

1. 研究对象

- 研究背景：传统渗透测试依赖人工专家，耗时且成本高，而现有自动化方法（如基于MDP或强化学习的框架）在灵活性、适应性和实施上存在不足
- 具体对象：目前LLM主要用于改进内部评估，外部较少，因此本文针对以客户端为主的外部评估（如Web应用程序、服务为目标）

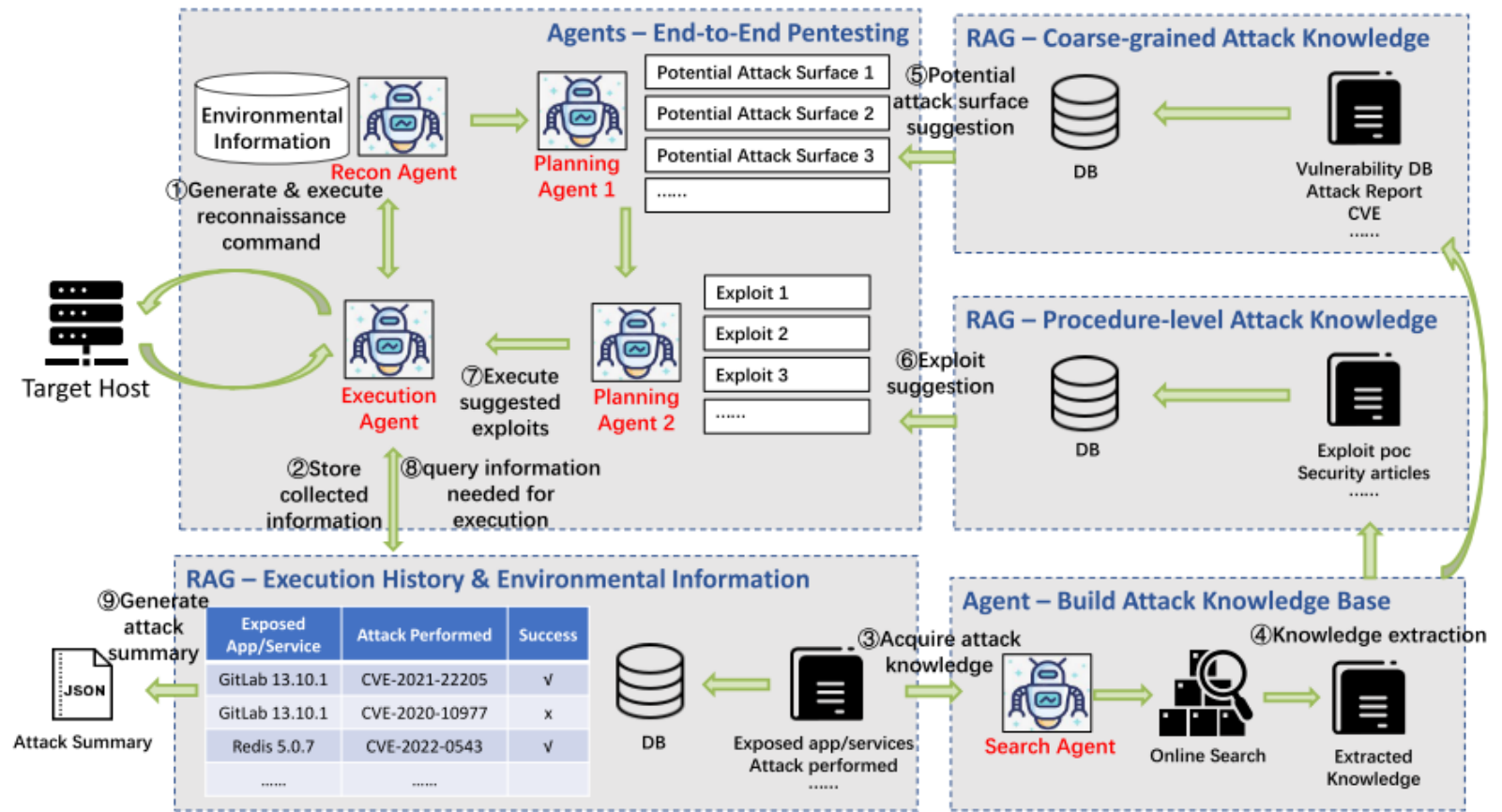
2. 研究目的

目标：通过LLM增强的多智能体协作，构建端到端的自动化渗透测试框架，减少人工参与，提升效率和成功率

- 解决现有问题=>LLM应用于渗透测试的挑战：
  - 知识局限：传统LLM在渗透测试领域的知识覆盖不足（如CVE漏洞库、特定工具链）
  - 短期记忆：由于上下文窗口的限制，模型可能会忘记之前的操作，导致执行冗余的重复任务，或出现上下文丢失，模型无法提供执行漏洞利用的详细说明
  - 自动化不足：现有工具依赖人工干预，难以动态适应复杂环境（如防御机制变化）
  - workflow集成：
    - 输出质量控制：要求LLM以符合预定义标准或协议的结构化格式生成输出
    - 有状态工作记忆管理：每个阶段通常需要不同的状态工作记忆集，包括漏洞发现、漏洞选择、目标环境细节、上下文信息等，如何实现平稳过渡

3. 具体流程

- 技术实现
  - LLM代理：配备额外工具的LLM，如：可以在线搜索和学习渗透测试知识，解决知识局限的问题；
  - RAG增强技术：
    - 索引（Indexing）：将外部知识库（如文档、数据库）转化为高效检索的结构化表示
    - 检索（Retrieval）：根据用户输入的查询，从索引中快速定位最相关的知识片段，动态补充模型上下文，解决短期记忆问题
    - 响应合成（Response Synthesis）：将检索到的信息与原始查询结合，生成最终回答，有效支持与维护有状态工作记忆管理
  - 多智能体协作：各智能体分工明确，通过结构化输出和状态管理实现流程衔接
  - 思维链CoT（Chain of Thought）：将复杂问题拆解为多个中间步骤，为每个工作流程指定一个停止条件以避免各智能体进入无限循环
