

# File Extension Base Crypto-Ransomware Vaccine - FEBRAV Project

Haci Emre Dasgin, [haci.dasgin@gmail.com](mailto:haci.dasgin@gmail.com)

Researcher

A Research Paper

September 2022

## **Abstract**

Crypto-ransomware attacks became one of the biggest cybersecurity threats to enterprises and government institutes. Traditional ransomware protection solutions have limited prevention techniques. This study focuses on file extensions to suggest a new ransomware vaccine solution for preventing ransomware encryption. Analysis and test results show that blurring file extension dramatically reduces encrypted files on the system. Therefore, the implications of this study could be used for approaching ransomware prevention differently and developing new methods or tools.

*Keywords:* ransomware, malware, virus, vaccine, vaccination, file extension, encryption, file extension blurring.

## Table of Contents

Abstract.....	2
Introduction .....	4
1. Background .....	6
1.1 Crypto-Ransomware Attack Anatomy .....	6
1.1.1 Initial Compromise .....	6
1.1.2 Infection .....	7
1.1.3 Privilege Escalation and Persistence .....	7
1.1.4 File Encryption.....	8
1.1.5 Payday .....	8
1.2 Traditional Crypto-Ransomware Vaccines.....	9
1.2.1 How Traditional Vaccines Work .....	9
1.2.2 Limitation to Tradition Vaccines .....	10
2. Methodology.....	10
2.1 New File Extension Definition in Windows Registry .....	10
2.1.1 Windows Registry .....	10
2.1.2 Windows Registry – HKEY_CLASSES_ROOT .....	11
2.1.3 HKEY_CLASSES_ROOT – File Extension Definition Keys .....	11
2.1.4 HKEY_CLASSES_ROOT – Application Launcher Keys .....	12
2.1.5 How To Define New File Extension (Registry) .....	13
2.1.6 How To Use New File Extension (File System) .....	13
2.2 File Extension Vaccine Development .....	14
2.3 Vaccine Components .....	14
2.3.1 Inputs .....	14
2.3.2 Random Extension Generator .....	15
2.3.3 Registry Modifier.....	15
2.3.4 File System Modifier .....	15
2.3.5 Vaccine Process.....	16
3. Test.....	19
2.4 How To Test .....	19
2.5 Test.....	20

2.5.1	Preparation .....	20
2.5.2	Execution & Execution Result (without Vaccine) .....	21
2.5.3	Execution & Execution Result (with Vaccine) .....	22
4.	Conclusion .....	24
5.	Appendices .....	24
6.	Reference .....	29

## List of Tables and Figures

Figure 1 - Crypto-Ransomware Attack Anatomy .....	6
Figure 2 - Fake attachment phishing technique (Source: VedaSecure) .....	7
Figure 3 - Embedded File Extension List of LockerGoga Ransomware (source: trendmicro.com) .....	8
Figure 4 - WannaCry Desktop image and Wana Decrypt0r 2.0 (source: ANY.RUN) .....	9
Figure 5 – Microsoft Windows Registry (Regedit App) .....	10
Figure 6 - File Extension Definition Key for Text File .txt and Its Default Value txtfile .....	12
Figure 7 - Application Launcher Key for txtfile which has defined notepad.exe command .....	12
Figure 8 – New Text File Extension Definition (.xyzt) .....	13
Figure 9 – A Text File with File Extension .xyzt - Opened with Notepad As Defined .....	14
Figure 10 – Vaccine Execution Processes .....	16
Figure 11 – Execution of Vaccine (shown provided inputs as action, extension and path) .....	17
Figure 12 – Generated Extension (shows it as token form) .....	17
Figure 13 – Files and Their Extension in Provided Directory (Before Vaccine Execution) .....	17
Figure 14 – Files and Their Extension After Vaccine Execution (Provided directory) .....	17
Figure 15 – Generated Extension Keys in Registry .....	18
Figure 16 – Extension Token .....	18
Figure 17 – Reverting Extension Changes (parameters: action and path) .....	18
Figure 18 – Reverted File Extension (In Provided Directory) .....	18
Figure 19 – Registry After Reverting Vaccine .....	19
Figure 20 – Before Execution .....	20
Figure 21 – Before Execution (File Content) RTF, TXT and ZIP .....	21
Figure 22 – After Execution (No Vaccine) .....	22
Figure 23 – Locking/Vaccinating Files (Specific Directory and Specific Files) .....	22
Figure 24 – Right Before Ransomware Execution .....	22
Figure 25 – After Ransomware Execution (Shown Vaccinated and Not Vaccinated Files) .....	23
Figure 26 – Content of RTF, TXT and ZIP Files (Shown that the content of txt file is encrypted) .....	23

## Introduction

The spread of ransomware is increasing and traditional detection-base protection such as antivirus or anti-malware became ineffective for preventing attacks. Many studies conducted by cybersecurity vendors to develop new ransomware vaccine for combatting infection of ransomware. But none of these vaccines have become effective for all crypto-ransomware. In most cases, traditional ransomware vaccines may only protect a target against current variant. Such vaccines are good short-term solution, once they are released publicly, ransomware authors discover new techniques to quickly bypass them.

Bypassing is simple as ransomware usually checks several points on the system before starting its encryption process, traditional vaccines are targeting those points to change them in order to avoid encryption. Therefore, ransomware author can easily adjust those checks for further attacks, thus vaccines fail against other variant or different ransomware family.

However, this paper aims to provide a new vaccine approach for protecting systems against all crypto-ransomware by focusing file extensions. As file extensions are used for identifying file types on systems therefore, almost all crypto-ransomware have an embedded list of file extension for identifying its target for encryption. It is important to find a way to blur file extension in order to prevent encryption. This paper explains how file extension blurring technique works, how system registry can be utilized for file extension management and eventually how this technique can be used for future cybersecurity solution development.

