The Cuba
Ransomware Epidemic:
Taking the Cyber World by Storm

The Cuba Ransomware group is rapidly expanding its attack tactics by seeking out vulnerabilities in open attack surfaces and zero-day flaws.

Author: Aravind Raj

Quick Heal

www.quickheal.co.in

Cuba Ransomware:
A Menace that is Rapidly Gaining Power

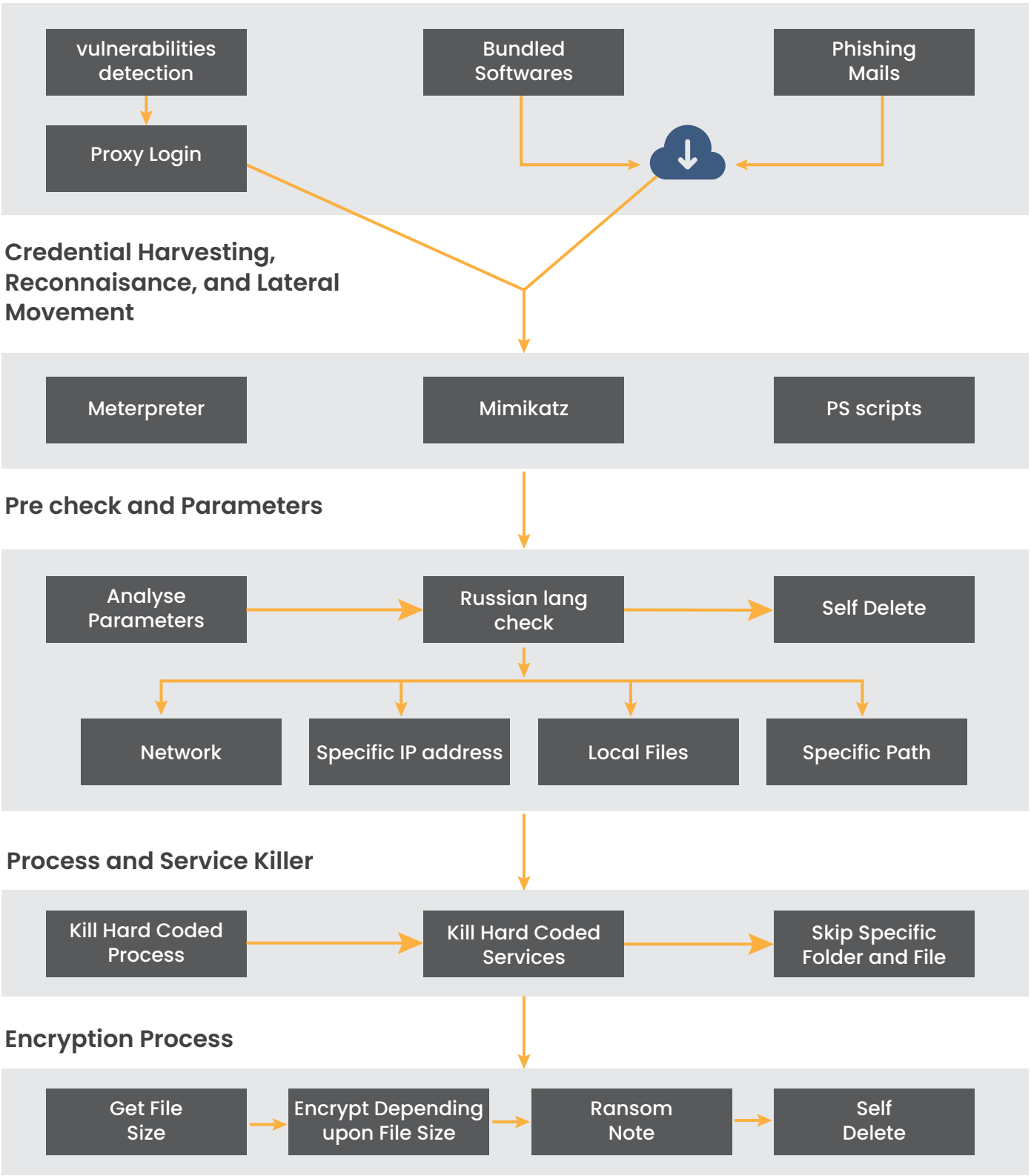
Cuba Ransomware is a family of ransomware that was tracked to a threat group named DEV-0671. While the group had been operational since December 2019, it wasn't until November 2021, when the FBI published an official notification (here) regarding its operations that it came under limelight. According to a joint study by the FBI and CISA, in 2022 alone, perpetrators of this ransomware have penetrated over 100 businesses globally, demanded over USD 145 million, and received over USD 60 million in ransom payments. The vast infrastructure, powerful tools, and related viruses of the ransomware gang makes it a key participant in the threat arena.

