POLONIEX Security review

What is Poloniex?

Poloniex is the largest altcoins trading platform in the world. With over 30 000 bitcoins volume every day, they are leading the so volatile and controversed altcoins market.

Funded by Tristan d'Agosta, the young director of the Poloniex start-up studied at the Rutgers University and got his Arts & Music Bachelor in 2009. He firstly created Polonius Sheet Music in October 2010 before finally launching Poloniex trading platform on January 2014. This success story let him and his team earn - with approximately 0.01% fees on each trade - a total of 60 bitcoins every day ($36600/day on 09/10/2016).

Why is Poloniex insecure?

As of 07/10/2016, I have found the 3rd vulnerability in 1 month of my really light testing. Firstly, you would like to know that I'm in any case an experimented neither diplomed web pen tester but just someone who likes to know having its funds in security. During those test, I have seen brain damageable code and irresponsibility of the Poloniex support as well as lack of customers' consideration.

This is a list I will come back on my article of Poloniex' bad coding practice:

- Using GET request instead of POST for every crypto currency transactions without any CSRF tokens! (1)
- No type check (2)
- Client side security (3)

It would be good to remember that Poloniex has already been hacked and lost 12.3% (approximately 50 BTC) of their Bitcoins. As Poloniex support Busoni declared on bitcointalk, "*The hacker discovered that if you place several withdrawals all in practically the same instant, they will get processed at more or less the same time. This will result in a negative balance, but valid insertions into the database, which then get picked up by the withdrawal daemon."* So what did happen really? Poloniex is using PHP + nginx for their server. Nginx is multithreaded it means it can perform many request at the same time, if the 2 withdrawals request are being performed in 2 different threads at the same time both of them will be validated because the first thread didn't update the number of bitcoins from one user in the database for the withdraw that the second thread already picked the number of bitcoins available from it.

It is good to note that Poloniex fully repay back the Btc stolen.

Despite this hack, Poloniex seems as weak as it was 2 years ago and did not increase its security using bad coding practices.

(1) Using GET request instead of POST for every crypto currency transactions without any CSRF tokens!

There are 2 important request to communicate informations between server and client: GET and POST request.

GET request are passed by your URL, they are good if you want to pass public information. (eg : you want to share a certain topic page : http://example.com/forum/poloniex-is-unsecure?page=3).

You use POST request for "private" action, those parameters can be send only from AJAX (javascript) or an input box so that you don't take the risk to click on a wrong link (eg : https://bank.com/withdraw/address=5a3z5a&amount5000)

There are 2 others security which certify a request can be trusted and is wanted by the user: the request comes from the bank website and not from another website AND it includes a csrf token. A csrf token is a uniq token generated by the bank that can be used for only one request. It is a hashed string generated randomly which change on each request. Only a request with the good csrf token will be validated by the bank, otherwise, it detect it's not the customer doing the request and abort it.

So now, the important question, what does poloniex do?

They are using … GET request! It is a terrible bad practice that any person involved in security would scream while discovering it. Does it mean that if you click on a link your funds will be withdraws from poloniex? Not really, Poloniex is using a really weak security: they are checking if the request has been made from their website (it's called the Referer) and as you can't share link anywhere (not even clickable link in the trollbox, only moderator can), this attack never happened.

An example of GET request Poloniex is using:

https://poloniex.com/private.php?currencyPair=BTC_NAV&rate=0.00006745&amount=1000&command=buy

To resume, this attack can happen in case of:

- Open URL Vulnerability: A poloniex webpage redirecting to another. Howewer, it can only happen from a javascript redirection (window.location=link) and not from a Location header) eg:http://mywebsite.com/redirection=/private.php?currencyPair=BTC_NAV&rate=0.00006745&amount=1000&command=buy
  I have been able to find 2 open URL vulnerability unexploitable on Poloniex but it lets think than there could be more …

- Moderator account hijacking/usurpation will let the attacker sharing clickable link and because they are from the Trollbox which is hosted on poloniex, the Referer will be valid and the buy/sell will be validated.(4)

(2) No type check

Computers works with different type of data. It treats data differently in terms of the type. Types are integer, float, string, array … A short and explicit example I can give is “1”+”1”=”11” and 1+1=2.

A safe website have to check that the type it receives is the expected type, otherwise, it could cause unexpected behaviour. From PHP type juggling to SQL vulnerability

Such coding is representative of bad security policy. For example, this request (https://poloniex.com/private?command=withdraw&currency=DOGE&amount=6&address[]=&confirmed=), afak that it is again a GET request, is generating this output:

Poloniex should check the withdrawal address to confirm his validity. This apparently inoffensive array vulnerability is a proof of the bad coding practice. Learn more about array vulnerability.

(3) Client side security

NEVER EVER implements client side security check! Why?

- Source code is visible by the attacker and it is easier to find vulnerability.
- Web page usually contains more source code and it weighing/slowdown the page.
- Hard coding informations do not let space to real time.

All those bad coding practice have led to a moderation client-side privilege escalation that I’m about to describe below.

(4) Trollbox moderation privilege escalation

```
var mods =
{"Chickenliver":1,"MobyDick":1,"InfiniteJest":1,"cybiko123":1,"SweetJohnDee
":1,"smallbit":1,"Wizwa":1,"OldManKidd":1,"Quantum":1,"busoni@poloniex":1,"
Thoth":1,"wausboot":1,"SolarPowered":1,"qubix":1,"Oldgamejunk":1,"Chewpacab
ra":1,"j33hopper":1,"Futterwacken":1,"ultim8um":1,"Atlanta":1};

// […]

if (trollboxRow['username'] in mods){
      // do some action
      if(trollboxRow['message'].indexOf("https://poloniex.com") != -1){
            // set clickable link
      }
```

}

The source code is available on web archive as the vulnerability has been patched after my report.

The “in” operator return true if the property is part of the given Object. So for example, toString is an element of trollboxRow[‘username’] and an element from the mods array ! Taking this username will grant me moderation client privilege which includes: having my name in blue and the ability to share clickable link.

This then allow me to share buying/selling cryptocurrency link because of weak verification (1) that I can even hide with Open url vulnerability.

History of my conversation with Poloniex support:

- Reported Open URL Vulnerability on the 2nd September

  - Got answer and bounty of 0.2 btc on the 4th September.

- Reported strange behaviour on withdrawal on 16th September

  - After many relaunch and contacting moderators, I still didn’t receive any answer. Bug is still not solved.

- Reported trollbox moderator pvilege escalation on the 7th October

  - After many relaunch and contacting moderators, I still didn’t receive any answer. But had been solved.

I’ve been alerting the moderation about 6/7 times I was awaiting answer for my tickets. Each time I got more or less the same answer: “*We apologize for the delay, I have pushed your tickets, rest assure the support will reply very shortly*.”

I’m a bit upset about the irresponsible communication and after my first report, I thought Poloniex was on the right path.