# 1. Background

## 1.1 Crypto-Ransomware Attack Anatomy

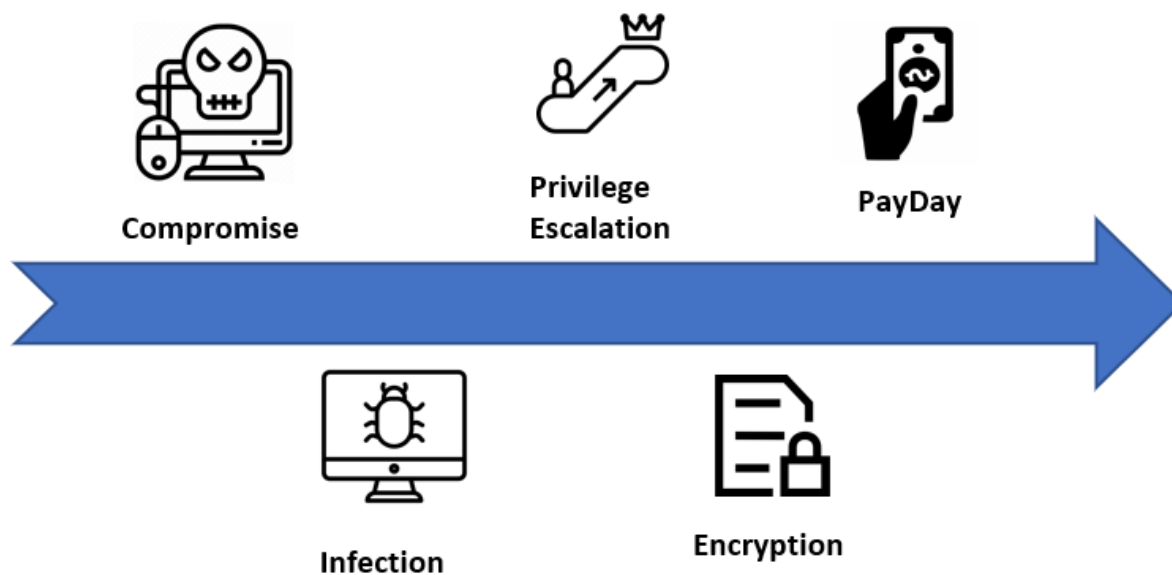


Figure 1 - Crypto-Ransomware Attack Anatomy

### 1.1.1 Initial Compromise

The first step is to distribute the malware to potential victims. Users are either tricked or forced to download the malware or payload via a phishing email, an exploit kit, or a drive-by download and activate it. The malware then initiates the infection. (Vioreanu, 2021)

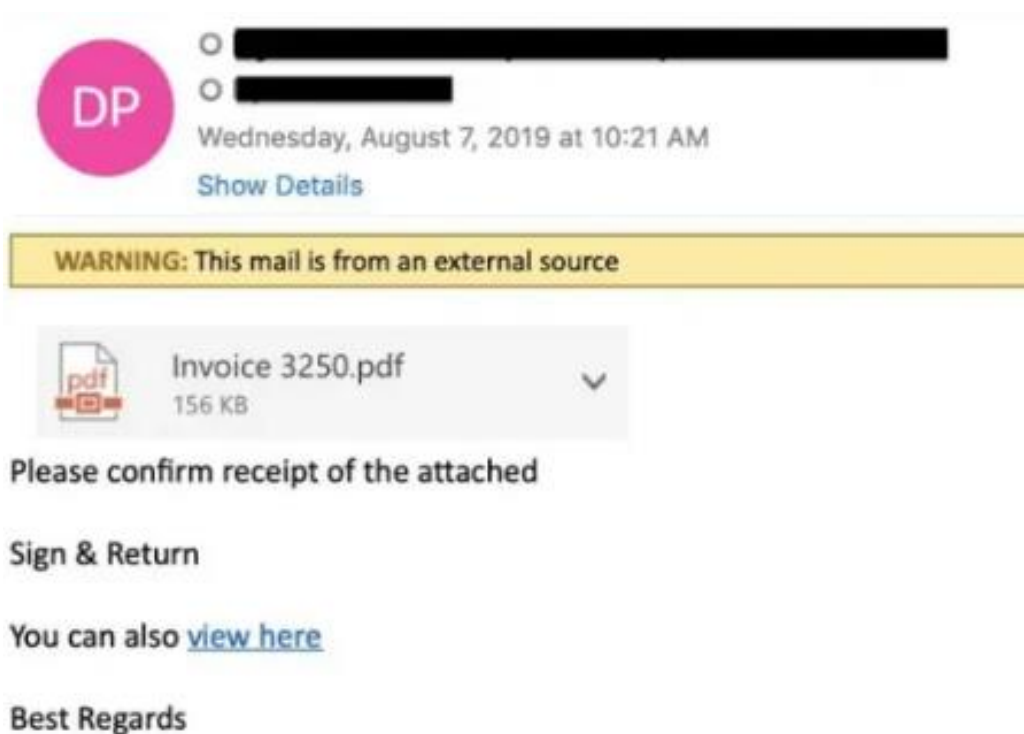


Figure 2 - Fake attachment phishing technique (Source: VedaSecure)

### 1.1.2 Infection

During infection stage, new processes are being launched and malware starts its infection process. Some processes might look like legitimate, but they run from strange locations in the file structure. The infected user will probably not notice much of what's going on.

### 1.1.3 Privilege Escalation and Persistence

In order to gain access to desired files and to make sure that malicious code stays on the victim's device, malware uses several techniques such as:

- Deleting shadow copies and propagate itself through file system
- Modifying configuration files or registry keys for various rights

### 1.1.4 File Encryption

The ransomware starts to encrypt all files it discovered while it's scanning. Then, it uploads the newly encrypted versions and deletes the originals. The ransomware only encrypts specific file types which are pre-defined within ransomware.

aLnk	db '.lnk',0		
	align 4		
aDoc	db '.doc',0		
	align 4	aPpt	db '.ppt',0
aDot	db '.dot',0		align 4
	align 4	aPps	db '.pps',0
aDocx	db '.docx',0		align 4
	align 4	aPot	db '.pot',0
aDocb	db '.docb',0		align 4
	align 4	aPpsx	db '.pps',0
aDotx	db '.dotx',0		align 4
	align 4	aPptx	db '.pptx',0
aDotb	db '.dotb',0		align 4
	align 4	aPosx	db '.posx',0
aWkb	db '.wkb',0		align 4
	align 4	aPotx	db '.potx',0
aXml	db '.xml',0		align 4
	align 4	aSldx	db '.sldx',0
aXls	db '.xls',0		align 4
	align 4	aPdf	db '.pdf',0
aXlsx	db '.xlsx',0		align 4
	align 4	aDb	db '.db',0
aXlt	db '.xlt',0		align 4
	align 4	aSql	db '.sql',0
aXltx	db '.xltx',0		align 10h
	align 4	aCs	db '.cs',0
aXlsb	db '.xlsb',0		align 4
	align 4	aTs	db '.ts',0
aXlw	db '.xlw',0		align 4
	align 4	aJs	db '.js',0
		aPy	db '.py',0

Figure 3 - Embedded File Extension List of LockerGoga Ransomware (source: trendmicro.com)

### 1.1.5 Payday

When the encryption is finish, the victim receives instructions on how to recover their file. A common way to do this is to show a ransom note like “The contents of this machine are encrypted, send us <enter currency here> or other crypto to get your files back”. (PHIPPS, 2022)





Figure 4 - WannaCry Desktop image and Wana Decrypt0r 2.0 (source: ANY.RUN)

## 1.2 Traditional Crypto-Ransomware Vaccines

Traditional vaccines have been used as a method for combatting the rise of ransomware and other malware for several years. (Burt, 2021)

### 1.2.1 How Traditional Vaccines Work

Traditional vaccines use infection markers to make system look like infected in order to avoid real infection from malware. (Constantin, 2016) Traditional malware vaccines are like medical vaccines which apply inactive or weakened parts of viruses to a person in order to protect but unlike medical vaccines, traditional malware vaccines do not improve security response of the system.

