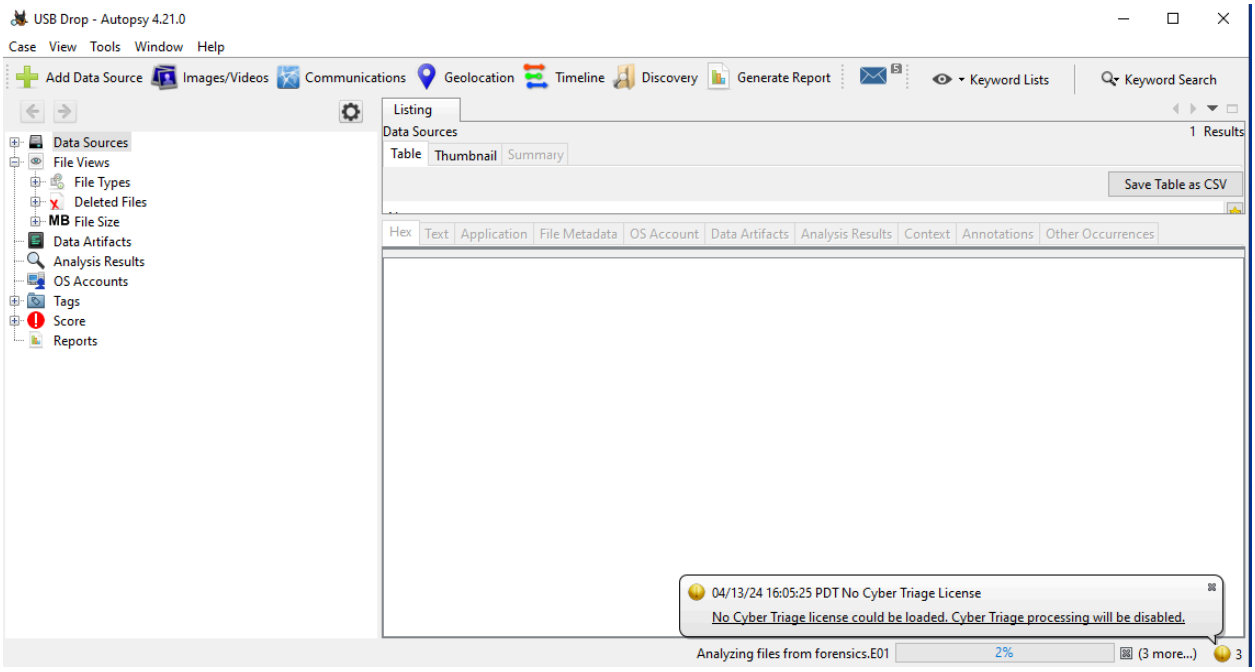
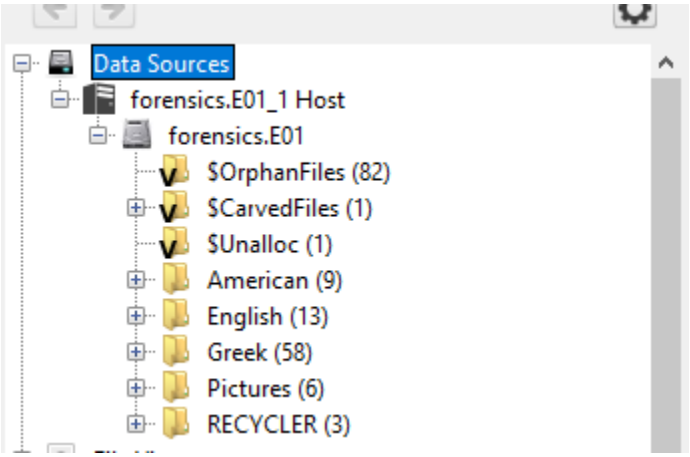
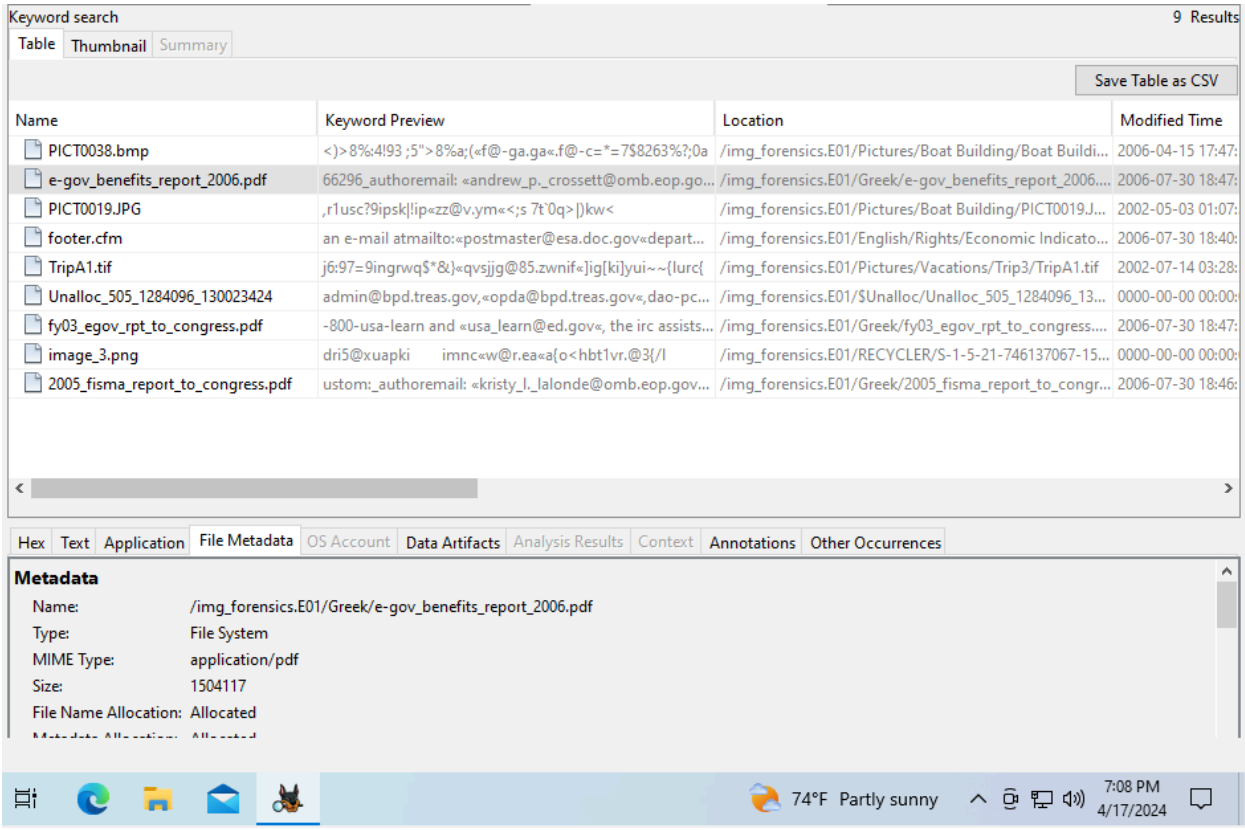
Task 1:



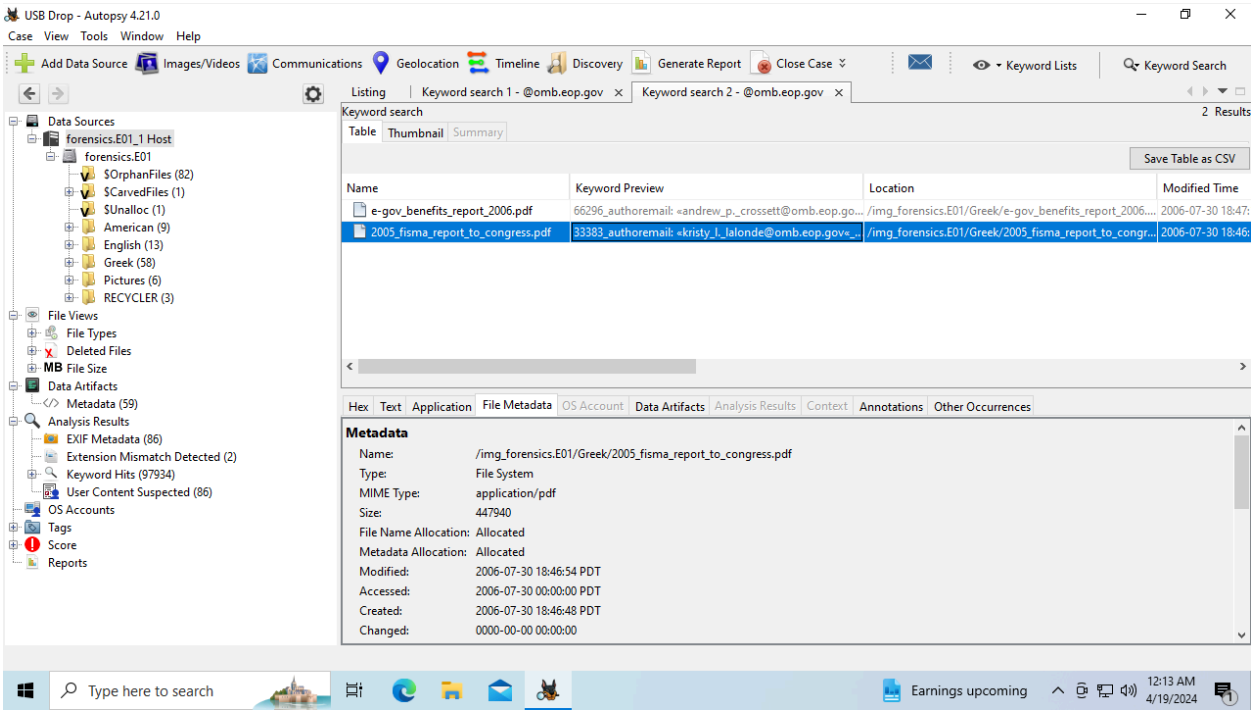
In this step, I have created a new case where I am using the forensics.e01 file as the data source.



