Sri Lanka Institute of Information Technology

IT21299902 (ZAKEY M.S.M.A)

Bug bounty Report 09.

Domain: curl.com

Web security – IE2062

B.Sc. (Hons) in Information Technology Specialization in cyber security.

Declaration:

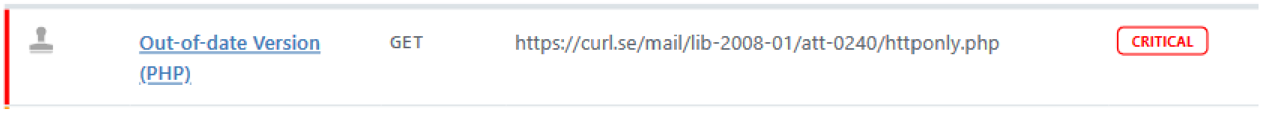
- I hereby certify that no part of this assignment has been copied from any other work or any other source. No part of this assignment has been written/produced for me by another person.
- I hold a copy of this assignment that I can produce if the original is lost or damaged.

1. Using components with known vulnerabilities.

• Out of date version of PHP.

I have identified many vulnerabilities in under domain **www.curl.com/**. I figured out it has vulnerabilities that listed by OWSAP Top 10. And the overall website risk level is **critical**. I used net sparker to identify vulnerabilities in the following domain.

I hope this message finds you well. I am writing to report a vulnerability I discovered under the domain of **https://curl.se/mail/lib-2008-01/att-0240/httponly.php** during my participation in the bug bounty program. Please find below the details of the vulnerability for your review and further action.



	Out-of-date Version (PHP)	GET	https://curl.se/mail/lib-2008-01/att-0240/httponly.php	CRITICAL

Vulnerability title: Out of-date version of PHP.

Severity: critical

Affected version: 7.4.0 to 7.4.20.

OWASP classification 2013: A9

CVSS 3.0 score: 9

CVSS 3.1 score: 9

CVSS string: -

Effected components: Performance issues.

Security.

Stability.

Date of Discovery: 2023/05/10

Date of Report: 2023/05/13

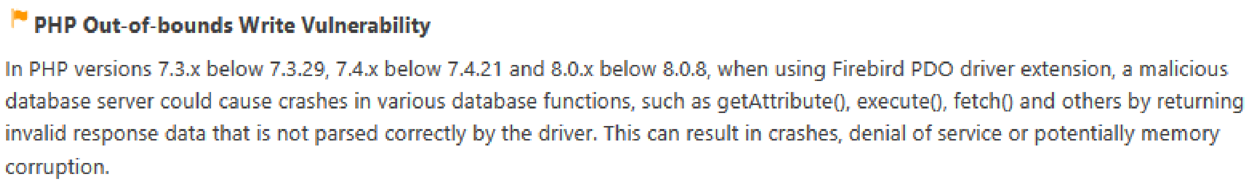
2. Vulnerability Description.

Under this OWSAP category I have identified many vulnerabilities like using components with known vulnerabilities and sensitive data exposure that can occur due to bad implementation security under domain of **https://curl.se/mail/lib-2008-01/att-0240/httponly.php**

Also detected that this web application is using an outdated, also vulnerable PHP version to full fill their back-end processes. Figured out that this version of PHP is vulnerable to following attacks. Also, many features are depreciated in this version of PHP.

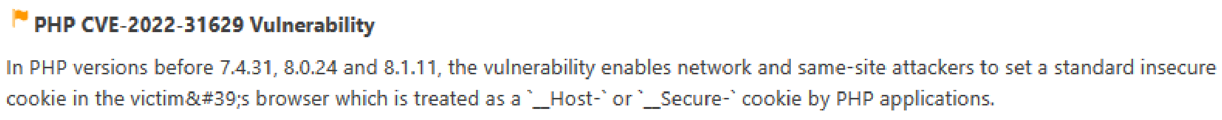
3. Impact assessment.

Known weaknesses in outdated PHP versions may exist and be used by attackers. Unauthorized access, remote code execution, SQL injection, cross-site scripting (XSS), and other sorts of attacks could be made possible by these vulnerabilities.



