LevelUp0x07 CTF

1. SUMMARY

The purpose of this document is to present the security loopholes of hosts in the https://07.levelupctf.com/ domain and the flags which were found in the process. The domain 07.levelupctf.com is the only one in the scope of the scan and all the vulnerabilities found in the process belong to the domain.

2. FINDINGS

There were 7 flags which were embedded into the domain and the process revealed all the flags in the following order as given in Table 1.

Flag (Format: FLAG{hash})	Location
FLAG{504e8256b8a208845fe18a77e7b304b7}	Source Code of 07.levelupctf.com/assets/js/login.js
FLAG{a445c73c8cb97421d1923a8c51c221fd}	communications.apk/res/values/strings.xml
FLAG{e8606532b027bfd324ea31d1b4f116c2}	User comments of 07.levelupctf.com/ /fa694c73da13c94e49cc82b/06a28bdb78b6c02e16862a3/chat/giraffe.jpg
FLAG{f514875849460428b4dc40dd72a5a29a}	07.levelupctf.com/radio
FLAG{f0dee25f617e2cb820b9b44fcdf90ed8}	Console.txt file in 07.levelupctf.com/agents/agent87.jpg
FLAG{022d8a7a561a02c371fd7c5ec3e5ea06}	File present in server at /home/agent1337/flag.txt accessed from 165.227.54.122:3389
FLAG{de9aaabc9ede23335e5}	07.levelupctf.com/gameover

Table 1

3. APPROACH

The domain was analyzed which is 07.levelupctf.com which showed a bunch of commands like help, brief, findings, intro, credits, discord, greetz. The first challenge starts with brief. Typing brief showed a message which pointed to the /radio directory in the URL.

Flag – 1 – FLAG{504e8256b8a208845fe18a77e7b304b7} :

The URL was now changed to 07.levelupctf.com/radio which redirected immediately to 07.levelupctf.com/login which showed that login was required to access the radio section of the website. Upon arrival to the site the first thing to check, is the components of the site by going through the source code in the browser through "view-source:" in the browser.

The view-source showed a JavaScript file which is situated at 07.levelupctf.com/assets/js/login.js (Appendix Figure 1), which contained the first flag.

Flag – 1 to Flag – 2 Transition :

The first flag, after it was found also led to the discovery of a link given as 07.levelupctf.com/222228a4e79d33a299f5d/s3cretc0mmunications which is the point of contact for the second flag.

Flag – 2 – FLAG{a445c73c8cb97421d1923a8c51c221fd} :

The link has an android application file which needs to be decompiled. The initial decompilation gives original, res, resources, sources, unknown directories and Android Manifest.xml and README.txt along with classes.dex file which is the main file to be decompiled to get the source code.

The decompilation of the classes.dex file gives the directories android, androidx, com, defpackage, okhttp3, okio. Any android application after installation shows files as "com.example.package". With that in mind, exploring the com/example/levelup/ showed three files BuildConfig.java, MainActivity.java, R.java. All the decompilation was done at http://www.javadecompilers.com/.

The MainActivity.java showed two links that was being connected to by the application which are
/d41d8cd98f00b204e9800998ecf8427e/8cd98f00b204e9800998/forgotpassword and
/fa694c73da13c94e49cc82b/06a28bdb78b6c02e16862a3/chat

The second link needed a key and value to be added into the header in the form of name,value given as “3NCRYPT3D-CH4T,key”. The key was obtained from stored values in the application context.

