MAC**

***eth.addr == 00:70:f4:23:18:c4***

### **Filtrado de puertos TCP / UDP**

***udp.port == 53***

***tcp.port == 25***

# **Visión general de tcpdump**

Tcpdump es un analizador de protocolos de red de línea de comandos.

El tráfico que captura es guardado en archivo (p-cap), contiene paquetes de datos interceptados desde una interfaz de red.

**Resumen: Captura de paquetes con tcpdump (con ejemplos incluidos)**

Para capturar tráfico de red con tcpdump, se requieren privilegios de superusuario, ya sea iniciando sesión como root o usando sudo.

### **Sintaxis básica:**

sudo tcpdump [-i interface] [option(s)] [expression(s)]

* sudo tcpdump: ejecuta el comando con permisos elevados.
* -i: especifica la interfaz de red, por ejemplo -i any captura en todas.
* [option(s)]: modifican el comportamiento del comando.
* [expression(s)]: filtran los paquetes capturados (se explica más abajo).

**Nota**: Use -D para listar interfaces de red disponibles.

### **Opciones comunes:**

**-w**: Guarda los paquetes en un archivo en lugar de mostrarlos en pantalla.  
 Ejemplo:  
  
 sudo tcpdump -i any -w packetcapture.pcap

**-r**: Lee un archivo de captura previamente guardado.  
 Ejemplo:  
  
 sudo tcpdump -r packetcapture.pcap

**-v, -vv, -vvv**: Ajusta el nivel de detalle (verbosidad) de la salida.  
 Ejemplo:  
  
 sudo tcpdump -r packetcapture.pcap -v

**-c**: Limita la cantidad de paquetes capturados.  
 Ejemplo:  
  
 sudo tcpdump -i any -c 3

**-n y -nn**: Evitan la resolución de nombres y puertos, lo cual mejora precisión y privacidad.  
 Ejemplo:  
  
 sudo tcpdump -r packetcapture.pcap -v -n

**Consejo**: Se pueden combinar opciones (ej. -vn), pero no si alguna requiere un parámetro.

### **Expresiones de filtrado:**

Permiten aislar paquetes según protocolos, direcciones IP, puertos, etc.

Para capturar solo tráfico IPv6:  
  
 tcpdump ip6

Para combinar condiciones con operadores booleanos (and, or, not):  
  
 sudo tcpdump -r packetcapture.pcap -n 'ip and port 80'

Uso de paréntesis para agrupar:  
  
 ip and (port 80 or port 443)

**Consejo**: Use comillas para asegurar la correcta interpretación de las expresiones.

### **Interpretar la salida:**

Cada línea representa un paquete y empieza con una **marca de tiempo**, seguida de:

* **IP de origen**
* **Puerto de origen**
* **IP de destino**
* **Puerto de destino**

El resto puede incluir detalles como banderas TCP, número de secuencia y más, especialmente si se usa la opción -v.

**Ejemplo de comando:**

sudo tcpdump -i any -v -c 1

Este captura un solo paquete con detalles, útil para análisis manual.

# **Métodos de Detección de Incidentes de Ciberseguridad**

**Detección** se refiere al descubrimiento prompt de los Eventos de Seguridad y **el Análisis** implica la investigación y validación de las alertas.

### **Caza de amenazas**

La **caza de** amenazas es la búsqueda proactiva de amenazas en una red.

Los especialistas en caza de amenazas se conocen como cazadores de amenazas. Los cazadores de amenazas realizan investigaciones sobre amenazas y ataques emergentes y luego determinan la probabilidad de que una organización sea vulnerable a un ataque concreto.

### **Inteligencia sobre amenazas**

**Inteligencia sobre** amenazas, que es información sobre amenazas basada en pruebas que proporciona contexto sobre las amenazas existentes o emergentes.

**La inteligencia sobre amenazas puede proceder de fuentes privadas o públicas como:**

**Informes de la industria**: A menudo incluyen detalles sobre las Tácticas, Técnicas y Procedimientos (TTP) de los atacantes.

**Avisos del Gobierno:** Al igual que los informes de la industria, los avisos gubernamentales incluyen detalles sobre la TTP de los atacantes.

Fuentes de**datos** sobre amenazas: Los feeds de datos sobre amenazas proporcionan un flujo de datos relacionados con las amenazas que pueden utilizarse para ayudar a protegerse contra atacantes sofisticados como las **amenazas persistentes avanzadas (**APT). Las APT son instancias en las que un actor de amenaza mantiene un acceso no autorizado a un sistema durante un largo periodo de tiempo.

### **Ciberengaño**

El engaño cibernético implica técnicas que engañan deliberadamente a los actores maliciosos con el objetivo de aumentar la Detección y mejorar las estrategias defensivas.

**Los Honeypots** son un ejemplo de mecanismo activo de ciberdefensa que utiliza la tecnología del engaño. Los Honeypots son sistemas o recursos que se crean como señuelos vulnerables a los ataques con el propósito de atraer a posibles intrusos.

# **Supervisión continua de CI/CD**

## **Resumen: Monitorización Continua de CI/CD para la Detección Automática de Amenazas**

Para proteger la cadena de suministro de software, la **monitorización continua de las canalizaciones de Integración Continua y Entrega/Despliegue Continuos (CI/CD)** es esencial. Esto va más allá de solo recopilar registros; se trata de usar la automatización para detectar **actividades inusuales** que puedan indicar amenazas de seguridad, permitiendo a los equipos de seguridad responder rápidamente.

### **¿Por qué es crucial la automatización?**

Las canalizaciones CI/CD agilizan la entrega de software, pero también introducen nuevas vulnerabilidades. Si un atacante las compromete, podría inyectar código malicioso, robar datos o interrumpir el funcionamiento del software. Por ello, es vital que la supervisión detecte automáticamente comportamientos anómalos en los procesos de construcción, código o despliegue.

### **Indicadores de Compromiso (IoC) comunes en CI/CD**

Es fundamental conocer los IoC típicos para una supervisión efectiva:

* **Cambios de código no autorizados:** Modificaciones realizadas por personas incorrectas, en momentos o lugares inusuales, o con contenido sospechoso (ej. código confuso, grandes eliminaciones sin justificación).
* **Patrones de despliegue sospechosos:** Despliegues en sistemas no aprobados, fuera de horario, o iniciados por cuentas de usuario inusuales.
* **Dependencias comprometidas:** Descubrimiento de vulnerabilidades conocidas (CVEs), adición repentina de dependencias inesperadas, o intentos de descarga desde fuentes no fiables.
* **Ejecución inusual de la canalización:** Fallos repentinos, tiempos de ejecución excesivamente largos sin razón, o cambios no aprobados en el orden de los pasos.
* **Intentos de exposición de secretos:** Accesos a secretos desde lugares no aprobados o descubrimiento de secretos codificados en el código.

### **Seguridad proactiva mediante la supervisión de IoC**

La monitorización continua de los IoC de CI/CD, con detección automatizada de anomalías, fortalece la seguridad. Esto permite:

* **Respuesta rápida:** Detectar IoC temprano facilita una acción inmediata, deteniendo ataques antes de que causen daños graves.
* **Limitación de daños:** Una respuesta ágil reduce el impacto potencial de un incidente.
* **Mejora del conocimiento de amenazas:** Entender cómo los atacantes actúan en el CI/CD ayuda a mejorar futuras estrategias de seguridad.

### **Métodos de automatización para encontrar anomalías e IoC**

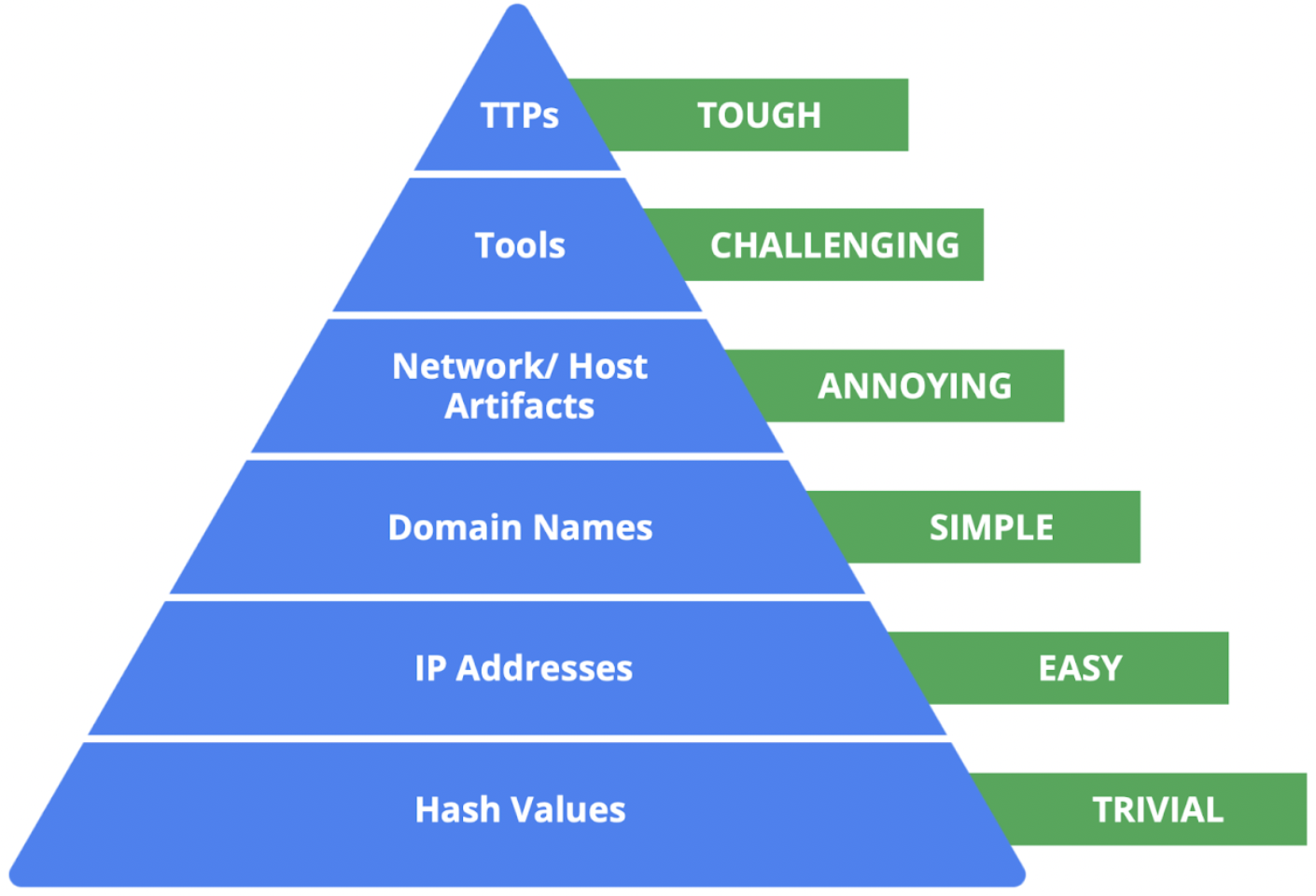
Para detectar automáticamente las amenazas en CI/CD, podemos usar:

* **Registro y auditoría exhaustivos:**
  + **Registros de ejecución de canalizaciones:** Herramientas especializadas establecen una "línea de base" del funcionamiento normal, alertando sobre desviaciones como tiempos de ejecución inusuales, errores inesperados o cambios en el orden de los pasos.
  + **Registros de confirmación de código:** Monitorean cambios sospechosos por autor, hora o contenido.
  + **Registros de acceso:** Identifican inicios de sesión inusuales (desde ubicaciones diferentes, intentos fallidos seguidos de éxito, cambios en configuraciones críticas).
  + **Registros de despliegue:** Alertan sobre despliegues inusuales en horario o ubicación.
* **Integración de la Administración de Información y Eventos de Seguridad (SIEM):** Conectar los registros de CI/CD a un SIEM permite:
  + **Detección automática de anomalías:** Usando aprendizaje automático para encontrar patrones inusuales.
  + **Alertas basadas en reglas:** Configurar reglas para detectar IoC conocidos, como hashes de archivos maliciosos o conexiones a servidores de Comando y Control (C2).
* **Alertas y notificaciones en tiempo real:** Notificaciones inmediatas sobre:
  + Fallos de compilación inusuales.
  + Cambios de código sospechosos.
  + Intentos de exposición de secretos.
  + Tráfico de red inusual desde servidores CI/CD.
* **Monitorización del rendimiento:** Aunque es para el buen funcionamiento, anomalías en el rendimiento (Indicadores de Ataque - IoA) pueden llevar al descubrimiento de IoC.
* **Exploración continua de vulnerabilidades:** Revisar la infraestructura CI/CD (herramientas, plugins, contenedores) en busca de CVEs o debilidades conocidas que puedan ser IoC y necesiten parches inmediatos.

## **Resumen: Indicadores de Compromiso y la Pirámide del Dolor**

Les presentamos el concepto de **Indicadores de Compromiso (IoC)**, que son evidencias observables de un posible incidente de seguridad (ej., un nombre de archivo asociado a malware). Se distinguen de los **Indicadores de Ataque (IoA)**, que son secuencias de eventos que señalan un incidente en tiempo real (ej., un proceso realizando una conexión de red inusual). Mientras los IoC identifican el "quién" y "qué" después de un ataque, los IoA se centran en el "por qué" y "cómo" de un ataque en curso. Es importante recordar que los IoC no siempre confirman un incidente; también pueden ser resultado de errores o fallos del sistema.

