Lp.	Podatność	Zagrożenie
1.	Możliwość spowodowania odmowy usług (DoS)	Wysokie
2.	Niewłaściwa obsługa pobierania plików	Wysokie
3.	Metadane	Średnie
4.	Odbijanie inputu użytkownika w nagłówkach HTTP	Średnie
5.	Nieaktualna wersja oprogramowania Wordpress	Średnie
6.	Odbijanie inputu użytkownika w kodzie strony	Średnie
7.	Niepoprawna konfiguracja serwera WWW	Średnie
8.	IDOR	Średnie
9.	Niepoprawnie skonfigurowana polityka Content Security Policy (CSP)	Niskie
10.	Enumeracja serwerów/frameworków – Nginx	Niskie
11.	Enumeracja serwerów/frameworków – motyw twentytwentyone	Niskie
12.	Brak bezpiecznych nagłówków HTTP	Niskie
13.	Nieaktualna wersja oprogramowania NextJS 14.1.4	Niskie
14.	Niespójne odpowiedzi serwera aplikacji	Informacyjne
15.	Niebezpieczne metody HTTP	Informacyjne
16.	Brak nagłówka Refferer-Policy	Informacyjne

Naukowa i Akademicka Sieć Komputerowa Państwowy Instytut Badawczy
ul. Kolska 12 | 01-045 Warszawa | www.nask.pl
NIP: 521-04-17-157 | Regon: 010464542 | KRS: 0000012938

1. Szczegółowe wyniki badań

Wysokie

1. Możliwość spowodowania odmowy usług (DoS)

Zagrożenie

Wysokie

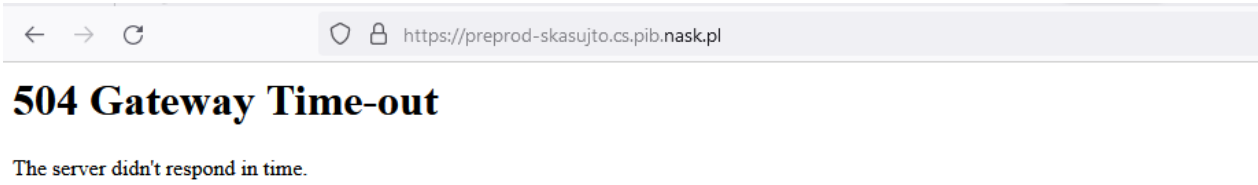
Opis

Ataki typu DoS (Denial of Service) polegają na zablokowaniu dostępu do danej usługi (np. aplikacji internetowej) bądź jej części (np. wysyłania powiadomień e-mail). Atak taki może być wykonany na wiele sposobów i może być celowany w kilka różnych punktów, np. samą aplikację bądź użytkowników aplikacji. Skutki takiej podatności różnią się w zależności od zaatakowanej usługi oraz samego rodzaju ataku, jednakże często prowadzą do wielu niepożądanych skutków biznesowych, np.: utrata klientów, straty finansowe czy straty wizerunkowe. W związku z tym, udostępniana usługa powinna być odporna na ataki typu DoS w najwyższym możliwym stopniu.

Przykład

Przy wstępnych testach obciążeniowych z targetem 1000 użytkowników podłączających się przez okres 120 sekund na endpoint https://preprod-skasujto.cs.pib.nask.pl serwis zaczął rzucać odpowiedzią 504 pochodzącą od nginx. Następnie sockets zostały kompletnie zamknięte.
Użyto narzędzia jmeter, test wykonano ok godziny 10:45 AM oraz 12:23 AM, 24.05.2024

brak dostępności usługi nastąpił w momencie próby uzyskania 159 sesji użytkownika



Rekomendacja

!!!Zależne od aplikacji.!!!