There are several methods for creating traditional malware vaccines such as causing error to malware to crash or placing invalid data in specific places to crash malware. Those methods can trick ransomware into not encrypting files. (Hahn, 2022)

### 1.2.2 Limitation to Tradition Vaccines

Traditional malware vaccines have several limitations which makes them unpopular and not user-friendly. (Hahn, 2022)

- Makes system look infected that leaves anti-malware solution in difficult situation for detection.
- Works only for specific malware variant or family.

## 2. Methodology

### 2.1 New File Extension Definition in Windows Registry

#### 2.1.1 Windows Registry

A central hierarchical database used in Windows 98, Windows CE, Windows NT, and Windows 2000 used to store information that is necessary to configure the system for one or

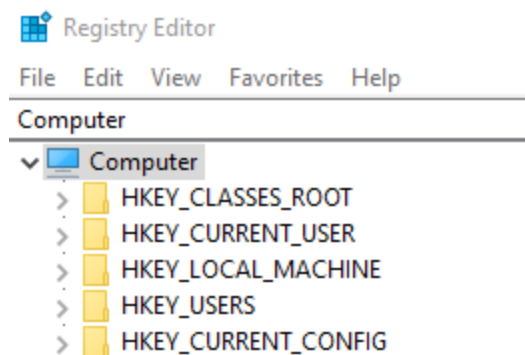


Figure 5 – Microsoft Windows Registry (Regedit App)

more users, applications, and hardware devices. The Registry contains information that Windows continually references during operation, such as profiles for each user, the applications installed on the computer and the types of documents that each can create, property sheet settings for folders and application icons, what hardware exists on the system, and the ports that are being used. (Team, 2022)

### **2.1.2 Windows Registry – HKEY\_CLASSES\_ROOT**

The HKEY\_CLASSES\_ROOT (HKCR) key contains file name extension associations and COM class registration information. The user-specific settings have priority over the default settings. For example, the default setting might specify a particular application to handle .doc files. But a user can override this setting by specifying a different application in the registry.

The *HKEY\_CLASSES\_ROOT* key provides a view of the registry that merges the information from these two sources *HKEY\_LOCAL\_MACHINE\Software\Classes* and *HKEY\_CURRENT\_USER\Software\Classes*. (Registry, 2021)

### **2.1.3 HKEY\_CLASSES\_ROOT – File Extension Definition Keys**

In HKEY\_CLASSES\_ROOT (the first sections of Windows Registry), there are list of keys is used for defining file extensions. Those keys (file extensions) have default value that point other keys in the same section which is basically application launching mapping.

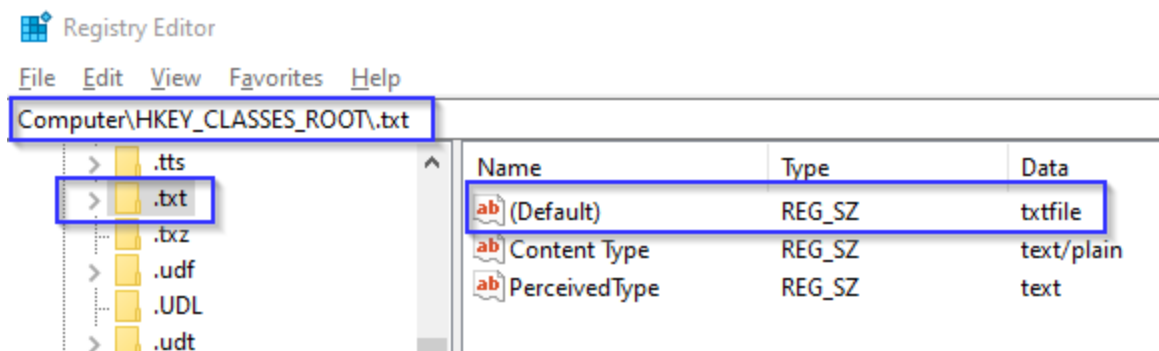


Figure 6 - File Extension Definition Key for Text File .txt and Its Default Value txtfile

#### 2.1.4 HKEY\_CLASSES\_ROOT – Application Launcher Keys

Another important key listed in HKEY\_CLASSES\_ROOT is application launcher keys. Those keys are used for defining commands in order to launch specific application when defined file is tried to open with its defined file extension.

For instance, file extension definition for text files (.txt key) is pointing to txtfile key which has defined command to open application with the file.

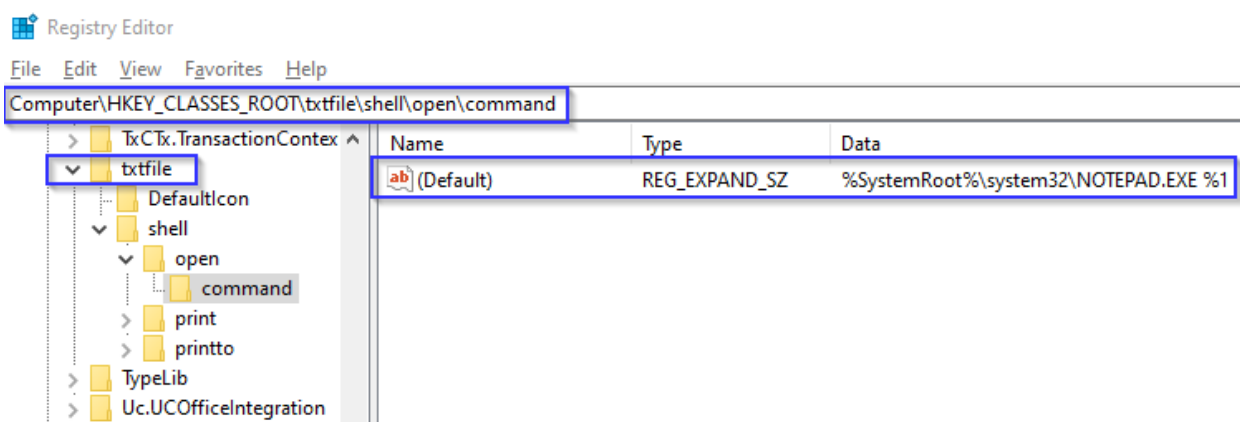


Figure 7 - Application Launcher Key for txtfile which has defined notepad.exe command

### 2.1.5 How To Define New File Extension (Registry)

HKEY\_CLASSES\_ROOT can be used for defining a new file extension. In order to protect a personal file from ransomware attack, a new file extension can be defined in the registry.

