Unbekannte Schwachstellen in Neos CMS

24. Mai 2022

Im Zuge eines Kundenprojekts identifizierte das it.sec Research Team mehrere unbekannte Cross-Site-Scripting-Schwachstellen (XSS) im Content Management System (CMS) Neos. Die Schwachstellen wurden in der Version 3.3.29 identifiziert und konnten in der zu dieser Zeit aktuellen Version 8.0.1 bestätigt werden. Es ist davon auszugehen, dass auch alle dazwischenliegenden Versionen betroffen sind.

Den Schwachstellen wurde die ID CVE-2022-30429 zugewiesen.

Neos stellt über folgende Links weitere Informationen zu den Schwachstellen sowie Update-Möglichkeiten bereit:

- https://discuss.neos.io/t/neos-bugfix-releases-5-3-10-7-0-9-7-1-7-7-2-6-7-3-4-8-0-2/5930
- https://www.neos.io/blog/xss-in-various-backend-modules.html

Ausnutzbarkeit und potenzieller Schaden

Die Schwachstellen könnten von einem Benutzer mit der Rolle Neos-Editor ausgenutzt werden, um andere Benutzer, wie beispielsweise Administratoren oder Besucher der bereitgestellten Webseite, anzugreifen.

XSS-Schwachstellen ermöglichen das Ausführen von eigenem JavaScript-Code im Kontext der verwundbaren Webseite, was eine Vielzahl von Angriffsmöglichkeiten eröffnet. Der potenzielle Schaden für die Anwendung selbst hängt von den Funktionalitäten der Anwendung und den Berechtigungen des angegriffenen Benutzers ab. Darüber hinaus kann das Endgerät des Benutzers angegriffen werden. Durch Änderungen der Darstellung der Webseite könnte ein Benutzer in einem Phishing-Angriff zur Preisgabe von Zugangsdaten verleitet werden.

