Alternative Method of Cryptocurrency Wallets Managing

Gabit Omarov [a], Dzholdas Dzhuruntayev [a], Andriy Fesenko [b], Sanzhar Umbet [a] and Serhii Dorozhynskyi [c]

[a] *Satbayev Kazakh National Technical University, 22 Satbaev St, Almaty, 050013, Kazakhstan*
[b] *Taras Shevchenko National University of Kyiv, 24 Bohdan Havrylyshyn St, Kyiv, 04116, Ukraine*
[c] *National Aviation University, 1 Liubomyr Huzar ave., Kyiv, 03058, Ukraine*

Abstract

With the rise of cryptocurrencies in the market, the convenience and safety of using cryptocurrency wallets became a top priority for people. Although cold cryptocurrency wallets are safe to use, they cannot be compared with hot cryptocurrency wallets and telegram bots at their convenience. In this research paper, hot crypto-wallets and telegrams bots for making and receiving transactions of cryptocurrencies with the least amount of investments will be discussed. Also, the possible methods and their implementations will be considered.

Keywords

Blockchain, blockchain wallets, cryptocurrency, telegram messenger, mobile applications, bitcoin (btc), litecoin (ltc), KZCash (kzc), Qiwi, API

1. Introduction

Blockchain - is neither just an object, a product, a trend nor an opportunity for people to make easy money. It consists of several parts, some of which work together, while others - separately and independently. Thanks to this modularity, the blockchain has an infinite number of use cases including financial services, tax regulation, Internet of Things (IoT), and so on. In the long run, most users will not know or understand that there is a blockchain in the software or service they use [1].

Blockchain is a decentralized transaction log that is part of a broader computing infrastructure that has wide-ranging functions. In the modern world, Blockchain plays a vital role as a ledger that allows people to make transactions in a decentralized manner [2].

2. Problem statement

Due to the trend in the development of the cryptocurrency industry, it is necessary to develop hot wallets for any cryptocurrency. This is usually an online wallet that works through a specific website or mobile application developed for frequently used operating systems.

Recently, fraudulent actions have become more frequent when exchanging online cryptocurrencies. Hence, those who do not know the intricacies of the blockchain often come across such cases. In this regard, for this research paper designed telegram wallet must also provide a secure and guaranteed exchange of fiat money for cryptocurrency.

The purpose of this work is to select and implement a relatively cheap but at the same time reliable method of managing a cryptocurrency wallet online.

3. Proposed method and solution

The first and the simplest method of providing access to the management of a blockchain wallet is the creation of a website that allows to control and manage cryptocurrencies.

Information Technology and Implementation (IT&I-2021), December 01–03, 2021, Kyiv, Ukraine
gabit.omarov@gmail.com (G. Omarov); joldas.zaurbek@gmail.com (D. Dzhuruntayev); aafesenko88@gmail.com (A. Fesenko); sanzhar.umbet@nu.edu.kz (S. Umbet); dorozhun1706@gmail.com (S. Dorozhynskyi)
ORCID: 0000-0002-2561-1110 (G. Omarov); 0000-0003-4751-2014 (D. Dzhuruntayev); 0000-0001-5154-5324 (A. Fesenko); 0000-0003-0506-7450 (S. Umbet); 0000-0002-5395-6423 (S. Dorozhynskyi)

This method can be used when you need to implement it quickly. However, there is a great risk associated with using this method. Namely, it may become vulnerable to all kinds of attacks, starting with DOS and DDOS attacks. Hence, it necessary requires proper protection and maintenance to function well. The second method is the development of a mobile application. Using this method requires the development of a mobile application for each operating system separately. Also, there is a need to publish it on special resources like Play Market or Appstore. The third method that was designed solely for this research paper's purposes is to use the Telegram messenger as an application. This method can overcome the disadvantages of the previously mentioned methods. There is no need to develop and maintain a mobile application since it is already provided by a telegram messenger.

