Błędy na stronie czy w implementacjach zdarzają się każdemu. Ostatnio przyjrzeliśmy się Dotpay'owi, a dziś padło na stronię jbzdy.co czyli słynne już JebZDzidy należące do firmy Cube Investments Sp. z o.o.

Zapewne słyszeliście, że w Polsce padła już pierwsza milionowa kara za RODO. Dziś przedstawimy drugich w kolejności kandydatów do tego zaszczytnego wyróżnienia jakim jest „Milion za RODO".

Nie każdy zna pojęcie „urban legend", ale na pewno każdy spotkał się z tym zjawiskiem. Urban legend, a dokładnie miejską legendą nazywamy krążące wśród znajomych, ale i po internecie nieweryfikowalne lub fałszywe historie oraz najczęściej o zabarwieniu emocjonalnym.

„**Legenda miejska** – pozornie prawdopodobna informacja rozpowszechniana w mediach, internecie lub w kręgach towarzyskich, która budzi wielkie emocje u odbiorców. Legenda miejska elektryzuje słuchaczy i prowokuje ich do komentarzy i dalszego jej przekazywania, dotyczy bowiem sytuacji mogących się przytrafić każdemu lub znanych osób. Era Internetu i szybkiej wymiany informacji znacznie przyspieszyła obieg tego rodzaju plotek, obecnych od dawna w grupach społecznych. Legendy miejskie często są przekształceniem wcześniejszych opowieści ludowych – na przykład opowieść o czarnych wołgach (którymi rzekomo porywano ludzi, zwłaszcza dzieci, w epoce PRL), czy o pedofilach (lub narkomanach) zabierających małe dzieci z centrów handlowych."

Każda społeczność ma taką miejską legendę, pastę lub inny charakterystyczny element. Przykładowo dla społeczności wykop.pl słowami kluczami są:

– astronauta

– afera zbożowa

– nocna zmiana w serwerowni

Dla 4chana

– „I lost my iPad"

– „over 9000 thousand"

– „I Herd U Liek Mudkips"

Powyższe sformułowania były hasłami rozpoznawczymi dla pierwszych członków społeczności Anonymous. Obecnie weszły one do powszechnego użycia. Czasem zdarza się, że członkowie społeczności internetowych w określonych celach używają tych sformułowań niestety coraz częściej są one tylko synonimami popularnych określeń.

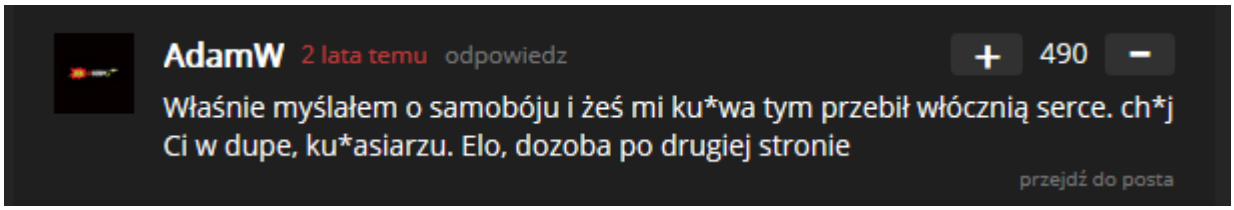
Dla społeczności wcześniej wspomnianego portalu JebZDzidy.pl słowami kluczami są:

– „wojna stulejna”