TrendMicro data shows that the Cuba Ransomware targets a variety of places and businesses. Some of the typical targeted industries included manufacturing, professional and legal services, construction and high technology industries. It's core target regions include Europe, North America, and Asia. The countries with the greatest assault attempts were the US and Turkey. According to Bleeping Computer's recent report, Cuba Ransomware operation is using the OWASSRF exploit, also known as CVE-2022-41080, to target Microsoft Exchange servers as an initial attack vector where they were also seen using bundled software and phishing mails as their initial vectors.

Overtime, Cuba Ransomware has seen lot of modification and updates through its journey with Tactics, Techniques, and Procedures (TTPs). In this blog, we will be looking into these closely.

Attack Chain

Initial Discovery and Access



Technical analysis

The ransomware procedure begins by analysing the language being used in the system; if it determines that the language is "Russian," it ends itself, but the malware file is still left without being deleted.

```
v4 = GetKeyboardLayoutList(16, &List);
v5 = 0;
if ( v4 > 0 )
{
  while ( *((_BYTE *)&List + 4 * v5) != 0x19 )
  {
    if ( ++v5 >= v4 )
      goto LABEL_6;
  }
  sub_4098B0();
}
```

Figure 1: Russian Language is equated by 0x19

Self-destruction Feature

There is also another feature to self-delete after encryption process completes

```
if ( GetModuleFileNameW(0, &Filename, 0x104u) )
{
  lstrcpyW(&String1, L"/c del ");
  lstrcatW(&String1, &Filename);
  lstrcatW(&String1, L" >> NUL");
  GetWindowsDirectoryW(&Buffer, 0x104u);
  lstrcatW(&Buffer, L"\\system32\\cmd.exe");
  if ( CreateProcessW(&Buffer, &String1, 0, 0, 0,
  {
    CloseHandle(ProcessInformation.hThread);
    CloseHandle(ProcessInformation.hProcess);
  }
  ExitProcess(0);
```

Figure 2: Self deleting code

Command line parameters

Due to the fact that this is a debut version, several of the modes are still being tested. The version we have examined offers six distinct parameters that may be passed.

- Net
- Netscan
- Local
- Stop
- IP_addr
- specific_path

While the new version offers only four.

- Network
- Local
- IP_addr
- specific_path

We will discuss about all this in detail

"Net" keyword parameter (Removed in latest version)

This protocol uses *"NetBIOS"* Name Server to look for a host on the internal network, and after doing so, it auto deletes itself using self-destruction.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.2.15	8.8.4.4	TCP	55	51804 → 443 [ACK] Seq=1 Ack=1 Win=63525 Len=1 [TCP segment of a reassembled PDU]
2	0.000377	8.8.4.4	10.0.2.15	TCP	60	443 → 51804 [ACK] Seq=1 Ack=2 Win=65535 Len=0
3	2.515721	10.0.2.15	8.8.4.4	TCP	55	58281 → 443 [ACK] Seq=1 Ack=1 Win=62827 Len=1 [TCP segment of a reassembled PDU]
4	2.516059	8.8.4.4	10.0.2.15	TCP	60	443 → 58281 [ACK] Seq=1 Ack=2 Win=65535 Len=0
5	5.019903	203.187.215.203	10.0.2.15	TCP	60	443 → 61311 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
6	5.019903	203.187.215.203	10.0.2.15	TCP	60	443 → 61310 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
7	5.109126	10.0.2.15	203.187.215.216	TCP	55	58332 → 443 [ACK] Seq=1 Ack=1 Win=62819 Len=1 [TCP segment of a reassembled PDU]
8	5.109406	203.187.215.216	10.0.2.15	TCP	60	443 → 58332 [ACK] Seq=1 Ack=2 Win=65535 Len=0
9	6.000566	10.0.2.15	35.213.89.133	TCP	55	58998 → 443 [ACK] Seq=1 Ack=1 Win=63490 Len=1 [TCP segment of a reassembled PDU]
10	6.001017	35.213.89.133	10.0.2.15	TCP	60	443 → 58998 [ACK] Seq=1 Ack=2 Win=65535 Len=0
11	6.501879	203.187.215.203	10.0.2.15	TCP	60	443 → 61312 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
12	7.503534	203.187.215.203	10.0.2.15	TCP	60	443 → 61313 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
13	10.616759	10.0.2.15	10.0.2.255	BROWSER	216	Get Backup List Request
14	10.617048	10.0.2.15	10.0.2.255	NBNS	92	Name query NB WORKGROUP<1b>
15	11.375382	10.0.2.15	10.0.2.255	NBNS	92	Name query NB WORKGROUP<1b>
16	12.142270	10.0.2.15	10.0.2.255	NBNS	92	Name query NB WORKGROUP<1b>
17	12.581178	111.119.15.0	10.0.2.15	TCP	60	80 → 50310 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
18	12.581332	10.0.2.15	111.119.15.0	TCP	54	50310 → 80 [ACK] Seq=1 Ack=2 Win=64240 Len=0
19	13.025502	10.0.2.15	111.119.15.0	TCP	54	50310 → 80 [FIN, ACK] Seq=1 Ack=2 Win=64240 Len=0

