



UTOPIIC Web Application Security Assessment Report

Date: 25/11/2022

Version: 1.0

Submitted by: SrivelEnterprise

Classification: CONFIDENTIAL

ATTENTION: This proposal contains information from SrivelEnterprise that is confidential and privileged. The information is intended for the private use of Utopiic Technologies. By accepting this proposal, you agree to keep the contents in confidence and not copy, disclose, or distribute this without written request to and written confirmation from Utopiic. If you are not the intended recipient, be aware that any disclosure, copying, or distribution of the contents of this document is prohibited.



DOCUMENT CONTROL

ITEM	DESCRIPTION
Document Title:	UTOPIIC Web Application Security Assessment Report
Reference:	NA
Version No.:	V1.0
Status:	Final
Assessment Date	23-11-2022 - 24-11-2022
Publish Date:	25/11/2022
Revision Date:	NA
Author	Sridhar
Analyst	Sridhar, Kotteeshwaran
Reviewed and Approved By	SrivelEnterprise Quality Team

REVISION RECORD

VERSION	MODIFIED BY	SIGNATURE / DATE	NOTES	Review & Approve
V1.0	Srinivasan M. S	25/11/2022	Final Release	Srinivasan M. S

CONTACT

NAME	E-Mail	Contact no
Rajesh Bommaan	rajeshbommaan@srivel.biz	+91-9844599605



Table of Contents

1	Executive Summary	4
1.1	Project Scope	4
1.2	Project Objectives	4
1.3	Target Systems.....	4
1.4	Assumption	5
1.5	Summary of Evaluation.....	5
1.6	Finding Rating Levels.....	5
2	VULNERABILITY ISSUES MAPPING	6
2.1	High-Level Recommendation:.....	6
2.2	Immediate attention:.....	6
2.3	Later Attention:.....	6
3	Detailed Web Application Penetration Testing Findings.....	8
3.1	REMOTE CODE EXECUTION.....	8
3.2	BROKEN AUTHENTICATION	10
3.3	STORED XSS	11
3.4	HTML INJECTION	13
3.5	IMPROPER INPUT VALIDATION.....	15
3.6	UNRESTRICTED FILE UPLOAD	17
3.7	SENSITIVE DATA DISCLOSURE.....	18
3.8	PHP INFORMATION DISCLOSURE	20
3.9	SESSION COOKIE SECURE FLAG NOT SET.....	21
4	Recommendations:.....	23

1 Executive Summary

SrivelEnterprise was contracted by Utopiic to conduct a **Web Application Vulnerability Assessment** to determine its exposure to a targeted attack. All activities were conducted in a manner that simulated a malicious actor engaged in a targeted attack against Utopiic with the goals of:

- ❖ Identifying if a remote attacker could penetrate Utopiic.
- ❖ Determining the impact of a security breach on:
 - Confidentiality of the company's private data
 - External infrastructure and availability of Utopiic information systems

Efforts were placed on the identification and exploitation of security weaknesses that could allow a remote attacker to gain unauthorized access to organizational data.

1.1 Project Scope

The assessment performed was focused on Utopiic Web Application. This result is intended to be an overall assessment of Utopiic applications that fall within the scope of this project.

Furthermore, the findings in this report reflect the conditions found during the testing, and do not necessarily reflect current conditions.

1.2 Project Objectives

This Report summarizes the results of the Assessment carried out to analyze the risks to the applications in Utopiic. Utopiic is constantly improving the IT infrastructure for providing enhanced services to its business users and customers. Protection of sensitive information is critical for Utopiic. Utopiic engaged SrivelEnterprise to perform a Web Application security assessment of its application. Based on the agreed scope and objective of the engagement as per the request, the following activities were performed:

- ❖ Identification of the vulnerabilities related to Utopiic Web Application.

This report presents the results of the engagement and recommendations. For this exercise, we conducted Automated and Manual review, one to one assessment and used some set of tools.

1.3 Target Systems

The following table lists all application that were targeted during this assessment.

S. No	Web Application Scope List
1	https://positiivplus.io/home/user_login

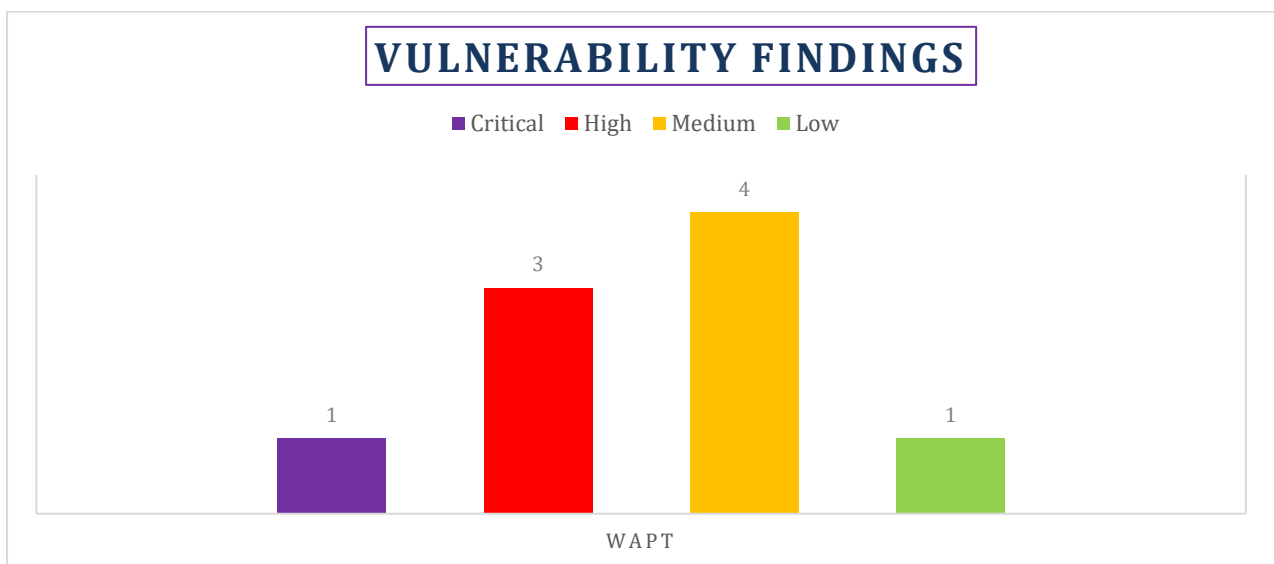
1.4 Assumption

We assumed that given hosts are Web Application only hosted and the organization has implemented the security policies available with them and the systems were installed and updated with latest antivirus and antivirus definitions. All the system and software applications are updated or upgraded to latest security updates and patches as suggested by the respective OEM.

1.5 Summary of Evaluation

- 👍 Analyze the services / ports on the given server. (Port Scanning)
- 👍 Understand the detail application and OS. (Finger Printing)
- 👍 Vulnerability Assessment on the application /OS. (Vulnerability Assessment)
- 👍 Match the corresponding exploit /risk. (Privilege Escalation)
- 👍 Summarize the individual risk and ratings. (Reporting)

S. No	Assessment Details	Critical	High	Medium	Low
1	Web Application PT Findings	1	3	4	1



1.6 Finding Rating Levels

In the following Findings section, we used a rating system using stars (*) to indicate the level of severity of our findings. All findings are vulnerabilities that have a business risk to the Utopiic.

4 Stars	****	Critical	Intruders can easily gain control of hosts and network. This needs immediate attention.
3 Stars	***	High	Intruders can possibly gain control of the host, or there may be potential leakage of highly sensitive information. This should be addressed as soon as possible.

2 Stars	**	Medium	Intruders may be able to collect sensitive information from the host, such as the precise version of software installed. With this information, intruders can easily exploit known vulnerabilities specific to software versions. Address this the next time you perform a minor reconfiguration of the host.
1 Stars	*	Low	Intruders can collect information about the host (open ports, services, etc.) and may be able to use this information to find other vulnerabilities. Address this the next time you perform a major reconfiguration of the host.
Informational			Serves as an informational purpose.

2 VULNERABILITY ISSUES MAPPING

S No	Web Application Penetration Testing Vulnerability	Severity	Status
1	Remote Code Execution	Critical	Open
2	Broken Authentication	High	Open
3	Stored XSS	High	Open
4	HTML Injection	High	Open
5	Improper Input Validation	Medium	Open
6	Unrestricted File Upload	Medium	Open
7	Sensitive Data Disclosure	Medium	Open
8	PHP Information Disclosure	Medium	Open
9	Session Cookie Secure Flag Not Set	Low	Open

2.1 High-Level Recommendation:

Based on the Web Application security Assessment, SrivelEnterprise recommends an action plan that should be addressed to mitigate all vulnerabilities found in this test report.

