Threat Thursday: BlackMatter RaaS - Darker Than DarkSide?

blogs.blackberry.com/en/2021/09/threat-thursday-blackmatter-ransomware-as-a-service

The BlackBerry Research & Intelligence Team

First identified in July 2021, BlackMatter is a new player in the Ransomware-as-as-Service (RaaS) arena that many researchers have dubbed the successor to the recently retired Russian ransomware gang DarkSide. However, a spokesperson for BlackMatter insists they are not the same operators.

BlackMatter has recently made headlines as the likely culprit behind cybersecurity incidents affecting a major medical technology company and a U.S. farming cooperative.

Operating System

Windows	MacOS	Linux	Android
Yes	No	Yes	No

Risk & Impact

Impact	**High**
Risk	**High**

About the BlackMatter Group

BlackMatter has been advertised on Russian underground forums, such as XSS and Exploit, looking to recruit affiliates. They claim to have adopted the “best” attributes of DarkSide, REvil and LockBit.

Although these underground forums are among those that have banned ransomware advertisements in the wake of the ransom attack on Colonial Pipeline in May 2021, BlackMatter circumvented this restriction by advertising for “initial access brokers.” These brokers are criminal groups that have gained access to corporate networks or machines.

In their posts, BlackMatter offers a payment of up to $100,000 USD. They state they are looking for access to corporate networks in English-speaking countries, with targets that have between 500 and 1500 hosts and a revenue of over $100M.

The BlackMatter website provides information about the group and even its motivations. The RaaS provider says it aims to fill a void in the market left by DarkSide and REvil pausing their activities. The group’s advertising promotes the strengths of its malware to compete with existing offerings, presumably to attract the most successful affiliates.

On their blog site, BlackMatter includes a list of rules defining sectors they do not attack; they claim to offer free decryption to any victims in these sectors. This is most likely an attempt to avoid the backlash suffered by REvil, Conti, and DarkSide in targeting such industries during 2021.

Unlike DarkSide and REvil, BlackMatter does not include any checks to ensure victims in certain geolocations are not encrypted. BlackMatter will encrypt Russian systems, which may be another way they’re trying to distinguish themselves from other threat groups.