Figure 3: NBNS

"Netscan" keyword parameter (Renamed to "Network" in latest version)

When this parameter is utilised, this option enumerates IPV4 addresses from the ARP database using the method "GetIpNetTable" This physical address is kept in an array list.

```
v7 = wcscmp(a3, L"netscan");
if ( v7 )
  v7 = -(v7 < 0) | 1;
if ( !v7 )
{
  v37 = 0;
  v31.QuadPart = 0i64;
  v32 = 0i64;
  v33 = 0i64;
  v34 = 0i64;
  v35 = 0i64;
  v36 = 0i64;
  lpMem = 0i64;
  lpMem.x = sub_407150(0, 0);
  sub_42004A(&v31, 2);
  LOBYTE(v37) = 0;
  sub_438710(&List, 0, 96);
  sub_41F3F2(&v40);
  sub_42004A((char *)&v45 + 4, 2);
  sub_4020E0((int)&List, (int)&savedregs, (int)&lpMem, 8);
  Point.x = 0;
  if ( GetIpNetTable(0, (PULONG)&Point, 0) == 122 )
  {
    v8 = (struct _MIB_IPNETTABLE *)sub_43D0A4(Point.x);
    if ( GetIpNetTable(v8, (PULONG)&Point, 0) )
```

Figure 4: Parameter "netscan" in older version

Additionally, it will simply enumerate the shares of the given IP address if one is provided.

```
wsprintfW(&servername, L"\\\\%d.%d.%d.%d", (unsigned __int8)v67, BYTE1(v67), BYTE2(v67), v67 >> 24);
do
{
  result = NetShareEnum(&servername, 1u, &bufptr, 0xFFFFFFFF, &entriesread, &totalentries, &resume_handle);
  v69 = result;
  v75 = result;
  if ( result && result != 234 )
    break;
  v70 = bufptr;
  v71 = 1;
  if ( entriesread >= 1 )
  {
    v72 = lpCriticalSection;
    do
    {
      wsprintfW(&FileName, L"%s\\%s\\*", &servername, *(_DWORD *)v70);
      if ( *((_DWORD *)v70 + 1) >= 0 )
      {
        retaddr = FindFirstFileW(&FileName, &FindFileData);
        if ( retaddr != (HANDLE)-1 )
        {
          wsprintfW(&FileName, L"%s\\%s\\", &servername, *(_DWORD *)v70);
          sub_401019(v72, &FileName);
          FindClose(retaddr);
        }
      }
      ++v71;
      v70 += 12;
    }
    while ( v71 <= entriesread );
    v70 = bufptr;
    v69 = v75;
  }
  result = NetApiBufferFree(v70);
```

Figure 5: Enumeration of Shares using IP address

This function in the older version uses Windows NT LAN Manager (NTLM) protocol that uses a challenge and response method to authenticate a client (This method is removed in the latest version)

10.0.2.15	10.0.2.2	TCP	66 58734 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
10.0.2.2	10.0.2.15	TCP	60 445 → 58734 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
10.0.2.15	10.0.2.2	TCP	54 58734 → 445 [ACK] Seq=1 Ack=1 Win=64240 Len=0
10.0.2.15	10.0.2.2	SMB	213 Negotiate Protocol Request
10.0.2.2	10.0.2.15	TCP	60 445 → 58734 [ACK] Seq=1 Ack=160 Win=65535 Len=0
10.0.2.2	10.0.2.15	SMB2	506 Negotiate Protocol Response
10.0.2.15	10.0.2.2	SMB2	282 Negotiate Protocol Request
10.0.2.2	10.0.2.15	TCP	60 445 → 58734 [ACK] Seq=453 Ack=388 Win=65535 Len=0
10.0.2.2	10.0.2.15	SMB2	590 Negotiate Protocol Response
10.0.2.15	10.0.2.2	TCP	54 58734 → 445 [ACK] Seq=388 Ack=989 Win=63252 Len=0
10.0.2.15	10.0.2.2	SMB2	220 Session Setup Request, NTLMSSP_NEGOTIATE
10.0.2.2	10.0.2.15	TCP	60 445 → 58734 [ACK] Seq=989 Ack=554 Win=65535 Len=0
10.0.2.2	10.0.2.15	SMB2	401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
10.0.2.15	10.0.2.2	SMB2	693 Session Setup Request, NTLMSSP_AUTH, User: DESKTOP-VRJEFRP\asdf
10.0.2.2	10.0.2.15	TCP	60 445 → 58734 [ACK] Seq=1336 Ack=1193 Win=65535 Len=0

