

SRI LANKA INSTITUTE OF INFORMATION TECHNOLOGY



Web Security- IE2062

Journal

Danuka Nuwan

IT22349842

Table of Contents

Date -23 April 2024.....	4
Summary of the day's activities:	4
Date -24 April 2024.....	5
Overview of the day's events:	5
Acquired knowledge of novel tools, techniques, or concepts:.....	5
Reflections and Takeaways:	6
Date: April 25, 2024.....	10
Date:April 26, 2024	15
New tools, skills, or concepts learned:	15
Reflections and takeaways:	15
Date: April 27, 2024.....	20
Summary of the day's activities:	20
Vulnerabilities discovered or explored:	20
Challenges faced and how they were overcome:.....	24
Reflections and takeaways:	24
Reflections and takeaways:	24
Date:April 28, 2024	25
Summary of the day's activities	25
Vulnerabilities discovered or explored:	25
Challenges faced and how they were overcome:.....	34
New tools, skills, or concepts learned:	35
Reflections and takeaways:	35
Date:April 29, 2024	36
Overview of the day's events:	36
Vulnerabilities Identified or Explored:	36
New tools, skills, or concepts learned:	38
Reflections and Takeaways:	39
Challenges Faced and How They Were Overcome:.....	39
Date:April 30, 2024	40
Overview of the day's events:	40
Vulnerabilities Identified or Explored:	40
New tools, skills, or concepts learned:	44

Challenges Faced and How They Were Overcome:.....	44
Reflections and Takeaways:	44
Date: May 1, 2024	45
Overview of the day's events:	45
Vulnerabilities Identified or Explored:	45
Challenges Faced and How They Were Overcome.....	47
New tools, skills, or concepts learned:	47
Reflections and Takeaways:	47
Date: May 3, 2024	48
Overview of the day's events:	48
Vulnerabilities Identified or Explored	48
Challenges Faced and How They Were Overcome:.....	49
New tools, skills, or concepts learned:	49
Reflections and Takeaways:	50
Date: May 4, 2024	51
Overview of the day's events:	51
Vulnerabilities Identified or Explored:	51
Challenges Faced and How They Were Overcome:.....	53
New tools, skills, or concepts learned:	53
Reflections and Takeaways:	54

Date -23 April 2024

Summary of the day's activities: Today, our focus was on doing thorough reconnaissance and collecting data to prepare for upcoming security assessments. I concentrated on thoroughly searching platforms such as HackerOne and Bugcrowd to pinpoint possible subjects for conducting penetration testing and vulnerability research.

Platforms Explored: I spent the entire day carefully searching through different platforms like HackerOne and Bugcrowd, thoroughly examining the listings and reports to find websites that are appropriate for security evaluations. In this phase of reconnaissance, I analyzed the extent, seriousness, and reputation of potential targets, making sure they matched my expertise and objectives.

Information acquiring: Besides exploring the platform, a substantial amount of time was dedicated to acquiring information on the discovered websites. This involved doing thorough investigations into the target's technology stack, infrastructure, documented vulnerabilities, previous security breaches, and any pertinent public disclosures. Through the compilation of extensive intelligence about the target, my objective was to gain a deeper understanding of its attack surface and identify potential vulnerabilities.

Tools and Techniques: Various tools and techniques were deployed during the information gathering process, including open-source intelligence (OSINT) tools, web scraping utilities, and reconnaissance frameworks. These tools facilitated the effective collection of data from several sources, including social media, public databases, and WHOIS information. This allowed me to gain significant insights about the target's digital presence.

Reflections and Takeaways: Today's activities underlined the crucial significance of reconnaissance in cybersecurity assessments. By investing time and effort in detailed information collecting, I created a solid foundation for future penetration testing operations. Engaging in platform exploration and information gathering not only improved my technical proficiency but also honed my analytical acumen in detecting potential security vulnerabilities. As I continue this journey, I am reminded of the need of thorough preparation and continual learning in the world of cybersecurity.

Date -24 April 2024

Overview of the day's events:

Today, I devoted my efforts to investigating SQL injection vulnerabilities and implementing them in a real-world scenario. I conducted an in-depth examination of the Navient website, specifically listed on the HackerOne platform, with the aim of detecting any possible weaknesses or vulnerabilities.

<https://navient.com/>

Vulnerabilities Identified or Explored:

After conducting comprehensive testing, I did not detect any instances of SQL injection vulnerabilities on the Navient website. This event underlined the significance of rigorous testing and the differing levels of security implemented by different websites.

I go through many platforms like netsparker, owasp zap and burp suite to find vulnerability and but in the end this website didn't have any sql vulnerability.

Challenges Faced and How They Were Overcome: One of the obstacles I confronted was the server impeding my requests during the last stages of the SQL injection attempts. In order to address this issue, I modified my testing methodology and investigated various attack pathways. Although my testing scope was restricted, it also helped me develop the ability to adjust to problems and improve my approaches.

Attempt to defeat popular SQLi mitigation mechanisms, such as input validation and Explore more complex SQLi tactics, such as time-based attacks or blind attacks.
I also Learned about web application firewalls (WAFs) and how they can help prevent SQLi attacks.

Acquired knowledge of novel tools, techniques, or concepts:

While studying SQL injection, I became acquainted with advanced tools like SQLmap, which greatly enhanced the efficiency and automation of the testing procedure. Additionally, I received insights into various evasion strategies and approaches to defeat security safeguards.

I also find out new tool name sqlfluffle.

Reflections and Takeaways: Although I did not succeed in uncovering vulnerabilities on the Navient website, this experience served as a wonderful learning opportunity. The significance of tenacity, adaptation, and continual learning in the realm of cybersecurity was strengthened. Furthermore, it underlined the need for robust security measures to protect against typical attacks like SQL injection.

Overall, while the result may not have been as intended, the knowledge and skills obtained from this experience will surely help to my progress as a cybersecurity enthusiast.

Navient Solutions LLC
Enhancing the financial success of our customers by delivering innovative solutions with compassion and personalized service.
<https://navient.com> · @navient

Reports resolved: 10 | Assets in scope: 23

[Submit report](#)

Vulnerability Disclosure Program
Launched in Nov 2023
Managed by HackerOne

[Give feedback](#) | [Bookmark](#) | [Subscribe](#)

[Policy](#) | [Scope](#) | [Hacktivity](#) | [Thanks](#) | [Updates \(0\)](#)

Policy

Maintaining the security of our applications and networks is a high priority for Navient Solutions LLC. If you have information related to security vulnerabilities of Navient products and services, please submit a report in accordance with the guidelines below.

- The vulnerabilities identified in the HackerOne reports will be classified by the degree of risk as well as the impact they present to the host system, this includes the amount and type of data exposed, privilege level obtained, proportion of systems or users affected.
- Do not try to further pivot into the network by using a vulnerability. The rules around Remote Code Execution (RCE), SQL Injection (SQLi), vulnerabilities allowing you to access file/folder structure, defacement and file uploads are listed below.
- Navient has a comprehensive vulnerability management program that includes receiving and acting on security notifications from third-party vendors. Published vulnerabilities in supported third-party and open source products (e.g., network infrastructure, operating systems, application servers) are not eligible for submission until 45 days after publication.
- Do not try to exploit service providers we use, prohibited actions include, but are not limited to bruteforcing login credentials of Domain Registrars, DNS Hosting Companies, Email Providers and/or others. Navient Solutions LLC does not authorize you to perform any actions to a non-Navient owned property/system/service/data.
- If you encounter Personally Identifiable Information (PII) contact us through the Hackerone portal immediately. Do not proceed with access and immediately purge any local information, if applicable.
- Please limit any automated scanning to 60 requests per second. Aggressive testing that causes service degradation will be grounds for removal from the program.

Response Efficiency

3 days
Avg time to first response

4 days
Avg time to triage

...

Avg time to close

94% of reports
Meet response standards
Based on last 90 days

Program Statistics

Updated Daily

4

Policy

Maintaining the security of our applications and networks is a high priority for Navient Solutions LLC. If you have information related to security vulnerabilities of Navient products and services, please submit a report in accordance with the guidelines below.

- The vulnerabilities identified in the HackerOne reports will be classified by the degree of risk as well as the impact they present to the host system, this includes the amount and type of data exposed, privilege level obtained, proportion of systems or users affected.
- Do not try to further pivot into the network by using a vulnerability. The rules around Remote Code Execution (RCE), SQL Injection (SQLi), vulnerabilities allowing you to access file/folder structure, defacement and file uploads are listed below.
- Navient has a comprehensive vulnerability management program that includes receiving and acting on security notifications from third-party vendors. Published vulnerabilities in supported third-party and open source products (e.g., network infrastructure, operating systems, application servers) are not eligible for submission until 45 days after publication.
- Do not try to exploit service providers we use, prohibited actions include, but are not limited to bruteforcing login credentials of Domain Registrars, DNS Hosting Companies, Email Providers and/or others. Navient Solutions LLC does not authorize you to perform any actions to a non-Navient owned property/system/service/data.
- If you encounter Personally Identifiable Information (PII) contact us through the Hackerone portal immediately. Do not proceed with access and immediately purge any local information, if applicable.
- Please limit any automated scanning to 60 requests per second. Aggressive testing that causes service degradation will be grounds for removal from the program.

Thank you for helping keep Navient Solutions LLC and our users safe!

Response Targets

Navient Solutions LLC will make a best effort to meet the following SLAs for hackers participating in our program:

Type of Response	SLA in business days
First Response	2 days
Time to Triage	2 days
Time to Resolution	depends on severity and complexity

We'll try to keep you informed about our progress throughout the process.

SCOPE

Please carefully review the scope section below. Many of our web properties may have "Navient" in the domain name, but may be 3rd party hosted and as such, are out of scope. In addition, an in scope asset may linked to or redirect away from our network range., but the resulting external link are also NOT in scope for the program.