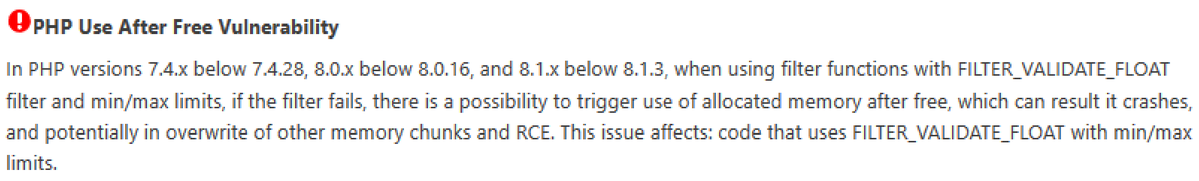
PHP Out-of-bounds Write Vulnerability

In PHP versions 7.3.x below 7.3.29, 7.4.x below 7.4.21 and 8.0.x below 8.0.8, when using Firebird PDO driver extension, a malicious database server could cause crashes in various database functions, such as getAttribute(), execute(), fetch() and others by returning invalid response data that is not parsed correctly by the driver. This can result in crashes, denial of service or potentially memory corruption.



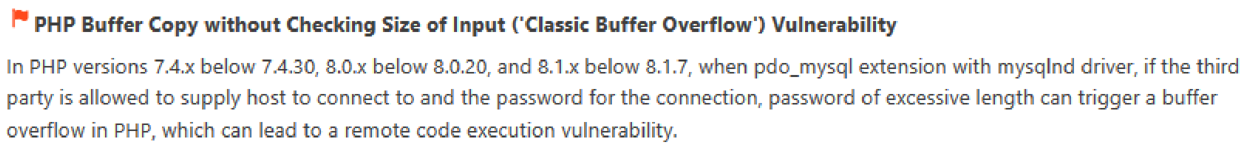
PHP CVE-2022-31629 Vulnerability

In PHP versions before 7.4.31, 8.0.24 and 8.1.11, the vulnerability enables network and same-site attackers to set a standard insecure cookie in the victim&#39;s browser which is treated as a `__Host-` or `__Secure-` cookie by PHP applications.



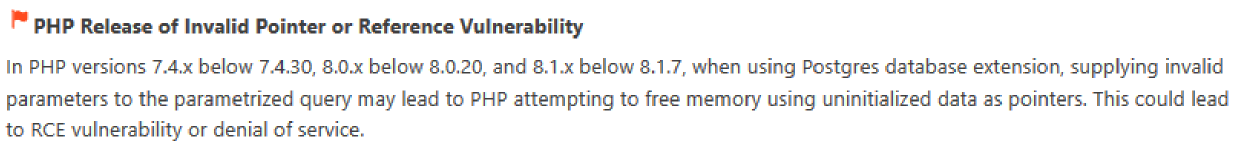
PHP Use After Free Vulnerability

In PHP versions 7.4.x below 7.4.28, 8.0.x below 8.0.16, and 8.1.x below 8.1.3, when using filter functions with FILTER_VALIDATE_FLOAT filter and min/max limits, if the filter fails, there is a possibility to trigger use of allocated memory after free, which can result it crashes, and potentially in overwrite of other memory chunks and RCE. This issue affects: code that uses FILTER_VALIDATE_FLOAT with min/max limits.



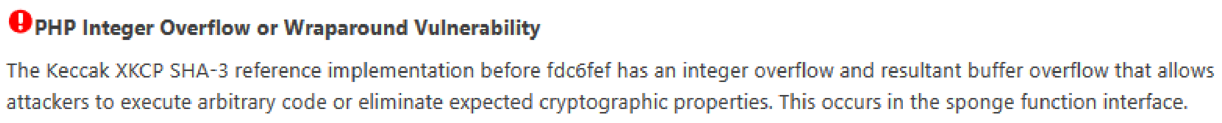
PHP Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') Vulnerability

In PHP versions 7.4.x below 7.4.30, 8.0.x below 8.0.20, and 8.1.x below 8.1.7, when pdo_mysql extension with mysqlnd driver, if the third party is allowed to supply host to connect to and the password for the connection, password of excessive length can trigger a buffer overflow in PHP, which can lead to a remote code execution vulnerability.