The exploration of stored values in the android application can be done at the location res/values/strings.xml. Going through the file revealed two values, one is the flag and one is the encrypted_chat_key which is 8b0955d2682eb74347b9e71ea0558c67 (Appendix Figure 2)

Flag – 2 to Flag – 3 Transition :

The two links found in the android application were the next point of contact for proceeding through this process. The header needed to be built from the name,value pair obtained from the app which showed REST API authentication form of header values. The other link showed a forgot password where users could access based on usernames (based on the first look of forgot password site). Using Burp Suite the header was modified during request adding name,value pair in the GET request which led to opening of the chat site. The values were (3NCRYPT3D-CH4T, 8b0955d2682eb74347b9e71ea0558c67). (Appendix Figure 3)

Flag – 3 – FLAG{e8606532b027bfd324ea31d1b4f116c2} :

The chat site showed a chat which did not make sense as it was in form of jumbled letters but all of them were ascii printable which pointed out that they were one form of substitution or shift ciphers. Using Caesar cipher chat was decoded but this was just an additional step as the real information found was the image. The image metadata which contained user comments had the flag.(Appendix Figure 4)

Flag – 3 to Flag – 4 Transition :

The image metadata revealed location coordinates which pointed to San Francisco Zoo (Appendix Figure 4). This was also discussed in the chat which revealed users names in the site. These user names were used in the password reset functionality at the location 07.levelupctf.com/d41d8cd98f00b204e9800998ecf8427e/8cd98f00b204e9800998/forgot password which had a security question given as “What is the name of your favorite lion at the zoo?” for the user agent_521bcd5. The comments showed San Francisco Zoo and the lion there was called Jahari which is the answer. The function showed temporary password to login. Hence login was successful.

Flag – 4 – FLAG{f514875849460428b4dc40dd72a5a29a} :

The purpose of the login in the first place, was the access to 07.levelupctf.com/radio which showed flag 4 and a secret key to images “pwn4llthebugz”

Flag – 4 to Flag – 5 Transition :

After login was successful it showed mission list and dashboard with target list link which showed nine images of HACK Agents four of which showed four number 1337,415,2099 and 921. The images were sourced from 07.levelupctf.com/agents directory which after enumeration showed that there was an extra image in form of agent87.jpg (Appendix Figure 5). The 07.levelupctf.com/radio showed us a secret to the image which is “pwn4llthebugz”. This pointed towards a steganography.

Flag – 5 – FLAG{f0dee25f617e2cb820b9b44fcdf90ed8} :

The image agent87.jpg was used with tool Steghide and secret key as “pwn4llthebugz” to reveal the file console.txt (Appendix Figure 6). The console.txt showed us the flag.

Flag – 5 to Flag – 6 Transition :

The discovery of four numbers pointed to initial idea that they were ports 1337, 415, 2099 and 921. The four ports were meant to be for port knocking as Nmap revealed that only 80,443,22 were open for 07.levelupctf.com that is new ports were to be externally opened for access. A simple python script (Appendix Code 1) knocked ports in all permutations and combinations possible to open port 3389 which led to next discoveries.

Flag – 6 – FLAG{022d8a7a561a02c371fd7c5ec3e5ea06} :

The opening of the 07.levelupctf.com:3389 in browser revealed a message but nothing else. The enumeration of the directory with dictionary lists led to discovery of 07.levelupctf.com:3389/console which had a python interpreter running. This was powered by Werkzeug which is pointing to Werkzeug Debug RCE. Using the string pattern “__import__(‘os’).popen(‘command_here’).read();” we could execute arbitrary commands in the shell and get the output on the site. This led to discovery of Flag 6 with __import__('os').popen(‘ls’).read(); revealing all the files in the directory with flag.txt as one of them and then __import__('os').popen(‘cat flag.txt’).read(); to reveal the flag. (Appendix Figure 7)

Flag – 6 to Flag – 7 Transition :

The exploring of the file system led to reveal of two files communication.apk and passwords.txt in the /opt directory which contained passwords and usernames of three users (Appendix Figure 8) with Matriarch as one of them which was the main Obelisk user according to previous sites.

Flag – 7 – FLAG{de9aaabc9ede23335e5} :

The user Matriarch credentials are used to login in to the dashboard which revealed a giant poster which contained the last flag in the 07.levelupctf.com/gameover.(Appendix Figure 9)

4. VULNERABILITIES & IMPACTS

Flag 1 - Client Side Logic Issues

By implementing Client Side Logic the user is able to know the logic behind the server by simple reading the JavaScript Code

Impact - User can attack the server as the logic or code is openly available.