– AdamW

– #witam

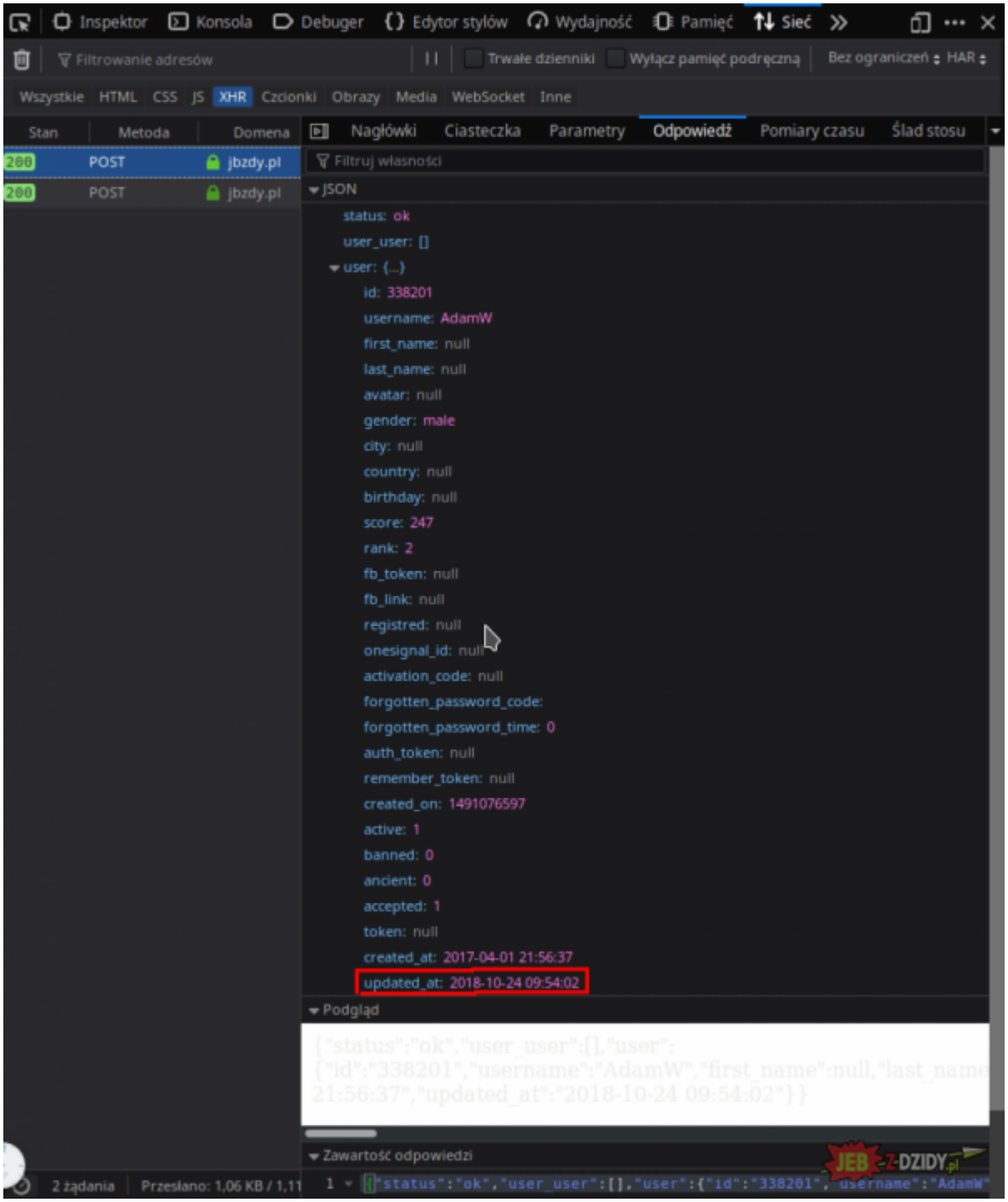
Dziś to właśnie między innymi wątkiem użytkownika o pseudonimie AdamW się zajmiemy. Cała historia zaczęła się od jednego łzawego obrazka w stylu „Prawdopodobnie nawet nie wiesz, że przegapiłeś miłość życia”. Jeden z użytkowników portalu zamieścił komentarz o treści:



Pierwszą reakcją użytkowników były niewybredne żarty czy życzenia powodzenia. Minęło kilka godzin i pośród żartów pojawił się pierwszy post mówiący, że może AdamW mówił serio. Niestety autor komentarza już nigdy się nie zalogował. Pod komentarzem są obecnie dziesiątki komentarzy, już nie tylko złośliwych i wyśmiewających, ale wyrażających obawy o Adama, czy po prostu świeczki.

Tutaj kończy się bezpośrednia historia użytkownika, a zaczyna się miejska legenda portalu JebZDzidy.pl. Wielu użytkowników od ponad dwóch lat dyskutuje na temat tego czy AdamW naprawdę targnął się na swoje życie czy też jest to jeden z najlepszych trollingów ostatnich lat.

Jakiś czas temu na mikroblogu (jeden z elementów składowych portalu JBZDy.pl) pojawił się screen pokazujący dane przekazywane do użytkownika w JSONIE. Screen miał dowodzić tego, że user AdamW logował się znacznie później na swoje konto.



Przyznam szczerze, że mocno mnie zastanawiało, jakim cudem (i przede wszystkim po co) takie dane są dostępne dla zwykłego usera. Pierwszą myślą było to, że może admin wrzucił screen by jednorazowo i bezspornie rozwiązać kwestie związane z losami użytkownika. Ponad to, lepiej wizerunkowo by było, gdyby użytkownicy portalu jednak się nie zabijali.

