# Preliminary Security Analysis Of a Mobile Application

## Performed on the 'eToro' app

Jeppe Stjernholm Schildt
Network Security Final Project
*Aarhus University*

***Abstract*—This paper presents the preliminary security analysis of eToro, an investment and trading platform. First, a threat analysis was performed, in which the adversaries and key policies were identified. Subsequently, the app was subjected to a security and network analysis, and while a few potential small issues were found such as allowing backup and associated entities having outdated algorithms implemented, the app proved extremely resilient and secure. The app includes possible intrusive trackers, mainly regarding analytics and marketing, however these are widely used globally and seemingly secure.**
**Lastly, the authentication measures were examined, and they included a large amount of great methods, both optional, and mandatory to conform to regulations.**

## I. Introduction

Investment is seen by many as something done by elderly men at Wall-street, or at least through your local bank, however with the expansion of internet capabilities, the investment opportunities have become much more accessible, and one of the platforms helping this, is the eToro trading platform. It provides opportunities to buy and sell fractional stocks, resources, crypto coins, to copy popular investors, and all the other margin and option possibilities you'd associate with any brokerage.
This report will firstly present a threat model of the application, depicting expected adversaries, attack surfaces, and listing policies which are essential for this application. Secondly, the system security will be analyzed, through different methods such as manual code review as well as static analyzers, presenting key security-relevant metrics such as encryption and certificate types, crypto implementations and dependencies, and highlight any issues found.
Subsequently, the network will be analyzed, including dependent servers, captured traffic, and man-in-the-middle attacks will be attempted. Lastly, the privacy and authentication measures of the application is examined, including app-permissions & possible trackers. The complete security state of eToro will be discussed at the end, with ideas for improvements.

## II. Threat Analysis

The following section will present the threat model imagined for teToro.

### *A. Adversaries & Attack Surfaces*

The following are the adversaries belived to have interest in a vulnerable version of eToro.

- **Customers:** Likely to utilize vulnerabilities allowing them to buy at non-updated market rates, increasing portfolio size or withdrawing non-existent funds.
- **Criminals:** Since the application handles sensitive information like account numbers, crypto addresses and such, criminals could steal these for personal gain.
- **Insiders:** Corrupt personnel could increase profit margins based upon portfolio value, to discretely pocket the difference.
- **Competitors:** If eToro's services were to have decreased availability due to denial of service attacks, user's would be likely to migrate to competitors.

With any apps, especially ones as large as eToro, the attack surface is massive, however the main attack points would likely be the UI/Interfaces, API's and data storage, such as databases or local storage. There's likely a very large amount of sensitive information being passed around the app and its associated entities, and as such, security to protect privacy is key.

### *B. Properties*

The following are the properties believed to be of high importance for a financial app such as eToro.

- **Confidentiality:** It is essential to keep sensitive information private. This includes associated bank numbers, portfolio size and content, planned purchases and more.
- **Integrity:** With the amount of transfers and financial activity existing on a brokerage platform, the integrity of each trade is essential, but also the integrity of the prices provided by the platform
- **Authenticity & Non-repudiation:** On platforms with as much sensitive information as eToro, this is a key policy. Authenticity and non-repudiation must be guaranteed. This is especially important, as option-trading (which is possible) can bankrupt individuals in seconds.
- **Accountability :** On the same note as above, it's essential to have a clear audit trail, to ascertain who's responsible for specific actions

- **Availability :** As eToro has nearly 3 million accounts with money deposited [6], and markets being real-time, having a low availability could lead to significant losses, especially for day-traders

## III. Software Security

The following section will present the methods performed to analyze the software security of the eToro. It's expected to find rigorous encryption using state-of-the-art methods and implementations,