Flag 2 - Application code is not obfuscated - OWASP M9

With the link found in JavaScript Code we can download the app and reverse engineer it to find the flag.

Impact - User can steal and sell as their own code or create malicious application

Flag 3 - CWE-1230 Exposure of Sensitive Information through Metadata, REST API Authentication Attack

Adding to header revealed encrypted chat which is REST API authentication attack Metadata in image found in chat revealed flag.

Impact - Data Theft, Data Loss in REST API, Sensitive Information Exposure can provide immense help to the attackers for future.

Flag 4 - Weak Password Reset Functionality, **Sensitive Information Exposure**

This can be broken with dictionary attack based on Users choices. Website revealed flag and key for steganography

Impact - Unauthorized access to systems compromising user accounts.

Flag 5 - Forced Browsing Software Attack, Weak Steganography

With Directory Enumeration, we could find hidden image files which contained secret files which had sensitive information in the form of steganography.

Impact - Sensitive Directory Exposure revealing information useful to attacker.

Flag 6, Flag 7 - Python Debug Remote Code Execution, CWE-256 Unprotected Storage of Credentials

Remote Code Execution is possible with python interpreter running on the site which can be used to run commands on the server and get the output on screen which led to the discovery of passwords.txt plaintext strong passwords file.

Impact - Attackers can execute malicious code remotely on the vulnerable server. Also they can traverse and get all sensitive files on the server.

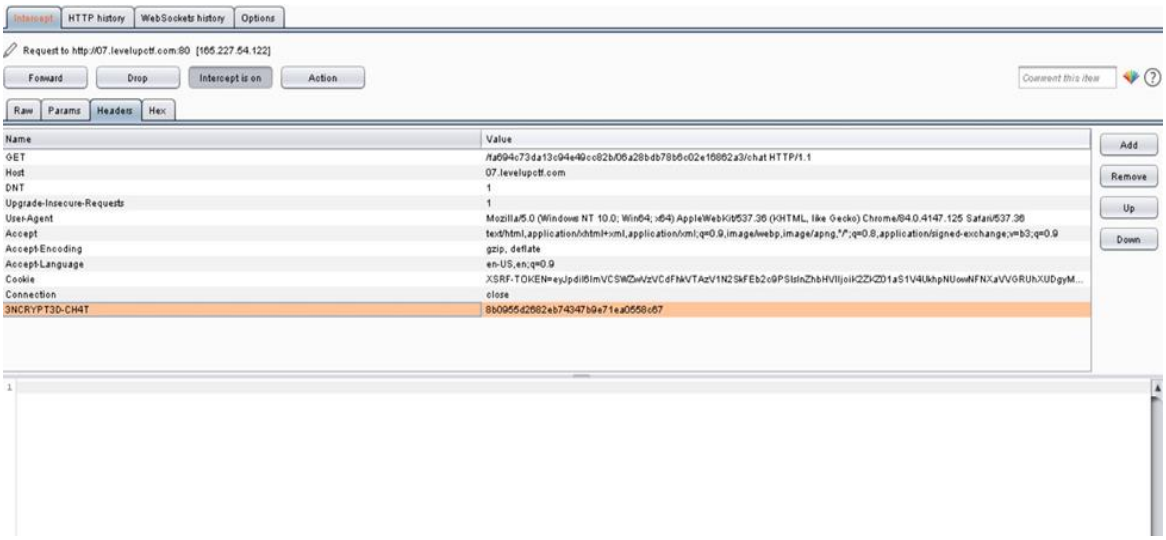
APPENDIX

```
"use strict";
var c = function(t, e) {
    this.$element = p(t), this.$indicators = this.$element.find(".carousel-indicators"), this.options = e, this.paused = null, this.slidi
= null, this.$items = null, this.options.keyboard && this.$element.on("keydown.bs.carousel", p.proxy(this.keydown, this)), "hover" == this.op
document.documentElement) && this.$element.on("mouseenter.bs.carousel", p.proxy(this.pause, this)).on("mouseleave.bs.carousel", p.proxy(this.
};

/*TODO

Add secret backdoor to download app - who looks in JS files anyway?
FLAG{504e8256b8a208845fe18a77e7b304b7}
    function return_app_link() {
 var url = window.location.href;
     if (url.searchParams.get('test')) {
    document.location.href = "https://07.levelupctf.com/222228a4e79d33a299f5d/s3cretc0mmunications";
    }
}
*/

function r(n) {
    return this.each(function() {
        var t = p(this),
            e = t.data("bs.carousel"),
            i = p.extend({}, c.DEFAULTS, t.data(), "object" == typeof n && n),
            o = "string" == typeof n ? n : i.slide;
        e || t.data("bs.carousel", e = new c(this, i)), "number" == typeof n ? e.to(n) : o ? e[o]() : i.interval && e.pause().cycle()
    })
}
```