3.1. VPS hostings

First, in order to create a telegram bot that will help us to maintain and control our crypto-wallet, the "father of bots" telegram bot was used (https://t.me/BotFather). It allows users with no experience in programming to easily set and manage their telegram bots. The second task is to configure the backend of our bot. It will be required to send user requests and receive responses. Since we need to ensure that the telegram bot works 24/7 and that there is no need to purchase a physical server that is much expensive than alternatives, it was decided to rent a virtual server (hereinafter referred to as VPS) on cloud services [3]. We need a VPS where we could install the Linux Ubuntu operating system to program the wallet. We were looking for a VPS with at least 4 GB of RAM and a permanent memory on an SSD of at least 50 GB [4]. One of the leading companies in Kazakhstan for virtual server rental company ps.kz offers the following solutions:

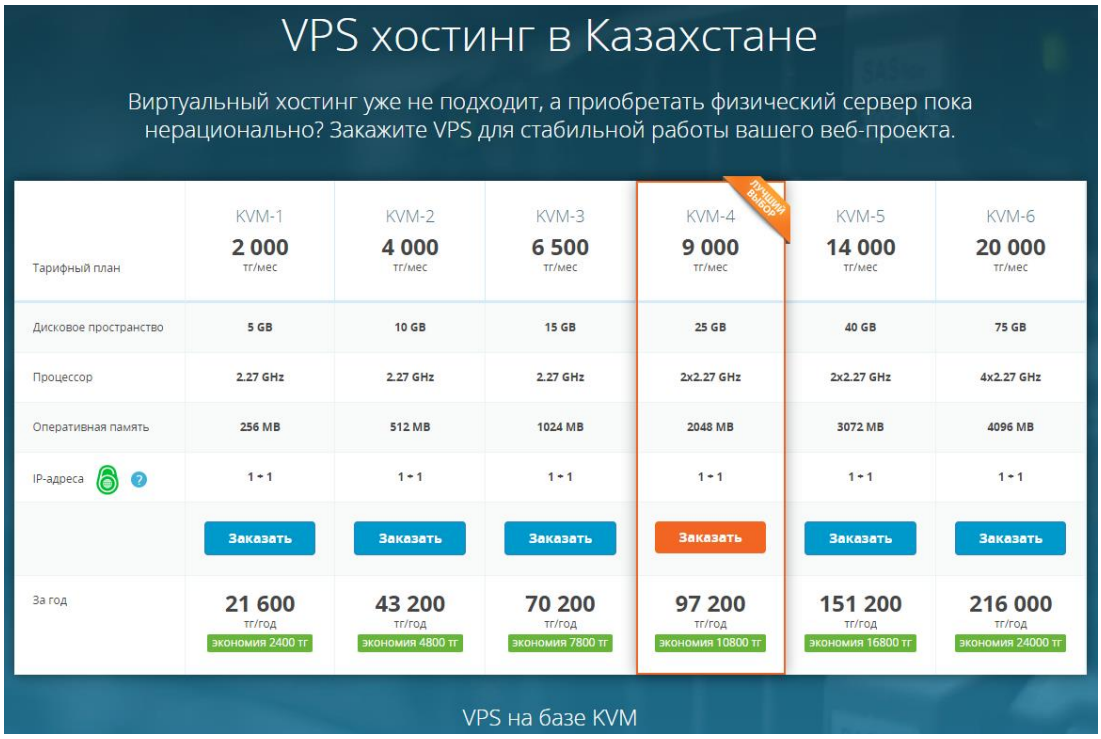


Figure 1. PS.kz website's prices for virtual servers

Of the proposed solutions, only the latest "KVM-6" for 20,000 tenges (equivalently 47 US dollars) per month is suitable for us. In contrast to similar services, the PS.kz prices occur to be expensive. Hence, choosing VPS hosting in Kazakhstan appears to be financially ineffective, although it is compensated with a high Internet speed. For the stable work of our crypto wallets, the internet speed is important, hence European servers compared to American ones are prioritized due to high and stable Internet connection between Kazakhstan and European servers. We reviewed the solutions of European companies, in particular, Aruba from Italy, which offers such solutions: Of the proposed, our requirements are covered by the "Large" solution for 12.50 Euros (equivalently 6500 tenges), specifically, because it has 4 GB of RAM and 80 GB of SSD, which is required for stable and reliable work of our crypto-wallet.