### A. Manual Decompilation

Using an APK Downloader [4], the Dex2Jar [3] tool, and JD-GUI [5], the de-compiled class files were examined manually. As expected, the application had dedicated a large amount of resources into their cryptography. This is seen by the hundreds of classes corresponding to encryption. These weren't obfuscated to the point of not being readable, but all functions and variables had extremely generic names. This can be seen below:

```java
public static String decryptPbkdf2(String paramString1,
       String paramString2) throws .. {
   String[] arrayOfString = paramString1.split(DELIMITER);
   if (arrayOfString.length == 3) {
     byte[] arrayOfByte1 = fromBase64(arrayOfString[0]);
     byte[] arrayOfByte2 = fromBase64(arrayOfString[1]);
     return decrypt(fromBase64(arrayOfString[2]),
          deriveKeyPbkdf2(arrayOfByte1, paramString2),
          arrayOfByte2);
   }
   throw new IllegalArgumentException("Invalid␣encypted␣
        text␣format");
}
```

Listing 1: 'Excert of 'Crypto' class

Throughout the application, the java.security framework [7] is utilized along side the Bouncy Castle API collection [8]. The Bouncy Castle does have some deprecated features [9], however these do not seem to be used anywhere in the eToro source code. No privatekey's were hard-coded anywhere either. Key pairs are generated by java.security's SecretKeyFactory using 'PBKDF2WithHmacSHA1'. Padding's done using PKCS5 with salt length 8, with is within recommendations. The source code includes multiple encryption algorithm implementations, but the main one seems to be AES/CBC. With an app as large and complex, it's difficult to guarantee anything based solely on de-compiled code, but the CBC algorithm does not seem used together with any resemblance of an oracle.
The application is setup to utilize 4 different signatures, namely RSA, DSA, ECDSA, or Anoynmous. This does not mean that all implementations are necessarily used, but RSA (2048-bit key length) definitly is. There's also found support for FIPS compliant algorithms, further proving the very deliberate focus on security on this application.
The manual review gave a good idea regarding the structure and size of the application, and due to it's size, static analyzers were deemed necessary.

### B. Static Analysis

Using the static analysis tool MobSF [1], the APK was examined. Below, the scorecard generated is found:

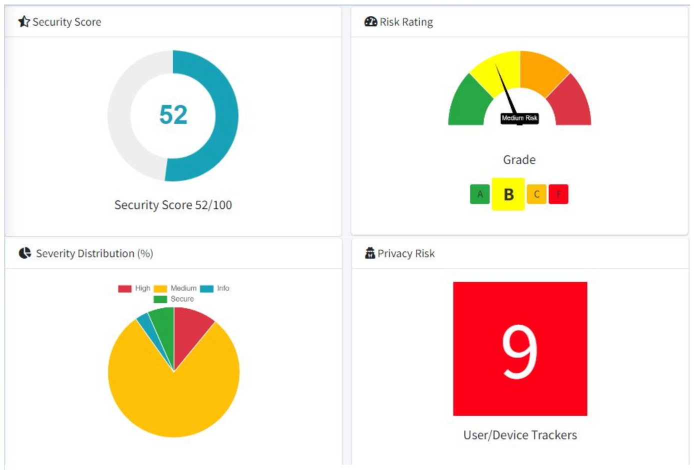


Fig. 1: MobSF Scorecard

The following critical warning points were provided by the analysis:

- Clear Text Traffic Enabled
- Privacy Trackers

The Clear Text Traffic is by default enabled for apps targeting API 27 or lower. This is most likely the reasoning behind it being enabled, however it lays part of foundational need for the Network analysis, through which it'll be examined further.
The Privacy trackers will be examined in the Privacy section.
Another tool was used for static analysis, namely ImmuniWeb [17]. Using their free tools, a 92 page long summary was provided. The following is the scorecard:

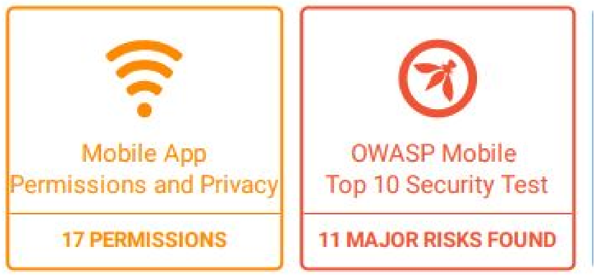


Fig. 2: ImmuniWeb Scorecard

This mentions the OSWAP Top 10, which is a listing of the most critical mobile application security risks [18]. The Static Analyzer found 5 'High Risk', 5 'Medium Risk, and 5 'Low Risk'. The Low Risk warnings have been examined, and deemed not noteworthy for this report.
The rest of the risks have been examined, and while there was some that was unable to be found (summary report pointing at files that seemingly didn't exist), the rest was investigated