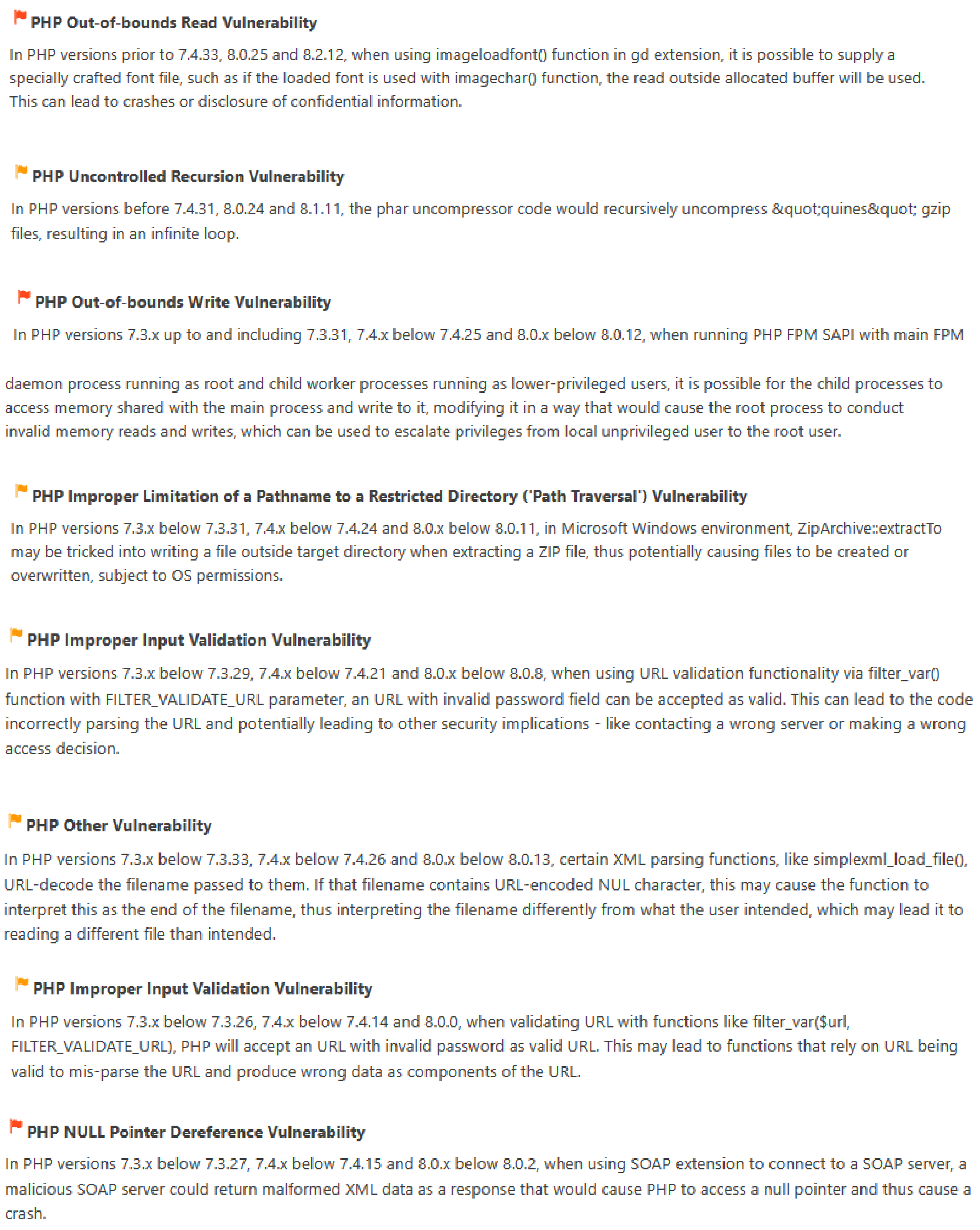
PHP Release of Invalid Pointer or Reference Vulnerability

In PHP versions 7.4.x below 7.4.30, 8.0.x below 8.0.20, and 8.1.x below 8.1.7, when using Postgres database extension, supplying invalid parameters to the parametrized query may lead to PHP attempting to free memory using uninitialized data as pointers. This could lead to RCE vulnerability or denial of service.