  - 自适应的执行流程：要求LLM进行角色扮演，有明确目标以及边界操作；同时引入自反思机制，通过LLM对失败任务进行分析并生成备选方案，同时通过JSON格式约束输出，减少随机性和错误传播
- 框架设计：四个组件执行三个主要阶段
  - 情报收集（Reconnaissance Agent）：侦察代理收集有关目标主机的环境信息，编译目标环境摘要，存储在环境信息数据库中
  - 漏洞分析（Search & Planning Agent）：
    - 搜索代理查询环境信息数据库以检索目标主机上暴露的服务和应用程序列表，搜索潜在的攻击面和程序
    - 计划代理利用RAG技术来查找潜在攻击面列表，确定目标环境的合适漏洞利用以及攻击计划
  - 漏洞利用（Execution Agent）：执行代理尝试在目标主机上执行攻击计划，生成与记录全面的渗透测试报告



4. 实验设置

- **受害者机器**：在Ubuntu 22.04 LTS上运行模拟的易受攻击应用程序，禁用所有需要在端口上监听的服务，比如SSH
- **攻击者机器**：在Kali Linux 2024.1上运行，包括Kali Linux中可用的所有预安装工具，没有安装其他工具
- **LLM模型**：我们使用OpenAI GPT-3.5和GPT-4模型

两者通过**NAT**保持网络连接，在受害者机器上创建易受攻击的容器，并将网络参数设置为受害者机器的IP，允许攻击者机器直接访问受害者机器容器中托管的易受攻击环境

5. 研究方向的拓展

- **因果推理与攻击链优化**：
  - **问题**：现有攻击链生成依赖概率推断，缺乏因果逻辑支持，可能导致低效或不可行的路径选择
  - **方案**：引入因果图模型（Causal Graph），基于因果关系优先选择高概率且可行的攻击路径，提升攻击链的合理性和成功率
- **记忆增强的长期规划**：
  - **挑战**：LLM的上下文窗口限制导致历史经验无法有效存储，影响长期任务的连贯性
  - **解决**：设计高效的外部记忆库，结合记忆蒸馏技术**优化存储**，防止**记忆爆炸问题**
- **云原生或其他场景扩展**：将框架扩展至云原生环境（如Kubernetes、容器）或物联网等新领域