Figure 6: Usage of NTLM Protocol

Additional features

The Administrative Shares folder could also be skipped using an additional special code, but it has been deleted in the recent version.

```
else
{
  v5 = *(const WCHAR **)v4;
  v6 = 0;
  lpString2 = L"IPC$";
  v13 = L"ADMIN$";
  while ( lstrcmpW(v5, (&lpString2)[v6]) )
  {
    if ( (unsigned int)++v6 >= 2 )
    {
      v2 = (void (*)(LPWSTR, LPCWSTR, ...))wsprintfW;
      goto LABEL_10;
    }
```

Figure 7: Admin Shares Special folder

"Local" keyword parameter

This parameter is used to encrypt local drives. It starts from C drive and continues to encrypt all folders and drives in Alphabetical order. It can encrypt both mounted and unmounted Drives since it targets volumes based on their ID.

Path keyword parameter

Only the path that is supplied as a command-line input can be encrypted by the ransomware.

"Stop" keyword parameter (Removed in latest version)

This Keyword does not do any operation, but just self-deletes.

List of terminated Services and processes

There is a hardcoded list of processes and services, and the Cuba ransomware looks for each of them and terminates them, one at a time. This is accomplished by using Windows API functions. This list of services and processes was quite limited in the initial versions of the Cuba ransomware, but has evolved and expanded in the recent variants.

```
{
  sub_4107A0(0, L"MySQL");
  sub_4107A0(0, L"MySQL80");
  sub_4107A0(0, L"MSSQLSERVER");
  sub_4107A0(0, L"SQLWriter");
  sub_4107A0(0, L"MSDTC");
  sub_4107A0(0, L"SQLBrowser");
  sub_410710(L"sqlservr.exe");
  sub_410710(L"sqlwriter.exe");
  sub_410710(L"msdtc.exe");
  sub_410710(L"sqlbrowser.exe");
  sub_410710(L"vmwp.exe");
  sub_410710(L"vmsp.exe");
  return sub_410710(L"vmms.exe");
}
```

Figure 8: List of process and services to be terminated

Added Feature

The most recent version uses *"SeDebugPrivilege"* to get the required access to end processes and services. The initial version did not have this functionality.

```
if ( OpenProcessToken(v0, 0x28u, &TokenHandle) )
{
  LookupPrivilegeValueA(0, "SeDebugPrivilege", &Luid);
  NewState.Privileges[0].Luid = Luid;
  NewState.PrivilegeCount = 1;
  NewState.Privileges[0].Attributes = 2;
  AdjustTokenPrivileges(TokenHandle, 0, &NewState, 0x10u, 0, 0);
}
sub_4029C9(v1, L"MySQL", -1);
sub_4029C9(v2, L"MySQL80", -1);
sub_4029C9(v3, L"SQLSERVERAGENT", -1);
sub_4029C9(v4, L"MSSQLSERVER", 4);
sub_4029C9(v5, L"SQLWriter", -1);
sub_4029C9(v6, L"SQLTELEMETRY", -1);
```

Figure 9: Privilege Escalation

Complete list of process and services to kill in the latest version

sqlagent.exe	MSDTC	MSExchangeMailboxReplication
sqlservr.exe	SQLBrowser	MSExchangeMailboxAssistants
sqlwriter.exe	vmcompute	MSExchangeIS
sqlceip.exe	vmms	MSExchangeIMAP4BE
msdtc.exe	MSExchangeUMCR	MSExchangeImap4
sqlbrowser.exe	MSExchangeUM	MSExchangeHMRecovery
vmwp.exe	MSExchangeTransportLogSearch	MSExchangeHM
vmsp.exe	MSExchangeTransport	MSExchangeFrontEndTransport
outlook.exe	MSExchangeThrottling	MSExchangeFastSearch
Microsoft.Exchange.Store.Worker.exe	MSExchangeSubmission	MSExchangeEdgeSync
MySQL	MSExchangeServiceHost	MSExchangeDiagnostics
MySQL80	MSExchangeRPC	MSExchangeDelivery
SQLSERVERAGENT	MSExchangeRepl	MSExchangeDagMgmt
MSSQLSERVER	MSExchangePOP3BE	MSExchangeCompliance
SQLWriter	MSExchangePop3	MSExchangeAntispamUpdate
SQLTELEMETRY	MSExchangeNotificationsBroker	