### **La Pirámide del Dolor**

El investigador David J. Bianco creó la **Pirámide del Dolor** para ayudar a los profesionales de la seguridad a entender el valor de los diferentes tipos de IoC y la dificultad que su bloqueo impone a los atacantes.

La pirámide jerarquiza los IoC según el "dolor" que causan a un atacante si son bloqueados:

* **Valores hash:** Referencias únicas a archivos maliciosos. Son fáciles de cambiar para los atacantes.
* **Direcciones IP:** Direcciones de red. También son fáciles de modificar.
* **Nombres de dominio:** Direcciones web. Relativamente fáciles de cambiar.
* **Artefactos de red:** Evidencias observables en la red (ej., cadenas User-Agent). Un poco más difíciles de alterar.
* **Artefactos de host:** Evidencias en un dispositivo conectado a la red (ej., nombre de archivo creado por malware). Más difíciles de cambiar.
* **Herramientas:** Software usado por el atacante (ej., John the Ripper). Implica más esfuerzo para el atacante si se bloquea el uso de su herramienta preferida.
* **Tácticas, Técnicas y Procedimientos (TTP):** El comportamiento del atacante (ej., cómo se mueve lateralmente, cómo persiste). Estos son los más difíciles de cambiar para un atacante y, por tanto, los más valiosos para que los equipos de seguridad los detectan y bloquean.

## **Análisis de Indicadores de Compromiso (IoC) con Herramientas de Investigación**

Cuando los analistas de seguridad investigan un incidente, no basta con identificar un único **Indicador de Compromiso (IoC)**, como una dirección IP. Es crucial **añadir contexto** a estos IoC para tener una visión completa del ataque y poder actuar de forma más informada. El contexto se construye buscando otros artefactos relacionados, como comunicaciones de red sospechosas o procesos inusuales.

### **El poder del Crowdsourcing en la Inteligencia de Amenazas**

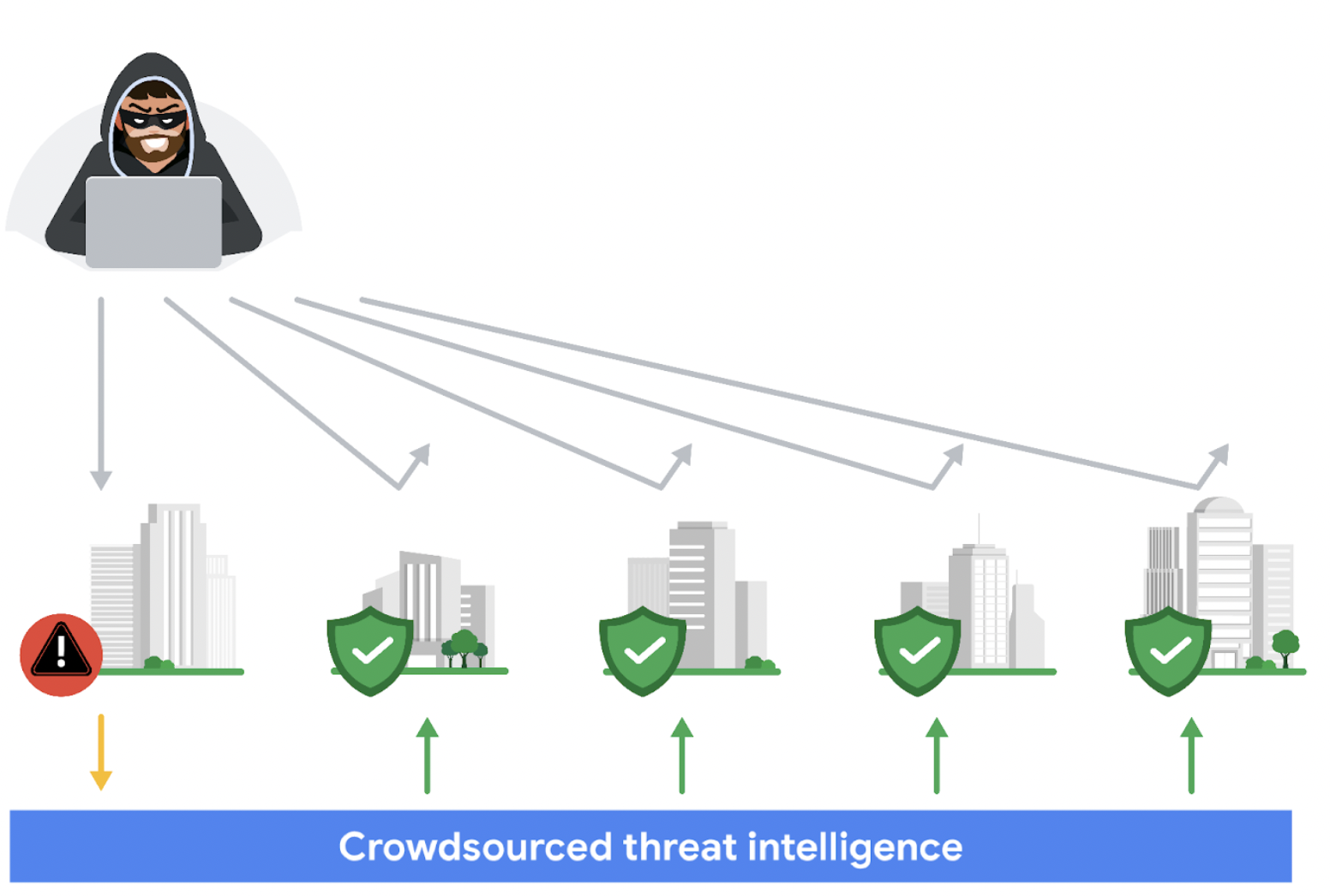
Tradicionalmente, la respuesta a incidentes se realizaba de forma aislada. Sin embargo, el **crowdsourcing** ha revolucionado la ciberseguridad, permitiendo a organizaciones y profesionales de todo el mundo compartir información sobre amenazas. Plataformas de Inteligencia sobre Amenazas, como los **Centros de Análisis e Intercambio de Información (ISAC)** o la **Inteligencia de Fuentes Abiertas (OSINT)**, recopilan y distribuyen estos datos, lo que mejora la detección y defensa de todos. Si una organización sufre un ataque, puede compartir los detalles (ej. archivos maliciosos, IPs, URLs) con la comunidad, ayudando a otras a protegerse.

### **VirusTotal: Una Herramienta Clave para el Análisis de IoC**

**VirusTotal** es un servicio gratuito y público que permite analizar archivos, dominios, URL y direcciones IP sospechosas en busca de contenido malicioso. Al enviar un artefacto, se genera un informe detallado que proporciona información crucial:

* **Detección:** Muestra los veredictos de varios proveedores de seguridad sobre si el IoC es malicioso, sospechoso, etc.
* **Detalles:** Proporciona un análisis estático del IoC, incluyendo hashes, tipos y tamaños de archivo, encabezados, y fechas de creación o envío.
* **Relaciones:** Muestra otros IoC conectados, como URL contactadas, dominios, direcciones IP y archivos descartados.
* **Comportamiento:** Detalla la actividad observada del artefacto en un entorno controlado (sandbox), incluyendo tácticas y técnicas detectadas, comunicaciones de red y acciones en el sistema.
* **Comunidad:** Permite a otros profesionales de la seguridad y usuarios compartir comentarios y estadísticas sobre el IoC.

En la parte superior del informe, podemos ver la **proporción de proveedores** que han marcado el IoC como malicioso, así como una **puntuación de la comunidad**. Cuanto mayor sea el número de detecciones y la puntuación de la comunidad, más probable es que el archivo sea realmente malicioso.

Utilizar estas herramientas de investigación nos permite no solo identificar IoC, sino también comprender el panorama completo de un ataque, lo que nos ayuda a mejorar nuestras estrategias de respuesta y prevención.

# **Buenas prácticas para una documentación eficaz**

**Documentación** es cualquier forma de contenido registrado que se utiliza para un fin específico, y es esencial en el Campo de la Seguridad. Los Equipos de Seguridad utilizan la documentación para apoyar las investigaciones, completar las tareas y comunicar los resultados.

La documentación eficaz tiene tres beneficios:

1. Transparencia
2. Estandarización
3. Claridad

### **Transparencia**

La Transparencia es fundamental para demostrar el cumplimiento de las Regulaciones y los procesos internos, para satisfacer los requisitos de los seguros y para los procedimientos legales.

**La cadena de** custodia es el proceso de documentar la posesión y el control de pruebas durante el ciclo de vida de un incidente.

**Estándares** son referencias que informan sobre cómo establecer políticas.

### **Estandarización**

Un ejemplo de documentación que establece una estandarización es un **plan de respuesta a incidentes**, que es un documento que describe los procedimientos a seguir en cada paso de la respuesta a incidentes.

### **Claridad**

Una documentación clara ayuda a las personas a acceder rápidamente a la Información que necesitan para poder tomar las medidas necesarias.

## **Mejores prácticas (Para documentar este tipo de documentos)**

### **Conozca a su público**

Adapte su documento a las necesidades de su público.

### **Sea conciso**

Para asegurarse de que su documentación es útil, establezca el propósito inmediatamente. Esto ayuda a la gente a identificar rápidamente el objetivo del documento.

### **Actualización periódica**

En materia de Seguridad, se descubren y explotan nuevas vulnerabilidades constantemente. La Documentación debe revisarse y actualizarse con regularidad para mantenerse al día de la evolución del panorama de las amenazas.

# **El proceso de Triaje**

El **Triaje** es la priorización de los incidentes según su nivel de importancia o urgencia.

El proceso de triaje consta de tres pasos:

1. Recibir y evaluar
2. Asignar prioridad
3. Recopilar y analizar

### **Recibir y evaluar**

Un analista de seguridad recibe una alerta de un sistema de alerta como un **sistema de detección de intrusiones** (IDS).

Para evaluar una intrusión deberemos realizarnos las siguientes preguntas:

**¿Es la alerta un falso positivo?**

**¿Se activó esta alerta en el pasado?**

**¿Está la alerta activada por una vulnerabilidad conocida?**

**¿Cuál es la gravedad de la alerta?**

### **Asignar prioridad**

**Impacto funcional:** Los incidentes de seguridad dirigidos a los sistemas de tecnología de la información repercuten en el servicio que estos sistemas prestan a sus usuarios.

Por ejemplo, un incidente de ransomware puede afectar gravemente a la confidencialidad, disponibilidad e integridad de los sistemas. Los Datos pueden ser encriptados o borrados, haciéndolos completamente inaccesibles para los usuarios.

**Impacto en la Información:** Los incidentes pueden afectar a la confidencialidad, integridad y disponibilidad de los datos y la Información de una organización. En un ataque de robo de datos, los actores maliciosos pueden robar datos sensibles.

**Fiabilidad de la recuperación:** La forma en que una organización se recupera de un incidente depende del tamaño y alcance del mismo y de la cantidad de recursos disponibles. En algunos casos, la recuperación puede no ser posible, como cuando un actor malicioso roba con éxito datos de propiedad y los comparte públicamente.

### **Recopilar y analizar**

El último paso del proceso de triaje implica que el analista de Seguridad realice un análisis exhaustivo del Incidente. El análisis implica recopilar pruebas de distintas fuentes, realizar una investigación externa y documentar el proceso de investigación. El objetivo de este paso es reunir suficiente información para tomar una decisión informada para abordarlo.

Dependiendo de la gravedad del incidente, puede ser necesario pasar a un analista de nivel dos o a un gestor.

## **Beneficios del triaje**

**Gestionar los Recursos:** El triaje de alertas permite a los equipos de Seguridad centrar sus recursos en las amenazas que requieren atención urgente.

**Enfoque estandarizado:** El Triaje proporciona un enfoque estandarizado para la gestión de incidentes. La documentación del proceso, como los manuales de estrategias, ayuda a mover las alertas a través de un proceso iterativo para garantizar que las alertas se evalúan y validan correctamente. Esto garantiza que sólo las alertas válidas pasen a investigarse.

## **Planificación de continuidad del negocio**

Un **plan de continuidad del negocio** **(BCP** ) es un documento que describe los procedimientos para mantener las operaciones de la empresa durante y después de una interrupción importante. Un BCP ayuda a las organizaciones a garantizar que las funciones críticas del negocio puedan reanudarse o puedan restablecerse rápidamente cuando se produzca un Incidente.

### **Considere los impactos del ransomware en la continuidad del negocio**

### **Estrategias de recuperación**

Los planes de continuidad de las operaciones pueden incluir estrategias de recuperación centradas en la vuelta a la normalidad de las operaciones. La capacidad de recuperación del emplazamiento es un ejemplo de estrategia de recuperación.

### **Resiliencia del emplazamiento**

La resiliencia es la capacidad de prepararse, responder y recuperarse de las interrupciones. Las organizaciones pueden diseñar sus sistemas para que sean resilientes, de modo que puedan seguir prestando servicios a pesar de enfrentarse a interrupciones.

