# Change Bender's password

W opisie zadania podane zostało, że hasło ma zostać zmienione na slurm4cl4ssic. Miało ono zostać zmienione użytkownikowi Bender, do którego konta udało się już dostać w jednym z wcześniejszych zadań.

W tym challengu nie można używać funkcji ‘Forgot password’, dlatego wykorzystano konto innego niż w zadaniu użytkownika w celu sprawdzenia dostępnych opcji.

Po zalogowaniu się na jedno z kont, w zakładce „Privacy and Security” dostępne było pole ‘Change password’. Pasowało więc ono do celu zadania.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatyczniePo wpisaniu danych w wymagane pola ‘Current password’, ‘New password’ oraz ‘Repeat new password’, w narzędziach developerskich można było znaleźć żądanie GET, które zostało wygenerowane.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Czcionka

Opis wygenerowany automatycznie

http://localhost:3000/rest/user/change-password?current=asdf&new=12345h&repeat=12345h

Po próbach manipulacji tym linkiem, po usunięciu części z ‘current’ udało się zmienić hasło – i to bez podawania obecnego. Do wysyłania requestów trzeba było wkleić w pole Athorization token Bearer, który wysyłany jest przez serwer po zalogowaniu.

http://localhost:3000/rest/user/change-password?new=12345h&repeat=12345h

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Skoro wiadomo już, że ta metoda zmiany hasła działa, należy teraz zalogować się na konto Bender’a, w celu uzyskania jego Bearer tokena w celu zmiany hasła na jego koncie. Ponieważ nie jest znane nam jego aktualne hasło, skorzystamy z metody wykorzystanej w poprzednich zadaniach.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Po poprawnym zalogwaniu się z użyciem tej metody skopowiany został token.

Obraz zawierający tekst, zrzut ekranu, numer, Czcionka

Opis wygenerowany automatycznie

Udało się w ten sposób zmienić hasło i zadanie zostało zaliczone.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

# Leaked Access Logs

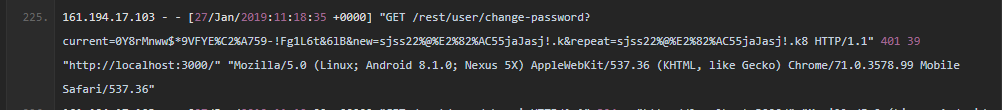
Następne zadanie opisane jest jako przeszukiwanie internetu w celu znalezienia zleakowanego hasła, a następnie zalogowanie się z jego pomocą do konta użytkownika, do którego należy.

Przeszukiwanie internetu doprowadziło do portalu stackoverflow i profilu Bjorna Kimminich’a, pierwotnego twórcy Juice Shopa. Znalezione zostało tam zadane przez niego pytanie dotyczące access logs, które pasowało do struktury znanej już z poprzednich zadań.

Obraz zawierający tekst, zrzut ekranu, Czcionka, Strona internetowa

Opis wygenerowany automatycznie

Przeglądając logi wklejone do pastebin, znalezione zostało hasło, o którym mowa w zadaniu:



**current**=0Y8rMnww$\*9VFYE%C2%A759-!Fg1L6t&6lB&**new**=sjss22%@%E2%82%AC55jaJasj!.k&**repeat**=sjss22%@%E2%82%AC55jaJasj!.k8

Widoczne jest, że w odpowiedzi zwrócony został błąd http 401 Unauthorized, co skłoniło do przyjrzenia się hasło podanym w new oraz repeat. Hasła te różnią się – w jednym na końcu jest znak ‘&’, a w drugim ‘8’.

Obraz zawierający tekst, oprogramowanie, zrzut ekranu

Opis wygenerowany automatycznie

Może oznaczać to, że hasło pozostało niezmienione, więc takie jak podane w ‘current’.

0Y8rMnww$\*9VFYE%C2%A759-!Fg1L6t&6lB

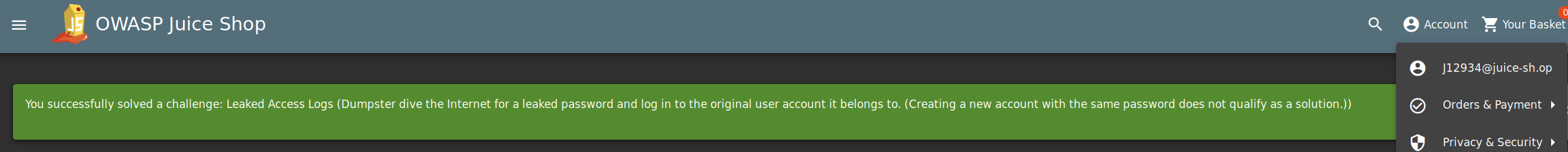
Ponieważ hasło to pochodzi z URLa musi zostać poddane operacji URL decode – wynikiem tego jest otrzymane hasło 0Y8rMnww$\*9VFYE§59-!Fg1L6t&6lB

Z pomocą znalezienia nazwy użytkownika przyszedł wcześniej odszukany panel administratora z rejestrem wszystkich użytkowników.