PHP Integer Overflow or Wraparound Vulnerability

The Keccak XKCP SHA-3 reference implementation before fdc6fef has an integer overflow and resultant buffer overflow that allows attackers to execute arbitrary code or eliminate expected cryptographic properties. This occurs in the sponge function interface.



PHP Out-of-bounds Read Vulnerability

In PHP versions prior to 7.4.33, 8.0.25 and 8.2.12, when using imageloadfont() function in gd extension, it is possible to supply a specially crafted font file, such as if the loaded font is used with imagechar() function, the read outside allocated buffer will be used. This can lead to crashes or disclosure of confidential information.

PHP Uncontrolled Recursion Vulnerability

In PHP versions before 7.4.31, 8.0.24 and 8.1.11, the phar uncompressor code would recursively uncompress &quot;quines&quot; gzip files, resulting in an infinite loop.

PHP Out-of-bounds Write Vulnerability

In PHP versions 7.3.x up to and including 7.3.31, 7.4.x below 7.4.25 and 8.0.x below 8.0.12, when running PHP FPM SAPI with main FPM

daemon process running as root and child worker processes running as lower-privileged users, it is possible for the child processes to access memory shared with the main process and write to it, modifying it in a way that would cause the root process to conduct invalid memory reads and writes, which can be used to escalate privileges from local unprivileged user to the root user.

PHP Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') Vulnerability

In PHP versions 7.3.x below 7.3.31, 7.4.x below 7.4.24 and 8.0.x below 8.0.11, in Microsoft Windows environment, ZipArchive::extractTo may be tricked into writing a file outside target directory when extracting a ZIP file, thus potentially causing files to be created or overwritten, subject to OS permissions.

PHP Improper Input Validation Vulnerability

In PHP versions 7.3.x below 7.3.29, 7.4.x below 7.4.21 and 8.0.x below 8.0.8, when using URL validation functionality via filter_var() function with FILTER_VALIDATE_URL parameter, an URL with invalid password field can be accepted as valid. This can lead to the code incorrectly parsing the URL and potentially leading to other security implications - like contacting a wrong server or making a wrong access decision.

PHP Other Vulnerability

In PHP versions 7.3.x below 7.3.33, 7.4.x below 7.4.26 and 8.0.x below 8.0.13, certain XML parsing functions, like simplexml_load_file(), URL-decode the filename passed to them. If that filename contains URL-encoded NUL character, this may cause the function to interpret this as the end of the filename, thus interpreting the filename differently from what the user intended, which may lead it to reading a different file than intended.

PHP Improper Input Validation Vulnerability

In PHP versions 7.3.x below 7.3.26, 7.4.x below 7.4.14 and 8.0.0, when validating URL with functions like filter_var($url, FILTER_VALIDATE_URL), PHP will accept an URL with invalid password as valid URL. This may lead to functions that rely on URL being valid to mis-parse the URL and produce wrong data as components of the URL.

PHP NULL Pointer Dereference Vulnerability

In PHP versions 7.3.x below 7.3.27, 7.4.x below 7.4.15 and 8.0.x below 8.0.2, when using SOAP extension to connect to a SOAP server, a malicious SOAP server could return malformed XML data as a response that would cause PHP to access a null pointer and thus cause a crash.

1. The PHP development team no longer provides security updates and patches for out-of-date PHP versions. This implies that any newly found flaws or problems won't be fixed, making your application more vulnerable to assaults.

2. Certain features or functions are designated as deprecated in PHP versions and then finally removed in newer versions. Utilizing an out-of-date PHP version puts your codebase at risk of future support and maintenance issues because it depends on deprecated features.

Because of this vulnerability marked as Critical It’s a must to address it with a good solution.

