Vulnerability Assessment Summary Report

- This vulnerability assessment report was conducted on the web application http://testphp.vulnweb.com/.
- The assessment was conducted using Acunetix WVS.
- The objective of this vulnerability assessment was to identify and assess the security risks associated with the web application http://testphp.vulnweb.com/. . The assessment was conducted in accordance with the Open Web Application Security Project (OWASP) Testing Guide.

1.Out-of-date Version (PHP)

- Acunetix 360 identified you are using an out-of-date version of PHP.
- **Impact**
- Since this is an old version of the software, it may be vulnerable to attacks.
- **PHP Other Vulnerability**
- Double free vulnerability in the format printer in PHP 7.x before 7.0.1 allows remote attackers to have an unspecified impact by triggering an error.
- **Affected Versions**
- 5.3.0 to 7.0.0
- **External References**
- CVE-2015-8880
- **Exploits**
- **PHP Improper Restriction of Operations within the Bounds of a Memory Buffer Vulnerability**
- An issue was discovered in the EXIF component in PHP before 7.1.27, 7.2.x before 7.2.16, and 7.3.x before 7.3.3. There is an uninitialized read in exif_process_IFD_in_TIFF.
- **Affected Versions**
- 5.3.0 to 7.0.0
- **External References**
- CVE-2019-9641

2.Password Transmitted over HTTP

Acunetix 360 detected that password data is being transmitted over HTTP.
Impact
If an attacker can intercept network traffic, he/she can steal users' credentials.

Vulnerabilities
5.1. http://testphp.vulnweb.com/login.php

- **Actions to Take**
- See the remedy for solution.
- Move all of your critical forms and pages to HTTPS and do not serve them over HTTP.

- **Remedy**
- All sensitive data should be transferred over HTTPS rather than HTTP. Forms should be served over HTTPS. All aspects of the application that accept user input, starting from the login process, should only be served over HTTPS.

3.Local File Inclusion

Acunetix 360 identified a Local File Inclusion vulnerability, which occurs when a file from the target system is injected into the attacked server page.
Acunetix 360 **confirmed** this issue by reading some files from the target web server.

Impact
The impact can vary, based on the exploitation and the read permission of the web server user. Depending on these factors, an attacker might carry out one or more of the following attacks:
•Gather usernames via an `"/etc/passwd"` file
•Harvest useful information from the log files, such as `"/apache/logs/error.log"` or `"/apache/logs/access.log"`
•Remotely execute commands by combining this vulnerability with some other attack vectors, such as file upload vulnerability or log injection

- **Remedy**
- If possible, do not permit appending file paths directly. Make them hard-coded or selectable from a limited hard-coded path list via an index variable.
- If you definitely need dynamic path concatenation, ensure you only accept required characters such as "a-Z0-9" and do not allow ".." or "/" or "%00" (null byte) or any other similar unexpected characters.
- It is important to limit the API to allow inclusion only from a directory and directories below it. This way you can ensure any potential attack cannot perform a directory traversal attack.

4.[Probable] SQL Injection

- Acunetix 360 identified a Probable SQL Injection, which occurs when data input by a user is interpreted as an SQL command rather than as normal data by the backend database.
- This is an extremely common vulnerability and its successful exploitation can have critical implications.
- Even though Acunetix 360 believes there is a SQL injection in here, it **could not confirm** it. There can be numerous reasons for Acunetix 360 not being able to confirm this. We strongly recommend investigating the issue manually to ensure it is an SQL injection and that it needs to be addressed. You can also consider sending the details of this issue to us so we can address this issue for the next time and give you a more precise result.
- **Impact**
- Depending on the backend database, database connection settings and the operating system, an attacker can mount one or more of the following type of attacks successfully:Reading, updating and deleting arbitrary data/tables from the database.
- Executing commands on the underlying operating system.

5.Boolean Based SQL Injection

- Acunetix 360 identified a Boolean-Based SQL Injection, which occurs when data input by a user is interpreted as a SQL command rather than as normal data by the backend database.
- This is an extremely common vulnerability and its successful exploitation can have critical implications.
- Acunetix 360 **confirmed** the vulnerability by executing a test SQL query on the backend database. In these tests, SQL injection was not obvious, but the different responses from the page based on the injection test allowed Acunetix 360 to identify and confirm the SQL injection.
- **Impact**
- Depending on the backend database, the database connection settings and the operating system, an attacker can mount one or more of the following type of attacks successfully:Reading, updating and deleting arbitrary data/tables from the database
- Executing commands on the underlying operating system

6. Local File Inclusion

Acunetix 360 identified a Local File Inclusion vulnerability, which occurs when a file from the target system is injected into the attacked server page.
Acunetix 360 **confirmed** this issue by reading some files from the target web server.

Impact
The impact can vary, based on the exploitation and the read permission of the web server user. Depending on these factors, an attacker might carry out one or more of the following attacks:
•Gather usernames via an `"/etc/passwd"` file
•Harvest useful information from the log files, such as `"/apache/logs/error.log"` or `"/apache/logs/access.log"`
•Remotely execute commands by combining this vulnerability with some other attack vectors, such as file upload vulnerability or log injection