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Prawidłową nazwą użytkownika powiązaną z otrzymanym hasłem okazał się [J12934@juice-sh.op](mailto:J12934@juice-sh.op)



# Extra Language

Kolejnym challengem było odnalezienie języka, który nigdy nie znalazł się w systemie produkcyjnym.

W prawym górnym rogu aplikacji możliwa jest zmiana jej języka.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Podczas zmiany języków w narzędziach developerskich można zauważyć, w jaki sposób nazywane są pliku zawierające poszczególne języki:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Podczas przeszukiwania internetu w celu znalezienia informacji o językach wspieranych przez Juice Shop oraz tego, w jaki sposób te tłumaczenia są wykonywane znaleziona została strona <https://crowdin.com/project/owasp-juice-shop>. Porównując języki znajdujące się w bazie na powyższej stronie odnaleziony został język Klingon, którego nie ma w aplikacji Juice Shop. Jest to wymyślony język, lecz na Wikipedii jest podany kod tego języka:

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Znany więc jest już pierwszy człon nazwy pliku, potrzebny jest jeszcze drugi – pisany wielkimi literami dwuznakowy kod. W celu odnalezienia pliku na stronie napisany został skrypt przeprowadzający atak brute force, który próbuje na stronie różne rodzaje nazw pliku w formacie tlh\_XX.json.

Z wykorzystaniem tego skryptu (find\_language.py w repozytorium) udało się znaleźć plik, o który chodziło i zadanie zostało ukończone.

Obraz zawierający tekst, Czcionka, zrzut ekranu, Grafika

Opis wygenerowany automatycznie

W celu użycia skryptu należy go pobrać, a następnie uruchomić w odpowiednim miejscu (tam, gdzie jest pobrany) komendą:

*python find\_language.py*

# Blockchain Hype

Wyzwanie to opiera się o konieczność odnalezienia informacji o sprzedaży tokenów w obrębie analizowanej aplikacji webowej, przed tym, jak wspomniana informacja trafi do oficjalnych wiadomości twórców. Podpowiedź zawarta w treści sugeruje, iż analizie powinien zostać poddany kod aplikacji, w którym ukryte mogą być dalsze wskazówki koniecznego do przeprowadzenia ciągu akcji.

Pierwszym krokiem było włączenie inspektora kodu, który wbudowany jest w obsługiwaną przeglądarkę. Następnie wyszukiwane przy pomocy narzędzia były pojęcia, które mogłyby być powiązane z przedmiotową sprawą. Do ów pojęć zaliczane były między innymi: „token”, „sale”, „token sale” oraz „token-sale”. Pierwsze dwie opcje zwracały zbyt wiele wyników, trzecia z kolei nie dostarczała żadnych informacji, natomiast ostatni parametr przynosił jeden wynik.

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

Następnie, po odnalezieniu przedstawionego fragmentu kodu, przystąpiono do analizy parametrów wchodzących w skład widocznej metody. Zamieszczonych było tam kilka linków, które nie były powiązane z aplikacją Juice Shop, a zewnętrznymi źródłami. Ostatni z parametrów natomiast prezentował się w sposób zbliżony do takiego, który przedstawiałby ścieżkę do pliku.

Obraz zawierający tekst, Czcionka, linia, oprogramowanie

Opis wygenerowany automatycznie

Po próbie przesłania przytoczonego na grafice parametru jako rozwinięcia adresu URL bazowego dla Juice Shop, użytkownik przekierowany został na nową podstronę. Widoczna była na niej niewielkich rozmiarów ikona, która poddana została analizie przy ponownym wykorzystaniu inspektora przeglądarki.

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

Analiza obiektu nie wykazała nowych informacji, natomiast wykonane akcje uznane zostały przez Juice Shop jako wystarczające dla uznania wyzwania za zakończone. Obnażone zostały podatności związane z brakiem bezpieczeństwa kodu źródłowego.

# Email Leak

Wyzwanie to dotyczyło wykonania próby ataku mającego na celu sprokurowanie wycieku danych, które nie powinny zostać udostępnione do informacji publicznej. Pomimo sugestii zawartej w tytule, odnoszącej się do wycieków odnoszących się do wiadomości mailowych bądź innych powiązanych aspektów, w projekcie skupiono się na pozyskaniu dowolnych dowodów wycieku.

W celu realizacji tego zadania wykonane zostały akcje przytaczane w poprzednich rozdziałach projektu, za pomocą których pozyskano metodę HTTP GET. Odnosiło się ono do informacji powiązanych z parametrami konta, z którego wykonywane są akcje. W trakcie przeprowadzania eksperymentu ruch sieciowy generowany był przez użytkownika jim@juice-sh.op.

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Widoczna metoda mogła zostać zmodyfikowana w sposób, który usuwałby zawartość pola odpowiedzialnego za autoryzację, by zweryfikować jakie informacje zostaną wypisane w odpowiedzi po wykonaniu tego działania.

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Co jest widoczne na przykładzie powyższego zrzutu ekranu, w odpowiedzi generowany zostaje szablon, który mógłby sugerować możliwość pozyskania dodatkowych informacji o użytkowniku, które wcześniej nie były dostępne z poziomu GUI lub analizy pakietów. Wprowadzona została zatem modyfikacja, która podawała nowy parametr „callback” odpowiedzialny za oczekiwanie pełnego zwrotu informacji w odpowiedzi http wygenerowanej przez aplikację.