Figure 1

```
    <string name="abc_searchview_description_clear">Clear query</string>
    <string name="abc_searchview_description_query">Search query</string>
    <string name="abc_searchview_description_search">Search</string>
    <string name="abc_searchview_description_submit">Submit query</string>
    <string name="abc_searchview_description_voice">Voice search</string>
    <string name="abc_shareactionprovider_share_with">Share with</string>
    <string name="abc_shareactionprovider_share_with_application">Share with %s</string>
    <string name="abc_toolbar_collapse_description">Collapse</string>
    <string name="app_name">LevelUp</string>
    <string name="encrypted_chat_key">8b0955d2682eb74347b9e71ea0558c67</string>
    <string name="flag">FLAG{a445c73c8cb97421d1923a8c51c221fd}</string>
    <string name="search_menu_title">Search</string>
    <string name="status_bar_notification_info_overflow">999+</string>
</resources>
```

Figure 2



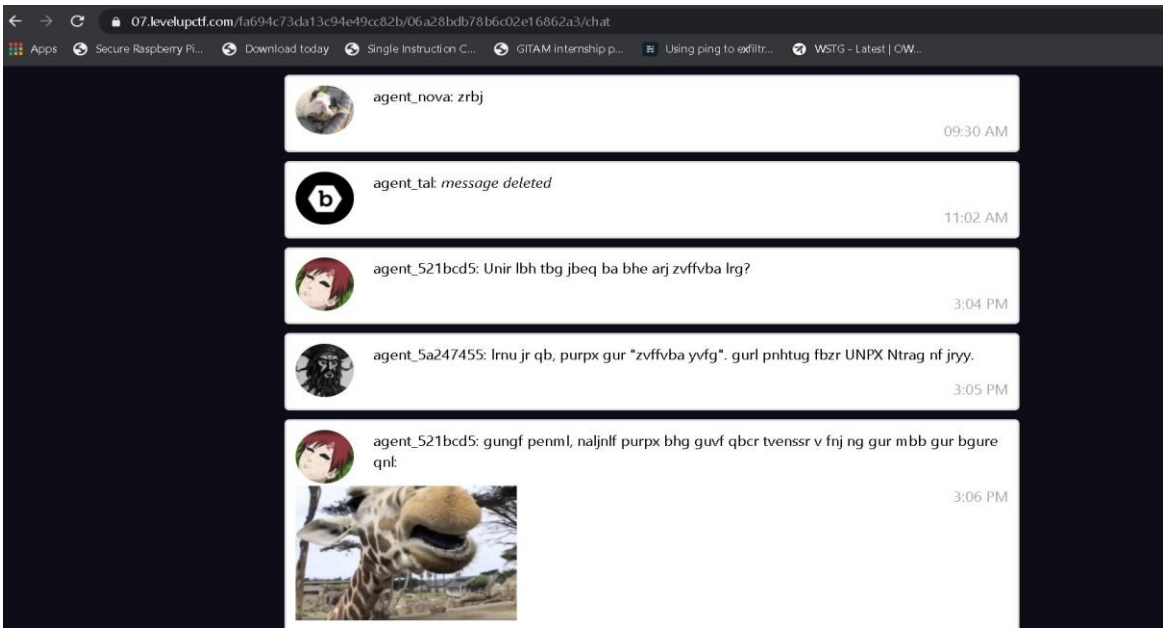


Figure 3`

EXIF

ResolutionUnit	inches
YCbCrPositioning	Centered
ExifVersion	0231
ComponentsConfiguration	Y, Cb, Cr, -
UserComment	FLAG{e8606532b027bfd324ea31d1b4f116c2}
FlashpixVersion	0100
GPSLatitudeRef	North
GPSLatitude	37.732925
GPSLongitudeRef	West
GPSLongitude	122.502355

