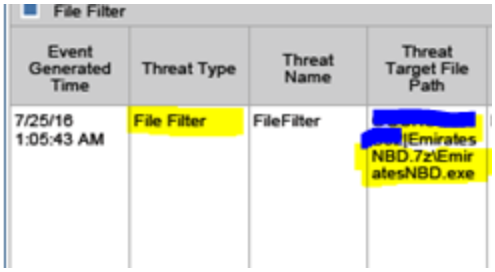
Luminosity RAT - Re-purposed

malwarenailed.blogspot.com/2016/07/luminosity-rat-re-purposed.html

malwarenailed



So I came across a sample which was sent inside a .7z file and strangely was detected by a file filter and not any spam or antivirus filter. The file was interestingly named as EmiratedNBD.exe, which indicates that this attack is targeted to the region as Emirates NBD is one of the biggest financial entities in the GCC region. The sample was not identified by any antivirus and also unknown to Virustotal till today.

MD5: 9b2da7bfb9dedaba7e4d14d623081d7f
SHA1: cfa3ce2a7743181775870d00f4f418efdd737a31

Moreover, the file seem encrypted as there were not many strings found and only a few libraries were visible in the import directory. The sample is coded in .NET C#.

I have performed basic static and dynamic analysis of the sample and inital findings strongly indicates that the sample is an encrypted payload of Luminosity RAT which is infecting the endpoint and performing a C2 communication with its server (C2 ip address: 204.45.103.37). Till date there is not much OSTI available on this ip address.

The sample when starts as a process, seems to start a child process in suspended mode and writes its memory space with the decrypted payload. The decrypted payload is responsible for all Botnet communication with the C2 server. The original parent process terminates after spawning the child process. I was able to extract many useful strings from the memory of the running child process. I also noticed another artifact left by the sample "explorer.exe" in the temp folder and placed in the startup as an IE shortcut.

The sample bot beaconing is periodic and seems to be utilizing a custom network protocol whereby destination port is 19881 and the information sent to the C2 server comprise of information on analysis tools running and the GUI processes which are running, the domain and user information, OS version, something "True", Antivirus running, a hash of some sort, current date, an "N" and lastly 8_=_8 in the end. It is noticeable that the request sent across

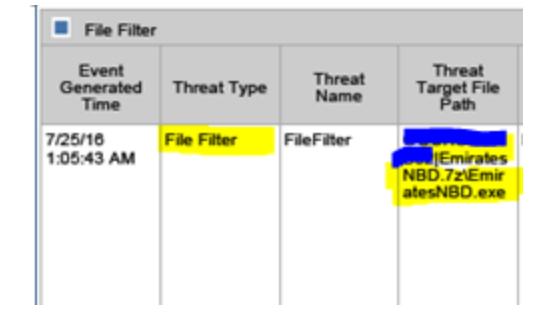
is always preceded by a ping to the ip address. The communication from the client starts with "CONNECT" and each message is preceded by "=P4CK3T". The response from the server is "ACT=P4CK3T=8_=_8".

FireEye detects the communication protcol as Trojan Luminosity Link.

Attempting to decompile the sample with .NET Reflector and JetBrains dotPeek yields no results obviously as the sample is encrypted. DEbugging with DILE also yields no results as the execution never stops at the entry point mentioned in the optional headers of the PE.

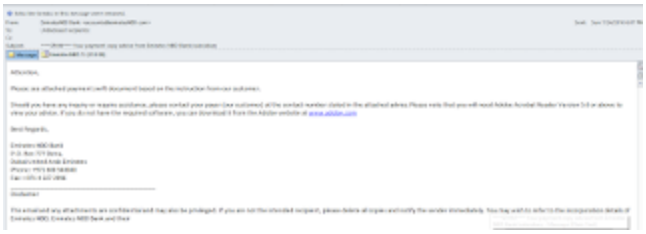
I had to resort on ILSPY to decompile to CIL opcode and see what is going on. Currently, I am also searching for some other ways to successfully debug the sample. The decompiled opcode shows the entry point of the code is the module lohnfraz("lohnfraz.NN v5.5.5").

The sample looks for certain config and ini files as well in the same directory as the sample, indicating that it was accompanied by some configuration and I am guessing that the threat actors have re-purposed the Luminosity RAT, encrypted the client payload and delivered through inside a weaponized .7z archive, aimed to target the GCC region.