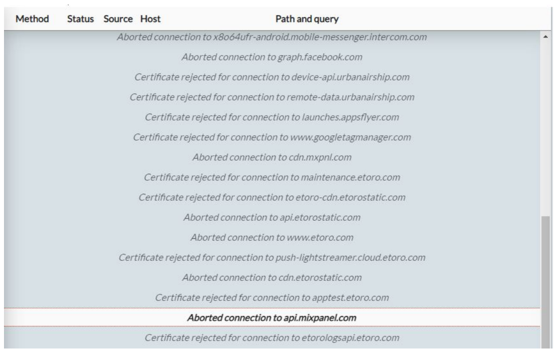
and the findings have been presented:
**High Risk : Exposure of potentially sensitive data** - this led at several Javascript files, all obfuscated to be nearly unreadable. These were de-obfuscated using jsnice.org, and the issues were examined. Mostly, the analyzer seems to just flag usages of the word 'username', or '@' as potential information exposures. Some major JS files handling logins/logouts and deposits were examined using SonarLint and JSHint [19], which, amongst fixing syntax and faulty logic, attempts to find security issues, however none were found.
**High Risk : External Data in SQL Queries** This led to a list of obfuscated smali code, which was decompiled back into java using jadx and the vscode 'Smali2Java' extension. The summary flags the function 'execSQL()' as a potential issue as input is likely raw and can be vulnerable to SQL Injections. Upon inspection, these SQL statements does however not seem to be anywhere connected to user inputs, and is deep within automated processes handling automatic data management.
**High Risk : Weak Hashing Algorithms** Implementations of different hash functions are found,specifically in the 'HashUtils' java file, and while the cryptographically secure SHA-256 is implemented, so is the insecure SHA-1, and MD5. This Utility file however is found under '/com/appsflyer', which is a platform dealing with analytics and marketing. AppsFlyer themselves claim to to have security be *'an essential part of our product inside and out'* [13], which makes you wonder why they have insecure hashing implementations. CVE, a database containing publicly disclosed security vulnerability, also assign relatively low scores (CVSS 5.4) [10] to Appsflyer. With an application as security oriented, as eToro, perhaps making sure partners security is top of the line, would be beneficial.
**Medium Risk : Enabled Application Backup** In the AndroidManifest.xml file, a potential issue is also found. The app has the AllowBackup flag set to true, which could allow adversaries to either access personal informatiom, or inject information which is then read during restorations. The data was first attempt accessed (/data/data/com.etoro.openbook) using ADB (Android Debug Bridge) [14] and Android Studios device file manager, however this was not possible due to the eToro package not being debuggable. In an attempt to circumvent this, the Genymotion emulator, which provides rooted devices, was used, using the Genymotion ARM Translation tool [15], however no matter the emulator chosen or the API level, the eToro APK would successfully install, request an 'Automatic Update', then be incompatible with the system. This was seemingly due to an eToro update being released the 7th of December, as the emulator worked before then. Using the Genymotion (rooted) emulator, eToro would not allow logins (generic error message), which in hindsight is likely due to their root detection.
I was not able to circumvent this issue using emulators, and the physical device used was not to be rooted. This would however be an obvious starting point if the security analysis was to be expanded.