Figure 4

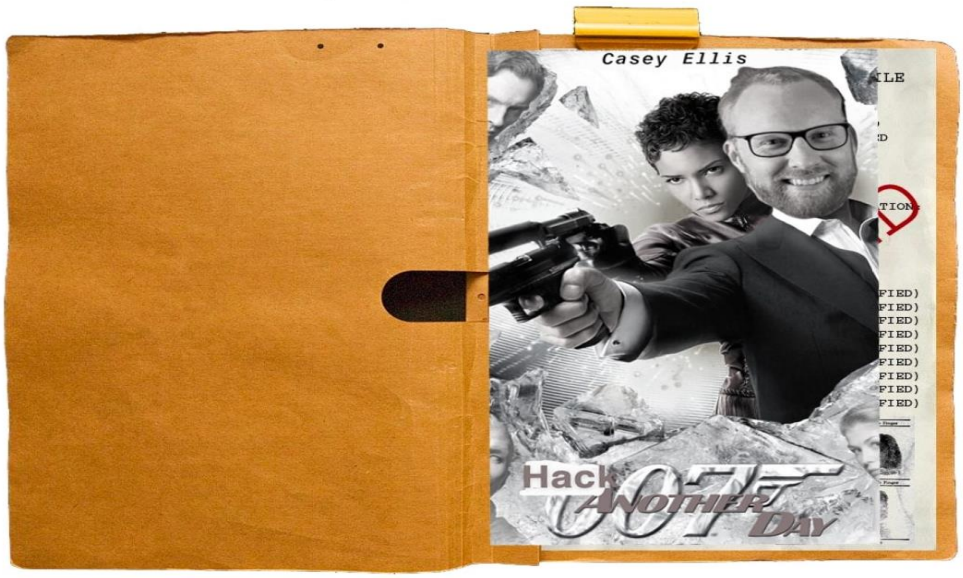


Figure 5

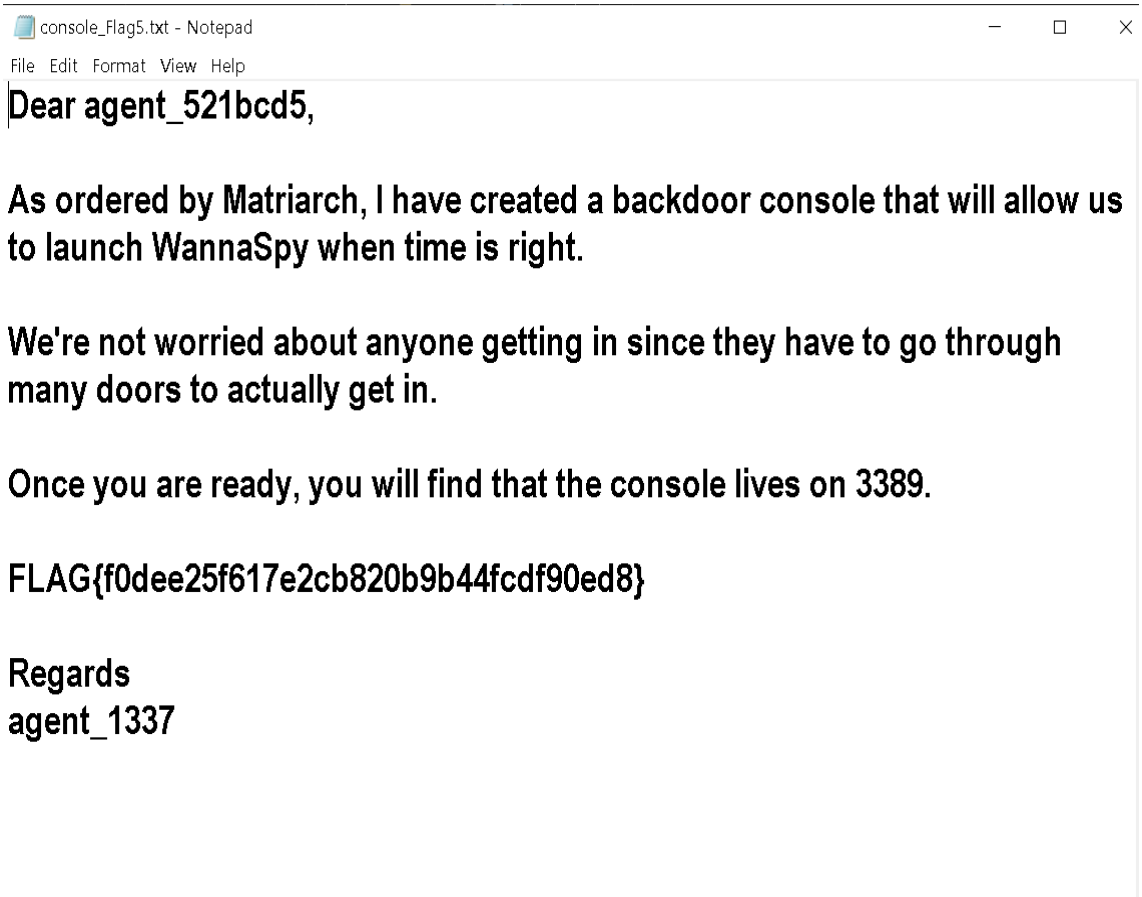


Figure 6

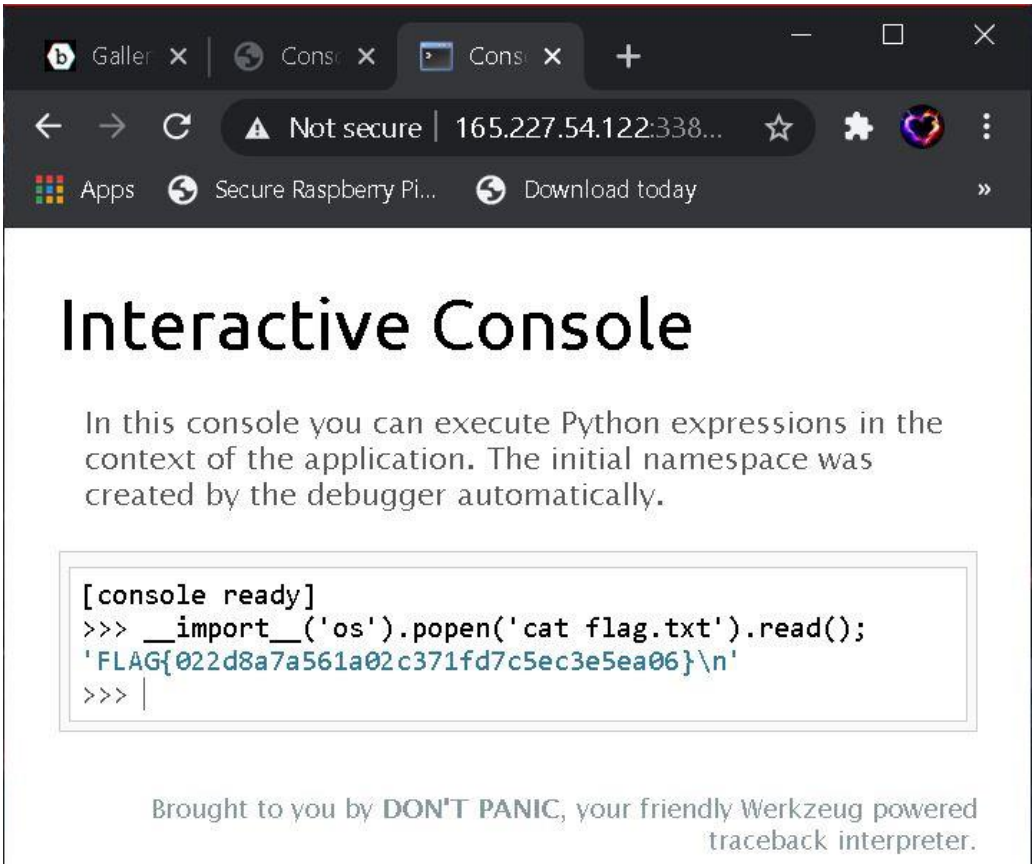


Figure 7

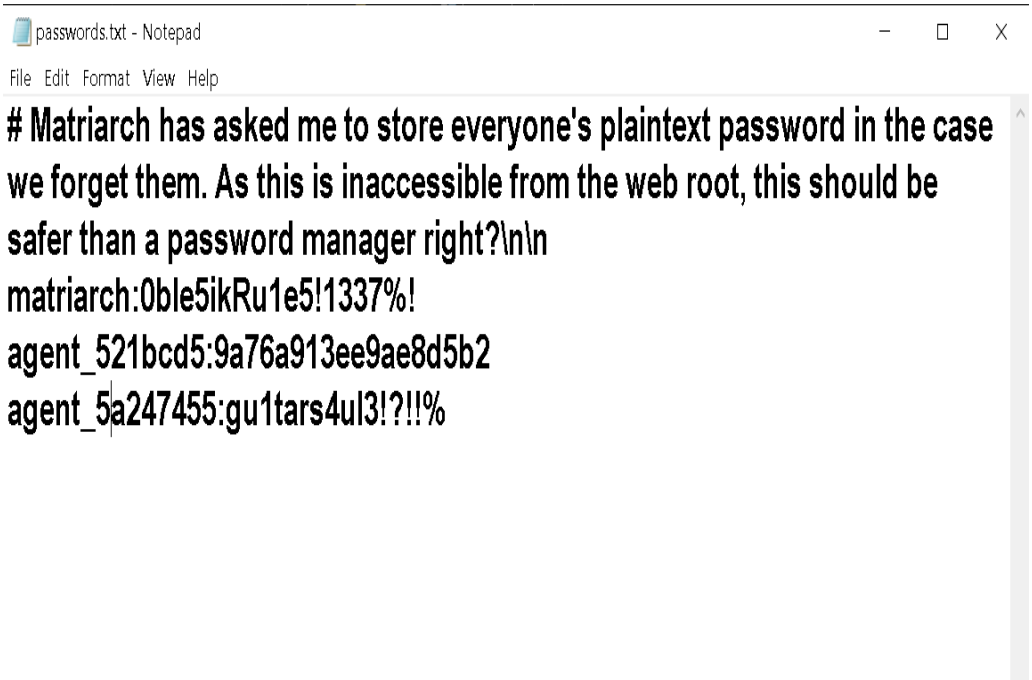


Figure 8

Figure 9

Code 1 – Port Knocking (Python)

```
1   #!/usr/bin/env python
2   from socket import *
3   from itertools import permutations
4   import time
5   ip = "165.227.54.122"          #target IP
6   def Knockports(ports):
7           for port in ports:
8                   try:
9                           print "[] Knocking on port: ", port
10                          s2 = socket(AF_INET, SOCK_STREAM)
11                          s2.settimeout(0.1)          # set timeout in 0.1s
12                          s2.connect_ex((ip, port))
13                          s2.close()
14                  except Exception, e:
```

```
15                                  print "[-] %s" % e
16      def main():
17              r = [1337,415,2099,921]
18              for comb in permutations(r):     # try all the possibility of 4-ports orders
19                      print "\n[] Trying sequence %s" % str(comb)
20                      Knockports(comb)
21              print "[*] Done"
22      main()
```