2.2 Immediate attention:

- Restrict file types accepted for upload: check the file extension and only allow certain files to be uploaded.
- Implement authentication.
- Sanitize every input that is being crafted by the client.

2.3 Later Attention:

- Ensure that the access to the pages or sensitive information's are restricted and proper access validation check is done on the sensitive pages.
- It is strongly recommended to make the data available only to the authorized user.



- Encryption of database password with strong algorithm can also be implanted here to reduce the impact.
- Disable the php info page.

.

3 Detailed Web Application Penetration Testing Findings

3.1 REMOTE CODE EXECUTION

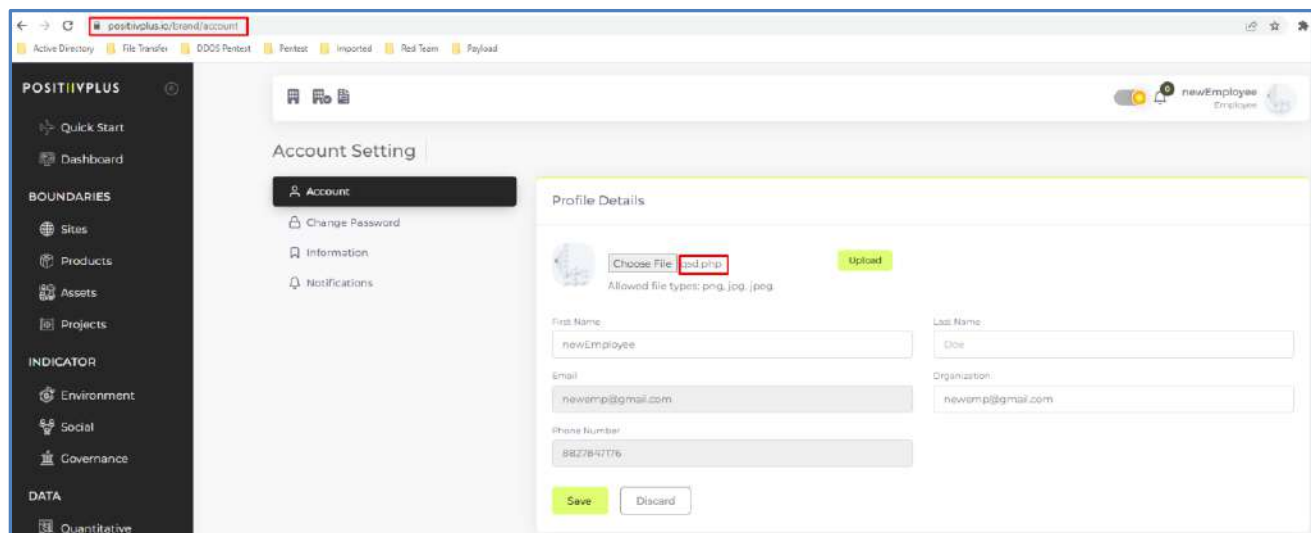
Threat Description: **Critical**

RCE vulnerabilities allow an attacker to execute arbitrary code on a remote device. Remote Code Execution is used to expose a form of vulnerability that can be exploited when user input is injected into a file or string and the entire package is run on the parser of the programming language.

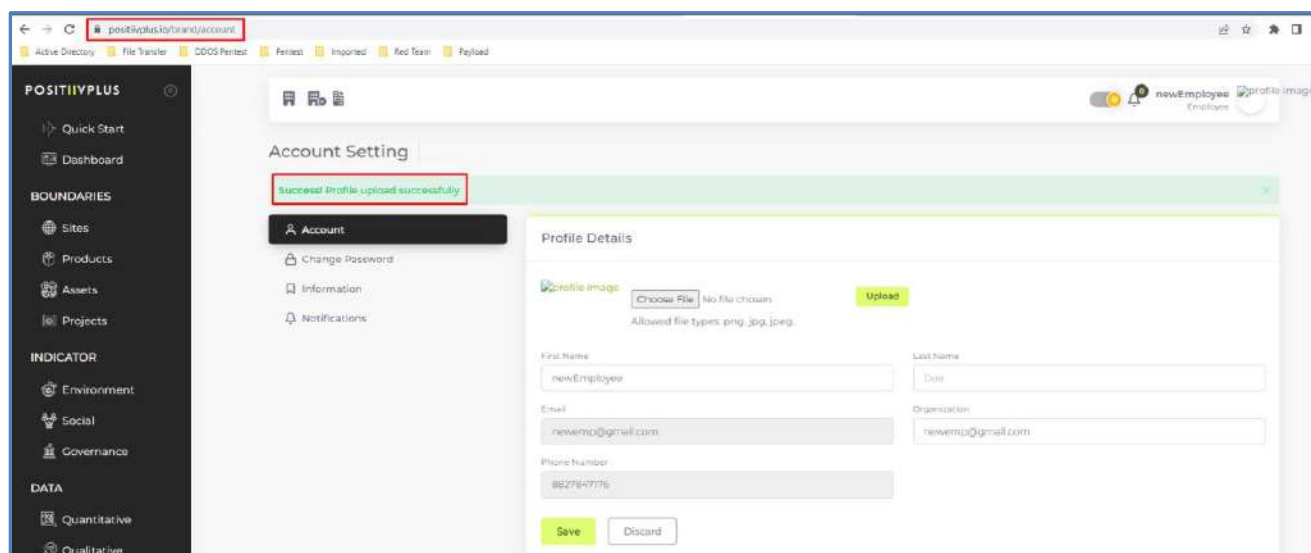
Methodology:

During the test, we observed that particular host is vulnerable for Remote Code Execution.

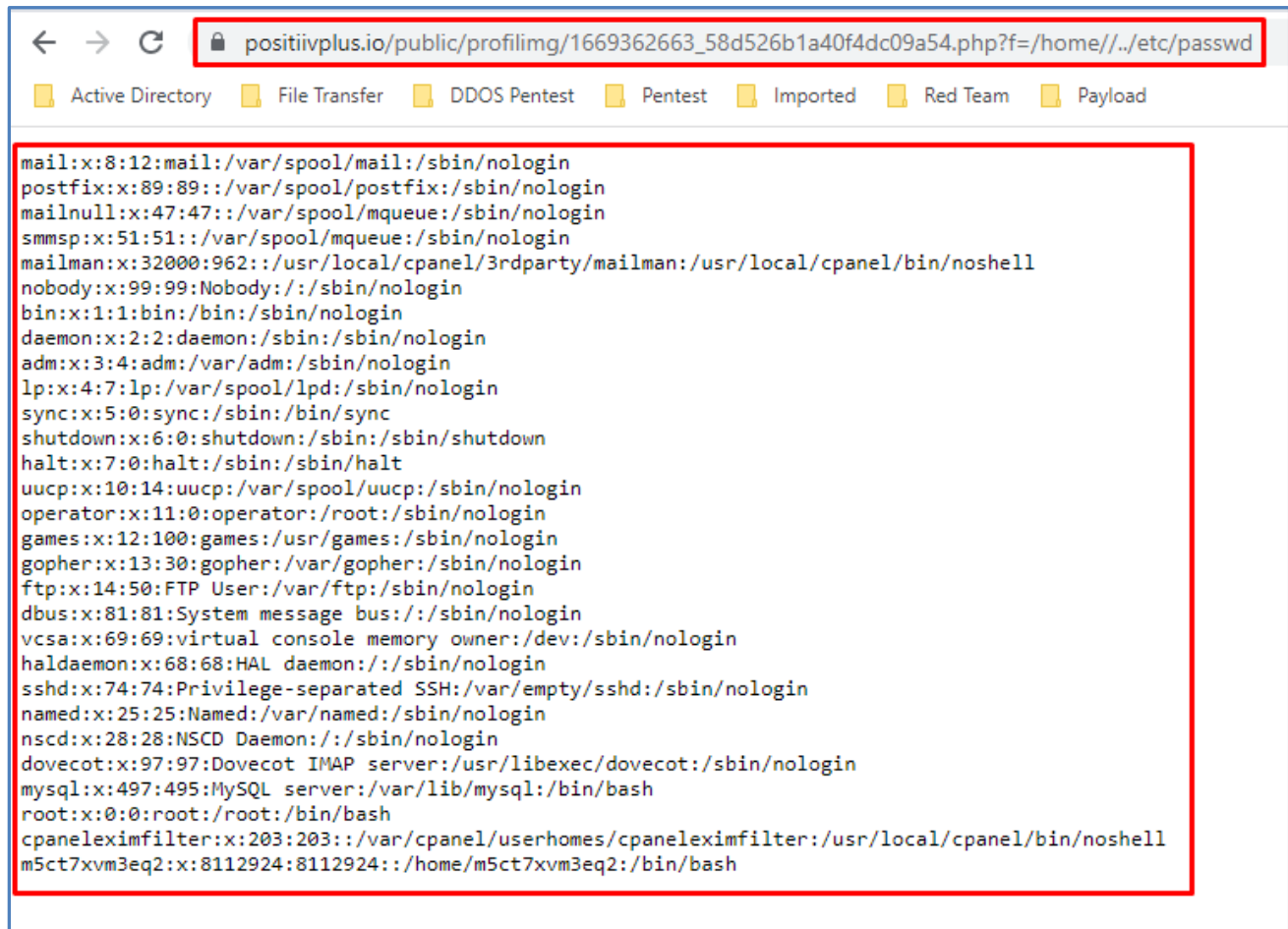
Step 1: First we tried to upload a malicious PHP file.