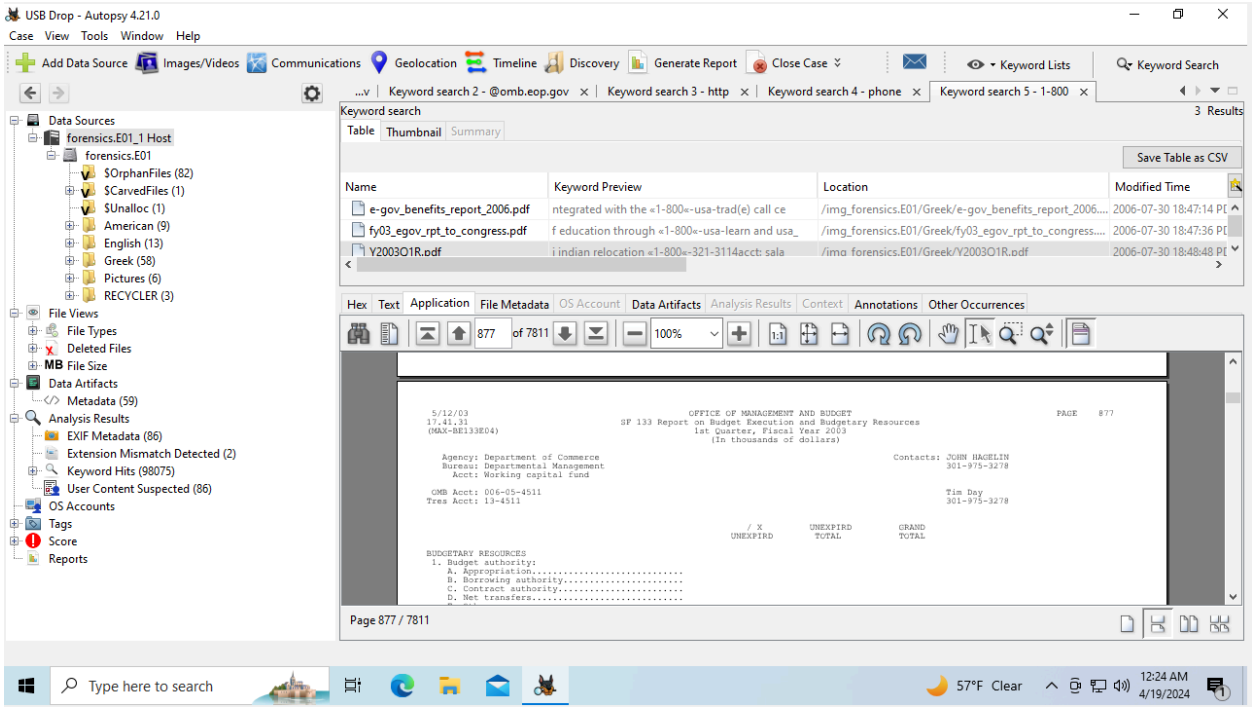
Here we can see that the drive folders are displayed.