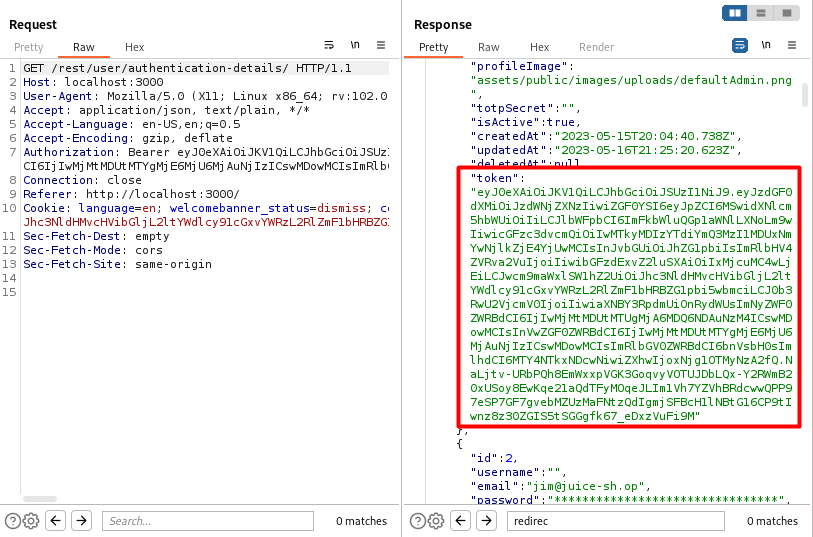
Obraz zawierający tekst, Czcionka, numer, Strona internetowa

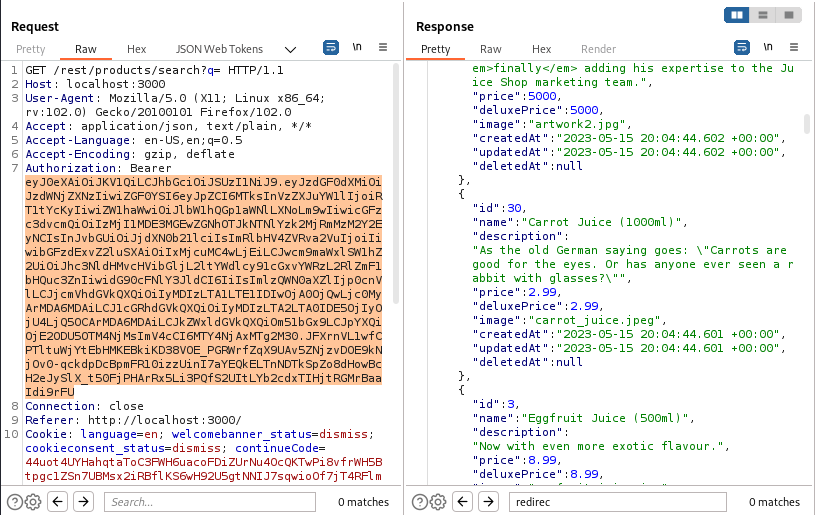
Opis wygenerowany automatycznie

Zawarte w odpowiedzi informacje nie powinny być dostępne do wglądu dla niepowołanych osób, możliwe jest zatem określenie występującego incydentu jako wyciek danych poza odpowiedni obszar. Parametr opisywany wcześniej, możliwy był również do zastosowania w odpowiednio spreparowanym adresie URL, kierowanym w stronę aplikacji. Dzięki niemu pozyskane zostałyby te same informacje, co w zaprezentowanym przykładzie.

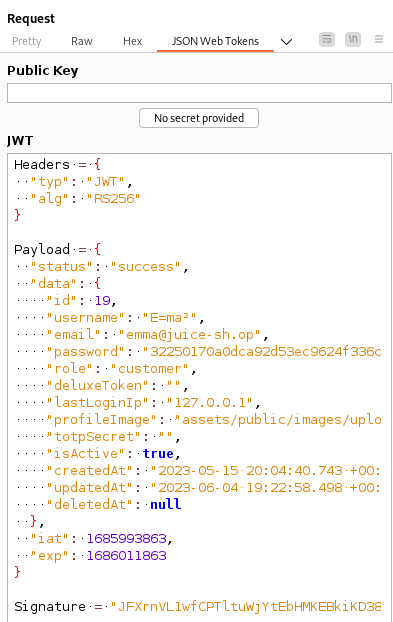
# Unsigned JWT

Zadanie polega na wygenerowaniu niepodpisanego tokenu JWT, który podszywa się pod (nieistniejącego) użytkownika [jwtn3d@juice-sh.op](mailto:jwtn3d@juice-sh.op). Jako pierwszy krok, odszukano JWT w żądaniach i odpowiedziach.