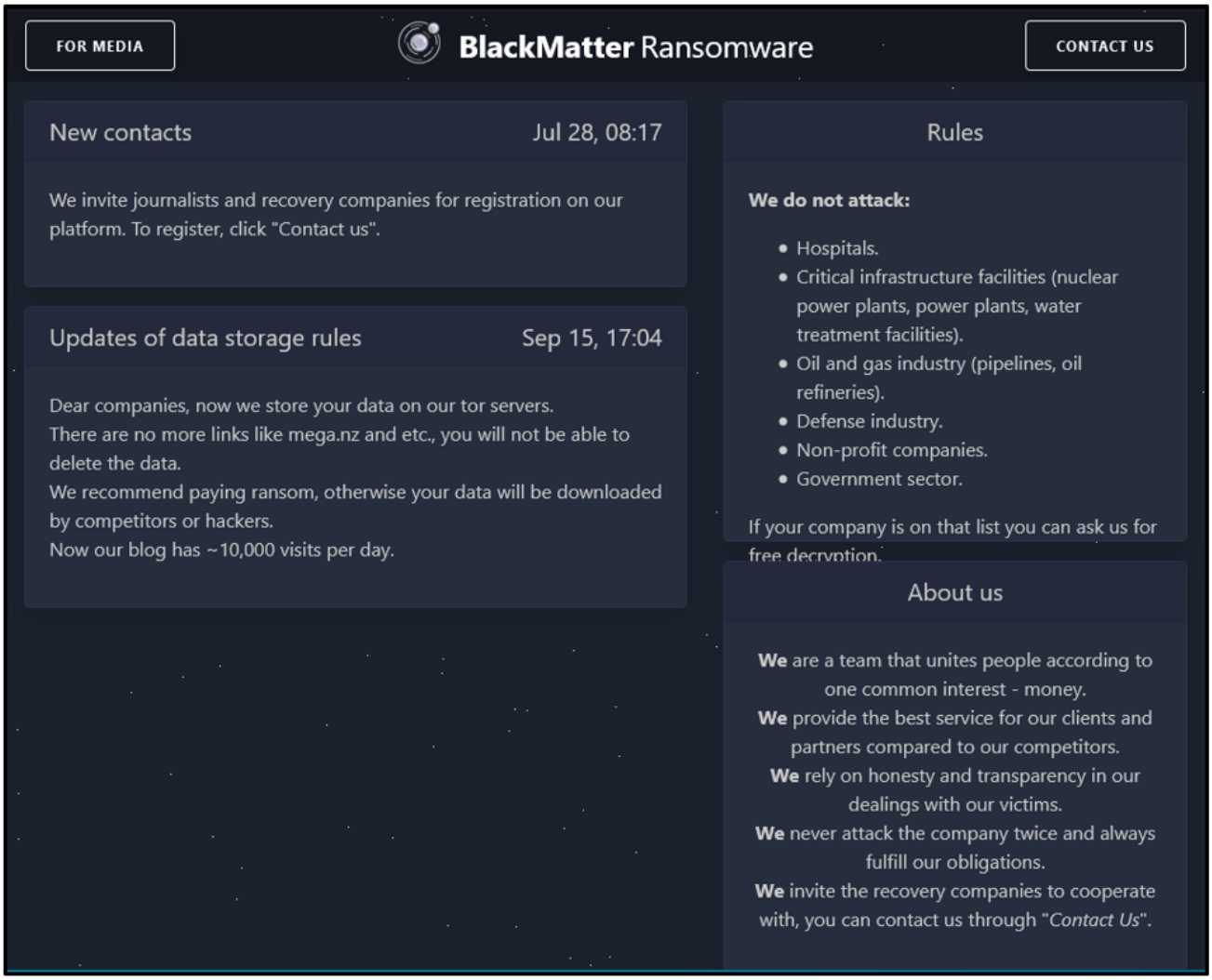

Figure 1: BlackMatter "About" information

Technical Analysis

Since July there have already been several versions and updates of BlackMatter identified. These include versions 1.2, 1.6, 1.9 & 2.0. Additionally, a Linux® variant is available.

Statically analyzing a recent sample identifies the file as a Windows® 32-bit executable with a compile stamp from the Aug. 16,2021. The file includes a .rsrc section, although the binary contains no resources. This section is instead where the encoded configuration information is stored. The sample reviewed is a BlackMatter variant version 2.0.

The binary uses only three libraries, and its import table contains a short list of Application Programming Interfaces (APIs).

library (3)	blacklist (0)	type (1)	imports (19)	description
gdi32.dll	-	implicit	9	GDI Client DLL
user32.dll	-	implicit	5	Multi-User Windows USER API Client DLL
kernel32.dll	-	implicit	5	Windows NT BASE API Client DLL

Figure 2: Static analysis of the file's imported libraries

BlackMatter employs various methods to help evade detection and hinder researchers. For example, most of the APIs and important strings are obfuscated. The executable will de-obfuscate these during runtime as required. Although this approach is common with threat actors, the way it is implemented by BlackMatter is very similar to the functionality found in DarkSide.

Directly after the file's entry point, the binary resolves the addresses of the APIs it requires when executing. The function shown in the image below dynamically loads additional libraries and APIs.

```
int (__stdcall *sub_407DB0())(int, _DWORD, _DWORD)
{
  int (__stdcall *result)(int, _DWORD, _DWORD); // eax
  int (__stdcall *v1)(_DWORD, _DWORD, _DWORD); // esi
  int (__stdcall *v2)(_DWORD, _DWORD, _DWORD); // edi

  result = (int (__stdcall *)(int, _DWORD, _DWORD))get_DLLs_from_hash(0x260B0745);
  if ( result )
  {
    result = (int (__stdcall *)(int, _DWORD, _DWORD))result(0x40000, 0, 0);
    v1 = result;
    if ( result )
    {
      result = (int (__stdcall *)(int, _DWORD, _DWORD))get_DLLs_from_hash(0x6E6047DB);
      v2 = result;
      if ( result )
      {
        decrypt_functions_from_hash(&unk_414DC8, dword_407A34, v1, result);
        decrypt_functions_from_hash(&unk_414E8C, dword_407AFC, v1, v2);
        decrypt_functions_from_hash(&unk_414F50, dword_407BC4, v1, v2);
        decrypt_functions_from_hash(&unk_414FA8, dword_407C20, v1, v2);
        decrypt_functions_from_hash(&unk_414FDC, dword_407C58, v1, v2);
        decrypt_functions_from_hash(&unk_415014, dword_407C94, v1, v2);
        decrypt_functions_from_hash(&unk_415028, dword_407CAC, v1, v2);
        decrypt_functions_from_hash(&unk_415044, dword_407CCC, v1, v2);
        decrypt_functions_from_hash(&unk_41506C, dword_407CF8, v1, v2);
        decrypt_functions_from_hash(&unk_415078, dword_407D08, v1, v2);
        decrypt_functions_from_hash(&unk_415080, dword_407D14, v1, v2);
        decrypt_functions_from_hash(&unk_415094, dword_407D2C, v1, v2);
        decrypt_functions_from_hash(&unk_4150C0, dword_407D5C, v1, v2);
        decrypt_functions_from_hash(&unk_4150D4, dword_407D74, v1, v2);
        return (int (__stdcall *)(int, _DWORD, _DWORD))decrypt_functions_from_hash(&unk_415100, dword_407DA4, v1, v2);
      }
    }
  }
  return result;
}
```