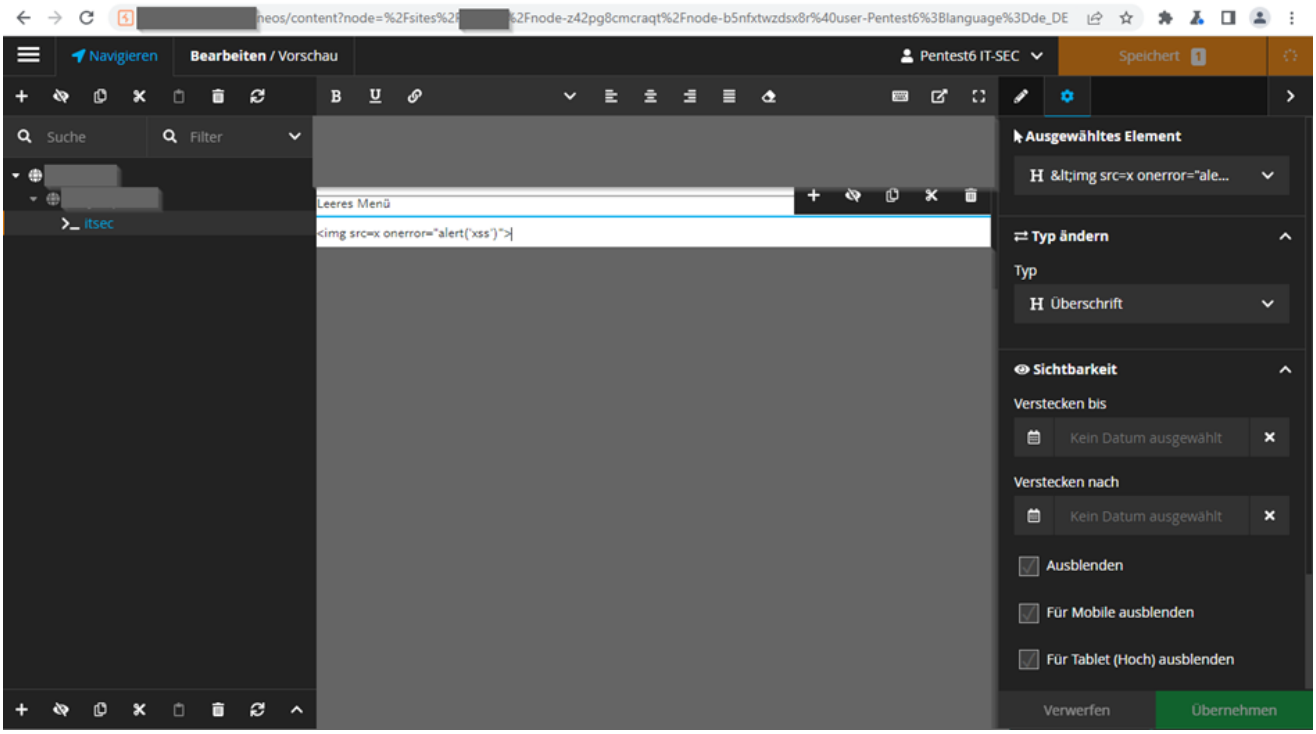
Schwachstellendetails

Folgende drei Cross-Site-Scripting-Schwachstellen werden nachfolgend erläutert:

1. Persistentes XSS in Neos-Editor
2. XSS beim Löschen von Assets
3. XSS in Arbeitsbereich-Titel

Persistentes XSS in Neos-Editor

Benutzer konnten über die Editier-Funktion eigenen JavaScript-Code in der Anwendung hinterlegen. Folgende Abbildung zeigt die Änderung des Seiteninhalts, die über die Browser-Oberfläche vorgenommen wurde.



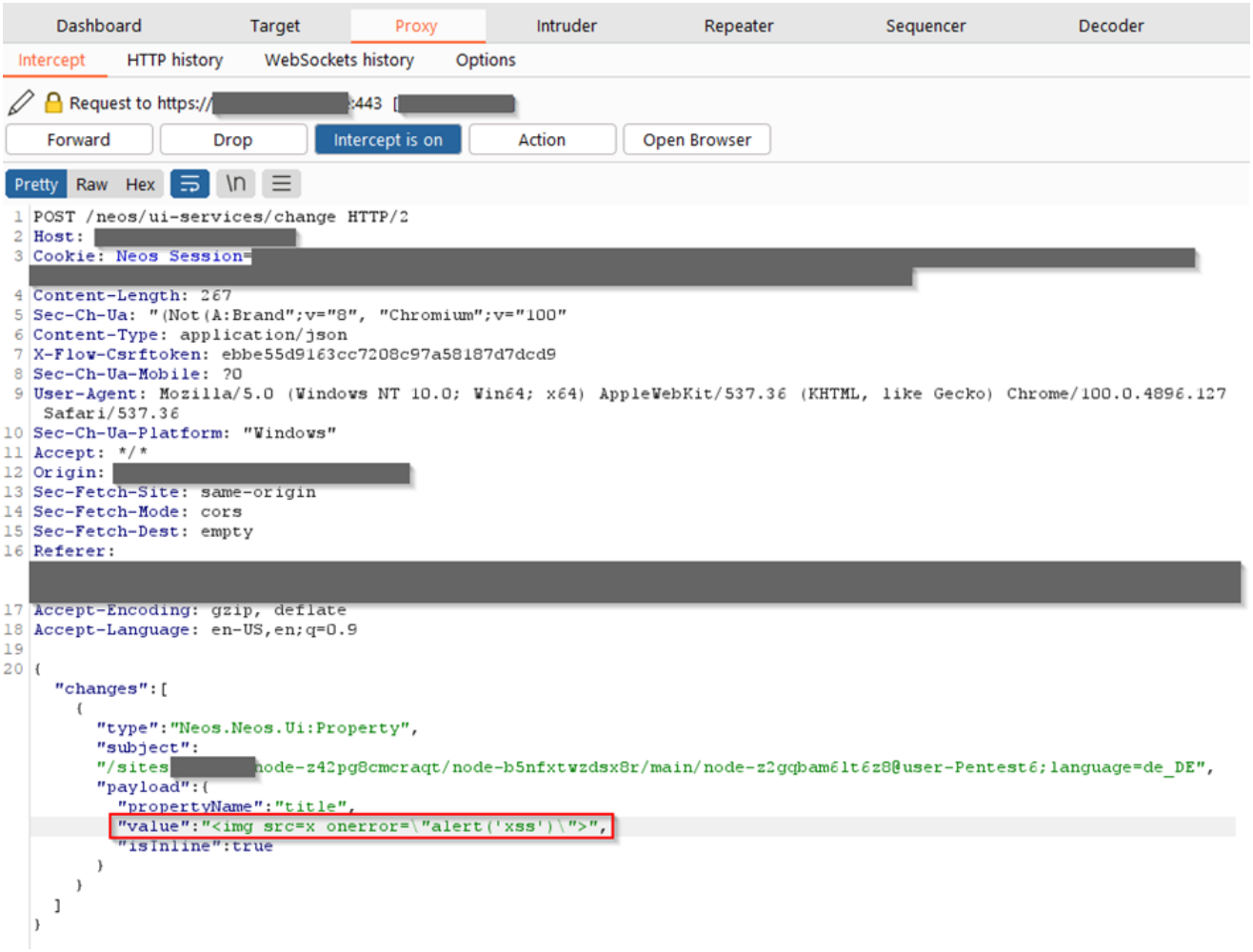
Dabei wurde eine HTTP-Anfrage ausgelöst, in welcher der JavaScript-Code

```
<img src=x onerror="alert('xss')">
```

