2025–2026 Cloudflare Turnstile 与 UAM 模式防御机制及自动化绕过技术深度研究报告

1. 引言：自动化防御的演变与现状

在网络安全与数据采集的对抗历史中，2025 年至 2026 年被视为一个关键的转折点。随着自动化技术的日益精进，防御方已从传统的基于规则的阻断（如 IP 黑名单、User-Agent 过滤）全面转向基于行为分析和生物特征验证的动态防御体系。Cloudflare 作为这一领域的领军者，其 "Under Attack Mode" (UAM) 及其核心验证组件 Turnstile，代表了当前反机器（Anti-Bot）技术的最高水平。传统的验证码（CAPTCHA）要求用户识别红绿灯或斑马线，这不仅破坏了用户体验，也逐渐被先进的计算机视觉模型（如 YOLO 系列）所攻破。相比之下，Turnstile 引入了一种“无感验证”的范式，即通过收集浏览器环境的微观遥测数据（Telemetry）来判断访问者的“含人量”（Proof of Humanity），仅在信任评分不足时才降级为交互式挑战——即用户必须点击的那个复选框。

本报告旨在详尽分析 Cloudflare 在 2025–2026 周期内的防御逻辑，并基于最新的研究资料，深入探讨针对 UAM 模式及 Turnstile 点击验证的绕过技术。我们将从网络层的 TLS 指纹伪造、应用层的浏览器环境仿真、以及交互层的拟人化点击逻辑三个维度展开，特别是针对“点击”这一看似简单实则包含复杂生物识别验证的动作进行解构。

1.1 防御范式的转移：从图灵测试到遥测分析

早期的防御系统假设“只有人类才能理解图像语义”。然而，随着多模态大模型的普及，这一假设已不再成立。Cloudflare Turnstile 的设计哲学不再依赖单一的认知测试，而是构建了一个多维度的信任模型。当一个请求到达 Cloudflare 边缘节点时，系统会立即启动一系列非交互式的检查：检查传输层安全（TLS）握手的特征、HTTP/3 协议的实现细节、以及 TCP 窗口的大小等[1]。

如果网络层特征通过，JavaScript 引擎会进一步探测浏览器的执行环境。这包括但不限于 Canvas 渲染指纹、WebGL 参数、AudioContext 噪声、以及屏幕分辨率与窗口大小的比例关系。只有当这些静态特征出现异常（例如，Headless 浏览器的 navigator.webdriver 属性泄露，或 Linux 服务器上却声明自己是 Windows 操作系统）时，Turnstile 才会进入“交互模式” [3]。此时，那个著名的“Verify you are human”复选框才会出现。而在 2025 年，这个复选框本身也已演变为一个陷阱：它被封装在封闭的 Shadow DOM 中，且能够通过 Chrome DevTools Protocol (CDP) 的微小差异来识别自动化点击 [5]。

---

2. Cloudflare Turnstile 核心架构解析

要有效地绕过防御，必须首先理解其内部构造。Turnstile 并非一个简单的 HTML 元素，而是一个动态加载、高度混淆且具备环境感知能力的 JavaScript 应用程序。

2.1 信任评分机制与三种工作模式

Turnstile 的核心在于其动态调整的“信任评分”（Trust Score）。这个评分决定了用户将面临何种级别的挑战 [1]。

工作模式	触发条件	验证逻辑	用户感知
非交互模式 (Invisible)	信任评分高	后台静默运行 JS 环境检测、工作量证明 (PoW) 计算。	无感知，直接通过。
瞬时检查 (Brief Check)	信任评分中等	类似非交互模式，但会显示“正在验证...”(Verifying...) 动画，增加了时间维	短暂等待后通过。

		度的检测。	
交互模式 (Interactive)	信任评分低	强制要求用户进行点击操作。后台同时记录鼠标轨迹、点击延迟及事件冒泡链。	需手动点击复选框。

对于自动化程序而言，最理想的状态是维持高信任评分，从而完全避免出现点击挑战。然而，在大规模数据采集场景下，IP 资源的复用和指纹的微小差异往往会导致评分下降，从而强制进入交互模式。此时，“如何点击”便成了核心难题。

2.2 封闭影子 DOM (Closed Shadow DOM) 的技术壁垒

在 2025 年的更新中，Cloudflare 普遍采用了 Shadow DOM 技术来封装 Turnstile 组件，并且将其模式设置为 closed [5]。

