Trình Duyệt Thật + Token Chính Hãng + Mô Phỏng Con Người + Phiên Phân Tán = Tự Động Hóa Không Thể Phát Hiện

### Lớp 1: Stealth Trước Khi Phát Hiện

* Ẩn các dấu hiệu tự động hóa trước khi script TikTok tải
* Thay đổi dấu vân tay và thuộc tính trình duyệt
* Thiết lập môi trường trình duyệt hợp pháp

### Lớp 2: Mô Phỏng Hành Vi

* Bắt chước các mô hình tương tác của con người trong quá trình tải trang
* Mô phỏng điều hướng và thời gian thực tế
* Tạo tín hiệu tương tác người dùng chính hãng

### Lớp 3: Xác Thực & Bảo Mật

* Sử dụng cơ chế bảo mật của chính TikTok
* Trích xuất và sử dụng token xác thực hợp pháp
* Tạo header bảo mật chính hãng bằng các hàm của TikTok

### Lớp 4: Phân Phối Phiên

* Phân phối yêu cầu qua nhiều instance trình duyệt
* Ngẫu nhiên hóa proxy, token và cấu hình trình duyệt
* Tạo vẻ ngoài của nhiều người dùng độc lập

## Các Cơ Chế Stealth Chi Tiết

### 1. Làm Giả Dấu Vân Tay Trình Duyệt

#### Triển Khai

await stealth\_async(page)

#### Cách Hoạt Động

Hàm stealth\_async chèn mã JavaScript chạy **trước** khi bất kỳ script nào của website tải:

// Ẩn thuộc tính webdriver  
Object.defineProperty(navigator, 'webdriver', { get: () => false });  
  
// Làm giả plugins và permissions  
Object.defineProperty(navigator, 'plugins', {   
 get: () => [/\* dữ liệu plugin giả \*/]   
});  
  
// Mock các API phát hiện tự động hóa  
window.navigator.webdriver = false;  
delete window.navigator.\_\_proto\_\_.webdriver;

#### Tại Sao Hiệu Quả

* **Lợi thế về thời gian**: Các thay đổi xảy ra trước khi mã phát hiện của TikTok tải
* **Bao phủ hoàn toàn**: Ghi đè tất cả các phương pháp phát hiện tự động hóa thông thường
* **Làm giả chính hãng**: Sử dụng dữ liệu trình duyệt thật để tạo dấu vân tay đáng tin cậy

### 2. Tạo Dấu Vân Tay Thiết Bị Động

#### Tạo Tham Số Phiên

async def \_\_set\_session\_params(self, session: TikTokPlaywrightSession):  
 user\_agent = await session.page.evaluate("() => navigator.userAgent")  
 language = await session.page.evaluate("() => navigator.language || navigator.userLanguage")  
 platform = await session.page.evaluate("() => navigator.platform")  
 device\_id = str(random.randint(10\*\*18, 10\*\*19 - 1))  
 screen\_height = str(random.randint(600, 1080))  
 screen\_width = str(random.randint(800, 1920))  
 timezone = await session.page.evaluate("() => Intl.DateTimeFormat().resolvedOptions().timeZone")

#### Tính Duy Nhất Của Thiết Bị

Mỗi phiên xuất hiện như một thiết bị hoàn toàn khác với: - **ID thiết bị độc nhất**: Số định danh 19 chữ số được tạo ngẫu nhiên - **Độ phân giải màn hình khác nhau**: Kích thước ngẫu nhiên trong phạm vi thực tế - **Lịch sử duyệt web khác nhau**: Mô phỏng độ dài lịch sử khác nhau - **Dữ liệu trình duyệt thật**: Được trích xuất từ các instance trình duyệt Playwright thực

#### Tham Số Xác Thực

session\_params = {  
 "aid": "1988",  
 "app\_language": language,  
 "app\_name": "tiktok\_web",  
 "browser\_language": language,  
 "browser\_name": "Mozilla",  
 "browser\_online": "true",  
 "browser\_platform": platform,  
 "browser\_version": user\_agent,  
 "channel": "tiktok\_web",  
 "cookie\_enabled": "true",  
 "device\_id": device\_id,  
 "device\_platform": "web\_pc",  
 "region": "US",  
 "screen\_height": screen\_height,  
 "screen\_width": screen\_width,  
 "tz\_name": timezone,  
}

### 3. Mô Phỏng Hành Vi Con Người

#### Di Chuyển Chuột

# Mô phỏng sự kiện cuộn bằng chuột để tránh phát hiện bot  
x, y = random.randint(0, 50), random.randint(0, 50)  
a, b = random.randint(1, 50), random.randint(100, 200)  
  
await page.mouse.move(x, y)  
await page.wait\_for\_load\_state("networkidle")  
await page.mouse.move(a, b)