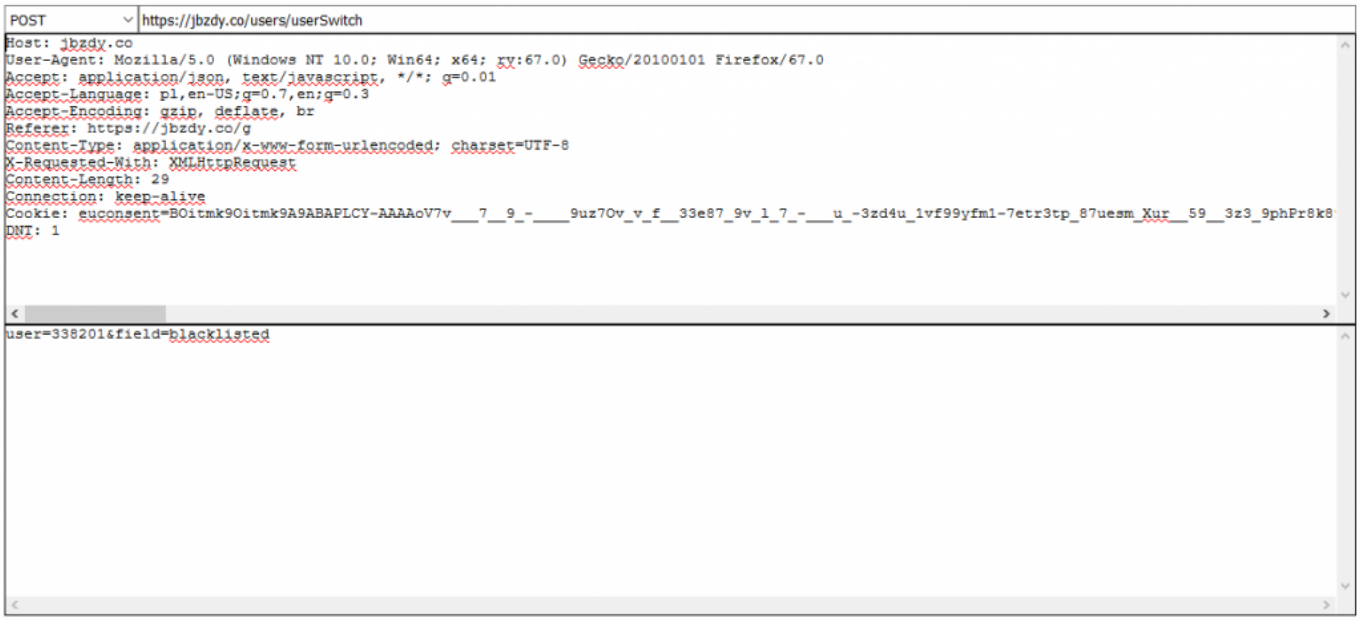
Po dłuższym przypatrywaniu się screenowi zrozumiałem, że osoba wrzucająca screen nie tylko nie umie korzystać z narzędzi WebDev w Firefoxie, ale również nie do końca rozumie czym jest JSON. Poprosiłem o informacje, w którym miejscu znajduje się ten request i nie mogłem uwierzyć swoim oczom, gdy zobaczyłem to czego nie widać w FF.

Błąd, a może feature „umieszczony jest w" guziku służącym do dodawania innego użytkownika na czarną listę. Z jakiegoś powodu guzik ten, nie tylko dodaje w bazie danego użytkownika do naszej czarnej listy, ale również zwraca w formacie JSON dane zapisane w panelu użytkownika oraz może się wydawać wszystko co zawiera tabela „users" w bazie. Parametrem odpowiedzialnym za otrzymany wynik widoczny na screenie jest „getUser". W wywołanym parametrze otrzymujemy między innymi takie dane jak:

- Płeć
- Datę urodzenia
- Imię
- Nazwisko
- Nick
- Id
- Czy user jest zbanowany
- Czy jest aktywny

I wiele innych kwestii, które może wprowadzić użytkownik.

Gdy zacząłem przyglądać się dogłębniej komunikacji JSON'owej zauważyłem, że nie tylko nie jest to jedyny request, który się pojawia, ale również jest to jeden z kilku requestów, który podaje nam dane użytkownika. Zapytanie z parametrem „getUser" pojawia się po najechaniu myszką, natomiast po kliknięciu go, otrzymujemy wynik zapytania o parametrze „userSwitch". Za pomocą dodatku do przeglądarki nagrałem ten fragment komunikacji sieciowej i potem modyfikując zawartość zapytań pobrałem dane z serwera.



Po jego wykonaniu zapytania moim oczom ukazały się poniższe dane.

```
{
   "status":"ok",
   "user_user":{
      "id":"916259",
      "user_id":"573719",
      "user_user_id":"338201",
      "blacklisted":"1",
      "followed":"0",
      "created_at":"2019-06-28 07:52:57",
      "updated_at":"2019-06-28 07:52:57"
   },
   "user":{
      "id":"338201",
      "ip_address":"[redacted]",
      "username":"AdamW",
      "first_name":null,
      "last_name":null,
      "avatar":null,
      "gender":"male",
      "city":null,
      "country":null,
      "birthday":null,
      "birthday_valid_to":null,
      "score":"345",
      "rank":"2",
      "password":"[redacted]",
      "fb_id":"0",
      "fb_token":null,
      "fb_link":null,
      "salt":"[redacted]",
      "registred":null,
      "email":"[redacted]",
      "onesignal_id":null,
      "activation_code":null,
      "forgotten_password_code":"",
      "forgotten_password_time":"0",
      "remember_code":"[redacted]",
      "auth_token":null,
      "remember_token":null,
      "created_on":"1491076597",
      "last_login":"1494350116",
      "active":"1",
      "banned":"0",
      "ancient":"0",
      "accepted":"1",
      "api_tester":"0",
      "token":null,
      "created_at":"2017-04-01 21:56:37",
      "updated_at":"2018-10-24 09:54:02"
   }
```

Aby uzmysłowić jak bardzo „złe” jest to, że powyższe dane są dostępne dla każdego zalogowanego w serwisie użytkownika, krótko omówię poszczególne części.

Pierwszym blokiem danych jest blok „user_user”. Jak wynika z jego treści zawiera on informacje, o tym jaki jest id danego zapytania (parametr ID) – dzięki temu możemy oszacować ilość akcji tego typu wykonanych przez użytkowników, a więc ocenić realną aktywność użytkowników w serwisie. Kolejnymi dwoma interesującymi parametrami są „user_id” oraz „user_user_id” są to parametry informujące odpowiednio, który user (user_id) wykonuje akcje wobec którego usera (user_user_id). Trywialność nazewnictwa może sugerować, iż programista tworzył swojego rodzaju prowizorkę. Kolejnymi parametrami jest „blacklisted” oraz „followed”, które zapewne informują o tym czy dodaliśmy użytkownika na czarną listę czy do obserwowanych. Ostatnimi parametrami w tym bloku danych to parametry „created_at” oraz „updated_at”. Parametry te informują, kiedy zostało stworzone zapytanie oraz kiedy jego wynik został ostatni raz zmodyfikowany. Prawdopodobnie rozbicie tego na dwa elementy ma na celu wyłapywanie opóźnień między wysłaniem zapytania a zwróceniem jego wyniku. Dokładność wynosi 1s.