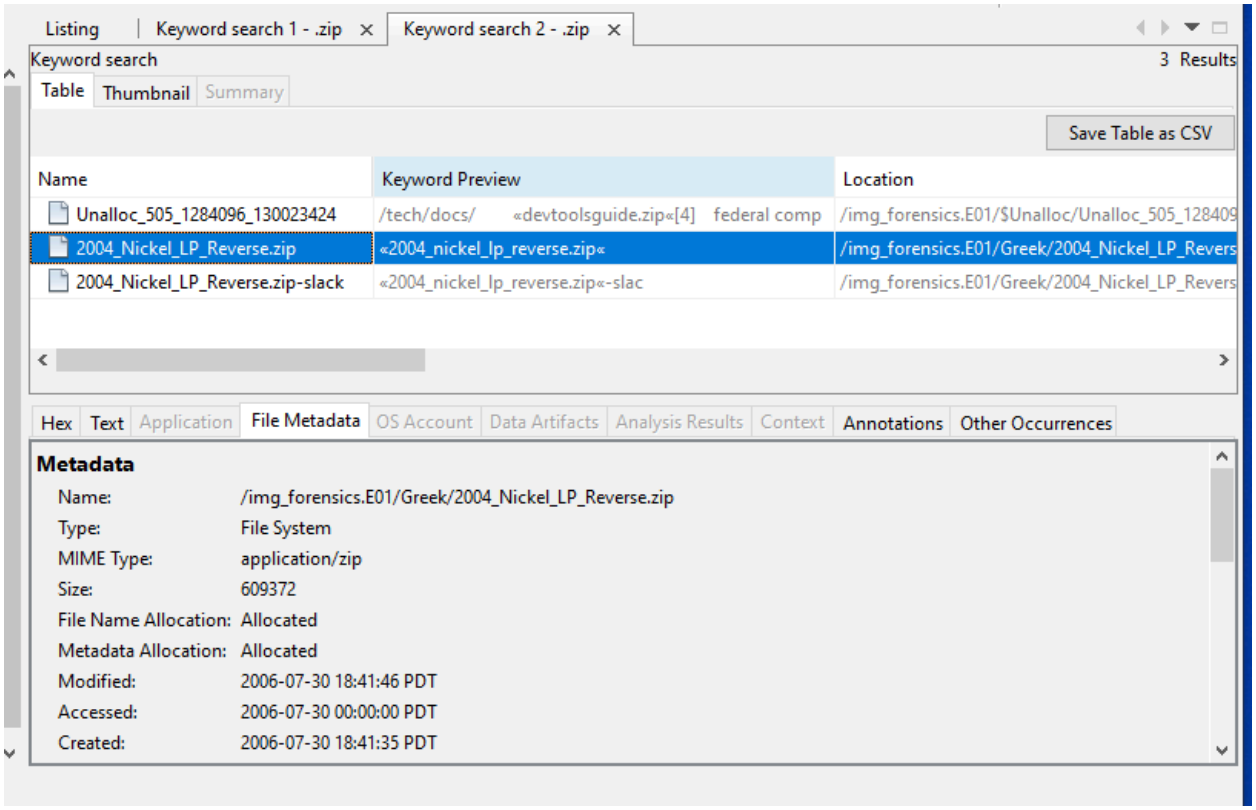
Here I was able to find an email: Andrew_P._Crossett@omb.eop.gov that was in a pdf. The pdf location is in the image above. I found this by using a regular expression search where I inputted “\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b”.



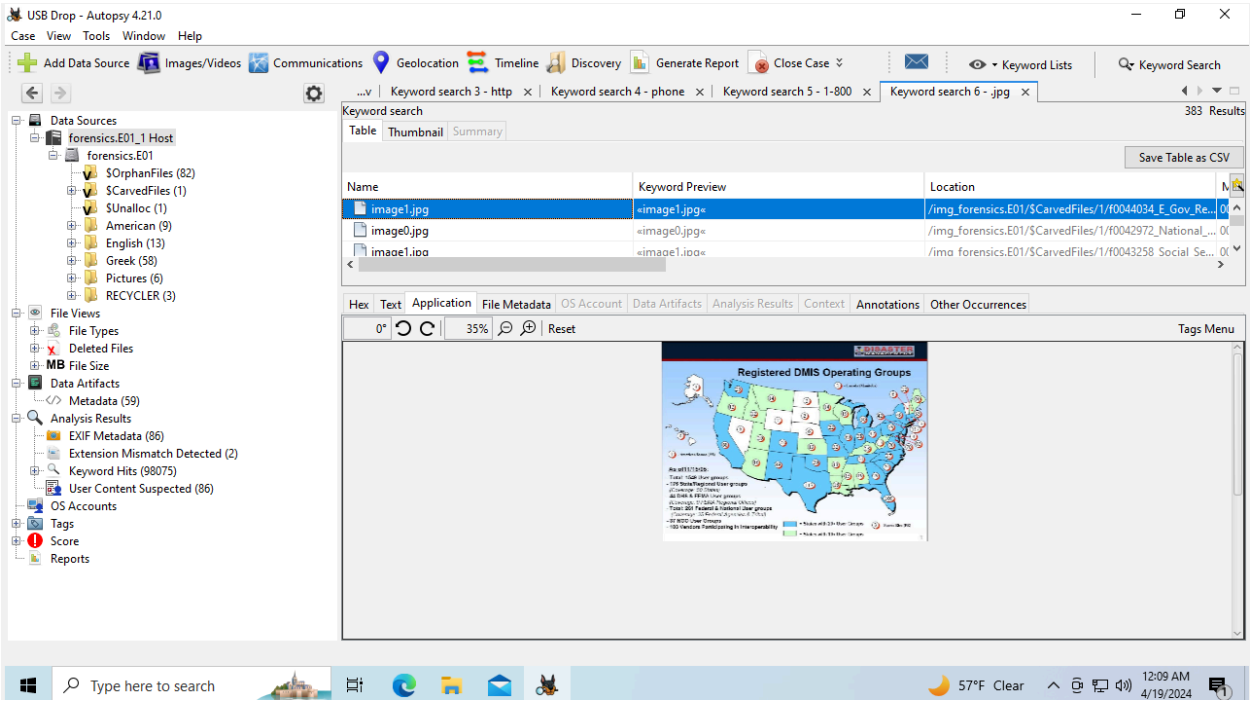
Here I was able to find kristy_l._lalonde@omb.eop.gov. I was able to find this by using the email I found in the previous step, taking the address and searching the directory for that substring (@emb.eop.gov).