Figure 3: Function to resolve APIs and functions

The DWORD values point to blocks of encrypted hashes for each API, followed by a trailing 0xCCCCCCCC. Each hash is decrypted by an XOR operation.

```
dword_407A34    dd 5617E84Fh, 0B7E8AE27h, 44F42F02h, 0B0261DBCh, 79507CE3h
                                        ; DATA XREF: sub_407DB0+4D↓o
                dd 99287FA3h, 0D058E37h, 1713AF9Bh, 0F96BACDAh, 1933ABDBh
                dd 9723DFA2h, 7953A0A2h, 797BDCE2h, 9943DBE2h, 7943DEE2h
                dd 1565C06Ah, 4AD1EB9Bh, 3B5EFB2h, 10B5E9DBh, 8F364DE9h
                dd 0E86CBB7Ch, 937A6336h, 29E7FCA1h, 57689B36h, 0BEFC6F79h
                dd 351DA176h, 80DA5E40h, 7BE4534h, 48BAC73Ch, 1EB9C28Ah
                dd 6CCEDE7Bh, 10C9F779h, 6FEBA4EFh, 0B6A87F97h, 4FCDEF55h
                dd 7B67CEE7h, 2FB81572h, 0B1C34A09h, 6F45B82h, 5DB2BF30h
                dd 0AA2CBDC1h, 4928D6E6h, 1B8AA828h, 7D93AFE1h, 0D3E62E2h
                dd 0A1E495CDh, 0B6F056F7h, 3671DD61h, 5B86EEC2h, 0CCCCCCCCh
```

Figure 4: Block of encrypted hashes

The image below illustrates the libraries loaded before execution of the function sub_407DB0.

Base	Module	Party	Path
00370000	windows.encryptor.exe	User	C:\Users\Jeff\Desktop\windows.encryptor.exe
755E0000	apphelp.dll	System	C:\Windows\SysWOW64\apphelp.dll
75750000	win32u.dll	System	C:\Windows\SysWOW64\win32u.dll
757D0000	kernelbase.dll	System	C:\Windows\SysWOW64\KernelBase.dll
75B40000	user32.dll	System	C:\Windows\SysWOW64\user32.dll
75CE0000	imm32.dll	System	C:\Windows\SysWOW64\imm32.dll
75D10000	kernel32.dll	System	C:\Windows\SysWOW64\kernel32.dll
76140000	msvcp_win.dll	System	C:\Windows\SysWOW64\msvcp_win.dll
76990000	gdi32full.dll	System	C:\Windows\SysWOW64\gdi32full.dll
76C50000	gdi32.dll	System	C:\Windows\SysWOW64\gdi32.dll
77680000	ucrtbase.dll	System	C:\Windows\SysWOW64\ucrtbase.dll
777D0000	ntdll.dll	System	C:\Windows\SysWOW64\ntdll.dll

Figure 5: DLLs loaded before function has executed

The following image illustrates the DLLs loaded into memory after the function to resolve APIs has executed.