* **Sitios calientes**: Una instalación totalmente operativa que es un duplicado del entorno primario de una organización. Los sitios calientes pueden activarse inmediatamente cuando el sitio primario de una organización experimenta un fallo o una interrupción.
* Sitios templado:Una instalación que contiene una versión totalmente actualizada y configurada del sitio caliente. A diferencia de los sitios calientes, los sitios templados no están totalmente operativos y disponibles para su uso inmediato, pero pueden ponerse en funcionamiento rápidamente cuando se produce un fallo o una interrupción.
* Sitios Fríos:Una instalación de reserva equipada con parte de la infraestructura necesaria para hacer funcionar el sitio de una organización. Cuando se produce una interrupción o un fallo, es posible que los sitios fríos no estén listos para su uso inmediato y necesiten trabajos adicionales para estar operativos.

# **Revisión posterior al incidente**

Esto suele hacerse mediante una **reunión sobre lecciones aprendidas**, también conocida como post mortem. Una reunión sobre lecciones aprendidas incluye a todas las partes implicadas después de un incidente grave.

El propósito de esta reunión es evaluar el Incidente en su totalidad, valorar las acciones de respuesta e identificar cualquier área de mejora.

**Ejemplos de preguntas que se abordan en esta reunión:**

* ¿Qué ocurrió?
* ¿A qué hora ocurrió?
* ¿Quién lo descubrió?
* ¿Cómo se contuvo?
* ¿Qué medidas se tomaron para la recuperación?
* ¿Qué se podría haber hecho de otra manera?

### **Recomendaciones**

Algunos ejemplos de cambios que pueden implementarse son la actualización y mejora de las instrucciones del Manual de estrategias o la implementación de nuevas herramientas y tecnologías de seguridad.

# **Informe Final**

* **Resumen ejecutivo**: Un resumen de alto nivel del informe que incluya las conclusiones clave y los hechos esenciales relacionados con el incidente
* **Cronología**: Una cronología detallada del incidente que incluya marcas de tiempo que fechen la secuencia de acontecimientos que condujeron al incidente
* **Investigación**: Una recopilación de las acciones llevadas a cabo durante la Detección y el Análisis del Incidente. Por ejemplo, el análisis de un artefacto de red como una captura de paquetes revela información sobre las actividades que tienen lugar en una red.
* **Recomendaciones**: Una lista de acciones sugeridas para la prevención futura

# **Mejores prácticas para la recogida y gestión de registros**

## **Registros**

Un **registro** A es un registro de los eventos que se producen en los sistemas de una organización.

Los Equipos de Seguridad acceden a los registros desde receptores de registros como las herramientas SIEM que consolidan los registros para proporcionar un repositorio central de datos de registro. Los profesionales de la Seguridad utilizan los registros para realizar **análisis** de **registros**, que es el proceso de examinar los registros para identificar los eventos de Interés.

### **Tipos de registros**

* **Red**: Los registros de red son generados por dispositivos de red como firewalls, routers o switches.
* **Sistema**: Los registros del sistema son generados por sistemas operativos como ChromeOS™, Windows, Linux o macOS®.
* **Aplicación**: Los registros de aplicación son generados por aplicaciones de software y contienen información relativa a los eventos que se producen dentro de la aplicación, como una aplicación de smartphone.
* **Seguridad**: Los registros de Seguridad son generados por varios dispositivos o sistemas como software antivirus y sistemas de detección de intrusiones. Los registros de Seguridad contienen información relacionada con la seguridad, como el borrado de archivos.
* **Autenticación**: Los registros de autenticación se generan cada vez que se produce una autenticación, como un intento de inicio de sesión con éxito en una computadora.

Ejemplo de registro de autenticación:

*Login Event [05:45:15] User1 Authenticated successfully*

Modo “Verbose”

*Login Event [2022/11/16 05:45:15.892673] auth\_performer.cc:470 User1 Authenticated successfully from device1 (192.168.1.2)*

## **Gestión de registros**

La **Gestión de** registros es el proceso de recopilación, almacenamiento, análisis y eliminación de los datos de registro.

### **Qué registrar**

Es importante filtrar la información por el contenido que nos interesa.

Algunos registros que contienen datos PIII no pueden ser considerados debido a la ley de protección de datos.

### **El problema del overlogging**

La práctica de **recopilar y almacenar una cantidad excesiva o innecesaria de registros (logs)** de sistemas, aplicaciones y redes.

### **Retención de registros**

Por ejemplo, algunas Regulaciones requieren que las organizaciones retengan los registros durante periodos de tiempo determinados y las organizaciones pueden implementar prácticas de retención de registros en su política de gestión de registros.

Las organizaciones que operan en las siguientes industrias podrían necesitar modificar su política de gestión de registros para cumplir con los requisitos normativos:

* Industrias del sector público, como la Ley Federal de Modernización de la Seguridad de la Información (FISMA)
* Industrias sanitarias, como la Ley de Transferencia y Responsabilidad de los Seguros Médicos de 1996 (HIPAA)
* Industrias de servicios financieros, como el Estándar de seguridad de los datos para la industria de tarjetas de pago (PCI DSS), la Ley Gramm-Leach-Bliley (GLBA) y la Ley Sarbanes-Oxley de 2002 (SOX)

## **Formatos de Registro en el Análisis de Seguridad**

Como analistas de seguridad, utilizamos el **análisis de registros** para examinar los registros y detectar eventos de interés. Es fundamental saber leer e interpretar los diferentes **formatos de registro** para identificar actividades inusuales o maliciosas. Hemos revisado los siguientes formatos de registro: **JSON, Syslog, XML, CSV** y **CEF**.

### **JSON (JavaScript Object Notation)**

**JSON** es un formato de archivo ligero, fácil de leer y escribir, utilizado para almacenar y transmitir datos, especialmente en tecnologías web y entornos de nube. Su sintaxis se deriva de JavaScript y consta de:

* **Pares clave-valor:** Un conjunto de datos con una clave y su valor correspondiente, por ejemplo, "Alert": "Malware".
* **Comas:** Separan los datos.
* **Comillas dobles:** Encierran los datos de texto (cadenas). Los números no llevan comillas.
* **Corchetes rizados {}:** Encierran un **objeto**, que almacena datos en una lista de pares clave-valor separados por comas.
* **Corchetes []:** Encierran un **array**, que almacena datos en una lista ordenada separada por comas.

### **Syslog**

**Syslog** es un estándar para el registro y la transmisión de datos, que puede referirse a su protocolo, servicio o formato de registro.

* **Protocolo:** Utiliza el puerto 514 (texto plano) o 6514 (encriptado) para transportar registros a un servidor centralizado.
* **Servicio:** Actúa como un servicio de reenvío que consolida registros de múltiples fuentes.
* **Formato de registro:** Es uno de los formatos más utilizados, nativo de sistemas Unix®. Consiste en:
  + **Encabezado:** Contiene la **marca de tiempo**, el **nombre de host** (nombre de la máquina que envía el registro), el **nombre de la aplicación** y el **ID del mensaje**.
  + **Datos estructurados:** Información adicional en pares clave-valor dentro de corchetes [].
  + **Mensaje:** El mensaje detallado sobre el evento.
  + **Prioridad (PRI):** Indicado entre corchetes angulares <>, señala la urgencia del evento (cuanto menor el número, más urgente).

### **XML (eXtensible Markup Language)**

**XML** es un lenguaje y formato utilizado para almacenar y transmitir datos, nativo de sistemas Windows. Su sintaxis utiliza:

* **Etiquetas:** Se utilizan para almacenar e identificar datos, con una etiqueta de inicio (<tag>) y una de fin (</tag>).
* **Elementos:** Incluyen tanto los datos dentro de una etiqueta como la propia etiqueta. Todas las entradas XML deben contener al menos un **elemento raíz**, que puede contener **elementos hijos**.
* **Atributos:** Proporcionan información adicional sobre los elementos y se incluyen dentro de la etiqueta de inicio, siempre entre comillas simples o dobles, por ejemplo, <Data Name='SubjectUserSid'>.

### **CSV (Valores Separados por Comas)**

**CSV** utiliza comas para separar los valores de los datos. En los registros CSV, la posición de los datos corresponde a su nombre de campo, aunque los nombres de campo pueden no estar incluidos en el registro. Es crucial comprender qué campos incluye el dispositivo de origen (como un firewall o IPS).

### **CEF (Formato de Evento Común)**

**CEF** es un formato de registro que utiliza pares clave-valor para estructurar los datos. Su sintaxis se define por los siguientes campos separados por una barra vertical |: CEF:Version|Device Vendor|Device Product|Device Version|Signature ID|Name|Severity|Extension.

La parte de **Extensión** de la entrada de registro CEF debe escribirse en un formato clave-valor. Syslog se utiliza a menudo para transportar registros CEF, añadiendo una marca de tiempo y un nombre de host antes del mensaje CEF.

┌──────────────────────────────────────┐

│ Sistemas de Detección de │

│ Intrusiones (IDS) │

└──────────────────────────────────────┘

│

┌────────────────────────┴────────────────────────┐

│ │

┌─────────────────────┐ ┌───────────────────────────┐

│ Basado en el Host │ │ Basado en la Red │

│ (HIDS) │ │ (NIDS) │

└─────────────────────┘ └───────────────────────────┘

│ │

▼ ▼

┌─────────────────────┐ ┌───────────────────────────┐

│🔍 Monitorea un solo │ │🌐 Vigila el tráfico de │

│ dispositivo │ │ red en puntos clave │

│ │ │ │

│- Tráfico de entrada/salida │- Analiza comunicaciones │

│- Archivos modificados │ entre dispositivos │

│- Uso de CPU/memoria │- Detecta tráfico malicioso│

│- Actividad de usuarios │ │

└─────────────────────┘ └───────────────────────────┘

┌────────────────────────────────────────┐

│ 🔗 Combinación HIDS + NIDS │

│ → Defensa multicapa y más robusta │

└────────────────────────────────────────┘

▼

┌────────────────────────┴────────────────────────┐

│ Técnicas de Detección de IDS │

└─────────────────────────────────────────────────┘

│ │

▼ ▼

┌─────────────────────────┐ ┌─────────────────────────────┐

│ 🔐 Análisis por Firmas │ │ 🧠 Análisis por Anomalías │

└─────────────────────────┘ └─────────────────────────────┘

│ │

▼ ▼

- Usa "huellas" de amenazas - Aprende comportamiento normal

- Detecta ataques conocidos - Detecta comportamientos inusuales

- Pocas falsas alarmas - Detecta amenazas desconocidas

- Requiere actualización constante - Alta tasa de falsos positivos

- No detecta día cero - Puede aprender comportamiento malicioso

# **Panorama de Suricata**

**Suricata** es una herramienta **open-source** de seguridad que puede actuar como:

* **IDS (Sistema de Detección de Intrusiones)**
* **IPS (Sistema de Prevención de Intrusiones)**
* **NSM (Monitoreo de Seguridad de Red)**

#### **🔧 Modos de uso:**

1. **IDS**: Monitorea el tráfico o actividad del sistema para generar alertas.
2. **IPS**: Detecta y **bloquea** tráfico malicioso en tiempo real (requiere configuración especial).
3. **NSM**: Registra y analiza tráfico (en tiempo real o desde archivos .pcap) para análisis forense.

#### **📜 Reglas y Firmas en Suricata**

* **Regla = Firma** (son sinónimos).
* Se componen de:  
  + **Acción** (ej: alertar, rechazar).
  + **Cabecera** (IP origen/destino, puertos, protocolo).
  + **Opciones** (condiciones específicas).
* Las reglas se **procesan en orden** y el orden **afecta el resultado final**.

#### **✏️ Reglas Personalizadas**

* Son necesarias porque cada entorno es distinto.
* Reducen **falsos positivos** y mejoran la detección.
* Es fundamental aprender a **escribir reglas personalizadas eficaces**.

#### **⚙️ Configuración**

* El archivo principal es \*\*suricata.yaml\*\*.
* Define el comportamiento del sistema (qué monitorear, cómo actuar, etc.).

#### **🗂️ Archivos de registro**

* **eve.json**:  
  + Registro detallado, en formato JSON.
  + Ideal para SIEM y análisis avanzados.
* **fast.log**:  
  + Registro básico (IP y puerto).
  + Menos útil para análisis forense.

#### **🔑 Ideas clave**

* Suricata ofrece detección, prevención y análisis de red en una sola herramienta.
* La personalización (reglas y configuración) es esencial para su eficacia.
* Entender Suricata es una **habilidad clave** en la ciberseguridad profesional.

# **Fuentes de registro e ingestión de registros**

## **Visión general del proceso SIEM**

Anteriormente se ha tratado el proceso SIEM. A modo de repaso, el proceso consta de tres pasos:

1. **Recopilar y agregar datos**: Las herramientas SIEM recopilan datos de eventos de varias fuentes de datos.
2. **Normalizar datos**: Los datos de eventos recopilados se normalizan. La normalización convierte los datos a un formato estándar para que se estructuren de forma coherente y sean más fáciles de leer y buscar. Aunque la normalización de datos es una característica común en muchas herramientas SIEM, es importante tener en cuenta que las herramientas SIEM varían en sus capacidades de normalización de datos.
3. **Analizar los datos**: Una vez recopilados y normalizados los datos, las herramientas SIEM los analizan y correlacionan para identificar patrones comunes que indiquen actividad inusual.

Esta lectura se centra en el primer paso de este proceso, la recopilación y agregación de datos.