Step 2: The upload was successful and then open the image in new tab.



Step 3: Now we successfully got a webshell.



Impact:

RCE allows an attacker to discover and exploit these vulnerabilities, escalating privileges and gaining access to connected systems.

Affected Hosts:

URL	Proof
https://positiivplus.io/brand/account	Available

Recommendations:

- Sanitize inputs
- Validating and sanitizing user-supplied input before allowing the application
- Restrict file types accepted for upload: check the file extension and only allow certain files to be uploaded.

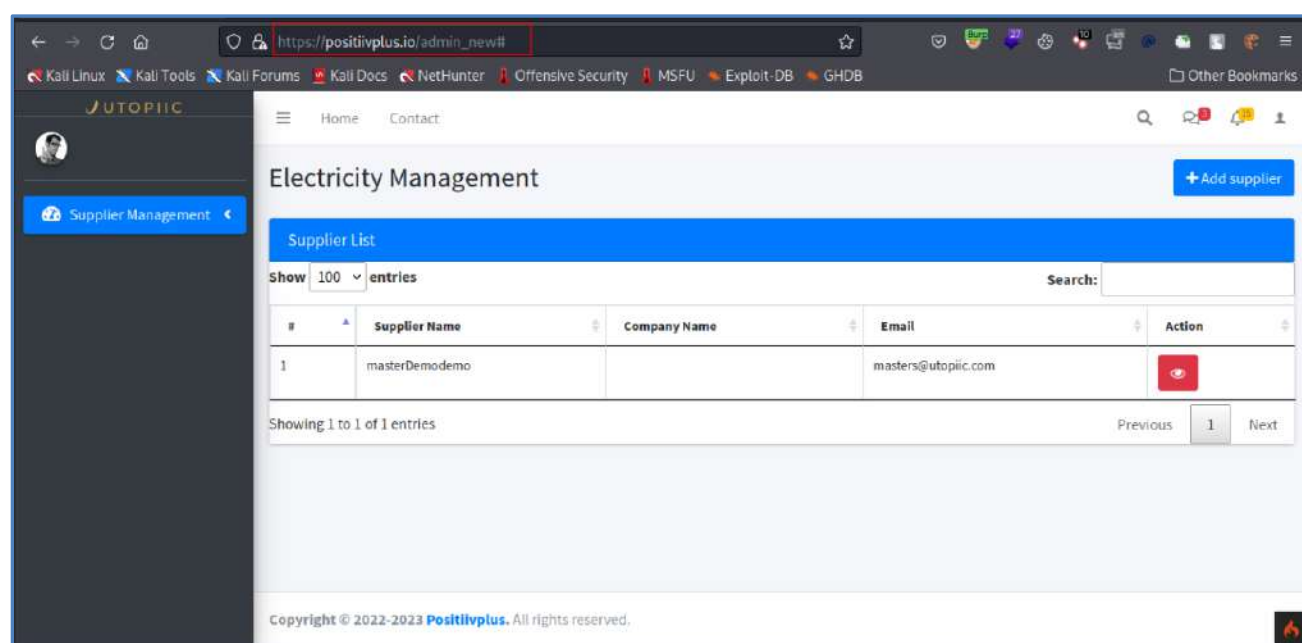
3.2 BROKEN AUTHENTICATION

Threat Description: **High**

Authentication and session management includes all aspects of handling user authentication and managing active sessions. Authentication is a critical aspect of this process, but even solid authentication mechanisms can be undermined by flawed credential management functions, including password change, forgot my password, remember my password, account update, and other related functions.

Methodology:

During our enumeration we found a directory called admin_news using directory bruteforce. And then we can able to access admin panel without any authentication.



Broken Authentication

Impact:

The goal of an attack is to take over one or more accounts, and for the attacker to get the same privileges as the attacked user. If the attacker successfully hijacks an admin account, the attacker could therefore do as much as an ordinary admin, which depending on the application could have a great impact. Other impact includes modification of data, adding and deleting new user, installing any backdoor application for continuous control over the application etc.

Affected Hosts

URL	Proof
https://tdsb.dhi-edu.com/tdsb_ipnss/#/forgotpassword/tdsb_ipnss	Available

Recommendations:

We recommend following the below methods to mitigate this vulnerability:

- Implement authentication.
- Generate a Unique session or access token for different user.
- Access token should be valid only for one-time login.
- User Input should be validated on every request.

- Multiple session supporting must be removed.
- Enforce concurrent login limits.
- Enforce session time limits to close the window for replay attacks.

3.3 STORED XSS

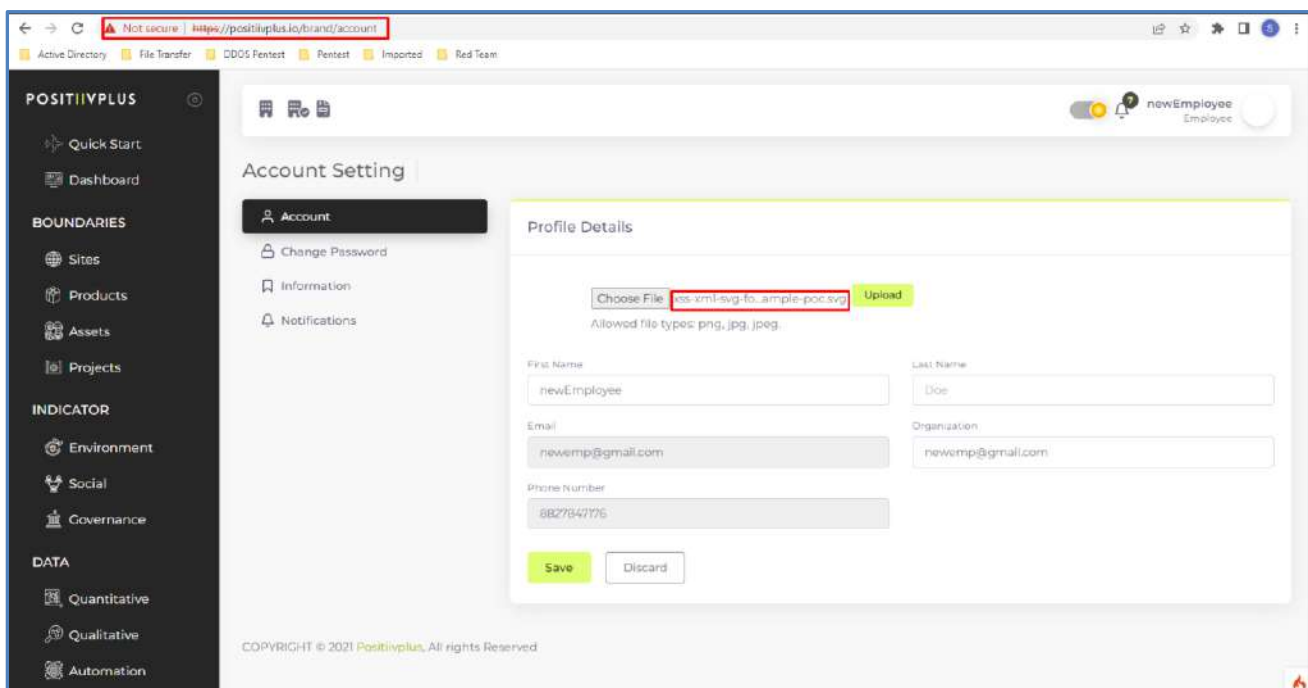
Threat Description: **High**

Cross-site Scripting (XSS) is a vulnerability, wherein an attacker can execute malicious scripts (also commonly referred to as a malicious payload) into a legitimate website or web application. XSS is amongst the most rampant of web application vulnerabilities and occurs when a web application makes use of un-validated or un-encoded user input within the output it generates.

