

1. Przygotowanie środowiska

Podstawowym i najważniejszym środowiskiem do przeprowadzania testów aplikacji iOS w moim przypadku jest iPhone 8 z systemem iOS w wersji 16.0, z jailbreakiem wykonanym za pomocą narzędzia checkra1n.

Pozostałe narzędzia niezbędne do przeprowadzenia testów:

OpenSSH - narzędzie to uruchamia serwer SSH, dzięki któremu w bardzo prosty sposób można uzyskać dostęp do terminala iPhone'a z poziomu systemu Linux.

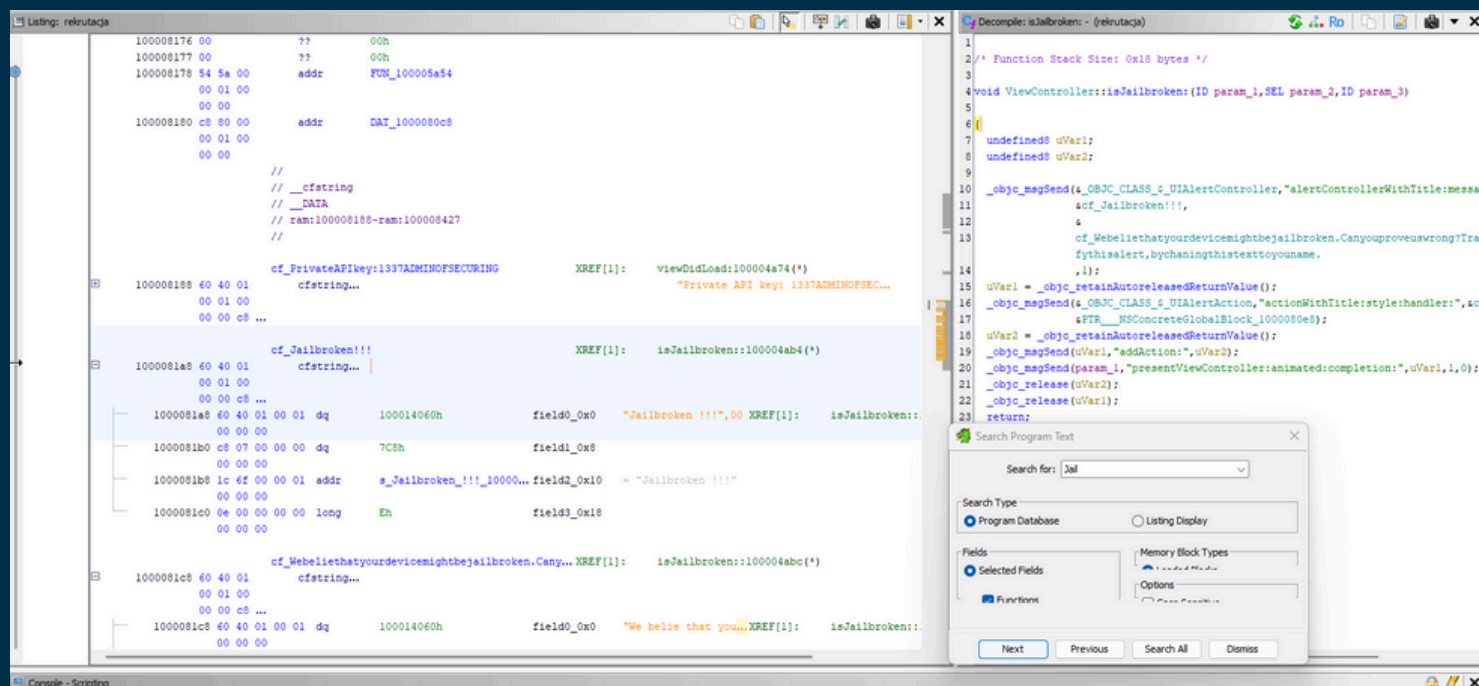
Ghidra - narzędzie do inżynierii wstecznej, dzięki któremu w prosty sposób można przejrzeć kod aplikacji i zrozumieć jego działanie.

Frida - bardzo rozbudowane narzędzie przydatne do analizy aplikacji.

Grapefruit - narzędzie bazujące na Fridzie, aczkolwiek podane w wygodniejszej formie aplikacji webowej z przyjemnym interfejsem. Jest przydatne np. do odczytywania keychain.