# **Métodos de búsqueda con herramientas SIEM**

### **🔍 Búsquedas en Splunk (SPL)**

* Utiliza su propio lenguaje: **SPL (Search Processing Language)**
* Ejemplo básico:  
   index=main fail → busca eventos en el índice *main* que contengan *fail*.
* Permite:  
  + Transformar resultados (por ejemplo, a gráficos).
  + Filtrar por criterios específicos.

#### **📌 Características:**

* **Tuberías (|)**: Conectan comandos y refinan resultados.  
  + Ejemplo:  
     index=main fail | chart count by host
* **Comodines (\*)**: Amplían coincidencias.  
  + Ejemplo: fail\* → coincide con "failed", "failure", etc.
* **Comillas dobles**: Para búsquedas exactas.  
  + Ejemplo: "login failure"

### **🔍 Búsquedas en Google SecOps (Chronicle)**

Dos tipos principales:

#### **1. Búsqueda UDM (Unified Data Model) – más rápida**

* Utiliza **datos normalizados e indexados**.
* Ejemplo:  
   metadata.event\_type = "USER\_LOGIN"
* Campos UDM comunes:  
  + **Entidades** (usuario, host, IP…)
  + **Metadatos del evento**
  + **Metadatos de red**
  + **Resultados de seguridad** (ej. detección de virus)

#### **2. Búsqueda en registros sin procesar**

* Busca en los logs **en bruto** (más lenta).
* Permite búsquedas con **expresiones regulares**.

### **🎯 Puntos clave**

* Cada SIEM tiene su **lenguaje y métodos de búsqueda**.
* Como analista, debes:  
  + Conocer varios entornos (Splunk, Chronicle…)
  + Optimizar búsquedas según necesidad
  + Aplicar filtros y expresiones regulares
* Habilidad fundamental para **detectar amenazas** y **responder a incidentes rápidamente**.

Course 7

**Resumen: Introducción a Python y su uso en ciberseguridad**

La programación consiste en crear instrucciones para que una computadora realice tareas específicas. Estas instrucciones, escritas en lenguajes de programación como Python, se traducen a código binario (0 y 1), que es el lenguaje que entiende la CPU.

**Python** es un lenguaje de propósito general, utilizado para crear sitios web, analizar datos o automatizar procesos. Funciona mediante un **intérprete**, que traduce el código línea por línea para que la computadora lo ejecute.

Existen diferentes **versiones de Python** (como Python 3, usado en este curso), y es importante conocerlas ya que pueden variar en su sintaxis (las reglas del lenguaje).

En **ciberseguridad**, Python es muy valioso para **automatizar tareas** repetitivas como:

* Análisis de registros
* Análisis de malware
* Gestión de accesos
* Detección de intrusiones
* Verificación de cumplimiento
* Escaneo de redes

**Puntos clave:** Python facilita la programación, debe ser interpretado para ejecutarse, tiene varias versiones con sintaxis distinta, y es muy útil en ciberseguridad para automatizar procesos.

# **Python Data Type**

**Ver videos tutoriales**

# **Asignar variables con Python**

**Ver videos tutoriales**

# **Condicionales Python**

**Ver videos tutoriales**

# **Más sobre bucles en Python**

### **Bucles for**

Un **bucle for** se utiliza para iterar sobre una **secuencia** (como una lista o cadena).

**Ejemplo básico:**

for i in ["elarson", "bmoreno", "tshah", "sgilmore"]:

print(i)

**Salida:**

elarson

bmoreno

tshah

sgilmore

* for indica el inicio del bucle.
* i es la **variable de bucle**.
* in señala la secuencia a recorrer.
* El cuerpo del bucle tiene **sangría** y contiene lo que se ejecuta en cada iteración (print(i)).

**Nota:** in también se usa en condicionales para verificar si un elemento pertenece a una secuencia:

if "elarson" in ["tshah", "bmoreno", "elarson"]:

print("True") # Esto se ejecuta

### **Bucles a través de listas**

computer\_assets = ["laptop1", "desktop20", "smartphone03"]

for asset in computer\_assets:

print(asset)

**Salida:**

laptop1

desktop20

smartphone03

### **Bucles a través de cadenas**

for char in "security":

print(char)

**Salida:**

s

e

c

u

r

i

t

y

### **Uso de range()**

range() genera una secuencia de números. Su sintaxis es: range(inicio, fin, incremento)

for i in range(0, 5, 1):

print(i)

**Salida:**

0

1

2

3

4

También puede omitirse el inicio y el incremento si son 0 y 1 respectivamente:

for i in range(5):

print(i)

**Salida:**

0

1

2

3

4

### **Bucles while**

Un **bucle while** se ejecuta mientras una condición sea **True**.

i = 1

while i < 5:

print(i)

i += 1

**Salida:**

1

2

3

4

* La condición i < 5 controla el bucle.
* i se inicializa fuera del bucle.
* El bucle se ejecuta hasta que la condición ya no es verdadera.

### **Condición con enteros**

login\_attempts = 0

while login\_attempts < 5:

print("Login attempts:", login\_attempts)

login\_attempts += 1

**Salida:**

Login attempts: 0

Login attempts: 1

Login attempts: 2

Login attempts: 3

Login attempts: 4

### **Condición con valores booleanos**

login\_status = True

count = 0

while login\_status:

print("Try again.")

if count == 4:

login\_status = False

count += 1

**Salida:**

Try again.

Try again.

Try again.

Try again.

El bucle termina cuando count llega a 4 y login\_status se vuelve False.

### **Gestión de bucles: break y continue**

#### **break: salir del bucle**

computer\_assets = ["laptop1", "desktop20", "smartphone03"]

for asset in computer\_assets:

if asset == "desktop20":

break

print(asset)

**Salida:**

laptop1

#### **continue: saltarse una iteración**

computer\_assets = ["laptop1", "desktop20", "smartphone03"]

for asset in computer\_assets:

if asset == "desktop20":

continue

print(asset)

**Salida:**

laptop1

smartphone03

### **Bucle infinito**

Un bucle que nunca termina:

while True:

print("Esto no termina nunca...")

* Detener con **CTRL-C** o **CTRL-Z**.
* Puede ser útil en servicios que deben ejecutarse constantemente, como servidores.

# **Funciones de Python en ciberseguridad**

### **Funciones en ciberseguridad**

Una **función** es una sección de código reutilizable. Son importantes en Python porque automatizan tareas repetitivas. En ciberseguridad, es común repetir procesos, como analizar múltiples registros en busca de inicios de sesión maliciosos.

Para evitar repetir código, puedes definir una función que reciba un registro como entrada y devuelva los intentos sospechosos. Esto permite aplicar la misma lógica a múltiples casos fácilmente.

### **Tipos de funciones en Python**

* **Funciones integradas**: vienen con Python, como print().
* **Funciones definidas por el usuario**: las crea el programador para necesidades específicas.

### **Cómo definir una función**

#### **Cabecera de función**

Se usa la palabra clave def, seguida del nombre de la función, paréntesis y dos puntos :.  
 Ejemplo:

def display\_investigation\_message():

* def: palabra clave para definir funciones.
* display\_investigation\_message: nombre descriptivo.
* (): paréntesis obligatorios.
* :: indica que comienza el cuerpo de la función.

**Consejo:** Usa nombres que indiquen claramente lo que hace la función.

#### **Cuerpo de la función**

Es el bloque de código con **sangría** que realiza la acción deseada.  
 Ejemplo:

def display\_investigation\_message():

print("investigate activity")

La sangría es esencial para distinguir la función del resto del código.

### **Llamar a una función**

Una vez definida, puedes usarla cuantas veces necesites escribiendo su nombre seguido de paréntesis.  
 Ejemplo:

display\_investigation\_message()

Esto ejecuta el código contenido en la función.

### **Ejemplo en contexto**

En un análisis de registros, puedes llamar a la función dentro de una estructura condicional:

if condition1:

display\_investigation\_message()

if condition2:

display\_investigation\_message()

Si solo condition1 se evalúa como True, el mensaje "investigate activity" se imprimirá una vez.

### **⚠️ Precaución: Bucles infinitos**

Llamar una función **dentro de sí misma sin condiciones de salida** puede crear un bucle infinito.

Ejemplo de bucle infinito:

def func1():

func1()

Este código hará que func1() se llame a sí misma sin fin, causando un error o bloqueo del programa.

Aquí tienes un **resumen con todos los ejemplos incluidos** del texto sobre funciones y variables en Python:

### **Funciones y variables en Python**

Esta lectura repasa conceptos clave sobre funciones: **parámetros, argumentos, sentencias return**, y **variables globales y locales**.

### **Parámetros y argumentos**

**Parámetros**: Son variables definidas en la cabecera de una función para usarse dentro de ella.  
  
 **Ejemplo**:  
  
 def remaining\_login\_attempts(maximum\_attempts, total\_attempts):

print(maximum\_attempts - total\_attempts)

* Aquí, maximum\_attempts y total\_attempts son parámetros.

**Argumentos**: Son los valores que se pasan a una función cuando se llama.  
  
 **Ejemplo**:  
  
 remaining\_login\_attempts(3, 2)

* Aquí, 3 y 2 son argumentos que se asignan a los parámetros maximum\_attempts y total\_attempts.

### **Sentencias return**

Se usan para devolver un valor desde una función. Se escriben con la palabra clave return (sin paréntesis).

**Ejemplo**:

def remaining\_login\_attempts(maximum\_attempts, total\_attempts):

return maximum\_attempts - total\_attempts

Las sentencias return permiten almacenar resultados en variables para usarlos después, por ejemplo en condiciones:  
  
 **Ejemplo**:  
  
remaining\_attempts = remaining\_login\_attempts(3, 3)

if remaining\_attempts <= 0:

print("Your account is locked")

Resultado:  
  
 Your account is locked

* Nota: Todo lo que esté después de return **no se ejecuta**.

### **Variables globales y locales**

**Variables globales**: Se definen fuera de cualquier función y están disponibles en todo el programa.  
  
 **Ejemplo**:  
  
 device\_id = "7ad2130bd"

**Variables locales**: Se definen dentro de una función (incluyendo los parámetros). No son accesibles desde fuera de ella.  
  
 **Ejemplo**:  
  
 def greet\_employee(name):

total\_string = "Welcome" + name

return total\_string

* Aquí, total\_string y name son variables locales. No pueden usarse fuera de la función.

### **Buenas prácticas**

Evita usar el **mismo nombre** para una variable global y una local, para evitar confusiones.  
  
 **Ejemplo de acceso a variable global desde una función**:  
  
 username = "elarson"

def identify\_user():

print(username)

identify\_user()

# Resultado: elarson

Si usas **el mismo nombre** dentro de una función, se crea una **nueva variable local** con ese nombre, sin afectar la global.  
  
 **Ejemplo**:  
  
 username = "elarson"

print(username) # 1: elarson

def identify\_user():

username = "bmoreno"

print(username) # 2: bmoreno

identify\_user()

print(username) # 3: elarson

* Como ves, la variable global no cambia.

### **Conclusión**

* Usa **parámetros** para pasar información a funciones, en lugar de depender de variables globales.
* Limita el uso de variables globales dentro de funciones para evitar errores y mantener el código limpio y predecible.

Aquí tienes un **resumen completo del contenido** sobre **funciones integradas en Python**, **manteniendo todos los ejemplos**:

### **Trabajar con funciones integradas en Python**

Las funciones integradas (o funciones *built-in*) ya existen en Python y se pueden usar directamente sin necesidad de definirlas. Algunas de las más comunes incluyen: print(), type(), max(), min() y sorted().

### **print()**

* Se usa para mostrar un objeto en la pantalla.
* Puede aceptar múltiples argumentos, separados por comas.

**Ejemplo:**

print("Investigate failed login attempts during September if more than", 100)

### **type()**

* Devuelve el tipo de dato del argumento que se le pasa.
* Acepta **solo un argumento**.

**Ejemplos:**

type("security") # Devuelve: <class 'str'>

type(7) # Devuelve: <class 'int'>

Para mostrar el resultado en pantalla, se usa junto con print():

print(type("security"))

# Salida: <class 'str'>

### **max() y min()**

* max() devuelve el valor más grande entre los argumentos.
* min() devuelve el valor más pequeño.

Ambas pueden recibir varios números como argumentos o una lista.

**Ejemplo de uso en ciberseguridad:**

Si tienes los tiempos de conexión de un usuario en minutos:

session\_times = [2, 12, 14, 19, 22, 32, 57]

print(min(session\_times)) # Salida: 2

print(max(session\_times)) # Salida: 57

### **sorted()**

* Ordena los elementos de un iterable (como listas o cadenas).
* Devuelve una **nueva lista ordenada**, sin modificar el original.
* Ordena números de menor a mayor y cadenas en orden alfabético.

**Ejemplo:**

session\_times = [12, 2, 32, 19, 57, 22, 14]

print(sorted(session\_times)) # Salida: [2, 12, 14, 19, 22, 32, 57]

print(session\_times) # Salida: [12, 2, 32, 19, 57, 22, 14]

Como se ve, la lista original no cambia.

**Nota:** sorted() no funciona con listas que contienen elementos de distintos tipos, como [1, 2, "hello"].

Aquí tienes un **resumen completo sin quitar los ejemplos** del contenido sobre *Importar módulos y bibliotecas en Python*:

### **📚 Módulos y bibliotecas en Python**

* **Módulo**: archivo Python con funciones, variables, clases y código ejecutable.
* **Biblioteca**: colección de módulos que ofrecen funcionalidades reutilizables.