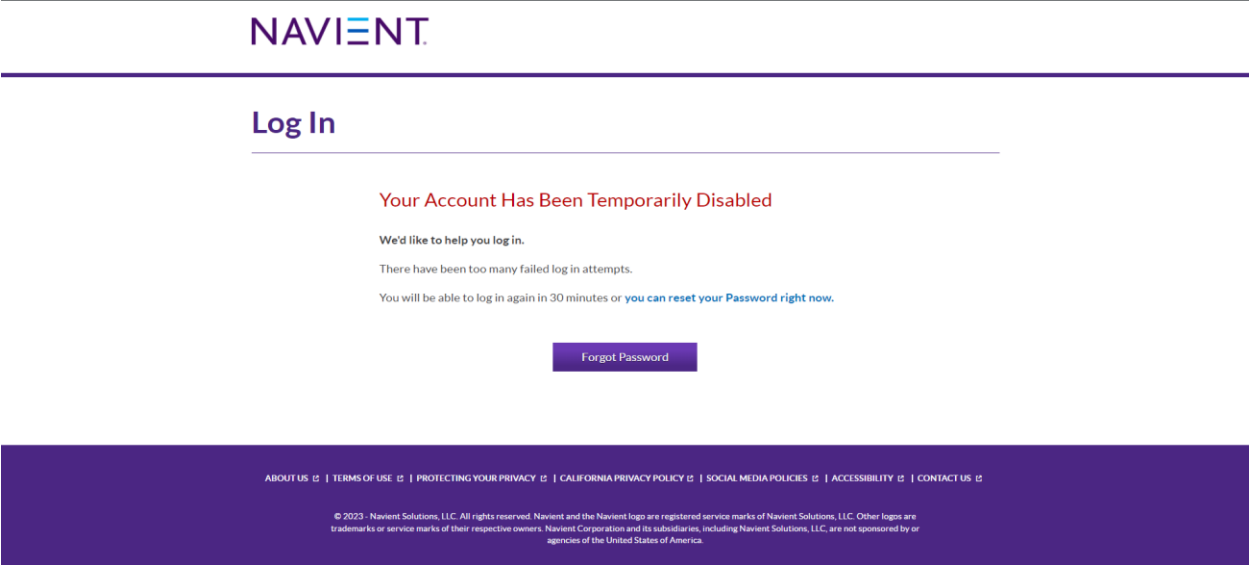
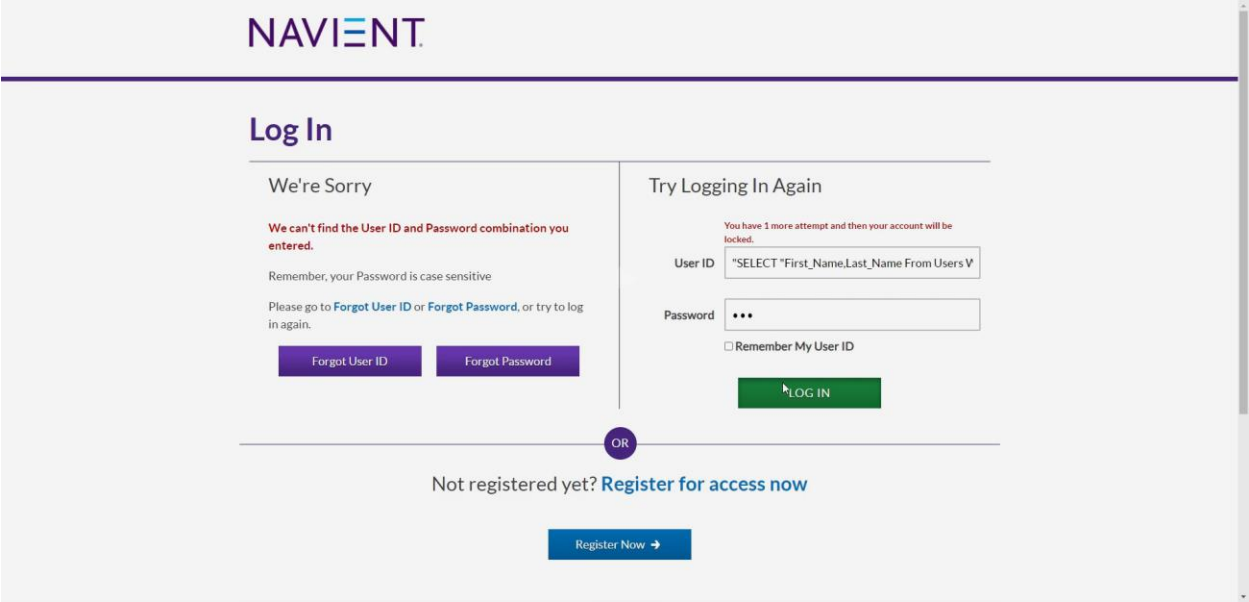
Scope determination criteria:

In addition to the applications that are explicitly listed as in or out of scope, only sites that resolve to the IP address ranges below should be considered in scope. Anything outside of these IP address ranges is out of scope (exceptions noted below).

167.104.0.0/16
207.250.125.0/28

Exceptions To IP Ranges Above (Assets below are in-scope and route through Akamai with a different IP range):

www.navient.com
www.dsparkingportal.com
www.dspayments.com
duncan.imageenforcement.com
www.pampayments.com
www.pamcollections.com
www.pamcollectionsin.com
sandiegoadmin.dsmyportal.com
sandiego.dsmyportal.com
www.nystapayment.com



Date: April 25, 2024

Summary of the day's activities: Today, I went on an exploration into Cross-Site Scripting (XSS) vulnerabilities, focusing my attention on the Temu website, an online marketplace listed on the HackerOne platform. Despite experiencing hurdles along the way, the day was filled with important learning experiences.

<https://www temu.com/>

Vulnerabilities detected or explored: Despite thorough testing efforts, no exploitable vulnerabilities were identified on the Temu website through manual review alone. However, a second time utilizing Netsparker discovered probable vulnerabilities or error states that could be symptomatic of underlying security problems. This result underlines the significance of adopting a variety of testing approaches and tools to comprehensively analyze the security posture of web services. While human testing may not always give instant results, supplementary automated scans might identify hidden vulnerabilities or weaknesses that could otherwise go unreported. Thus, the absence of vulnerabilities through manual testing does not necessarily reflect the absence of security issues, underscoring the necessity for thorough and multifaceted methods to web application security testing.

Challenges faced and how they were overcome: Throughout the examination, I faced various problems, including the application of server-side blocking measures. Despite my attempts to evade these difficulties, I was met with minimal success, underlining the strength of the website's protective measures. However, each hurdle served as a chance for advancement, motivating me to explore additional routes for exploitation and further polishing my talents in the process.

New tools, skills, or concepts learned: Today's exploration not only expanded my knowledge of XSS vulnerabilities but also introduced me to a great addition to my toolkit: PwnXSS. PwnXSS is a free and open-source automated XSS vulnerability scanner program available on GitHub. This Python-based tool is specifically designed to detect cross-site scripting vulnerabilities, easing the process of finding XSS problems in web applications.

In addition to PwnXSS, I researched other tools such as Cross Site "Scripter" (XSSer), which offers advanced capabilities for XSS detection and exploitation. These tools, paired with insights from the XSS cheatsheet accessible on GitHub, provided thorough coverage of XSS testing approaches and procedures, boosting my proficiency in web security assessments.

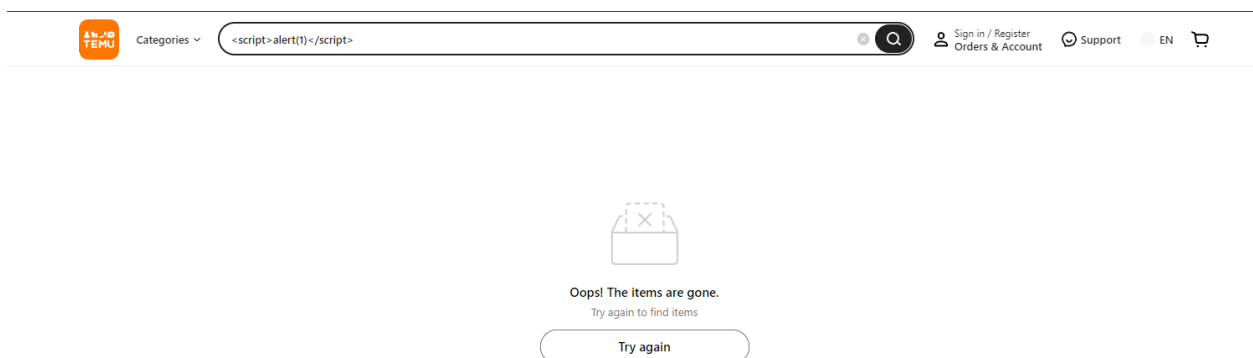
<https://github.com/payloadbox/xss-payload-list>

Reflections and takeaways: Today's endeavors highlighted the delicate nature of web security and the constant growth of attack and defense strategies. While the challenges faced were severe, they served as drivers for learning and growth. The discovery of XSS vulnerabilities on the Temu website underlined the need of proactive security measures in securing online applications from malicious exploitation. Moving forward, I am equipped with newfound information and tools, ready to tackle the ever-changing landscape of cybersecurity with resilience and resolve.

Manual method



I found this web site on hackerone platform and I tried to use some simple xss script.



But it didn't go as planned. So as anext step I tried to use netsparker (automation tool) for find if there any xss vulnerability.

Vulnerability Summary

CONFIRM	VULNERABILITY	METHOD	URL	PARAMETER
	 [Possible] Cross-site Scripting	GET	https://temu.com/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x000023)%3C/scRipt%3E	nsextt
	 [Possible] Cross-site Scripting	GET	https://temu.com/.well-known/apple-app-site-association?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x000034)%3C/scRipt%3E	nsextt
	 [Possible] Cross-site Scripting	GET	https://temu.com/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x000040)%3C/scRipt%3E	nsextt
	 [Possible] Cross-site Scripting	GET	https://temu.com/sitemap.xml'%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x00007A)%3C/scRipt%3E	URI-BASED

After completing test finally I found some proof this site has some xss vulnerability. But it is possible.

1. [Possible] Cross-site Scripting

MEDIUM 4

Netsparker detected Possible Cross-site Scripting, which allows an attacker to execute a dynamic script (*JavaScript, VBScript*) in the context of the application.

This allows several different attack opportunities, mostly hijacking the current session of the user or changing the look of the page by changing the HTML on the fly to steal the user's credentials. This happens because the input entered by a user has been interpreted as HTML/JavaScript/VBScript by the browser. Cross-site scripting targets the users of the application instead of the server. Although this is a limitation, since it allows attackers to hijack other users' sessions, an attacker might attack an administrator to gain full control over the application.

Although Netsparker believes there is a cross-site scripting in here, it could **not confirm it**. We strongly recommend investigating the issue manually to ensure it is cross-site scripting and needs to be addressed.

Impact

There are many different attacks that can be leveraged through the use of XSS, including:

- Hijacking user's active session.
- Changing the look of the page within the victim's browser.
- Mounting a successful phishing attack.
- Intercepting data and performing man-in-the-middle attacks.