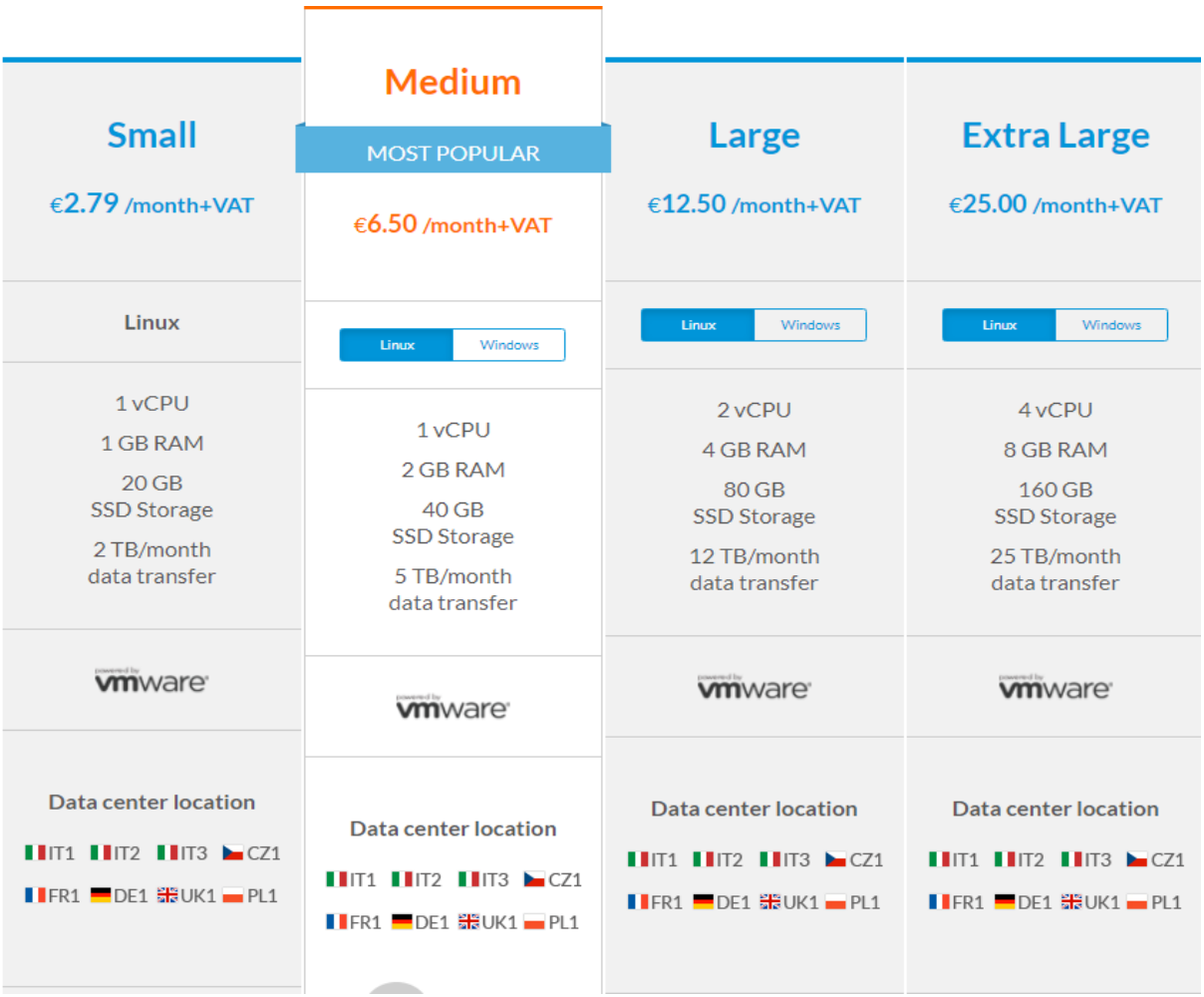


Figure 2. Aruba website's prices for VPS hosting, Italy.

Additionally, one of the largest VPS hosting in Germany "Contabo" was considered. They have the following prices for their service: On the left side, they offer VPS servers with HDD and SSD, while on the right side, they offer VPS series that work solely on SSD. Of these, our requirements are covered by the first solution "VPS S SSD". The cost of this service is 4.99 Euros, equivalent to about 2600 tenge. We settled on this solution. This server has a 4-core 2.2GHz virtual processor, with 8 GB of RAM, and a 200 GB SSD disk, which is enough to sustain stable and reliable work of our wallet.