### **📦 Biblioteca estándar de Python**

Viene incluida con Python e incluye muchos módulos útiles:

* **re**: búsqueda de patrones (por ejemplo, en archivos de registro).
* **csv**: manejo de archivos .csv.
* **glob y os**: interacción con la línea de comandos.
* **time y datetime**: manejo de fechas y tiempos.
* **statistics**: cálculo de estadísticas en datos numéricos.

#### **Ejemplos de funciones del módulo statistics:**

* mean(): calcula la media (promedio).
* median(): calcula la mediana (valor central).

### **🔽 Cómo importar módulos de la biblioteca estándar**

#### **✅ Importar un módulo completo**

Se usa la palabra clave import.  
 Ejemplo:

import statistics

monthly\_failed\_attempts = [10, 23, 12, 15, 16, 14, 20, 178, 30, 22, 17, 18]

mean\_failed\_attempts = statistics.mean(monthly\_failed\_attempts)

print("mean:", mean\_failed\_attempts)

# Salida: mean: 35.25

median\_failed\_attempts = statistics.median(monthly\_failed\_attempts)

print("median:", median\_failed\_attempts)

# Salida: median: 20.5

🔹 Nota: al importar el módulo completo, debes llamar las funciones con el prefijo del módulo, como statistics.mean().

#### **✅ Importar funciones específicas**

Se usa la palabra clave from.  
 Ejemplo:

from statistics import mean, median

monthly\_failed\_attempts = [10, 23, 12, 15, 16, 14, 20, 178, 30, 22, 17, 18]

print("mean:", mean(monthly\_failed\_attempts))

# Salida: mean: 35.25

print("median:", median(monthly\_failed\_attempts))

# Salida: median: 20.5

🔹 Ventaja: ya no necesitas escribir el nombre del módulo antes de la función.

### **🌐 Bibliotecas externas**

Además de los módulos estándar, puedes instalar y usar **bibliotecas externas**, como:

* **Beautiful Soup (bs4)**: para analizar HTML.
* **NumPy (numpy)**: para operaciones matemáticas y manejo de arrays.

#### **Para instalarlas en Jupyter o Google Colab:**

%pip install numpy

#### **Luego puedes importarlas como:**

import numpy

Aquí tienes un **resumen claro y completo del contenido sobre cómo garantizar una sintaxis y legibilidad adecuadas en Python**, **manteniendo los ejemplos**:

## **✅ Garantizar una sintaxis y legibilidad adecuadas en Python**

### **📝 Comentarios**

Los **comentarios** ayudan a explicar lo que hace el código y facilitan su comprensión.

#### **🔹 Comentarios de una sola línea**

* Se escriben con # al inicio.
* Deben mantenerse por debajo de 79 caracteres.

**Ejemplo:**

# Print elements of 'computer\_assets' list

computer\_assets = ["laptop1", "desktop20", "smartphone03"]

for asset in computer\_assets:

print(asset)

#### **🔹 Comentarios multilínea**

Se usan cuando se necesita más espacio para explicar funciones u operaciones complejas.

**Dos formas de hacerlo:**

1. Varias líneas con #:

# remaining\_login\_attempts() function takes two integer parameters,

# the maximum login attempts allowed and the total attempts made,

# and it returns an integer representing remaining login attempts

def remaining\_login\_attempts(maximum\_attempts, total\_attempts):

return maximum\_attempts - total\_attempts

1. Con **docstrings** usando comillas triples """ """:

"""

remaining\_login\_attempts() function takes two integer parameters,

the maximum login attempts allowed and the total attempts made,

and it returns an integer representing remaining login attempts

"""

### **⬅️ Indentación correcta**

La **indentación** (sangría) define la estructura del código en Python.

* Se recomienda usar **4 espacios** por nivel.
* Es obligatoria en **bucles**, **condicionales**, y **funciones**.

**Ejemplo:**

count = 0

login\_status = True

while login\_status == True:

print("Try again.")

count = count + 1

if count == 4:

login\_status = False

🔎 Nota: La condición if está indentada 8 espacios (4 del while + 4 adicionales).

### **🧱 Mantener una sintaxis correcta**

Los errores de sintaxis son comunes en Python. Algunos aspectos clave:

#### **🔹 Tipos de datos:**

* **Cadenas**: entre comillas → username = "bmoreno"
* **Números y booleanos**: sin comillas → login\_attempts = 5, login\_status = True
* **Listas**: entre corchetes y con comas → username\_list = ["bmoreno", "tshah"]

#### **🔹 Dos puntos en cabeceras:**

Toda función, bucle o condicional debe terminar con :  
 **Ejemplo:**

def remaining\_login\_attempts(maximum\_attempts, total\_attempts):

return maximum\_attempts - total\_attempts

Aquí tienes un **resumen con los ejemplos incluidos** del tema **“Cadenas y el analista de seguridad”**:

### **💻 Importancia de las cadenas en ciberseguridad**

En ciberseguridad, los **datos tipo cadena** son muy comunes. Representan elementos como:

* Direcciones IP
* Nombres de usuario
* URLs
* IDs de empleados

Estas cadenas **no se manipulan matemáticamente**, sino que se procesan para extraer, verificar o transformar información.

### **🔢 Trabajar con índices en cadenas**

Un **índice** es la posición numérica de cada carácter en una cadena. Los índices empiezan en 0 (de izquierda a derecha) y también pueden ser negativos (de derecha a izquierda).

Ejemplo con la cadena "h32rb17":

| **Carácter** | **Índice (+)** | **Índice (–)** |
| --- | --- | --- |
| h | 0 | -7 |
| 3 | 1 | -6 |
| 2 | 2 | -5 |
| r | 3 | -4 |
| b | 4 | -3 |
| 1 | 5 | -2 |
| 7 | 6 | -1 |

### **🧩 Notación entre corchetes para extraer caracteres**

Se usa para acceder a caracteres individuales o porciones de una cadena.

device\_id = "h32rb17"

print(device\_id[0]) # Resultado: h

También se pueden extraer **rebanadas (slices)**:

print(device\_id[0:3]) # Resultado: h32

Nota: el segundo índice **no se incluye** en el resultado.

### **⚙️ Funciones y métodos de cadenas**

#### **✅ str()**

Convierte un número u objeto en cadena.

string\_id = str(19329302)

#### **✅ len()**

Devuelve la cantidad de caracteres en la cadena.

if len("h32rb17") == 7:

print("The device ID has 7 characters.")

#### **✅ .upper()**

Convierte todos los caracteres a mayúsculas.

print("Information Technology".upper()) # Resultado: INFORMATION TECHNOLOGY

#### **✅ .lower()**

Convierte todos los caracteres a minúsculas.

print("Information Technology".lower()) # Resultado: information technology

### **🔍 Uso del método .index()**

Encuentra **la primera aparición** de un carácter o subcadena, devolviendo su índice.

print("h32rb17".index("r")) # Resultado: 3

Si no se encuentra, **lanza un error**:

print("h32rb17".index("a")) # ❌ Error: "a" no está en la cadena

Si hay más de una ocurrencia, solo se devuelve la **primera**:

print("r45rt46".index("r")) # Resultado: 0

#### **📌 Buscar subcadenas completas:**

cadena = "tsnow, tshah, tpatel"

print(cadena.index("tshah")) # Resultado: 7

⚠️ Si buscas una subcadena parcial como "ts", puede devolver una posición inesperada (por ejemplo, 0, correspondiente a "tsnow").

### **📋 Uso de listas en ciberseguridad**

Como analista de Seguridad, trabajarás frecuentemente con **listas en Python**, una estructura de datos que almacena múltiples elementos, incluso de distintos tipos (como nombres de usuario, direcciones IP o URLs).

Puedes recorrer listas con bucles for y aplicar condiciones para operar solo sobre ciertos elementos.

### **🔢 Índices en listas**

Cada elemento de una lista tiene un **índice**, que comienza en 0.

**Ejemplo:** Lista: ["elarson", "fgarcia", "tshah", "sgilmore"]  
 Índices:

* "elarson" → 0
* "fgarcia" → 1
* "tshah" → 2
* "sgilmore" → 3

### **🧱 Notación entre corchetes**

Se usa para **extraer** o **modificar** elementos:

Extraer un elemento por índice:  
  
 username\_list[2] # Resultado: 'tshah'

Extraer un **trozo** (sublista):  
  
 username\_list[0:2] # Resultado: ['elarson', 'fgarcia']

### **✏️ Modificar elementos**

Las **listas sí se pueden modificar**, a diferencia de las cadenas que son inmutables.

**Ejemplo:**

username\_list = ['elarson', 'fgarcia', 'tshah', 'sgilmore']

username\_list[1] = 'bmoreno'

# Resultado:

# Antes: ['elarson', 'fgarcia', 'tshah', 'sgilmore']

# Después: ['elarson', 'bmoreno', 'tshah', 'sgilmore']

### **⚙️ Métodos comunes de listas**

#### **1. .insert(indice, valor)**

Inserta un elemento en una posición específica.

**Ejemplo:**

username\_list.insert(2, 'wjaffrey')

# Resultado:

# Antes: ['elarson', 'bmoreno', 'tshah', 'sgilmore']

# Después: ['elarson', 'bmoreno', 'wjaffrey', 'tshah', 'sgilmore']

#### **2. .remove(valor)**

Elimina la **primera aparición** del elemento.

**Ejemplo:**

username\_list.remove('elarson')

# Resultado:

# Antes: ['elarson', 'bmoreno', 'wjaffrey', 'tshah', 'sgilmore']

# Después: ['bmoreno', 'wjaffrey', 'tshah', 'sgilmore']

#### **3. .append(valor)**

Agrega un elemento al **final** de la lista.

**Ejemplo simple:**

username\_list.append('btang')

# Resultado:

# Antes: ['bmoreno', 'wjaffrey', 'tshah', 'sgilmore']

# Después: ['bmoreno', 'wjaffrey', 'tshah', 'sgilmore', 'btang']

**Ejemplo con bucle:**

numbers\_list = []

for i in range(10):

numbers\_list.append(i)

# Resultado:

# Antes: []

# Después: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

#### **4. .index(valor)**

Devuelve el **índice** de la **primera aparición** del valor en la lista.

**Ejemplo:**

username\_list.index('tshah') # Resultado: 2

Este conjunto de técnicas permite **analizar, modificar y buscar información en colecciones de datos**, tareas comunes para un analista de ciberseguridad.

Aquí tienes un **resumen con los ejemplos incluidos** del contenido sobre expresiones regulares:

### 

### 

### 

### **Más información sobre expresiones regulares**

#### **✅ Conceptos básicos**

Una **expresión regular (regex)** es una secuencia de caracteres que forma un patrón. Se usa en Python para buscar coincidencias como direcciones IP, correos electrónicos o IDs de dispositivos.

Para usarlas, primero se importa el módulo re:

import re

Se almacenan como cadenas de texto y se usan con funciones como re.findall(), que devuelve una lista con todas las coincidencias del patrón. Requiere dos argumentos: el **patrón** y la **cadena** donde se busca.

**Ejemplo:**

re.findall("ts", "tsnow, tshah, bmoreno")

# Resultado: ['ts', 'ts']

### **🔣 Símbolos de expresión regular**

#### **Símbolos para tipos de caracteres**

* \w → cualquier carácter alfanumérico (incluye el guion bajo \_)

**Ejemplo:**

re.findall("\w", "h32rb17")

# Resultado: ['h', '3', '2', 'r', 'b', '1', '7']

Otros símbolos útiles:

* . → cualquier carácter
* \d → cualquier dígito [0-9]
* \s → espacio en blanco
* \. → el punto (literal)

**Ejemplo con \d:**

re.findall("\d", "h32rb17")

# Resultado: ['3', '2', '1', '7']

#### **Símbolos para cuantificar ocurrencias**

* + → una o más repeticiones

**Ejemplo:**

re.findall("\d+", "h32rb17")

# Resultado: ['32', '17']

* \* → cero o más repeticiones

**Ejemplo:**

re.findall("\d\*", "h32rb17")

# Resultado: ['', '32', '', '', '17', '']

* {n} → exactamente *n* repeticiones

**Ejemplo:**

re.findall("\d{2}", "h32rb17xy82pa29ss94")

# Resultado: ['32', '17', '82', '29', '94']

* {m,n} → entre *m* y *n* repeticiones

**Ejemplo:**

re.findall("\d{1,3}", "h32rb17xy825pa0ss299z4")

# Resultado: ['32', '17', '825', '0', '299', '4']

### **🧩 Construcción de un patrón**

Para encontrar información específica, se descompone en partes más pequeñas usando los símbolos vistos.  
 Por ejemplo, para extraer el **nombre de usuario y número de intentos** desde esta cadena:

employee\_logins\_string = "1001 bmoreno: 12 Marketing 1002 tshah: 7 Human Resources 1003 sgilmore: 5 Finance"

Se puede usar el patrón:

* \w+ → nombre de usuario
* : → carácter dos puntos
* \s → espacio
* \d+ → número de intentos

**Código:**

re.findall("\w+: \d+", employee\_logins\_string)

# Resultado: ['bmoreno: 12', 'tshah: 7', 'sgilmore: 5']

### **⚠️ Nota final:**

Es importante **probar tus expresiones regulares** para evitar errores, como incluir información innecesaria o excluir coincidencias relevantes.