2. Ominięcie Jailbreak-Check

Rozwiązałem to zadanie za pomocą narzędzia Ghidra, gdzie na początku udało mi się zlokalizować funkcję odpowiedzialną za Jailbreak-check, korzystając z narzędzia wyszukiwania.



Następnie znalazłem finalne miejsce w kodzie, gdzie pojawiał się tekst wyświetlany po kliknięciu "Jailbreak check bypass", i zamieniłem go na własny tekst zawierający moje imię.

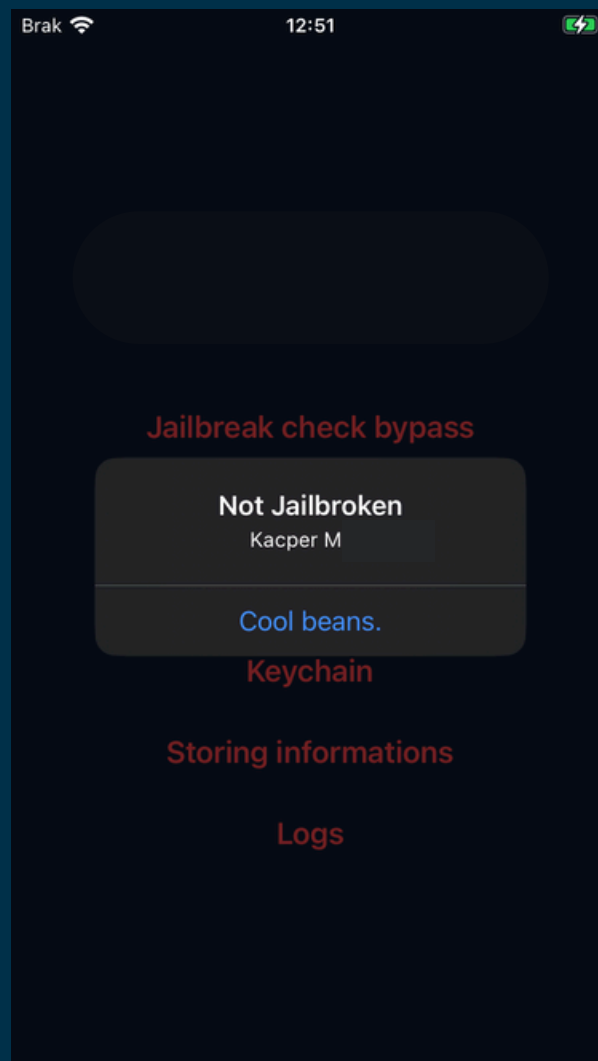
```
s_Kacper_M _100006f2b XREF[1,1]: 1000081b8 (*), 1000081d8 (*)
s_Not_Jailbroken_Kacper_M 100006f1c
100006f1c 4e 6f 74 ds "Not Jailbroken Kacper M "
20 4a 61
69 6c 62 ...
```

Po zatwierdzeniu zmian spatchowałem plik za pomocą skryptu SavePatch, pobranego ze strony: https://github.com/schlafwandler/ghidra_SavePatch.

Ostatnim zadaniem było przerzucenie nowej binarki do aplikacji za pomocą SSH. Plik musiałem umieścić w tym samym miejscu, gdzie znajdował się stary, zmienić jego nazwę oraz nadać mu uprawnienia do wykonywania.

```
(kali@kali)~[~/Downloads]
$ ssh root@192.168.0.215
(root@192.168.0.215) Password for root@iPhone:
iPhone:~ root# mv rekrutacja-new /Applications/rekrutacja.app
iPhone:~ root# cd /Applications/rekrutacja.app
iPhone:/Applications/rekrutacja.app root# ls
Assets.car Base.lproj/ Info.plist PkgInfo To\ Jay/ _CodeSignature/ embedded.mobileprovision rekrutacja* rekrutacja-new
iPhone:/Applications/rekrutacja.app root# mv rekrutacja rekrutacja-old
iPhone:/Applications/rekrutacja.app root# mv rekrutacja-new rekrutacja
iPhone:/Applications/rekrutacja.app root# chmod +x rekrutacja
iPhone:/Applications/rekrutacja.app root# ls
Assets.car Base.lproj/ Info.plist PkgInfo To\ Jay/ _CodeSignature/ embedded.mobileprovision rekrutacja* rekrutacja-old*
iPhone:/Applications/rekrutacja.app root# uicache -a
iPhone:/Applications/rekrutacja.app root#
```

Efekt końcowy:



3. Analiza zawartości paczki

Do pobrania paczki użyłem Fridy oraz dodatku napisanego przy użyciu Pythona i JavaScriptu. Dodatek ten można znaleźć pod linkiem: <https://github.com/AloneMonkey/frida-ios-dump>. Na początku Frida pozwoliła mi wyświetlić pełny identyfikator aplikacji, niezbędny w następnym kroku.