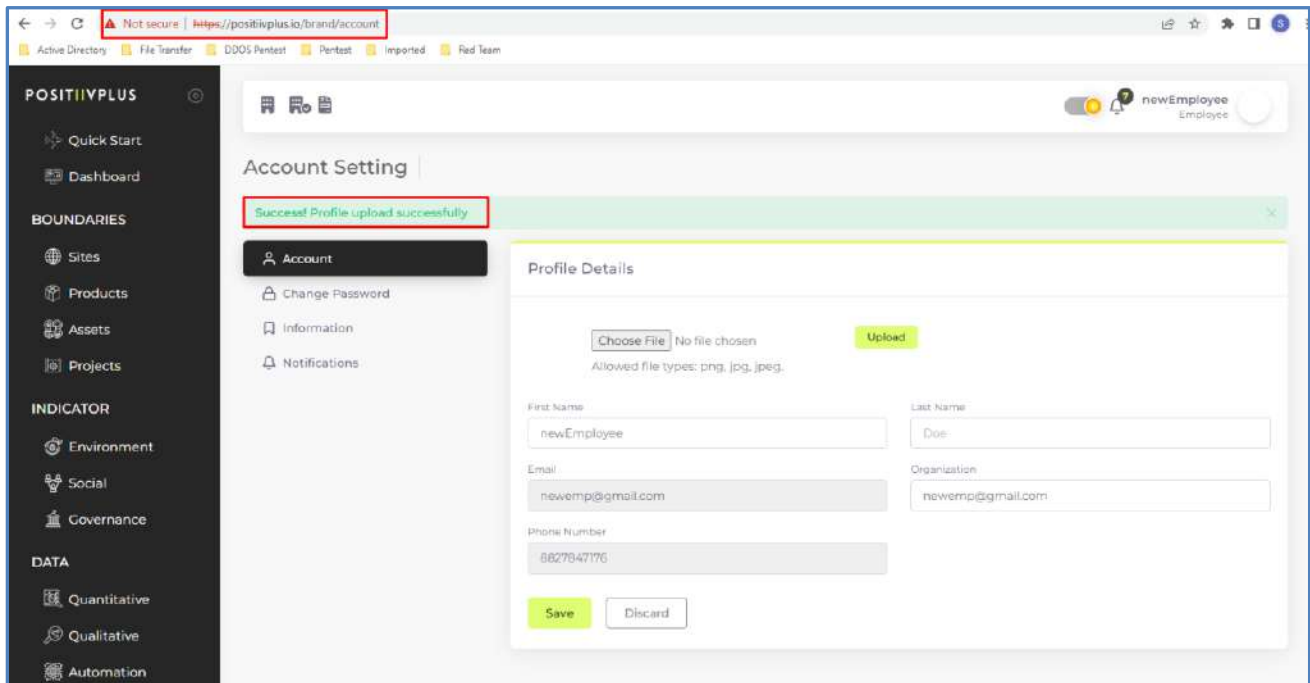
Methodology:

During the assessment we found the profile photo upload functionality is vulnerable for the stored XSS.

Step 1: Choose the malicious xss svg file for the profile photo upload functionality.



Step 2: Click the upload button the svg file was successfully upload in profile photo



Step 3: Select the profile photo open in new tab.



Stored XSS

Impact:

XSS can have huge implications for a web application and its users. User accounts can be hijacked, credentials could be stolen, sensitive data could be exfiltrated, and lastly, access to your client computers can be obtained.

Affected Hosts

URL	Proof
https://positivplus.io/brand/account	Available

Recommendations:

We recommend following methods to mitigate this vulnerability:

- Never Insert Untrusted Data except in Allowed Locations.
- HTML Escape before Inserting Untrusted Data into HTML Element Content.
- Attribute Escape before Inserting Untrusted Data into HTML Common Attributes.
- JavaScript Escape before Inserting Untrusted Data into JavaScript Data Values.
- URL Escape before Inserting Untrusted Data into HTML URL Parameter Values.
- Sanitize HTML Markup with a Library Designed for the Job.

- URL Escape then JavaScript Escape before Inserting Untrusted Data into URL Attribute Sub context within the Execution Context.
- Populate the DOM using safe JavaScript functions or properties.
- Implement Content Security Policy.
- Use an Auto-Escaping Template System.
- HTML Escape then JavaScript Escape before Inserting Untrusted Data into HTML Sub context within the Execution Context.
- Validate the all profile photo
- Only allow the Jpg, png image.

3.4 HTML INJECTION

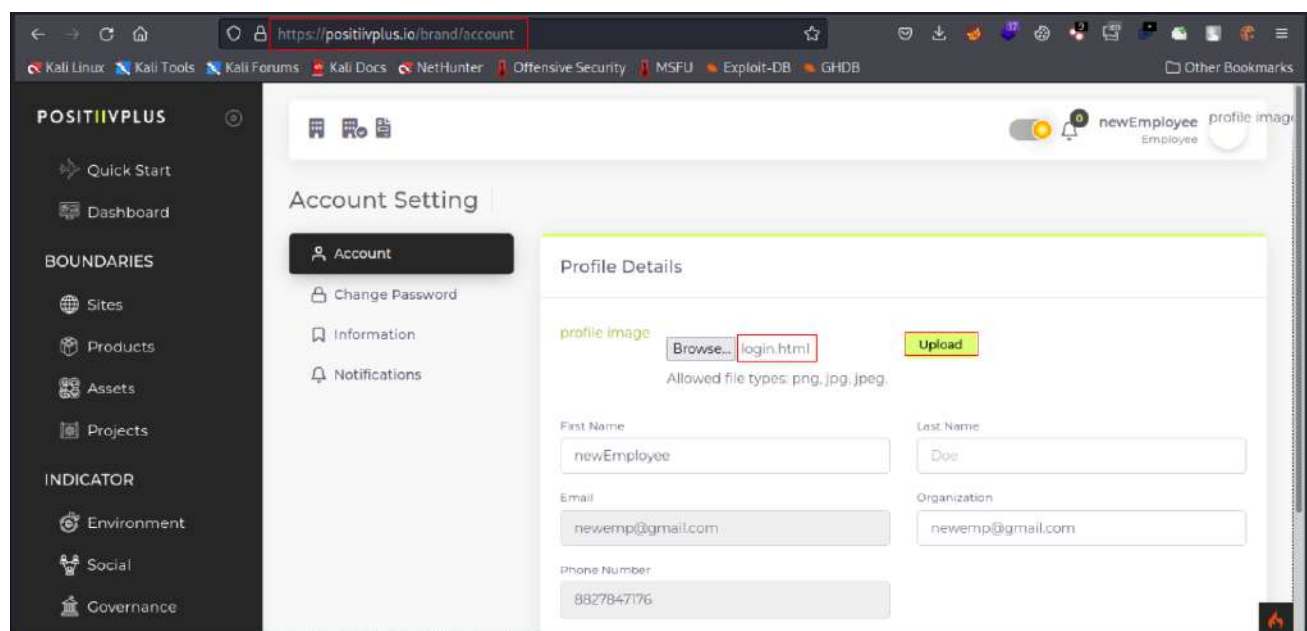
Threat Description: **High**

HTML injection is an attack that is like Cross-site Scripting (XSS). While in the XSS vulnerability the attacker can inject and execute JavaScript code, the HTML injection attack only allows the injection of certain HTML tags. When an application does not properly handle user supplied data, an attacker can supply valid HTML code, typically via a parameter value, and inject their own content into the page. This attack is typically used in conjunction with some form of social engineering, as the attack is exploiting a code-based vulnerability and a user's trust.

