Finding Name: Malicious Code Execution (CVE-2024-4367)

Name	Team	Role	Project	Quality Assurance	Is this a re-tested Finding?
Wahidullah Hashimi	OnTrack	Pen-Tester	AppAttack	Nicholas Krcevianc	Yes

Was this Finding Successful?
Yes

Finding Description

While testing vulnerabilities for Ontrack, I started by using OWASP ZAP and discovered that it's using a vulnerable version of a JavaScript library (3.11.174). This issue is listed in the CVE (Common Vulnerabilities and Exposures) system as CVE-2024-4367. CVE is a well-known reference point for publicly disclosed security vulnerabilities.

CVE-2024-4367 is a serious flaw in the PDF.js library, which is widely used to view PDFs in web browsers. The problem arises because the library doesn't properly handle certain font data, which opens the door for attackers to inject malicious JavaScript. This vulnerability affects Firefox versions before 126, Firefox ESR versions prior to 115.11, Thunderbird versions earlier than 115.11 and Ontrack.

If a user opens a malicious PDF crafted to exploit this flaw, JavaScript could execute, leading to some pretty severe consequences, like data theft, XSS attacks, or even remote code execution (RCE). It's crucial for anyone using these versions to update their software to avoid falling victim to these types of attacks.

This vulnerability was also discovered in T1 2024. You can find the insights of other penetration tester in AppAttack x ThothTech.pdf on page 56.

Risk Rating

Impact: High

Likelihood: Medium

Impact values				
Very Minor	**Minor**	**Significant**	Major	**Severe**
Risk that holds little to no impact. Will not cause damage and regular activity can continue.	Risk that holds minor form of impact, but not significant enough to be of threat. Can cause some damage but not enough to impede regular activity.	Risk that holds enough impact to be somewhat of a threat. Will cause damage that can impede regular activity but will be able to run normally.	Risk that holds major impact to be of threat. Will cause damage that will impede regular activity and will not be able to run normally.	Risk that holds severe impact and is a threat. Will cause critical damage that can cease activity to be run.

Likelihood				
Rare	**Unlikely**	**Moderate**	**High**	**Certain**
Event may occur and/or if it did, it happens in specific circumstances.	Event could occur occasionally and/or could happen (at some point)	Event may occur and/or happens.	Event occurs at times and/or probably happens a lot.	Event is occurring now and/or happens frequently.

Business Impact

The business impact of CVE-2024-4367 can be significant, especially for organizations that rely on PDF.js or similar libraries to handle PDF files within their applications. Here are some key areas of concern:

1. Data Breach Risks: If the vulnerability is exploited, attackers could access sensitive user data, like personal information, trade secrets, or intellectual property. This can lead to data theft and damage the company's reputation as customers lose trust.
2. Financial Losses: Exploiting the vulnerability could allow attackers to take control of systems, steal funds, or manipulate transactions. Additionally, the company may face fines for violating data protection regulations like GDPR or CCPA.
3. Reputation Damage: If a successful attack occurs, it can significantly harm the company's reputation. Customers expect their data to be secure, and a breach could result in a loss of trust and reduced business.
4. Legal and Compliance Consequences: Organizations that handle sensitive data could face legal action if they don't take adequate security measures. Failure to patch vulnerabilities may lead to penalties for non-compliance with regulations like GDPR or HIPAA.
5. Operational Disruption: Attackers exploiting the vulnerability could cause system downtime, service outages, or data corruption. This disruption affects business operations and incurs additional costs for recovery.