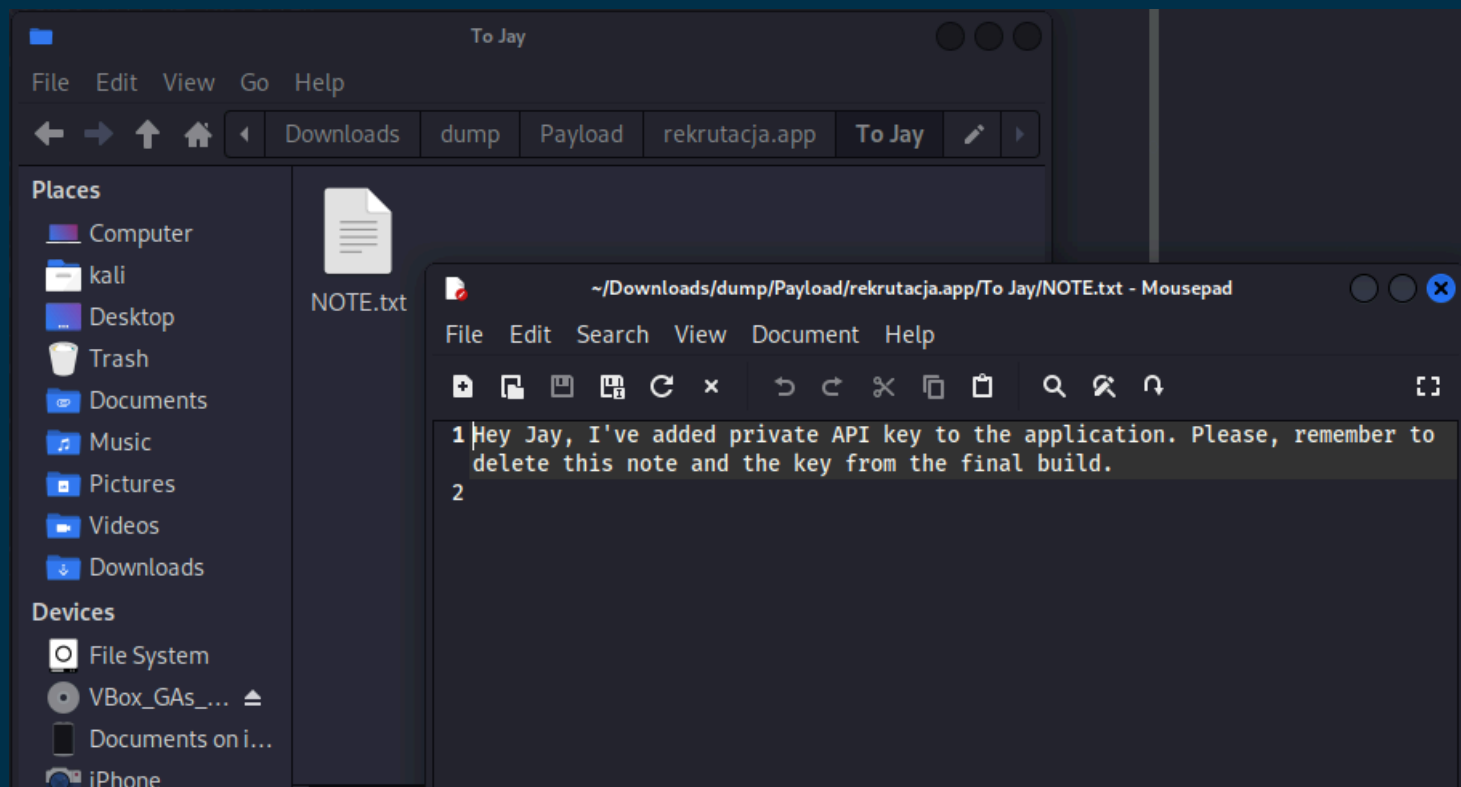
```
(kali@kali)-[~/Downloads/dump]
$ frida-ps -Uai
PID      Name           Identifier
-----
4190     Aparat         com.apple.camera
6110     App Store      com.apple.AppStore
13115    Cydia          com.saurik.Cydia
7111     DVIA-v2       com.hackititutehacks.DVIAswiftv2
9444     Dom            com.apple.Home
4257     Filza         com.tigisoftware.Filza
6123     Giełda        com.apple.stocks
12353    Kalkulator     com.apple.calculator
4252     Mapy          com.apple.Maps
4155     NewTerm       ws.hbang.Terminal
13114    Podcasty      com.apple.podcasts
6154     Porady        com.apple.tips
12355    Safari        com.apple.mobilesafari
7040     Sileo         org.coolstar.SileoStore
4220     Telefon       com.apple.mobilephone
4213     Ustawienia   com.apple.Preferences
9353     Watch         com.apple.Bridge
9922     Wiadomości    com.apple.MobileSMS
9938     Zdjęcia      com.apple.mobileslideshow
4254     Zegar        com.apple.mobiletimer
```