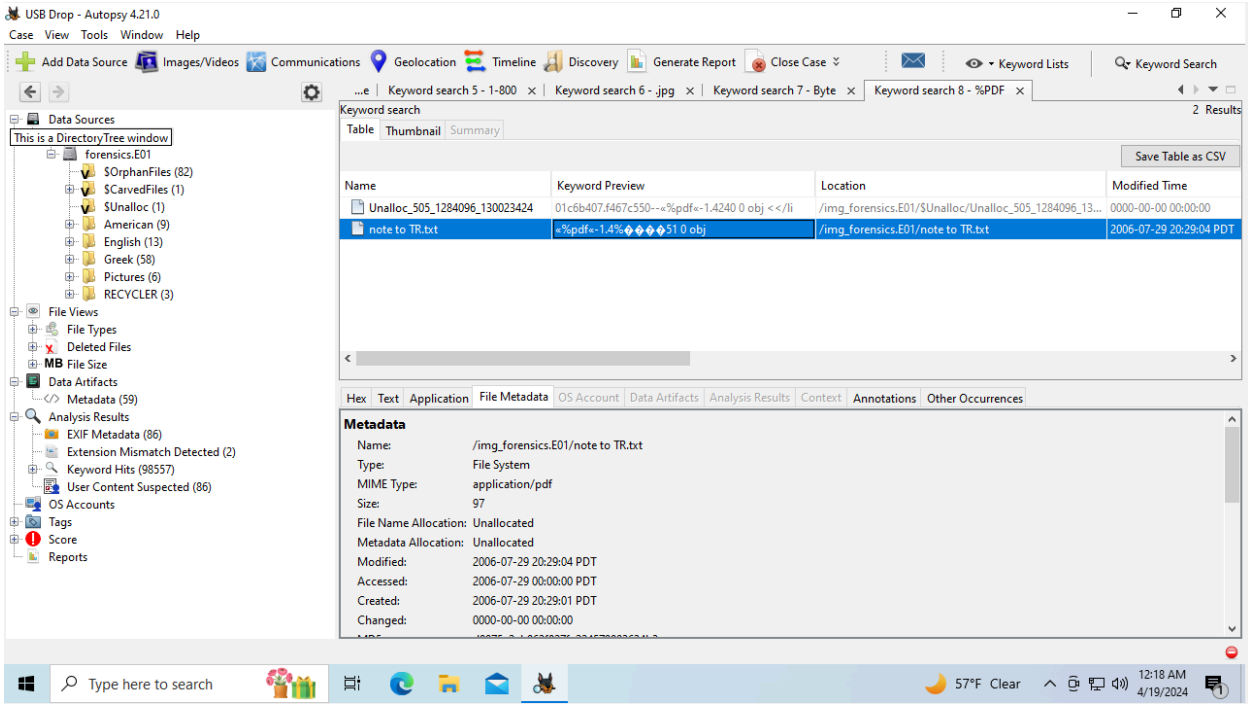
In this image, I was able to find 2 phone numbers in one pdf. I did this by searching for “1-800” as a substring and was able to find this pdf. The phone numbers in this pdf are 301-975-3278 and it belongs to John Hagelin and 301-975-3278 which belongs to Tim Day. The location of this pdf is “/img_forensics.E01/Greek/Y2003Q1R.pdf”.



Here I was able to find a zip file. I did this by searching .zip and clicking the substring match option. The location of this file is included in the image.



I was able to find this jpg file by searching for ".jpg" as a substring. This file is located at "/img_forensics.E01/$CarvedFiles/1/f0044034_E_Gov_Report.pdf/image1.jpg".



Here I was able to find a PDF Magic Byte Hex Code by searching “%PDF” as a substring. The file location is in the image above.

e no suspicion, it is said, that any
of their fellow-citizens will deceive them. At Basel the
principal revenue of the state arises from a small custom upon
goods exported. All the citizens make oath that they will pay
every three months all the taxes imposed by the law. All
merchants and even all innkeepers are trusted with keeping
themselves the account of the goods which they sell either within
or without the territory. At the end of every three months they
send this account to the treasurer with the amo

----------METADATA----------

Metadata

Name:	/img_forensics.E01/English/Rights/_o-3.txt
Type:	File System
MIME Type:	text/plain
Size:	8789
File Name Allocation:	Unallocated
Metadata Allocation:	Unallocated
Modified:	2006-07-30 18:16:42 PDT
Accessed:	2006-07-30 00:00:00 PDT
Created:	2006-07-30 18:16:41 PDT
Changed:	0000-00-00 00:00:00
MD5:	58a5b1dd02c47c9960d4ec370a4a2c42
SHA-256:	ad1e948a28f7416a58c978176287e1a610ee814b7cb2534281f413cc66d16ed8
Hash Lookup Results:	UNKNOWN
Internal ID:	132

Listing | Keyword search 1 - @omb.eop.gov | Keyword search 2 - @omb.eop.gov | Keyword search 3 - http | Keyword search 4 - phone ...