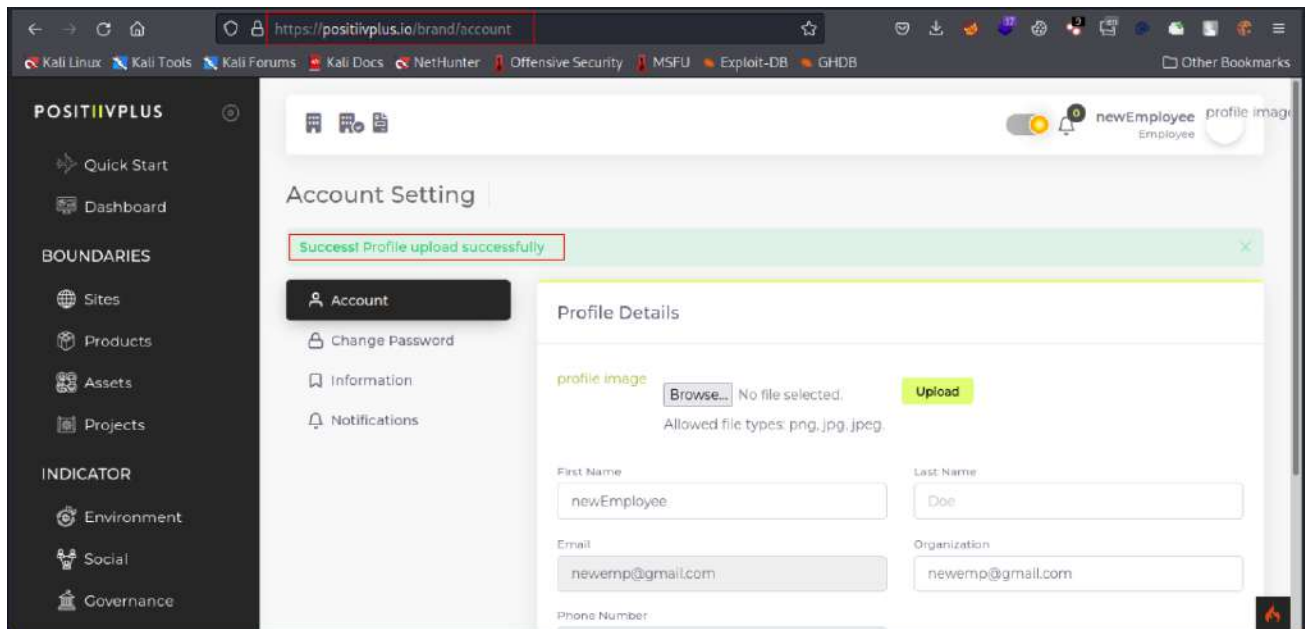
Methodology:

During the assessment we found the profile photo upload functionality is vulnerable for the HTML Injection.

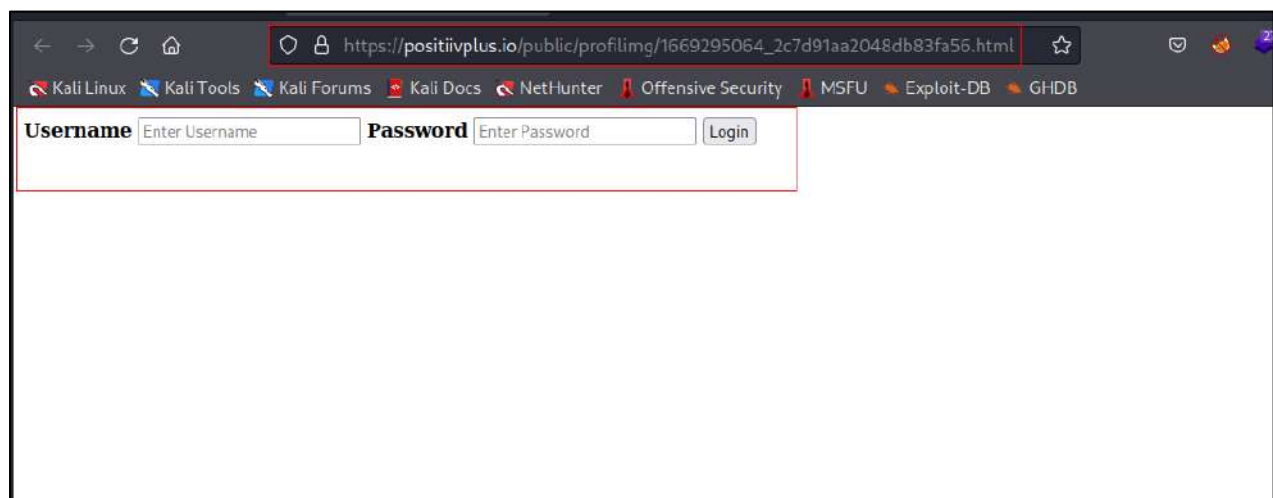
Step1: Select the profile photo and Choose the HTML malicious file.



Step2: Click to submit the HTML file was successfully upload in profile photo proceed to login.



Step 3: select the profile photo open in new tab.



HTML Injection

Impact:

Exploiting this vulnerability could lead to Website defacement, sensitive information loss, remote code execution and further complex and advanced attacks such as social engineering and spear-phishing attacks and spam attacks. With HTML code injection, the attacker could control which site the user views and how the site appears to the user and execute malicious scripts on the client side causing, remote code executions and defacements.

Affected Hosts

URL	Proof
https://positiivplus.io/brand/account	Available

Recommendations:

We recommend following the below methods to mitigate this vulnerability:

- Never Insert Untrusted Data except in Allowed Locations
- HTML Escape before Inserting Untrusted Data into HTML Element Content
- Attribute Escape before Inserting Untrusted Data into HTML Common Attributes

- JavaScript Escape before Inserting Untrusted Data into JavaScript Data Values
- CSS Escape and Strictly Validate Before Inserting Untrusted Data into HTML Style Property Values
- Validate the all profile photo
- Only allow the Jpg, png image.