**Mục đích**: Tạo chuyển động con trỏ thực tế mà phân tích hành vi của TikTok hiểu là tương tác con người.

#### Hành Vi Điều Hướng

await page.goto(url)  
await page.goto(url) # hack: tiktok chặn yêu cầu đầu tiên, có thể là phát hiện bot

**Tải 2 lần**: - **Yêu cầu đầu tiên**: Thường bị chặn bởi phát hiện bot ban đầu - **Yêu cầu thứ hai**: Xuất hiện như người dùng làm mới trang tải chậm - **Mô hình giống con người**: Bắt chước hành vi thông thường của người dùng

#### Thời Gian Thực Tế

if ms\_token is None:  
 await asyncio.sleep(sleep\_after) # Chờ tạo msToken  
   
# Exponential backoff cho việc thử lại  
if exponential\_backoff:  
 await asyncio.sleep(2\*\*retry\_count)

**Chiến Lược Thời Gian**: - Độ trễ tự nhiên giữa các hành động - Chờ đợi kiên nhẫn để tạo token - Mô hình thử lại thực tế với exponential backoff

### 4. Quản Lý Token Xác Thực

#### Chèn Token

if ms\_token is not None:  
 if cookies is None:  
 cookies = {}  
 cookies["msToken"] = ms\_token  
  
formatted\_cookies = [  
 {"name": k, "value": v, "domain": urlparse(url).netloc, "path": "/"}  
 for k, v in cookies.items()  
 if v is not None  
]  
await context.add\_cookies(formatted\_cookies)

#### Trích Xuất Token Trực Tiếp

if ms\_token is None:  
 await asyncio.sleep(sleep\_after)  
 cookies = await self.get\_session\_cookies(session)  
 ms\_token = cookies.get("msToken")  
 session.ms\_token = ms\_token

**Chiến Lược Token**: - Sử dụng token hợp pháp được cung cấp khi có - Trích xuất token từ phiên trình duyệt trực tiếp khi cần - Token 100% chính hãng vì được tạo bởi chính TikTok

### 5. Tạo Header Bảo Mật Động

#### Tạo Header X-Bogus

async def generate\_x\_bogus(self, url: str, \*\*kwargs):  
 await session.page.wait\_for\_function("window.byted\_acrawler !== undefined", timeout=timeout\_time)  
 result = await session.page.evaluate(  
 f'() => {{ return window.byted\_acrawler.frontierSign("{url}") }}'  
 )  
 return result

**Phương Pháp Thiên Tài**: - **Sử dụng hàm ký của chính TikTok**: window.byted\_acrawler.frontierSign() - **Chờ script bảo mật của TikTok**: Đảm bảo hàm ký được tải - **Tạo header chính hãng**: Chữ ký bảo mật giống hệt yêu cầu hợp pháp

#### Quá Trình Ký URL

async def sign\_url(self, url: str, \*\*kwargs):  
 x\_bogus = (await self.generate\_x\_bogus(url, session\_index=i)).get("X-Bogus")  
 if x\_bogus is None:  
 raise Exception("Failed to generate X-Bogus")  
   
 if "?" in url:  
 url += "&"  
 else:  
 url += "?"  
 url += f"X-Bogus={x\_bogus}"  
 return url

### 6. Kiến Trúc Đa Phiên

#### Tạo Pool Phiên

await asyncio.gather(  
 \*(  
 self.\_\_create\_session(  
 proxy=random\_choice(proxies),  
 ms\_token=random\_choice(ms\_tokens),  
 url=starting\_url,  
 context\_options=context\_options,  
 sleep\_after=sleep\_after,  
 cookies=random\_choice(cookies),  
 suppress\_resource\_load\_types=suppress\_resource\_load\_types,  
 timeout=timeout,  
 )  
 for \_ in range(num\_sessions)  
 )  
)

#### Lựa Chọn Phiên Ngẫu Nhiên

def \_get\_session(self, \*\*kwargs):  
 if kwargs.get("session\_index") is not None:  
 i = kwargs["session\_index"]  
 else:  
 i = random.randint(0, len(self.sessions) - 1)  
 return i, self.sessions[i]

**Chiến Lược Phân Phối**: - Nhiều phiên trình duyệt độc lập - Phân phối proxy ngẫu nhiên qua các phiên - Token xác thực khác nhau cho mỗi phiên - Lựa chọn phiên ngẫu nhiên cho mỗi yêu cầu

### 7. Xác Thực Header Yêu Cầu

#### Chặn Header

request\_headers = None  
  
def handle\_request(request):  
 nonlocal request\_headers  
 request\_headers = request.headers  
  
page.once("request", handle\_request)

#### Sử Dụng Header

if headers is not None:  
 headers = {\*\*session.headers, \*\*headers}  
else:  
 headers = session.headers