File System 208 Result

Table | Thumbnail | Summary

Save Table as CSV

Name	S	C	O	Modified Time	Change Time	Access Time	Created Time
Mayflower-compact.txt				2006-07-30 18:11:42 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:11:4
_o-1.txt				2006-07-30 18:15:36 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:15:3
_o-1.txt				2006-07-30 18:15:36 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:15:3
_o-2.txt				2006-07-30 18:16:10 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:16:0
_o-3.txt				2006-07-30 18:16:42 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:16:4
_o-3.txt				2006-07-30 18:16:42 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:16:4
_o-4.txt				2006-07-30 18:17:08 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:17:0
_o-4.txt				2006-07-30 18:17:10 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:17:0
_o-5.txt				2006-07-30 18:17:36 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:17:3
_o-5.txt				2006-07-30 18:17:36 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:17:3
_o-6.txt				2006-07-30 18:18:06 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:18:0
_o-7.txt				2006-07-30 18:18:44 PDT	0000-00-00 00:00:00	2006-07-30 00:00:00 PDT	2006-07-30 18:18:4

In this step, I have found a deleted file and am looking at its data. In the pictures above, I have its contents, metadata and its file type, which is a txt file.

Task 2:

```
┌──(timothyd㉿kali)-[~]
└─$ yara --help
YARA 4.2.3, the pattern matching swiss army knife.
Usage: yara [OPTION]... [NAMESPACE:]RULES_FILE... FILE | DIR | PID

Mandatory arguments to long options are mandatory for short options too.

        --atom-quality-table=FILE          path to a file with the atom quality table
  -C,   --compiled-rules                   load compiled rules
  -c,   --count                            print only number of matches
  -d,   --define=VAR=VALUE                 define external variable
        --fail-on-warnings                 fail on warnings
  -f,   --fast-scan                        fast matching mode
  -h,   --help                             show this help and exit
  -i,   --identifier=IDENTIFIER            print only rules named IDENTIFIER
        --max-process-memory-chunk=NUMBER  set maximum chunk size while reading process memory (default=1073741824)
  -l,   --max-rules=NUMBER                 abort scanning after matching a NUMBER of rules
        --max-strings-per-rule=NUMBER      set maximum number of strings per rule (default=10000)
  -x,   --module-data=MODULE=FILE          pass FILE's content as extra data to MODULE
  -n,   --negate                           print only not satisfied rules (negate)
  -N,   --no-follow-symlinks               do not follow symlinks when scanning
  -w,   --no-warnings                      disable warnings
  -m,   --print-meta                       print metadata
  -D,   --print-module-data                print module data
```

Here I have installed yara.

```
BCryptOpenAlgorithmProvider
BCryptDestroyHash
BCryptKeyDerivation
BCryptHashData
BCryptFinishHash
BCryptGenerateSymmetricKey
BCryptCloseAlgorithmProvider
BCryptDestroyKey
BCryptDeriveKeyPBKDF2
BCryptCreateHash
BCryptGetProperty
BCryptEncrypt
BCryptDecrypt
BCryptSetProperty
BCryptImportKeyPair
BCryptExportKey
BCryptFreeBuffer
BCryptEnumRegisteredProviders
NCryptOpenStorageProvider
NCryptGetProperty
NCryptSetProperty
```

In this step, I have run the strings tool and piped it to the “less” utility. Then I have run teh BCrypt function in the less editor.

```
File: /usr/share/windows-resources/mimikatz/x64/mimikatz.exe
001382E0  4C 8B DF 49  C1 E3 04 48   8B CB 4C 03  D8 00 00 00
001382F0  33 FF 45 85  C0 41 89 75   00 4C 8B E3  0F 84 00 00
00138300  33 F6 45 89  2F 4C 8B F3   85 FF 0F 84  25 02 00 C0
00138310  8B DE 48 8D  0C 5B 48 C1   E1 05 48 8D  05 00 00 00
00138320  33 FF 41 89  37 4C 8B F3   45 85 C0 74  BD FF FF FF
00138330  33 FF 45 89  37 48 8B F3   45 85 C9 74  EF FF FF FF
00138340  33 FF 41 89  37 4C 8B F3   45 85 C9 74  DD FF FF FF
00138350  45 89 34 24  4C 8B FF 8B   F3 45 85 C0  74 00 00 00
00138360  28 0A 00 00  00 00 00 00   0D 00 00 00  00 00 00 00
00138370  E0 9E 13 40  01 00 00 00   00 00 00 00  00 00 00 00
00138380  00 00 00 00  00 00 00 00   FC FF FF FF  00 00 00 00
00138390  00 00 00 00  00 00 00 00   00 00 00 00  00 00 00 00
001383A0  00 00 00 00  00 00 00 00   00 00 00 00  00 00 00 00
```