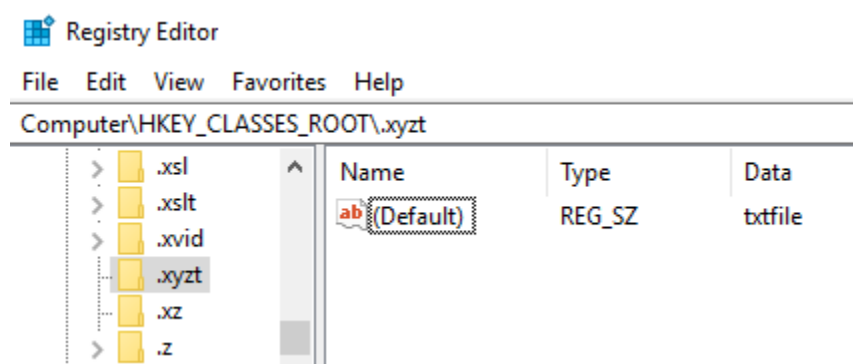


Figure 8 – New Text File Extension Definition (.xyzt)

As it can be seen in *Figure - 8*, a new extension is defined and it is pointed to notepad launcher. This definition can be used instead of “.txt”.

### 2.1.6 How To Use New File Extension (File System)

Defined new file extension can be easily used on the files. For instance, changing a known file extension to something randomly generated and defined in the registry will not cause any issue to daily file usage.

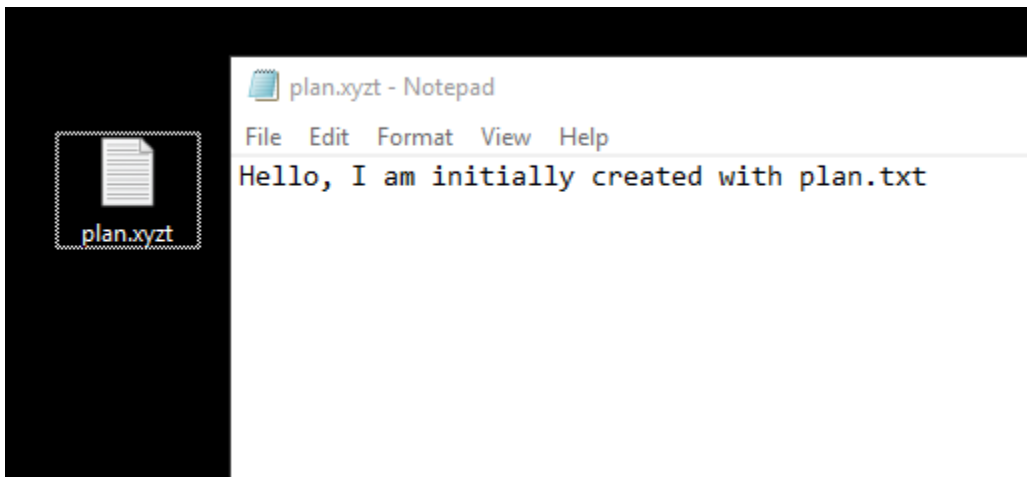


Figure 9 – A Text File with File Extension .xyzt - Opened with Notepad As Defined

As *Figure – 9* shows that defining new file extension in Windows Registry and mapping its application launcher opens the file without any issue.

## 2.2 File Extension Vaccine Development

As crypto-ransomware finds specific files by looking at their file extension and attacking them to encrypt their content therefore it is very important to blur file extension of the files without damaging file execution from end-user perspective.

For instance, in order to protect text files on the system, the file extension can be changed from (.txt) to (.xyzt) for avoiding encryption on text files.

To do this extension change on registry and file system requires proper vaccine development.

## 2.3 Vaccine Components

### 2.3.1 Inputs

It is very important to get specific inputs from vaccine runner such as:

- Extension list, as vaccine should focus only on provided file extension in order to avoid changes on OS system files.
- Directory, as vaccine should only work in provided directory and its sub directories for avoiding changes on OS system files.
- Additionally, vaccine can get action inputs such as 'lock' or 'unlock' for changing extension or reverting changed extensions back.

### **2.3.2 Random Extension Generator**

Random extension generator, reads provided file extension list and generates random file extension for usage in registry and in file system.

### **2.3.3 Registry Modifier**

Registry modifier, it is set of operation for registry modification such as:

- Create Key, creates defined registry keys.
- Set Value of Key, sets values for created keys.
- Get Key, reads value of any keys.
- Delete Key, deleted any keys.

### **2.3.4 File System Modifier**

File system modifier, it modifies extension of the files which are located in the provided directory.