## IV. Network Security

This section will examine the network security properties of the eToro app. Firstly, the entire source code was searched for API's, and 19 different API endpoints were found. Of these endpoints, 17 where reachable, and a Qualys SSL Test was performed on them. Below, the results have been shown. Multiple endpoints responding to the same API has been removed for clarity (e.g 'api.dev1.idnow.de', 'api.dev2.idnow.de').
Any non A+ grade seems to be due to the missing DNS CAA, which is mandated by the CA/Browser Forum [23]

| URL (Https) | Qualys SSL Grade |
|---|---|
| api.idnow.de | A+ |
| api.online-ident.ch | A+ |
| combine.asnapieu.com | A |
| device-api.asnapieu.com | A |
| device-api.urbanairship.com | A |
| remote-data.asnapieu.com | A |
| wallet-api.urbanairship.com | A |
| wallet-api.asnapieu.com | A |

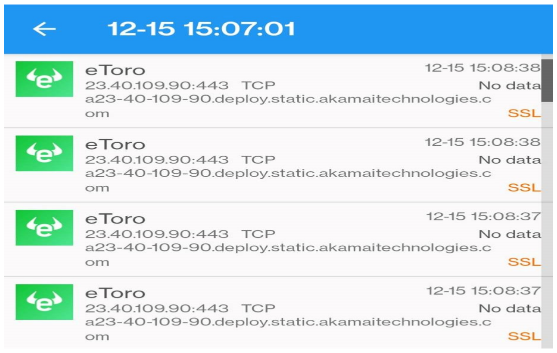
Subsequently, the web traffic was attempt-captured by the HTTP Tool kit, using their CA certificate. This certificate was however rejected by every single recipient as shown below:



```
Method   Status   Source   Host            Path and query
Aborted connection to x8o64ufr-android.mobile-messenger.intercom.com
Aborted connection to graph.facebook.com
Certificate rejected for connection to device-api.urbanairship.com
Certificate rejected for connection to remote-data.urbanairship.com
Certificate rejected for connection to launches.appsflyer.com
Certificate rejected for connection to www.googletagmanager.com
Aborted connection to cdn.mxpnl.com
Certificate rejected for connection to maintenance.etoro.com
Certificate rejected for connection to etoro-cdn.etorostatic.com
Aborted connection to api.etorostatic.com
Aborted connection to www.etoro.com
Certificate rejected for connection to push-lightstreamer.cloud.etoro.com
Aborted connection to cdn.etorostatic.com
Certificate rejected for connection to apptest.etoro.com
Aborted connection to api.mixpanel.com
Certificate rejected for connection to etorologsapi.etoro.com
```

Fig. 3: Rejected Connections

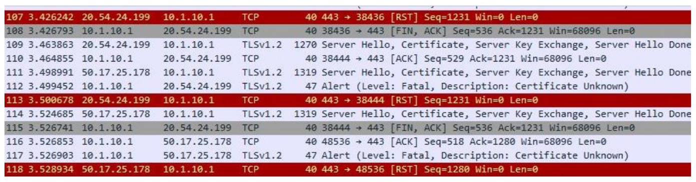
The ways to (attempt to) circumvent this, would be to utilize a rooted device, or emulator and inject a system certificate. As mentioned, due to the issues with Emulators, which existed on 2 different computers, this was not possible. Another attempt to capture web traffic was performed using the Packet Capture app by Grey Shirts [12], directly on the phone. This intercepts network traffic to analyze it, however while utilizing it, all packets were empty. This is likely due to the certificate being refused.



```
←  12-15 15:07:01
eToro                                        12-15 15:08:38
23.40.109.90:443  TCP                               No data
a23-40-109-90.deploy.static.akamaitechnologies.c
om                                                      SSL
eToro                                        12-15 15:08:38
23.40.109.90:443  TCP                               No data
a23-40-109-90.deploy.static.akamaitechnologies.c
om                                                      SSL
eToro                                        12-15 15:08:37
23.40.109.90:443  TCP                               No data
a23-40-109-90.deploy.static.akamaitechnologies.c
om                                                      SSL
eToro                                        12-15 15:08:37
23.40.109.90:443  TCP                               No data
a23-40-109-90.deploy.static.akamaitechnologies.c
om                                                      SSL
```