O ile dane z powyższego bloku, nie są w żaden sposób wrażliwe, to dane z bloku nazwanego „user” są już ewidentną nominacją do nagrody „Milion za RODO”, ale po kolei…

Blok danych „user” zbudowany jest z poniżej zaprezentowanych danych.

Parametr	Opis
id	ID użytkownika, którego dane są wyświetlane.
ip_address	Adres IP ostatniego logowania.
username	Nick ,którym użytkownik identyfikuje się w serwisie.
first_name	Imię podane w panelu.
last_name	Nazwisko podane w panelu.
avatar	Nazwa pliku z awatarem.
gender	Płeć podana w panelu.
city	Miasto podane w panelu.
country	Kraj podany w panelu.
birthday	Data urodzenia podana w panelu.
birthday_valid_to	Data „ważności" daty urodzenia – prawdopodobnie po tym czasie użytkownik zostanie ponownie zapytany o datę urodzenia.
score	?
rank	Prawdopodobnie poziom uprawnień użytkownika.
password	Hash hasła w md5.
fb_id	ID użytkownika FB – uzupełniany podczas logowania via fb.
fb_token	Token użytkownika FB.
fb_link	?
salt	„sól" hasła używana ho hashowania.
registred	?
email	Adres email użytkownika.
onesignal_id	?
activation_code	Kod aktywacyjny – więcej informacji w dalszej części artykułu.
forgotten_password_code	Kod do resetu hasła – więcej informacji w dalszej części artykułu.
forgotten_password_time	Czas wywołania resetu hasła.
remember_code	?
auth_token	Token autoryzacyjny?
remember_token	?
created_on	Data stworzenia konta.
last_login	Data ostatniej aktualizacji profilu.
active	Czy użytkownik jest aktywny?
banned	Czy użytkownik jest zbanowany?
ancient	?
accepted	?
api_tester	?
token	?
created_at	Data stworzenia profilu w formacie zrozumiałym dla człowieka.
updated_at	Data ostatniej aktualizacji profilu w formacie zrozumiałym dla człowieka.

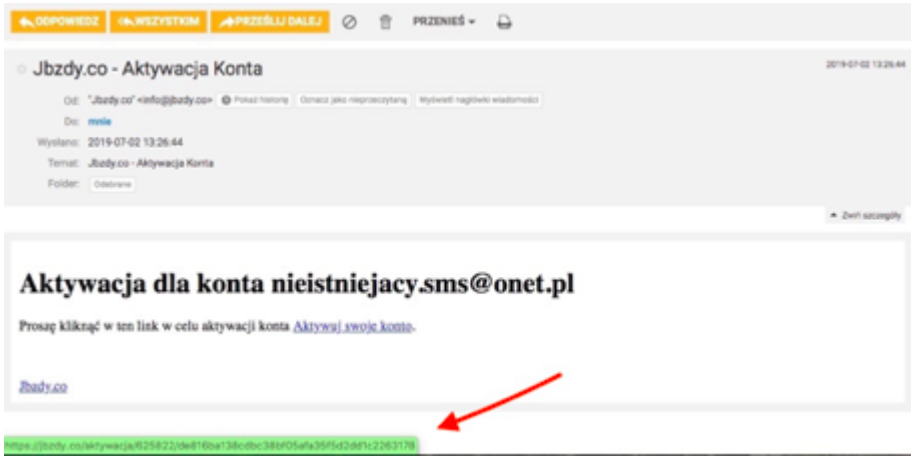
Teraz, gdy znamy już zastosowanie większości wartości zwracanych spróbujmy zaplanować oraz przeprowadzić dwa „ataki" na spójność danych w bazie.

Pierwszym atakiem jest założenie konta bez posiadania istniejącego maila. Pozwoli nam na to parametr „activation_code". Założyłem mail na Onecie nieistniejący.sms@onet.pl i zarejestrowałem na niego konto po to by zobaczyć, jak wygląda link aktywacyjny.