在标准的 Web 开发中，开发者可以通过 document.querySelector 轻松获取页面上的元素。然而，Shadow DOM 创建了一个隔离的 DOM 树，原本旨在保护组件样式不被外部 CSS 污染。当模式设为 open 时，外部脚本仍可通过 element.shadowRoot 属性访问内部结构；但当设为 closed 时，该属性返回 null。这意味着，常规的自动化框架（如 Playwright 或 Puppeteer）使用的 CSS 选择器无法“穿透”这层壁垒，导致脚本抛出“元素未找到”的错误 [9]。

这一设计直接打击了依赖 DOM 结构的自动化脚本。攻击者不仅需要找到 iframe，还需要找到 iframe 内部被多层嵌套且封闭的 Shadow Root 中的 checkbox 元素。这迫使绕过技术从“结构化查询”向“视觉识别”和“协议层交互”演进。

2.3 浏览器指纹与环境一致性

Turnstile 极其依赖环境的一致性检测。它会交叉验证不同 API 返回的数据。例如，它会检查 User-Agent 声明的操作系统是否与 navigator.platform 一致，是否与 TCP/IP 协议栈的行为

（如 TTL 值）匹配。此外，它还会检测显卡渲染行为。Headless 浏览器通常使用软件渲染（如 LLVMpipe），而真实用户通常有硬件加速的 GPU。这种微小的渲染差异（如抗锯齿算法的不同）会生成不同的 Canvas Hash，从而暴露自动化行为[11]。

---

3. 网络层指纹识别与欺骗技术 (TLS/JA4)

在探讨如何“点击”之前，必须确保请求能够到达能够点击的页面。Cloudflare 在网络层部署了极为严苛的 TLS 指纹识别系统。如果 TLS 握手特征通过不了，脚本连加载 Turnstile JS 的机会都没有，直接会被拦截在 403 Forbidden 页面之外。

3.1 从 JA3 到 JA4 的指纹迭代

早期的防御系统依赖 JA3 指纹，它是 TLS 握手包中 Cipher Suites（加密套件）、Extensions（扩展）等字段的十进制值哈希。然而，Chrome 等浏览器开始引入 ClientHello 随机化机制（TLS Extension Permutation），导致同一个浏览器的 JA3 指纹在每次连接时都会变化，这使得基于固定哈希的封锁变得困难[13]。

作为回应，Cloudflare 采用了 JA4 指纹标准。JA4 不再关注扩展的绝对顺序，而是聚合了协议类型（TCP/QUIC）、TLS 版本、SNI 是否存在、以及加密套件和扩展的**数量**等特征，生成一个更加人类可读且稳定的指纹字符串（例如 t13d1516h2...）[2]。这使得它能够忽略随机化带来的噪音，精准识别出底层的 HTTP 客户端库。

3.2 Python Requests 的局限性

标准的 Python requests 库基于 urllib3 和 OpenSSL。其发出的 TLS ClientHello 数据包特征非常明显：加密套件列表较短、扩展顺序固定、缺乏浏览器特有的 GREASE 值。Cloudflare 的边缘防火墙可以轻易识别出这是脚本流量而非浏览器流量，从而直接阻断[3]。

3.3 解决方案：curl_cffi 与底层伪造

为了绕过 JA4 检测，2025 年的主流方案是使用 curl_cffi。这是一个基于 Python 的库，但在底

层调用了修改版的 curl（即 curl-impersonate），能够完美复刻 Chrome、Firefox 或 Safari 的 TLS 握手细节[15]。

关键技术点：

- **模拟浏览器版本：** 通过指定 impersonate="chrome124"，库会自动调整加密套件列表、扩展顺序和伪头部（Pseudo-headers）的排列，使其与真实的 Chrome 124 完全一致。
- **HTTP/2 与 HTTP/3 支持：** 现代浏览器倾向于使用 HTTP/2 甚至 HTTP/3 (QUIC)。如果一个声称是 Chrome 的客户端却不支持 ALPN 协商中的 h2 或 h3 协议，会被立即标记为异常。curl_cffi 支持完整的 HTTP/3 握手，确保协议栈层面的一致性[17]。

