Spuckwaffel#5000

Thundermine – Cracked

In this repository I am analysing a tool called “ThunderMine” that apparently works like this (it’s complete bullshit):

How does Thunder Mine work?

```
- ThunderMine exploits an issue with a popular wallet app's
technique in generating unique seed phrases, which reduces
the range of unique seeds from 2^256 to 2^64. This means they
can easily be mined (albeit, it still would require a bit of
time), to find the private keys of these wallets.

- Once it locates these wallets, it will immediately drain
the bitcoin to your own wallet address, which you define in
the settings. A default fee of 0.00003 BTC is applied, but it
can be modified to anything.
```

This picture is taken from their Discord server with over 30000 members. They have a vouches channel with many fake vouches but also screenshots of people who bought it.

Let’s investigate the program:

On a side note: Users must buy the program for 35$ in cryptocurrency to get a Discord download link. The owner made a video 1 day ago for a showcase but didn’t blur the Discord attachment link, so I just typed it out and got the program for free.

We now have a .rar file and when we unpack the file, we get the following content:

Config.json	4/05/2022 8:32 PM	JSON File	1 KB
Instructions.txt	4/05/2022 8:35 PM	Text Document	1 KB
Newtonsoft.Json.dll	17/03/2021 9:03 PM	Application exten...	686 KB
ThunderMine.exe	4/05/2022 8:40 PM	Application	78 KB

The Config file is completely useless, it wont affect anything. I can already see that I wont even need IDA to crack the program, because the program is written in C# and compiled with the .NET compiler. A simple tool called “dnSpy” can get us the entire source code.

The Main entry doesn’t look too suspicious, it requires the user to either login or register. The program uses a popular auth called auth.gg. To connect to the auth, the program needs to have the name, id, secret and version of the application and send a request to the auth.

```
OnProgramStart.Initialize("Solarcrypt", "366093", "ffUNwzxKuZFrKkMSlTZwT7pY1WzE6R296jw", "1.0");
```

After we logged in, we have the following options:

```
"[1] Public Mining");
"[2] Private Mining");
"[3] Connect your wallet");
"[4] Settings and Statistics");
"[5] Exit");
```

Public Mining doesn't work at all, it will just give you the message "Public Mining is currently under development." And forces the user to restart the application.

Let's look at Private Mining:

```
Thread.Sleep(500);
Console.Clear();
Program.PrintPrivateLogo();
Program.PrivateInfinite();
```

Their program has a huge number of sleeps everywhere to make it look legit. The first function just prints an ascii art of their logo. The second function is the interesting part:

```
int randomInt = Program.GetRandomInt(1, 3);
Console.Title = "ThunderMine Private";
Random random = new Random();
random.Next().ToString("X");
Console.ForegroundColor = ConsoleColor.Magenta;
Thread.Sleep(250);
Console.WriteLine("[+] Note: You will join a private server dedicated to you, any succesful entries will result in the amount accounted to your user id.");
Console.Write("\n[+] Press Enter to Connect...");
Console.ReadLine();
Thread.Sleep(800);
Console.Clear();
Thread.Sleep(300);
Program.PrintPrivateLogo();
Program.FirstLaunch();
Thread.Sleep(20);
Console.Clear();
Thread.Sleep(30);
Program.PrintPrivateLogo();
Console.Title = "ThunderMine Private0" + randomInt.ToString();
Console.Write("\n[+] Connecting to ThunderMine Private0" + randomInt.ToString() + "... ");
int num = random.Next(17, 70);
using (ProgressBar progressBar = new ProgressBar())
{
    for (int i = 0; i <= 100; i++)
    {
        progressBar.Report((double)i / (double)num);
        Thread.Sleep(20);
    }
}
Console.WriteLine("Completed Succesfully!");
Thread.Sleep(50);
Console.WriteLine("\n[+] Logged in as " + User.Username);
Thread.Sleep(500);
Console.ForegroundColor = ConsoleColor.Green;
Console.WriteLine("\n[+] Workers Online: " + Program.GetRandomInt(7, 12).ToString());
Console.ForegroundColor = ConsoleColor.Magenta;
Thread.Sleep(100);
```

It tells the user that it connects to some private server, with a bunch of random number generators and sleeps to make it look even more legit. Even with a progress bar that is completely fake.

```
Thread.Sleep(100);
Console.WriteLine("\n[+] Installing P2WPKH DB         [=====]");
Thread.Sleep(20);
Console.WriteLine("[+] Loading...                    [========]");
Thread.Sleep(120);
Console.WriteLine("[+] Loading...                    [==========]");
Thread.Sleep(80);
Console.WriteLine("[+] Indexing Completed!           [============]");
Thread.Sleep(20);
Console.WriteLine("[+] Wallets Loaded!               [==============]");
Thread.Sleep(20);
Console.WriteLine("[+] Loading Hashes                [================]");
Thread.Sleep(20);
Console.WriteLine("[+] Loading...                    [==================]");
Thread.Sleep(75);
Console.WriteLine("[+] Loading...                    [====================]");
Thread.Sleep(130);
Console.WriteLine("[+] Hash Function Loaded          [======================]");
Thread.Sleep(75);
Console.WriteLine("[+] Job Completed!                [========================]");
Thread.Sleep(75);
Console.WriteLine("[+] Installing Required Modules   [==========================]");
Thread.Sleep(250);
Console.WriteLine("\n[+] Modules Loaded 7/7           [============================]");
Console.Write("\n[+] Validating... ");
```

They really like to let the thread sleep after every console output, huh? Super legit.