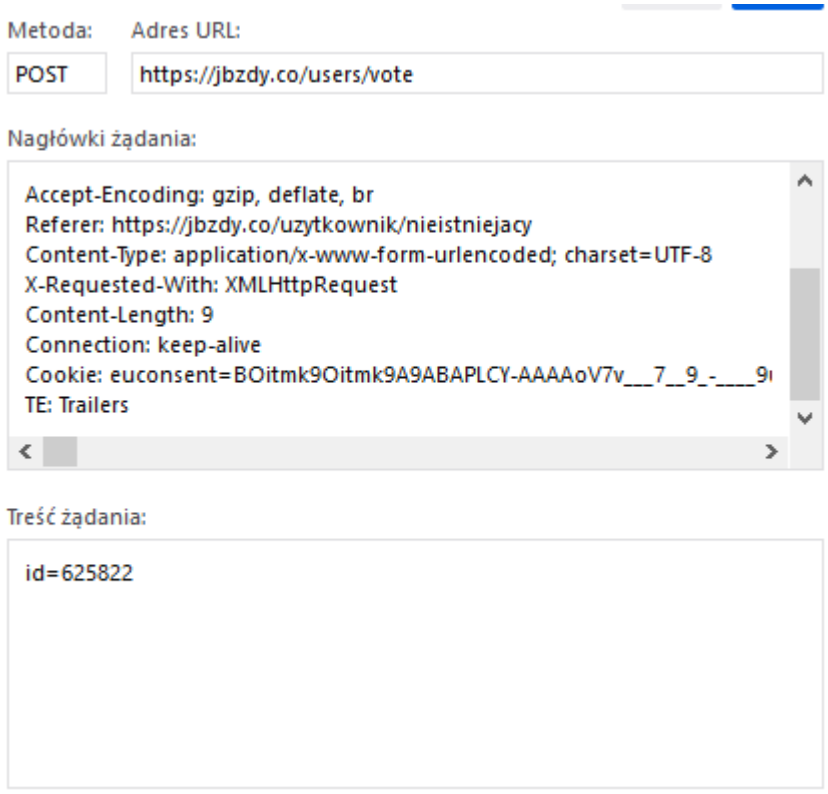
Gdy otrzymałem mail sprawdziłem, jak wygląda link aktywacyjny.

Zgodnie z przewidywaniami było to według określonego wzoru:

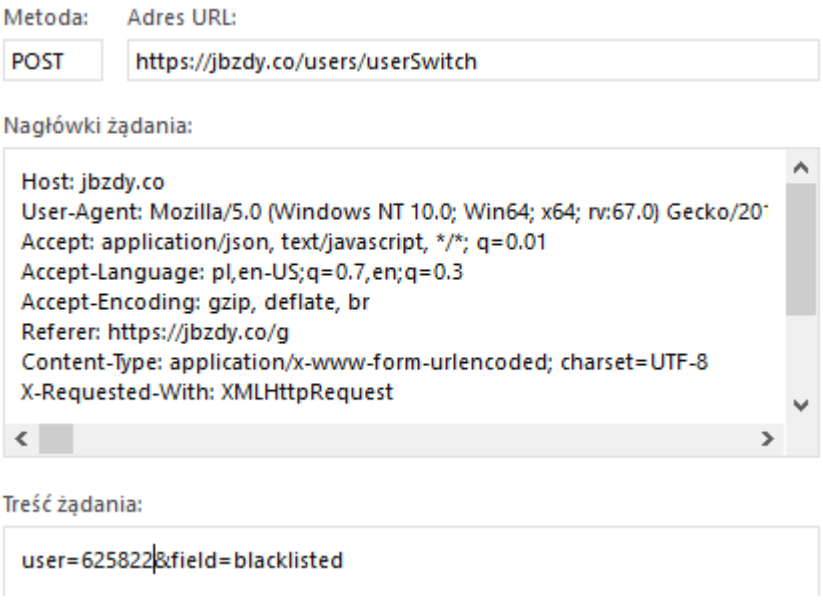
Domena.pl/id_usera/kod_aktywacyjny



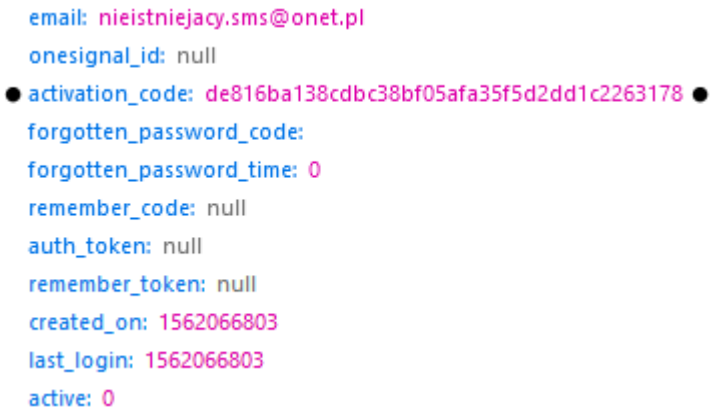
Aby poprawnie przeprowadzić atak, potrzebujemy znać jaki ID został nadany naszemu użytkownikowi. Zaraz po utworzeniu profil nowego usera jest dostępny do wglądu. W tym przypadku jest to https://jbzdy.co/uzytkownik/nieistniejacy. Wystarczy podejrzeć requesty, aby odkryć jaki jest właściwy numer ID. ID użytkownika możemy znaleźć w requeście odpowiadającym za dodawanie plusów.



Kiedy mamy już ID użytkownika, za pomocą requestu z parametrem „userSwitch” znajdujemy kod aktywacyjny.