Vulnerabilities

1.1. [https://temu.com/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3E%3CEnetsparker\(0x000023\)%3C/scRipt%3E](https://temu.com/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3E%3CEnetsparker(0x000023)%3C/scRipt%3E)

Method	Parameter	Value
GET	nsextt	'"@--></style></scRipt><scRipt>netsparker(0x000023)</scRipt>

Proof URL

[https://temu.com/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3E%3Calert\(0x000023\)%3C/scRipt%3E](https://temu.com/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3E%3Calert(0x000023)%3C/scRipt%3E)

Certainty



Request

GET /.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x000023)%3C/scRipt%3E HTTP/1.1
Host: temu.com
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
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) : 1867.8429 Total Bytes Received : 10195 Body Length : 9851 Is Compressed : No

HTTP/1.1 200 OK
Set-Cookie: api_uid=CnBKqWYqDacSFACSAwXXAg==; expires=Fri, 25-Apr-25 08:00:39 GMT; domain=temu.com; path=/; secure
Server: nginx
Connection: keep-alive
Content-Encoding:
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
cip: 112.134.157.103
Date: Thu, 25 Apr 2024 08:00:39 GMT
Vary: Accept-Encoding

```
<html><body><script type="text/javascript"></script><script>var _0x43f1=['fwUcS','keys','reload','wHLJB','WoDem','setRequestHeader','gFDVx','message','rep','onreadystatechange','error_msg','ZFfOY','swSm  
y','WCvPA','Sz1Ge','userAgent','xIfpS','fromCharCode','tcd9797685cfc5f50dd7475d74d63c42a7','vfxMb','met  
hod','/.well-known/?nsextt='%22@--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0x000023)%3C/scRip  
t%3E','MLWUt','slice','OIXif','HsxCT','application/json','mPRLv','GRWYj','NdwJC','uSCgy','JPgvy','ImGr  
d','SNcrq','gEOGy','zEAbc','apply','c-jc','gCKRy','QwXya','lkz1V','DOWuQ','stringify','ZA1Rj','onSucces  
s','open','bfx1e','ZOGTb','send','yYqeu','zBMaP','rhGoG','atob','jKraI','ur1','aAyDE','eGdPO','subtl  
e','catch','key','i/s','then','status','LlCXk','QalwX','charCodeAt','splHr','ikYWdyld806wTnou7GZZF07BwM  
Jic7+M/AcbrhKXYwVpD80G6aodYp104NHJhxUsgMj7yh9VksAhnLyg4U11EvvmEsp72f66UFRB06wzbEvC6C7UaoByyhEmRsTTEvU0  
4b28cmP13JkgxXtj2RZ9UwtKZWfPyuedcYBLtdCtrhowCKM1AdQZUg0FU6n57LKeA160Y9K4CquXNaarsy3N5f7jz1cnFFLwaKS6zC  
5sUuqv99cmgIGBRPf4zGJhqJ5gUHOFGVgkXJ30hA+nhON5w==','IfwYq','KnWdt','xCKPq','WVlmi','importKey','header  
s','sVytz','fUBCJ','iY0qI','iCzJz','GET','POST','data','href','split','xn1Lx','forEach','QiCRA','readyS  
tate','zWKMr','eQYat','WCFvR','map','comdA','decrypt'];(function(_0x208cd6,_0x43f1d5){var _0x41323a=fun  
ction(_0x14ded8){while(--_0x14ded8){_0x208cd6['push'](_0x208cd6['shift']());}};_0x41323a(++_0x43f1d5);}  
(_0x43f1,0xec));var _0x4132=function(_0x208cd6,_0x43f1d5){_0x208cd6=_0x208cd6-0x0;var _0x41323a=_0x43f1  
[_0x208cd6];return _0x41323a;};!function(){var _0x231244={};_0x231244['xIfpS']=function(_0x191b5d,_0x1a  
b40f){return _0x191b5d;}}
```

Date: April 26, 2024

Summary of the day's activities:

We were working on our cybersecurity project and spent the day analyzing Identification and Authentication Failures issue. I looked deep into this vulnerability, attempted to comprehend its shades and undertones, effects as well as avoidance strategies.

Vulnerabilities detected or explored:

I didn't identify any particular vulnerabilities today but spent a lot of time investigating Identification and Authentication Failures. Also, I played around with tools like Hydra or John the Ripper which can be utilized for exploiting situations. Most sites I visited required an email address plus password combo for logging in.

Challenges Faced and How They Were Overcome:

An excessive amount of information was available which made it impossible to identify helpful items amidst all that noise; so many resources seemed dubious at best. Nevertheless, my critical thinking kept me going by permitting a selective approach towards what appeared to be worth analyzing critically from only reliable sources while ignoring others outright.

While trying to test authentication procedure got locked out by one server so highlighting ethical hacking etiquette besides keeping to system restrictions; next time round will guarantee my testing are responsible within ethical confines.

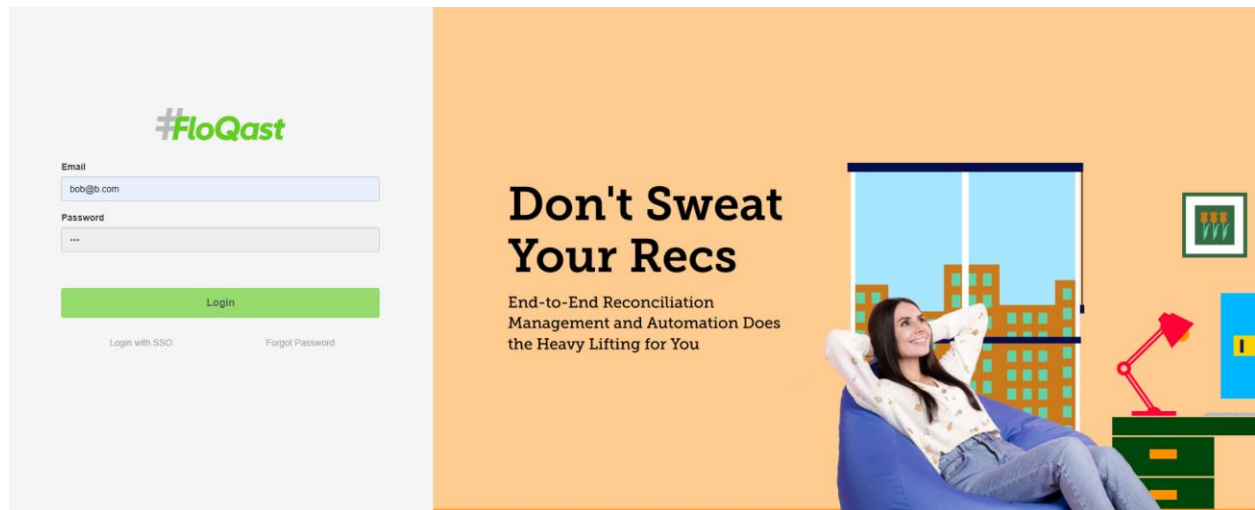
New tools, skills, or concepts learned:

During the course of today, i have been introduced to two password cracking tools dubbed Hydra and John the Ripper. Besides that, I was taught how ID and authentication failures can be avoided in a system with multi-factor authentication (MFA), having strong passwords and safe coding techniques.

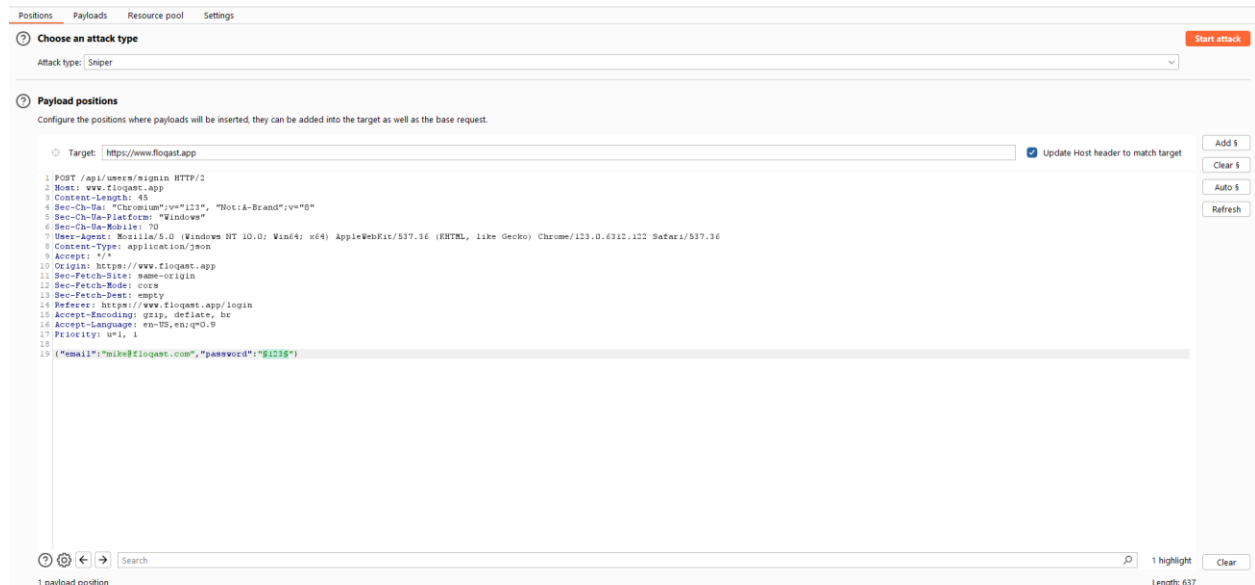
Reflections and takeaways:

I am delighted that i got to know about Identification as well as Authentication failures more deeply than before. No matter if there were no flaws detected during the hands-on session, this knowledge will always stay relevant because it highlights why we should

utilize solid security methods around our credentials. It only made me realize once again how crucial cyber security is for keeping digital assets safe when working on them or storing them somewhere online; also constant development through knowledge upgrading in such rapidly changing environment cannot be overemphasized too much. Now I would like to utilize these new skills wherever feasible so that they can help develop robustness into diverse parts of security within computer networks.



I found this site on hackerone platform. Try to bypass its login page.