3.2. Implementation

First, we need to install Linux Ubuntu operating system on our server. In the following picture, Linux Ubuntu 16.04 was installed: Python was chosen as the service development language. Therefore, a Python development environment was installed on this server. This language was not chosen by chance. There are ready-made libraries for managing blockchain wallets for the Python language, and it is convenient to use it to access various services using the API since our Telegram bot will have to exchange different services via API [5].

Blockchain wallets of cryptocurrencies - KZ Cash, Bitcoin, Litecoin were installed on the server. To install, it is enough to run the following commands in the Linux OS command line [6]:

 # wget https://raw.githubusercontent.com/kzcashteam/mn_install/master/kzcash_mn_install.sh
 # chmod +x kzcash_mn_install.sh
 # ./kzcash_mn_install.sh
 # wget https://bitcoincore.org/bin/bitcoin-core-0.20.1/bitcoin-0.20.1-x86_64-linux-gnu.tar.gz
 # tar xvf bitcoin-0.20.1-x86_64-linux-gnu.tar.gz
 # wget https://download.litecoin.org/litecoin-0.17.1/linux/litecoin-0.17.1-x86_64-linux-gnu.tar.gz
 # tar xvf litecoin-0.17.1-x86_64-linux-gnu.tar.gz

Figure 3. Contabo prices for the VPS hosting, Germany.

After installation, it is needed to configure the ports with which they will work in the configuration file of each coin. For the KZ Crash coin, the settings file will be the following data:

rpcuser={user_name kzc}
rpcpassword={user_password kzc}
rpcport=8279
listen=1
server=1
rpcallowip=127.0.0.1

For Bitcoin (btc):

rpcuser={user_name btc}
rpcpassword={user_password btc}
rpcport=9341
listen=1
server=1
rpcallowip=127.0.0.1

For Litcoin (ltc):