Extensions to skip

These extensions are skipped during the encryption

```
{
  return (unsigned __int8)sub_40B8C0(a1, L".exe")
      || (unsigned __int8)sub_40B8C0(a1, L".dll")
      || (unsigned __int8)sub_40B8C0(a1, L".sys")
      || (unsigned __int8)sub_40B8C0(a1, L".cuba")
      || (unsigned __int8)sub_40B8C0(a1, L".lnk");
}
```

Figure 10: Files with Extensions to skip

Folders to skip

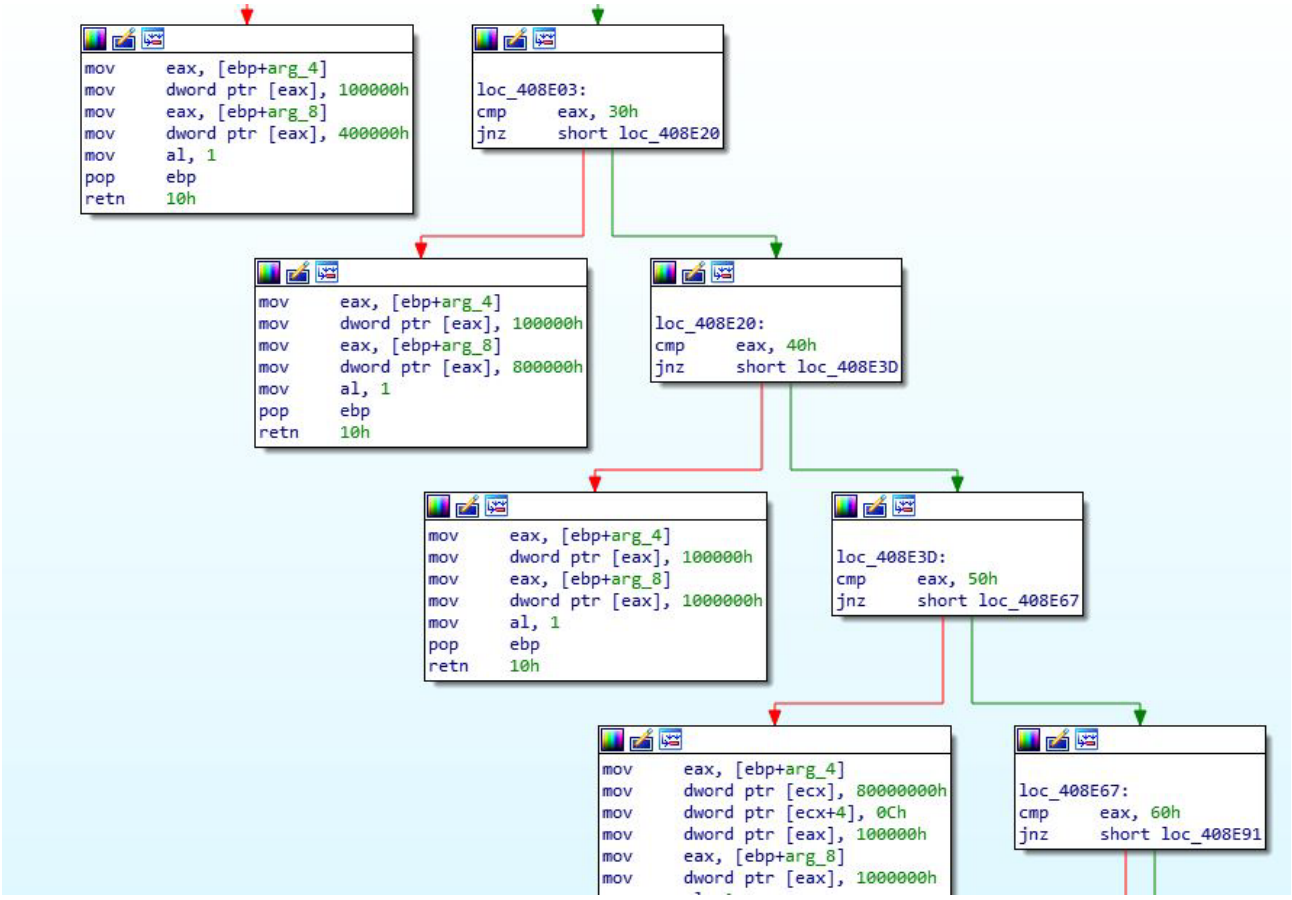
A few files and directories have been altered in the most recent version, and some have been optimised. If the file path contains the name of these folders, the encryption process would be bypassed: -