3.5 IMPROPER INPUT VALIDATION

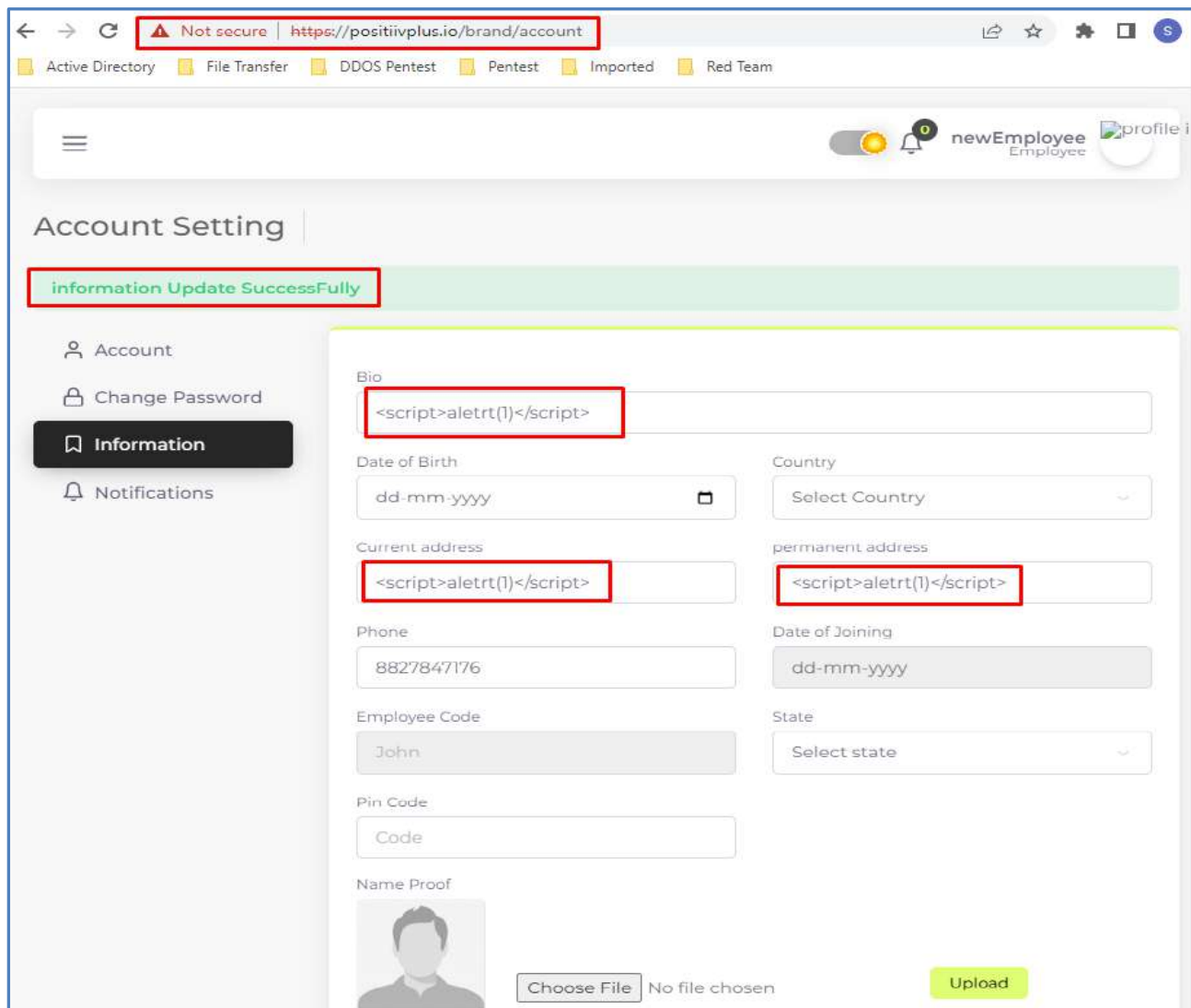
Threat Description: Medium

When application does not validate the input properly, an attacker is able to craft the input in a form that is not expected by the rest of the application. This will lead to parts of the system receiving unintended input, which may result in altered control flow, arbitrary control of a resource, or arbitrary code execution.

Methodology:

During the assessment, we found that the application was not properly validating, as shown in the below POC.

Step 1: We tried entering malicious script in the input fields and we can able to successfully update the information.



The screenshot shows a web browser at <https://positiivplus.io/brand/account>. The page title is "Account Setting". A green banner at the top says "information Update SuccessFully". On the left is a sidebar with "Account", "Change Password", "Information" (selected), and "Notifications". The main form contains fields for Bio, Date of Birth, Country, Current address, permanent address, Phone, Date of Joining, Employee Code, State, Pin Code, and Name Proof. The Bio, Current address, and permanent address fields contain the script payload `<script>alert(1)</script>`. The Bio field is highlighted with a red box. The Date of Birth field has a placeholder "dd-mm-yyyy". The Country field is a dropdown with "Select Country". The Phone field contains "8827847176". The Date of Joining field has a placeholder "dd-mm-yyyy". The Employee Code field contains "John". The State field is a dropdown with "Select state". The Pin Code field has a placeholder "Code". The Name Proof field has a placeholder image and a "Choose File" button. An "Upload" button is at the bottom right.

Improper Input Validation

Impact:

The root cause of Improper Input validation is that the application trusts all inputs, rather than validating, data inputs from the user. An attacker could potentially use this vulnerability to craft an exploit to cause arbitrary control of a resource, arbitrary code execution, cross-site-scripting, SQL injection, buffer overflow attack, or Remote code execution attacks in the web application.

Affected Hosts

URL	Proof
https://positiivplus.io/brand/account	Available

Recommendations:

We recommend following methods to mitigate this vulnerability:

- Use `filter_var($email, FILTER_VALIDATE_EMAIL)` to validate email or use Regex like `/^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+){2,3}$/`
- Need to implement input validation across the application.
- Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications or transform it into something that does. When performing input validation, consider all potentially relevant properties, including length,

type of input, the full range of acceptable values, missing or extra inputs, syntax and consistency across related fields, and conformance to business rules.

- Whitelist - Allowing only the known good characters. E.g. a-z,A-Z,0-9 are known good characters in the whitelist and are hence accepted by the filter.
- Blacklist - Allowing anything except the known bad characters. E.g. <, /,>,'," are known bad characters in the blacklist and are hence blocked by the filter.

3.6 UNRESTRICTED FILE UPLOAD

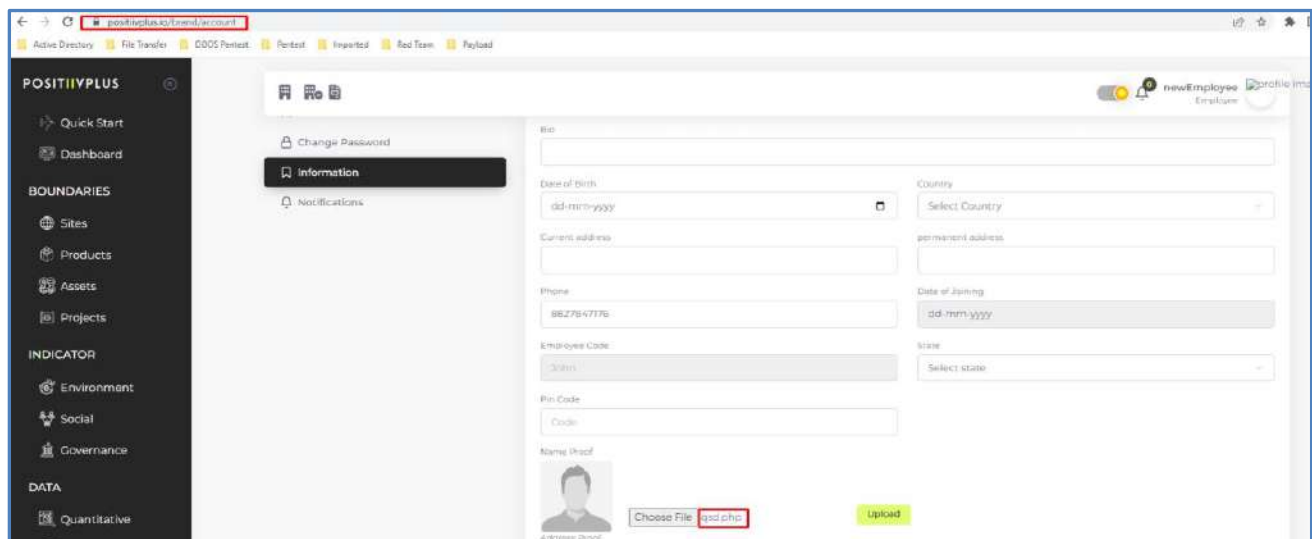
Threat Description: **Medium**

The application is vulnerable to unrestricted file uploads. Effective controls have not been implemented to restrict users from uploading malicious content to the web server. Files containing active code can be uploaded and executed on the server allowing for a number of serious attacks against the application and infrastructure.

Methodology:

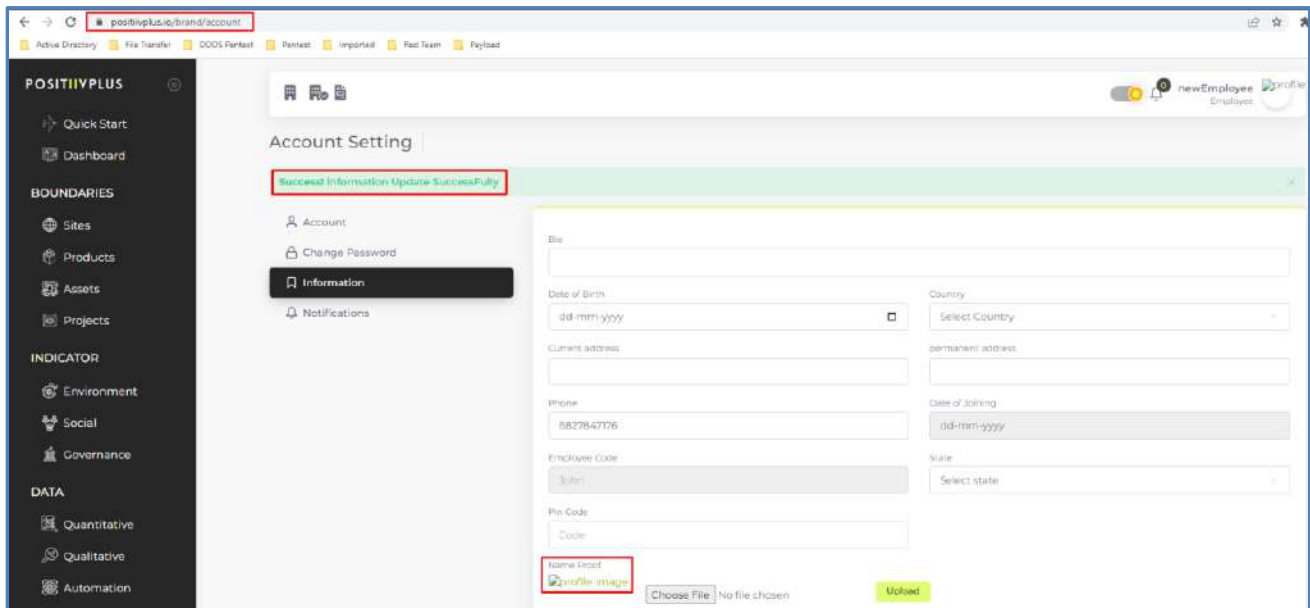
During the assessment, the web application was found to be vulnerable to unrestricted file upload vulnerability which allows the attacker to upload php files.