Base	Module	Party	Path	Status
00370000	windows.encryptor.exe	User	C:\Users\Jeff\Desktop\windows.encryptor.exe	Unloaded
70480000	adsldpc.dll	System	C:\Windows\SysWOW64\adsldpc.dll	Unloaded
704C0000	activeds.dll	System	C:\Windows\SysWOW64\activeds.dll	Unloaded
70500000	logoncli.dll	System	C:\Windows\SysWOW64\logoncli.dll	Unloaded
71B10000	ntasn1.dll	System	C:\Windows\SysWOW64\ntasn1.dll	Unloaded
71B40000	srvcli.dll	System	C:\Windows\SysWOW64\srvcli.dll	Unloaded
71BE0000	samcli.dll	System	C:\Windows\SysWOW64\samcli.dll	Unloaded
71C50000	ncrypt.dll	System	C:\Windows\SysWOW64\ncrypt.dll	Unloaded
71C80000	rstrtmgr.dll	System	C:\Windows\SysWOW64\RstrtMgr.dll	Unloaded
72750000	netapi32.dll	System	C:\Windows\SysWOW64\netapi32.dll	Unloaded
728A0000	wtsapi32.dll	System	C:\Windows\SysWOW64\wtsapi32.dll	Unloaded
72940000	netutils.dll	System	C:\Windows\SysWOW64\netutils.dll	Unloaded
72960000	wkscli.dll	System	C:\Windows\SysWOW64\wkscli.dll	Unloaded
74EF0000	winspool.drv	System	C:\Windows\SysWOW64\winspool.drv	Unloaded
75180000	wininet.dll	System	C:\Windows\SysWOW64\wininet.dll	Unloaded
755E0000	apphelp.dll	System	C:\Windows\SysWOW64\apphelp.dll	Unloaded
75680000	msvcrt.dll	System	C:\Windows\SysWOW64\msvcrt.dll	Unloaded
75750000	win32u.dll	System	C:\Windows\SysWOW64\win32u.dll	Unloaded
757D0000	kernelbase.dll	System	C:\Windows\SysWOW64\KernelBase.dll	Unloaded
75AE0000	wldap32.dll	System	C:\Windows\SysWOW64\wldap32.dll	Unloaded
75B40000	user32.dll	System	C:\Windows\SysWOW64\user32.dll	Unloaded
75CE0000	imm32.dll	System	C:\Windows\SysWOW64\imm32.dll	Unloaded
75D10000	kernel32.dll	System	C:\Windows\SysWOW64\kernel32.dll	Unloaded
760A0000	oleaut32.dll	System	C:\Windows\SysWOW64\oleaut32.dll	Unloaded
76140000	msvcp_win.dll	System	C:\Windows\SysWOW64\msvcp_win.dll	Unloaded
761C0000	ole32.dll	System	C:\Windows\SysWOW64\ole32.dll	Unloaded
76310000	combase.dll	System	C:\Windows\SysWOW64\combase.dll	Unloaded
76840000	bcrypt.dll	System	C:\Windows\SysWOW64\bcrypt.dll	Unloaded
768D0000	advapi32.dll	System	C:\Windows\SysWOW64\advapi32.dll	Unloaded
76990000	gdi32full.dll	System	C:\Windows\SysWOW64\gdi32full.dll	Unloaded
76AC0000	shlwapi.dll	System	C:\Windows\SysWOW64\shlwapi.dll	Unloaded
76B10000	rpcrt4.dll	System	C:\Windows\SysWOW64\rpcrt4.dll	Unloaded
76BD0000	sechost.dll	System	C:\Windows\SysWOW64\sechost.dll	Unloaded
76C50000	gdi32.dll	System	C:\Windows\SysWOW64\gdi32.dll	Unloaded
770C0000	shell32.dll	System	C:\Windows\SysWOW64\shell32.dll	Unloaded
77680000	ucrtbase.dll	System	C:\Windows\SysWOW64\ucrtbase.dll	Unloaded
777D0000	ntdll.dll	System	C:\Windows\SysWOW64\ntdll.dll	Unloaded

Figure 6: DLLs loaded after the function has executed

BlackMatter also employs anti-debugging techniques to hide threads from the debugger, which makes it trickier to analyze. If it is analyzed while running under a debugger, the application will crash.

The binary accepts command line arguments when executed. If no argument is supplied, its default action is to first verify the rights of the current user. If required, it will attempt to elevate privileges and bypass User Account Control (UAC).

The malware creates a mutex to ensure that only one instance of the ransomware is running. The mutex name is generated from a registry value relating to the MachineGuid; for example, “Global\21661c2e54b253e217f64acc8644f973”.

The executable deletes three services relating to shadow copies – “vmicvss,” “vmvss” and “vss.” Removing shadow copies is a common practice with ransomware, as it prevents victims from easily restoring their systems.

BlackMatter will terminate common productivity-related processes to increase its impact. Terminating these processes ensures that important files will not be locked, and valuable files can be encrypted.

The ransomware takes a multithreaded approach to enumerating the filesystem and executing the encryption routine, to ensure that files are locked quickly. Local files and any found on connected drives will be encrypted, while vital system files are skipped.

The BlackMatter encryption routine shares similarities with that of DarkSide. It uses custom implementations of the Salsa20 and RSA-1024 algorithms. Only the first megabyte of each file is encrypted, and the encrypted key is added to the end of the file. Partially encrypting files makes the process much faster, which shortens the attack duration and leaves little time for the victim to react.

Encrypted files are appended with an extension consisting of an alpha-numeric string that varies between attacks. Below is an example of an encrypted filename.