### 2.3.5 Vaccine Process

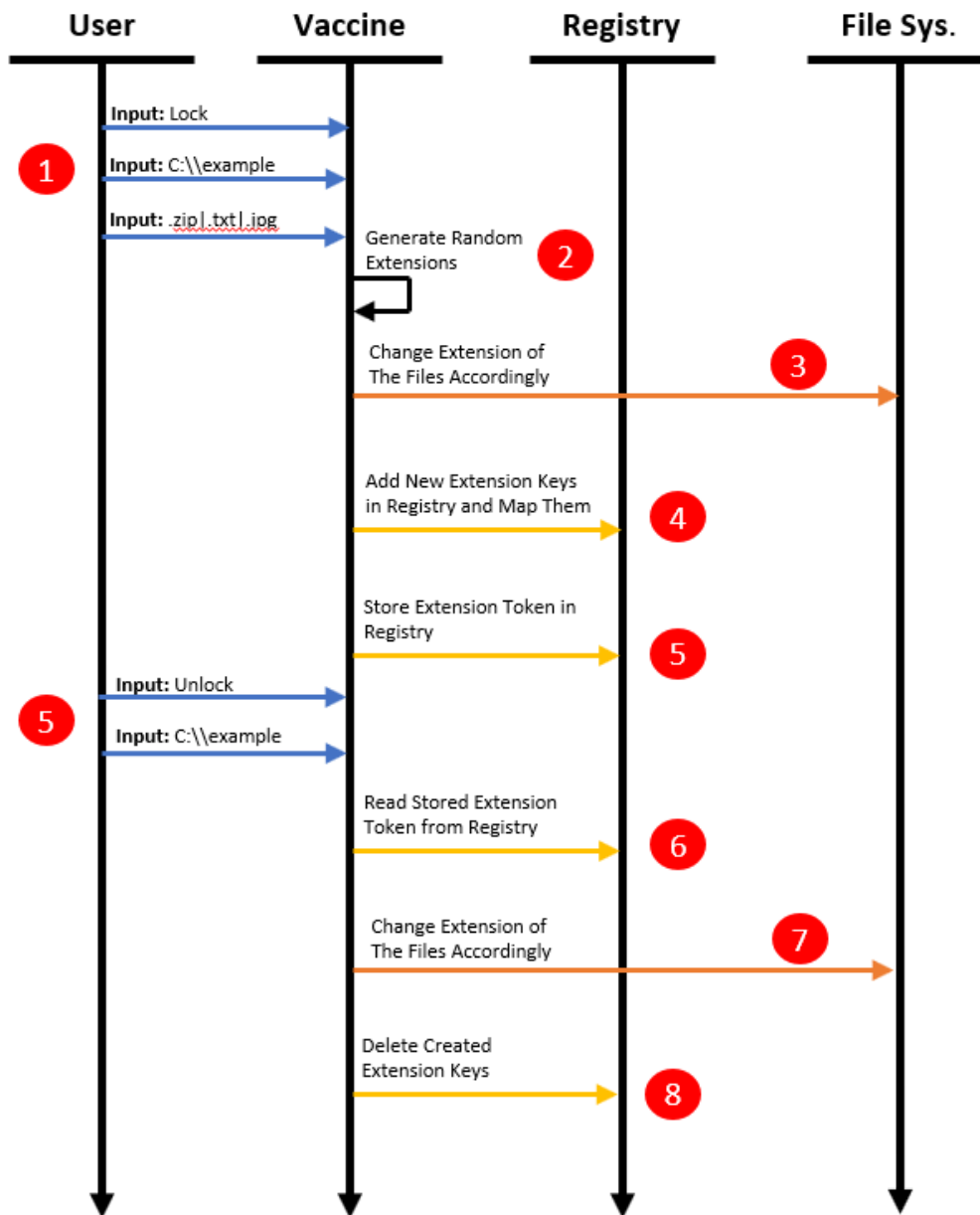


Figure 10 – Vaccine Execution Processes

As it is shown in Figure – 10, Vaccine has several processes for modify system and reverting modification back.

1. **(1)** – Inputs where user provides to vaccine script/application.



```
python ex_bluring.py --action lock --extension ".zip|.rtf|.txt" --path C:\\Users\\REM\\Desktop\\Test
```

Figure 11 – Execution of Vaccine (shown provided inputs as action, extension and path)

2. (2) – Vaccine reads provided extension list and generates random extensions and also keeps them as extension token

```
zip@.0A3uA7OZ|.rtf@.1Oze2GI6|.txt@.2e5F456
```

Figure 12 – Generated Extension (shows it as token form)

3. (3) – It reads extension token and change extension of files in provided directory.






	pdf_file_format.pdf	PDF File	8/22/2010 3:14 PM	488 KB
	png_file_format.png	PNG File	3/26/2021 2:15 PM	2 KB
	rich_text_format.rtf	Rich Text Document	9/24/2022 7:08 PM	1 KB
	text_file_format.txt	TXT File	9/24/2022 7:10 PM	0 KB
	zip_file_format.zip	zip Archive	9/24/2022 7:10 PM	1 KB

Figure 13 – Files and Their Extension in Provided Directory (Before Vaccine Execution)






	pdf_file_format.pdf	PDF File	8/22/2010 3:14 PM	488 KB
	png_file_format.png	PNG File	3/26/2021 2:15 PM	2 KB
	rich_text_format.1Oze2GI6	Rich Text Document	9/24/2022 7:08 PM	1 KB
	text_file_format.2e5F4560	Notepad++ Docu...	9/24/2022 7:10 PM	0 KB
	zip_file_format.0A3uA7OZ	zip Archive	9/24/2022 7:10 PM	1 KB

Figure 14 – Files and Their Extension After Vaccine Execution (Provided directory)

4. (4) – Generated extension keys in registry, which helps OS to open proper application with the files that got effected from vaccine.

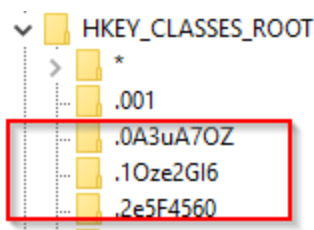


Figure 15 – Generated Extension Keys in Registry

5. (5) – After vaccine execution, script/app store extension token in the registry for reverting back, this token also printed on the screen after execution so that user can copy and secure it for reverting system back.

Computer\HKEY_CLASSES_ROOT\extension_log			
Explorer.ZipSele	Name	Type	Data
ExplorerCLSIDFla	(Default)	REG_SZ	(value not set)
extension_log	log	REG_SZ	.zip@.0A3uA7OZ .rtf@.1Oze2GI6 .txt@.2e5F4560
Extensions			

Figure 16 – Extension Token

6. (6) – Reverting all changes back by executing vaccine script/app with similar parameters.

```
python ex_bluring.py --action unlock --path C:\\Users\\REM\\Desktop\\Test
```

Figure 17 – Reverting Extension Changes (parameters: action and path)

7. (7) – All modified extensions reverted back.

pdf_file_format.pdf	PDF File	8/22/2010 3:14 PM	488 KB
png_file_format.png	PNG File	3/26/2021 2:15 PM	2 KB
rich_text_format.rtf	Rich Text Document	9/24/2022 7:08 PM	1 KB
text_file_format.txt	TXT File	9/24/2022 7:10 PM	0 KB
zip_file_format.zip	zip Archive	9/24/2022 7:10 PM	1 KB

Figure 18 – Reverted File Extension (In Provided Directory)

8. (8) – Generated keys deleted in the registry.

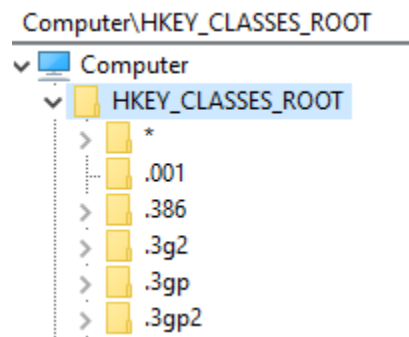


Figure 19 – Registry After Reverting Vaccine

## 3. Test

### 2.4 How To Test

In order to prove effectiveness of the crypto-ransomware vaccine, it must be tested with real ransomware and the testing methodology will be simple as creating certain files with proper content and certain file extension in a specific directory. After preparing files, the vaccine will be applied. In order to show ransomware functionality, the test will have ransomware execution result with vaccine and without vaccine.

## 2.5 Test

### 2.5.1 Preparation

As below table shows that there will three different files in test directory. All test will be done on Windows 10 operating system and the ransomware will be a variant of WannaCry.