Fig. 4: Packet Capture

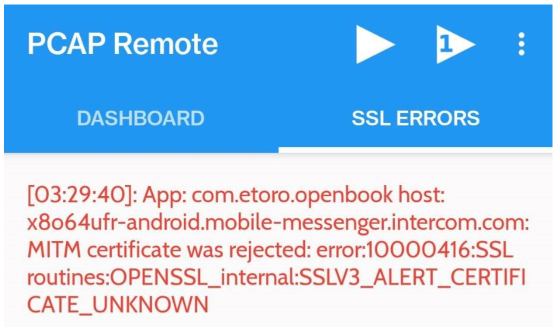
Attempts to update or install another certificate was made, however due to Android disallowing user or admin-added CA's by default, after target API 24, this was futile. Another attempt to capture network traffic was done using PCAP Remote, which features MITM functionality to allow Wireshark analysis. eToro's network traffic was captured during the login-flow, saved as a pcap file and examined using Wireshark. While capturing network traffic, eToro would not allow logins, providing 'Something went wrong. Try again' errors.
When looking at the packets, the issue is quite clear:



```
107 3.426242 20.54.24.199  10.1.10.1     TCP       40 443 → 38436 [RST] Seq=1231 Win=0 Len=0
108 3.426793 10.1.10.1     20.54.24.199  TCP       40 38436 → 443 [FIN, ACK] Seq=536 Ack=1231 Win=68096 Len=0
109 3.463863 20.54.24.199  10.1.10.1     TLSv1.2 1270 Server Hello, Certificate, Server Key Exchange, Server Hello Done
110 3.464855 10.1.10.1     20.54.24.199  TCP       40 38444 → 443 [ACK] Seq=529 Ack=1231 Win=68096 Len=0
111 3.498991 50.17.25.178  10.1.10.1     TLSv1.2 1319 Server Hello, Certificate, Server Key Exchange, Server Hello Done
112 3.499452 10.1.10.1     20.54.24.199  TLSv1.2   47 Alert (Level: Fatal, Description: Certificate Unknown)
113 3.500678 20.54.24.199  10.1.10.1     TCP       40 443 → 38444 [RST] Seq=1231 Win=0 Len=0
114 3.524685 50.17.25.178  10.1.10.1     TLSv1.2 1319 Server Hello, Certificate, Server Key Exchange, Server Hello Done
115 3.526741 10.1.10.1     20.54.24.199  TCP       40 38444 → 443 [FIN, ACK] Seq=536 Ack=1231 Win=68096 Len=0
116 3.526853 10.1.10.1     50.17.25.178  TCP       40 48536 → 443 [ACK] Seq=518 Ack=1280 Win=68096 Len=0
117 3.526903 10.1.10.1     50.17.25.178  TLSv1.2   47 Alert (Level: Fatal, Description: Certificate Unknown)
118 3.528934 50.17.25.178  10.1.10.1     TCP       40 443 → 48536 [RST] Seq=1280 Win=0 Len=0
```

Fig. 5: Wireshark excerpt

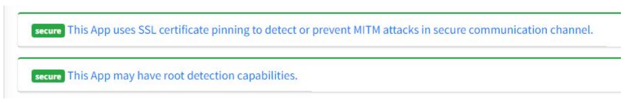
The server Requests the certificate, certificate is rejected due to being 'unknown', and a TCP Reset packet is delivered, terminating the login request. Closing PCAP, logging in normally, and then reopening PCAP to capture packets, didn't provide better results. The application became unresponsive, provided error messages when attempting to deposit, and claimed 'No funds available' when attempting to withdraw, even though there was enough eligible funds on the account. Looking at PCAP's SSL Erros, the following was seen:



[03:29:40]: App: com.etoro.openbook host: x8o64ufr-android.mobile-messenger.intercom.com: MITM certificate was rejected: error:10000416:SSL routines:OPENSSL_internal:SSLV3_ALERT_CERTIFICATE_UNKNOWN

Fig. 6: MITM Certificate rejected

This is likely due to the Certificate Pinning that the static analyzers found (seen below), or even just the Android certificate security settings.



secure This App uses SSL certificate pinning to detect or prevent MITM attacks in secure communication channel.

secure This App may have root detection capabilities.

Fig. 7: Security Measures against Fraudulent Certificates

In summary, the network security seems perfectly fine. eToro and all associated entities use TLSv1.2, and supports TSL 1.3. All API endpoints received high ratings on the SSL Test, using RSA 2048 bit keys. No data was intercepted, and certificate pinning was definitely in play.