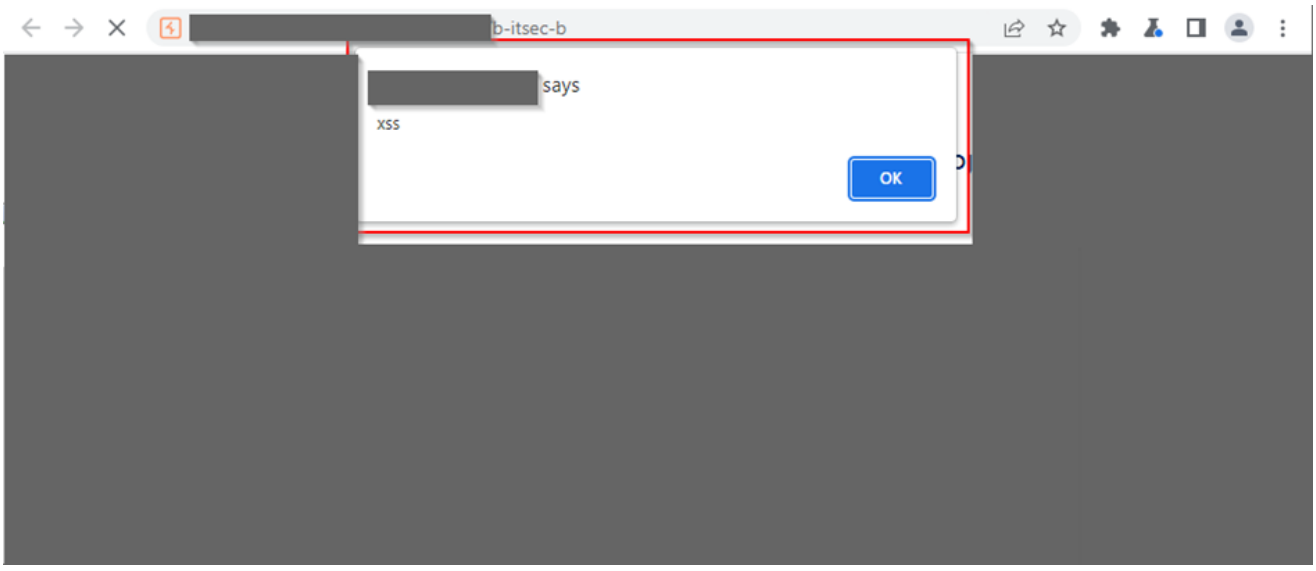
vom Browser HTML-kodiert an den Server gesendet wurde, wie nachfolgende Abbildung zeigt. Dieser Code löst beim Laden ein Alert-Pop-Up aus.

