



## Mission Name

TheFencesDataCard

## History Background

Claire and Ethar are in Asuncion (Paraguay) with she (The Fence). Claire must analyze The Fence's datacard in order to get any relationship among The Fence and SHAX community.

## Technical High-Level Overview

The Fence's Datacard is provided as an iPhone evidence. The aim is to analyse in depth to locate any clue that allow Claire to identify SHAX community. There is an email from SHAX community located in Protonmail App.

## Short Description

You're going to analyse The Fence's Datacard based on an iPhone evidence. Your goal is to locate any clue related to SHAX, if you find it, please give us the SHAX Contact.

## Mission Description

The Fence's Datacard is provided based on an iPhone evidence. The aim is to analyse in depth to locate any clue that allow Claire to identify SHAX community. Your goal is to locate any clue related to SHAX, if you find it, please give us the SHAX contact.

## Location

ASUNCION | PARAGUAY



## Tools

- 7Z
- iLEAPP - <https://github.com/abrignoni/iLEAPP>
- SQLite Studio
- CyberChef

## Questions

How many emails contains ProtonMail App?

- 2

Which day of the week was installed ProtonMail App?

- Wednesday

## Hints

1. Use iLEAPP to find any strange App installed.
2. Analyse ProtonMail App.
3. ProtonMail SQLITE could be tampered and using a XOR encryption.

## Write Up

First of all, player must unzip Fence's data card. In this scenario, player is going to face an iPhone evidence. Considering that there are a lot of pieces of information from Apps, and iOS, this challenge is more complicated than others. The first step, would be to use iLEAPP tool to analyse which apps are installed on the Iphone:

```
jmma@demowindows:/mnt/c/THREATIA/Iphone/iLEAPP-1.9.4/iLEAPP-1.9.4$ python3 ileapp.py
usage: ileapp.py [-h] [-t {fs,tar,zip,gz,itunes}] [-o OUTPUT_PATH] [-i INPUT_PATH] [-p]
ileapp.py: error: No OUTPUT folder path provided
```

**Figure 1**

Player must launch the previous command using "-t fs" -o output\_path and -i path\_to\_extract evidence. Once iLEAPP finishes, an index.html is created. This file has all the necessary information:

iLEAPP 1.9.4					
<ul style="list-style-type: none"> <li>⌚ App Snapshots</li> <li>⌚ Application State DB</li> <li>⌚ Apps - Itunes Metadata</li> <li>⌚ Bundle ID by AppGroup &amp; PluginKit IDs</li>   <li>_KEYBOARD APPLICATION USAGE</li> <li>_KEYBOARD Dynamic LEXICON</li> <li>KNOWLEDGE</li> <li>App Activity</li> <li>App In Focus</li> <li>App Usage</li> <li>Application Activity</li> <li>Safari</li> <li>Battery Level</li> <li>Device is Backlit</li> <li>Device Locked</li> <li>Installed Apps</li> </ul>	Start	End	Bundle ID	App Category	App Name
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.Home		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.news		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.tv		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.facetime		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.Bridge		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.iBooks		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.stocks		
	2021-06-01 20:35:00	2021-06-01 20:35:00	com.apple.mobileme.fmf1		
	2021-06-02 20:33:00	2021-06-02 20:33:00	com.googleMaps	Navegación	Google Maps - tráfico y comida
	2021-06-02 20:37:00	2021-06-02 20:37:00	com.google.chrome.ios	Utilidades	Google Chrome
	2021-06-02 20:53:00	2021-06-02 20:53:00	ch.protonmail.protonmail	Productividad	ProtonMail - Correo cifrado
	Start	End	Bundle ID	App Category	App Name

**Figure 2**

Above picture shows that Proton Mail is installed on the iPhone among other Apps. Last time that protonmail was focused :

	20:52:25	20:52:27		
T Keyboard Application Usage	2021-06-02 20:52:25	2021-06-02 20:52:27	com.apple.UpNextWidget.extension	2
KEYBOARD DYNAMIC LEXICON	2021-06-02 20:52:28	2021-06-02 20:52:30	com.apple.Preferences	2
T Keyboard Dynamic Lexicon	2021-06-02 20:52:32	2021-06-02 20:52:55	com.apple.AppStore	2
KNOWLEDGECL	2021-06-02 20:52:55	2021-06-02 20:53:01	com.apple.PassbookUIService	6
App Activity	2021-06-02 20:53:01	2021-06-02 20:53:29	com.apple.AppStore	2
App In Focus	2021-06-02 20:53:29	2021-06-02 20:56:21	ch.protonmail.protonmail	1
App Usage	2021-06-02 20:56:25	2021-06-02 20:57:25	ch.protonmail.protonmail	6
Application Activity	2021-06-02 20:57:51	2021-06-02 20:57:53	SBPowerDownController	2
Safari	2021-06-02 21:00:19	2021-06-02 21:00:29	SBPowerDownController	1
Battery Level	2021-06-02 21:21:04	2021-06-02 21:21:06	ch.protonmail.protonmail	2
Device is Backlit	2021-06-02 21:21:49	2021-06-02 21:21:53	com.apple.CoreAuthUI	4
Device Locked	2021-06-02 21:22:07	2021-06-02 21:22:07	com.apple.mobilemail.MailCacheDeleteExtension	0
Installed Apps				
Intents				
Plugged In				
Safari Browsing				
Siri Usage				
Web Usage				

