Blockchain

David Herel

> ## Bitcoin task
>
> The goal of this lab is to get familiar with blockchain (and bitcoin as the famous example of this technology) and brute force attacks.

> There exists a fairly popular site https://www.bitaddress.org which allows you to generate new bitcoin addresses in your browser. For the purposes of this homework we have modified the site and introduced a vulnerabiliy and upload the vulnerable code here.

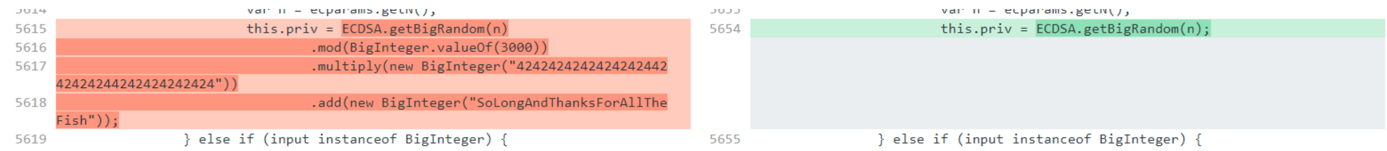
> The task is to find a private key for existing bitcoin wallet with real money on it. Based on your personal preferences you can follow either of following stories.

> ### Story 1 - Good guy (at least at beggining?)
>
> There are multiple incidents of bitcoin stealing - as this is not actually unpossible (today), guys just had to share their private keys, right? Well, most of them used unofficial clone of bitaddress.org.html website to generate their keys. As the algorithm is written in java-script and included into html website, you can run the code localy. Many of them did so and still - someone rob their money! Get in touch with the code and find how it is possible. For which address(es) with real money did you find a private key? Will you aware a community? (Will you grap the money?)

> ### Story 2 - Evil guy
>
> There is popular key generator [NOTE: modified for purpose of this exercise]. It is used by many guys to (securely) get their own unique keys. Well, securely. But, can you look into the code and find vulnerability? Can you somehow brute force generated keys and find some with money? Let's steal some money!

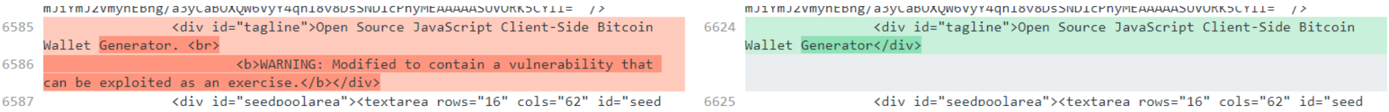
> ## The task
>
> You will complete the homework by sending the correct private key to me via email along with **text description and code** of your solution.

At first I will compare the modified code with the original one. And find the modifications, which I hope will guide me further.



```
                        var n = ecparams.getN();
                        this.priv = ECDSA.getBigRandom(n)
                                .mod(BigInteger.valueOf(3000))
                                .multiply(new BigInteger("4242424242424242442
4242424424242424242424"))
                                .add(new BigInteger("SoLongAndThanksForAllThe
Fish"));
                } else if (input instanceof BigInteger) {
```

```
                        var n = ecparams.getN();
                        this.priv = ECDSA.getBigRandom(n);
                } else if (input instanceof BigInteger) {
```



```
mJ1YmJ2vmynEBng/a3yCaBUXQW6vyY4qn18v8DsSNDIcPnyMEAAAAASUVORK5CYII=  />
                <div id="tagline">Open Source JavaScript Client-Side Bitcoin
Wallet Generator. <br>
                        <b>WARNING: Modified to contain a vulnerability that
can be exploited as an exercise.</b></div>
                <div id="seedpoolarea"><textarea rows="16" cols="62" id="seed
```

```
mJ1YmJ2vmynEBng/a3yCaBUXQW6vyY4qn18v8DsSNDIcPnyMEAAAAASUVORK5CYII=  />
                <div id="tagline">Open Source JavaScript Client-Side Bitcoin
Wallet Generator</div>
                <div id="seedpoolarea"><textarea rows="16" cols="62" id="seed
```

I found the differences using https://www.diffchecker.com/diff

So from the pictures we can see that the generation of the private key is modified. We use modulo on the original random number so it can not be bigger than 3000. Also it is multiplied by `4242...24` then another number is added to it.