2. Niewłaściwa obsługa pobierania plików

Zagrożenie

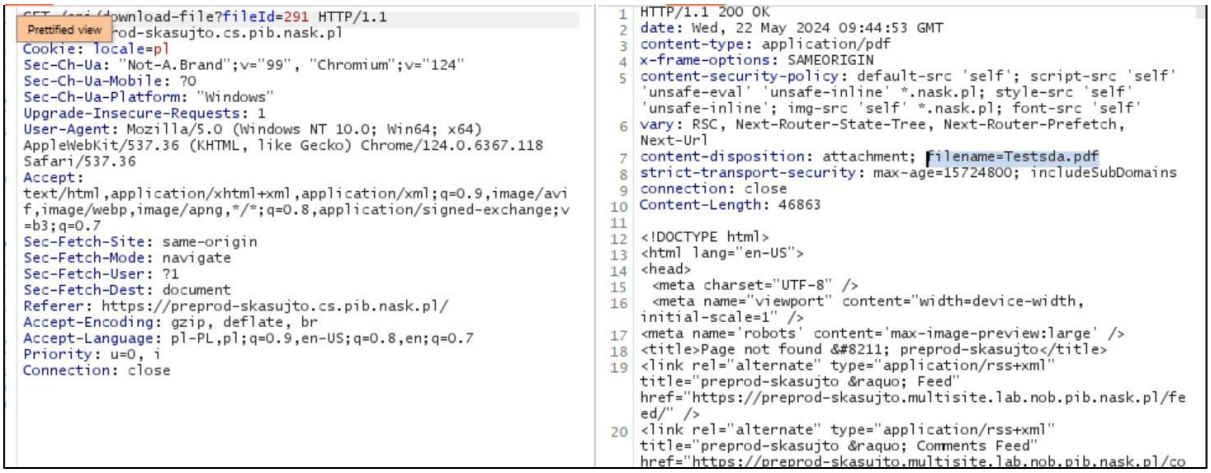
Wysokie

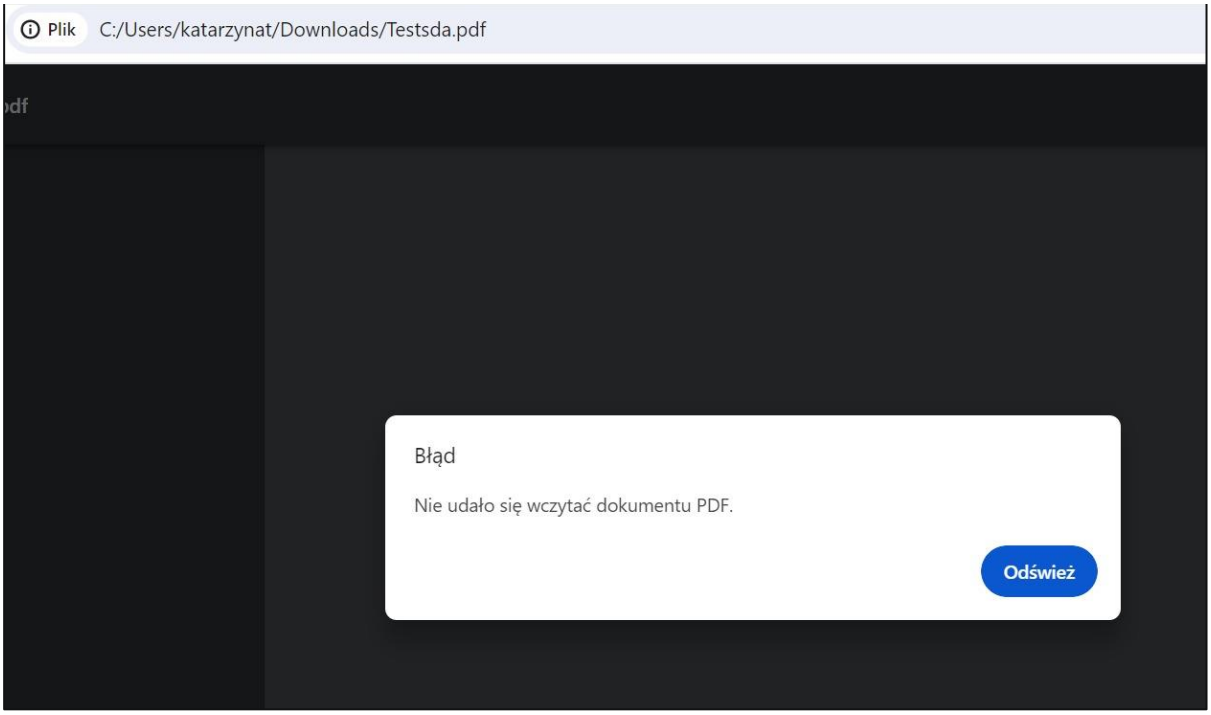
Opis

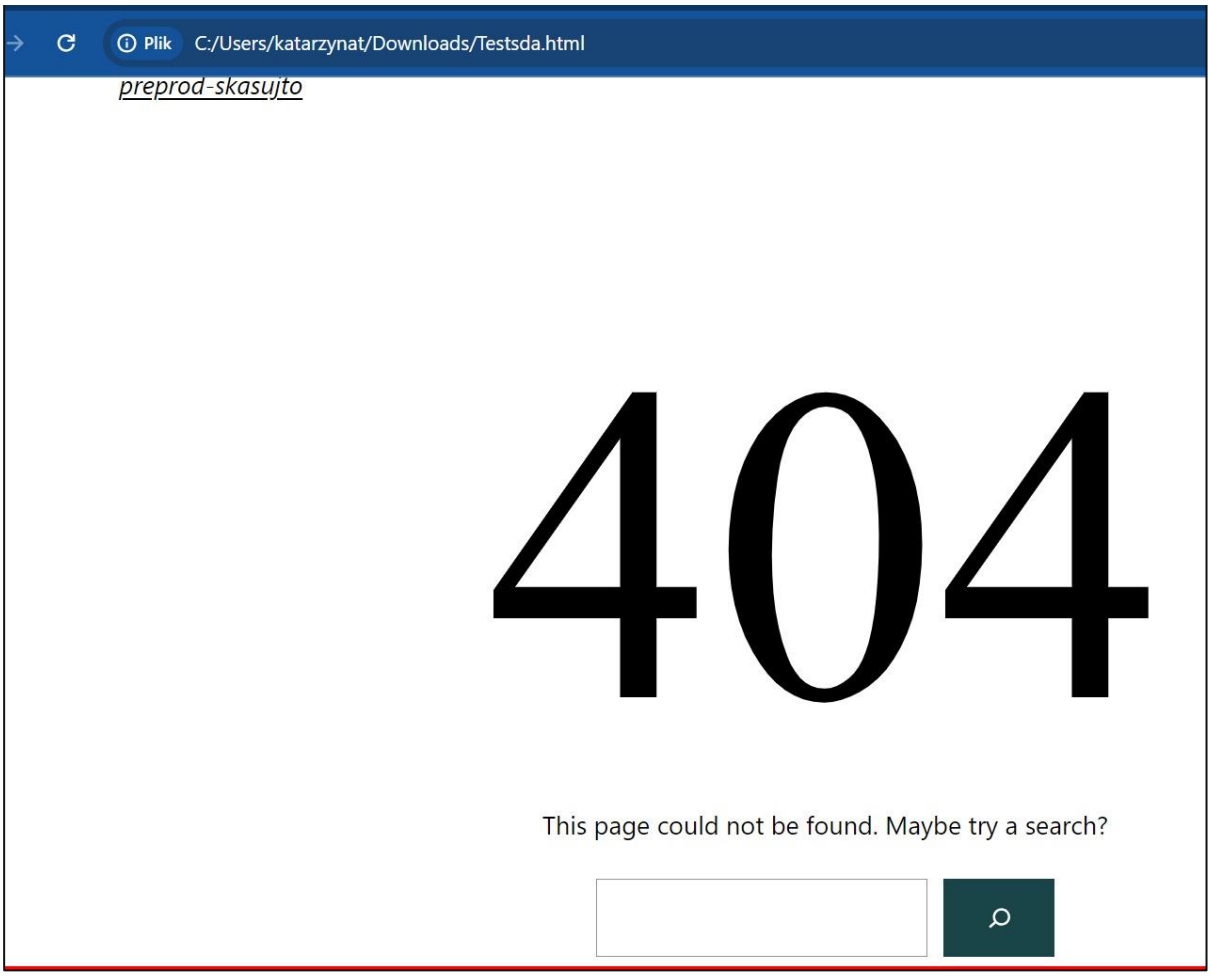
Aplikacja podczas próbuy pobrania raportu pobiera plik Testda.pdf

Przykład

https://preprod-skasujto.cs.pib.nask.pl/api/download-file?fileId=291




Plik
C:/Users/katarzynat/Downloads/Testsda.pdf
Błąd
Nie udało się wczytać dokumentu PDF.
Odśwież


Plik
C:/Users/katarzynat/Downloads/Testsda.html
preprod-skasujto
404
This page could not be found. Maybe try a search?