代码实现逻辑：

Python

```
from curl_cffi import requests

# 伪装成 Chrome 124，这会自动处理 TLS 握手细节和 JA4 指纹
response = requests.get(
    "https://target-cloudflare-site.com",
    impersonate="chrome124",
    headers={
        # 必须保持 Header 顺序与真实浏览器一致
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36...",
        "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9...",
        "Accept-Language": "en-US,en;q=0.9"
    }
```

)

这种网络层的伪装是所有后续自动化操作的基石。没有正确的 JA4 指纹，任何点击策略都无从谈起。

---

4. 浏览器自动化框架的代际更迭 (Nodriver/Patchright)

当请求成功通过网络层检查并加载了页面后，下一步是在浏览器环境中执行 JavaScript 并处理交互。传统的 Selenium 和 Puppeteer 由于特征明显，在 2025 年已逐渐被新一代的“隐形”框架所取代。

4.1 传统框架的衰落：WebDriver 泄露

Selenium 依赖 WebDriver 协议与浏览器通信。在启动时，它会在浏览器的 navigator 对象中留下明显的标记，即 navigator.webdriver = true。虽然可以通过脚本在加载前将其删除（Object.defineProperty），但 Cloudflare 还会检查 CDP 命令的执行痕迹。例如，开启 Runtime.enable 这一 CDP 命令通常是自动化工具的标准起手式，这在真实用户会话中是绝不会出现的 [19]。

4.2 Nodriver：异步与无 WebDriver 架构

Nodriver（前身为 undetected-chromedriver 的相关演进项目，但在 2025 年已独立发展为基于 asyncio 的高性能框架）是目前 Python 生态中最受推崇的工具之一 [1]。

核心优势：

- **摒弃 WebDriver 二进制文件：** Nodriver 不通过 chromedriver.exe 通信，而是直接连接浏览器的 CDP WebSocket 接口。这从根源上消除了 navigator.webdriver 属性的存在，使得浏览器看起来更像是一个原生的 Chrome 实例。
- **异步设计：** 专为现代 Python 的 asyncio 设计，这不仅提高了并发性能，还使得其流量模式（Traffic Pattern）与阻塞式的 Selenium 截然不同，更接近人类的浏览行为（非线性加

载）。

- **内置隐匿补丁：** 它内置了针对 Cloudflare 常见检测点（如 CDC 变量名检测）的绕过逻辑[21]。

4.3 Patchright：Playwright 的底层修正

Playwright 虽然功能强大，但其“噪音”极大。Patchright 是 Playwright 的一个分支版本，专注于在 C++ 层面修补浏览器指纹[19]。

技术修正细节：

- **Runtime.enable 漏洞修补：** 能够避免全局开启 Runtime 域，而是通过隔离的 Execution Context 执行 JS，从而规避 Cloudflare 的监听。
- **Console API 屏蔽：** 禁用了 Console.enable，防止反爬脚本通过 console 对象的行为差异来探测调试器。
- **User-Agent Client Hints (UA-CH) 对齐：** 自动确保 HTTP 请求头中的 UA 与 JS 环境中的 navigator.userAgentData 保持严格一致。

4.4 SeleniumBase UC Mode：集大成者

SeleniumBase 的 UC Mode（Undetected ChromeDriver Mode）是目前最为成熟的解决方案之一，特别是针对需要复杂交互的场景[23]。

独家功能：

- 它不仅修补了 chromedriver 的二进制特征，还提供了一套专门针对 Turnstile 的 GUI 交互 API（如 uc_gui_click_captcha）。这些 API 不走 DOM 事件，而是调用操作系统的鼠标驱动，这将在后文详细阐述。

---

5. 交互式验证挑战：点击行为的深度模拟

这是用户查询的核心：“用什么方法来绕过 Turnstile 点击”。在 2025 年，简单的 element.click() 已经失效，甚至会直接导致验证失败。

5.1 CDP 点击与“不可信事件”(Untrusted Events)

在 JavaScript 事件模型中，事件分为“可信”（Trusted）和“不可信”（Untrusted）。由用户物理硬件触发的事件（鼠标点击、键盘按键）其 isTrusted 属性为 true。通过脚本（如 document.dispatchEvent）生成的事件为 false。虽然 CDP 可以在一定程度上模拟 isTrusted 为 true 的事件，但 Cloudflare 发现了 CDP 点击的一个底层缺陷[6]。