Here I am in the mimikatz.exe file and am at offset 1382E0.

```
rule mimikatz_x64_exe
{
        strings:
                $hex = {4C 8B DF 49  C1 E3 04 48   8B CB 4C 03  D8 00 00 00}
                $string = "BCrypt" nocase
        condition:
        all of them
}
```

In this step, I have added the hex code to the ruleset.

```
┌──(timothyd㉿kali)-[~]
└─$
sudo yara -r mimikatz.yar /usr/share/windows-resources
mimikatz_x64_exe /usr/share/windows-resources/mimikatz/x64/mimikatz.exe
mimikatz_x64_exe /usr/share/windows-resources/powershell-empire/empire/server/csharp/Covenant/Data/EmbeddedResources/SharpSploit.Resources.powerkatz_x64.dll
mimikatz_x64_exe /usr/share/windows-resources/powershell-empire/empire/server/csharp/Covenant/Data/ReferenceSourceLibraries/SharpSploit/SharpSploit/Resources/powerkatz_x64.dll
```

In this step, I am running yara and can see files being returned.

Task 3:

ed01ebfbc9eb5bbea545af4d01bf5f10716618404804 39c6e5babe8e080e41aa

Sign in | Sign up

Join the VT Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label: ransomware.wannacry/wannacryptor

Threat categories: ransomware, trojan

Family labels: wannacry, wannacryptor, wanna

Security vendors' analysis

Do you want to automate checks?

AhnLab-V3	Trojan/Win32.WannaCryptor.R200571	Alibaba	Ransom:Win32/WannaCry.ali1020010
AliCloud	RansomWare	ALYac	Trojan.Ransom.WannaCryptor
Antiy-AVL	Trojan[Ransom]/Win32.Scatter	Arcabit	Trojan.Ransom.WannaCryptor.A
Avast	Win32:WanaCry-A [Trj]	AVG	Win32:WanaCry-A [Trj]
Avira (no cloud)	TR/Ransom.JB	Baidu	Win32.Trojan.WannaCry.c
BitDefender	Trojan.Ransom.WannaCryptor.A	BitDefenderTheta	Gen:NN.ZexaF.36802.wt0@aGEmS3di
Bkav Pro	W32.WanaCryptBTTc.Worm	ClamAV	Win.Ransomware:Wannacryptor-994018...
CrowdStrike Falcon	Win/malicious_confidence_100% (W)	Cylance	Unsafe
Cynet	Malicious (score: 100)	DeepInstinct	MALICIOUS
DrWeb	Trojan.Encoder.11432	Elastic	Malicious (high Confidence)
Emsisoft	Trojan.Ransom.WannaCryptor.A (B)	eScan	Trojan.Ransom.WannaCryptor.A
ESET-NOD32	Win32/Filecoder.WannaCryptor.D	Fortinet	W32/WannaCryptor.6F87!tr.ransom
GData	Win32.Trojan-Ransom.WannaCry.A	Gridinsoft (no cloud)	Ransom.Win32.Filecoder.dd

Here we can see some vendors identify the hash as WannaCry.