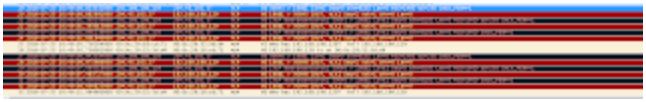


The sample detection by file filter

The email header



The delivery



Contained communication

Process spawning the child process

Proc Mon Activity output

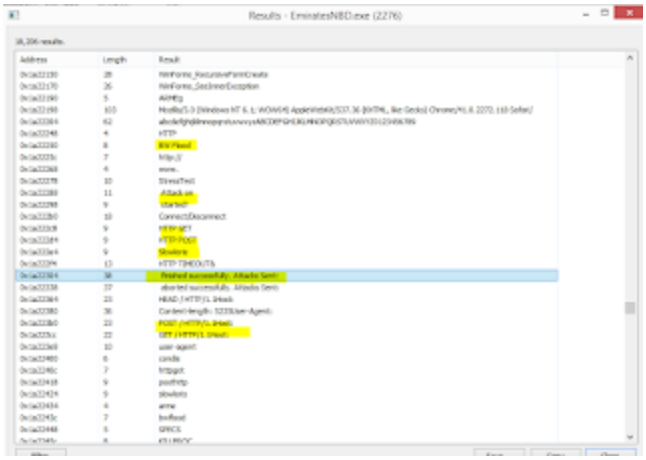
Looking for config file

Looking for INI configuration

Malware can check if there are any unusual entries under the IFEO key(s) as a way of determining if it has landed on an analysts machine and change its behavior accordingly.

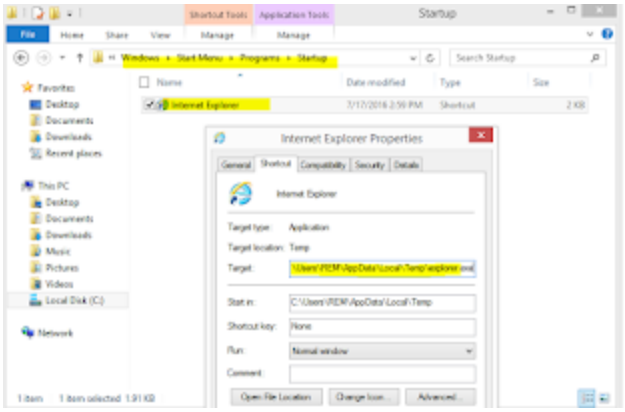
This file is created in the temp folder with the MD5 hash value of the original binary. I am guessing this is an integrity check being done by the sample of itself

Some of the interesting strings extracted from the decrypted child process memory space:

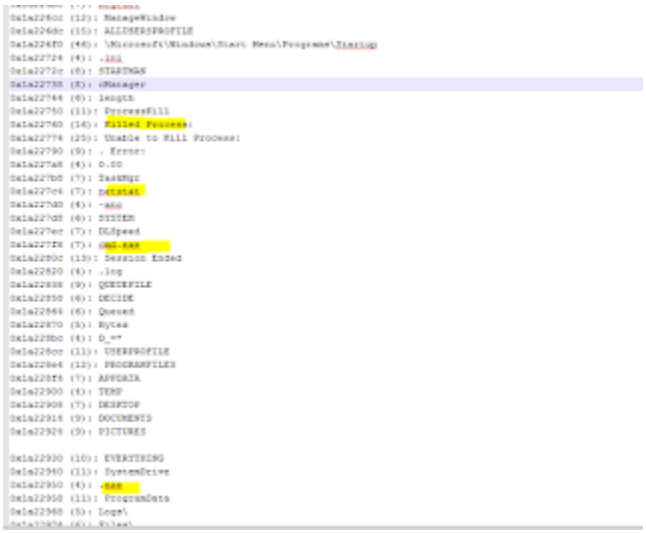


```
0x67f570 (60): C:\Windows\system32\ws2_32.dll
0x67f600 (60): C:\Windows\SYSTEM32\bcrypt.dll
```

```
13760  0x1a223cc (22): GET / HTTP/1.1
13761  Host:
13762  0x1a223e8 (10): user-agent
13763  0x1a22400 (6): condis
13764  0x1a22430 (7): httpget
13765  0x1a22418 (9): pidhttp
13766  0x1a22424 (9): slowloris
13767  0x1a22434 (6): arms
13768  0x1a2243c (7): hwflood
13769  0x1a22448 (5): SPECS
13770  0x1a2245c (8): WSILPROC
13771  0x1a22468 (7): BOTKILL
13772  0x1a2247c (7): ADDHOST
13773  0x1a22490 (6): NKFILE
13774  0x1a2249c (6): FWFILE
13775  0x1a224a8 (6): SCLEAN
13776  0x1a224b4 (6): DELREG
13777  0x1a224c0 (6): Once
13778  0x1a224d8 (5): HOSTS
13779  0x1a224e4 (7): SETHOST
13780  0x1a224f8 (4): CLIP
13781  0x1a22508 (7): SETCLIP
13782  0x1a22514 (5): SKILL
13783  0x1a22520 (10): STARTSHELL
13784  0x1a22530 (7): DELFILE
13785  0x1a2253c (9): STOPSHELL
13786  0x1a2254c (12): RESTARTSHELL
13787  0x1a22564 (5): SWLIB
13788  0x1a22570 (4): HWND
13789  0x1a22578 (4): CWND
13790  0x1a22588 (4): LDIR
13791  0x1a22600 (13): ClientHandler
13792  0x1a22634 (11): UserProfile
13793  0x1a22644 (6): FLOC
13794  0x1a2264c (17): Shell.Application
13795  0x1a22664 (9): NameSpace
13796  0x1a22674 (8): CopyHere
13797  0x1a22680 (5): Items
13798  0x1a22694 (5): FWDEL
```

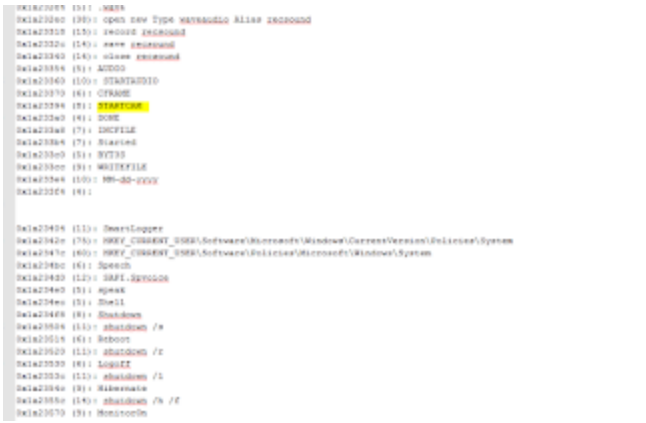


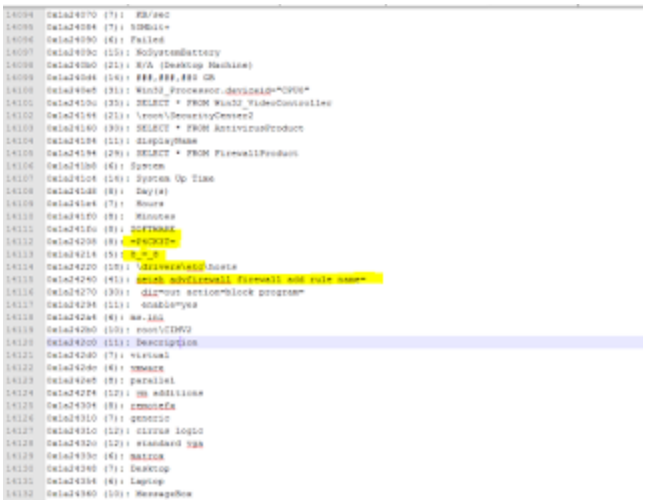
The explorer.exe artifact left by the sample

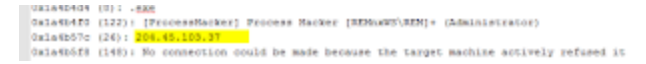


Some more strings showings various commands to be received from the server and executed on the client. We can also see reference to Luminosity in some of the detected strings

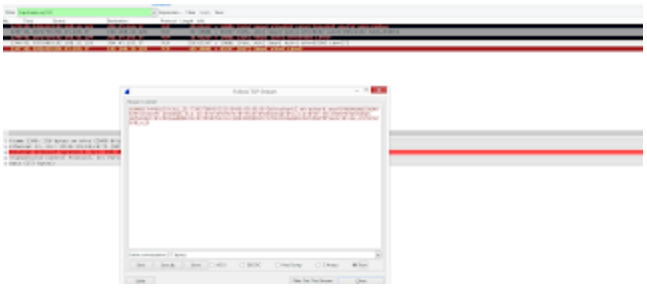
net user and net localgroup commands to add local users and to add them to most probably local admin group.