rpcuser={user_name ltc}
rpcpassword={user_password ltc}

rpcallowip=127.0.0.1
rpcport=9332
listen=1
server=1

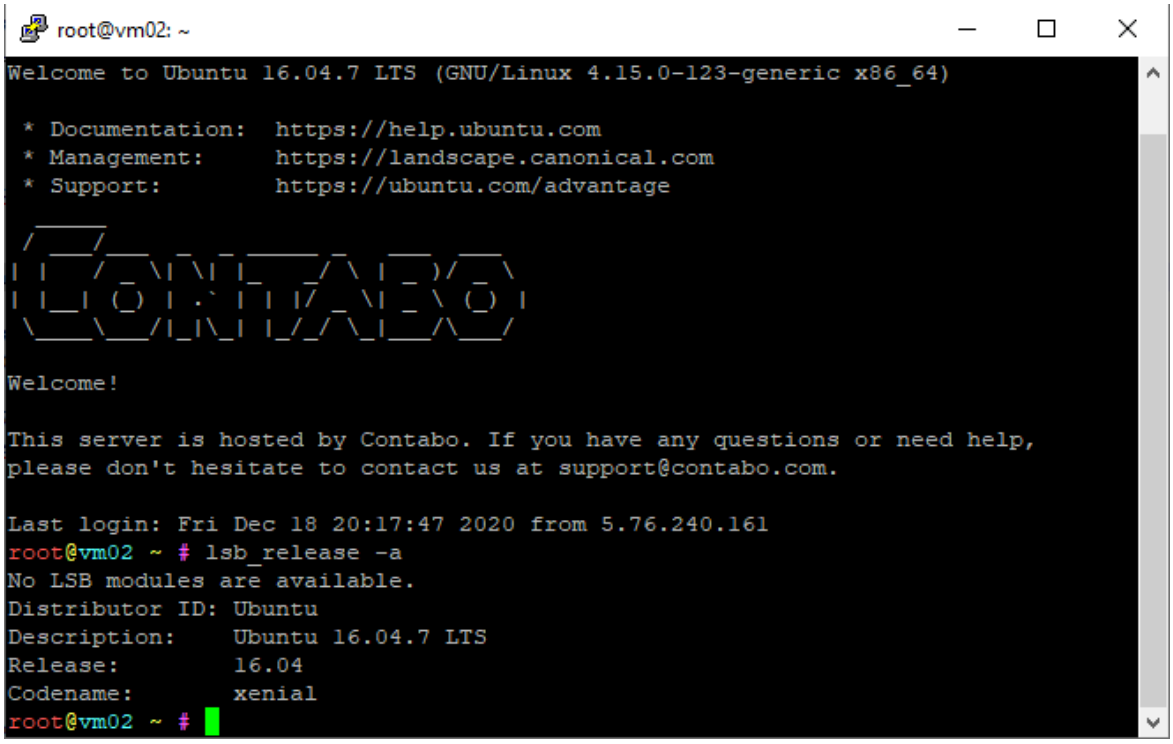


Figure 4. Server with installed Linux Ubuntu operating system.

After making changes to the data in the configuration files, you need to run the wallet daemons:

 # kzcashd -daemon
 # bitcoind –daemon
 # litecoind –daemon

Now we go to the Python settings. It is necessary to create a configuration file there too, which will help wallets to address. Example of a configuration file:

```
wallet_host = "127.0.0.1"
wallet_port_kzc = 8279
wallet_user_kzc = {user_name kzc}
wallet_passwd_kzc = {user_password kzc}
wallet_url_kzc="http://{user}:{passwd}@{host}:{port}".format(user=wallet_user_kzc,
passwd=wallet_passwd_kzc, host=wallet_host, port$
wallet_port = 9341
wallet_user = { user_name btc}
wallet_passwd = {user_password btc}
wallet_url="http://{user}:{passwd}@{host}:{port}".format(user=wallet_user,
passwd=wallet_passwd, host=wallet_host, port=wallet_port)
wallet_port_ltc = 9332
wallet_user_ltc = { user_name ltc}
wallet_passwd_ltc = {user_password ltc}
wallet_url_ltc="http://{user}:{passwd}@{host}:{port}".format(user=wallet_user_ltc,
passwd=wallet_passwd_ltc, host=wallet_host, port$
```

Below is a code example of how a cryptocurrency blockchain wallet can be accessed using Python [7-9]:

```
from bitcoinrpc.authproxy import AuthServiceProxy, JSONRPCException
from config import wallet_url, wallet_url_kzc, wallet_url_ltc
def main(coin, wallet_addr):
```

```
        if coin == 'kzc':
                cur_url = wallet_url_kzc
        elif coin == 'ltc':
                cur_url = wallet_url_ltc
        else:
                cur_url = wallet_url
        acc = AuthServiceProxy(cur_url)
        unspent = acc.listunspent(0)
        for i in unspent:
                if i["address"] == wallet_addr:
                        if count >= amount_fee:
                                break
                        txid_vout.append({"txid":i["txid"], "vout":i["vout"]})
                        count += i["amount"]
if __name__ == "__main__":
        main()
```

It is necessary to use a database to account for certain data, for this the MySQL DBMS was selected. This DBMS is easy to use and ensures the operation of our service [10].

To install MySQL DBMS, the following command can be used:

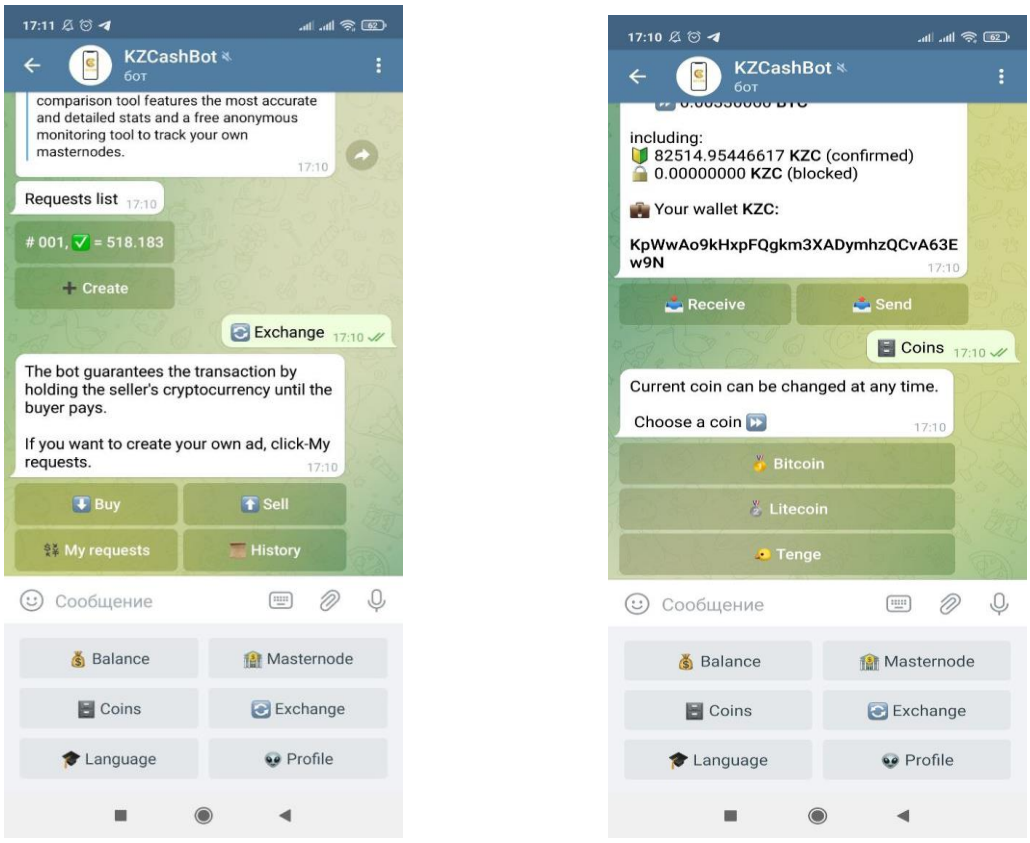
```
# apt-get install mysql-server
```

To connect to the database from Python, you need to connect the appropriate library and use it to perform manipulations [11]:

```
import pymysql
        class sql_db:
                def __init__(self):
                        try:    # connection to the database
                                self.conn = pymysql.connect(
                                        unix_socket=config.ms_unix_socket,
                                        user=config.ms_db_user,
                                        passwd=config.ms_db_passwd,
                                        db=config.ms_db_name,
                                        use_unicode=True, charset='utf8')
                                        except pymysql.OperationalError as e:
                                print("can't connect to Mysql")
                                print(e)
                        try:    self.cursor = self.conn.cursor()
                                self.cursor.execute('SET autocommit = 0;')
                        except pymysql.OperationalError as e:
                                print("can't get cursor")
                                print.error(e)
                                def select_user_id(self, user_id):
                        try:    self.cursor.execute("SELECT * FROM table1 where id =
    '{user_id}'".format(user_id=user_id))
                        except pymysql.Error as e:
                                print("select_user_id, error")
                                print(e)
                                return
                        result = self.cursor.fetchall()
                        if len(result) <= 0:
                                print("select_user_id, error, len(res)=0,
    user_id={user_id}".format(user_id=user_id))
                                return False
                        else: return result[0]
```

Next, in order to connect to the financial system, the Qiwi service was selected. The advantage of this service is that it can be used both for non-cash and cash payments using Qiwi terminals. This service also provides access to your account via the API. Sample code from Python [12]:

```
import requests
import json
def main():
        s = requests.Session()
        s_tok = "токен"
        s_log = "логин"
        s = requests.Session()
        s.headers['authorization'] = 'Bearer ' + s_tok
        parameters = {'rows': '10', 'operation': 'IN'}
        h=s.get('https://edge.qiwi.com/payment-history/v1/persons/'+s_log+'/payments',
params = parameters)
        r_str=json.loads(h.text)
        print(r_str)
```



a *b*

Figure 5. The start page of the telegram bot and the main menu (a); The menu of added coins and the menu for exchanging cryptocurrencies (b).