坐标系异常（The Coordinate Bug）： 当真实鼠标点击 iframe 内的元素时，浏览器报告的 screenX 和 screenY 是相对于整个显示器屏幕的坐标（通常数值较大，如 1024, 768）。然而，通过 CDP 发送的点击命令，在某些 Chrome 版本中，其坐标是相对于 iframe 视口（Viewport）的。这意味着坐标值往往非常小（如 50, 50）。Cloudflare 利用这一差异，编写了专门的检测脚本：如果点击事件的 screenX/Y 小于特定阈值，或者与 clientX/Y 的关系不符合全屏逻辑，则判定为机器人[6]。

5.2 解决方案一：操作系统级输入模拟 (OS-Level Input Simulation)

为了解决上述问题，最有效的方法是彻底放弃浏览器内部的点击指令，转而控制操作系统的光标。

工具：

- **SeleniumBase UC Mode:** 提供了 uc_gui_click_captcha() 方法[25]。
- **PyAutoGUI / PyDirectInput:** Python 的桌面自动化库。

实现逻辑：

1. **视觉定位：** 首先通过截图和图像识别（详见第 6 节）找到 Checkbox 在屏幕上的绝对坐标。
2. **硬件驱动模拟：** 调用 Windows API 或 Linux X11 接口，发送底层的鼠标移动和点击信号。
3. **效果：** 这种点击对于浏览器来说，与其说是来自脚本，不如说是来自物理鼠标驱动。浏览器会生成完美的 Event 链（mousemove -> hover -> mousedown -> mouseup -> click），且 screenX/Y 坐标完全正确，能够完美欺骗 Cloudflare 的检测逻辑[20]。

5.3 解决方案二：键盘无障碍导航 (Tab Navigation)

另一种规避 Shadow DOM 和点击检测的方法是利用网页的无障碍（Accessibility）特性[8]。

原理：

Turnstile 必须符合无障碍标准，支持键盘操作。因此，可以通过发送键盘的 Tab 键将焦点移动到 Checkbox 上，然后发送 Space（空格键）或 Enter 键进行选中。

风险：

虽然这种方法不需要处理复杂的坐标，但 Cloudflare 同样会监测。如果一个用户在没有任何鼠标移动的情况下，以极快的速度精准 Tab 到控件并按下空格，这也是异常行为。因此，这种方法必须配合随机的按键延迟（Typing Latency）和偶尔的鼠标抖动来使用。

---

6. 视觉计算与坐标系攻防

由于 Shadow DOM 的封闭性，基于 DOM 选择器的定位方法（如 XPath, CSS Selectors）在 Turnstile 面前往往束手无策。因此，“视觉流”成为 2026 年的主流定位技术。

6.1 基于 OpenCV 的元素定位

这种方法将网页视为一张图像，而非代码结构。

技术流程：

1. **全屏截图：** 脚本控制浏览器对当前视口进行截图。
2. **模板匹配 (Template Matching)：** 预先准备 Turnstile 复选框的特征图片（如那个灰色的圆圈或 Cloudflare 的 Logo）。使用 OpenCV 的 cv2.matchTemplate 算法在截图中搜索该特征[5]。
3. **坐标换算：** 算法返回匹配区域的中心点坐标 (X, Y)。
4. **反算与点击：** 将图像坐标转换为屏幕坐标，结合前述的 OS 级模拟进行点击。

代码示例逻辑（伪代码）：

Python

```
import cv2
import numpy as np
import pyautogui

# 截取屏幕
screenshot = pyautogui.screenshot()
screenshot = cv2.cvtColor(np.array(screenshot), cv2.COLOR_RGB2BGR)

# 读取预存的 Turnstile 模板图
template = cv2.imread('turnstile_checkbox_template.png')

# 匹配
result = cv2.matchTemplate(screenshot, template, cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(result)

if max_val > 0.8: # 设定置信度阈值
    target_x = max_loc + template_width // 2
    target_y = max_loc + template_height // 2

    # 结合贝塞尔曲线移动鼠标并点击
    human_move_and_click(target_x, target_y)
```