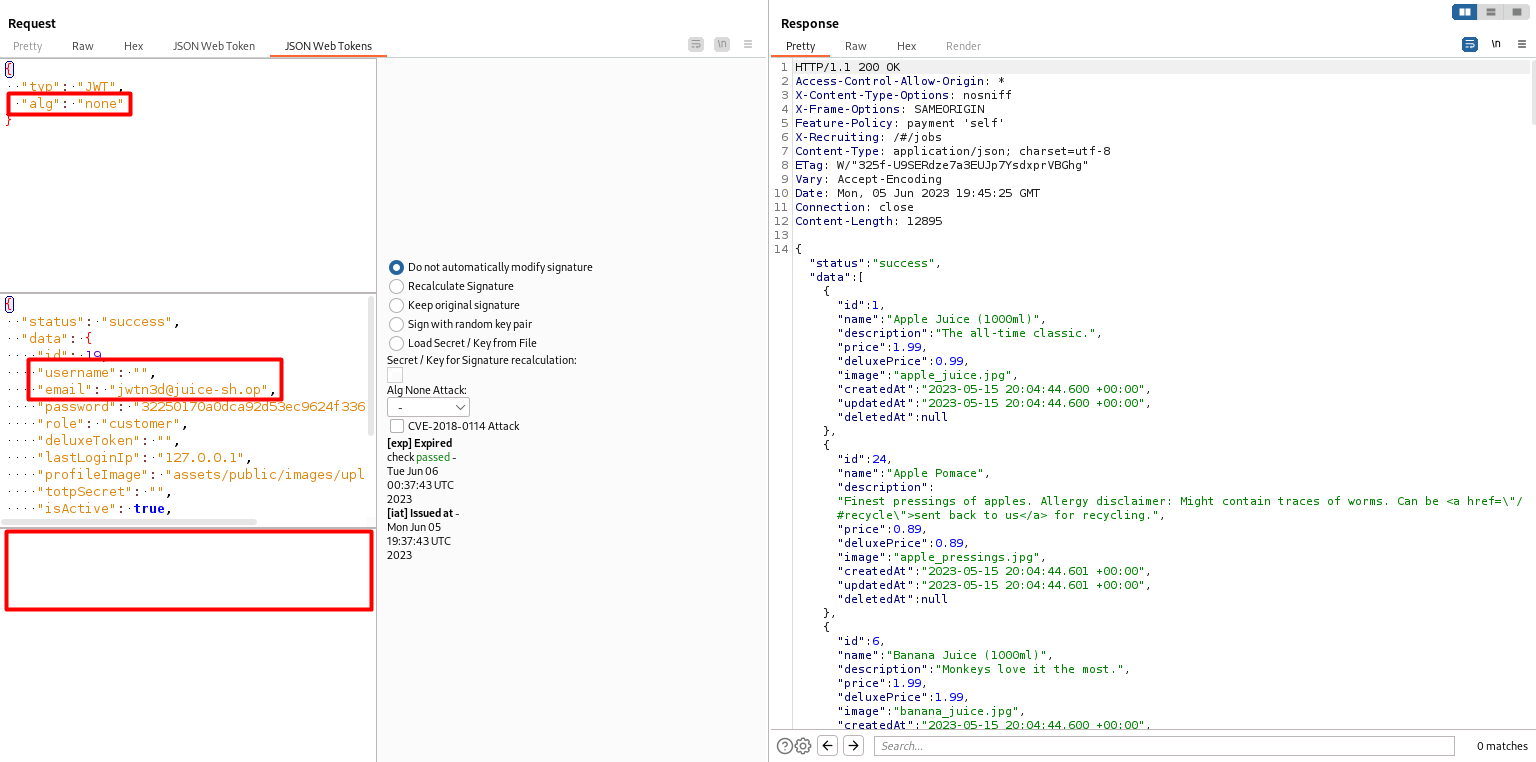




Za pomocą rozszerzenia JSON Web Tokens, odszyfrowano token:



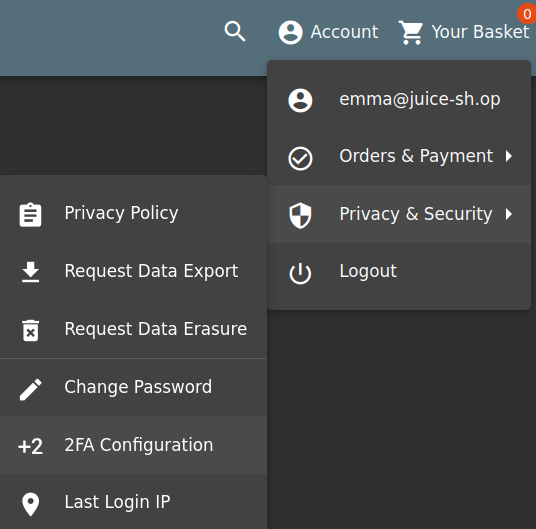
Postanowiono usunąć username, edytować email oraz zmienić algorytm na ‘none’ (token ma być niepodpisany) i usunąć podpis.