**Chiến Lược Header**: - Bắt header trình duyệt chính hãng từ yêu cầu thật - Bao gồm User-Agent, Accept-Language và dấu vân tay hợp pháp khác - Kết hợp với header bổ sung trong khi duy trì tính chính hãng

### 8. Tối Ưu Hóa Tải Tài Nguyên

#### Chặn Tài Nguyên Có Chọn Lọc

if suppress\_resource\_load\_types is not None:  
 await page.route(  
 "\*\*/\*",  
 lambda route, request: route.abort()  
 if request.resource\_type in suppress\_resource\_load\_types  
 else route.continue\_(),  
 )

**Lợi Ích Hiệu Suất**: - Tải trang nhanh hơn (xuất hiện giống con người hơn) - Giảm bề mặt phát hiện - Sử dụng băng thông thấp hơn (bắt chước người dùng có ad blocker) - Ít yêu cầu mạng hơn để phân tích

### 9. Cơ Chế Phục Hồi Linh Hoạt

#### Điều Hướng Dự Phòng

if attempts == max\_attempts:  
 raise TimeoutError(f"Failed to load tiktok after {max\_attempts} attempts")  
  
try\_urls = [  
 "https://www.tiktok.com/foryou",   
 "https://www.tiktok.com",   
 "https://www.tiktok.com/@tiktok",   
 "https://www.tiktok.com/foryou"  
]  
await session.page.goto(random.choice(try\_urls))

**Chiến Lược Phục Hồi**: - Nhiều URL dự phòng cho các trang TikTok khác nhau - Lựa chọn ngẫu nhiên ngăn chặn mô hình có thể dự đoán - Mô phỏng người dùng điều hướng đến các phần khác khi trang không tải

## Luồng Yêu Cầu Với Tích Hợp Stealth

### Quá Trình Yêu Cầu Hoàn Chỉnh

async def make\_request(self, url: str, headers: dict = None, params: dict = None, retries: int = 3, exponential\_backoff: bool = True, \*\*kwargs):  
 # 1. Lấy phiên ngẫu nhiên  
 i, session = self.\_get\_session(\*\*kwargs)  
   
 # 2. Kết hợp tham số phiên  
 if session.params is not None:  
 params = {\*\*session.params, \*\*params}  
   
 # 3. Sử dụng header chính hãng  
 if headers is not None:  
 headers = {\*\*session.headers, \*\*headers}  
 else:  
 headers = session.headers  
   
 # 4. Đảm bảo xác thực msToken  
 if params.get("msToken") is None:  
 if session.ms\_token is not None:  
 params["msToken"] = session.ms\_token  
 else:  
 cookies = await self.get\_session\_cookies(session)  
 ms\_token = cookies.get("msToken")  
 params["msToken"] = ms\_token  
   
 # 5. Ký URL với X-Bogus chính hãng  
 encoded\_params = f"{url}?{urlencode(params, safe='=', quote\_via=quote)}"  
 signed\_url = await self.sign\_url(encoded\_params, session\_index=i)  
   
 # 6. Thực thi yêu cầu với logic thử lại  
 result = await self.run\_fetch\_script(signed\_url, headers=headers, session\_index=i)  
   
 return json.loads(result)

## Phân Tích Hiệu Quả

### Tại Sao Phương Pháp Này Hiệu Quả

1. **Môi Trường Trình Duyệt Chính Hãng**
   * Sử dụng trình duyệt Playwright thật, không phải user agent giả
   * Môi trường thực thi JavaScript chính hãng
   * DOM và API trình duyệt thật
2. **Cơ Chế Bảo Mật Của Chính TikTok**
   * Sử dụng hàm ký của TikTok cho header bảo mật
   * Trích xuất token xác thực thật từ phiên trực tiếp
   * Tận dụng quản lý cookie và phiên của chính TikTok
3. **Mô Hình Hành Vi Thực Tế**
   * Thời gian tự nhiên giữa các hành động
   * Di chuyển chuột và điều hướng giống con người
   * Mô hình thử lại và xử lý lỗi thực tế
4. **Dấu Vân Tay Phân Tán**
   * Nhiều dấu vân tay thiết bị độc nhất
   * Địa chỉ IP khác nhau qua xoay vòng proxy
   * Cấu hình trình duyệt và token khác nhau

### Khả Năng Chống Phát Hiện

Hệ thống có khả năng chống phát hiện cao vì:

* **Không có chữ ký giả**: Mọi thứ đều sử dụng dữ liệu trình duyệt thật
* **Thông tin xác thực chính hãng**: Token và header được TikTok tạo chính hãng
* **Thời gian giống con người**: Độ trễ tự nhiên và mô hình tương tác
* **Kiến trúc phân tán**: Xuất hiện như nhiều người dùng độc lập
* **Tự phục hồi**: Cơ chế dự phòng và phục hồi tự động