6.2 针对 Shadow DOM 的 iframe 偏移计算

如果不使用图像识别，另一种“半盲”方法是定位包裹 Turnstile 的 iframe。虽然无法进入 iframe 内部，但 iframe 本身是在主文档（Light DOM）中的。可以通过获取 iframe 的 boundingBox（边界框）坐标，然后根据经验值（例如 Checkbox 通常位于 iframe 左上角偏移 30px, 30px 的位置）进行盲点[28]。

---

7. 行为生物识别与轨迹仿真

Cloudflare 不仅仅检测点击的那一瞬间，它记录的是点击前后的整个时空轨迹。这就是所谓的“行为生物识别”（Behavioral Biometrics）。

7.1 鼠标轨迹的贝塞尔曲线 (Bezier Curves)

人类移动鼠标绝不是两点之间的直线（线性插值）。人类的手腕运动具有弧度，且受肌肉震颤影响，轨迹会有微小的抖动。

Ghost Cursor 算法： 自动化脚本必须引入生成贝塞尔曲线的算法[5]。

- **起步与刹车：** 遵循 Fitts's Law（费茨定律），光标在启动时加速，在接近目标时减速。
- **过冲 (Overshoot)：** 真实用户往往会稍微超过目标一点点，然后修正回来。
- **随机抖动：** 在路径上加入高斯噪声，模拟手部的不稳定性。

7.2 时间维度的伪装

除了空间轨迹，时间也是关键变量。

- **点击持续时间：** mousedown 和 mouseup 之间的时间差不能是 0ms。物理微动开关的触发通常需要 50ms 到 150ms。脚本必须通过 sleep(random.uniform(0.05, 0.15)) 来模拟这一物理延迟[11]。
- **反应潜伏期：** 页面加载完成后，人类需要时间处理视觉信息。脚本不应在页面加载完成的 10ms 内立即行动，而应引入 500ms 到 2000ms 的随机“思考时间”。

---

8. 第三方解决方案与 API 集成分析

对于无法投入大量资源维护复杂绕过逻辑的团队，使用第三方 Solver API 是另一种选择。

8.1 代理不一致问题 (Proxy Mismatch)

使用 2Captcha 或 CapSolver 等服务时，一个常见的陷阱是 IP 不一致。

- **问题：** Cloudflare 的 Turnstile Token 是绑定 IP 和 User-Agent 的。如果 Solver 服务使用他们的 IP 解开了验证码，返回 Token 给你的爬虫，而你的爬虫使用另一个 IP 提交该 Token，Cloudflare 会拒绝验证，认为发生了 Token 盗用（Replay Attack）[29]。
- **解决方案：** 必须使用“Proxy-Through”模式。即在调用 API 时，将自己的代理服务器信息传给 Solver 服务，强制 Solver 使用与爬虫相同的出口 IP 进行验证。

8.2 废弃 DOM 操作，拥抱 CDP 注入

由于 Shadow DOM 的阻碍，现代 Solver 服务（如 CapSolver）提供了一种无需点击的解决方案。它们通过 CDP 注入一段特殊的 JavaScript 垫片（Shim），拦截 Turnstile 的初始化函数（turnstile.render），直接获取回调参数。这种方式绕过了物理点击的难题，直接在数据层完成验证[31]。

---

9. 新兴威胁：AI 迷宫 (AI Labyrinth) 与蜜罐技术

2025 年末，Cloudflare 推出了一项名为“AI Labyrinth”的新技术，这标志着防御策略从“阻断”转向了“消耗”[32]。

9.1 机制原理

当 Cloudflare 检测到某些虽通过了 Turnstile 验证但行为仍有可疑（例如遍历速度过快、资源下载模式单一）的爬虫时，它不再直接返回 403 错误，而是动态生成一组由 AI 实时幻造的虚假页面。

- 这些页面内容看起来非常真实，包含合理的文本、图片和链接。

- 链接指向无尽的深层结构，构成一个死循环迷宫。
- 爬虫会深陷其中，消耗大量的计算资源和时间去抓取毫无价值的垃圾数据。

9.2 应对策略

这一技术的出现意味着仅仅绕过 UAM 页面已不足够。爬虫工程师必须：

- **引入语义校验：** 使用轻量级 NLP 模型验证抓取内容的逻辑一致性，判断是否落入蜜罐。
- **控制抓取深度与广度：** 避免无差别遍历，采用精准的 URL 列表抓取。
- ** IP 隔离：** 一旦发现某个 IP 进入了迷宫，必须立即废弃该 IP，因为该指纹已被标记。