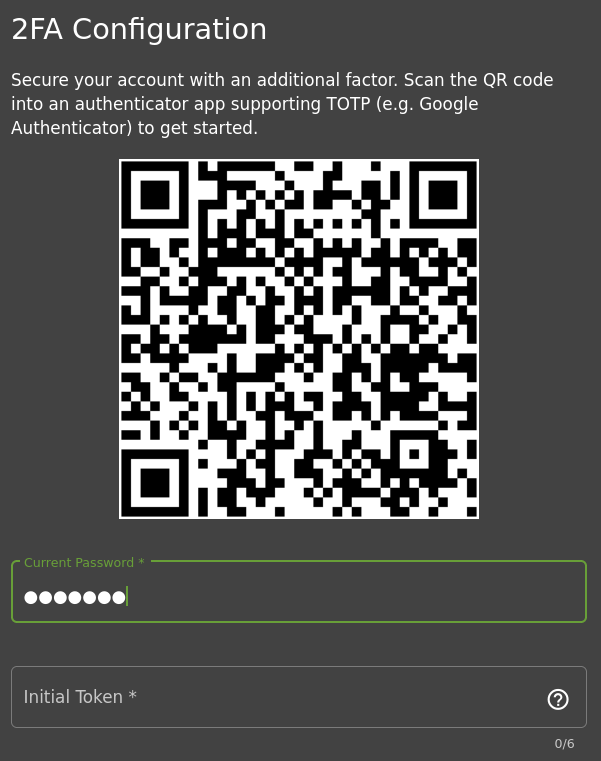
W ten sposób udało się rozwiązać challenge.

# Two Factor Authentication

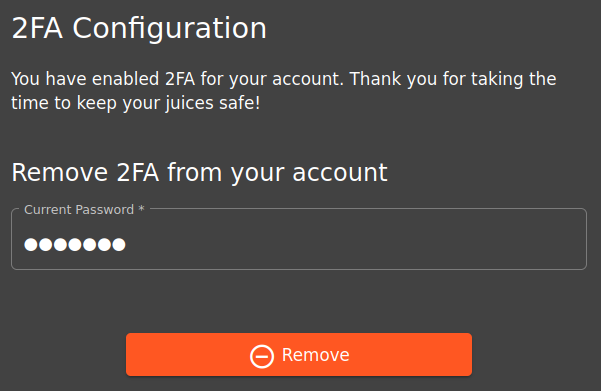
Zadanie polega na złamaniu 2FA dla użytkownika wurstbrot. Jako pierwsze poszukano informacji o 2FA w aplikacji:



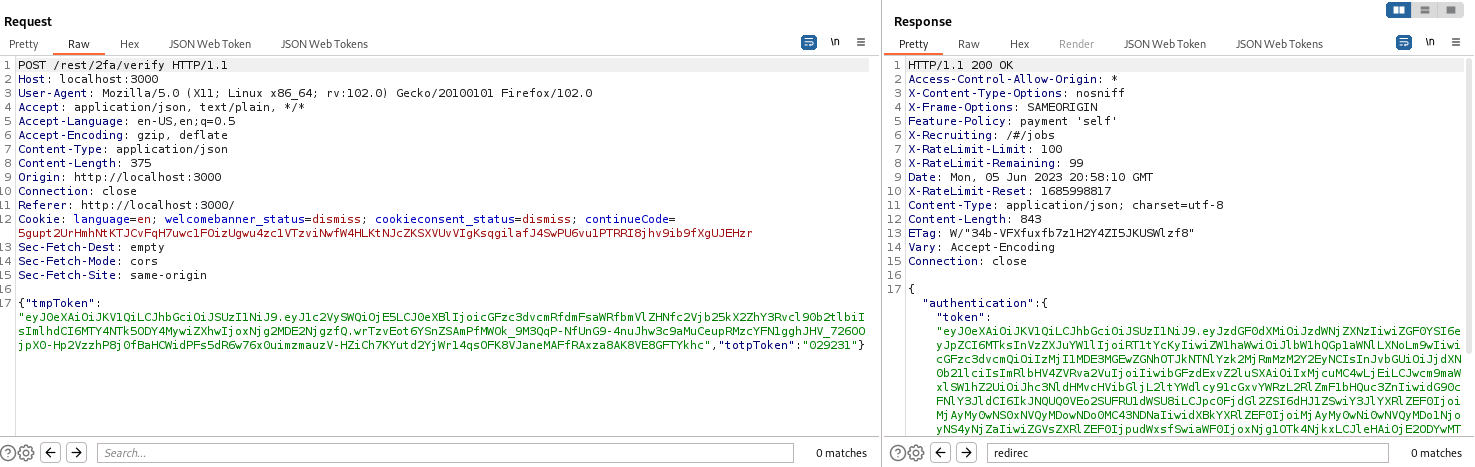
Pobrano aplikację Google Authenticator na prywatne urządzenie i zeskanowano kod QR:

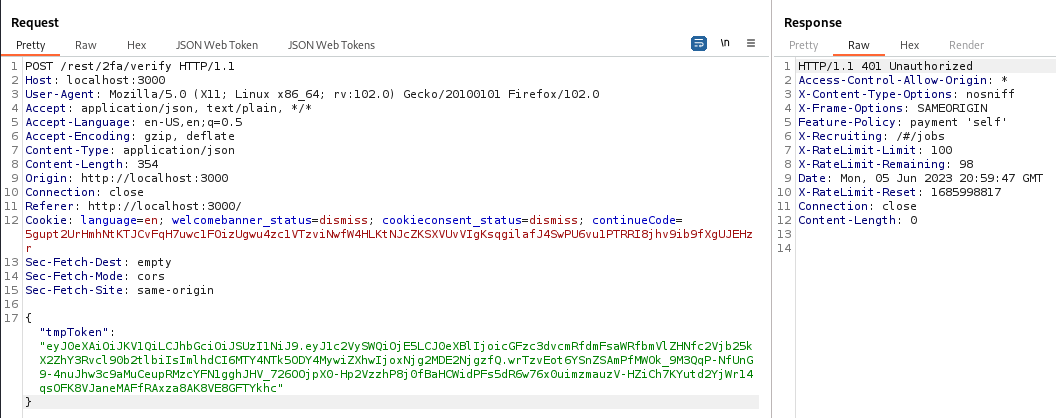


Po wpisaniu kodu pojawił się komunikat o włączeniu 2FA:

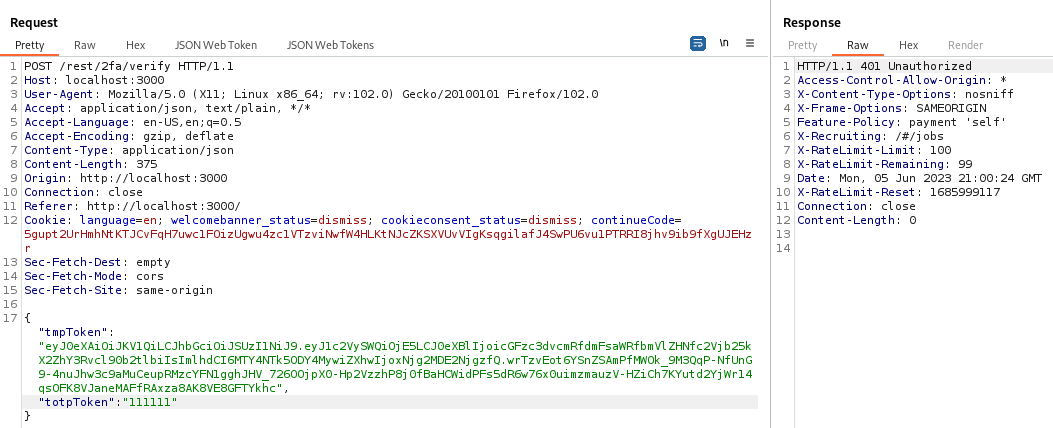


Wylogowano się i zalogowano spokojnie, aby sprawdzić, czy 2FA działa:

Fragmenty żądań i odpowiedzi niestety nic nie mówiły. Próbowano usunąć token:



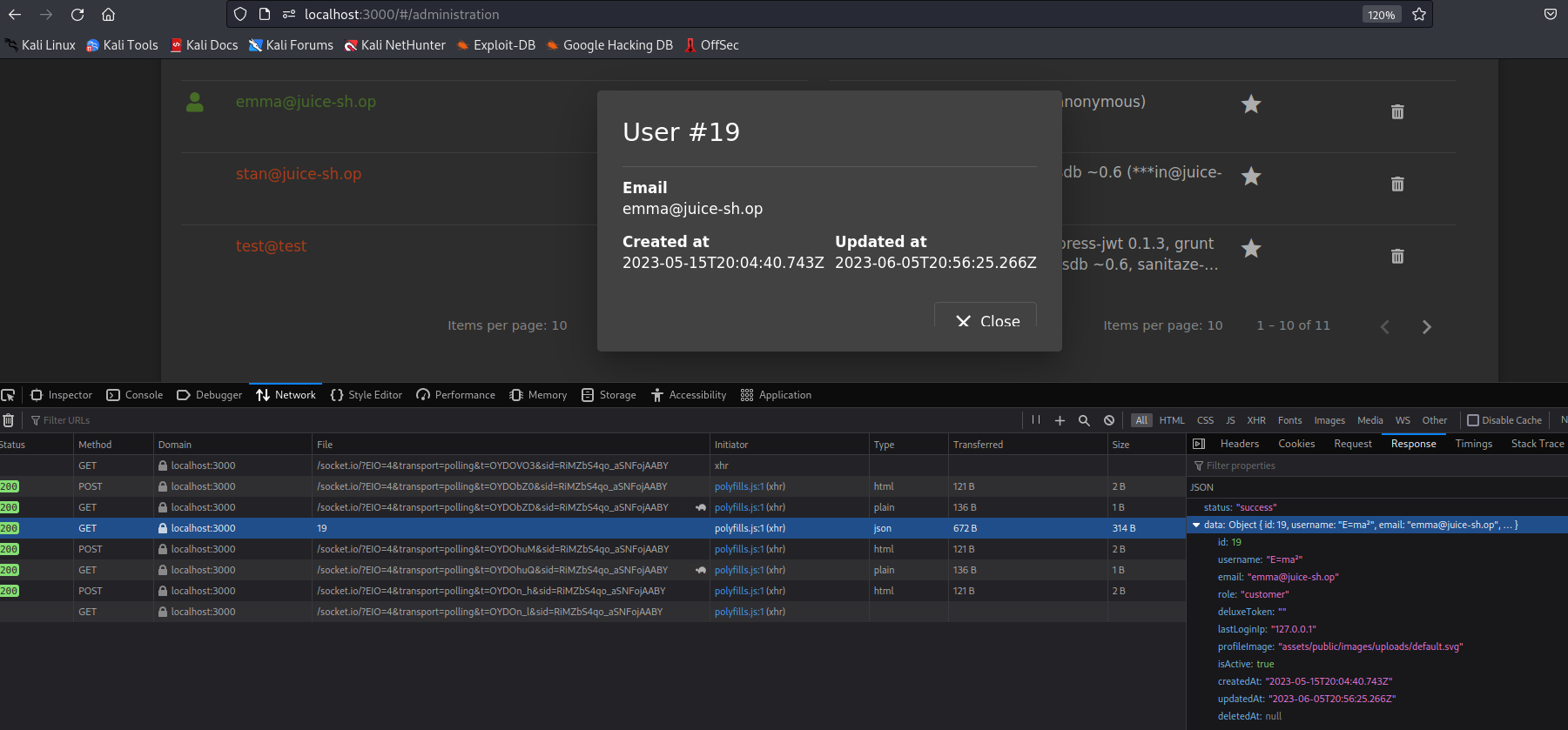
Oraz złamać token:



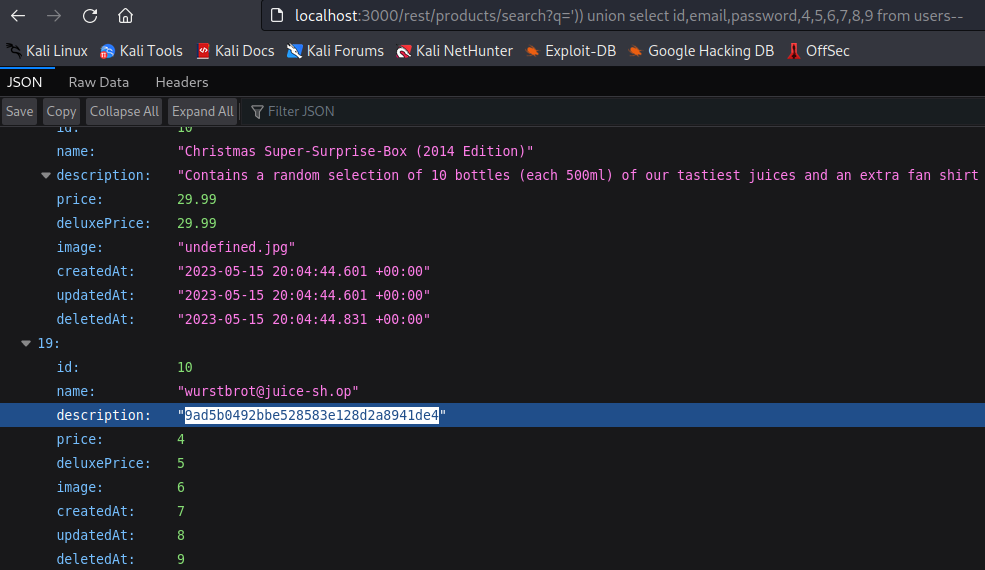
Postanowiono poszukać informacji o użytkownikach na koncie administratora. Znaleziono użytkownika wurstbrot:



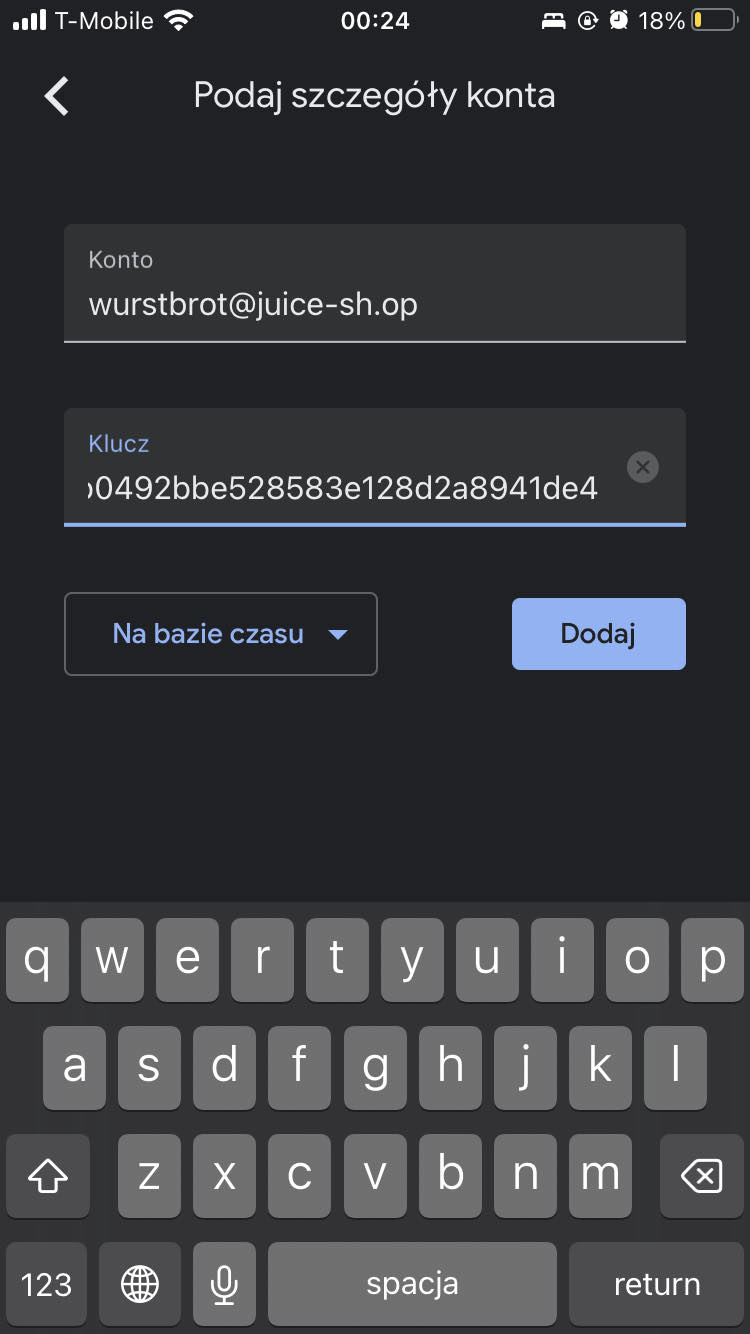
Sprawdzono te informacje u Emmy za pomocą DevTools (zakładka Network):