4. Steps to Reproduce.

1. Determine the specific outdated PHP version you want to test. Note the exact version number.

2. Make a test script that includes the code patterns that are relevant to the found vulnerabilities. To achieve this, code fragments that cause integer overflow, wraparound, or use-after-free behavior may need to be created.

3. To create inputs that result in integer overflow or wraparound scenarios, use fuzzing frameworks that you have created or tools like the AFL (American Fuzzy Lop). To examine the code and identify flaws, these tools produce test inputs automatically.

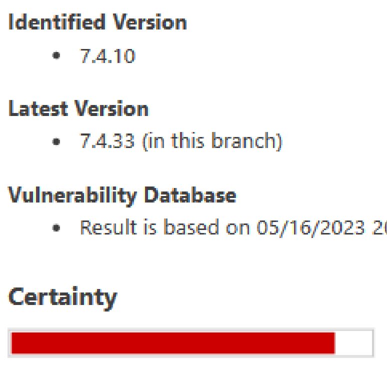
4. To find use-after-free vulnerabilities, use programs like AddressSanitizer (ASan) or Valgrind (Memcheck). These tools can be used to detect erroneous memory accesses brought on using released objects.

5. To exploit the vulnerabilities found, run the test script or use the fuzzing tool's generated inputs.

6. Watch how the PHP interpreter behaves while the test is running. Keep an eye out for errors, crashes, or other strange output that can point to a vulnerability.

7. Examine how the vulnerabilities affect the PHP interpreter and the system. Examining crash logs, memory dumps, or error messages may be necessary to gauge the seriousness and consequences of the vulnerabilities.

5. Proof of the Existence of vulnerability.

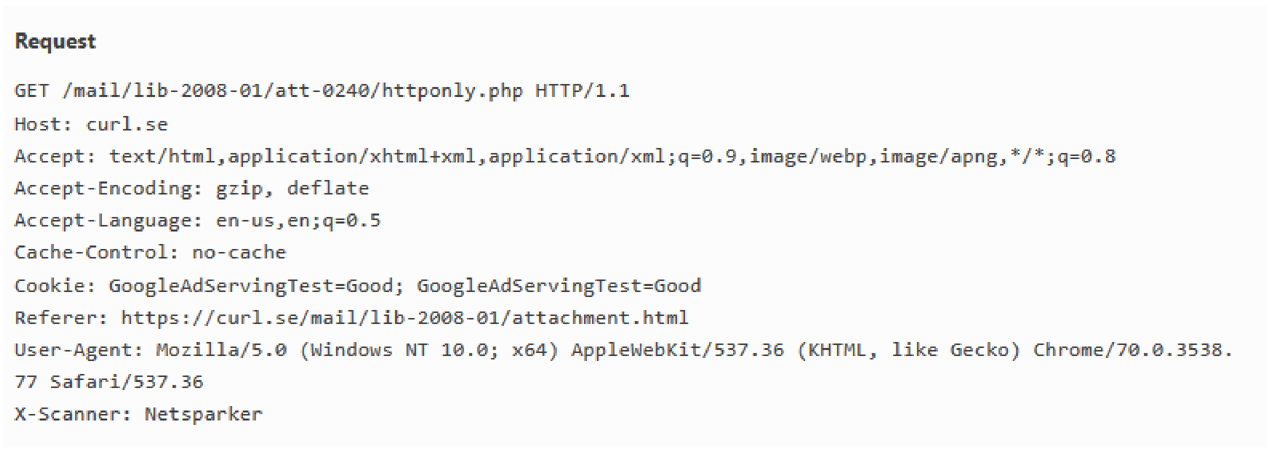


Identified Version
- 7.4.10

Latest Version
- 7.4.33 (in this branch)