Po wykonaniu zapytania naszym oczom ukazuje się…

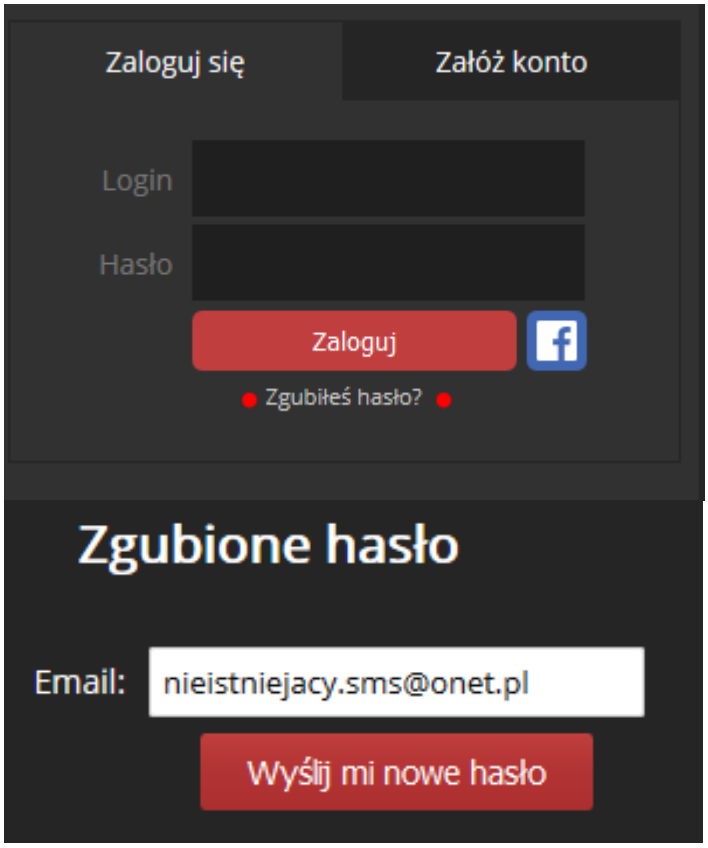


Jak widać, możemy aktywować konto spisując dane. Co nam to daje? Nie tylko możemy aktywować sobie konto bez podawanie swojego prawdziwego adresu email – jest to w pewien sposób anonimizacja użytkownika, ale także możemy założyć konto na kogoś, przykładowo „andrzej.duda@prezydent.pl” i postować cenzodudy. Niestety, są też inne wykorzystania tej luki. Można stworzyć konto sugerujące, iż jego właścicielem jest ktoś inny i postować treści, które mogą kogoś obciążać.

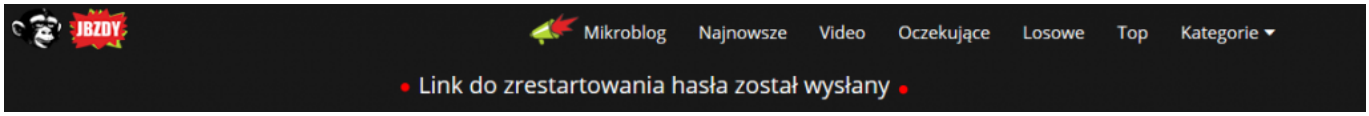
Drugi atak jaki możemy przeprowadzić wykorzystując dane, które serwuje nam strona to przejęcie konta dowolnego użytkownika, którego ID jesteśmy w stanie odczytać z komunikacji z stroną.

Pomoże nam w tym przesyłany parametr „forgotten_password_code”. Dla przykładu znów posłuży nam użytkownik „nieistniejący”.

Aby przejąć kontrolę nad kontem najpierw musimy znać jego id. Tę informację pozyskaliśmy poprzednio oraz jego adres email, który również odczytaliśmy za pomocą parametru „userSwitch”. Gdy znamy już email użytkownika przechodzimy do formularza przypomnienia hasła.



Po podaniu hasła klikamy „wyślij mi nowe hasło”. I po chwili ukazuje się komunikat o tym, iż nowe hasło zostało wysłane.

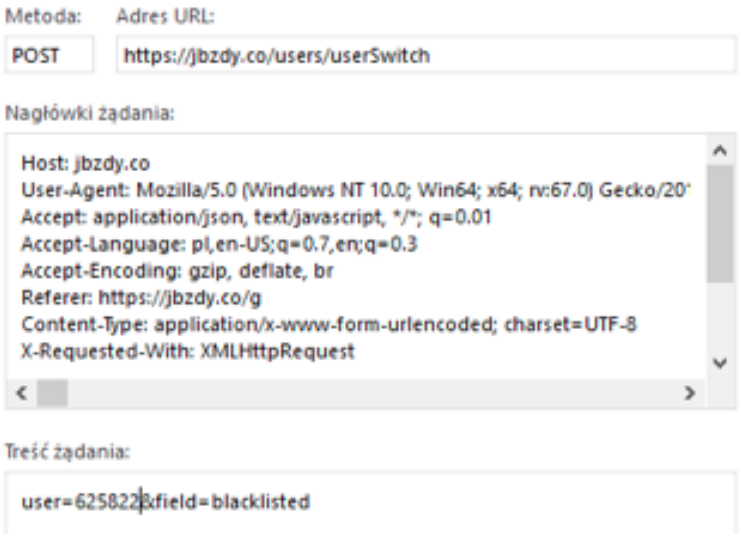


Aby wykorzystać zdobyte do tej pory dane musimy ustalić jaki link jest do zmiany hasła. Tutaj również nie było większego zaskoczenia.

