Cyber Security Project

Topic.no: 5

Topic: Malware Analysis and Reverse Engineering

Members:

20BCE7186 -> K. Mourya Achuth (VIT-AP)

20BCE7303 -> Manikanta Meduri (VIT-AP)

20BCE7304 -> R. Sai Kamal Teja (VIT-AP)

20BCN7084 -> CH. Hemanth Sai (VIT-AP)

Report of Team 561

Submitted to SmartInternz

Ransomware attack- How it is done and how to recover

Code for ransomware attack

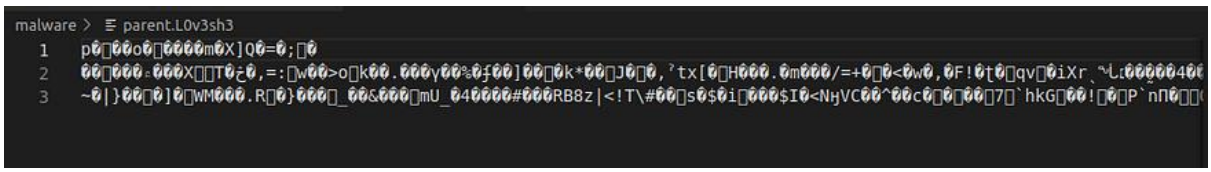
```
import
os
        from pathlib import Path
        from Crypto.PublicKey import RSA
        from Crypto.Cipher import PKCS1_OAEP, AES




        privateKeyFile = 'private.pem'




        def scanRecurse(baseDir):
            '''
            Scan a directory and return a list of all
        files
            return: list of files
            '''
            for entry in os.scandir(baseDir):
                if entry.is_file():
                    yield entry
                else:
                    yield from scanRecurse(entry.path)




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

```
    # read private key from file
    extension = dataFile.suffix.lower()
    with open(privateKeyFile, 'rb') as f:
        privateKey = f.read()
        # create private key object
        key = RSA.import_key(privateKey)


    # read data from file
    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) ]


    # decrypt the session key
    cipher = PKCS1_OAEP.new(key)
    sessionKey =
cipher.decrypt(encryptedSessionKey)


    # decrypt the data with the session key
    cipher = AES.new(sessionKey, AES.MODE_EAX,
nonce)
    data = cipher.decrypt_and_verify(ciphertext,
tag)


    # save the decrypted data to file
    dataFile = str(dataFile)
    fileName= dataFile.split(extension)[0]
    fileExtension = '.decrypted' # mark the file
was decrypted
    decryptedFile = fileName + fileExtension
    with open(decryptedFile, 'wb') as f:
        f.write(data)


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

```
directory = './' # CHANGE THIS


# BONUS for you
dir = input('put your directory (default is "./"
):')
if dir:
  directory = dir




# because we need to decrypt file focus on
.L0v3sh3 extension here is the code
includeExtension = ['.l0v3sh3'] # CHANGE THIS
make sure all is lower case


for item in scanRecurse(directory):
    filePath = Path(item)
    fileType = filePath.suffix.lower()
    # run the decryptor just if the extension is
.l0v3sh3
    if fileType in includeExtension:
      #print(Path(filePath)) # testing the
scanning file
      decrypt(filePath, privateKeyFile)
```

How it works

After encryption



Decryption code

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




        privateKeyFile = 'private.pem'




        def scanRecurse(baseDir):
            '''
            Scan a directory and return a list of all
        files
            return: list of files
            '''
            for entry in os.scandir(baseDir):
                if entry.is_file():
                    yield entry
                else:
                    yield from scanRecurse(entry.path)




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


            # read private key from file
            extension = dataFile.suffix.lower()
            with open(privateKeyFile, 'rb') as f:
                privateKey = f.read()
                # create private key object
```

```
        key = RSA.import_key(privateKey)


    # read data from file
    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) ]


    # decrypt the session key
    cipher = PKCS1_OAEP.new(key)
    sessionKey =
cipher.decrypt(encryptedSessionKey)


    # decrypt the data with the session key
    cipher = AES.new(sessionKey, AES.MODE_EAX,
nonce)
    data = cipher.decrypt_and_verify(ciphertext,
tag)


    # save the decrypted data to file
    dataFile = str(dataFile)
    fileName= dataFile.split(extension)[0]
    fileExtension = '.decrypted' # mark the file
was decrypted
    decryptedFile = fileName + fileExtension
    with open(decryptedFile, 'wb') as f:
        f.write(data)


    print('Decrypted file saved to ' +
decryptedFile)


directory = './' # CHANGE THIS


# BONUS for you
```

```
dir = input('put your directory (default is "./"
):')
if dir:
  directory = dir




# because we need to decrypt file focus on
.L0v3sh3 extension here is the code
includeExtension = ['.l0v3sh3'] # CHANGE THIS
make sure all is lower case


for item in scanRecurse(directory):
    filePath = Path(item)
    fileType = filePath.suffix.lower()
    # run the decryptor just if the extension is
.l0v3sh3
    if fileType in includeExtension:
      #print(Path(filePath)) # testing the
scanning file
      decrypt(filePath, privateKeyFile)
```