---

10. 结论与未来展望

综上所述，2025–2026 年绕过 Cloudflare UAM 及 Turnstile 点击验证，已不再是单一技术的比拼，而是全栈技术的对抗。

核心技术栈总结：

1. **网络层：** 必须使用 **curl_cffi** 配合 HTTP/3 协议，伪造完美的 JA4 指纹，确保请求不被边缘防火墙丢弃。
2. **框架层：** 弃用标准 Selenium，转而使用 **Nodriver (Python)** 或 **Patchright**，彻底移除 WebDriver 特征并修补 CDP 泄露。
3. **交互层：** 面对 Turnstile 的点击挑战，**不要使用 DOM 点击**。应采用 **OpenCV 视觉定位** 配合 **OS 级鼠标模拟 (SeleniumBase UC Mode)**，并辅以贝塞尔曲线轨迹和物理延迟，通过“图灵测试”。

未来展望：

随着硬件认证（如 Private Access Tokens, PATs）的推广，纯软件层面的绕过将变得愈发艰难。未来的自动化可能需要结合真实设备农场（Real Device Farms）或利用浏览器本身的辅助功能接口，甚至是利用 AI Agent 来动态应对不断变化的验证逻辑。对于数据采集者而言，这不仅是技术的博弈，更是成本与收益的持续权衡。

技术环节	2024 年主流方案	2025/2026 年最新方案	核心差异
TLS 指纹	JA3 伪造	**JA4 伪造 + HTTP/3**	应对扩展顺序随机化，模拟 QUIC 协议
浏览器控制	Selenium / Puppeteer	**Nodriver / Patchright**	移除 WebDriver，修补 CDP 监听漏洞
点击验证	element.click()	**视觉识别 + OS 级模拟**	绕过 Shadow DOM 和 CDP 坐标检测
验证码解决	Token 注入	**Proxy-Through + 行为模拟**	解决 Token 与 IP 绑定问题

本报告所展示的技术细节均基于截至 2026 年初的最新研究成果。由于攻防对抗的动态性，建议持续关注 Cloudflare 的技术博客及开源社区的最新补丁，以保持绕过能力的实效性。

Works cited

1. How to Bypass Cloudflare When Web Scraping in 2026 – Scrapfly, accessed January 22, 2026, https://scrapfly.io/blog/posts/how-to-bypass-cloudflare-anti-scraping
2. JA4+ Network Fingerprinting. TL;DR | by John Althouse | FoxIO | Medium, accessed January 22, 2026, https://medium.com/foxio/ja4-network-fingerprinting-9376fe9ca637
3. Bypass Cloudflare with Puppeteer (2025 Guide) — Scrape Protected Sites –

Browserless, accessed January 22, 2026, https://www.browserless.io/blog/bypass-cloudflare-with-puppeteer
4. How to Bypass Cloudflare with Playwright in 2026 – ZenRows, accessed January 22, 2026, https://www.zenrows.com/blog/playwright-cloudflare-bypass
5. Click Cloudflare Turnstile Checkbox – no bullshit – Kameleo, accessed January 22, 2026, https://kameleo.io/blog/click-cloudflare-turnstile-checkbox
6. Google patches 100% precise Cloudflare Turnstile bot check – Web Scraper, accessed January 22, 2026, https://webscraper.io/blog/google-patches-100-precise-cloudflare-turnstile-bot-check
7. How to Solve Turnstile Captcha: Tools and Techniques in 2026 – CapSolver, accessed January 22, 2026, https://www.capsolver.com/blog/Cloudflare/how-to-solve-turnstile-captcha
8. Invalid Turnstile solutions · Issue #211 – GitHub, accessed January 22, 2026, https://github.com/Theyka/Turnstile-Solver/issues/211
9. [Feature request] support closed shadow DOM · Issue #28 · Kaliiiiiiiiii-Vinyzu/patchright, accessed January 22, 2026, https://github.com/Kaliiiiiiiiii-Vinyzu/patchright/issues/28
10. Locators | Playwright Python, accessed January 22, 2026, https://playwright.dev/python/docs/locators
11. How to Bypass Cloudflare with Playwright in 2026 – BrowserStack, accessed January 22, 2026, https://www.browserstack.com/guide/playwright-cloudflare
12. 5 Working Methods to Bypass Cloudflare (August 2025 Updated) – Scrape.do, accessed January 22, 2026, https://scrape.do/blog/bypass-cloudflare/
13. Advancing Threat Intelligence: JA4 fingerprints and inter-request signals, accessed January 22, 2026, https://blog.cloudflare.com/ja4-signals/
14. How to Bypass Cloudflare in Python – ZenRows, accessed January 22, 2026,