Rekomendacja

Średnie

3. Metadane

Zagrożenie

Średnie

Opis

Metadane to informacje wyższego poziomu zawarte w pliku, które opisują pewnego rodzaje dane (np. zdjęcie lub dokument). Przykładem metadanych są: nazwa twórcy dokumentu, nazwa sprzętu użytego do wykonania zdjęcia czy data utworzenia dokumentu. Jednakże, niektóre metadane mogą zdradzać pewne informacje wrażliwe, takie jak:

- Lokalizację GPS.
- Nazwę użytkownika systemowego.
- Ścieżki systemowe.
- Wersje użytego oprogramowania.
- Adresy e-mail.

W związku z tym, czyszczenie metadanych przed upublicznianiem danego pliku jest kluczową czynnością, która znacznie redukuje ryzyko upublicznienia wrażliwych danych.

Przykład

https://preprod-skasujto.cs.pib.nask.pl/_next/image?url=https%3A%2F%2Fpreprod-skasujto.multisite.lab.nob.pib.nask.pl%2Fwp-

content%2Fuploads%2Fsites%2F14%2F2024%2F05%2Fswgfl-strap-white-text.png&w=1920&q=75

HistorySoftwareAgent: Affinity Designer 1.8.0

HistoryWhen: 2022:03:07 10:44:27Z

ICCProfileName: sRGB IEC61966-2.1

PrimaryPlatform: Microsoft Corporation

ProfileCMMType: Little CMS

ProfileName: sRGB IEC61966-2.1

ProfileVersion: 4.3.0

Rekomendacja

Zaleca się, aby funkcjonalności ładowania nowych plików implementowały mechanizm do zarówno weryfikacji metadanych (czy znajdują się w pliku) oraz do kasowania ew. metadanych. W takim przypadku, wrażliwe metadane zawarte w ładowanym pliku zostaną automatycznie usunięte.

4. Odbijanie inputu użytkownika w nagłówkach HTTP

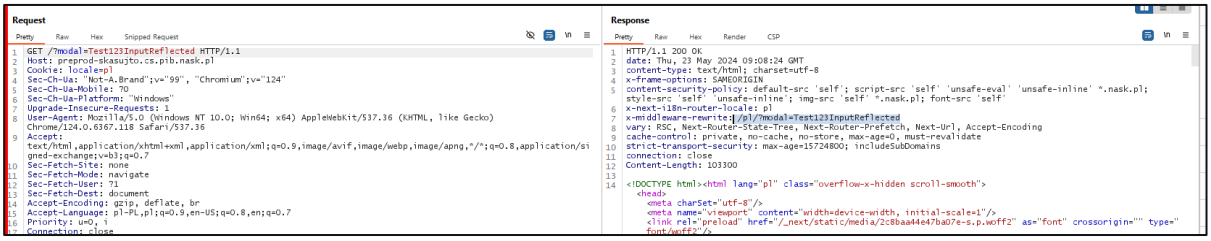
Zagrożenie

Średnie

Opis

Zwracanie wartości kontrolowanej przez użytkownika w nagłówkach HTTP odpowiedzi serwera bywa często niebezpieczna i może prowadzić do np. ataków typu HTTP response splitting. Ataki te prowadzą do sytuacji, w których do odpowiedzi serwera wstrzykiwany jest dodatkowy nagłówek HTTP, który zostanie następnie przetworzony przez przeglądarkę ofiary. Innym przykładem ataku może być np. rozluźnienie polityki SOP (Same Origin Policy) przeglądarki, poprzez dodanie własnych nagłówków CORS (Cross Origin Resource Sharing).

Przykład



Rekomendacja

5. Nieaktualna wersja oprogramowania Wordpress

Zagrożenie

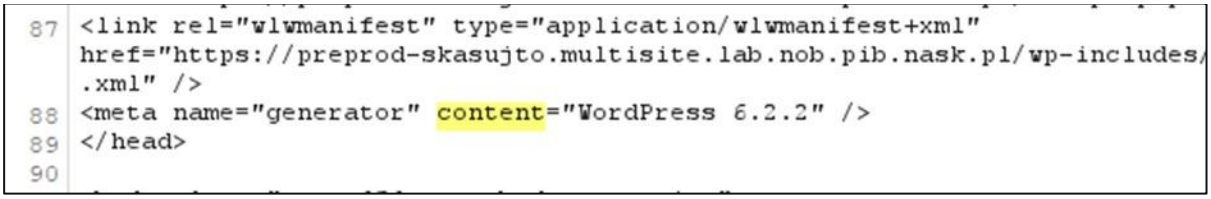
Średnie

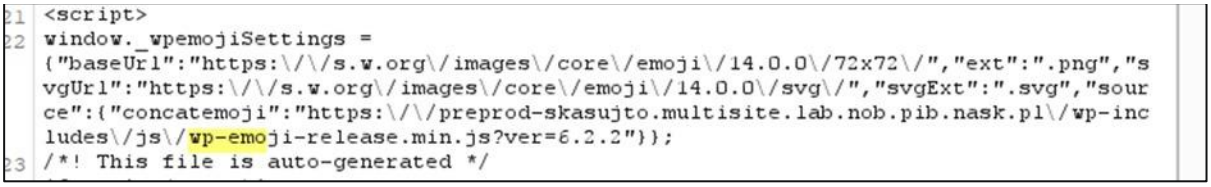
Opis

Nieaktualna wersja frameworku czy biblioteki to luka w zabezpieczeniach wynikająca z użycia przestarzałych wersji oprogramowania, które zawierają już znane błędy bezpieczeństwa. Atakujący mogą wykorzystać te luki w celu zdobycia dostępu do systemów lub przeprowadzenia innych ataków na aplikacje lub sieci (np. DoS), które wykorzystują podatne oprogramowanie.