To connect to banks, it was not possible to get an API to access your account, and therefore it was implemented through a third-party service - Zen Money. This is a service for home accounting. It has the opportunity to connect to many second-tier banks in Kazakhstan. And this service has access to its account via the API [13-15]. Sample code for accessing the Zen Money service:

```
s_tok = "tocken"
        dt = datetime.datetime.now()
        timestamp = time.mktime(dt.timetuple())
        s = requests.Session()
        s_servtime = ""
        s.headers['Authorization'] = 'Bearer ' + s_tok
        logger.info("currentClientTimestamp={0}, serverTimestamp={1}".format(timestamp,
s_servtime))
        parameters = {"currentClientTimestamp": timestamp, "serverTimestamp":
float(s_servtime)}
```

```
h = s.post("http://api.zenmoney.ru/v8/diff", json = parameters)
r_str=json.loads(h.text)
print(r_str)
```

The working version of the telegram bot is now available at this address (fig. 5) - https://t.me/KZCashBot/ . The security of proposed approach can be improved by cryptographic methods and random number generation means, for example [16-19].

4. Conclusion

In this article, the well-known methods of managing hot (online) wallets were considered. We needed an optimal option that would be implemented quickly, supported on all platforms, not complicated maintenance, and with the lowest costs.

As a result, the optimal implementation option was chosen through the telegram messenger. Also, to optimize the costs of server equipment, a VPS was rented on cloud services. The cost of renting a VPS for 1 month is 4.99 EUR. About 40 days were spent on the implementation of this task.

This wallet has also been integrated with fiat money through the Qiwi payment system. It is possible to partially integrate with other second-tier banks using the Zen Money service. It is also planned to add bot tokens on smart contracts to this telegram, as well as a service for launching and managing masternodes.

5. References

[1] L. Leloup, Blockchain. Paris, 2017, 206 p.
[2] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, 2008.
[3] Telegram Bot Lessons. Make Telegram Bot using Python, https://groosha.gitbook.io/telegram-bot-lessons/
[4] B. Al Housani, B. Mutrib and H. Jaradi, “The Linux review - Ubuntu desktop edition - version 8.10”, 2009 International Conference on the Current Trends in Information Technology (CTIT), 2009, pp. 1-6, doi: 10.1109/CTIT.2009.5423142.
[5] Bitcoin Developer Reference, https://bitcoin.org/en/developer-reference
[6] Thomas Sterling, “Linux”, in Beowulf Cluster Computing with Linux, MIT Press, 2001, pp. 61-93.
[7] G. Ayoade, V. Karande, L. Khan, K. Hamlen, “Decentralized IoT Data Management Using Block Chain and Trusted Execution Environment”, 2018 IEEE International Conference on Information Reuse and Integration (IRI), 2018, pp. 15-22, doi: 10.1109/IRI.2018.00011. \
[8] Ahmed Banafa, "Blockchain Technology and Applications," in Blockchain Technology and Applications, River Publishers, 2020, pp.i-xvi.
[9] API reference (JSON-RPC), https://en.bitcoin.it/wiki/API_reference_(JSON-RPC)
[10] MySQL, https://metanit.com/sql/mysql/
[11] Python 3 for beginners, https://pythonworld.ru
[12] API Qiwi, https://developer.qiwi.com/ru/qiwi-wallet-personal/#intro
[13] Cryptocurrency payments by own hands, https://habr.com/ru/post/350430/
[14] Telegram Bot via webhook, https://retifrav.github.io/blog/2018/12/02/telegram-bot-webhook-ru/
[15] Zenmoney:: The art of investing, https://www.zenmoney.com/
[16] M. Iavich, S. Gnatyuk, E. Jintcharadze, Y. Polishchuk, A. Fesenko and A. Abisheva, Comparison and Hybrid Implementation of Blowfish, Twofish and RSA Cryptosystems, Proceedings of 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON), Lviv, Ukraine, 2019, pp. 970-974.
[17] Iavich M., Kuchukhidze T., Gnatyuk S., Fesenko A. Novel certification method for quantum random number generators, International Journal of Computer Network and Information Security, Volume 13, Issue 3, pp. 28-38, 2021.
[18] Gnatyuk S., Okhrimenko T., Azarenko O., Fesenko A., Berdibayev R. Experimental Study of Secure PRNG for Q-trits Quantum Cryptography Protocols, Proceedings of the 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT 2020), Kyiv, Ukraine, May 14, 2020, pp. 183-188.
[19] Iashvili G., Iavich M., Gagnidze A., Gnatyuk S. Increasing usability of TLS certificate generation process using secure design, CEUR Workshop Proceedings, Vol. 2698, pp. 35-41, 2020.