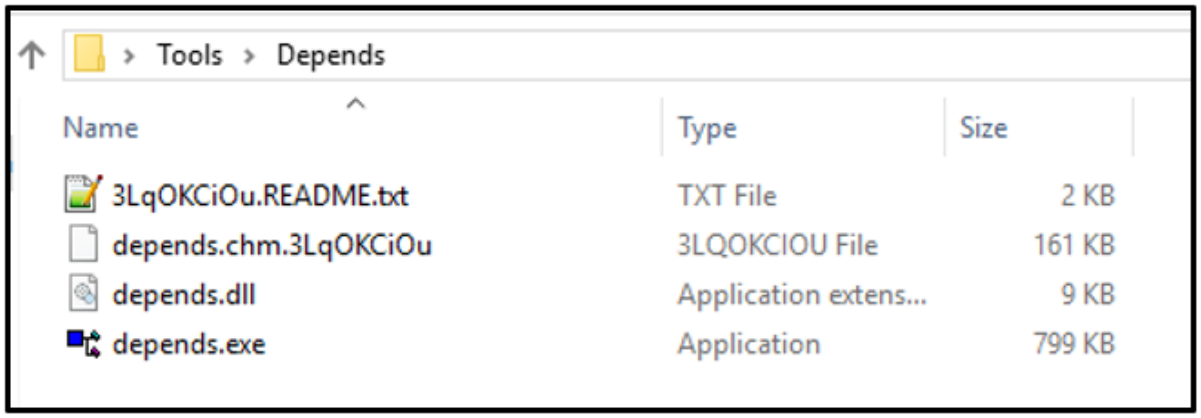

Figure 7: Encrypted file with appended extension ".3LqOKCiOu"

The ransomware drops a bitmap image file to C:\ProgramData\ and sets this as the victim's background through the registry.

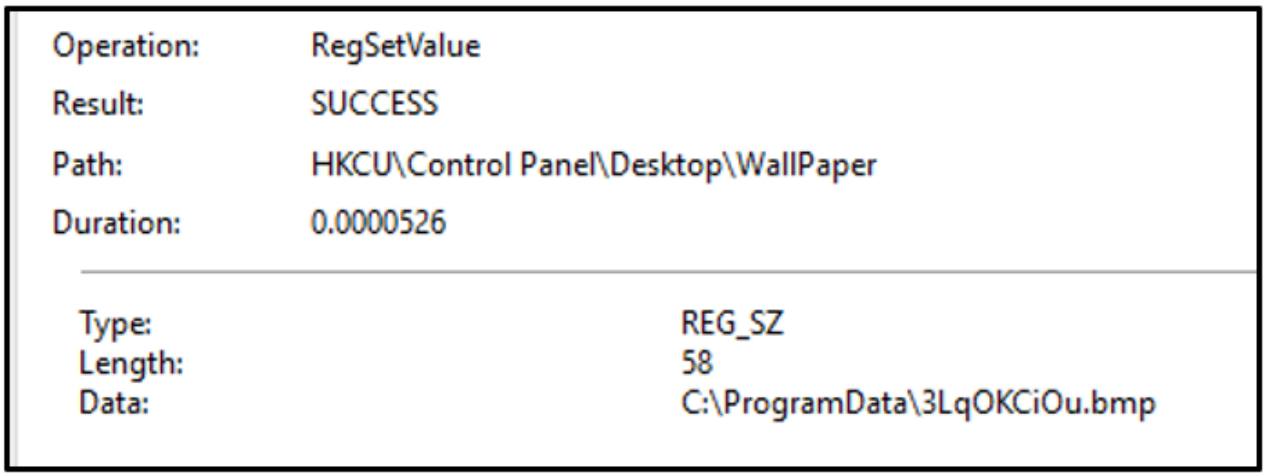

Figure 8: Defining wallpaper image in the registry - 3LqOKCiOu.bmp

The wallpaper notifies the user their files have been encrypted by BlackMatter. The image itself appears very similar to the one used by DarkSide.

Figure 9: Desktop wallpaper of an infected endpoint

A ransom note is also dropped into each directory as a text file. This text file is named using the same extension as the encrypted files, appended with README.txt. For example, "3LqOKCiOu.README.txt."