Older version	Latest version
windows	\windows\
System32	\program files\microsoft office\
Config	\program files (x86)\microsoft office\
microsoft office	\program files\avs\
$recycle.bin	\program files (x86)\avs\
boot	\$recycle.bin\
config.msi	\boot\
documents and settings	\recovery\
recovery	\system volume information\
system volume information	\msocache\
msocache	\users\all users\
all users	\users\default user\
default user	\users\default\
default	\temp\
	\inetcache\
	\google\

Encryption Routine

Files are encrypted according to size of the file

Cuba Ransomware encrypts data based on the file size for performance reasons. The entire file is encrypted if it is less than 2 Mb in size. If the file is larger, there are particular criteria to encrypt it based on its size. The original contents of the file are preserved in between each section of encryption, rendering the file useless or damaged.



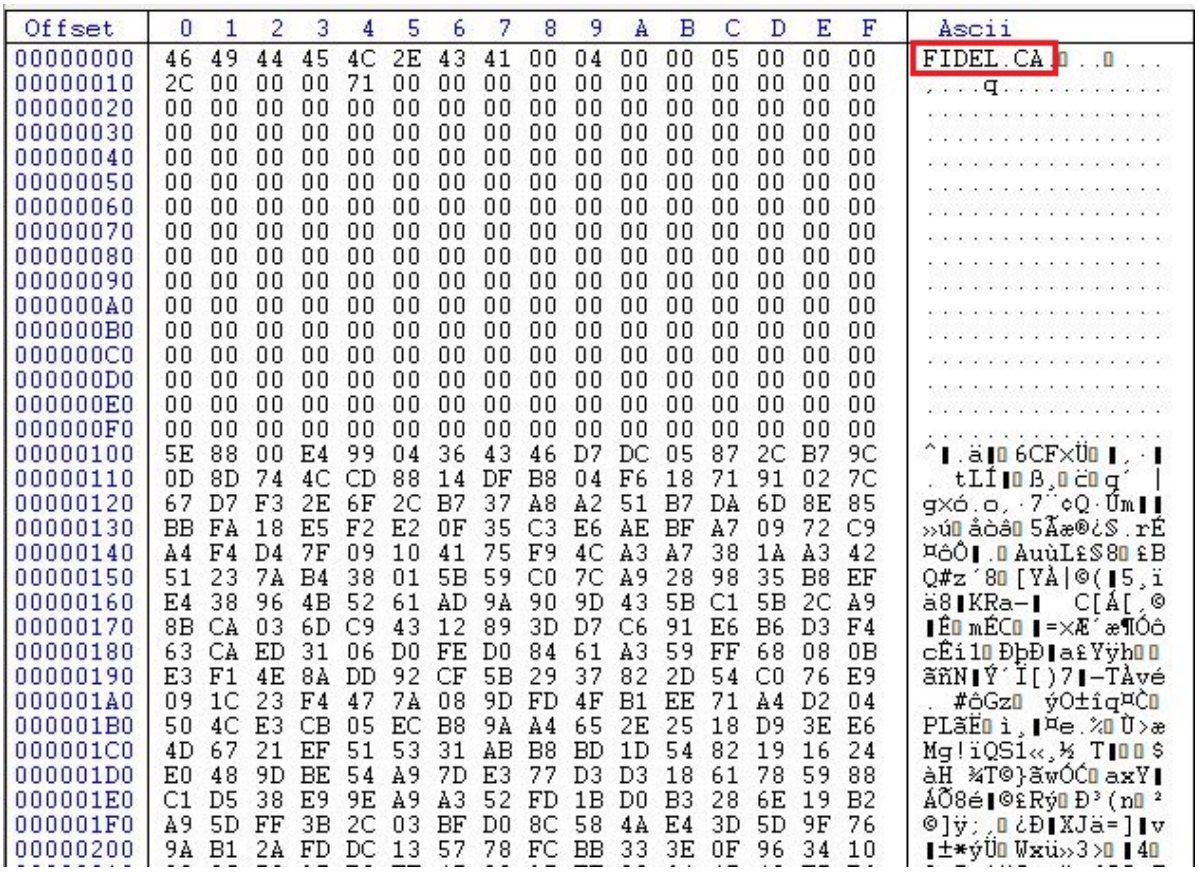
Depending on the file size, a sequential combination of encryption size and skip size is used to encrypt the whole file.

File Size	Encryption Size	Skip Size
Less than 0x200000	Entire Size	N/A
Between 0x200000 & 0xA00000	0x100000	0x400000
Between 0xA00000 & 0x3200000	0x100000	0x800000
Between 0x3200000 & 0xC800000	0x100000	0x1000000
Between 0xC800000 & 0x280000000	0x100000	0xC800000
Greater than 0x280000000	0x100000	0x1F400000

Final Encrypted file

The core Cuba Ransomware payload has remained roughly the same since its discovery in 2019. The encryption uses specifically ChaCha20 for file encryption and RSA for key encryption.

An initial 1024-byte header is prepended to every encrypted file. The first 256 bytes contain the string "FIDEL.CA" followed by zeros and random strings. The next 512 bytes consists of an RSA-4096 encrypted block, which includes the ChaCha20 KEY/IV, encrypted with a public RSA key. The remaining space is filled with zeros as padding. After encryption ".cuba" is appended to the file



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	46	49	44	45	4C	2E	43	41	00	04	00	00	05	00	00	00	FIDEL.CA
00000010	2C	00	00	00	71	00	00	00	00	00	00	00	00	00	00	00	,...q.....
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000100	5E	88	00	E4	99	04	36	43	46	D7	DC	05	87	2C	B7	9C	^ä6CF×Ü,·
00000110	0D	8D	74	4C	CD	88	14	DF	B8	04	F6	18	71	91	02	7C	tLÍß¸ö q' |
00000120	67	D7	F3	2E	6F	2C	B7	37	A8	A2	51	B7	DA	6D	8E	85	g×ó.o,·7¨¢Q·Úm
00000130	BB	FA	18	E5	F2	E2	0F	35	C3	E6	AE	BF	A7	09	72	C9	»ú åòâ 5Ãæ®¿§ rÉ
00000140	A4	F4	D4	7F	09	10	41	75	F9	4C	A3	A7	38	1A	A3	42	¤ôÔ AuùL£§8 £B
00000150	51	23	7A	B4	38	01	5B	59	C0	7C	A9	28	98	35	B8	EF	Q#z´8 [YÀ|©(5¸ï
00000160	E4	38	96	4B	52	61	AD	9A	90	9D	43	5B	C1	5B	2C	A9	ä8KRa- C[Á[,©
00000170	8B	CA	03	6D	C9	43	12	89	3D	D7	C6	91	E6	B6	D3	F4	ÊmÉC =×Æ'æ¶Óô
00000180	63	CA	ED	31	06	D0	FE	D0	84	61	A3	59	FF	68	08	0B	cÊí1 ÐþÐ a£Yÿh
00000190	E3	F1	4E	8A	DD	92	CF	5B	29	37	82	2D	54	C0	76	E9	ãñNÝ'Ï[)7-TÀvé
000001A0	09	1C	23	F4	47	7A	08	9D	FD	4F	B1	EE	71	A4	D2	04	#ôGz ýO±îq¤Ò
000001B0	50	4C	E3	CB	05	EC	B8	9A	A4	65	2E	25	18	D9	3E	E6	PLãË ì¸¤e.% Ù>æ
000001C0	4D	67	21	EF	51	53	31	AB	B8	BD	1D	54	82	19	16	24	Mg!iQS1«¸½ T $
000001D0	E0	48	9D	BE	54	A9	7D	E3	77	D3	D3	18	61	78	59	88	àH ¾T©}ãwÓÓ axY
000001E0	C1	D5	38	E9	9E	A9	A3	52	FD	1B	D0	B3	28	6E	19	B2	ÁÕ8é©£Rý Ð³(n ²
000001F0	A9	5D	FF	3B	2C	03	BF	D0	8C	58	4A	E4	3D	5D	9F	76	©]ÿ;, ¿ÐXJä=]v
00000200	9A	B1	2A	FD	DC	13	57	78	FC	BB	33	3E	0F	96	34	10	±*ýÜ Wxü»3> 4

Figure 12: Encrypted File

Ransom Note

The ransom note looks almost exactly the same as what is presented below, with the exception that the earlier version omits the second paragraph regarding sending files to servers and their reluctance to negotiate after a set amount of time.