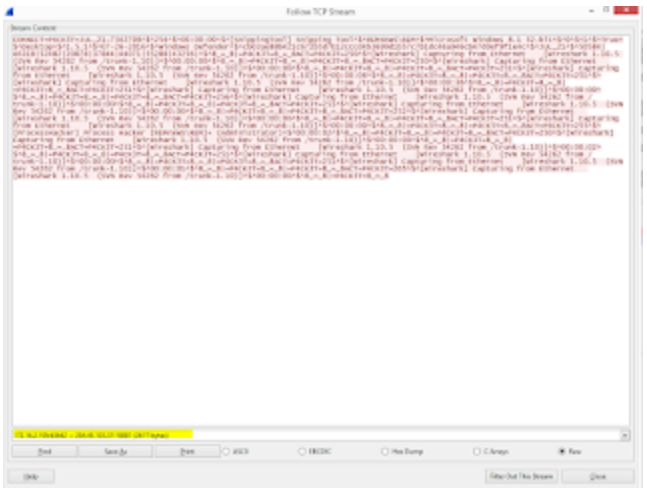


ip address of C2 server hard coded

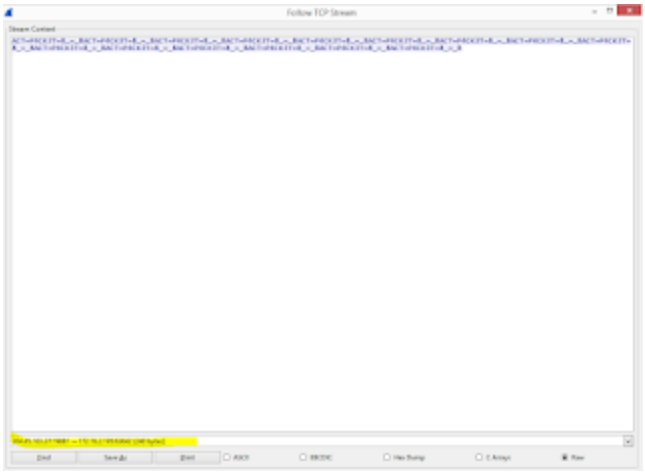


One of client message (hello to C2 server)

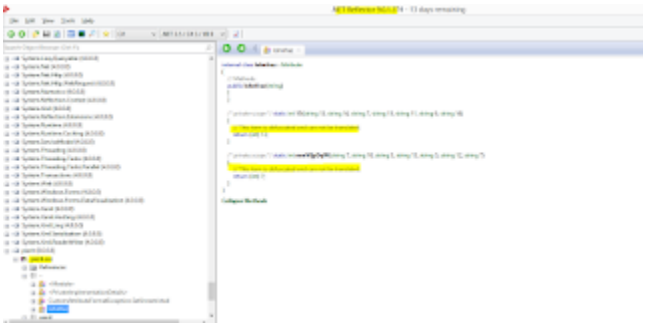
ping before every request



Client communications

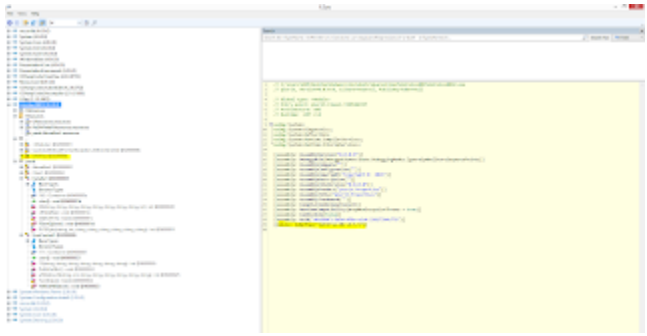


Server replies



De-compile failure

De-compile failure



Reversing the CIL opcode