## V. Privacy

Some say that data privacy is essential, other's believe its an oxymoron. This section will present potential issues regarding application permissions, and trackers, either eToro's own or by 3rd parties.

### A. *Permissions*

As any application, eToro requests a list of Android Permissions. These include very standard ones like allowing to be paired to Bluetooth, or allowing the application internet access. There is however a few permissions granted to eToro, that are interesting. These are as follows:

- Camera: While needed for Authentication, this is seemingly the only functionality interfacing with the camera. It would be in the Users interest to disallow camera use after first authentication.
- Read and Write_External_Storage : This could potentially be an issue, as it allows the application access to shared storage, i.e all your personal files, and the ability to add new or delete existing. Reading data could be needed to upload pictures or files to eToro's social 'News Feed', however even after using the app in my personal life, the need to write to storage is not evident. However, as eToro targets SDK 29, which was evident from MobSF, this permission gets overhauled next update, as SDK 30 (Android 11) features security changes to external storage [24]

### B. *Trackers*

Using Exodus Privacy, the APK was subsequently analyzed for trackers. It found the following:

| Name | Tracks: |
|---|---|
| AppsFlyer | Analytics |
| Countly | Analytics & Profiling |
| Facebook Analytics | Analytics |
| Facebook Login | Identification |
| Google Firebase Analytics | Analytics |
| Tencent Stats | Analytics |
| Urbanairship | Analytics |

TABLE I: Trackers found

It's evident that eToro collects large amounts of information about their users. The java files for each tracker has been manually examined, however due to the complexity size and obfuscation at places, it's not evident what statistics are gathered. What, however is more clear, is a few potential issues found:
As earlier mentioned, Appsflyer utilizes outdated MD5 and SHA1 hashing, however it seemingly only converts already SHA256-hashed messages to MD5 as shown below. The reasoning remains unclear.

```
String md5 = HashUtils.toMD5(HashUtils.toSha256(sb.toString
    ()));
```

Listing 2: 'Excerpt of 'Crypto' class

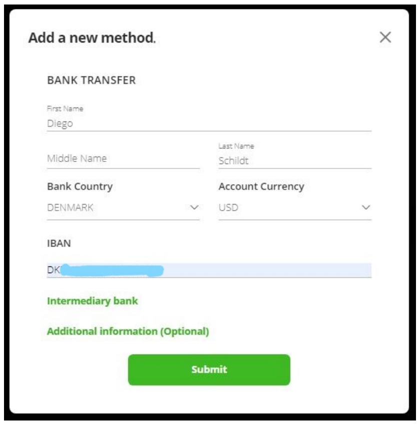
Urbanairship typecasts https connections to http, however this is likely just to transfer non-sensitive information quicker, and thus seemingly provides no security risk.
Tencent Stats also has MD5 implemented but due to the large amount of obfuscation, it was not possible to figure out if the function was called elsewhere. Furthermore, Tencent also uses http, which at one point connects to a Chinese IP, 'www.pingma.qq.com', which seemingly is an MTA server and would thus benefit from using TLS encrypted messages.

## VI. Authentication

As touched upon in the thread analysis section, authenticity and non-repudiation is a key policy. eToro is a financial app, and is thus subject to a strict set of guidelines. As it is a global app, licensing entities vary, however in Europe, they are regulated by the Cyprus Securities and Exchange Commission (CySEC) [16]. The actual regulations and laws are out of scope for this report, but eToro has a responsibility to ensure you, the user, is who you claim, to amongst other things, prevent money laundering and fraud.
The authentication measures will be split into two categories. Firstly, there's the ones that are one-time things, such as proof of identity and proof of address. Secondly, the authentication measures that is routine, such as username and password.