From this we can reverse engineer the keys. Because we know that random number is between 0 and 3000 and then multiplied and added with that other numbers. So we can get all 3k private keys. Then the other problem will be to look at all associated public addreses and find some with bitcoins.

So lets first generate those private keys

Unfortunately the source code of the web is in the javascript, so my code in python did not work because in the source code they are using a lot of fucntions which are neccesary and would be dificult to rewrite in pytohn. So I had to switch to javascript and add the code to the end of the html file.

I am not a familiar with javascript and had a lot of problems extracting the data. My final solution was to store it in json and then output it to the console and

```
// arrays for storing priv and public key
```

```
var priv_arr = [];
var pub_arr = [];
for (let i = 0; i < 3000; i++){
    var priv = BigInteger.valueOf(i).multiply(new BigInteger("424242424242424244242424244242424242424")).add(new BigInteger("SoLongAndTh
    // functions needed to get the key
    var key = new Bitcoin.ECKey(priv);
    key.setCompressed(true);
    // private key
    var priv_key = key.getBitcoinWalletImportFormat();
    priv_arr.push(priv_key);
    // public key
    var pub_key = key.getBitcoinAddress();
    pub_arr.push(pub_key);
}
console.log(JSON.stringify(priv_arr));
console.log(JSON.stringify(pub_arr));
```

```
SyntaxError: invalid syntax (<ipython-input-3-3be62cc8444b>, line 1)
```

Show error details

The code ran for a while and browser got freezed. But then I opened console with `ctrl + shift + j` and copied content of the json.

Now we need to look on every public address and check if there is some bitcoin balance. Luckily now I can do this in python since I have the data.

```
#data is from previous task
pub_arr = ["19Rn11MzzrVKh76ne5qEuCyQDNtWzoH8Bh","13ra3h6M45ETbgAXrYccMWywjUwztDpZcG","1JDTvDxq7Bunx5Qs4bY2gCYdnJ4nASMkZr","1uqbJSDheReb7
priv_arr = ["KwDiBf89QgGbjEhKnhXJuH7LrciVsVfNi52z85pscMLC79CFmLNs","KwDiBf89QgGbjEhKnhXJuH7gUvKh8PV42rMqDtRkakUm1QWqJLNe","KwDiBf89QgGbj
#3k priv 3k public
print(len(priv_arr))
print(len(pub_arr))

#install blockchain api
!pip install blockchain
from blockchain import blockexplorer

#zip priv and pub arrays
for priv, pub in zip(priv_arr, pub_arr):
    address = blockexplorer.get_address(pub)
    if(address.total_received > 0):
        print("This address received some bitcoins: " + str(pub))
        print("It's private key is: " + str(priv))
        print("Current balance of this address is: " + str(address.final_balance))
```

```
3000
3000
Requirement already satisfied: blockchain in /root/venv/lib/python3.7/site-packages (1.4.4)
Requirement already satisfied: enum-compat in /root/venv/lib/python3.7/site-packages (from blockchain) (0.0.3)
Requirement already satisfied: future in /shared-libs/python3.7/py/lib/python3.7/site-packages (from blockchain) (0.18.2)
WARNING: You are using pip version 20.1.1; however, version 21.3.1 is available.
You should consider upgrading via the '/root/venv/bin/python -m pip install --upgrade pip' command.
This address received some bitcoins: 1E2mSN7MXVuS4ecafhTLtaokf5RixcYUEU
It's private key is: KwDiBf89QgGbjEhKnhXJuY4GUMKjkbiQLBXrUaWStqmWnp3XBMte
Current balance of this address is: 39500
```

I have used blockchain api and managed to get the bitcoin address which had bitcoins in the past. So the corresponding address is: `1E2mSN7MXVuS4ecafhTLtaokf5RixcYUEU` with the private key `KwDiBf89QgGbjEhKnhXJuY4GUMKjkbiQLBXrUaWStqmWnp3XBMte` and the current balance of it's address is `18 dollars`, which coresponds to `0.00039500 BTC`.