<b>Operating System</b>	Windows 10
<b>Test Directory</b>	C:\Users\<user>\Desktop\ptest
<b>Text File</b>	txt_file_format.txt
<b>Rich Text File</b>	rtf_file_format.pdf
<b>Zip File</b>	zip_file_format.zip
<b>Ransomware</b>	WannaCry
<b>Ransomware MD5</b>	84C82835A5D21BBCF75A61706D8AB549

As shown below, three files (*rtf\_file\_format.rtf*, *text\_file\_format.txt* and *zip\_file\_format.zip*) have been created in defined directory (*C:\Users\<user>\Desktop\ptest*)

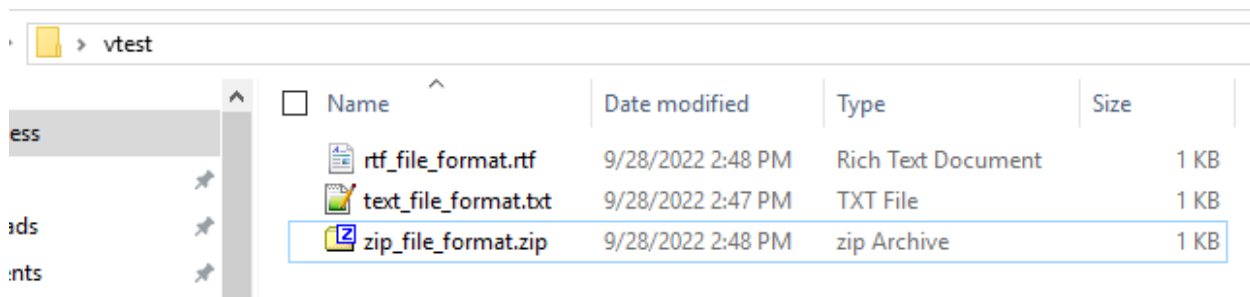


Figure 20 – Before Execution

These three files have a content in order to see encryption, text files (rtf and txt) have a content written as *“This is a <rtf or text> file with this specific content!”* and zip file just contains the copy of txt and rtf files.

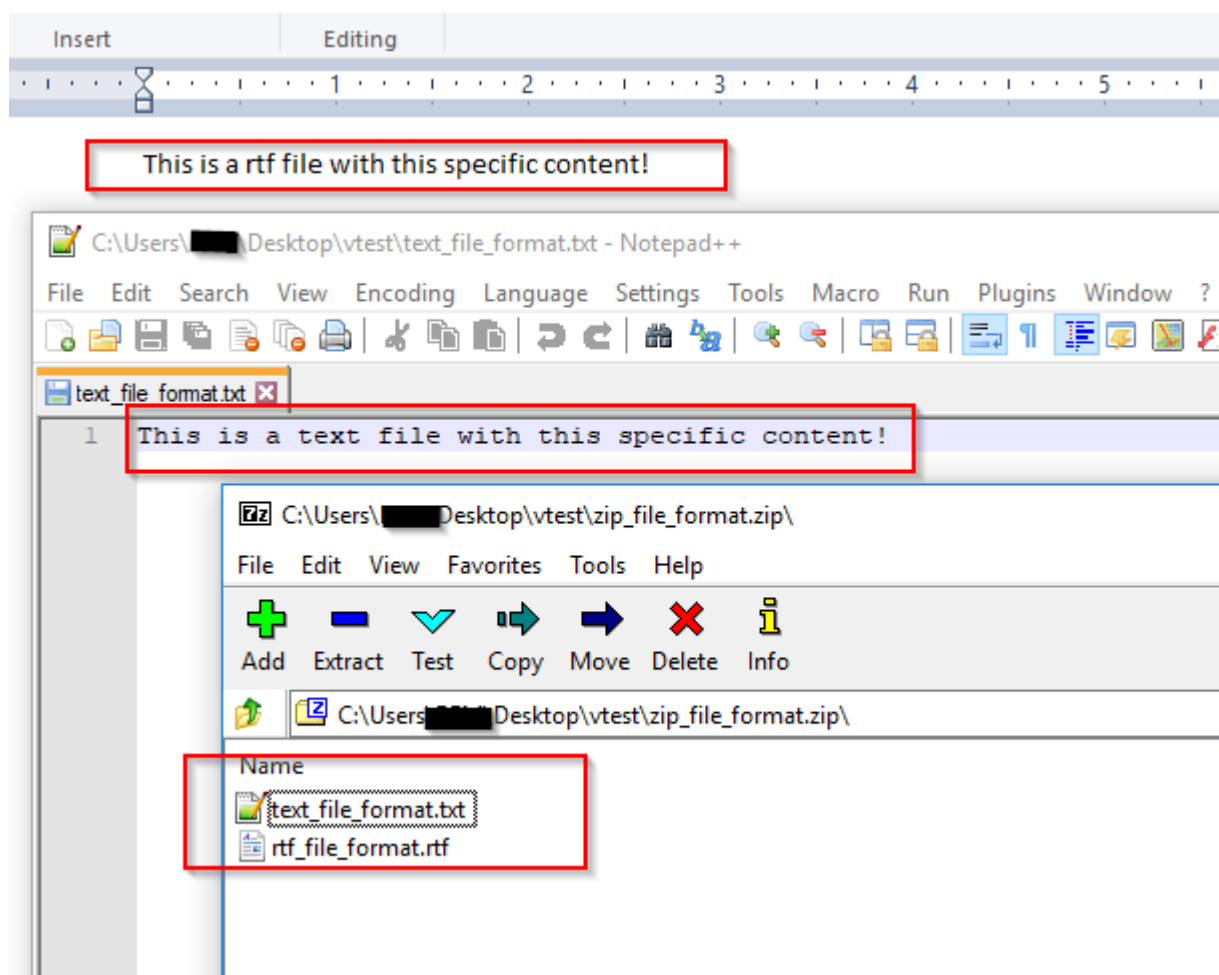


Figure 21 – Before Execution (File Content) RTF, TXT and ZIP

### 2.5.2 Execution & Execution Result (without Vaccine)

In this execution, ransomware will be executed without any vaccination. The result can be seen in below Figure.

Name	Date modified	Type	Size
@Please_Read_Me@.txt	9/28/2022 3:10 PM	TXT File	1 KB
@WanaDecryptor@.exe	9/28/2022 3:10 PM	Shortcut	1 KB
rtf_file_format.rtf.WNCRY	9/28/2022 2:48 PM	WNCRY File	1 KB
text_file_format.txt.WNCRY	9/28/2022 2:47 PM	WNCRY File	1 KB
zip_file_format.zip.WNCRY	9/28/2022 2:48 PM	WNCRY File	1 KB

Figure 22 – After Execution (No Vaccine)

As result shows that prepared files were encrypted and even the extension of executed ransomware “WNCRY” is shown. Note that two files were created by the ransomware.