**Resumen: Automatización de Tareas en CI/CD con Python**

**Objetivo principal:** Automatizar tareas de seguridad en canalizaciones de Integración Continua y Entrega/Despliegue Continuo (CI/CD) usando Python, incorporando prácticas de **DevSecOps** (Desarrollo + Seguridad + Operaciones).

### **¿Por qué automatizar la seguridad en CI/CD con Python?**

* ✅ **Rapidez y eficiencia:** Los scripts Python permiten realizar comprobaciones de seguridad de forma rápida y repetible.
* ✅ **Detección temprana:** Ayuda a identificar vulnerabilidades desde las primeras fases del desarrollo.
* ✅ **Consistencia:** Evita errores humanos, ya que ejecuta las mismas comprobaciones en cada versión.
* ✅ **Menos carga manual:** Libera al equipo de seguridad de tareas repetitivas.
* ✅ **Fomenta una cultura DevSecOps:** La seguridad se integra desde el principio como una responsabilidad compartida.

### **Tareas de seguridad que se pueden automatizar en CI/CD con Python**

1. **Pruebas de seguridad:**
   * 🔹 **SAST (Análisis Estático):** Detecta debilidades antes de compilar. Python puede lanzar herramientas, analizar resultados y generar informes.
   * 🔹 **DAST (Análisis Dinámico):** Pruebas sobre aplicaciones en ejecución. Python lanza herramientas, analiza y reporta.
   * 🔹 **SCA (Composición de Software):** Revisa dependencias y librerías de terceros en busca de vulnerabilidades.
2. **Exploración de vulnerabilidades:**
   * Python programa escaneos de imágenes, infraestructura y pipeline; recopila resultados y envía alertas.
3. **Comprobaciones de cumplimiento:**
   * Verifica que el código siga buenas prácticas y que la infraestructura cumpla políticas de seguridad.
4. **Gestión de secretos:**
   * Python detecta credenciales en código y trabaja con gestores de secretos (como Vault) para integrarlos de forma segura.
5. **Aplicación de políticas (Policy as Code):**
   * Define y aplica políticas que bloquean despliegues si se detectan problemas de seguridad.

### **Cómo se integra Python con herramientas de CI/CD**

* 🛠 **Jenkins, GitLab CI, CircleCI, etc.**: Permiten ejecutar scripts Python directamente en el pipeline.
* 🔌 **APIs:** Python puede interactuar con APIs de herramientas CI/CD y de seguridad para lanzar acciones y analizar resultados.
* 📦 **Plugins/extensiones:** Muchas integraciones ya vienen listas para usar con Python.

### **Otras tareas útiles en CI/CD automatizadas con Python**

* 🖥 **Configuración de entornos:** Automatiza la creación de entornos seguros para pruebas.
* 📏 **Control de calidad del código:** Usa linters y analizadores de estilo para detectar errores y problemas de seguridad.
* 🚀 **Publicación segura:** Automatiza despliegues a producción siguiendo prácticas seguras (transporte de archivos, configuración de red, etc.).

### **Conclusión:**

Python es una herramienta poderosa para automatizar la seguridad en CI/CD. Gracias a su flexibilidad, compatibilidad con APIs y bibliotecas especializadas, te permite:

* Automatizar revisiones de seguridad.
* Detectar errores antes de que lleguen a producción.
* Mejorar la velocidad de desarrollo sin sacrificar la seguridad.
* Construir una cultura DevSecOps sólida.

👉 El siguiente paso será aprender los fundamentos de Python: variables, condicionales, bucles, funciones y archivos, para crear tus propios scripts de automatización de seguridad.

## **🧰 Componentes esenciales de Python para automatizar tareas**

### **📌 ¿Qué es la automatización?**

Uso de tecnología para reducir el trabajo manual en tareas repetitivas.  
 👉 En **ciberseguridad**, se usa Python para automatizar investigaciones, auditorías y análisis de datos.

## **🧩 Componentes de Python clave para automatizar**

| **Componente** | **¿Qué es?** | **¿Para qué sirve en automatización?** |
| --- | --- | --- |
| 🧮 **Variables** | Contenedores de datos | Guardar y reutilizar información sin reescribirla |
| 🔀 **Condicionales (if, else)** | Evalúan condiciones | Deciden si ejecutar acciones o no |
| 🔁 **Bucles (for, while)** | Repetición de código | Automatizan tareas sobre listas o condiciones |
| 🧱 **Funciones** | Bloques reutilizables de código | Evitan duplicar código y aceptan parámetros |
| 🔡 **Cadenas** | Texto (strings) | Acceso a caracteres, búsqueda, comparación, limpieza |
| 📋 **Listas** | Colecciones ordenadas | Acceso, modificación y procesamiento masivo de datos |

## **💡 Ejemplo práctico: contar inicios de sesión sospechosos**

🔎 **Escenario:** Contar cuántas veces un usuario marcado se ha conectado.

**Componentes usados:**

* for: para iterar sobre la lista de intentos de inicio de sesión.
* if: para verificar si coincide con el usuario marcado.
* contador: variable que suma si hay coincidencia.
* función: permite reutilizar la lógica para cualquier usuario.

def contar\_logins(usuario\_marcado, lista\_usuarios):

contador = 0

for usuario in lista\_usuarios:

if usuario == usuario\_marcado:

contador += 1

return contador

## **📂 Trabajo con archivos (.txt y .csv)**

🔐 **Importancia:** Muchos datos de seguridad (como registros de accesos) están en archivos de texto.

| **Tipo de archivo** | **Características** |
| --- | --- |
| .txt | Texto sin formato, separadores variados (espacios, tabuladores) |
| .csv | Valores separados por comas, ideal para listas y tablas |

✅ Python permite:

* Leer y escribir datos.
* Extraer valores de registros.
* Convertir entre formatos.

## **📌 Conclusión:**

Estos **componentes básicos de Python** son las piezas clave para automatizar tareas en ciberseguridad.  
 Te permiten analizar archivos, procesar datos, aplicar lógica y reutilizar tu código de forma eficiente.

Aquí tienes un **resumen claro y estructurado** del contenido *"Importar archivos a Python"*, conservando los conceptos clave y ejemplos importantes:

## 

## 

## 

## **🗂️ Importar archivos a Python**

### **🔐 Importancia en ciberseguridad**

Los analistas de seguridad trabajan frecuentemente con archivos, como **archivos de registro (logs)**, que contienen eventos del sistema (p. ej., intentos de inicio de sesión o fallos de software). Analizarlos permite detectar actividades sospechosas.

## **📖 Abrir archivos en Python**

### **Sintaxis básica:**

with open("archivo.txt", "r") as file:

contenido = file.read()

* with: gestiona recursos y cierra el archivo automáticamente.
* open(): abre el archivo.  
  + Primer parámetro: nombre o ruta del archivo.
  + Segundo parámetro:  
    - "r" → leer
    - "w" → escribir (sobrescribe)
    - "a" → añadir (append)
* as file: crea una variable para usar dentro del bloque with.

#### **Ejemplo con ruta absoluta:**

with open("/home/analyst/logs/access\_log.txt", "r") as file:

## **📚 Leer archivos**

### **Método .read()**

Convierte el contenido del archivo en una **cadena**:

with open("update\_log.txt", "r") as file:

updates = file.read()

print(updates)

Una vez leído, puedes aplicar métodos como .index() o len() a la cadena.

## **✍️ Escribir archivos**

### **Cuando usar:**

* "w" para sobrescribir o crear un archivo.
* "a" para añadir contenido al final del archivo existente.

### **Método .write()**

Escribe una cadena en el archivo.

#### **Ejemplo:**

line = "jrafael,192.168.243.140,4:56:27,True"

with open("access\_log.txt", "a") as file:

file.write(line)

⚠️ **Importante:** si no usas with, asegúrate de cerrar el archivo manualmente con file.close().

Aquí tienes un **resumen completo** de la lectura **“Trabajar con archivos en Python”**, **manteniendo los ejemplos** explicativos:

## 

## **🔍 Análisis sintáctico de archivos en Python**

Trabajar con archivos no solo implica abrirlos y leerlos, sino también **dar formato** a su contenido para hacerlo más útil y legible. Esto se conoce como **análisis sintáctico** (parsing). Dos métodos muy útiles para ello son .split() y .join().

## **✂️ .split(): De cadena a lista**

El método .split() convierte una **cadena (string)** en una **lista**, separándola usando un carácter delimitador.

### **📌 Ejemplo 1: Separar por comas**

before .split(): elarson,bmoreno,tshah,sgilmore,eraab

after .split(): ['elarson', 'bmoreno', 'tshah', 'sgilmore', 'eraab']

Aquí, la cadena se divide en cada coma ,, convirtiéndose en una lista de nombres.

### **📌 Ejemplo 2: Separar por espacios (sin argumentos)**

before .split(): wjaffrey jsoto abernard jhill awilliam

after .split(): ['wjaffrey', 'jsoto', 'abernard', 'jhill', 'awilliam']

Sin argumento, .split() separa por **espacios en blanco**, incluyendo espacios, saltos de línea (\n), etc.

### **📁 Aplicación a archivos**

Para trabajar con el contenido de un archivo como lista, se puede combinar .read() y .split():

with open("update\_log.txt", "r") as file:

updates = file.read()

updates = updates.split()

Esto permite, por ejemplo, recorrer updates con un bucle for. El archivo se cierra automáticamente al salir del with.

## **🔗 .join(): De lista a cadena**

El método .join() hace lo contrario de .split(): une los elementos de una **lista** en una **cadena**.

### **📌 Ejemplo 3: Unir con comas**

before .join(): ['elarson', 'bmoreno', 'tshah', 'sgilmore', 'eraab']

after .join(): elarson,bmoreno,tshah,sgilmore,eraab

Aquí, ",".join(lista) concatena los elementos separados por comas.

### **📌 Ejemplo 4: Unir con saltos de línea**

"\n".join(['line1', 'line2', 'line3'])

Devuelve:

line1

line2

line3

Ideal para estructurar contenido con múltiples líneas.

### **📁 Aplicación a archivos**

Para guardar una lista en un archivo, primero se convierte en cadena:

updates = " ".join(updates) # Convertir lista a cadena separada por espacios

with open("update\_log.txt", "w") as file:

file.write(updates) # Sobrescribe el archivo con el nuevo contenido

"w" en open() indica modo escritura: borra lo anterior y escribe el nuevo contenido.

## **✅ Conclusión**

* Usa .split() para dividir cadenas en listas (útil para analizar texto de archivos).
* Usa .join() para unir listas en cadenas (útil para guardar listas en archivos).
* Estos métodos permiten transformar fácilmente datos entre texto plano y estructuras manipulables en Python.

# Depuración de errores en Python

Esta lectura profundiza en las técnicas de **depuración de errores en Python**, ampliando lo visto anteriormente con estrategias adicionales y ejemplos prácticos.

### **Tipos de errores**

#### **1. Errores de sintaxis**

Son errores en la estructura del código.

Ejemplo:

message = "You are debugging a syntax error

Este código genera:

SyntaxError: EOL while scanning string literal

La causa es que falta una comilla al final de la cadena. También puede aparecer un IndentationError si la sangría es incorrecta.

#### **2. Errores lógicos**

El código se ejecuta sin fallos, pero no produce el resultado esperado. Ejemplo:

login\_attempts = 5

if login\_attempts >= 5:

print("User has not reached maximum number of login attempts.")

Este mensaje es incorrecto, ya que el usuario sí ha alcanzado el límite. El error está en la condición >= en lugar de <.

#### **3. Excepciones**

Se producen cuando el código es sintácticamente correcto, pero no puede ejecutarse por otros motivos.

**NameError**: se refiere a una variable no definida.  
  
 print("Unusual logins:", unusual\_logins)

Error:  
  
 NameError: name 'unusual\_logins' is not defined

**IndexError**: se accede a un índice fuera del rango de una lista.  
  
 usernames = ["bmoreno", "tshah", "elarson"]

print(usernames[3]) # genera IndexError

* **TypeError**: se usa un tipo de dato inapropiado (por ejemplo, sumar un string a un número).
* **FileNotFound**: se intenta abrir un archivo que no existe.

### **Estrategias de depuración**

#### **Depuradores (Debuggers)**

Los entornos de desarrollo (IDEs) como Visual Studio Code ofrecen depuradores que permiten:

* Colocar **puntos de interrupción** para detener la ejecución en líneas específicas.
* Ver el valor de variables mientras el código se ejecuta paso a paso.

#### **Herramientas de IA**

Herramientas como **Gemini Code Assist** ayudan en IDEs a sugerir correcciones, explicar errores y generar código. Sin embargo, sus sugerencias deben ser validadas antes de usarlas, ya que no siempre son precisas o seguras.

### **Uso de instrucciones print() para depurar**

Imprimir mensajes en puntos estratégicos del código permite rastrear la ejecución y encontrar errores lógicos.

**Ejemplo:**

approved\_users = ["bmoreno", "tshah", "elarson"]

new\_users = ["sgilmore", "bmoreno"]

for user in new\_users:

print("line 5 - inside for loop")

if user in approved\_users:

print("line 7 - inside if statement")

print(user, "already in list")

print("line 9 - before .append method")

approved\_users.append(user)

print(approved\_users)

**Salida:**

line 5 - inside for loop

line 9 - before .append method

line 5 - inside for loop

line 7 - inside if statement

bmoreno already in list

line 9 - before .append method

['bmoreno', 'tshah', 'elarson', 'sgilmore', 'bmoreno']

Aunque el mensaje indica que "bmoreno" ya está en la lista, se añade de nuevo. Esto muestra un error lógico: la línea approved\_users.append(user) debería estar dentro de un bloque else.