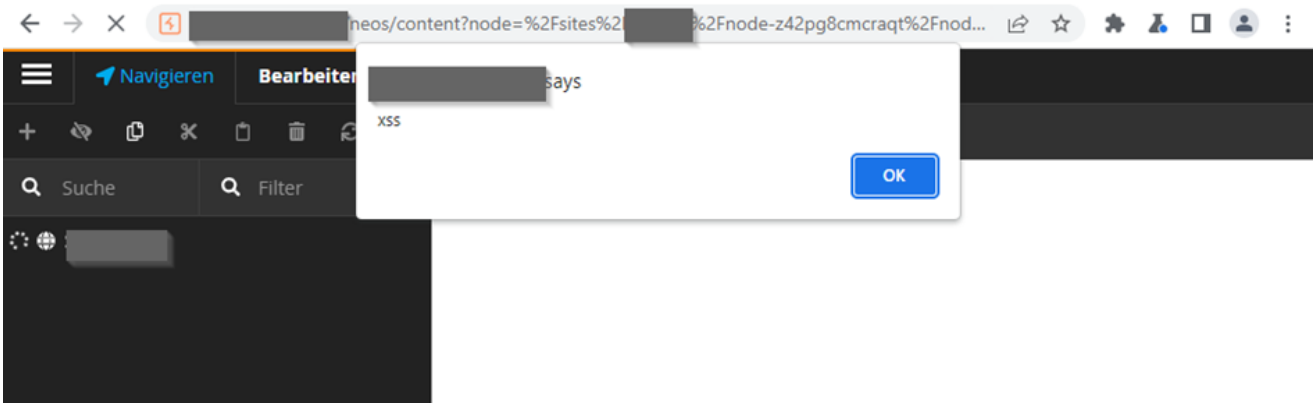
```
POST /neos/ui-services/change HTTP/2
Host: [redacted]
Cookie: Neos_Session=[redacted]
Content-Length: 267
Sec-Ch-Ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Content-Type: application/json
X-Flow-Csrftoken: ebbe55d9163cc7208c97a58187d7dcd9
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.127 Safari/537.36
Sec-Ch-Ua-Platform: "Windows"
Accept: */*
Origin: [redacted]
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: [redacted]
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

{
  "changes":[
    {
      "type":"Neos.Neos.Ui:Property",
      "subject":
      "/sites/[redacted]node-z42pg8cmcraqt/node-b5nfxtwzdsx8r/main/node-z2gqbam6lt6z8@user-Pentest
      6;language=de_DE",
      "payload":{
        "propertyName":"title",
        "value":"&lt;img src=x onerror=\"alert('xss')\"&gt;",
        "isInline":true
      }
    }
  ]
}
```

Da eine clientseitige Eingabevalidierung erfolgte, wurde die HTTP-Anfrage zum Speichern von neuem Seiteninhalt mit dem Tool Burp Suite abgefangen und editiert, bevor sie zum Server weitergeleitet wurde. Über diese Methode ist stets ein Umgehen der clientseitigen Validierung möglich. Die nächste Abbildung zeigt die HTTP-Anfrage nach dem Editieren. Die HTML-kodierten Zeichen wurden in der Anfrage durch „<“ und „>“ ersetzt.



Der hinterlegte Code wurde nach dem Veröffentlichen der Änderungen beim Besuch der erstellten Seite https://example.com/de/webseite/b-itsec-b im Browser sowie in der Neos-Oberfläche ausgeführt, wie die folgenden beiden Abbildungen zeigen.