### 2.5.3 Execution & Execution Result (with Vaccine)

Firstly, defined file extensions will be vaccinated, these extensions are “.zip” and “.rtf” as shown below figure. Note developed python script can be found in appendix section below.

```
python ex_v.py --action lock --extension ".zip|.rtf" --path C:\\Users\\[redacted]\\Desktop\\vtest
.zip@.0i8Y7wTa|.rtf@.18wjHS8
```

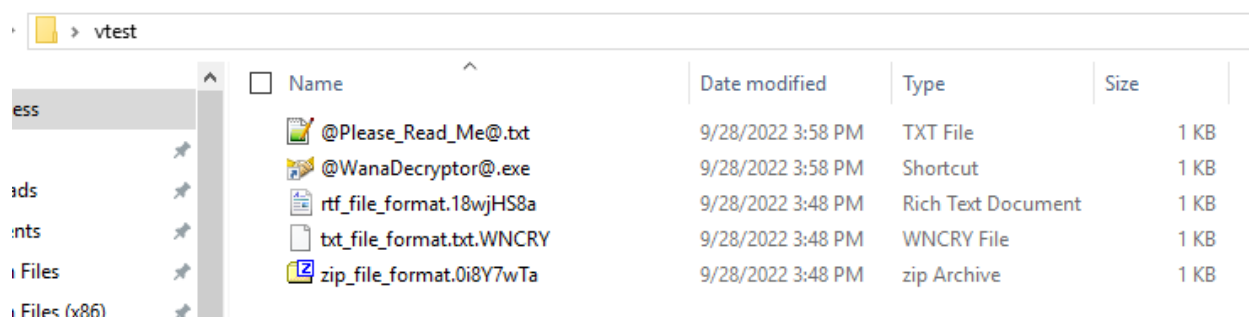
Figure 23 – Locking/Vaccinating Files (Specific Directory and Specific Files)

After vaccination, only two files are now having different “random” extension. On purpose, one file, text file (extension is .txt) will be left without vaccine to show ransomware infection in the folder.

Name	Date modified	Type	Size
rtf_file_format.18wjHS8a	9/28/2022 3:48 PM	Rich Text Document	1 KB
txt_file_format.txt	9/28/2022 3:48 PM	TXT File	1 KB
zip_file_format.0i8Y7wTa	9/28/2022 3:48 PM	zip Archive	1 KB

Figure 24 – Right Before Ransomware Execution

After ransomware execution, as it can be seen in below Figure that already vaccinated files “.rtf” and “.zip” are not infected, at least the file extension looks like has not been changed.



Name	Date modified	Type	Size
@Please_Read_Me@.txt	9/28/2022 3:58 PM	TXT File	1 KB
@WanaDecryptor@.exe	9/28/2022 3:58 PM	Shortcut	1 KB
rtf_file_format.18wjHS8a	9/28/2022 3:48 PM	Rich Text Document	1 KB
txt_file_format.txt.WNCRY	9/28/2022 3:48 PM	WNCRY File	1 KB
zip_file_format.0i8Y7wTa	9/28/2022 3:48 PM	zip Archive	1 KB

Figure 25 – After Ransomware Execution (Shown Vaccinated and Not Vaccinated Files)

If file contents are reviewed, the result is the same. it can be seen in below figure that not vaccinated file (.txt file in this case) has encrypted content.

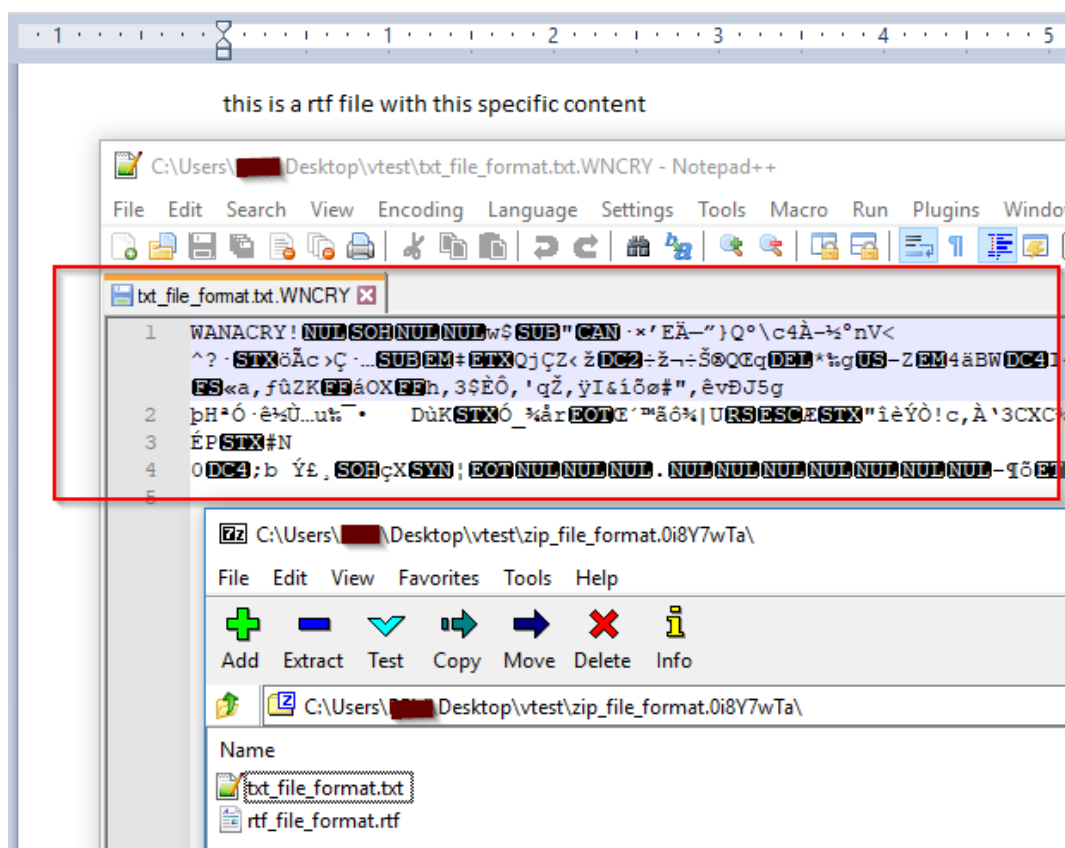


Figure 26 – Content of RTF, TXT and ZIP Files (Shown that the content of txt file is encrypted)

## 4. Conclusion

As it can be seen from the result that this new vaccine method is effective and it has a very high protection rate against crypto-ransomware attack.

Without any doubt, there will new methods from crypto-ransomware authors to bypass this new generic vaccine method, but in general, this research aims to show the ways to approach this problem. As this method is just one window to look at this problem, there can be many developments or fixes over this prove of concept as such, securing extension token in a better way, instead of extension blurring, magic byte blurring can be used or random extension generation can be applied each time the OS boots up.