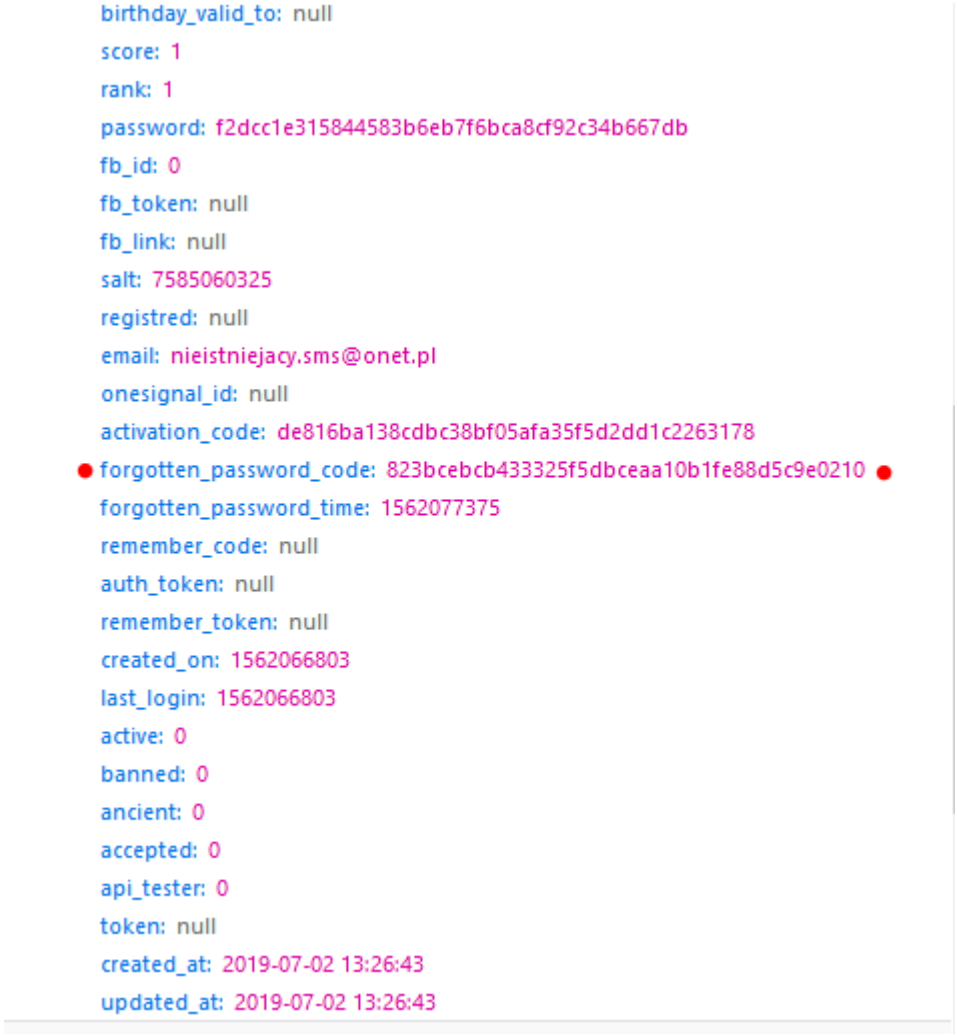
Format linku jest również dość prosty:

Domena.pl/zmiana-hasla/kod

Aby uzyskać kod ponownie wykonujemy zapytanie:



Po wysłaniu zapytania otrzymujemy dane potrzebne do resetu hasła:

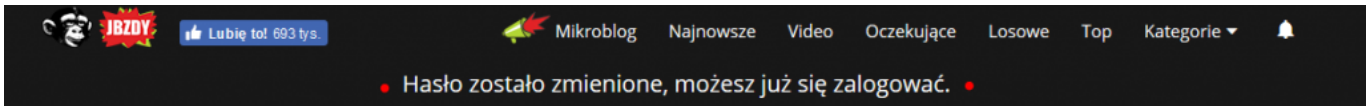


Z powyższych danych przygotowujemy link do resetu.

https://jbzdy.co/zmiana-hasla/823bcebcb433325f5dbceaa10b1fe88d5c9e0210

Po wejściu w ten link docieramy do formularza ustawienia nowego hasła.

Podajemy nowe hasło i klikamy zmień hasło.



W ten sposób w kilku prostych krokach pokazałem, jak można przejąć dowolne konto w portalu jbzdy.pl.