But it gets better:

In their final loop they get the current bitcoin price via CoinDesk
(https://api.coindesk.com/v1/bpi/currentprice.json)
and then display random generated "addresses":

```
public static string StringLowercaseInfinity(int length)
{
    return new string((from s in Enumerable.Repeat<string>("ABCDEFabcdef0123456789", length)
    select s[Program.random.Next(s.Length)]).ToArray<char>());
}
```

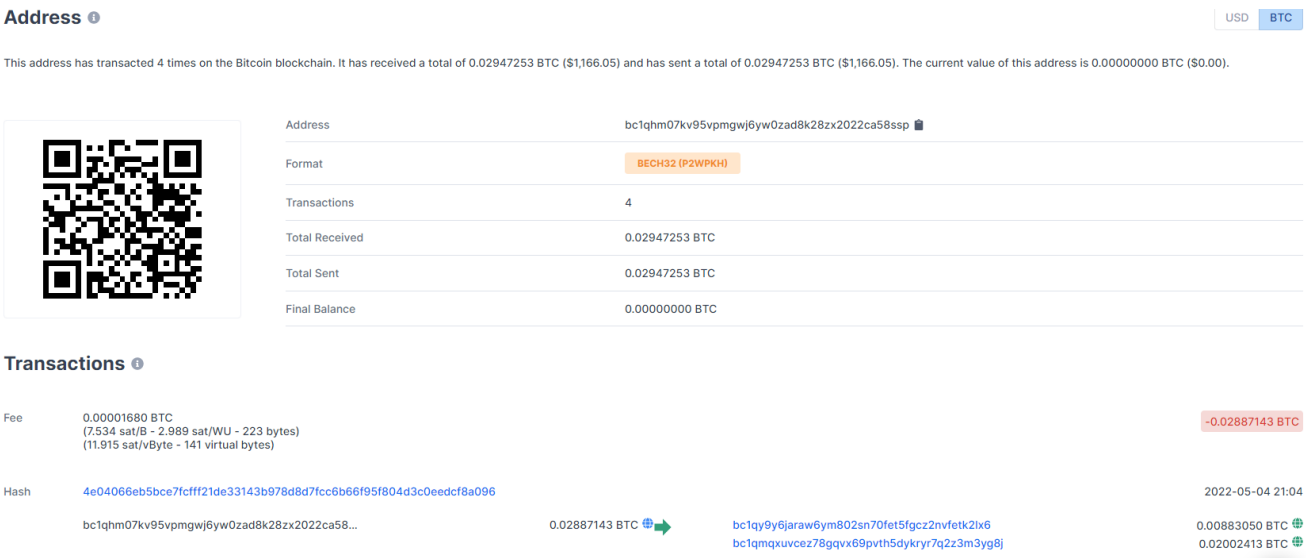
The console always displays 0.00BTC:

```
new Program.ColoredString(ConsoleColor.Magenta, "\n[-] "),
new Program.ColoredString(ConsoleColor.White, "BTC | bc1" + str2 + " - BALANCE: "), //str2: random generated "address"
new Program.ColoredString(ConsoleColor.Red, "0.00 BTC "),
new Program.ColoredString(ConsoleColor.White, "| Sector: " + str + Program.RandomInt(4) + " | Wallet Type: P2WPKH")
```

Of course, coloured text everywhere. But to make sure the user doesn't feel like he got scammed the developer chose to do the following:

```
if (new Random().Next(1, 10000) == 1)
{
    contents = num2.ToString();
    File.WriteAllText(path2, contents);
    Program.StringLowercaseInfinity(32);
    double value3 = Program.RandomDoubleFrom(0.0006, 1.0);
    Thread.Sleep(500);
    Program.WriteConsoleColor(new Program.ColoredString[]
    {
        new Program.ColoredString(ConsoleColor.Magenta, "\n[-] "),
        new Program.ColoredString(ConsoleColor.Green, "BTC | bc1qmqxuvcez78gqvx69pvth5dykryr7q2z3m3yg8j - BALANCE: 0.08591418 BTC | Sector: " + str + Program.RandomInt(4) + " |
          Wallet Type: P2WPKH\n")
    });
```

The user has the chance of 0.01% that the console will show them that they got money. Let's take a look at the bitcoin address (https://www.blockchain.com/btc/address/bc1qhm07kv95vpmgwj6yw0zad8k28zx2022ca58ssp):



This address made transactions, in case the user may look up the address, they showed a legit one). Furthermore, the program tells the user that it sent 0.02947253 to their address. (The exact amount that is in the screenshot above at "Total Sent") and opens a link to the blockchain website with the address (Link above). After that the program asks the user if he wants to continue mining. If the user wants to continue mining, it will continue with the same endless loop.

Very sad how the developer scammed so many people. According to a friend the developer made over 30000$ within a few weeks. Please never fall for this, things like this are illegal (if they would work). The source code is in this repository for educational purposes.