## 5. Appendices

----- Python Script Starts-----

```
import os
import random
import string
import winreg
import argparse

class RegManager():

    def __init__(self, unit, type_=winreg.REG_SZ):
        if unit == "HCR":
            self.root = winreg.HKEY_CLASSES_ROOT
        elif unit == "HCU":
            self.root = winreg.HKEY_CURRENT_USER
        elif unit == "HLM":
            self.root = winreg.HKEY_LOCAL_MACHINE
        elif unit == "HU":
            self.root = winreg.HKEY_USERS
        elif unit == "HCC":
            self.root = winreg.HKEY_CURRENT_CONFIG
```



```

else:
    pass
self.type_ = type_

def _is_key_exist(self, key):
    try:
        handle = winreg.ConnectRegistry(None, self.root)
        winreg.OpenKeyEx(handle, key, 0, winreg.KEY_ALL_ACCESS)
        winreg.CloseKey(handle)
        return True
    except FileNotFoundError:
        return False

def _open_key(self, key):
    if self._is_key_exist(key):
        handle = winreg.ConnectRegistry(None, self.root)
        return winreg.OpenKeyEx(handle, key, 0, winreg.KEY_ALL_ACCESS)
    else:
        return 1

def create_key(self, sub_key):
    if self._is_key_exist(sub_key):
        key_handle = self._open_key(sub_key)
    else:
        key_handle = winreg.CreateKeyEx(self.root, sub_key, 0, winreg.KEY_ALL_ACCESS)
    return key_handle

def set_key_value(self, sub_key, value):
    if self._is_key_exist(sub_key):
        winreg.SetValue(self.root, sub_key, self.type_, value)
    else:
        self.create_key(sub_key)
        winreg.SetValue(self.root, sub_key, self.type_, value)
    return 0

def set_subkey_value(self, key, value_name, value):
    if self._is_key_exist(key):
        winreg.SetValueEx(self._open_key(key), value_name, 0, self.type_, value)
    else:
        self.create_key(key)
        winreg.SetValueEx(self._open_key(key), value_name, 0, self.type_, value)
    return 0

def get_key(self, key, value_name):

```

```

if self._is_key_exist(key):
    raw_value = winreg.QueryValueEx(self._open_key(key), value_name)
    return raw_value[0]
else:
    return 1

def delete_key(self, sub_key):
    if self._is_key_exist(sub_key):
        winreg.DeleteKey(self.root, sub_key)
        return 0
    else:
        return 1

class ExtensionChanger():

    def __init__(self, path, ex_def):
        self.path = path
        self.ex_def = ex_def

    def set_new_extension(self):
        for root, subFolders, files in os.walk(self.path):
            for file in files:
                full_filename = os.path.join(root, file)
                fname = os.path.splitext(full_filename)[0]
                old_extension = os.path.splitext(full_filename)[1]
                for i in self.ex_def.split("|"):
                    if i.split("@")[0] == old_extension:
                        os.rename(full_filename, fname + i.split("@")[1])

    def set_old_extension(self):
        for root, subFolders, files in os.walk(self.path):
            for file in files:
                full_filename = os.path.join(root, file)
                fname = os.path.splitext(full_filename)[0]
                old_extension = os.path.splitext(full_filename)[1]
                for i in self.ex_def.split("|"):
                    if i.split("@")[1] == old_extension:
                        os.rename(full_filename, fname + i.split("@")[0])

class RandomStringGen():
    def __init__(self, length):
        self.length = length

    def get_random_string(self):

```

```
# abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
letters = string.digits+string.ascii_letters+string.digits
result_str = ''.join(random.choice(letters) for i in range(self.length))
return result_str
```

```
class Manager():
```

```
    def __init__(self, dir, ext):
        self.directory = dir
        self.extension_list = ext
        self.ran_str = RandomStringGen(7)

    def start_change(self):
        change_keeper = str()
        # Extension changing
        for i in range(len(self.extension_list.split("|"))):
            change_keeper = change_keeper + self.extension_list.split("|")[i] + "@"
            new_ext = self.ran_str.get_random_string()
            change_keeper = change_keeper + "." + str(i) + new_ext + "|"
        change_keeper = change_keeper[:-1]
        ext_chgr = ExtensionChanger(self.directory, change_keeper)
        ext_chgr.set_new_extension()

        # Registry changing
        reg_man = RegManager("HCR")
        for i in change_keeper.split("|"):
            # Initially clean registry key any exists
            reg_man.delete_key("\\\\"+i.split("@")[1])

            try:
                # Get old extension set program
                used_app = reg_man.get_key("\\\\"+i.split("@")[0], "")
            except FileNotFoundError:
                pass

            # Create Key
            reg_man.create_key(i.split("@")[1])

            # Set default value
            reg_man.set_subkey_value("\\\\"+i.split("@")[1], "", used_app)

        reg_man.create_key("extension_log")
        reg_man.set_subkey_value("\\\\"+extension_log, "log", change_keeper)
```

```

print(change_keeper[:-1])
# .zip@.01y7E4z1|.exe@.1Z9F3Jk|.txt@.2THORC8P|.pdf@.3R6ZvH0x

def revert_change(self):
    # Read log
    reg_man = RegManager("HCR")
    extension_log = reg_man.get_key("\\extension_log", "log")
    # Change back
    ext_chgr = ExtensionChanger(self.directory, extension_log)
    ext_chgr.set_old_extension()

    for i in extension_log.split("|"):
        reg_man.delete_key("\\"+i.split("@")[1])

def runner(args):
    agent = Manager(args.path, args.extension)
    if args.action == 'lock':
        agent.start_change()
    elif args.action == 'unlock':
        agent.revert_change()
    else:
        print("ERROR - Something went wrong!")

def main():
    my_parser = argparse.ArgumentParser()

    my_parser.add_argument('--extension', type=str, help="Provide your file extension to secure: '--extension .zip|.pdf|.txt' (required only with [action lock])")

    my_parser.add_argument('--action', type=str, help="Provide your action over file(s): '--action lock/unlock' (required)")

    my_parser.add_argument('--path', type=str, help="Provide your full path: '--path C:\\Users\\<username>\\Desktop' (required)")

    args = my_parser.parse_args()

    runner(args)

if __name__ == "__main__":
    main()
----- Python Script Ends-----

```

## 6. Reference

- Burt, J. (2021). Retrieved from <https://www.esecurityplanet.com/threats/cybersecurity-vaccines-ransomware-vulnerability-defense/>
- Constantin, L. (2016). Retrieved from <https://www.pcworld.com/article/420295/free-bitdefender-tool-prevents-locky-other-ransomware-infections-for-now.html>
- Hahn, K. (2022). Retrieved from <https://www.gdatasoftware.com/blog/2022/01/malware-vaccines>
- PHIPPS, G. (2022). Retrieved from <https://www.cybergRX.com/resources/research-and-insights/blog/know-the-phases-of-a-ransomware-attack#:~:text=The%20basic%20kill%20chain%20phases,demanded%2C%20your%20options%20become%20limited>
- Registry, W. (2021). Retrieved from <https://learn.microsoft.com/en-us/windows/win32/sysinfo/hkey-classes-root-key>
- Team, M. W. (2022). Retrieved from <https://learn.microsoft.com/en-us/troubleshoot/windows-server/performance/windows-registry-advanced-users>
- Vioreanu, D. (2021). Retrieved from [https://www.cyberghostvpn.com/en\\_US/privacyhub/how-a-ransomware-works-and-how-to-prevent-it/](https://www.cyberghostvpn.com/en_US/privacyhub/how-a-ransomware-works-and-how-to-prevent-it/)