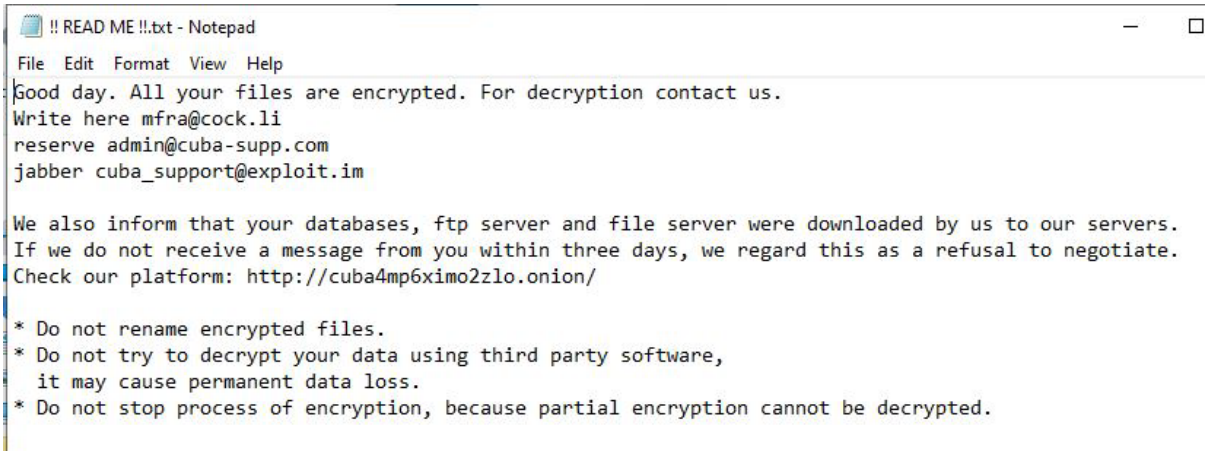


Figure 13: Encryption Note

How Quick Heal protects its users

Quick Heal and Seqrite protect their users from multiple stages of malware and attacks. In addition to static and behavioural protection mechanisms, Quick Heal prevents malicious program execution through other modules, which include URL filtering, Anti Malware protection, Cloud, and Anti Ransomware protection.

Figure 14: Generic Detection in Seqrite

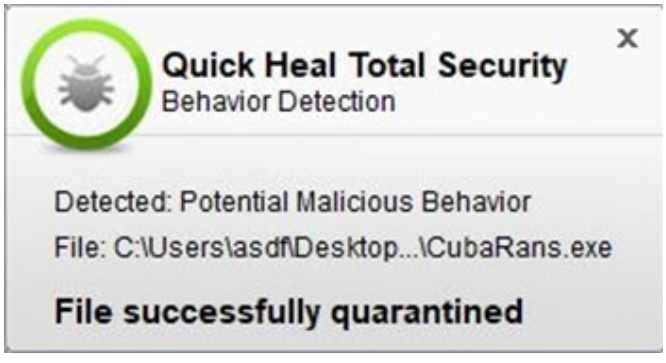


Figure 15: Behaviour Detection in Quick Heal

Conclusion

The Cuba Ransomware group is rapidly expanding its attack tactics by seeking out vulnerabilities in open attack surfaces and zero-day flaws. There is no indication of the group's activities slowing down, and it is likely that they will continue to use more advanced version of ransomware in future attacks. Users of Microsoft Exchange are advised to prioritize the patching of the OWASSRF bug to avoid potential exploitation attempts. Quick Heal and Seqrite enterprise security solutions protect its customers from such files and its functionalities. So, remember to keep the endpoint security solutions always updated.

Indicators of Compromise

00b2679e73e28343fd153df9858bc910
03c835b684b21ded9a4ab285e4f686a3
23d0033fe765242cbc07ceeab7ba3736
246b2207cfb8ef03049f11a80fba06bc
286a7aa55ea888813b6df7c047aada5d
2af30ca88d11eb0c1a4bd4f0aa0ce685
3e96efd37777cc01cabb3401485297aa
5ab2962110135f986777c938ac8bdb67
7982a49032fd9ff757a60ec271cb4ae5
a12e733ddbe6f404b27474fa0e5de61d
b8018958476178596817f734894ff64c
bcd57da0c23eae47fbe5b54db614cbc6
c0451fd7921342e0d2fbf682091d4280
c3299f7783df63fd1682b5ad63d80325
ce9c4f5439c48aeeca3bc9f2cdfaf826
d663bd6d72fa66cc0b8e64c205875ef8
d8fd19fef4605b4217cb2546c470a918
f503991495275a4d5a88b691498cbf09
f739977004981fbe4a54bc68be18ea79

Mitre Attack

ID	Name
T1566	Phishing mails
T1190	Exploit Public-Facing Application
T0807	Command-Line Interface
T1027	Obfuscated Files or Information
T1027.002	Software Packing
T1134	Access Token Manipulation
T1614.001	System Language Discovery
T1057	Process Discovery
T1007	System Service Discovery
T1082	System Information Discovery
T1135	Network Share Discovery
T1049	System Network Connections Discovery
T1106	Native API
T1489	Service Stop
T1543.003	Create or Modify System Process: Windows Service
T1486	Data Encrypted for Impact
T1070.004	Indicator Removal: File Deletion

Quick Heal Technologies Ltd.
Marvel Edge, Office No. 7010 C & D, 7th Floor, Viman Nagar, Pune,
Maharashtra, India - 411014.
Phone: 1800 212 7377 | info@quickheal.co.in | www.quickheal.com