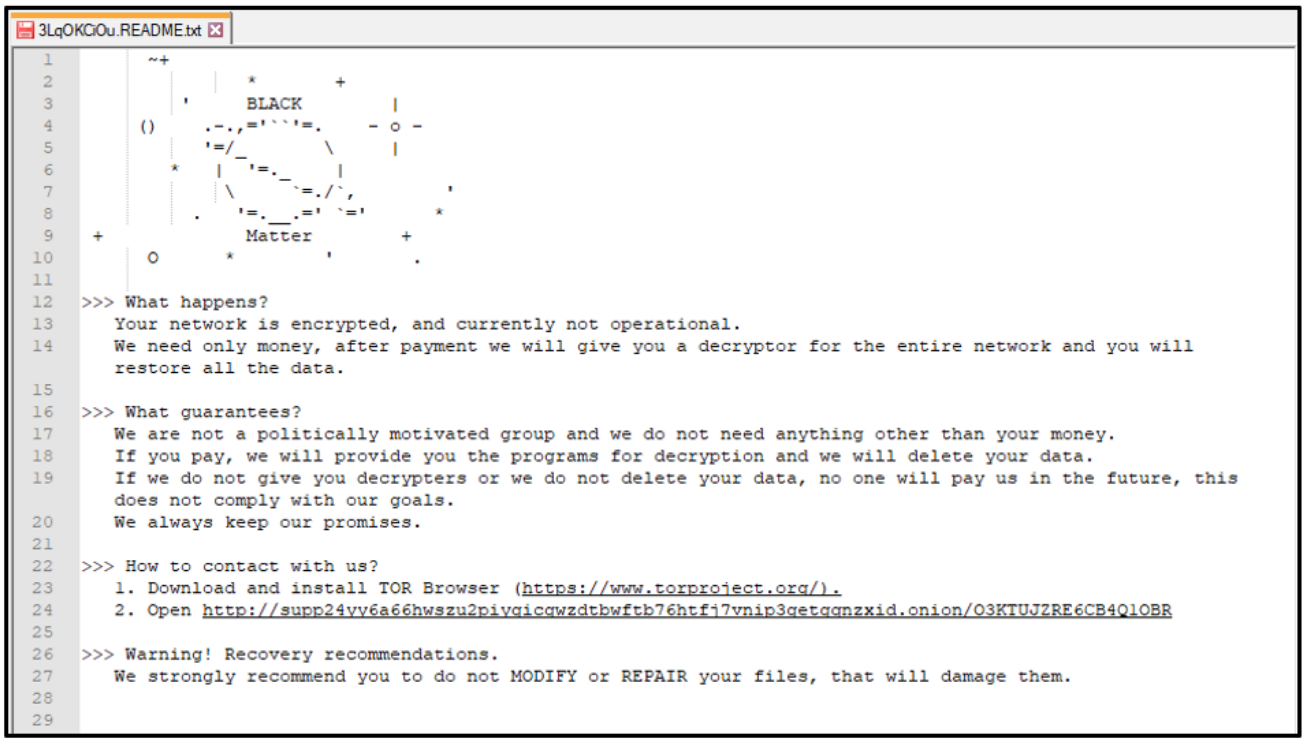

Figure 10: BlackMatter ransom note

The ransom note provides the URL of an onion website (one accessible via TOR) where the victim can pay if they wish to obtain the decryptor for their network.