Compiler Products

[IMP] Windows Server 2003 SP1 DDK build 4035 count=13

[---] Unmarked objects count=163

[C++] VS98 (6.0) SP6 build 8804 count=7

[RES] VS98 (6.0) SP6 cvtres build 1736 count=1

id: 0xc, version: 7291 count=2

id: 0xb, version: 8047 count=1

id: 0xe, version: 7299 count=4

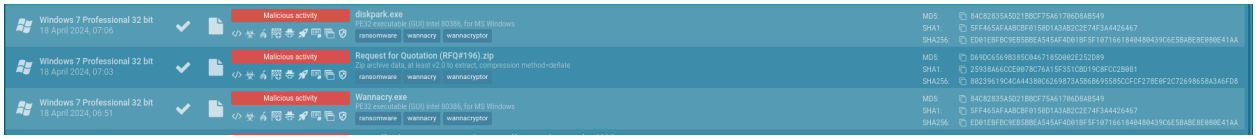
id: 0xa, version: 8047 count=11

id: 0x4, version: 8047 count=4

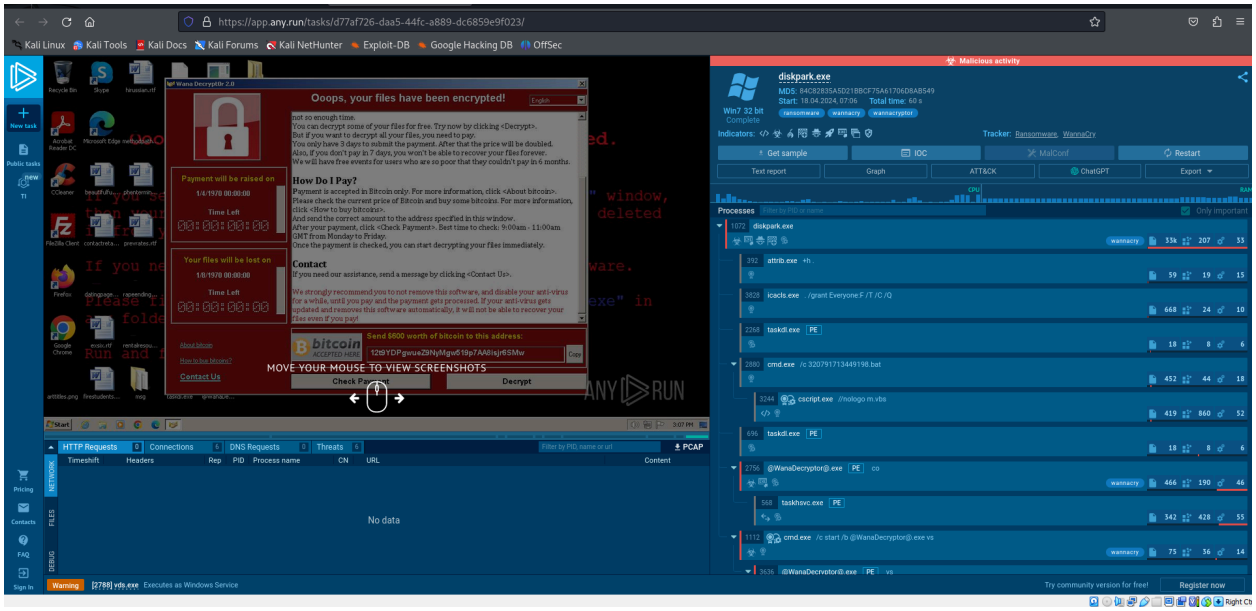
Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2010-11-20 09:05:05 UTC
Entry Point	30650
Contained Sections	4

The image above shows the tools that compiled the program and that it was compiled at 9:05 on 11/20/2010.



In this step, I have navigated to https://app.any.run/submissions/ and searched “WannaCry”. I can see that there are multiple submissions with the md5 “84c82835a5d21bbcf75a61706d8ab549”.



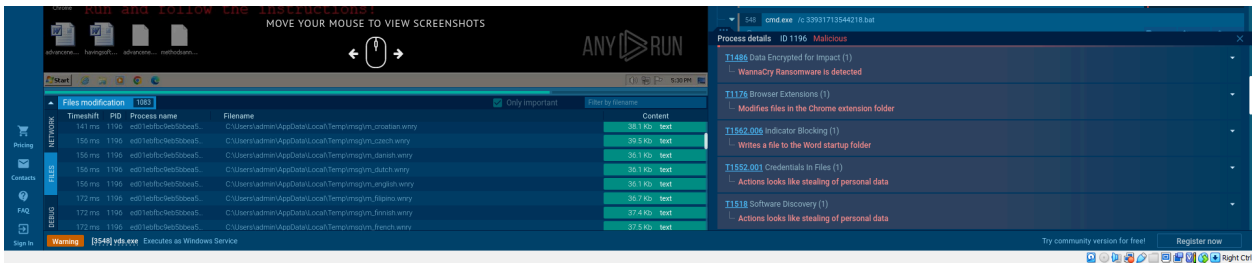
Here we can see that that malware is telling the user that their files have been encrypted and that they will be lost if they don't pay a fee.

HTTP Requests 0 | Connections 6 | DNS Requests 0 | Threats 6 | Filter by PID, domain, name or ip | PCAP

NETWORK | FILES | DEBUG

Timeshift	Protocol	Rep	PID	Process name	CN	IP	Port	Domain	ASN	Traffic
BEFORE	UDP	✓	4	System	?	192.168.100.255	138	–	–	↑ 2.38 Kb ↓ –
BEFORE	UDP	✓	4	System	?	192.168.100.255	137	–	–	↑ 944 b ↓ –
620 ms	UDP	?	–	–	?	224.0.0.252	5355	–	–	↑ 48 b ↓ –
2721 ms	UDP	?	1080	svchost.exe	?	224.0.0.252	5355	–	–	↑ 48 b ↓ –
46748 ms	TCP	?	568	taskhsvc.exe		109.105.109.162	60784	–	NORDUnet	↑ 1.40 Kb ↓ 3.58 Kb
46760 ms	TCP	?	568	taskhsvc.exe		171.25.193.9	80	–	Foreningen for digitala fri- o...	↑ 2.96 Kb ↓ 34.1 Kb

Here we can see that there were 6 connections made while the malware was running.



Here we can see that these files were written too when running the malware.