Step 1: We tried to upload to php file in the profile information tab.



The screenshot shows the 'POSITIVEPLUS' user interface. On the left is a sidebar with navigation options: Quick Start, Dashboard, BOUNDARIES (Sites, Products, Assets, Projects), INDICATOR (Environment, Social, Governance), and DATA (Quantitative). The main content area is titled 'Information' and contains various form fields for user profile data: Bio, Date of Birth (dd-mm-yyyy), Country (Select Country), Current address, permanent address, Phone (8827847776), Date of Joining (dd-mm-yyyy), Employee Code (3000), State (Select state), Pin Code (Code), Name (Proof), and Address (Proof). At the bottom, there is a 'Choose File' button (highlighted with a red box) and an 'Upload' button. A file named 'php.php' is shown as selected.

Step 2: We selected the php file and we click upload and then we can successfully upload file.



Unrestricted File Upload

Impact:

Unrestricted file upload vulnerability has significant impact on the application and its infrastructure. An attacker with the ability to upload a malicious file to the application can set up drive-by-download attacks or gain access to the file system through a web shell. Once an attacker has remote access to the server, they can exfiltrate sensitive data, compromise the integrity of application data, or cause denial of service to the application.

Affected Hosts

URL	Proof
https://positiivplus.io/brand/account	Available

Recommendations:

We recommend the below methods to mitigate this vulnerability.

- Restrict file types for upload: Check the file extension and only allow certain files to be uploaded.
- Use a whitelist approach instead of a blacklist. Check for double extensions such as .php.png.
- Check for files without a filename like .htaccess (on ASP.NET, check for configuration files like web.config). Change the permissions on the upload folder so the files are not executable.
- Ensure that configuration files such as ".htaccess" or "web.config" cannot be replaced using file uploaders. Ensure that appropriate settings are available to ignore the ".htaccess" or "web.config" files if uploaded in the upload directories.
- Ensure that files with double extensions (e.g. "file.php.txt") cannot be executed.

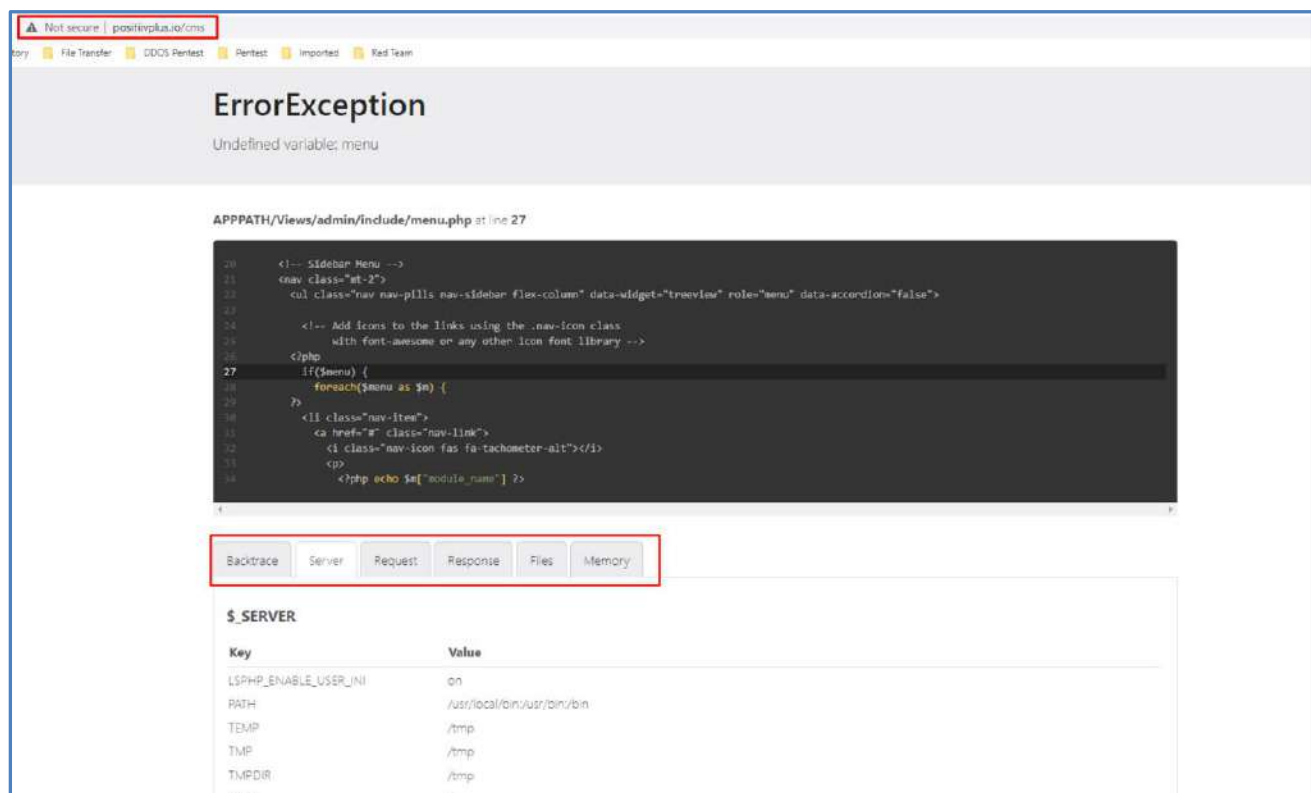
3.7 SENSITIVE DATA DISCLOSURE

Threat Description: **Medium**

Sensitive Data Exposure occurs when an application does not adequately protect sensitive information. The data can vary and anything from username, passwords, session tokens, credit card, internal email communication to private health data and more can be exposed.

Methodology:

During our enumeration we found a directory called cms using directory bruteforce.



Application Information Disclosure

Impact:

As the finding only applies to sensitive data, the potential impact is always considered Critical. What the data consists of varies and so does the impact. The danger lies in the data being exposed, and the potential impact reflects the data's sensitivity.

For example, if credit card data is stolen, the attacker can empty the victim's bank account. If passwords are exposed, the attacker can abuse these credentials. In our testing we observed the customer email id, phone no and account number being exposed without any masking of this personal information. In real time a hacker can gain this information and use it for his advantage. There are even chances the attacker might sell this valuable information to other organization or companies.

Affected Hosts

URL	Proof
https://positivplus.io/cms	Available

Recommendations:

We recommend the following below:

- Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Ensure that the access to the pages or sensitive information's are restricted and proper access validation check is done on the sensitive pages.

- Do not store sensitive data unless it is important.
- It is strongly recommended to make the data available only to the authorized user.

Encryption of database password with strong algorithm can also be implanted here to reduce the impact.