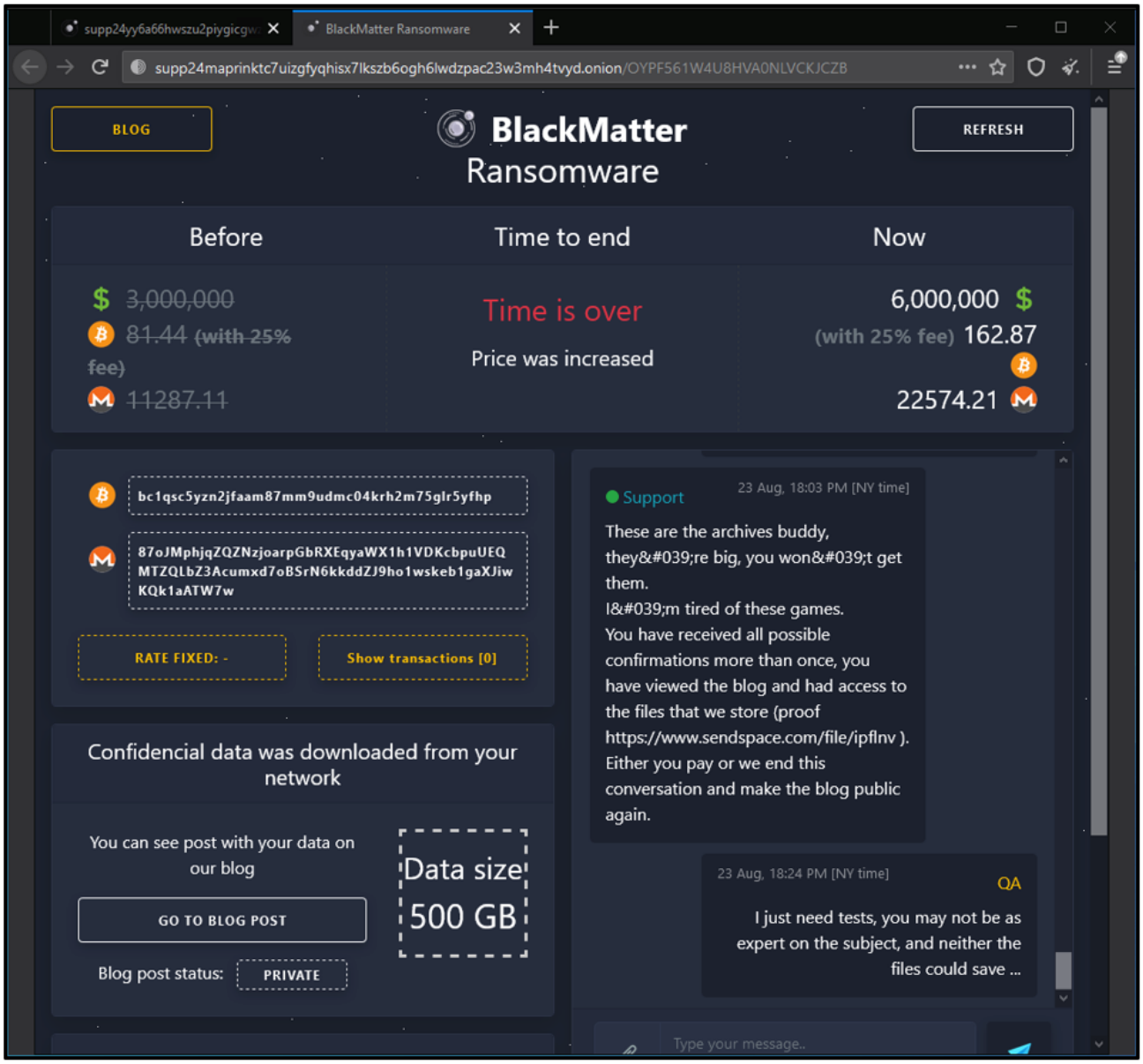

Figure 11: BlackMatter payment website

The malware collects information from the victim's machine and sends this data to its command-and-control (C2) servers in an encoded POST request. During our analysis, the sample in question communicated with http[s]://mojobiden[.]com and http[s]://nowautomation[.]com.

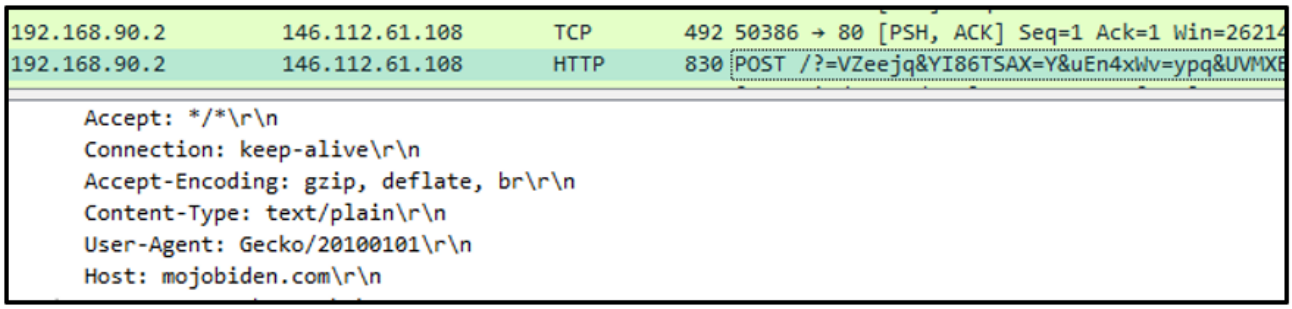

Figure 12: POST request to C2

Like most modern RaaS providers, BlackMatter uses the technique of double extortion. A leak site is available on the dark net. Victims are threatened that their confidential data will be released publicly if they choose not to pay.