Przykład

wp-emoji-release.min.js?ver=6.2.2





Rekomendacja

Należy zaktualizować framework do najnowszej wersji oraz upewnić się, że aktualizacje będą wprowadzane w trybie ciągłym.

6. Odbijanie inputu użytkownika w kodzie strony

Zagrożenie

Średnie

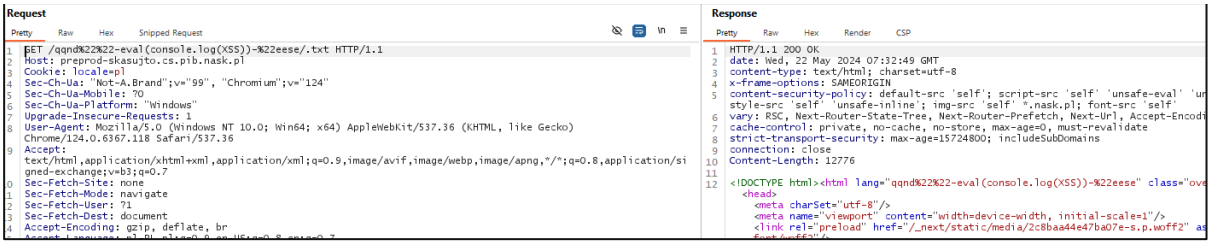
Opis

Reflected Injection polega na wstrzyknięciu kodu do źródła strony. Występuje w sytuacji, kiedy parametr wprowadzony przez użytkownika aplikacji jest prezentowany zwrotnie w treści strony, bez poprawnej weryfikacji zarówno na etapie wprowadzania (inputu) i jak i prezentacji w treści (outputu). Może to prowadzić do permanentnej modyfikacji wyglądu elementów strony lub jej zachowania. Podatność ta jest często wykorzystywana przez parametry

kontrolowane przez użytkownika przekazywane razem z URL (metody GET) czy wprowadzana za pomocą metod POST. Może to prowadzić do przejęcia sesji użytkownika, zmiany wyglądu strony czy też przekierowania go do innej, niebezpiecznej witryny.

Przykład

```
GET /?_rsc=1vdcs&vw6ix%3cscript%3ealert(1)%3c%2fscript%3eluh2b=1 HTTP/1.1
Host: preprod-skasujto.cs.pib.nask.pl
Sec-Ch-Ua: "Not-A.Brand";v="99", "Chromium";v="124"
Next-Router-State-Tree:
%5B%22%22%2C%7B%22children%22%3A%5B%5B%22locale%22%2C%22pl%22%2C%22d%22%5D%2C%7B%22
%22report%5C%22%7D%22%2C%7B%7D%5D%7D%2Cnull%2Cnull%2Ctrue%5D%7D%5D
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, li
Next-Url: /pl
Rsc: 1
Sec-Ch-Ua-Platform: "Windows"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://preprod-skasujto.cs.pib.nask.pl/?modal=report
Accept-Encoding: gzip, deflate, br
Accept-Language: pl-PL,pl;q=0.9,en-US;q=0.8,en;q=0.7
Priority: u=1, i
Connection: close
```



Rekomendacja

Aby zapobiec występowaniu podatności zalecane jest wprowadzenia skutecznej weryfikacji pobierania i wyświetlania powyższych parametrów we wszystkich miejscach, w których występują. Wszystkie niebezpieczne znaki, prezentowane w treści wynikowej powinny być "HTML enkodowane", zgodnie z poniższą listą:

- “ — "
- ‘ — '
- & — &
- < — <

7. Niepoprawna konfiguracja serwera WWW

Zagrożenie

Średnie

Opis

Serwer WWW jest ważnym elementem środowiska aplikacji webowych, a jego prawidłowa konfiguracja stanowi kluczową rolę w ich bezpieczeństwie. Błędy oraz przeoczenia popełnione w procesie konfigurowania serwera WWW mogą stanowić zagrożenie nie tylko dla bezpieczeństwa aplikacji i jej użytkowników, lecz także dla systemu operacyjnego, na którym jest ona uruchomiona. Przykładami często popełnianych błędów w tym zakresie są:

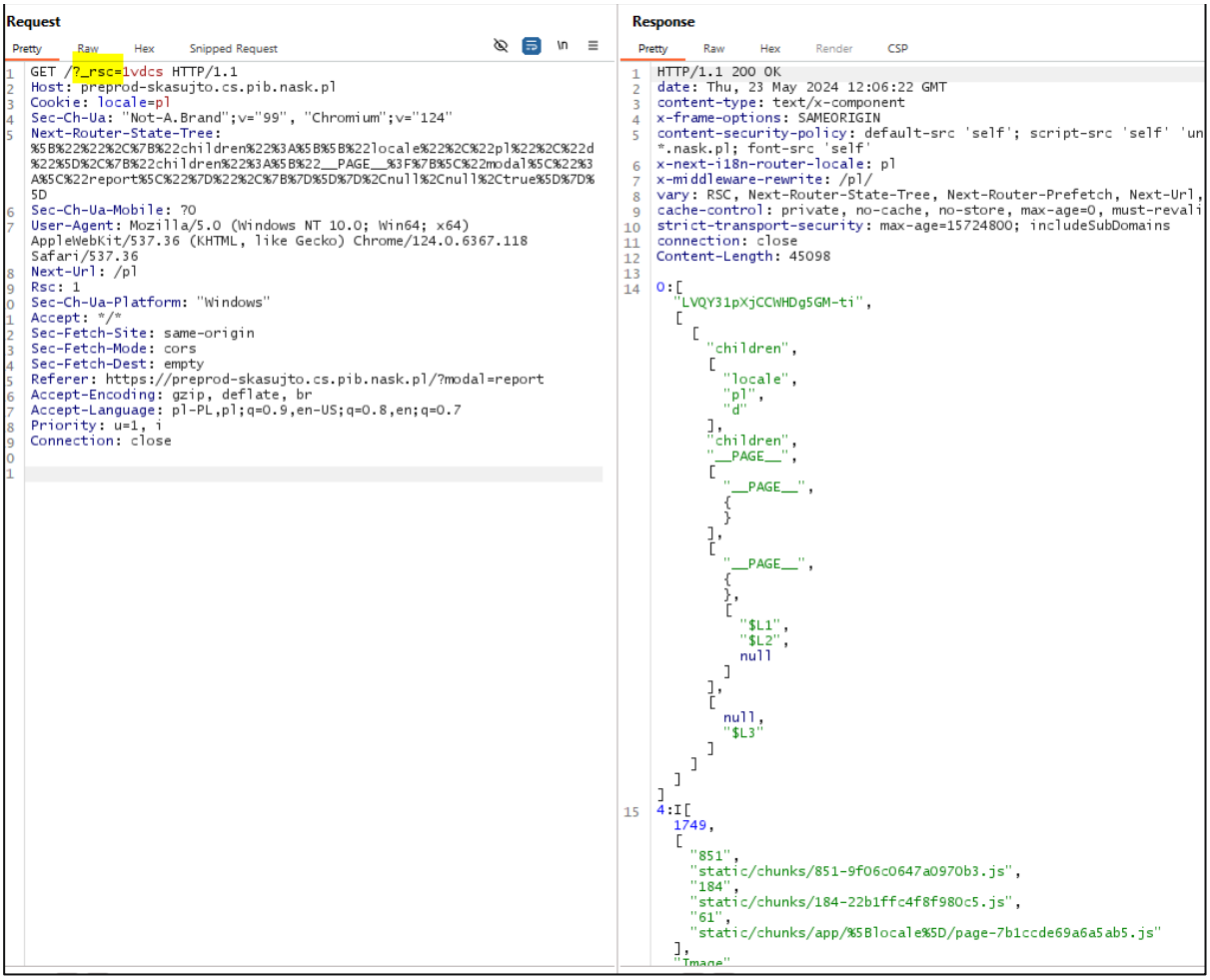
- wyświetlanie listy plików w katalogach obsługiwanych przez serwer WWW (ang. directory listing)
- pozostawienie domyślnych komunikatów błędów, ujawniających szczegóły dot. serwera WWW (np. nazwa i numer wersji serwera)
- brak nagłówków HTTP chroniących użytkowników przed pewnymi typami ataków (Clickjacking, XSS) lub niepoprawne ich ustawienie
- niepoprawna konfiguracja SSL/TLS (błędny certyfikat, dopuszczenie słabych algorytmów szyfrów)

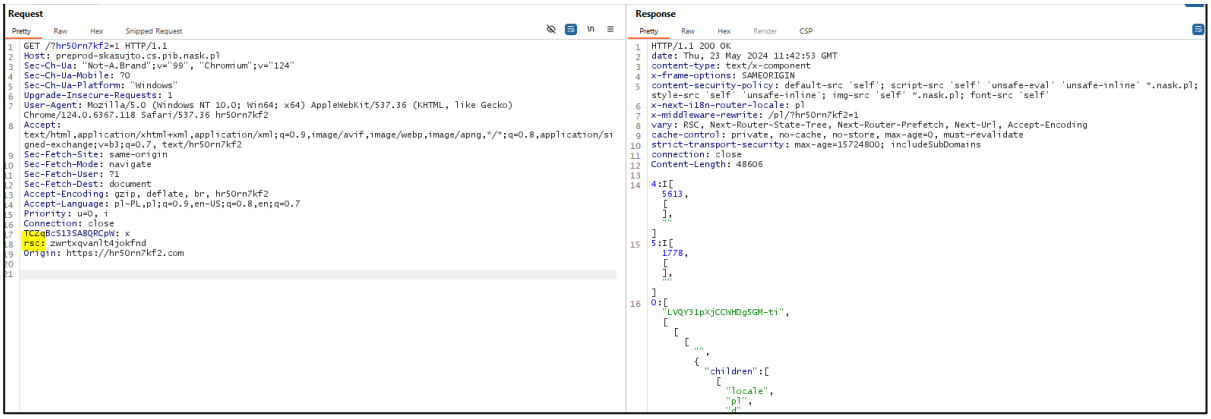
Przykład

mamy dwa endpointy

/?_rsc=losowa_Wartość

/ plus dodaje do nagłówków żądania nowy header rsc: losowa_Wartość

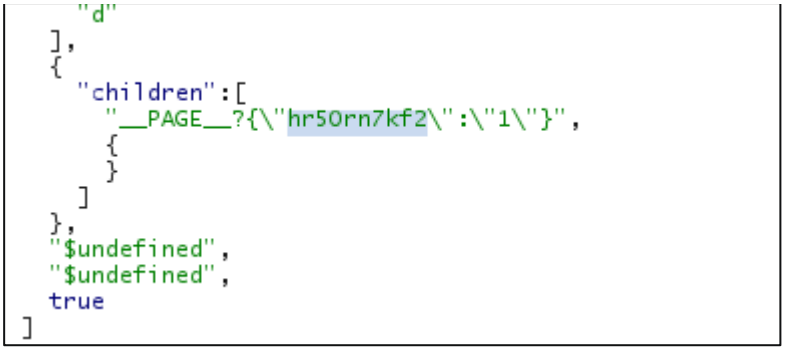




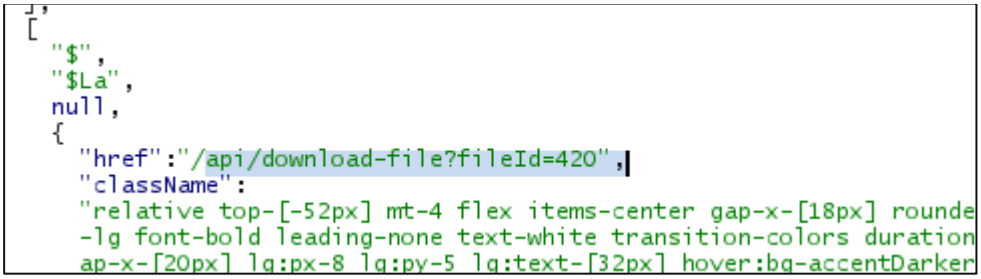
Next.js źle obsługuje nagłowek „vary”, który instruuje warstwy/usługi cachujące, jak cachować zawartość. Jeśli warstwa cachowana nie obsługuje lub niepoprawnie obsługuje nagłowek Vary wtedy serwer wysyła jsona.