Attack Save ⓘ ⓘ

▼ Filter: Showing all items

Request	Response
1	2
3	4
5	6
7	8
9	10
11	12
13	14
15	16
17	18
19	20
21	22
23	24
25	26
27	28
29	30
31	32
33	34
35	36
37	38
39	40
41	42
43	44
45	46
47	48
49	50
51	52
53	54
55	56
57	58
59	60
61	62
63	64
65	66
67	68
69	70
71	72
73	74
75	76
77	78
79	80
81	82
83	84
85	86
87	88
89	90
91	92
93	94
95	96
97	98
99	100

0 highlight

2. Includer attack on https://www.floqast.app

Attack

Save

Filter

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
23	mustang	423	324			3436	
24	1234567890	423	318			3436	
25	michael	423	336			3436	
26	654321	423	316			3436	
27	superman	423	320			3436	
28	1qaz2wsx	423	321			3436	
29	7777777	423	322			3436	
30	121212	423	320			3436	
31	000000	423	350			3436	

Request Response

Pretty

Raw

Hex

Render



```

1 HTTP/2 423 Locked
2 Date: Thu, 25 Apr 2024 16:42:01 GMT
3 Content-Security-Policy: script-src 'self' blob: https://js.pusher.com https://stats.pusher.com
https://static.floqast.app https://static.floqast.com https://services.floqast.app
https://resource-maps.floqast.app https://fq-production-internal-ip-restricted.s3-us-west-2.amazonaws.com
https://super-assets.floqast.app https://*.churnzero.net *.aptrinsic.com https://browser.sentry-cdn.com
https://js.sentry-cdn.com https://*.sentry.io; style-src 'self' 'unsafe-inline' https://fonts.googleapis.com
http://static.floqast.app https://static.floqast.com https://services.floqast.app
https://fq-production-internal-ip-restricted.s3-us-west-2.amazonaws.com https://super-assets.floqast.app
https://*.churnzero.net https://fonts.gstatic.com *.aptrinsic.com; img-src 'self' data: https://s3.amazonaws.com
https://s3-us-west-2.amazonaws.com https://static.floqast.app https://services.floqast.app
https://static.floqast.com https://fq-production-internal-ip-restricted.s3-us-west-2.amazonaws.com
https://super-assets.floqast.app https://*.churnzero.net *.aptrinsic.com
https://avatars-production.floqast.engineering https://avatars.floqast.app https://storage.googleapis.com;
connect-src 'self' wss://ws.pusherapp.com wss://ws-eu.pusher.com wss://ws-mtl.pusher.com https://api.floqast.app
https://api.floqast.com https://fq-production-txm.s3.us-west-2.amazonaws.com
https://fq-production-txm.s3-accelerate.amazonaws.com https://*.churnzero.net https://*.floqast.app
https://www.floqast.app/ https://stitch.mongodb.com https://us-west-2.aws.stitch.mongodb.com
https://fq-production-amortization-uploaded-items.s3.us-west-2.amazonaws.com
https://fq-production-amortization-export-rec.s3.us-west-2.amazonaws.com
https://production-large-payload-store.s3.us-west-2.amazonaws.com
https://fq-production-collaborate-dirty-bucket.s3.us-west-2.amazonaws.com
https://production-serverless-document-request.s3.us-west-2.amazonaws.com
https://fq-production-application-temporary-exports.s3.us-west-2.amazonaws.com *.aptrinsic.com sentry.io
*.sentry.io https://floqademy.floqast.com https://test-floqademy.skilljar.com https://px-esp.floqast.app; font-src
'self' data: https://fonts.gstatic.com https://static.floqast.app https://static.floqast.com
https://fq-production-internal-ip-restricted.s3-us-west-2.amazonaws.com https://super-assets.floqast.app
https://fonts.googleapis.com https://*.churnzero.net; object-src 'none'; media-src 'self' https://*.churnzero.net;
frame-src 'self' https://*.churnzero.net/ https://www.youtube.com/ https://www.youtube-nocookie.com/
https://drive.google.com; frame-ancestors 'self'; worker-src 'self' blob:; child-src 'self' blob:
https://*.churnzero.net https://www.youtube.com http://www.youtube.com https://player.vimeo.com
http://player.vimeo.com https://play.vidyard.com http://play.vidyard.com
4 Referrer-Policy: no-referrer-when-downgrade
5 X-Permitted-Cross-Domain-Policies: none
6 X-Xss-Protection: 1; mode=block

```

I was locked by the server

```
danuka@kali:~$ hydra
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in
military or secret service organizations, or for illegal purposes (this is n
on-binding, these ** ignore laws and ethics anyway).

Syntax: hydra [(f-l LOGIN)-l FILE] [-p PASS-P FILE] | [-c FILE] [-e nsp] [
-o FILE] [-t TASKS] [-m FILE [-T TASKS]] [-w TIME] [-W TIME] [-f] [-s PORT] [
-x MIN:MAX:CHARSET] [-c TIME] [-ISOUvVd46] [-m MODULE_OPT] [service://server[
:PORT][[/OPT]]]

Options:
-l LOGIN or -l FILE login with LOGIN name, or load several logins from FIL
E
-p PASS or -p FILE try password PASS, or load several passwords from FILE
-c FILE colon separated "login:pass" format, instead of -l/-p options
-m FILE list of servers to attack, one entry per line, ':' to specify por
t
-t TASKS run TASKS number of connects in parallel per target (default: 16)
-u service module usage details
-m OPT options specific for a module, see -U output for information
-h more command line options (COMPLETE HELP)
server the target: DNS, IP or 192.168.0.0/24 (this OR the -M option)
service the service to crack (see below for supported protocols)
OPT some service modules support additional input (-U for module help
)

Supported services: adms500 asterisk cisco cisco-enable cobaltstrike cvs fir
ebird ftp[s] http[s]--[head|get|post] http[s]--[get|post]-form http-proxy http-
proxy-urlenum icq imap[s] irc ldap2[s] ldap3[-(cram|digest|md5)][s] memcached
mongodb mssql mysql nntp oracle-listener oracle-sid pcanywhere pcnfs pop3[s]
postgres radmin2 rdp redis rexec riogin rpcap rsh rftp s7-300 sip smb smtp[s]
smtp-enum snmp socks5 ssh sshkey svn teamspeak telnet[s] vmauthd vnc xmp

Hydra is a tool to guess/crack valid login/password pairs.
Licensed under AGPL v3.0. The newest version is always available at:
https://github.com/vanhauser-thc/thc-hydra
Please don't use in military or secret service organizations, or for illegal
purposes. (This is a wish and non-binding - most such people do not care about
laws and ethics anyway - and tell themselves they are one of the good ones.)

Example: hydra -l user -P passlist.txt ftp://192.168.0.1

danuka@kali:~$
```

Hydra tool



ProductsServicesPublicationsResourcesWhat's new

John the Ripper password cracker

John the Ripper is an Open Source password security auditing and password recovery tool available for many operating systems. **John the Ripper jumbo** supports hundreds of hash and cipher types, including for: user passwords of Unix flavors (Linux, BSD, Solaris, AIX, QNX, etc.), macOS, Windows, "web apps" (e.g., WordPress), groupware (e.g., Notes/Domino), and database servers (SQL, LDAP etc.); network traffic captures (Windows network authentication, WPA/WPA-PSK, etc.); encrypted private keys (SSH, GnuPG, cryptocurrency wallets, etc.); filesystems and disks (macOS .dmg files and "sparse bundles", Windows BitLocker, etc.); archives (ZIP, RAR, 7z), and document files (PDF, Microsoft Office's, etc.). These are just some of the examples - there are many more.

[Follow @Openwall on Twitter for new release announcements and other news](#)

John the Ripper is free and Open Source software, distributed primarily in source code form. If you would rather use a commercial product, please consider [John the Ripper Pro](#), which is distributed primarily in the form of "native" packages for the target operating systems and in general is meant to be easier to install and use while delivering optimal performance.

Proceed to [John the Ripper Pro](#) homepage for your OS:

- John the Ripper Pro for Linux
- John the Ripper Pro for macOS
- On Windows, consider Hash Suite (developed by a contributor to John the Ripper)
- On Android, consider Hash Suite Droid

Download the latest John the Ripper jumbo release ([release notes](#)) or development snapshot:

- 1.9.0-jumbo-1 sources in [tar.xz](#), 33 MB (signature) or [tar.gz](#), 43 MB (signature)
- 1.9.0-jumbo-1 64-bit Windows binaries in [7z](#), 22 MB (signature) or [zip](#), 63 MB (signature)
- 1.9.0-jumbo-1 32-bit Windows binaries in [7z](#), 21 MB (signature) or [zip](#), 61 MB (signature)
- Development source code in [GitHub repository](#) (download as [tar.gz](#) or [zip](#))

Run John the Ripper jumbo in the cloud (AWS):


- [John the Ripper in the cloud](#) homepage

Download the latest John the Ripper core release ([release notes](#)):

- 1.9.0 core sources in [tar.xz](#), 8.6 MB (signature) or [tar.gz](#), 13 MB (signature)
- Development source code in [CVS repository](#)

To verify authenticity and integrity of your John the Ripper downloads, please use our [GnuPG public key](#). Please refer to these pages on [how to extract John the Ripper source code from the tar.gz and tar.xz archives](#) and [how to build \(compile\) John the Ripper core](#) (for jumbo, please refer to instructions inside the archive). You can also consider the [unofficial builds](#) on the contributed resources list further down this page.

These and **older versions** of John the Ripper, patches, unofficial builds, and many other related files are also [available from the Openwall file archive](#).



Get John the Ripper apparel at O-Day Clothing and support the project



























John the ripper

Date: April 27, 2024

Summary of the day's activities: The focus of today was on finding old software versions on the Kindred Group website. I ran tests with Netsparker automatic scanners that have helped me to point out vulnerabilities.

Vulnerabilities discovered or explored: Moment.js and jQuery were both found to be outdated during the scanning process. These old versions can create security risks since there are known vulnerabilities which can be exploited by attackers.