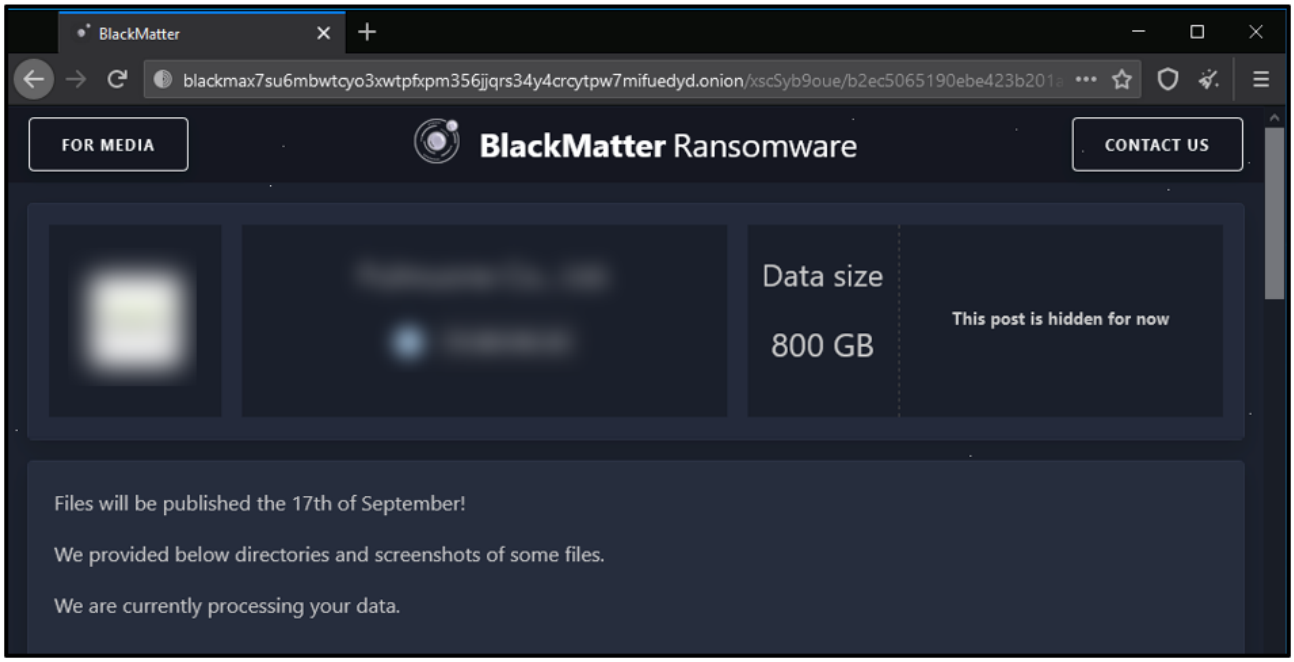

Figure 13: BlackMatter leak site

YARA Rule

The following YARA rule was authored by the BlackBerry Research & Intelligence Team to catch the threat described in this document:

```
import "pe"
import "hash"

rule Mal_Win_Ransom_BlackMatter {

meta:
description = "BlackMatter Ransomware September 2021"
author = "BlackBerry Threat Research Team"
date = "2021-09"
condition:
pe.is_32bit() and
filesize < 90KB and
filesize > 60KB and
pe.number_of_imports == 3 and
pe.imphash() == "2e4ae81fc349a1616df79a6f5499743f" and
hash.md5(pe.sections[0].raw_data_offset, pe.sections[0].raw_data_size) ==
"100da8cf342d6d8f3bd24b367e0ea999" and
pe.sections[3].name == ".rsrc" and
pe.number_of_signatures == 0 and
pe.number_of_sections == 5
}
```

Indicators of Compromise (IoCs)

Files Dropped:
C:\ProgramData\<alpha-numeric_extension>.bmp
e.g.: C:\ProgramData\3LqOKCiOu. Bmp

Mutex
Global\<alpha-numeric_string>
e.g.: Global\21661c2e54b253e217f64acc8644f973

Services Terminated
Vmicvss
Vmvss
Vss

Encrypted Files:
<filename>.<alpha-numeric_extension>
e.g.: test.jpg.3LqOKCiOu

Ransom note:
< alpha-numeric_extension >.README.txt
e.g.: 3LqOKCiOu.README.txt

CnC Servers:
http[s]://mojobiden[.]com
http[s]://nowautomation[.]com

Payment URL:
hxxp://supp24yy6a66hwszu2piygicgwzdtbwftb76htfj7vnip3getgqnzxid.onion/O3KTUJZRE6CB4Q1OBR

BlackBerry Assistance

If you're battling this malware or a similar threat, you've come to the right place, regardless of your existing BlackBerry relationship.

The BlackBerry Incident Response team is made up of world-class consultants dedicated to handling response and containment services for a wide range of incidents, including ransomware and Advanced Persistent Threat (APT) cases.

We have a global consulting team standing by to assist you by providing around-the-clock support, if required, as well as local assistance. Please contact us here: https://www.blackberry.com/us/en/forms/cylance/handraiser/emergency-incident-response-containment

Want to learn more about this threat? Watch our new demo video: ***BlackBerry vs. BlackMatter RaaS.***

About The BlackBerry Research & Intelligence Team

The BlackBerry Research & Intelligence team examines emerging and persistent threats, providing intelligence analysis for the benefit of defenders and the organizations they serve.

Back