### A. One time authentication

eToro requires your account to be verified before you can deposit funds. This is done to conform with article 62(2) of the 'Prevention of Money Laundering and Terrorist Financing Law' which states *"The verification of the identity of the customer and the beneficial owner is performed before the establishment of a business relationship or the carrying out of the transaction"* [21]. This is done in two different ways. [22]
The first is using 'Proof Of Identity', which requires the user to take a photograph of an identifying document. This could be driving license, government issued ID, or passport. The next is a 'Proof Of Address', a document which has to contain your full name, address and be 3 months old at latest. Personally I used a Credit Card statement when performing my identification, but phone bills, official letters or similar would also work. The authentication is done, at-least partly, by hand, and thus takes a few days to complete, but adds a layer of reinsurance. It's also worth noting that eToro does not allow withdrawals to services or bank accounts, that are not registered in the same name as your eToro account. This can be seen below:
Firstly, a withdrawal is attempted, using a pseudo-random name and otherwise correct information.


Add a new method.

BANK TRANSFER

First Name
Diego

Middle Name

Last Name
Schildt

Bank Country
DENMARK

Account Currency
USD

IBAN
DK[redacted]

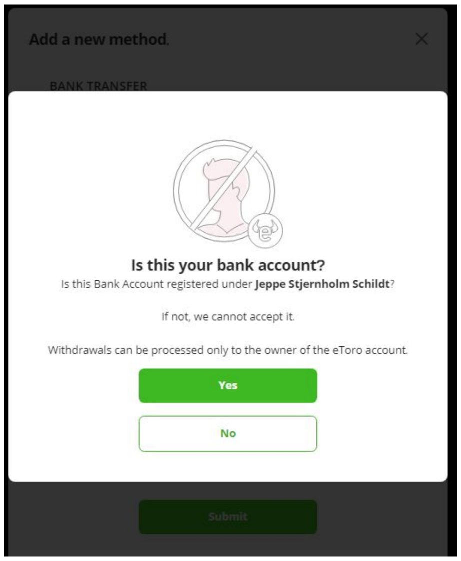
Intermediary bank

Additional information (Optional)

Submit

Fig. 8: Withdraw Attempt

Upon proceeding, the app attempts to verify if the bank-account is registered under the Owners Name.


Add a new method.

BANK TRANSFER

**Is this your bank account?**

Is this Bank Account registered under **Jeppe Stjernholm Schildt**?

If not, we cannot accept it.

Withdrawals can be processed only to the owner of the eToro account.

Yes

No

Submit

Fig. 9: Withdrawal Authentication

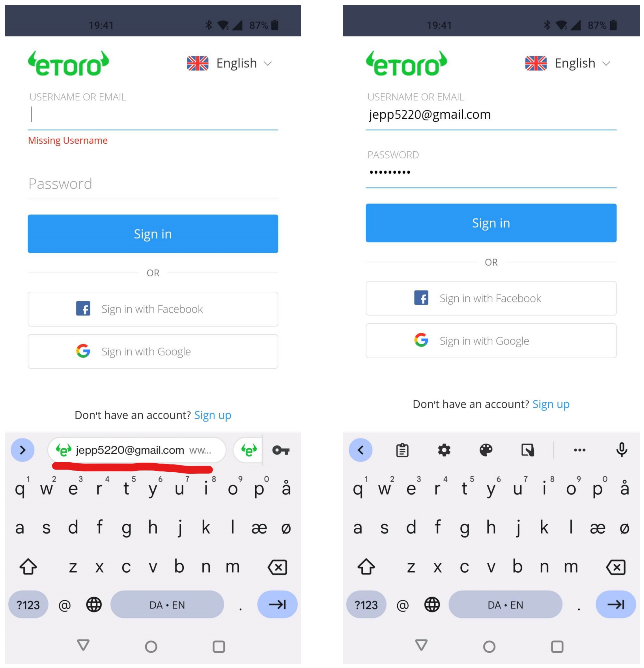
The IBAN provided is not registered under my name, and I do not wish to withdraw funds, so the 'experiment' is stopped however my withdrawal would definitly be cancelled upon inspection of the Bank account.

### B. Routine Authentication

When opening the app, the user is prompted to log in using a username and password, as is customary. The password must be 8 characters long, and at most 32, and has to contain either uppercase letters or symbols. To increase security, eToro also requires 2 factor authentication (2FA), using either email or phone, after the first deposit. Biometric Authentication, using facial or fingerprint recognition is also supported, replacing the username, password and 2FA code upon login [11].