Wiąże się z tym kilka podatności :

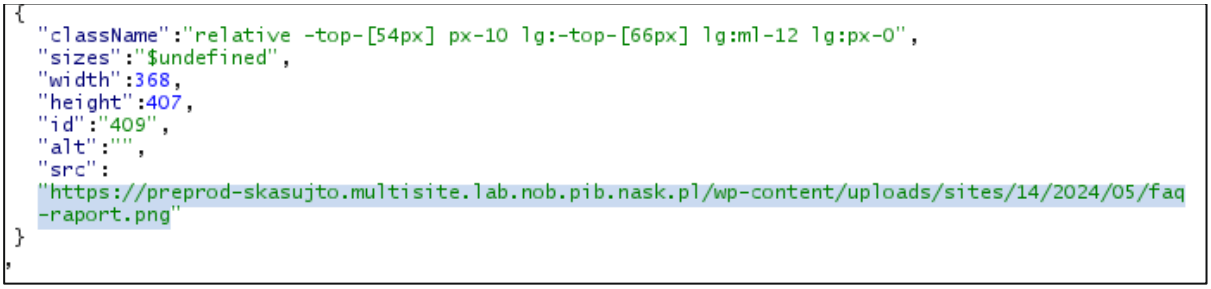
- [tylko w przypadku rsc jako header] input jest odbijany w kodzie strony



- ujawnienie ścieżki API



- adres backendu



Rekomendacja

8. IDOR

Zagrożenie

Średnie

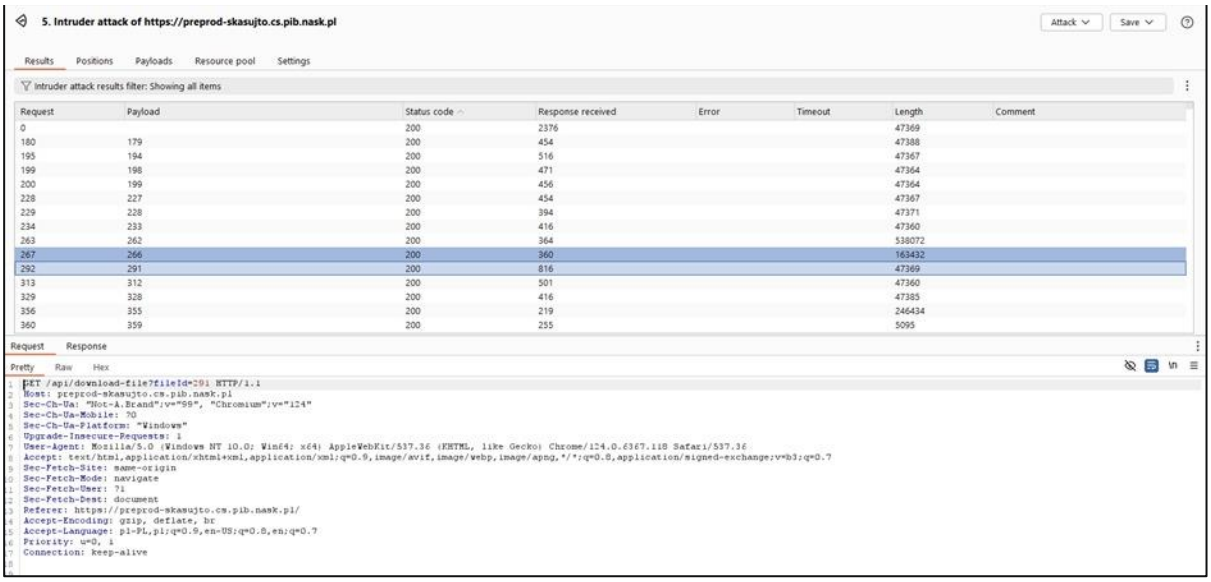
Opis

Niebezpieczne bezpośrednie odniesienie do obiektu (IDOR) to luka w zabezpieczeniach, która pojawia się, gdy osoby atakujące mogą uzyskać dostęp do obiektów lub je zmodyfikować poprzez manipulację identyfikatorami używanymi w adresach URL lub parametrach aplikacji internetowej. Dzieje się tak na skutek braku kontroli dostępu, które nie sprawdzają, czy użytkownik powinien mieć dostęp do określonych danych.

Przykład

Możliwe jest przeglądanie wszystkich plików w lokalizacji za pomocą modyfikacji parametru fileId:

```
  </title>
  <meta content="OpenGraph opis" name="description"/>
  <meta content="website" property="og:type"/>
  <meta content="Skasuj to!" property="og:title"/>
  <meta content="OpenGraph opis" property="og:description"/>
  <meta content="/api/download-file?fileId=227" property="og:description"/>
  <link rel="icon" href="/favicon.ico" type="image/x-icon" sizes="16x16"/>
  <meta name="next-size-adjust"/>
  <script src="/_next/static/chunks/polyfills-c67a75d1b6f99dc8.js" noModule="">
  </script>
</head>
```



Rekomendacja

Aby zabezpieczyć się przed tego typu atakiem, można podjąć kilka działań:

I. **Walidacja uprawnień**: Sprawdzanie, czy użytkownik, który próbuje uzyskać dostęp do zasobu, ma do tego odpowiednie uprawnienia. Każda próba pobrania pliku powinna być sprawdzana pod kątem tego, czy użytkownik ma prawo dostępu do konkretnego pliku.

II. **Używanie losowych lub trudnych do odgadnięcia identyfikatorów**: Zamiast używać sekwencyjnych identyfikatorów, można używać UUID (Universal Unique Identifier) lub innych losowych ciągów, które są trudniejsze do odgadnięcia.