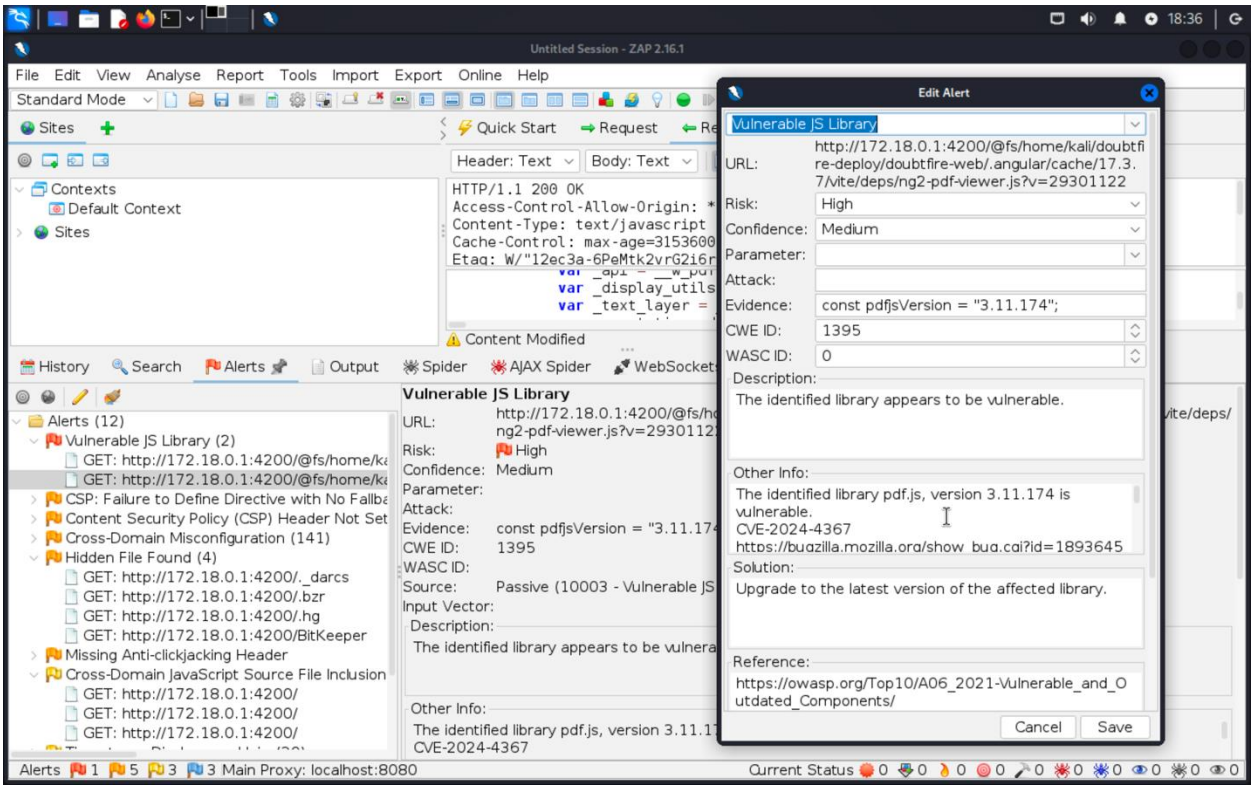
Affected Assets

- Doubtfire API
- OnTrack Frontend
- PDF.js Library
- Firefox (versions earlier than 126)

Evidence

Step 1: [Identify vulnerability]

I used OWASP ZAP to do a deep scan of the web application (Ontrack). As shown in the screenshot below, I found out that the web application uses a vulnerable JS Library.