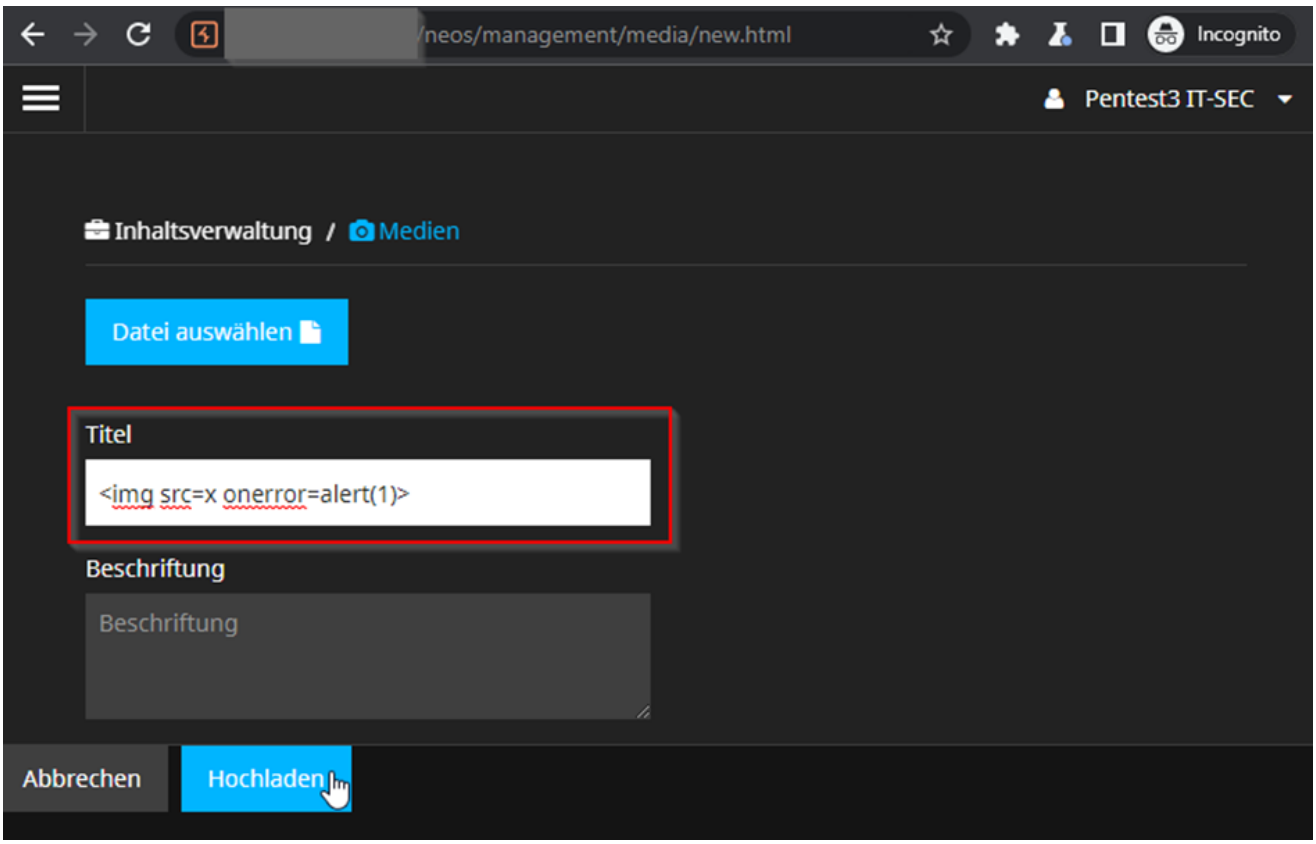




XSS beim Löschen von Assets

Über den Medien-Bereich konnten sogenannte Assets hochgeladen werden. Im Testfall wurde im Titel eines Assets der folgende Code angegeben:

```
<img src=x onerror=alert(1)>
```



Der Code wurde beim Löschen des Assets ausgeführt. Die erste der folgenden beiden Abbildungen zeigt die Oberfläche zum Löschen eines Assets, die zweite zeigt die Ausführung des Codes nach Bestätigung von „Ja, Asset löschen“.

XSS in Arbeitsbereich-Titel

Folgende Abbildung zeigt das Editieren eines zuvor erstellten Arbeitsbereich-Titels. Als Titel wurde der JavaScript-Code

```
<img src=x onerror=alert(1)>
```

verwendet.

Nach dem Klicken auf „Änderungen übernehmen“ wurde der Code, wie unten zu sehen ist, ausgeführt. Der JavaScript-Code wurde ebenso beim Löschen des Arbeitsbereichs ausgeführt.