III. **Kontrola dostępu na poziomie aplikacji**: Implementacja mechanizmów kontroli dostępu w aplikacji, które sprawdzają, czy użytkownik ma dostęp do zasobu przed jego udostępnieniem.

IV. **Zasada najmniejszych uprawnień**: Przyznawanie użytkownikom tylko takich uprawnień, jakie są niezbędne do wykonania ich zadań.

Niskie

9. Niepoprawnie skonfigurowana polityka Content Security Policy (CSP)

Zagrożenie

Niskie

Opis

Polityka Bezpieczeństwa Treści (Content Security Policy, CSP) to mechanizm zabezpieczający, który pozwala na kontrolę zasobów, które użytkownik może wykonać na stronie internetowej. Jej celem jest zapobieganie pewnym atakom, takim jak cross-site scripting (XSS).

Brak polityki CSP na stronie oznacza, że strona nie ogranicza źródeł zewnętrznych zasobów, które mogą być załadowane i wykonane. To otwiera drzwi dla potencjalnych ataków:

Ataki Cross-Site Scripting (XSS): Atakujący może próbować wstrzyknąć złośliwe skrypty do strony, które następnie są wykonywane przez przeglądarkę odwiedzającego.Jest to najczęstsza kategoria ataków, której można zapobiec dzięki CSP.

Wstrzykiwanie zasobów: Atakujący może próbować wstrzyknąć złośliwe zasoby (np. obrazy, skrypty, style) z nieautoryzowanych źródeł.

Przekierowanie użytkownika: Brak CSP może pozwolić atakującym na przekierowywanie użytkowników do szkodliwych stron lub na pobieranie szkodliwego oprogramowania.

Ataki typu "man-in-the-middle": Bez odpowiedniej polityki CSP, atakujący może próbować przejąć pośrednictwo w komunikacji między użytkownikiem a stroną, wprowadzając zmiany w treści lub kradnąc dane.

Stosowanie polityki CSP pozwala właścicielom witryn określić, które źródła są zaufane, ograniczając tym samym ryzyko potencjalnych ataków. Polityka ta jest deklarowana w nagłówkach odpowiedzi HTTP i mówi przeglądarkom, które skrypty, style, obrazy, czcionki itp. są dozwolone do ładowania na stronie.

Przykład

1. Unsafe-eval i unsafe-inline w script-src:

'unsafe-eval': Zezwolenie na eval (i podobne metody) jest niebezpieczne, ponieważ może to pozwolić na wykonanie złośliwego kodu JavaScript.

'unsafe-inline': Zezwolenie na wykonywanie skryptów inline eliminuje główną zaletę CSP, która polega na zapobieganiu XSS. Skrypty inline mogą być łatwo wykorzystane przez atakujących.

2. unsafe-inline w style-src:

'unsafe-inline': Zezwolenie na style inline jest ryzykowne, ponieważ atakujący mogą wstrzykiwać złośliwe style, które mogą wpływać na wygląd strony w sposób, który może wprowadzać użytkowników w błąd lub nawet wykradać dane (np. poprzez techniki phishingu).

3. **Zbyt szerokie pozwolenia na zewnętrzne zasoby:**

script-src i img-src: Pozwolenie na zasoby ze wszystkich subdomen (*.nask.pl) może być ryzykowne, ponieważ każda subdomena, która zostanie zhakowana, może wstrzykiwać złośliwy kod.

Rekomendacja

Zaleca się stosowanie odpowiednio skonfigurowanej polityki CSP, poprzez dodanie nagłówka Content-Security-Policy, na przykład:

```
Content-Security-Policy: default-src 'self'; script-src 'self' https://example.com; img-src 'self' https://example.com; style-src 'self' https://example.com; font-src 'self' https://example.com
```

Opracowywanie CSP powinno obejmować precyzyjnie założenia oraz logikę biznesową aplikacji, a w jej wykonaniu pomocnym może się okazać dokumentacja Mozilla, gdzie zostały wymienione i opisane poszczególne parametry nagłówka:

```
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy
```

10. Enumeracja serwerów/frameworków – Nginx

Zagrożenie

Niskie

Opis

Serwery HTTP czy frameworki często posiadają domyślne strony, zarówno te powitalne jak i strony komunikujące błąd po stronie aplikacji (zwracane zazwyczaj z kodami HTTP 403, 404 czy 500). Modyfikacja tych domyślnych stron jest istotna z punktu widzenia bezpieczeństwa, ponieważ ujawniają one informacje dot. środowiska (użyte oprogramowanie, wersje oprogramowania i inne). Informacje te są cenne dla potencjalnego atakującego, jako iż umożliwiają ukierunkowanie dalszych ofensywnych działań i znacznie przyspieszają przeprowadzenie udanego ataku.

Przykład

Rekomendacja

Aby uniemożliwić enumerację użytego frameworku, należy zmodyfikować lub usunąć stronę z komunikatem błędu.

11. Enumeracja serwerów/frameworków – motyw twentytwentyone

Zagrożenie

Niskie

Opis

Serwery HTTP czy frameworki często posiadają domyślne strony, zarówno te powitalne jak i strony komunikujące błąd po stronie aplikacji (zwracane zazwyczaj z kodami HTTP 403, 404 czy 500). Modyfikacja tych domyślnych stron jest istotna z punktu widzenia bezpieczeństwa, ponieważ ujawniają one informacje dot. środowiska (użyte oprogramowanie, wersje oprogramowania i inne). Informacje te są cenne dla potencjalnego atakującego, jako iż umożliwiają ukierunkowanie dalszych ofensywnych działań i znacznie przyspieszają przeprowadzenie udanego ataku.