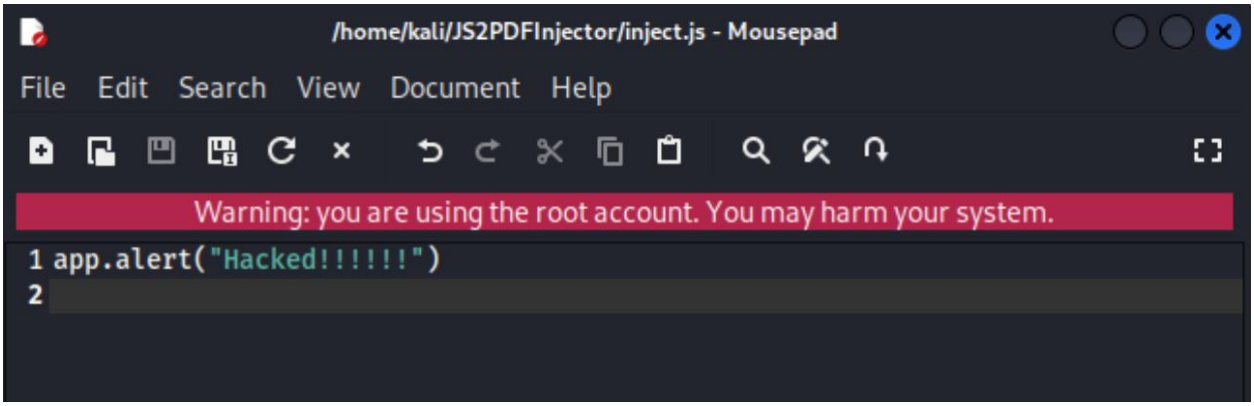


Step 2: [Research]

I did some research on how to exploit this vulnerability and found that I can embed malicious code using JS2PDFInjector tool. When this pdf file is clicked the malicious code will be executed.

Step 3: [Creating the Malicious file]

Clone JS2PDFInjector tool from github in your computer. Create a payload as shown in the screenshot below.