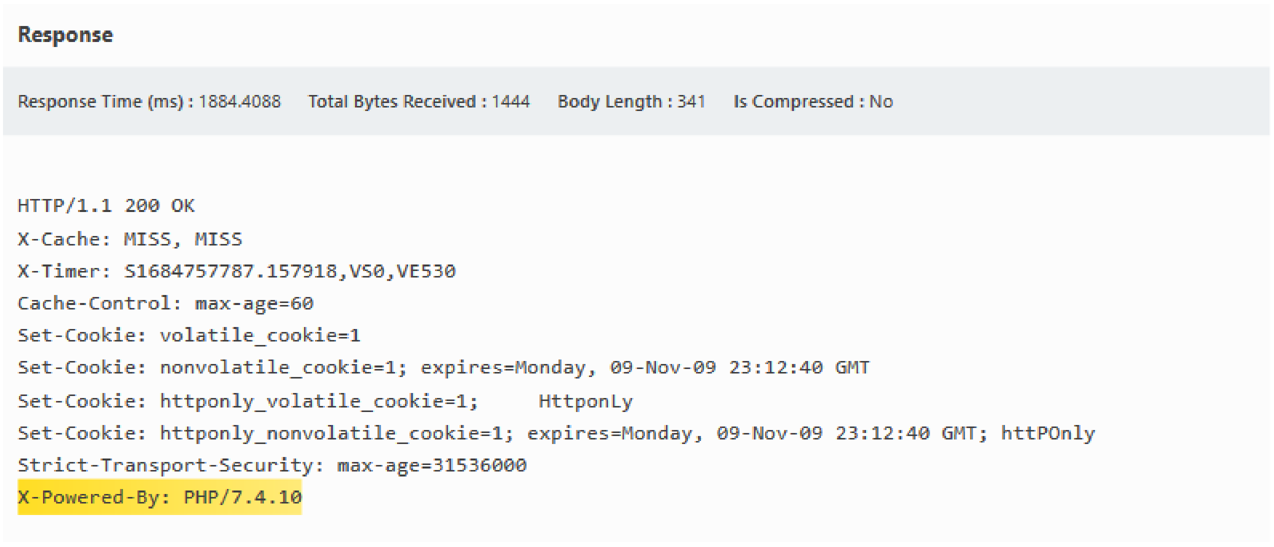
Vulnerability Database
- Result is based on 05/16/2023 20:30:00 vulnerability database content.

Certainty



Request

```
GET /mail/lib-2008-01/att-0240/httponly.php HTTP/1.1
Host: curl.se
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
Cache-Control: no-cache
Cookie: GoogleAdServingTest=Good; GoogleAdServingTest=Good
Referer: https://curl.se/mail/lib-2008-01/attachment.html
User-Agent: Mozilla/5.0 (Windows NT 10.0; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.
77 Safari/537.36
X-Scanner: Netsparker
```



Response

Response Time (ms) : 1884.4088   Total Bytes Received : 1444   Body Length : 341   Is Compressed : No

```
HTTP/1.1 200 OK
X-Cache: MISS, MISS
X-Timer: S1684757787.157918,VS0,VE530
Cache-Control: max-age=60
Set-Cookie: volatile_cookie=1
Set-Cookie: nonvolatile_cookie=1; expires=Monday, 09-Nov-09 23:12:40 GMT
Set-Cookie: httponly_volatile_cookie=1;     HttponLy
Set-Cookie: httponly_nonvolatile_cookie=1; expires=Monday, 09-Nov-09 23:12:40 GMT; httPOnly
Strict-Transport-Security: max-age=31536000
X-Powered-By: PHP/7.4.10
```

6. Proposed mitigation/fix.

1. Regular PHP upgrades to the most recent stable release or a supported version that receives security updates and fixes are essential to reducing these risks. This makes sure that your web application takes advantage of the most recent PHP development team features, performance upgrades, and security patches.

2. Applying patches created by the PHP community to address the specific vulnerabilities you've found can be an option if updating to the most recent version of PHP is not possible right away. The known security flaws in the out-of-date version may be fixed by these patches.

3. You can further defend your application against potential vulnerabilities by adhering to best practices in web application security, such as input validation, safe coding techniques, and frequent security audits.

4. To enforce security settings, review and alter the php.ini configuration file. Disable, for instance, elements of your program that are known to be weak points or superfluous. To assist in locating potential security concerns, enable proper error reporting.

5. To add an additional layer of defense against typical web vulnerabilities, use a web application firewall. WAFs can aid in the detection and mitigation of attacks utilizing known exploits that target PHP vulnerabilities.