Przykład

https://preprod-skasujto.cs.pib.nask.pl/api/download-file?fileId=291&_rsc=1r46i

```
{justify-content:space-between;}
</style>
<style id='wp-webfonts-inline-css'>
@font-face{font-family:"Source Serif
Pro";font-style:normal;font-weight:200
900;font-display:fallback;src:url('/wp-content/themes/twent
ytwentytwo/assets/fonts/source-serif-pro/SourceSerif4Variab
le-Roman.ttf.woff2')
format('woff2');font-stretch:normal;}@font-face{font-family
:"Source Serif Pro";font-style:italic;font-weight:200
900;font-display:fallback;src:url('/wp-content/themes/twent
ytwentytwo/assets/fonts/source-serif-pro/SourceSerif4Variab
le-Italic.ttf.woff2') format('woff2');font-stretch:normal;}
</style>
<link rel='stylesheet' id='twentytwentytwo-style-css'
href='https://preprod-skasujto.multisite.lab.nob.pib.nask.p
l/wp-content/themes/twentytwentytwo/style.css?ver=1.0'
media='all' />
```

Rekomendacja

Aby uniemożliwić enumerację użytego frameworku, należy zmodyfikować lub usunąć stronę z komunikatem błędu.

12. Brak bezpiecznych nagłówków HTTP

Zagrożenie

Niskie

Opis

Istnieje pewien zdefiniowany zestaw nagłówków HTTP, które uważane są za niezbędne z punktu widzenia bezpieczeństwa aplikacji internetowej oraz użytkowników tej aplikacji. Brak

któregoś z nagłówków może prowadzić do sytuacji, w której wykonanie niektórych ataków może okazać się zdecydowanie prostsze. Poniższa lista przedstawia nagłówki, które zawsze powinny być zawarte w odpowiedzi serwera:

- X-Content-Type-Options: nosniff
- X-Xss-Protection
- Refferer-Policy

Przykład

[-] Cache-Control

[-] X-Xss-Protection

[-] X-Frame-Options

[-] X-Content-Type-Options

[-] Referrer-Policy

Rekomendacja

Zaleca się dodanie następujących nagłówków do wszystkich odpowiedzi serwera:

• X-Content-Type-Options: nosniff

- X-Xss-Protection

[-] X-Xss-Protection

[-] X-Frame-Options

[-] X-Content-Type-Options

[-] Referrer-Policy

13. Nieaktualna wersja oprogramowania NextJS 14.1.4

Zagrożenie

Niskie

Opis

Podatność Nieaktualna wersja biblioteki/oprogramowania (ang. Outdated Software/Library Vulnerability) to luka w zabezpieczeniach wynikająca z użycia przestarzałych wersji bibliotek lub oprogramowania, które zawierają już znane błędy bezpieczeństwa. Atakujący mogą wykorzystać te luki w celu zdobycia dostępu do systemów lub przeprowadzenia innych ataków na aplikacje lub sieci, które wykorzystują te biblioteki lub oprogramowanie. Z tego powodu

ważne jest, aby stale monitorować i aktualizować używane biblioteki i oprogramowanie, aby zapobiec pojawianiu się podatności wynikających z nieaktualnych wersji.

Przykład

Rekomendacja

Należy zaktualizować oprogramowanie NextJS do najnowszej wersji.

Informacyjne

14. Niespójne odpowiedzi serwera aplikacji

Zagrożenie

Informacyjne

Opis

Przy próbie odwołania się przez użytkownika do nieistniejących zasobów, serwer powinien zwracać odpowiedni komunikat HTTP 404 (Not Found). Zamiast tego zwracając stronę główną bez zmiany adresu URL. Może to być związane z niewłaściwą konfiguracją serwera lub z działaniem frameworka używanego do tworzenia aplikacji webowej.

Przykład

Rekomendacja

15. Niebezpieczne metody HTTP

Zagrożenie

Informacyjne

Opis

Na serwerze włączone są metody HTTP, takie jak TRACE, PUT i DELETE. Metody te mogą pozwolić osobie atakującej na dołączenie i/lub usunięcie plików lub przeprowadzenie ataków typu cross-site tracing.

Przykład

Payload	Status code
GET	200
PUT	200
POST	200
DELETE	200
PATCH	200
HEAD	
OPTIONS	200
TRACE	405
CONNECT	400

Rekomendacja

16. Brak nagłówka Refferer-Policy

Zagrożenie

Informacyjne

Opis

Nagłówek Referer-Policy w odpowiedzi HTTP określa, jakie informacje o stronie źródłowej (referer) powinny być dołączane podczas przenoszenia użytkownika na inną stronę lub podczas żądania zasobów z innej domeny. To jest ważne z punktu widzenia prywatności użytkownika, ale także bezpieczeństwa aplikacji.

Brak nagłówka Referer-Policy w odpowiedzi może prowadzić do różnych problemów i ryzyk:

Ujawnienie wrażliwych informacji: Jeśli URL, z którego pochodzi użytkownik, zawiera wrażliwe informacje (np. identyfikatory sesji, tokeny czy parametry), to brak odpowiedniej polityki referera może prowadzić do ujawnienia tych danych stronie docelowej lub innym zasobom.

Ryzyko ataków typu "phishing": Atakujący może wykorzystać informacje zawarte w refererze, aby lepiej dostosować ataki phishingowe, bazując na stronach, które użytkownik odwiedził.

Problemy z analizą ruchu: Dla wielu witryn, które korzystają z narzędzi analitycznych, referer jest kluczowym źródłem informacji o tym, skąd pochodzą ich użytkownicy. Brak odpowiedniej polityki może prowadzić do niepełnych lub mylących danych w narzędziach analitycznych.

Naruszenie prywatności: Dla wielu użytkowników kwestia prywatności w sieci jest kluczowa. Przekazywanie informacji o odwiedzanych stronach bez ich wiedzy i zgody może być uważane za naruszenie tej prywatności.

Przykład

Rekomendacja

Aby unikać powyższych problemów, zaleca się dodanie nagłówka Referer-Policy do odpowiedzi HTTP i odpowiednie skonfigurowanie go. Istnieje kilka wartości, które można ustawić dla tego nagłówka, takich jak no-referrer (nie przesyła informacji o refererze), same-origin (przesyła referer tylko w przypadku żądań w obrębie tej samej domeny) czy strict-origin-when-cross-origin (przesyła tylko źródło w przypadku żądań międzydomenowych), w zależności od potrzeb i poziomu bezpieczeństwa, który chcemy osiągnąć.