Vulnerability Summary

CONFIRM	VULNERABILITY	METHOD	URL	PARAMETER
	 Out-of-date Version (Moment.js)	GET	https://www.kindredgroup.com/	
	 Out-of-date Version (jQuery)	GET	https://www.kindredgroup.com/	
	 [Possible] Phishing by Navigating Browser Tabs	GET	https://www.kindredgroup.com/	
	 Missing Content-Type Header	GET	https://www.kindredgroup.com/globalassets/images/	
	 Missing X-Frame-Options Header	GET	https://www.kindredgroup.com/	
	 Cookie Not Marked as HttpOnly	GET	https://www.kindredgroup.com/	
	 Cookie Not Marked as Secure	GET	https://www.kindredgroup.com/	
	 Insecure Frame (External)	GET	https://www.kindredgroup.com/	
	 Internal Server Error	GET	https://www.kindredgroup.com/globalassets/images/	
	 Content Security Policy (CSP) Not Implemented	GET	https://www.kindredgroup.com/	
	 Expect-CT Not Enabled	GET	https://www.kindredgroup.com/	
	 Missing X-XSS-Protection Header	GET	https://www.kindredgroup.com/	
	 Referrer-Policy Not Implemented	GET	https://www.kindredgroup.com/	

1. Out-of-date Version (Moment.js)

HIGH  1

Netsparker identified that the target web site is using Moment.js and detected that it is out of date.

Impact

Since this is an old version of the software, it may be vulnerable to attacks.

Moment.js Other Vulnerability

moment is a JavaScript date library for parsing, validating, manipulating, and formatting dates. Affected versions of moment were found to use an inefficient parsing algorithm. Specifically using string-to-date parsing in moment (more specifically rfc2822 parsing, which is tried by default) has quadratic (N^2) complexity on specific inputs. Users may notice a noticeable slowdown is observed with inputs above 10k characters. Users who pass user-provided strings without sanity length checks to moment constructor are vulnerable to (Re)DoS attacks. The problem is patched in 2.29.4, the patch can be applied to all affected versions with minimal tweaking. Users are advised to upgrade. Users unable to upgrade should consider limiting date lengths accepted from user input.

Affected Versions

2.18.0 to 2.29.3

External References

- [CVE-2022-31129](#)

Moment.js Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') Vulnerability

Moment.js is a JavaScript date library for parsing, validating, manipulating, and formatting dates. A path traversal vulnerability impacts npm (server) users of Moment.js between versions 1.0.1 and 2.29.1, especially if a user-provided locale string is directly used to switch moment locale. This problem is patched in 2.29.2, and the patch can be applied to all affected versions. As a workaround, sanitize the user-provided locale name before passing it to Moment.js.

Affected Versions

1.0.1 to 2.29.1

External References

- [CVE-2022-24785](#)

Vulnerabilities

1.1. <https://www.kindredgroup.com/>

Identified Version

- 2.24.0

Latest Version

- 2.30.1 (in this branch)

Request

GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Date: Sat, 27 Apr 2024 21:09:25 GMT

2. Out-of-date Version (jQuery)

MEDIUM 1

Netsparker identified the target web site is using jQuery and detected that it is out of date.

Impact

Since this is an old version of the software, it may be vulnerable to attacks.

jQuery Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Vulnerability

In jQuery versions greater than or equal to 1.0.3 and before 3.5.0, passing HTML containing `<option>` elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. `.html()`, `.append()`, and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.

Affected Versions

1.9.0 to 3.4.1

External References

- [CVE-2020-11023](#)

jQuery Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Vulnerability

In jQuery versions greater than or equal to 1.2 and before 3.5.0, passing HTML from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. `.html()`, `.append()`, and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.

Affected Versions

1.9.0 to 3.4.1

External References

- [CVE-2020-11022](#)

jQuery Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Vulnerability

Cross Site Scripting vulnerability in jQuery 2.2.0 through 3.x before 3.5.0 allows a remote attacker to execute arbitrary code via the `<options>` element.

Affected Versions

2.2.0 to 3.4.1

External References

- [CVE-2020-23064](#)

Request

```
GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

```
HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Date: Sat, 27 Apr 2024 21:09:25 GMT
```

Challenges faced and how they were overcome: One problem I encountered was how long it took for scans to complete. At first, I did not realize just how much time is needed in order for a scan to be carried out properly. Nevertheless, after making some changes on scan settings and giving priority to areas that seemed more critical than others; then this helped me simplify the procedure thus enabling quick results which can be acted upon.

Reflections and takeaways: During my exploration today, I got an opportunity of trying out Netsparker automation tools in scanning for vulnerabilities. This has shown me why we should always check if our software is up-to-date or not as well as its implications towards security. Also, it taught me how can i speed up my scans without compromising on their quality at all.

Reflections and takeaways: The finding of outdated versions on the Kindred Group website highlights the need of deadly vulnerability management. Identifying and resolving vulnerabilities swiftly is critical for maintaining powerful cybersecurity defenses. Moving forward, I will argue for regular vulnerability assessments and underscore the significance of being watchful against emerging threats. Today's experience reaffirmed the dynamic nature of cybersecurity and the constant need for adaptation and development in security policies.

Date: April 28, 2024

Summary of the day's activities: Today was devoted to penetrating Insecure design vulnerabilities with a main focus on the Kindred site a system noted on the HackerOne platform. The goal was to determine and also analyze possible protection weak points coming from hazardous layout techniques.

Vulnerabilities discovered or explored:

During the study of the Kindred internet site a number of dangerous style susceptibilities were discovered. These consisted of:

Insecure iframe: The visibility of unconfident iframes within the site's material allowed possible shot strikes and also cross-origin information leakages.

Missing out on Content-Type header: The lack of the Content-Type header in particular HTTP reactions elevated problems concerning the proper handling and also analysis of information by client-side elements.

Missing Out On X-Frame-Options header: The absence of X-Frame-Options header left the site prone to clickjacking strikes possibly allowing harmful stars to overlay misleading web content on the site.

6. Insecure Frame (External)

LOW 

1

CONFIRMED 

1

Netsparker identified an external insecure or misconfigured iframe.

Impact

IFrame sandboxing enables a set of additional restrictions for the content within a frame in order to restrict its potentially malicious code from causing harm to the web page that embeds it.

The Same Origin Policy (SOP) will prevent JavaScript code from one origin from accessing properties and functions - as well as HTTP responses - of different origins. The access is only allowed if the protocol, port and also the domain match exactly.

Here is an example, the URLs below all belong to the same origin as *http://site.com*:

http://site.com
http://site.com/
http://site.com/my/page.html

Whereas the URLs mentioned below aren't from the same origin as *http://site.com*:

http://www.site.com (a sub domain)
http://site.org (different top level domain)
https://site.com (different protocol)
http://site.com:8080 (different port)

When the `sandbox` attribute is set, the `iframe` content is treated as being from a unique origin, even if its hostname, port and protocol match exactly. Additionally, sandboxed content is re-hosted in the browser with the following restrictions:

- Any kind of plugin, such as ActiveX, Flash, or Silverlight will be disabled for the `iframe`.
- Forms are disabled. The hosted content is not allowed to make forms post back to any target.
- Scripts are disabled. JavaScript is disabled and will not execute.
- Links to other browsing contexts are disabled. An anchor tag targeting different browser levels will not execute.
- Unique origin treatment. All content is treated under a unique origin. The content is not able to traverse the DOM or read cookie information.

When the `sandbox` attribute is not set or not configured correctly, your application might be at risk.

A compromised website that is loaded in such an insecure `iframe` might affect the parent web application. These are just a few examples of how such an insecure frame might affect its parent:

- It might trick the user into supplying a username and password to the site loaded inside the `iframe`.
- It might navigate the parent window to a phishing page.
- It might execute untrusted code.
- It could show a popup, appearing to come from the parent site.

Vulnerabilities

6.1. <https://www.kindredgroup.com/>

CONFIRMED

Frame Source(s)

- https://www.youtube.com/embed/3hcGibYHmjM?version=3&loop=1&mute=1&autoplay=1&showinfo=0&rel=0&playsinline=0&modestbranding=1&controls=1&cc_load_policy=0
- <https://www.youtube.com/embed/KklWksJKRT4?mute=1&autoplay=1&showinfo=0&rel=0&playsinline=0&modestbranding=1>

Parsing Source

- DOM Parser

Request

```
GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

```
HTTP/1.1 200 OK
Set-Cookie: EpiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Date: Sat, 27 Apr 2024 21:09:25 GMT
```

```

<!DOCTYPE html>
<html lang="en" class="no-js" data-categories="Regulatory">
<head>
<meta charset="utf-8">

<title>Kindred Group plc &#x2013; We continue to transform gambling</title>

<script>document.documentElement.classList.remove("no-js")</script>

<script type="text/javascript">
!function (T, l, y) { var S = T.location, k = "script", D = "instrumentationKey", C = "ingestionendpoint", I = "disableExceptionTracking", E = "ai.device.", b = "toLowerCase", w = "crossOrigin", N = "POST", e = "appInsightsSDK", t = y.name || "appInsights"; (y.name || T[e]) && (T[e] = t); var n = T[t] || function (d) { var g = !1, f = !1, m = { initialize: !0, queue: [], sv: "5", version: 2, config: d }; function v(e, t) { var n = {}, a = "Browser"; return n[E + "id"] = a[b](), n[E + "type"] = a, n["ai.operation.name"] = S && S.pathname || "_unknown_", n["ai.internal.sdkVersion"] = "javascript:snippet_" + (m.sv || m.version), { time: function () { var e = new Date; function t(e) { var t = "" + e; return 1 === t.length && (t = "0" + t), t } return e.getUTCFullYear() + "-" + t(1 + e.getUTCMonth()) + "-" + t(e.getUTCDate()) + "T" + t(e.getUTCHours()) + ":" + t(e.getUTCMinutes()) + ":" + t(e.getUTCSeconds()) + "." + ((e.getUTCMilliseconds() / 1e3).toFixed
...

```

Remedy

- Apply sandboxing in inline frame

```
<iframe sandbox src="framed-page-url"></iframe>
```

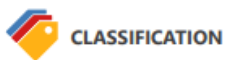
- For untrusted content, avoid the usage of seamless attribute and allow-top-navigation, allow-popups and allow-scripts in sandbox attribute.

External References

- [HTML5 Security Cheat Sheet](#)

Remedy References

- [How to Safeguard your Site with HTML5 Sandbox](#)
- [Play safely in sandboxed IFrames](#)



CLASSIFICATION

OWASP 2017	A6
SANS Top 25	16
WASC	15
ISO27001	A.14.1.2

8. Missing Content-Type Header

LOW  1

Netsparker detected a missing Content-Type header which means that this website could be at risk of a MIME-sniffing attack.

Impact

MIME type sniffing is a standard functionality in browsers to find an appropriate way to render data where the HTTP headers sent by the server are either inconclusive or missing.

This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the intended content type.

The problem arises once a website allows users to upload content which is then published on the web server. If an attacker can carry out XSS (Cross-site Scripting) attack by manipulating the content in a way to be accepted by the web application and rendered as HTML by the browser, it is possible to inject code in e.g. an image file and make the victim execute it by viewing the image.