This all points at a very strong authentication measure, however there's a few potential issues. The first lies in the timer begun when closing the app. If you reopen the app within a certain time frame, even if the app has been forcible closed, you'll still be logged in. The specific time has not been found,

but seems to be between 1 hour and 3 hours.
Another potential issue lies in android/google's saved credentials. Given an unlocked phone, any adversary would be able to, with 2 clicks, enter someones eToro account with full access. The flow is shown in figure 10, with a red line underlining the saved credentials that just require a click to be auto-filled.



(a) Password Manager Prompt (b) Filled out information

Fig. 10: Password Manager Flow

While the credentials themselves are saved securely by Google, and this is a deliberate choice of the developers to improve the user experience, with the portfolio sizes existing, perhaps adding a small verification check, such as pin, would be worth the nuisance.

## VII. Conclusion

This report served as the documentation of a preliminary security analysis of the eToro mobile application, an investment and social trading platform. Firstly, a threat model was performed highlighting the adversaries, attack surfaces and key policies, and subsequently the application was examined using a broad range of tools. A few potential issues were found, such as associated analytic companies using MD5 and SHA-1, and the app having enabled application backup. No hard coded sensitive information was found, the app and it's associated entities all used proper protocols and while packet sniffing and man in the middle attacks were attempted, eToro and Android had guards against this. The app utilizes obfuscation to a large extent, especially in their JavaScript files, to lower the risk of these being entry points. In general, the application seems very secure, with no real issues found in neither the application or associated entities, however as issues arose with emulators, no certificate injection was attempted. This being said, early play-testing and the static analysis both pointed at eToro not allowing logins using rooted devices.
Lastly, the app has a wide variety of both mandatory and optional authentication measures, ensuring non-repudiation and authenticity.

## References

[1] MobSF Static Analyzer.
https://mobsf.live
[2] Genymotion Android Emulators.
https://www.genymotion.com
[3] Dex to Jar converter.
https://github.com/pxb1988/dex2jar
[4] APK file Downloader.
https://apps.evozi.com/apk-downloader/
[5] Java GUI used for CLASS files.
http://java-decompiler.github.io
[6] eToro Monthly Active Users.
https://www.businessofapps.com/data/etoro-statistics/
[7] Java's Security Framework.
https://www.javatpoint.com/java-security-framework
[8] BouncyCastle, cryptographic implementations.
https://www.bouncycastle.org
[9] Deprecated BouncyCastle features.
https://www.bouncycastle.org/docs/pkixdocs1.5on/index.html?deprecated-list.html
[10] CVE Details Appsflyer rating
https://www.cvedetails.com/vulnerability-list/vendor_id-13652/Appsflyer.html
[11] eToro's Authentication measures.
https://www.etoro.com/en-us/customer-service/help/1281239912/how-can-i-make-my-account-more-secure/
[12] Packet Capture Application
https://play.google.com/store/apps/details?id=app.greyshirts.sslcapture
[13] AppsFlyer Security Policy
https://www.appsflyer.com/trust/security/
[14] Android Debug Bridge.
https://developer.android.com/studio/command-line/adb
[15] Genymotion ARM Translation tool
https://github.com/m9rco/Genymotion_ARM_Translation
[16] Cyprus Securities & Exchange Commission.
https://www.etoro.com/customer-service/regulation-license
[17] Immuniweb Static Analyzer.
https://www.immuniweb.com
[18] OSWAP Top 10 Mobile Threats (2016)
https://owasp.org/www-project-mobile-top-10
[19] JavaScript Static Code Analysis Tool. https://jshint.com/
[20] JavaScript De-obfuscation Tool.
http://jsnice.org/
[21] AML/CFT Law, Article 62.
https://www.cysec.gov.cy/CMSPages/GetFile.aspx?guid=127a39aa-c325-4a84-9390-0ec29da860df
[22] eToro's Account Verification.
https://www.etoro.com/customer-service/account-verification/
[23] DNS CAA Mandate.
https://cabforum.org/2017/03/08/ballot-187-make-caa-checking-mandatory/
[24] Android 11 Storage Changes.
https://developer.android.com/about/versions/11/privacy/storage