Course 8

🔐 Importancia de la Clasificación de Datos y Activos

Proteger los datos y activos de una organización frente a amenazas, riesgos y vulnerabilidades es clave para la seguridad empresarial. Clasificar adecuadamente los datos permite reducir riesgos y prevenir incidentes.

📂 Tipos de Clasificación de Datos

1. **Datos públicos**
   * **Accesibles al público**.
   * Riesgo mínimo si se comparten, pero aún deben tener protección básica.
   * *Ejemplos:* comunicados de prensa, descripciones de puestos, material de marketing.
2. **Datos privados**
   * **No deben ser públicos**.
   * El acceso no autorizado puede generar riesgos serios.
   * *Ejemplos:* emails internos, ID de empleados, datos de investigación.
3. **Datos sensibles**
   * **Acceso solo a personal autorizado**.
   * Su filtración puede afectar gravemente las finanzas y reputación.
   * *Ejemplos:* cuentas bancarias, contraseñas, datos médicos, números de pasaporte.
4. **Datos confidenciales**
   * **Clave para operaciones comerciales**.
   * Acceso muy restringido, a veces bajo acuerdos de confidencialidad (NDA).
   * *Ejemplos:* secretos comerciales, registros financieros, datos administrativos sensibles.

🧩 Clasificación de Activos

* **Activos de bajo nivel:** Riesgo mínimo si se exponen.  
  + *Ejemplo:* dirección web de la empresa, datos públicos.
* **Activos de alto nivel:** Su exposición puede causar daños significativos.  
  + *Ejemplo:* correos internos con información confidencial, datos sensibles.

### **🔐 Importancia de la Continuidad del Negocio y Recuperación ante Desastres**

Aunque los profesionales de seguridad se esfuerzan en proteger los datos y recursos, los incidentes pueden ocurrir. Estos pueden causar pérdidas económicas y dañar la reputación de la empresa. Por eso, se deben crear planes para **mantener las operaciones** y **recuperarse tras un ataque o interrupción**.

### **🛡️ Etapas del Proceso de Seguridad**

1. **Identificar** los recursos críticos de la organización.
2. **Analizar amenazas** potenciales.
3. **Detectar** riesgos mediante herramientas y procesos.
4. **Crear planes** de continuidad del negocio y recuperación ante desastres.

### **📘 Plan de Continuidad del Negocio (BCP - Business Continuity Plan)**

Permite **mantener las operaciones** durante y después de una interrupción. Incluye:

1. **Análisis del Impacto en el Negocio**: Estudia cómo afectaría una interrupción.
2. **Documentar pasos de recuperación** de procesos y funciones críticas.
3. **Formar un equipo de continuidad del negocio**: Miembros de ciberseguridad, TI, RRHH, comunicaciones y operaciones.
4. **Entrenamiento del equipo**: Simular escenarios de riesgo para prepararse ante amenazas reales.

### **💾 Plan de Recuperación ante Desastres (DRP - Disaster Recovery Plan)**

Ayuda a **minimizar el impacto** de un incidente de seguridad y restaurar sistemas. Incluye:

* Estrategias para **recuperar software y hardware**.
* Identificación de **aplicaciones y datos afectados** por el incidente.
* Apoyo a la resolución de la amenaza (ej. recuperación tras un ataque de ransomware).

### 

### 

### 

### 

### 

### **🚨 ¿Qué es el Escalado de Incidentes?**

El **escalado de incidentes de seguridad** es el proceso de **identificar un posible incidente** y **derivarlo a un miembro del equipo más experimentado o al departamento adecuado**.  
 🔎 Ejemplo: múltiples intentos fallidos de inicio de sesión pueden ser una señal de actividad maliciosa y deben ser reportados.

### **📝 Importancia de la Escalada**

* Evita que problemas pequeños se conviertan en **grandes amenazas**.
* Asegura una **respuesta rápida y adecuada**.
* Protege a la organización y a sus **clientes** frente a daños financieros o de reputación.

### **📢 Notificación de Violaciones**

* Muchos países exigen **leyes de notificación de violaciones** si se ve comprometida información sensible (PII).
* Como analista, debes conocer y seguir las **leyes locales de privacidad**.
* La PII incluye: números de identificación, historiales médicos, direcciones, etc.

### **⚠️ Problemas de Seguridad de Bajo Nivel**

Son riesgos menores que **no exponen PII directamente**, pero **pueden escalar** si no se investigan:

* Ejemplo 1: intentos fallidos de inicio de sesión.
* Ejemplo 2: descarga de software no autorizado (posible malware).

📌 Aunque parezcan inofensivos, estos incidentes **pueden ocultar amenazas reales** y deben ser monitoreados o escalados si es necesario.

### **📋 Proceso de Escalado de Casos**

* Cada empresa tiene **protocolos propios**.
* Las políticas definen:  
  + A **quién notificar**.
  + Qué hacer si esa persona **no está disponible**.
  + Cómo escalar: vía herramienta de gestión, mesa de ayuda o comunicación directa.

### **🧭 Resumen: Reconocer las funciones y responsabilidades durante la Elevación de Incidentes**

Cuando ocurre un incidente de seguridad, su **elevación** (o escalado) implica informar al miembro del equipo adecuado según la naturaleza del problema. A continuación se describen los roles clave:

#### **👤 Propietario de los datos**

* Decide quién puede acceder, editar, usar o eliminar la información.
* Responsable de la clasificación y protección de los datos.
* **Ejemplo:** Acceso no autorizado a un software se escala al propietario de esos datos.

#### **⚙️ Responsable del tratamiento de datos**

* Define cómo y por qué se tratan los datos personales.
* Asegura el cumplimiento con normas de seguridad y privacidad.
* **Ejemplo:** Fuga de información de clientes → se eleva al responsable del tratamiento.

#### **🛠️ Encargado del tratamiento de datos**

* Procesa los datos en nombre del responsable (suele ser un proveedor externo).
* Implementa medidas de seguridad.
* **Ejemplo:** Fallo en la seguridad de un proveedor externo → se comunica al encargado del tratamiento.

#### **🔐 Custodio de datos**

* Administra el acceso a sistemas y datos.
* Implementa controles de seguridad, supervisa amenazas y uso de los datos.
* **Ejemplo:** Vulnerabilidad en controles de seguridad → se informa al custodio de datos.

#### **🛡️ Delegado de protección de datos (DPO)**

* Supervisa el cumplimiento de políticas internas de protección de datos.
* Asesora y realiza auditorías de seguridad.
* **Ejemplo:** Incumplimiento de normativas de protección → se notifica al DPO.

### 

### 

### 

### **🛡️ Momento de la Elevación – Resumen**

**Importancia del Rol del Analista de Nivel Básico**

* Aunque los incidentes pequeños parezcan insignificantes, si no se escalan correctamente, pueden tener un gran impacto.
* Los analistas de seguridad de nivel básico juegan un papel crucial en detectar y escalar eventos antes de que se conviertan en incidentes graves.

**Toma de Decisiones en Seguridad**

* La ciberseguridad es un entorno dinámico, donde hay que tomar decisiones con rapidez y responsabilidad.
* El analista debe ayudar al equipo a identificar qué eventos requieren ser elevados a miembros superiores del equipo.

**Confianza y Aprendizaje Constante**

* Es vital confiar en el propio juicio, especialmente al tomar decisiones sobre escalado.
* Estudiar y comprender la **política de escalado** de la organización es esencial.
* Hacer preguntas demuestra compromiso con el aprendizaje y la mejora.

**Diferencias entre Eventos de Seguridad**

* No todos los incidentes tienen el mismo nivel de gravedad.
* Es clave identificar qué **datos y recursos** son más críticos para la organización.
* Esto permite priorizar incidentes correctamente según su impacto en las operaciones del negocio.

**Ejemplos de Tipos de Incidentes**

* **Malware**: Infección por software malicioso.
* **Acceso no autorizado**: Alguien obtiene acceso sin permiso.
* **Uso indebido**: Violación de las políticas internas por parte de empleados.

**Priorización Ejemplar**

* Un acceso no autorizado a una aplicación crítica de fabricación debe escalarse antes que un malware en un sistema no esencial.

**Consejos Clave para Elevar Incidentes**

1. Conozca bien la política de escalado de su empresa.
2. Siga dicha política rigurosamente.
3. Pregunte siempre que tenga dudas.

### **🧩 ¿Qué son las partes interesadas?**

Son individuos o grupos con interés en las decisiones o actividades de una organización. En seguridad, se refiere a las personas que deben ser informadas sobre incidentes y riesgos.

### **🏢 Niveles de partes interesadas en seguridad**

1. **Gestor de riesgos de ciberseguridad** Encargado de identificar, evaluar y mitigar riesgos de seguridad.  
    → *Comunicación frecuente con analistas principiantes.*
2. **Director de operaciones** Supervisa las operaciones diarias relacionadas con la seguridad.  
    → *Colabora activamente con los analistas para responder a incidentes.*
3. **CISO (Chief Information Security Officer)** Máxima autoridad en temas de seguridad de la información.  
    → *Contacto poco frecuente con analistas de nivel básico.*
4. **CEO (Director General) y CFO (Director Financiero)** Altos directivos que se enfocan en el impacto global o financiero de los incidentes.  
    → *Rara vez se comunican directamente con analistas.*

### **📢 Comunicación con partes interesadas**

* Como analista principiante, informarás sobre:  
  + Incidentes observados (por ejemplo, intentos fallidos de inicio de sesión).
  + Posibles causas y riesgos asociados.
* Las partes interesadas (como el director de operaciones) se encargan de coordinar la respuesta y elevar el caso si es necesario.

### **🔁 Flujo de información**

1. El analista informa al director de operaciones o gestor de riesgos.
2. Ellos, a su vez, escalan la información al CISO o CFO.
3. Así, aunque no hables directamente con los niveles más altos, tu trabajo influye en las decisiones estratégicas.

### **🧠 Objetivo de la comunicación**

La comunicación con las partes interesadas debe ser:

* Clara
* Directa
* Adaptada a su nivel de conocimiento y rol

### **🎯 Ir al grano**

Antes de comunicarte, pregúntate:

1. **¿Qué** necesita saber la persona?
2. **¿Por qué** es importante que lo sepa?
3. **¿Cuándo** debe actuar?
4. **¿Cómo** puedes explicarlo en términos **no técnicos**?

### **📋 Seguir los protocolos**

* Infórmate de los **canales de comunicación aceptados** por la organización.
* Puede incluir: correo electrónico, videollamadas, reuniones presenciales, apps de mensajería interna, etc.
* Usar el **medio adecuado** ayuda a transmitir el mensaje de forma eficaz y profesional.

### **🏢 Adaptar la comunicación al perfil del destinatario**

* Si hablas con **directores de operaciones** o **gestores de riesgos**, céntrate en aspectos operativos diarios (ej. anomalías en registros).
* Para **altos directivos** (CISO, CFO, CEO), enfoca en riesgos globales o impacto financiero.

### **📡 Métodos de comunicación recomendados**

Elige el canal adecuado según el contexto:

| **Tipo de mensaje** | **Canal sugerido** |
| --- | --- |
| Breve y urgente | Mensaje instantáneo o llamada |
| Complejo o detallado | Correo electrónico o reunión |
| Datos y análisis | Hoja de cálculo o presentación |

### **Resumen: Cuadros de Mando Visuales para Comunicaciones Efectivas en Ciberseguridad**

Como ya sabemos, en el campo de la ciberseguridad, **comunicar eficazmente** con las partes interesadas es clave, especialmente porque suelen estar ocupadas. Para esto, los **cuadros de mando visuales** son herramientas muy potentes.

Un **cuadro de mando visual** es una forma de presentar rápidamente distintos tipos de datos en un solo lugar, usando **cuadros, gráficos e infografías**. Su objetivo es transformar datos complejos en **información clara y accionable**, ayudando a los responsables a identificar riesgos y tomar decisiones informadas sobre la postura de seguridad de la organización.

Podemos utilizarlos cuando necesitemos contar una "historia" con datos, como los resultados de una auditoría de *phishing* por departamento. En lugar de un largo correo electrónico, un gráfico lo hará mucho más fácil de entender. Podemos crear estos cuadros de mando con herramientas como **Google Sheets** o **Apache OpenOffice**.

En resumen, cuando la información es densa o numérica, los elementos visuales son la mejor opción para asegurar que nuestros mensajes de ciberseguridad sean impactantes y bien recibidos.

# **Estrategias para comprometerse con la comunidad de la ciberseguridad**

# Información:

Este elemento incluye contenido que aún no se tradujo a tu idioma preferido.

Ha aprendido mucho sobre el Campo de la Seguridad, desde los orígenes de la Seguridad y su importancia para las organizaciones de todo el mundo hasta el reconocimiento de los Incidentes de Seguridad y la Comunicación con las partes Interesadas.

La Seguridad es una industria que evoluciona rápidamente, por lo que es importante mantenerse al día de las últimas noticias y tendencias. Esta lectura se centrará en cómo mantenerse comprometido con la comunidad de la Ciberseguridad después de completar este Programa.