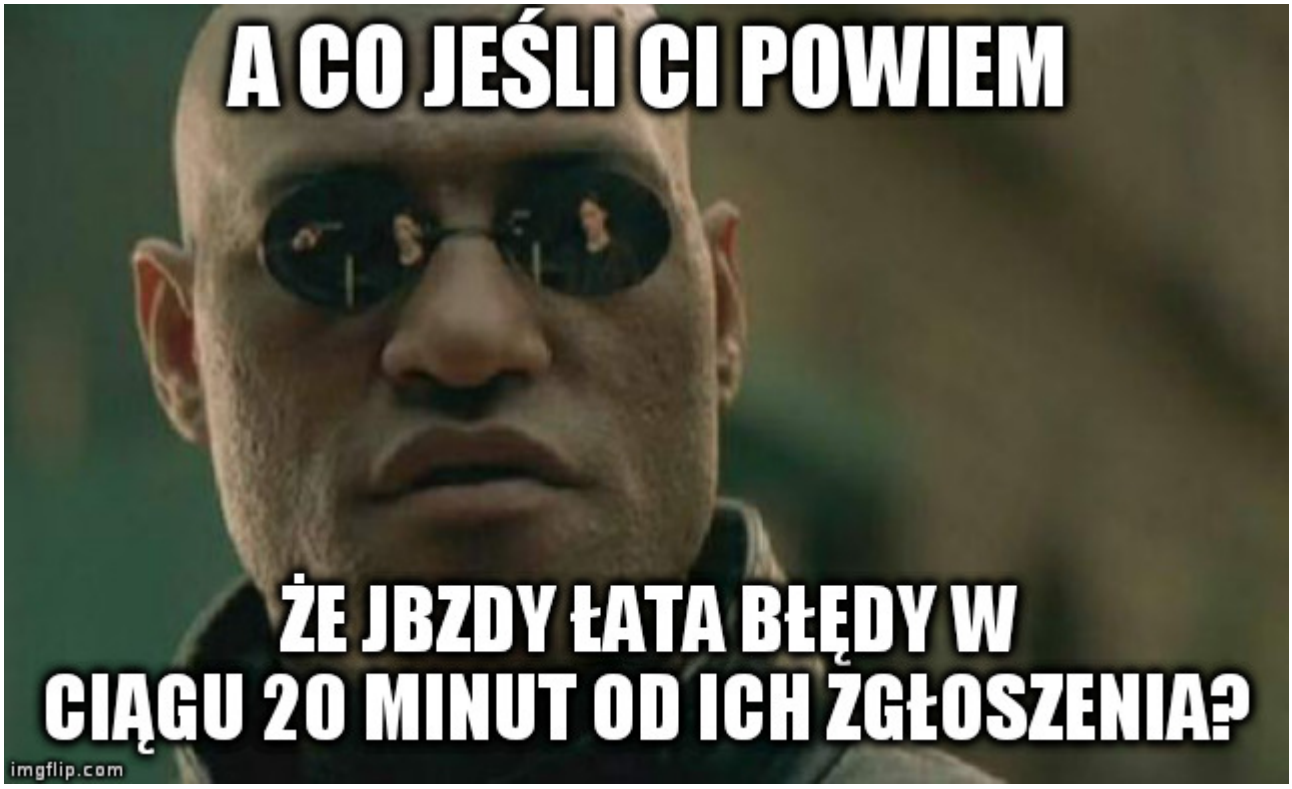
Jak widać z portalu można nie tylko wydobyć dane wrażliwe podlegające pod ochronę ustawy o ochronie danych osobowych, nie mówiąc już o tym, że pozwalają one między innymi na przejęcie konta użytkownika, a przypuszczalnie i administratorów strony.

Co na to Cube Investments Sp. Z o. o., które jest właścicielem portalu?

Cube nie tylko stanął na wysokości zadania, ale również bardzo pozytywnie nas zaskoczył. Kontakt z właścicielem JBZD nawiązałem dzień po skończeniu artykułu. Około godziny 13 dostałem potwierdzenie, że moja wiadomość została przeczytana. Zaledwie dwadzieścia minut później zauważyłem pierwsze poprawki na stronie. Tego samego dnia wieczorem otrzymałem maila od przedstawiciela JBZDY.

> Panie Piotrze,
> Dziękujemy za sympatię do Jebzdzidy, doceniamy głęboką wiedzę o społeczności, która kwitnie na naszym serwisie.
> Staramy się utrzymywać nasz serwis w jak najlepszej kondycji a bezpieczeństwo jest dla nas priorytetem. Dlatego czujemy się szczególnie źle po przedstawionych przez Pana niedoskonałościach. Tym bardziej, że ma Pan 100% rację w każdym z aspektów. Programiści działają już pełną parą.
> Bez względu na Pańską wiadomość jesteśmy w przed dzień wdrożenia nowej,

ulepszonej wersji serwisu, w której poruszanych przez Pana zagadnień nie ma. Przejrzeliśmy Waszą firmową stronę i chętnie zapoznamy się z Waszą ofertę na dogłębne sprawdzenie nowej wersji jebzdzidy, albo raczej jbzdy.co. Tu sięgnę do tematu zmienianych przez nas domen. Nie ma w tym zagadnieniu tajemnic: dostajemy tzw. bana od google i aby utrzymać serwis musimy przekierować go na inną domenę.



Co z tego wynika? Otóż jasno widać, że Cube wzięło sobie do serca wskazane przeze mnie błędy oraz bardzo profesjonalnie i sprawnie na nie zareagowało poprawiając je od ręki. Jak wynika z dalszej komunikacji, nie tylko mając w planach nową wersję strony nie olali sprawy, ale i poprawili obecną. W porównaniu z innymi instytucjami, którym zgłaszałem błędy, ustanowili absolutny rekord w czasie reakcji na zgłoszone zdarzenie. Oby więcej firm z takim podejściem.

Dziś rano spotkała mnie kolejna niespodzianka. Do moich drzwi zawitał kurier z paczką, a w niej…


W podziękowaniu za energetyczne testy bezpieczeństwa 😉

Serdecznie dziękujemy za napoje! Na pewno się przydadzą. Gratulujemy też błyskawicznej i świetnej reakcji na zgłoszenie!

PS. Każda podatność zgłaszana przez PHT/S.M.S. jest publikowana po naprawieniu/7 dniach bez odpowiedzi/kontaktu z strony zainteresowanych.