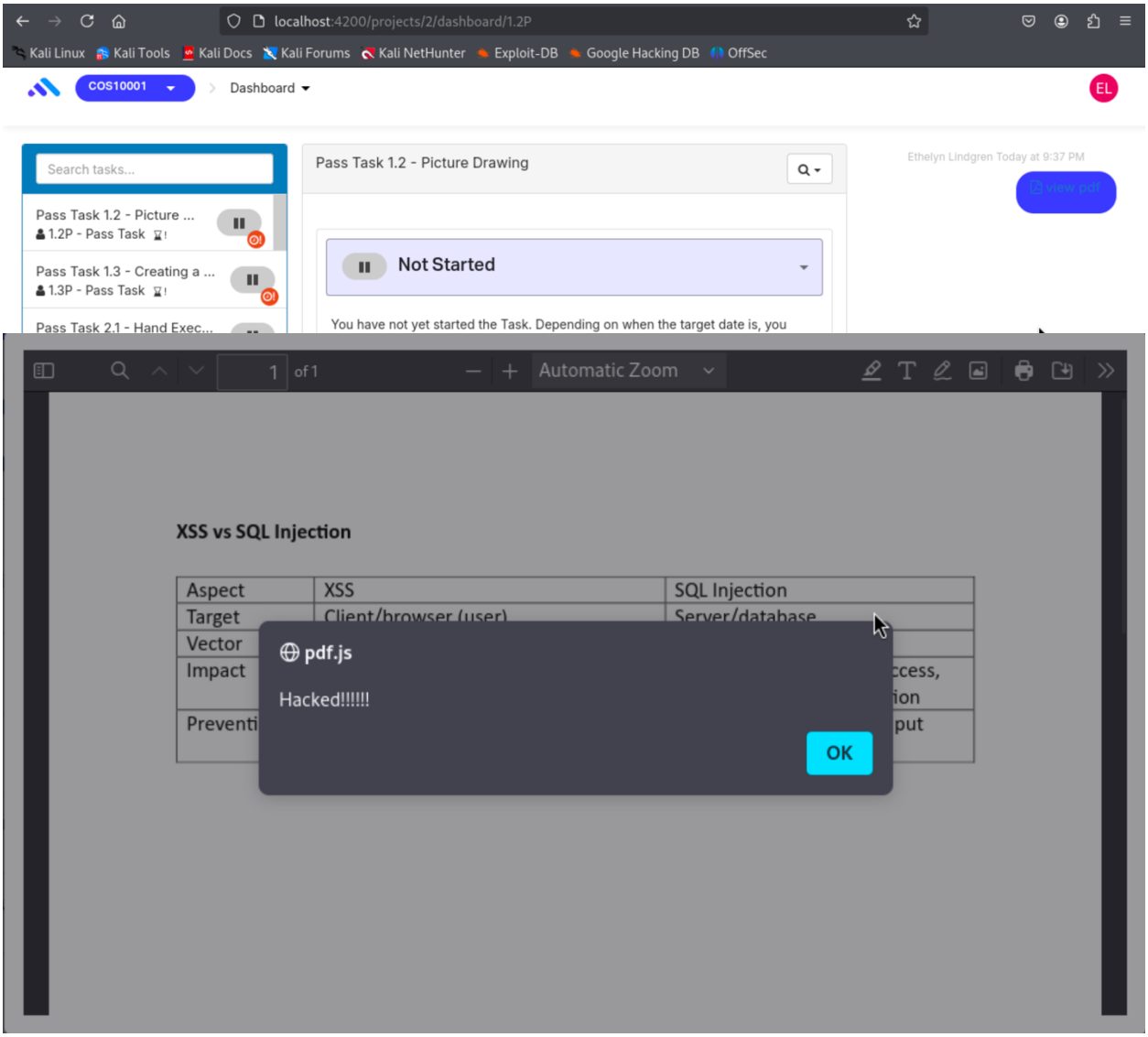


Next, embed this malicious script into a real pdf file using the command shown in the screenshot below.

```
┌──(kali㉿kali)-[~/JS2PDFInjector]
└─$ sudo java -jar JS2PDFInjector-1.0.jar /home/kali/JS2PDFInjector/sub.pdf /
home/kali/JS2PDFInjector/inject.js

[*] Original PDF: /home/kali/JS2PDFInjector/sub.pdf
[*] JavaScript Payload: /home/kali/JS2PDFInjector/inject.js
[*] Output File Path: /home/kali/JS2PDFInjector/js_injected_sub.pdf
[*] Poisoned File Created: /home/kali/JS2PDFInjector/js_injected_sub.pdf
```

Finally, you send this malicious pdf to your marker/tutor. When your marker clicks on this pdf file, the script will be executed. In the screenshot below you can see the executed pdf file.



CVE-2024-4367 Summary:
CVE-2024-4367 is a serious security flaw found in the PDF.js library, which many web browsers use to display PDF files. The issue occurs because the library doesn't properly check types when dealing with fonts, which could allow attackers to inject and run malicious JavaScript in the PDF. If exploited, this vulnerability could lead to remote code execution (RCE), data theft, or cross-site scripting (XSS) attacks through specially crafted PDF files.

Mitigation Points:

1. **Update PDF.js**: Make sure to update to the latest version of PDF.js that fixes this vulnerability.
2. **Patch Affected Browsers**: Ensure that browsers like Firefox and Thunderbird that rely on vulnerable PDF.js versions are updated to a secure version.
3. **Sanitize PDFs**: Set up processes to scan and clean PDFs for malicious JavaScript before opening them.
4. **Limit JavaScript Execution**: Configure PDF viewers to block or restrict JavaScript execution for added security.

References

OWASP TOP 10:2021, A06:2021 – Vulnerable and Outdated Components. accessible at: https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/

Cornerpirate/ JS2PDFInjector, accessible at:

https://github.com/cornerpirate/JS2PDFInjector

National Vulnerability Database (NVD) (2024). CVE-2024-4367 Detail. Accessible at: https://nvd.nist.gov/vuln/detail/CVE-2024-4367

Contact Details

Name: Wahidullah Hashimi
Email: s223397352@deakin.edu.au

Pentest Leader Feedback.

Nicholas Krcevianc:

- Next time please follow the AppAttack Findings Checklist.docx for correct formatting, this time I fixed your style and size.
- Besides that this document is perfect