https://www.zenrows.com/blog/bypass-cloudflare-python
15. Impersonate guide – curl_cffi documentation – Read the Docs, accessed January 22, 2026, https://curl-cffi.readthedocs.io/en/v0.7.2/impersonate.html
16. How to Solve TLS/JA3 Fingerprinting in Web Scraping with curl_cffi – CapSolver, accessed January 22, 2026, https://www.capsolver.com/blog/All/web-scraping-with-curl-cffi
17. Built an HTTP client that matches Chrome's JA4/Akamai fingerprint : r/webscraping – Reddit, accessed January 22, 2026, https://www.reddit.com/r/webscraping/comments/1q76ec1/built_an_http_client_that_matches_chromes/
18. Built an HTTP client that matches Chrome's JA4/Akamai fingerprint : r/Python – Reddit, accessed January 22, 2026, https://www.reddit.com/r/Python/comments/1q77abp/built_an_http_client_that_matches_chromes/
19. Kaliiiiiiiiii-Vinyzu/patchright: Undetected version of the Playwright testing and automation library. – GitHub, accessed January 22, 2026, https://github.com/Kaliiiiiiiiii-Vinyzu/patchright
20. How to bypass Cloudflare in 2026: 5 simple methods – Roundproxies, accessed January 22, 2026, https://roundproxies.com/blog/bypass-cloudflare/
21. The 6 best Patchright alternatives in 2026 – Roundproxies, accessed January 22, 2026, https://roundproxies.com/blog/best-patchright-alternatives/
22. How to Scrape with Patchright and Avoid Detection – ZenRows, accessed January 22, 2026, https://www.zenrows.com/blog/patchright
23. New Video Tutorial: Undetectable Automation – SeleniumBase, accessed January 22, 2026, https://seleniumbase.com/new-video-tutorial-undetectable-automation/
24. New Video: Undetectable Automation 2 (with UC Mode and Python) –

SeleniumBase, accessed January 22, 2026, https://seleniumbase.com/new-video-undetectable-automation-2-with-uc-mode-and-python/

25. Major updates have arrived in `4.28.0` (mostly for UC Mode) · Issue #2865 – GitHub, accessed January 22, 2026, https://github.com/seleniumbase/SeleniumBase/issues/2865
26. Selenium headless: How to bypass Cloudflare detection using Selenium – Stack Overflow, accessed January 22, 2026, https://stackoverflow.com/questions/68289474/selenium-headless-how-to-bypass-cloudflare-detection-using-selenium
27. Interacting with Shadow DOM – The Playwright way – Testing Mavens, accessed January 22, 2026, https://www.testingmavens.com/blogs/interacting-with-shadow-dom-the
28. Access iframe in closed shadow root + click on checkbox from iframe body #1272 – GitHub, accessed January 22, 2026, https://github.com/FlareSolverr/FlareSolverr/pull/1272
29. Top Cloudflare Challenge Solvers in 2026: Performance Rankings – CapSolver, accessed January 22, 2026, https://www.capsolver.com/blog/Cloudflare/top-challenge-solver-ranking
30. How to solve Cloudflare Turnstile Captcha with Python – CapSolver, accessed January 22, 2026, https://www.capsolver.com/blog/Cloudflare/bypass-cloudflare-turnstile-captcha-python
31. how can someone solve cloudflare turnstile captcha using python selenium wihtn 2captcha api – Stack Overflow, accessed January 22, 2026, https://stackoverflow.com/questions/76463045/how-can-someone-solve-cloudflare-turnstile-captcha-using-python-selenium-wihtn-2
32. Trapping misbehaving bots in an AI Labyrinth – The Cloudflare Blog, accessed January 22, 2026, https://blog.cloudflare.com/ai-labyrinth/
33. AI Labyrinth – Bots – Cloudflare Docs, accessed January 22, 2026, https://developers.cloudflare.com/bots/additional-configurations/ai-

labyrinth/