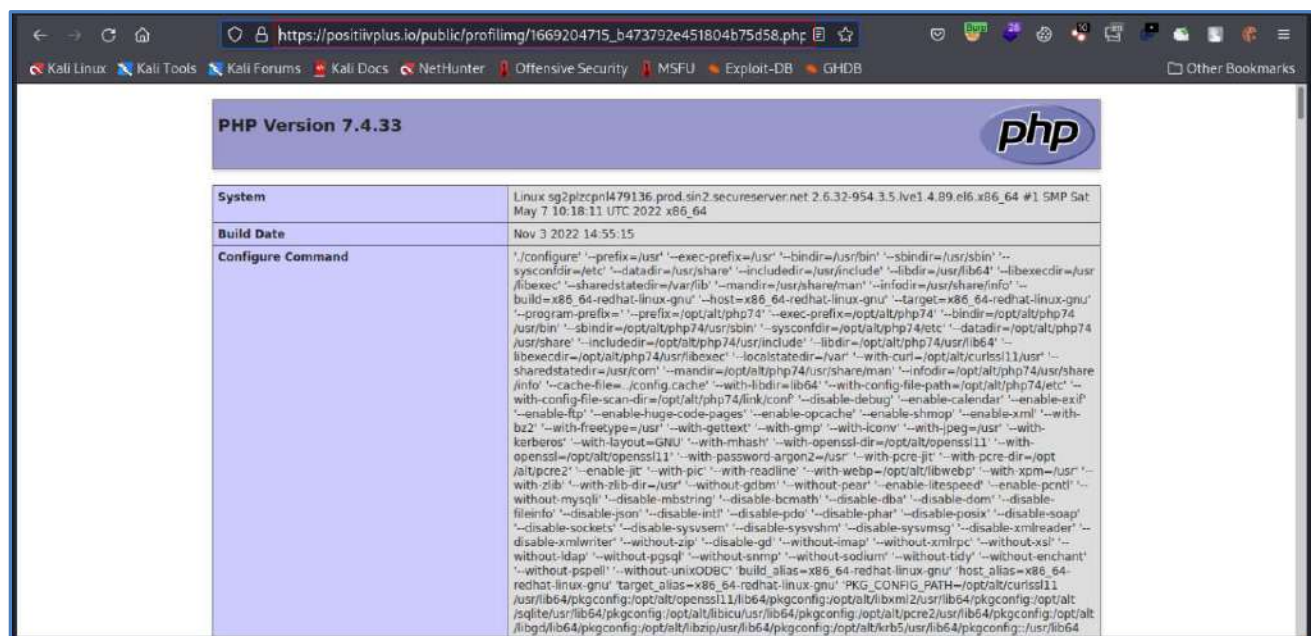
3.8 PHP INFORMATION DISCLOSURE

Threat Description: **Medium**

During the assessment it was found that requesting servers '/1669362663_58d526b1a40f4dc09a54.php' page from the host, gives information about the currently running apache web server configuration, Admin mail and OS information.

Methodology:

This page was identified while brute forcing the directory and files in the application. The 1669362663_58d526b1a40f4dc09a54.php input in the URL was successful, and the entire information was disclosed. Anyone with an internal access will be able to access the server status just by visiting the page



PHP Information Disclosure

Impact:

The php.info function is a powerful one, you can learn quite a lot about your PHP installation. Sensitive information such as process ID, configuration file path, cache info and session details are exposed to the internet by accessing this default path. This will help the attacker to launch or even automate more powerful attacks.

Affected Hosts

URL	Proof
https://positiivplus.io/public/profilimg/1669362663_58d526b1a40f4dc09a54.php?phpinfo=true	Available

Recommendations:

It is recommended to disable this function if not needed or restrict access to unauthorized users.

- To disable the php info page, follow the below steps:
- Go to server's php.ini file.
- Now change the line that includes the disable_functions directive, so that it says
disable_functions = phpinfo.
- Restart the web server daemon to put this change into effect.

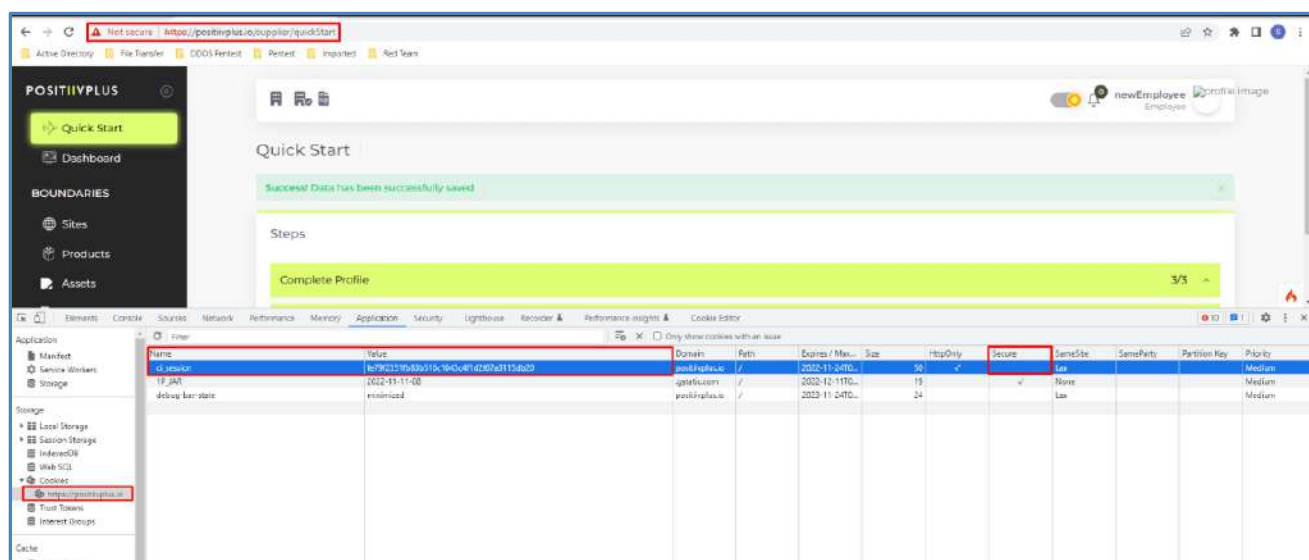
3.9 SESSION COOKIE SECURE FLAG NOT SET

Threat Description: **Low**

If the secure flag is set on a cookie, then browsers will not submit the cookie in any requests that use an unencrypted HTTP connection, thereby preventing the cookie from being trivially intercepted by an attacker monitoring network traffic. If the secure flag is not set, then the cookie will be transmitted in clear-text.

Methodology:

The remote web application sets various cookies throughout a user's unauthenticated and authenticated session. However, during the assessment it was found that one or more of those cookies are not set with a 'Secure flag', meaning that a malicious client-side script, such as JavaScript, could read them.



Session Cookie Secure Flag Not Set

Impact:

If the secure flag is not set, then the cookie will be transmitted in clear-text if the user visits any HTTP URLs within the cookie's scope. An attacker may be able to induce this event by feeding a user suitable links, either directly or via another web site. Even if the domain that issued the cookie does not host any content that is accessed over HTTP, an attacker may be able to use links of the form ("http://example.com:443/ ") to perform the same attack. Once the attacker has gained the plaintext Cookies or session, it can lead to a Session Hijacking attack.

Affected Hosts

URL	Proof
-----	-------

Recommendations:

- The secure flag is an option that can be set by the application server when sending a new cookie to the user within an HTTP Response.
- The secure flag should be set on all cookies that are used for transmitting sensitive data when accessing content over HTTPS.
- Setting up secure cookies depends upon the technology in use. We recommend some of the sample code or syntax to set the secure cookie flag.

In Apache:

Go to the Apache2.conf file and set the secure cookie flag as follows:

```
LoadModule headers_module /usr/lib/apache2/modules/mod_headers.so
```

```
Header edit Set-Cookie ^(.*)$ $1;HttpOnly;Secure
```

Java:

Servlet 3.0 (Java EE 6) introduced a standard way to configure secure attribute for the session cookie, this can be done by applying the following configuration in web.xml.

```
<session-config>  
<cookie-config>  
  <secure>true</secure>  
</cookie-config>  
</session-config>
```

4 Recommendations:

As identified in the assessment the current security controls implemented are not adequate, as a part of immediate sealing drills the following steps can be taken.

- 👍 Restrict file types accepted for upload: check the file extension and only allow certain files to be uploaded.
- 👍 Implement authentication.
- 👍 Sanitize every input that is being crafted by the client.
- 👍 Ensure that the access to the pages or sensitive information's are restricted and proper access validation check is done on the sensitive pages.
- 👍 It is strongly recommended to make the data available only to the authorized user.
- 👍 Encryption of database password with strong algorithm can also be implanted here to reduce the impact.
- 👍 Disable the php info page.

We recommend that Utopiic create a detailed plan for closure of the gaps found during this penetration test as soon as possible. The closure plan should address the Critical followed by High – level vulnerabilities and Medium – level findings. The closure plan should be tested before making any changes to the production environment. Some of the recommendations for the closure plan are:

- 👍 Test the critical application on a periodic basis either quarterly or half yearly.
- 👍 Develop and render an awareness program to keep employees up to date on information security and its implications.

We thank Utopiic for giving us this opportunity and we assure our full assistance in securing information and IT infrastructure in the long run.