Figure 3

Considering other apps and installed, and checking that proton App was the last application, it would be a good way to start the analysis. Player must locate folder which contains App protonmail:

- \private\var\mobile\Container\Shared\AppGroup\16A0E996-690B-4BEE-BE80-AE37A6C986AD\ProtonMail.sqlite

Next step, would be to open **ProtonMail.sqlite** using SQLite Studio and locate the following table:

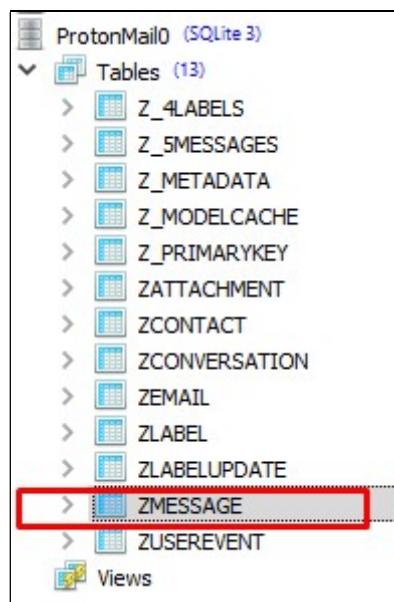


Figure 4

Then locate ZSENDER column:

ZSENDER	ZTITLE
AZSJ2Q]\FSQF## # +*'&&#	♦N♦(3♦7♦A7♦!♦e.♦6♦'IN8♦♦♦♦N%L^{F♦♦♦♦noh♦T♦♦♦♦t♦♦♦♦b♦♦♦♦X♦♦♦♦
AZSJ2Q]\FSQF## # +*'&&#	♦n♦♦♦♦♦L♦♦♦♦4I-B♦o)^\t♦♦♦♦d♦Gb♦R♦p♦vZ♦♦♦♦/♦♦♦♦H♦♦♦♦T♦♦♦♦]♦♦♦♦j♦M♦♦♦♦

Figure 5

AZSJ2Q]\FSQF## # +\*'&&#

Player must try to decrypt this information applying his best efforts. This piece of information it's a XOR encryption.



The key it's unknown, so player has to launch a little brute force. To accomplish this way, player could use CyberChef. Player doesn't know anything about the key, so player will go step by step, increasing until key 12:

The screenshot shows the CyberChef interface with the following configuration:

- Recipe:** XOR Brute Force
- Input:** AZSJ2Q]\FSQF(## # +\*'&&
- Key length:** 12 (highlighted with a red box)
- Sample length:** 100
- Sample offset:** 0
- Scheme:** Standard
- Output Options:** Null preserving (unchecked), Print key (checked), Output as hex (unchecked)
- Crib (known plaintext string):** (empty)

The **Output** section shows the results of the attack:

- time: 570ms
- length: 2424794 bytes
- Size: 2.424.794 bytes
- DOWNLOAD (button highlighted with a red box)
- SHOW ALL
- Search bar: 0 to 256 KiB

Figure 6



Player must download created file with all possibilities:

Key = 1207: SJAM VOLVICA.41 I J J .44
Key = 1208: SRAB YOTT [CN:+1(1(9"5.4+
Key = 1209: SSAC XOUTZCO:*1)1)9#5/4*
Key = 120a: SPA@ [OVTYCL:)1*1*9 5,4)
Key = 120b: SQAA ZOWTXCM: (1+1+9!5-4 (
Key = 120c: SVAF ]OPT_CJ:/1,1,9&5*4/
Key = 120d: SWAG \OQT^CK:.1-1-9'5+4.
Key = 120e: STAD _ORT]CH:-1.1.9\$5(4-
Key = 120f: SUAE ^OST\CI:,1/1/9%5)4,
Key = 1210: SJAZ AOLTCCV:310109:5643
Key = 1211: SKAT @OMTBCW:211119:5742
Key = 1212: SHAX CONTACT:11212985441
Key = 1213: SIAY BOOT@CU:01313995540
Key = 1214: SNA^ EOHTGCR:714149>5247
Key = 1215: SOA_ DOITFCS:615159?5346
Key = 1216: SLA\ GOJTECP:516169<5045
Key = 1217: SMA] FOKTDCQ:417179=5144
Key = 1218: SBAR IODTKC^:;1818925>4;
Key = 1219: SCAS HOETJC_::1919935?4:

Figure 7

Finally, the SHAX CONTACT is located:

**11212985441**

## Flag Information

flag{11212985441}