Następnie użyłem komendy iproxy 2222 22 do przekierowania SSH przez USB oraz wykonałem końcową komendę która pobrała paczkę na moją maszynę.

```
(kali@kali)-[~/Downloads/dump]
$ python3 dump.py .rekrutacja
Start the target app securig.rekrutacja
Dumping rekrutacja to /tmp
start dump /Applications/rekrutacja.app/rekrutacja
rekrutacja.fid: 100% [70.7k/70.7k [00:00<00:00, 872kB/s]
embedded.mobileprovision: 212kB [00:00, 354kB/s]
0.00B [00:00, ?B/s]Generating "rekrutacja.ipa"

(kali@kali)-[~/Downloads/dump]
$ iproxy 2222 22
Creating listening port 2222 for device port 22
waiting for connection
New connection for 2222→22, fd = 5
waiting for connection
Requesting connection to USB device handle 1 (serial:
[]
```

Informacjami, które nie powinny się znaleźć w paczce, jest przede wszystkim informacja o tym, że prywatny klucz API został "ukryty" w kodzie oraz samo to, że ten klucz wciąż się tam znajduje.

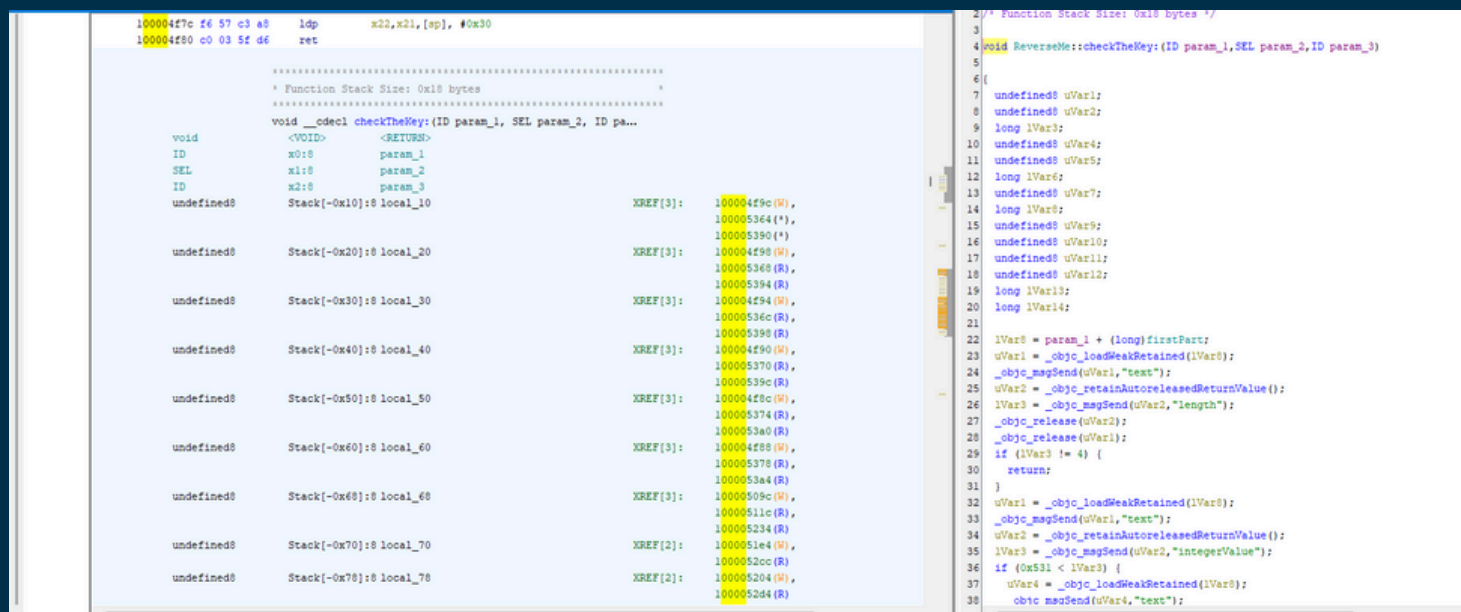


Sam klucz znalazłem z pomocą Ghidry.

```
100006ef7 50 72 69      ds      "Private API key: 1337ADMINOFSECURING"  
       76 61 74  
       65 20 41 ...
```

4. Inżynieria wsteczna

Oczywiście, do inżynierii wstecznej także użyłem Ghidry, gdzie udało się znaleźć całą funkcję odpowiedzialną za sprawdzanie poprawności klucza.



The screenshot displays the Ghidra decompiler interface. On the left, the assembly view shows instructions at addresses 100004f7c and 100004f80. The main window shows the decompiled C code for a function named `checkTheKey`. The function signature is `void __cdecl checkTheKey(ID param_1, SEL param_2, ID param_3)`. The code includes several stack frame declarations and a series of `XREF` (cross-reference) entries pointing to various memory locations. The right-hand pane shows the C code for a function named `ReverseMer::checkTheKey`, which is a more detailed decompiled version of the assembly shown on the left. It includes variable declarations for `uVar1` through `uVar14` and `lVar3`, and a series of operations involving `_objc_loadWeakRetained`, `_objc_msgSend`, and `_objc_release`.

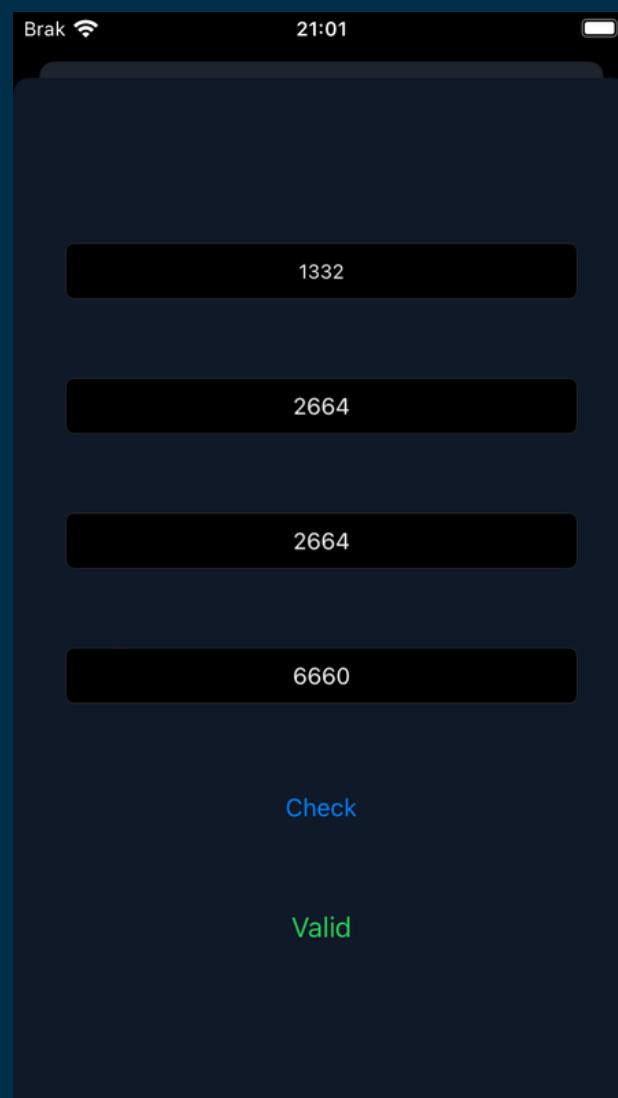
Nie ukrywam, że w analizie kodu wspierałem się pomocą ChatGPT i udało mi się dowiedzieć, że pierwsza liczba musi zawierać się w przedziale od 1329 do 1340. Następna liczba jest dwukrotnie większa od poprzedniej, trzecia liczba jest taka sama jak druga, a czwarta jest sumą poprzednich cyfr.

Poniżej znajduje się kod napisany w Pythonie, który generuje przykładowe klucze spełniające kryteria, oraz zdjęcie, na którym aplikacja potwierdza poprawność wpisanych liczb.



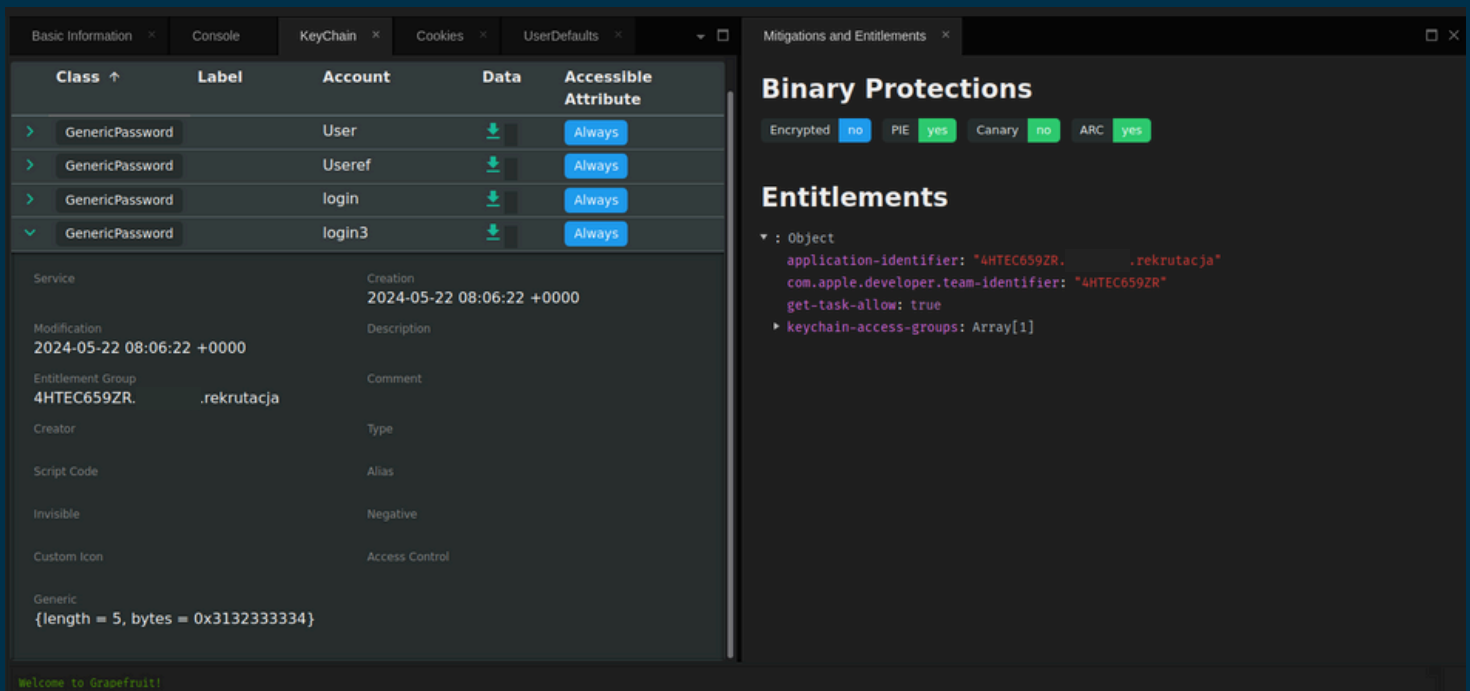
generator_key.py

```
import random
def generate_key():
    first = random.randint(1329, 1340)
    second = first * 2
    third = second
    fourth = first + second + third
    key = f"{first:04d}-{second:04d}-{third:04d}-{fourth:04d}"
    return key
print(generate_key())
```

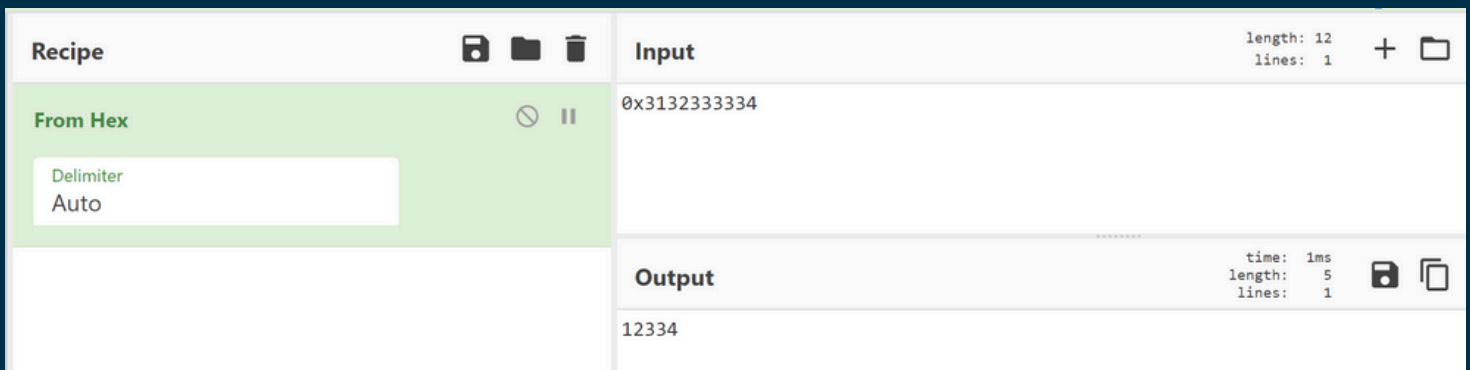


5. Keychain

W tym zadaniu postanowiłem użyć narzędzia Grapefruit, które bazuje na Fridzie i charakteryzuje się przyjaznym graficznym interfejsem dostępnym z poziomu przeglądarki. Niestety okazało się, że dostęp do Keychaina jest bardzo łatwy, co pozwala dokładnie zobaczyć, jaki login użytkownika został wprowadzony.



Niestety, co gorsza, równie łatwy dostęp jest do hasła, które nie jest w żaden sposób zaszyfrowane i znajduje się po prostu w formie heksadecymalnej.



6. Przechowywanie wrażliwych danych.

Przeglądając funkcję odpowiedzialną za zapisywanie hasła i loginu do Keychaina odkryłem również, że te dane są zapisywane do NSUserDefaults.

```
25 _objc_msgSend(uVar3,"text");
26 uVar5 = _objc_retainAutoreleasedReturnValue();
27 _objc_release(uVar3);
28 _objc_msgSend(uVar4,"dataUsingEncoding:",4);
29 uVar3 = _objc_retainAutoreleasedReturnValue();
30 _objc_msgSend(uVar5,"dataUsingEncoding:",4);
31 uVar6 = _objc_retainAutoreleasedReturnValue();
32 _objc_msgSend(uVar2,"setObject:forKey:",uVar6,*(undefined8 *)PTR_kSecAttrGen
33 _objc_msgSend(uVar2,"setObject:forKey:",uVar3,*(undefined8 *)PTR_kSecAttrAcc
34 _objc_msgSend(uVar2,"setObject:forKey:",*(undefined8 *)PTR_kSecAttrAccessibl
35 *(undefined8 *)PTR_kSecAttrAccessible_100008008);
36 iVar1 = _SecItemAdd(uVar2,0);
37 if (iVar1 != 0) {
38     NSLog(&cf_Errorwhileaddingkeytokeychain);
39 }
40 _objc_msgSend(&_OBJC_CLASS_$_NSUserDefaults,"standardUserDefaults");
41 uVar7 = _objc_retainAutoreleasedReturnValue();
42 _objc_msgSend(uVar7,"setObject:forKey:",uVar4,&cf_User);
43 _objc_msgSend(uVar7,"synchronize");
44 _objc_msgSend(uVar7,"setObject:forKey:",uVar5,&cf_Password);
45 _objc_msgSend(uVar7,"synchronize");
46 _objc_msgSend(&_OBJC_CLASS_$_UIAlertController,"alertControllerWithTitle:mes
47 &cf_Correct!,&cf_Credentialshavebeensaved.,1);
```