Vulnerabilities

8.1. <https://www.kindredgroup.com/globalassets/images/>

Certainty



Request

```
GET /globalassets/images/ HTTP/1.1
Host: www.kindredgroup.com
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: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114; EpiStateMarker=true; ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114; ShowMenu=true; cookie-consent-date=1714252212312; cookie-consent-settings={%22ad_storage%22:%22granted%22%2C%22analytics_storage%22:%22granted%22%2C%22functionality_storage%22:%22granted%22%2C%22personalization_storage%22:%22granted%22%2C%22security_storage%22:%22granted%22}
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) : 566.9974 Total Bytes Received : 194 Body Length : 0 Is Compressed : No

```
HTTP/1.1 500 Internal Server Error
CF-Cache-Status: DYNAMIC
CF-RAY: 87b1cd4ebecd513a-CMB
Server: cloudflare
Connection: keep-alive
Content-Length: 0
Date: Sat, 27 Apr 2024 21:10:54 GMT
```

Remedy

1. When serving resources, make sure you send the content-type header to appropriately match the type of the resource being served. For example, if you are serving an HTML page, you should send the HTTP header:

```
Content-Type: text/html
```

2. Add the X-Content-Type-Options header with a value of "nosniff" to inform the browser to trust what the site has sent is the appropriate content-type, and to not attempt "sniffing" the real content-type.

```
X-Content-Type-Options: nosniff
```

External References

- [MIME Sniffing: feature or vulnerability?](#)
- [X-Content-Type-Options HTTP Header](#)

9. Missing X-Frame-Options Header

LOW  1

Netsparker detected a missing X-Frame-Options header which means that this website could be at risk of a clickjacking attack.

The X-Frame-Options HTTP header field indicates a policy that specifies whether the browser should render the transmitted resource within a frame or an iframe. Servers can declare this policy in the header of their HTTP responses to prevent clickjacking attacks, which ensures that their content is not embedded into other pages or frames.

Impact

Clickjacking is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on a framed page when they were intending to click on the top level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to other another page, most likely owned by another application, domain, or both.

Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker.

Vulnerabilities

9.1. <https://www.kindredgroup.com/>

Certainty



Request

```
GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

```
HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Date: Sat, 27 Apr 2024 21:09:25 GMT
```

```
<!DOCTYPE html>
<html lang="en" class="no-js" data-categories="Regulatory">
<head>
<meta charset="utf-8">

<title>Kindred Group plc &#x2013; We continue to transform gambling</title>

<script>document.documentElement.classList.remove("no-js")</script>

<script type="text/javascript">
!function (T, l, y) { var S = T.location, k = "script", D = "instrumentationKey", C = "ingestionendpoint", I = "disableExceptionTracking", E = "ai.device.", b = "toLowerCase", w = "crossOrigin", N = "POST", e = "appInsightsSDK", t = y.name || "appInsights"; (y.name || T[e]) && (T[e] = t); var n = T[t] || function (d) { var g = !1, f = !1, m = { initialize: !0, queue: [], sv: "5", version: 2, config: d }; function v(e, t) { var n = {}, a = "Browser"; return n[E + "id"] = a[b](), n[E + "type"] = a, n["ai.operation.name"] = S && S.pathname || "_unknown_", n["ai.internal.sdkVersion"] = "javascript:snippet_" + (m.sv || m.version), { time: function () { var e = new Date; function t(e) { var t = "" + e; return 1 === t.length && (t = "0" + t), t } return e.getUTCFullYear() + "-" + t(1 + e.getUTCMonth()) + "-" + t(e.getUTCDate()) + "T" + t(e.getUTCHours()) + ":" + t(e.getUTCMinutes()) + ":" + t(e.getUTCSeconds()) + "." + ((e.getUTCMilliseconds() / 1e3).toFixed
```

Remedy


- Sending the proper X-Frame-Options in HTTP response headers that instruct the browser to not allow framing from other domains.
 - X-Frame-Options: DENYIt completely denies to be loaded in frame/iframe.
 - X-Frame-Options: SAMEORIGINIt allows only if the site which wants to load has a same origin.
 - X-Frame-Options: ALLOW-FROM URLIt grants a specific URL to load itself in a iframe. However please pay attention to that, not all browsers support this.
- Employing defensive code in the UI to ensure that the current frame is the most top level window.

External References

- [Clickjacking](#)
- [Can I Use X-Frame-Options](#)
- [X-Frame-Options HTTP Header](#)

Remedy References

- [Clickjacking Defense Cheat Sheet](#)

 CLASSIFICATION	
OWASP 2013	A5
OWASP 2017	A6
SANS Top 25	693
CAPEC	103
ISO27001	A.14.2.5

Challenges faced and how they were overcome:

One considerable difficulty faced throughout the evaluation was the recognition of the missing out on safety headers.

In spite of the lack of specific signs in the first HTTP reactions thorough evaluation along with screening making use of internet browser programmer devices as well as network interception proxies aided find the vulnerabilities.

In addition, efficiently connecting the extent along with effects of the risky layout searching for to customers positioned a problem calling for clear as well as short explanation of the dangers included.

New tools, skills, or concepts learned:

The evaluation of dangerous style vulnerabilities gave important understandings right into the relevance of aggressive security steps in software program growth. In addition, the process presented me to different devices coupled with strategies for determining as well as reducing unconfident layout problems consisting of web browser designer devices, network interception proxies as well as protection header scanning energies.

Reflections and takeaways:

Today's expedition of unsafe layout vulnerabilities underscored the vital function of safe style concepts in mitigating cybersecurity risks. The study of vulnerabilities within the Kindred web site highlighted the extensive nature of dangerous layout methods plus the demand for increased recognition as well as persistence in software program growth procedures. Progressing, I am devoted to pushing for safe layout methods and also advertising a society of security-conscious advancement to shield versus possible dangers as well as vulnerabilities.

Date: April 29, 2024

Overview of the day's events:

Today was committed to analyzing vulnerabilities associated to unconfident TLS method variations with a major focus on the Temu internet site, recognized with the HackerOne system. The aim was to assess the internet site's TLS setup along with discover probable weak points that can show delicate information to interception or control.

Vulnerabilities Identified or Explored:

During the examination of the Temu web site, vulnerabilities linked with unconfident TLS technique variants were found. These vulnerabilities comprised of:

Out-of-date TLS procedure variations: The internet site's TLS setup indicated assistance for out-of-date as well as unconfident method variations such as SSL 2.0, SSL 3.0 as well as TLS 1.0 which endure different cryptographic strikes as well as vulnerabilities.

Absence of Perfect Forward Secrecy (PFS): The lack of Perfect Forward Secrecy in the TLS setup can perhaps reveal session secrets to concession in case of a personal trick concession jeopardizing the privacy of encrypted interactions.

4. Insecure Transportation Security Protocol Supported (TLS 1.0)

LOW 

1

CONFIRMED 

1

Netsparker detected that insecure transportation security protocol (TLS 1.0) is supported by your web server.

TLS 1.0 has several flaws. An attacker can cause connection failures and they can trigger the use of TLS 1.0 to exploit vulnerabilities like BEAST (Browser Exploit Against SSL/TLS).

Websites using TLS 1.0 are considered non-compliant by PCI since 30 June 2018.

Impact

Attackers can perform man-in-the-middle attacks and observe the encryption traffic between your website and its visitors.

Vulnerabilities

4.1. <https://temu.com/>

CONFIRMED

Request

[NETSPARKER] SSL Connection

Response

Response Time (ms) : 1 Total Bytes Received : 27 Body Length : 0 Is Compressed : No

[NETSPARKER] SSL Connection

Actions to Take

We recommended to disable TLS 1.0 and replace it with TLS 1.2 or higher. See Remedy section for more details.

Remedy

Configure your web server to disallow using weak ciphers. You need to restart the web server to enable changes.

- For Apache, adjust the SSLProtocol directive provided by the mod_ssl module. This directive can be set either at the server level or in a virtual host configuration.

```
SSLProtocol +TLSv1.2
```

23 / 49

- For Nginx, locate any use of the directive ssl_protocols in the nginx.conf file and remove TLSv1.

```
ssl_protocols TLSv1.2;
```

- For Microsoft IIS, you should make some changes on the system registry. **Incorrectly editing the registry may severely damage your system. Before making changes to the registry, you should back up any valued data on your computer.**
 1. Click on Start and then Run, type regedt32 or regedit, and then click OK.
 2. In Registry Editor, locate the following registry key or create if it does not exist:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\
```

3. Locate a key named Server or create if it doesn't exist.
 4. Under the Server key, locate a DWORD value named Enabled or create if it doesn't exist and set its value to "0".
- For lighttpd, put the following lines in your configuration file:

```
ssl.use-ssl2 = "disable"  
ssl.use-ssl3 = "disable"  
ssl.openssl.ssl-conf-cmd = ("Protocol" => "-TLSv1.1, -TLSv1, -SSLv3") # v1.4.48 or up  
ssl.ec-curve = "secp384r1"
```

New tools, skills, or concepts learned:

The examination of unconfident TLS process changes offered me to different equipment as well as ways for examining TLS setups plus determining vulnerabilities. This consisted of TLS scanning devices such as SSL Labs and also Qualys SSL Server Test which supplied extensive evaluations of TLS setups as well as recognized places needing renovation. Additionally, notions such as Perfect Forward Secrecy (PFS) as well as cipher collection hardness analysis were studied out, boosting my grasp of safe TLS release approaches.

Reflections and Takeaways:

Today's exploration of unconfident TLS procedure modifications emphasized the important necessity of persistent TLS setups in protecting the private combined with honesty of on the internet interactions.

The research of vulnerabilities within the Temu site demonstrated the extended nature of unconfident TLS arrangements and the necessity for aggressive measures to lessen linked threats.

Progressing I focused to promoting for the development of current TLS procedures plus safe and secure cipher collections to enhance the safety and security position of internet applications plus secure versus probable hazards as well as vulnerabilities.

Challenges Faced and How They Were Overcome:

One major hurdle ran into throughout the analysis was the recognition of prone TLS process alterations and cipher collections.

In spite of the dearth of precise indications, the first network encounters thorough assessment including screening making use of TLS scanning devices along with cryptographic evaluation energies aided locate the vulnerabilities.

In addition, communicating the seriousness coupled with impacts of unconfident TLS setups to investors positioned a difficulty, demanding clear as well as brief expression of the risks entailed.

Date: April 30, 2024

Overview of the day's events: Today, the main focus was on carrying out a comprehensive vulnerability assessment, paying attention in particular to detection and analysis of phishing-related vulnerabilities that are exploited via browser tab manipulation, and the main aim was to scrutinize the security status of the website www.kindredgroup.com with respect to identifying possible weak spots that can make users vulnerable to phishing.

Vulnerabilities Identified or Explored: During the assessment, I found the following phishing vulnerabilities associated with opening different browser tabs. The user interface design such that the application would manipulate the browser tabs necessitating probable phishing attacks that involve presenting deceptive content of some kind. Inappropriately handling browser windows/tabs by not closing them properly raises the chances whereby a user may unknowingly view harmful data and/or even disclose personal information. Poor cross-origin communication capabilities can make phishing attacks easy by enabling the app to interact with content from untrusted source.

3. [Possible] Phishing by Navigating Browser Tabs

LOW  1

Netsparker identified possible phishing by navigating browser tabs but was unable to confirm the vulnerability.

Open windows with normal hrefs with the tag `target="_blank"` can modify `window.opener.location` and replace the parent webpage with something else, even on a different origin.

Impact

While this vulnerability doesn't allow script execution, it does allow phishing attacks that silently replace the parent tab. If the links lack `rel="noopener noreferrer"` attribute, a third party site can change the URL of the source tab using `window.opener.location.assign` and trick the users into thinking that they're still in a trusted page and lead them to enter their sensitive data on the malicious website.

Vulnerabilities

3.1. <https://www.kindredgroup.com/>

External Links

- <https://www.facebook.com/KindredGroup/>
- <https://www.instagram.com/kindredgroup/>
- <https://www.linkedin.com/company/kindred-group-plc/>
- <https://twitter.com/KindredGroup>
- <https://www.youtube.com/c/kindredgroup-plc>
- <https://www.facebook.com/KindredGroup/>
- <https://www.instagram.com/kindredgroup/>
- <https://www.linkedin.com/company/kindred-group-plc/>
- <https://twitter.com/KindredGroup>
- <https://www.youtube.com/c/kindredgroup-plc>

Certainty

Request

GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
...
den-2-image-16-9.jpg?width=1000&format=jpg&quality=90">
<meta property="og:locale" content="en">
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="https://twitter.com/KindredGroup">
<meta name="twitter:title" content="Kindred Group plc – We continue to transform gambling">
<meta name="twitter:description" content="Kindred Group has brought together leading and
...

</div>
<div class="social-bar">
<div class="social-bar__item"></div>
<div class="social-bar__item"></div>
<div class="social-bar__item"></div>
<div class="social-bar__item"></div>
<div class="social-bar__item"></div>
</div>

```

<a href="https://www.youtube.com/c/kindredgroup-plc" class="pod-logo-background">
<div class="full-youtube-logo">

</div>
</a>
<a href="https://open.spotify.c
...
</span>
<div class="podcast-cotent-footer">
<div>
<h3>Subscribe to new episodes</h3>
<div class="pod-logo-group">
<a href="https://www.youtube.com/c/kindredgroup-plc" class="pod-logo-background">
<div class="full-youtube-logo">

</div>
</a>
<a href="https://open.spotify.c
...
</div>
<div class="footer-bottom-left">
<div class="footer__social">
<div class="social-bar">
<div class="social-bar__item"><a href="https://www.facebook.com/KindredGroup/" target="_blank"><img src
="/assets/icon-facebook.svg" alt="facebook" /></a></div>
<div class="social-bar__item"><a href="https://www.instagram.com/kindredgroup/" target="_blank"><img sr
c="/assets/icon-instagram.svg" alt="instagram" /></a></div>
<div class="social-bar__item"><a href="https://www.linkedin.com/company/kindred-group-plc/" target="_bl
ank"></a></div>
<div class="social-bar__item"><a href="https://twitter.com/KindredGroup" target="_blank"></a></div>
<div class="social-bar__item"><a href="https://www.youtube.com/c/kindredgroup-plc" target="_blank"></a></div>
</div>
</div>
</div>
</div>

</footer>      </main>
</div>

```

Remedy

- Add `rel=noopener` to the link to prevent pages from abusing `window.opener`. This ensures that the page cannot access the `window.opener` property in Chrome and Opera browsers.

15 / 59

- For older browsers and in Firefox, you can add `rel=noreferrer` which additionally disables the Referer header.

```
<a href="..." target="_blank" rel="noopener noreferrer">...</a>
```

External References

- [Reverse Tabnabbing](#)
- [Blankshield & Reverse Tabnabbing Attacks](#)
- [Target=" blank" - the most underestimated vulnerability ever](#)



CLASSIFICATION

OWASP 2013	A5
OWASP 2017	A6
SANS Top 25	16
WASC	15
ISO27001	A.14.1.2

New tools, skills, or concepts learned:

The assessment given today provided very valuable information about the trickery in browser tabs that makes way for phishing attacks. This made me know about different ways one can copy or reproduce realistic phishing situations on web apps which have made me better at vulnerability assessment methods. Moreover, I came to understand why it is necessary that user-generated inputs be carefully authenticated and cleaned so as to prevent browser tab manipulation attacks.

Challenges Faced and How They Were Overcome:

Phishing attacks simulation and replication as part of the assessment presented a central challenge. To overcome it, scrupulous analysis of the application's user interface design and the cross-origin communication capabilities were done to pinpoint possible attack vectors. In addition, communicating the seriousness and consequences of the vulnerabilities to those with a stake in them involved condensing the risks which needed to be taken into account and the things that ought to be done.

Reflections and Takeaways:

Robust security measures remain critical in today's activities in order to mitigate phishing attacks on web applications. The discovery of vulnerabilities on www.kindredgroup.com has demonstrated how pervasive security risks can be when user interfaces are poorly designed and cross-origin communication practices are not followed. In the future I am dedicated to supporting strict security measures like Content Security Policy (CSP) and educating users on phishing prevention in order for web applications to be more secure and for users to avoid possible threats and vulnerabilities.

Date: May 1, 2024

Overview of the day's events:

The aim of the day was to thoroughly assess for how weak ciphers enabled by www temu.com on its communication networks may expose it to vulnerabilities. Its primary goal was detecting any cryptanalysis cracks which would compromise privacy or message authenticity.

Vulnerabilities Identified or Explored:

I have identified the following vulnerabilities with weak ciphers: Weak cryptographic ciphers such as RC4 or DES among others supported by the application's SSL/TLS configurations may allow outdated versions of SSL/TLS protocol (e.g. RC4) to be used on this server/network. Additionally, the usage of weak ciphers in networking communication protocols (such as HTTPS) tends to make it easier for attackers to recover original plaintext through a process called cryptanalysis upon intercepted data while connecting with the Internet."

3. Weak Ciphers Enabled

MEDIUM



1

CONFIRMED



1

Netsparker detected that weak ciphers are enabled during secure communication (SSL).

You should allow only strong ciphers on your web server to protect secure communication with your visitors.

Impact

Attackers might decrypt SSL traffic between your server and your visitors.

Vulnerabilities

3.1. https://temu.com/

CONFIRMED

List of Supported Weak Ciphers

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
- TLS_RSA_WITH_AES_128_CBC_SHA (0x002F)
- TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
- TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (0x0088)
- TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (0x0045)
- TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (0x0084)
- TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (0x0041)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xC013)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xC014)
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0x0067)
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x006B)
- TLS_RSA_WITH_AES_128_CBC_SHA256 (0x003C)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (0x003D)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xC027)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xC028)
- TLS_ECDHE_RSA_WITH_CAMELLIA_256_CBC_SHA384 (0xC077)
- TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA256 (0x00C4)
- TLS_ECDHE_RSA_WITH_CAMELLIA_128_CBC_SHA256 (0xC076)
- TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA256 (0x00BE)
- TLS_RSA_WITH_CAMELLIA_256_CBC_SHA256 (0x00C0)
- TLS_RSA_WITH_CAMELLIA_128_CBC_SHA256 (0x00BA)

Request

[NETSPARKER] SSL Connection

Response

Response Time (ms) : 1 Total Bytes Received : 27 Body Length : 0 Is Compressed : No

[NETSPARKER] SSL Connection

Actions to Take

1. For Apache, you should modify the SSLCipherSuite directive in the httpd.conf.

```
SSLCipherSuite HIGH:MEDIUM:!MD5:!RC4
```

2. Lighttpd:

```
ssl.honor-cipher-order = "enable"  
ssl.cipher-list = "EECDH+AESGCM:EDH+AESGCM"
```

3. For Microsoft IIS, you should make some changes to the system registry. **Incorrectly editing the registry may severely damage your system. Before making changes to the registry, you should back up any valued data on your computer.**

a. Click Start, click Run, type regedt32 or type regedit, and then click OK.

b. In Registry Editor, locate the following registry key: HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders

c. Set "Enabled" DWORD to "0x0" for the following registry keys:

```
SCHANNEL\Ciphers\DES 56/56  
SCHANNEL\Ciphers\RC4 64/128  
SCHANNEL\Ciphers\RC4 40/128  
SCHANNEL\Ciphers\RC2 56/128  
SCHANNEL\Ciphers\RC2 40/128  
SCHANNEL\Ciphers\NULL  
SCHANNEL\Hashes\MD5
```

Challenges Faced and How They Were Overcome:

I faced a big problem during assessment when we tried to know and evaluate fragile cryptographic ciphers used in SSL/TLS settings. The best way out for this problem was scanning through tools for automation about security deeply or using human judgement pay attention to every detail when needed.

To validate the impact of weak ciphers on data confidentiality and integrity, another way has been for us to simulate cryptographic attacks like brute-force or known-plaintexts attacks so as to assess the susceptibility transmitted data.

New tools, skills, or concepts learned:

The assessment had some continuance that was vital in the number of weaknesses in communication networks associated with poor ciphers and how they can be protected against these flaws. It introduced me to different tools and methods for evaluating SSL/TLS settings and for identifying weak cryptographic ciphers, which made me better at vulnerability assessment methodologies. Besides, I came to understand how to ensure communication protocols are secure at all times which entails implementing Perfect Forward Secrecy (PFS) as well as ensuring that secure cipher suites take precedence during SSL/TLS negotiation.

Reflections and Takeaways:

The activities that took place today have highlighted how it is really important to make sure that we use strong codes for the safety of our communication networks as well as when sending such information through security threats- be it viruses or other problems. The discovery of vulnerabilities within the www temu.com website has brought to light the serious dangers which result from using poor ciphers and serves as a reminder that we ought to take preventive actions that would reduce the risks. For the future, I pledge myself to advancing the use of common industry encryption algorithms and its best practices in order to improve the security stance of web applications, and also protect any sensitive data from being altered or accessed without permission.

Date: May 3, 2024

Overview of the day's events: Today i concentrated in making a very good assessment of the “Cookie Not Marked as HttpOnly” vulnerability that is present in www.Kindredgroup.com. Our main aim was to point out those cookies which had not been marked as HttpOnly and to examine what dangers (if any) exist with regards to cross-site scripting.

Vulnerabilities Identified or Explored: I discovered the following weaknesses concerning cookies without the HttpOnly attribute when conducting the analysis: – Some cookies were not marked as HttpOnly causing them to be accessed through document.cookie among other ways by client-side scripts. The absence of HttpOnly attribute exposes these cookies to stealing through XSS attacks which may result in the attacker being capable of hijacking session tokens or gaining access to private data kept within them.

4. Cookie Not Marked as HttpOnly

LOW



1

CONFIRMED



1

Netsparker identified a cookie not marked as HTTPOnly.

HTTPOnly cookies cannot be read by client-side scripts, therefore marking a cookie as HTTPOnly can provide an additional layer of protection against cross-site scripting attacks.

Impact

During a cross-site scripting attack, an attacker might easily access cookies and hijack the victim's session.

Vulnerabilities

4.1. <https://www.kindredgroup.com/>

CONFIRMED

Identified Cookie(s)

- EPiStateMarker

Cookie Source

- HTTP Header

Request

```
GET / HTTP/1.1
Host: www.kindredgroup.com
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
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) : 515.5052 Total Bytes Received : 92412 Body Length : 91711 Is Compressed : No

```
HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/
Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;SameSite=None;Secure;Domain=www.kindredgroup.com
Request-Context: appId=cid-v1:f6f0367f-9d11-44c1-bd87-a01942e8c453
CF-RAY: 87b1cb230fc9513a-CMB
Server: cloudflare
CF-Cache-Status: DYNAMIC
Connection: keep-alive
Content-Encoding:
Strict-Transport-Security: max-age=2592000
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
HTTP/1.1 200 OK
Set-Cookie: EPiStateMarker=true; path=/

Set-Cookie: ARRAffinity=f62dc48792b6e16789f38b9331562ab71aac9ec805fac06e15282a091076b114;Path=/;HttpOnly;Secure;Domain=www.kindredgroup.com
Set-Cookie: ARRAffinitySameSite=f62dc48792b6e16789f38b9331
...
```

Challenges Faced and How They Were Overcome: During the assessment, a major challenge was the identification of cookies not tagged as HttpOnly and evaluating probable XSS vulnerabilities. To overcome this challenge, I carefully analyzed security by inspecting HTTP responses and identifying cookies which do not have HttpOnly attribute in them using browser developer tools and proxy tools. Also, there was a need for adequate planning and execution in order to show how an intruder could exploit the weakness by pretending to be launching XSS attacks from within the application.

New tools, skills, or concepts learned: The importance of marking cookies as HttpOnly is to mitigate the risk of XSS attacks and session hijacking. This has educated me on different tools and approaches utilized for stalking cookies so as to discover weaknesses associated with using sessions and logging in systems. Besides, I got to know how to write codes securely to avoid XSS weaknesses such as input validation, output encoding, and cleaning up user-generated contents in an effective manner.

Reflections and Takeaways: Activities today highlighted the crucial significance of distinguishing cookies and marking them as HttpOnly in order to avoid access by JavaScript from clients. Discovering vulnerabilities in www.Kindredgroup.com has shown how vital it is to mark cookies as HttpOnly; if not then this exposes it XSS attacks leading into session hijacking.

In the future, my dedication is to promote secure coding practices and frequent security assessments so as to mitigate cookie-related security vulnerabilities and bolster general security of web applications.

Date: May 4, 2024

Overview of the day's events: The main aim today was to do a complete review about the HSTS imposing ability or policy, which is not enabled on www.cargo.indrive.com page, pertaining to safety against cyber thefts. Its top priority was to identify whether HSTS was absent then find out potential risks like those posed by SSL-stripping attacks and downgrade attacks.

Vulnerabilities Identified or Explored:

When the HTTP Strict Transport Security (HSTS) policy is not implemented, the application becomes vulnerable to SSL-stripping attacks as well as man-in-the-middle attacks. The non-existence and; hence; lack of HSTS puts into danger users who might have their information being intercepted or changed by unauthorized parties leading to loss of privacy in communications and integrity is compromised between sent information.

1. HTTP Strict Transport Security (HSTS) Policy Not Enabled

MEDIUM  1

Netsparker identified that HTTP Strict Transport Security (HSTS) policy is not enabled.

The target website is being served from not only HTTPS but also HTTP and it lacks of HSTS policy implementation.

HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure (HTTPS) connections. The HSTS Policy is communicated by the server to the user agent via a HTTP response header field named "Strict-Transport-Security". HSTS Policy specifies a period of time during which the user agent shall access the server in only secure fashion.

When a web application issues HSTS Policy to user agents, conformant user agents behave as follows:

- Automatically turn any insecure (HTTP) links referencing the web application into secure (HTTPS) links. (For instance, <http://example.com/some/page/> will be modified to <https://example.com/some/page/> before accessing the server.)
- If the security of the connection cannot be ensured (e.g. the server's TLS certificate is self-signed), user agents show an error message and do not allow the user to access the web application.

Vulnerabilities

1.1. <https://cargo.indrive.com/>

Certainty



Request

```
GET / HTTP/1.1
Host: cargo.indrive.com
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
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) : 780.6575 Total Bytes Received : 96722 Body Length : 96153 Is Compressed : No

```
HTTP/1.1 200 OK
cache-control: s-maxage=31536000, stale-while-revalidate
content-encoding:
X-Amz-Cf-Id: V7feg3jGYMvMwmsWqdKsD5WFpsZHncZFpwWk5U_856b_keCmwDKySA==
server: istio-envoy
x-powered-by: Next.js
Connection: keep-alive
x-envoy-upstream-service-time: 11
Via: 1.1 a7adf71acf6767d8f3fb252f00dfd348.cloudfront.net (CloudFront)
X-Cache: Miss from cloudfront
X-Amz-Cf-Pop: SIN2-P3
vary: Accept-Encoding
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
date: Sun, 28 Apr 2024 21:17:46 GMT
etag: "4xampkjzwn221e"
x-nextjs-cache: HIT
```

```
<!DOCTYPE html><html dir="ltr" lang="en"><head><meta charset="utf-8"/><meta http-equiv="X-UA-Compatibl
e" content="IE=edge"/><meta name="viewport" content="minimum-scale=1, initial-scale=1, width=device-wid
th, shrink-to-fit=no, user-scalable=no, viewport-fit=cover"/><title>Online Cargo Transportation Service
- cargo.inDrive</title><meta name="robots" content="index,follow"/><meta name="description" content="F
reight deliveries at your price. We verify every driver's basic info and documents in the app. Learn mo
re about inDrive transportation services!"/><meta property="og:title" content="Online Cargo Transportat
ion Service - cargo.inDrive"/><meta property="og:description" content="Freight deliveries at your pric
e. We verify every driver's basic info and documents in the app. Learn more about inDrive transportatio
n services!"/><link rel="canonical" href="https://cargo.indrive.com"/><link rel="preload" as="image" im
agesrcset="/_next/image?url=%2Fassets%2Fhero.jpg&w=640&q=100 640w, /_next/image?url=%2Fassets%2
Fhero.jpg&w=750&q=100 750w, /_next/image?url=%2Fassets%2Fhero.jpg&w=828&q=100 828w, /_n
ext/image?url=%2Fassets%2Fhero.jpg&w=1080&q=100 1080w, /_next/image?url=%2Fassets%2Fhero.jpg&w=1200&q=100 1200w, /_next/image?url=%2Fassets%2Fhero.jpg&w=1920&q=100 1920w, /_next/imag
e?url=%2Fassets%2Fhero.jpg&w=2048&q=100 2048w, /_next/image?url=%2Fassets%2Fhero.jpg&a
...

```

Remedy

Configure your webserver to redirect HTTP requests to HTTPS.

i.e. for Apache, you should have modification in the httpd.conf. For more configurations, please refer to External References section.

```
# load module
LoadModule headers_module modules/mod_headers.so

# redirect all HTTP to HTTPS (optional)
```

5 / 76

```
<VirtualHost *:80>
    ServerAlias *
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [redirect=301]
</VirtualHost>

# HTTPS-Host-Configuration
<VirtualHost *:443>
    # Use HTTP Strict Transport Security to force client to use secure connections only
    Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"

    # Further Configuration goes here
    [...]
</VirtualHost>
```

External References

- [Wikipedia - HTTP Strict Transport Security](#)
- [Configure HSTS \(HTTP Strict Transport Security\) for Apache/Nginx](#)
- [HTTP Strict Transport Security \(HSTS\) HTTP Header](#)
- [Mozilla SSL Configuration Generator](#)

Challenges Faced and How They Were Overcome:

There was one very important challenge that we faced while assessing this system – how to detect whether it doesn't have HSTS in place and how does that affect its HTTPS security. Overcoming it meant looking at HTTP response headers sent by the app to find out whether they lacked/misconfigured Strict Transport Security header (HSTS).

Careful planning and execution were essential to expose the chances of attacking SSL-stripping within the app.

New tools, skills, or concepts learned: Deploying HTTP Strict Transport Security (HSTS) to bolster the safety of web apps; this project exposed me to different methods a developer

could use assess secureness of HTTPS, find out all the bugs that may arise from SSL-strip attack as well as how to deal with downgrade attacks that occur in these scenarios. I also learned about HSTS setup recommendations that discuss things like recommended values on max-age with examples shown on how we could apply preloads directives so that they will appear on preload lists within top browsers too.

Reflections and Takeaways:

The critical importance of implementing HTTP Strict Transport Security (HSTS) to protect users from potential risks is underscored by today's activities, including SSL-stripping attacks and downgrade attacks. Without HSTS implementation at www.cargo.indrive.com, vulnerabilities were discovered on this site showing major risks while facing data loss and corruption. For the future, I am dedicated to promoting HSTS standards as well as frequent security evaluations to advance the general security stance of web applications and secure user data against possible threats and vulnerabilities.