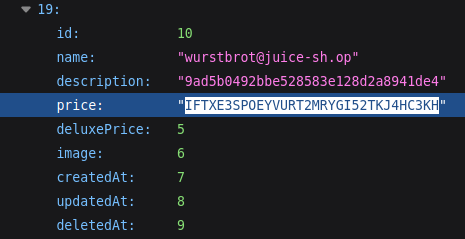
Jednak nic nowego się tam nie pojawiło. Postanowiono wykorzystać poprzednio znalezioną podatność SQL injection (jedno z wyzwań 4 stars) i wylistować użytkowników. Wurstbrot to użytkownik o numerze 10. Znaleziono następującą wartość:



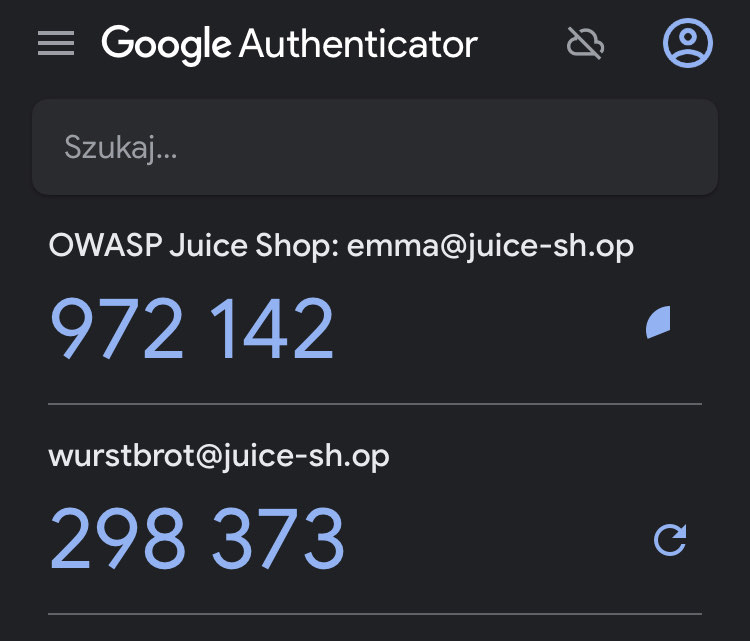
Spróbowano te dwie wartości wpisać w authenticatorze:



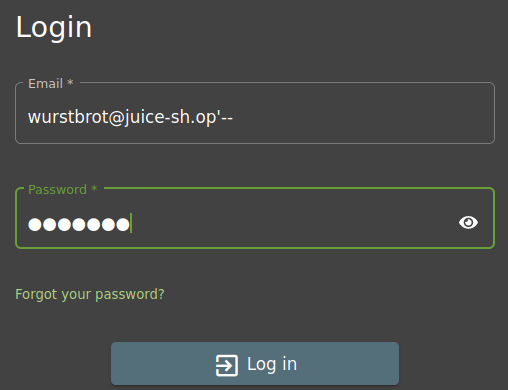
Klucz okazał się nieprawidłowy, spróbowano także wpisać go dużymi literami, jednak to też nic nie dało. Poszukano zatem innych informacji w bazie danych. Z uwagi na fakt, że w żądaniu token został podany jako totpToken, dodano to do zapytania SQL.



Token nie pojawił się, za to pojawiła się specyficzna cena. Sprawdzono to także jako klucz w aplikacji authenticator.



W ten sposób udało się zdobyć kody do konta wurstbrot. Następnie zalogowano się klasycznie na to konto – przy użyciu SQL injection:



Następnie wpisano kod 2FA z aplikacji:

