爬虫代理（以langchain为框架）

原版爬取信息逻辑

调用 `GoogleSearcher.search_keyword`，在 Google 上以关键词（如 “ActiveMQ 5.17.3 exploit poc”）进行检索，将前十个网页抓取并过滤（剔除无关域名、翻译成英文、裁剪无效内容），再用 LLM 辅助判断哪些文档“直接有助于渗透”。最终将标记为“useful”的页面及其原文本保存到本地。

对于 GitHub 维度，类似地调用 `GithubSearcher.search_keyword`，主要检索公开仓库中与 CVE、exploit 相关的代码或 POC。

研究点：研究如何结合无头浏览器（Headless Browser）与 LLM 来动态模拟人类访问行为，降低被反爬机制屏蔽的概率，并评估其对获取深层次 PoC 的提升效果。

反爬机制通常通过以下方式检测爬虫：

- 请求频率过高
- 请求头信息（如User-Agent）不符合正常浏览器
- 行为模式（如点击流、鼠标移动）异常
- IP地址异常（短时间内同一IP大量请求）

传统爬虫思路

- User-Agent 与 Header 伪装
- IP/Proxy 池轮换
- 头部请求延迟

拟人化行为的研究

- 鼠标轨迹 / 触摸轨迹模拟
- 页面滚动与点击节奏建模
- 脚本化 JS 挑战绕过

在打开搜索结果和目标链接时，使用 Headless 浏览器（Playwright）模拟人类行为（随机滚动、鼠标轻微移动、随机停留、随机 User-Agent、代理 IP 轮换等），并且所有“动作”都由一个内置的 LLM做高层次的决策：

LLM 根据“当前抓取到的 HTML 文本片段”、“历史动作序列”、“被拦截信号”（如 403/JS 验证跳转）等信息，动态决定下一步动作，降低被静态反爬规则屏蔽的概率

- 构造搜索 Query：`"<服务名> <版本号> CVE list"`（例如 `"ActiveMQ 5.17.3 CVE list"`）。
- 让 Agent 使用“Headless 浏览器 + LLM 驱动动作决策” 去 Google上查询这条关键词。
- 在搜索结果列表中，找到最有可能包含 CVE 列表的链接（例如某些安全博客、CVE 官方页面、厂商安全公告）。
- 打开这些页面后，抓取渲染后的 HTML，将其中符合 `CVE-XXXX-YYYY` 格式的字符串提取出来，汇总去重，得到一个“CVE ID 列表”

每当浏览器打开一个新页面，或执行完一波动作后，我们都会用 LLM 来判断“是否已经找到所需信息”,如果没有，就让 LLM 输出下一步应该模拟的人类动作序列