## **Organizaciones y conferencias de Seguridad**

Asistir a conferencias sobre seguridad y unirse a organizaciones le brinda la oportunidad de adquirir conocimientos de profesionales experimentados que buscan constantemente nuevas formas de mejorar sus estrategias y técnicas de seguridad.

**Encuentre la organización adecuada**

¿A qué organización de Seguridad debería unirse? Esta pregunta depende de su interés específico por la Seguridad. ¿Es usted alguien que quiere centrarse en reaccionar ante los incidentes de Seguridad o en evitar que se produzcan? ¿Le interesa la Seguridad forense o el registro de datos? ¿Tiene aspiraciones de ser un CISO algún día? Es importante tener una idea clara de cuáles son sus intereses antes de acotar su búsqueda de una organización o conferencia de ciberseguridad.

**Comenzar la búsqueda**

Una vez que comprenda cuáles son sus intereses, realice una búsqueda en Internet de organizaciones o conferencias en su zona. Por ejemplo, puede teclear "conferencias sobre ciberseguridad de Respuesta ante incidentes en mi zona" Esta búsqueda le dará una Lista de conferencias de ciberseguridad centradas en la Respuesta ante incidentes. Si le interesa la Seguridad forense, puede teclear "organizaciones de Seguridad forense en mi zona" o una frase similar en su buscador web. Sean cuales sean sus intereses, puede hacer una búsqueda en Internet para encontrar una organización de ciberseguridad o una conferencia centrada en ese ámbito.

### **Utilice las redes sociales**

Las redes sociales son otra forma estupenda de encontrar organizaciones o conferencias de ciberseguridad. [LinkedIn®](https://www.linkedin.com/)

, por ejemplo, es una plataforma de redes sociales que conecta a los profesionales de los negocios entre sí. Puede utilizar LinkedIn® para encontrar grupos de seguridad u organizaciones a las que unirse. En la barra de búsqueda de LinkedIn®, puede probar con consultas de búsqueda como:

* "Grupos de ciberseguridad de respuesta ante incidentes".
* "Organizaciones para analistas de ciberseguridad"

**sea consciente de la ingeniería social**

Aunque las Redes sociales son una buena forma de conectar con otros profesionales del sector de la seguridad, también es importante tener en cuenta que los hackers utilizan las redes sociales para engañar a los usuarios para que les faciliten información privada. Ya ha aprendido que la ingeniería social es una técnica de manipulación que explota el error humano para obtener información privada, acceso u objetos de valor. Para protegerse de la ingeniería social cuando utilice las redes sociales para encontrar Recursos, recuerde siempre no hacer clic en enlaces o archivos adjuntos inesperados enviados por usuarios desconocidos en las redes sociales.

### **Listas de correo para la Seguridad**

Otra forma estupenda de mantenerse en contacto con el sector de la seguridad es inscribirse en diferentes listas de correo sobre ciberseguridad. Estas listas de correo envían información periódicamente sobre diversos temas de Seguridad. La Agencia de Ciberseguridad y Seguridad de las Infraestructuras (CISA) ofrece dos listas de correo sobre ciberseguridad a las que puede unirse:

* Una lista centrada en información sobre amenazas a la seguridad, mejores prácticas para la ciberseguridad y análisis de los socios nacionales e internacionales de CISA en materia de seguridad
* Una Lista que proporciona resúmenes semanales de las nuevas vulnerabilidades que podrían suponer un riesgo para la red de una organización

### **Integración de la IA en Ciberseguridad**

En el ámbito de la ciberseguridad, en constante evolución, la **Inteligencia Artificial (IA)** se ha vuelto indispensable para afrontar las amenazas modernas. Esta lección tiene como objetivo familiarizar a los profesionales de la ciberseguridad con las herramientas y aplicaciones de la IA para mejorar sus operaciones diarias y fortalecer las defensas contra los ciberataques.

**¿Qué aprenderemos?**

* **Fundamentos de la IA:** Se cubrirán los conceptos esenciales de la inteligencia artificial, entendida como programas capaces de realizar tareas cognitivas humanas.
* **Herramientas de IA en Ciberseguridad:** Se explorarán las diversas herramientas de IA empleadas en el sector, que asisten en la automatización de tareas rutinarias, resumen de información y análisis de datos.
* **Casos de Uso Prácticos:** Se presentarán ejemplos concretos de cómo los profesionales de ciberseguridad de Google utilizan la IA en su trabajo diario para mejorar la productividad y la eficacia.
* **Práctica con IA:** Tendrán la oportunidad de practicar el uso de la IA para optimizar sus propias tareas de ciberseguridad.

**La doble cara de la IA:**

Aunque la IA es una herramienta potente para la ciberseguridad, también presenta vulnerabilidades significativas. Los ciberdelincuentes pueden explotar la IA para lanzar ataques más complejos y evadir la detección. Por lo tanto, es crucial que los profesionales de la ciberseguridad entiendan cómo proteger los sistemas de IA y anticiparse a su uso malicioso.

Al finalizar esta lección, estarán preparados para aprovechar el poder de la IA para salvaguardar sus organizaciones y contribuir a un entorno digital más seguro.

En su trabajo como profesional de la ciberseguridad, puede aprovechar las herramientas de IA generativa para realizar tareas tanto prácticas como creativas. Considere estas aplicaciones de las herramientas de IA generativa que pueden ayudarle a trabajar de forma más eficiente y eficaz:

* **Crear contenidos**. Puede utilizar herramientas de IA generativa para generar texto, imágenes y otros medios. Por ejemplo, puede crear un gran conjunto de datos falsos para probar las herramientas de ciberseguridad que utiliza su organización.
* **Analizar información rápidamente**. Las herramientas de IA generativa pueden analizar grandes cantidades de contenido con rapidez. Por ejemplo, puede utilizar IA generativa para resumir informes o transcripciones de reuniones que contengan información importante relacionada con la seguridad de su organización, lo que le ayudará a identificar más rápidamente los detalles clave.
* **Responder a preguntas de forma detallada y matizada**. La IA generativa es eficaz a la hora de resumir información, lo que la hace útil para la investigación. Por ejemplo, puede pedir a una herramienta de IA generativa que le proporcione información sobre los tipos más comunes de amenazas a la ciberseguridad, como el malware y el ransomware.
* **Simplificar el trabajo diario**. También puede utilizar la IA generativa para aumentar las tareas rutinarias. Por ejemplo, las herramientas de IA pueden proporcionar rápidamente un análisis inicial sobre la probabilidad de que un correo electrónico sea malicioso.

## 

## **🤖 La IA en Ciberseguridad: Una Herramienta Imprescindible para Profesionales 🛡️**

La Inteligencia Artificial (IA) está transformando rápidamente el campo de la ciberseguridad, y entender cómo usarla es crucial para su éxito profesional.

### **El Rol Creciente de la IA**

* **Esencial para el éxito:** Las herramientas de IA son cada vez más comunes en ciberseguridad. Comprenderlas y usarlas mejora su capacidad para **identificar riesgos**, **automatizar respuestas** y **priorizar amenazas**.
* **Protección de sistemas de IA:** Como profesional, también podría estar involucrado en la seguridad de los propios sistemas de IA de su organización.

### **IA Generativa: Trabaje de Forma Más Inteligente y Rápida**

La **IA generativa** es un tipo de IA capaz de crear contenido nuevo. Nos permite trabajar de manera más eficiente:

* **Creación de Contenido:** Generar listas de mejores prácticas o guías.
* **Análisis y Resumen:** Resumir grandes informes de seguridad o datos complejos.
* **Respuesta a Preguntas:** Obtener respuestas rápidas sobre amenazas comunes.
* **Simplificación de Tareas:** Detectar, por ejemplo, signos de phishing en correos electrónicos.

Para obtener los mejores resultados de una IA generativa, es útil usar un marco como **TCREI** (Tarea, Contexto, Referencias, Evaluar e Iterar) para guiar sus indicaciones.

### **Uso Responsable de la IA Generativa**

Aunque la IA es potente, tiene limitaciones. Es vital usarla de forma responsable:

* **Revisar los Resultados:** Siempre verifique la precisión y utilidad de la información generada por la IA.
* **Divulgar el Uso:** Sea transparente si ha usado IA para crear contenido.
* **Privacidad y Seguridad:** Evite introducir información sensible en herramientas de IA generativa.
* **Enfoque Humano:** La IA debe ser un **complemento** de nuestras habilidades humanas, no un reemplazo.

### **IA en Acción: Aplicaciones Reales**

La IA generativa, como Gemini, puede ayudar a los profesionales de ciberseguridad a:

* **Entender marcos de seguridad complejos.**
* **Escanear código** en busca de errores y vulnerabilidades.
* **Sugerir mejoras en el código** (Python, etc.).
* **Describir vulnerabilidades** y cómo mitigarlas.
* **Apoyar en la detección y respuesta** (ej., investigación de alertas).

Experimentar con estas herramientas le permitirá dedicar menos tiempo a tareas repetitivas y más a proteger personas, organizaciones y datos, que es su valor principal como profesional de la ciberseguridad.

# **Prepararse para las entrevistas técnicas**

Anteriormente ha aprendido sobre el proceso de entrevista, que generalmente consiste en una entrevista introductoria, una segunda entrevista con el responsable de contratación y una entrevista de grupo. En esta lectura, se centrará más en las entrevistas técnicas que tienen lugar durante el proceso de entrevista. Aunque las entrevistas técnicas pueden variar de una empresa a otra, los conceptos tratados en esta lectura le ayudarán a prepararse para cualquier entrevista técnica.

## **¿Qué son las entrevistas técnicas?**

Después de solicitar un empleo en ciberseguridad y recibir el interés de un reclutador, es posible que se le invite a una entrevista introductoria, a una segunda entrevista con un responsable de contratación y a una entrevista de grupo. Además, es posible que le pidan que participe en una entrevista técnica. La principal diferencia entre las entrevistas técnicas y otras entrevistas es que se centran en los conocimientos necesarios de herramientas específicas.

## **Python**

Python es un lenguaje de programación que constituye una herramienta importante en el ámbito de la seguridad, y es posible que le pregunten por él durante una entrevista técnica. Será importante que menciones tus conocimientos básicos de Python. Es posible que recuerdes que Python es popular por su facilidad de uso, así como por sus amplias bibliotecas e integraciones. Puede aplicarse a diversas tareas de seguridad que requieren automatización. Durante la entrevista, es posible que te pidan que escribas un pseudocódigo en Python. Ser capaz de utilizar con confianza la terminología de Python durante una entrevista puede ayudarte a destacar como candidato potencial. Esto permitirá que el entrevistador sepa que tienes una sólida comprensión de lo que es Python y para qué se puede utilizar.

## **Técnicas generales**

Durante su entrevista técnica, es posible que se espere que demuestre conocimientos básicos de varios conceptos generales de seguridad. Por ejemplo, puede que tenga que demostrar que está familiarizado con **los marcos de seguridad**, que son directrices utilizadas para elaborar planes que ayuden a mitigar los riesgos y las amenazas para los datos y la privacidad. Al hablar de marcos de seguridad, sería útil mencionar su conocimiento de marcos específicos del NIST, como el Marco de Ciberseguridad (CSF). Otro concepto técnico del que puede hablar durante una entrevista técnica es la Seguridad de red. Puede que recuerdes que **la seguridad de red** es la práctica de mantener la infraestructura de red de una organización a salvo de accesos no autorizados. Repasar los diferentes conceptos técnicos que ha aprendido a lo largo de este programa es una buena manera de prepararse para una entrevista técnica. Afinará sus habilidades y le ayudará a dejar una buena impresión en el entrevistador.

Además, puede ser una buena idea escribir toda la pregunta en un papel antes de responder. A menudo, las preguntas de las entrevistas técnicas constan de varias partes. A veces, la gente se apresura a dar una respuesta y demostrar sus conocimientos, pero no cubre completamente todo lo que pide la pregunta. Escribir la pregunta puede ayudarle a asegurarse de que la ha entendido bien y de que es capaz de dar una respuesta estructurada.

## **Posibles preguntas de la entrevista técnica**

Cada entrevista técnica será diferente, dependiendo de la empresa y de los entrevistadores. No obstante, a continuación te ofrecemos algunas posibles preguntas técnicas que te ayudarán a prepararte:

* ¿Qué es el Modelo TCP/IP?

El Modelo TCP/IP es un marco utilizado para visualizar cómo se organizan y transmiten los datos a través de una red.

* ¿Qué es el modelo OSI?

El modelo OSI son las siglas de Modelo de interconexión de sistemas abiertos (OSI). Es un concepto estandarizado que describe las siete capas que utilizan los ordenadores para comunicarse y enviar datos a través de la red.

* ¿Qué son las herramientas SIEM y para qué sirven?

Las herramientas SIEM son herramientas de administración de información y eventos de seguridad que utilizan los profesionales de la seguridad para identificar y analizar amenazas, riesgos y vulnerabilidades de seguridad.

## **Puntos clave**

Las entrevistas técnicas son una gran oportunidad para mostrar a tu potencial empleador tus habilidades técnicas en ciberseguridad, como la codificación, durante la entrevista. Incluso si no tienes experiencia previa en seguridad, puedes aprovechar este momento para compartir lo que has aprendido en el programa del Certificado de Ciberseguridad de Google y expresar tu entusiasmo por poner en práctica lo aprendido. Si te preparas para el proceso de la entrevista técnica, estarás en el buen camino para labrarte una carrera en el ámbito de la ciberseguridad.