NSUserDefaults można bardzo łatwo odczytać dzięki narzędziu Objection:

```
.rekrutacja on (iPhone: 16.0) [usb] # ios nsuserdefaults get
{
  AKLastIDMSEnvironment = 0;
  AddingEmojiKeyboardHandled = 1;
  AppleLanguages = (
    "pl-PL"
  );
  AppleLanguagesDidMigrate = 20A362;
  AppleLanguagesSchemaVersion = 3000;
  AppleLocale = "pl_PL";
  ApplePasscodeKeyboards = (
    "en_US",
    emoji
  );
  INNNextFreshmintRefreshDateKey = "737820010.191065";
  NSInterfaceStyle = macintosh;
  NSLanguages = (
    "pl-PL",
    en
  );
  PKKeychainVersionKey = 8;
  PKLogNotificationServiceResponsesKey = 0;
  Password = password;
  User = test;
  "com.apple.content-rating.AppRating" = 1000;
  "com.apple.content-rating.ExplicitBooksAllowed" = 1;
  "com.apple.content-rating.ExplicitMusicPodcastsAllowed" = 1;
  "com.apple.content-rating.MovieRating" = 1000;
  "com.apple.content-rating.TVShowRating" = 1000;
}
.rekrutacja on (iPhone: 16.0) [usb] #
```

Informacje te zostały również zapisane w pliku securing.rekrutacja.plist, który znajduje się w plikach aplikacji, a dokładniej w lokalizacji:

/var/mobile/Containers/Data/Application/1D7303A8-764D-4D23-8E82-602AE5C9FD98/Library/Preferences.

```
/var/mobile/Containers/Data/Application/1D7303A8-764D-4D23-8E82-602AE5C9FD98/Library/Preferences
.rekrutacja on (iPhone: 16.0) [usb] # ls
NSFileType  Perms  NSFileProtection  Read  Write  Owner  Group  Size  Creation  Name
Regular     384    CompleteUntilFirstUserAuthentication  True  True  mobile (501)  mobile (501)  78.0 B  2024-05-23 10:36:01 +0000  .rekrutacja.plist

Readable: True Writable: True
.rekrutacja on (iPhone: 16.0) [usb] # ios plist cat .rekrutacja.plist
{
  Password = password;
  User = test;
}
.rekrutacja on (iPhone: 16.0) [usb] #
```

Przechowywanie danych w taki sposób jest bardzo niebezpieczne. Łatwy dostęp i przede wszystkim brak jakiegokolwiek szyfrowania sprawiają, że wrażliwe dane są narażone na pozyskanie przez niepowołane osoby.

7. Logi

Logi z zapisu wrażliwych danych odczytałem za pomocą narzędzia Xcode, dostępnego na macOS, gdzie miałem dostęp do logów podpętego urządzenia na żywo. W tych logach można zobaczyć login oraz hasło w formie plaintextu.

The screenshot shows the Xcode console with 30 messages. The logs are filtered by 'All Messages'. The process name is 'rekrutacja'. The logs show a sequence of events, including UI events, window management, and system errors. A key log entry is highlighted in blue: 'Credentials saved: test : password123'.

Type	Time	Process	Message
	09:00:21.886681+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.886738+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.887888+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.902765+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.902822+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.902878+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.919758+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.919806+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.919855+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.936681+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.936681+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.936741+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.950429+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 0; ignoreInteractionEvent: 0
	09:00:21.952604+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.952661+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.952718+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.987301+0200	rekrutacja	Evaluating dispatch of UIEvent: 0x281d0c000; type: 0; subtype: 0; backing type: 11; shouldSend: 1; ignoreInteractionEvent: 0
	09:00:21.987356+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to windows: 1
	09:00:21.987577+0200	rekrutacja	Sending UIEvent type: 0; subtype: 0; to window: <UIWindow: 0x102e09e60>; contextId: 0xCAD053A5
	09:00:21.998256+0200	rekrutacja	Error while adding key to keychain
	09:00:22.019166+0200	rekrutacja	<UIWindowScene: 0x102e082e0> (22B34BF2-5B79-43FA-AB3D-ECE1C2D9B10F) Scene updated orientation preferences: (Pu Ll Lr)
	09:00:22.022005+0200	rekrutacja	Credentials saved: test : password123
	09:00:22.022981+0200	rekrutacja	Override focusSystemState: (ENABLED) for reason(s): ({ " <UIAlertControllerPhoneTVMacView 0x10a819600" })
	09:00:22.036760+0200	rekrutacja	_willShowAlertController: <UIAlertController: 0x10a819600>
	09:00:22.037090+0200	rekrutacja	_addAlertControllerToStack: Adding Alert to stack : <UIAlertController: 0x10a819600>
	09:00:22.037359+0200	rekrutacja	Reloading input views for: <UIAlertController: 0x10a819600> force: 0

rekrutacja
Subsystem: -- Category: <Missing Description> Details

Credentials saved: test : password123