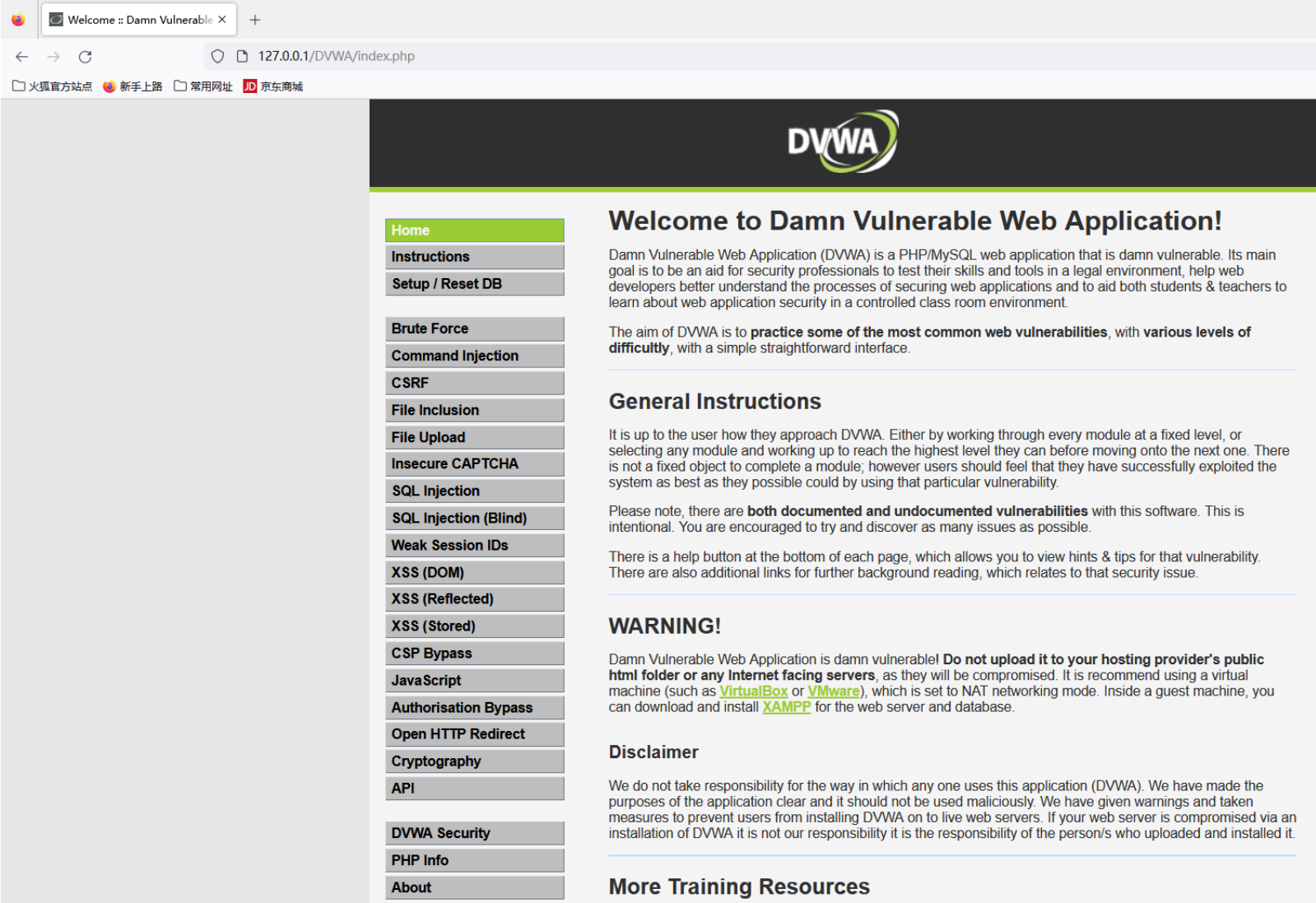
渗透测试学习

漏洞测试基准

- **PENTESTGPT基准测试[PENTESTGPT\_USENIX\_2024]**：包括13个目标和182个子任务，涵盖OWASP十大漏洞以及常见弱点枚举(CWE)项目，基于这个基准，可以对LLM提供提示和目标机器信息，逐步生成渗透测试操作
- **PentestAgent基准测试[PENTESTGPT\_USENIX\_2024]**：选择vulhub：[基于 docker-compose 的预构建易受攻击环境](#)作为基准数据集，同时用漏洞评分系统(CVSS)了解漏洞利用的难度，并通过漏洞预测评分系统(EPSS)了解脆弱环境的现实程度，编制了一个包含67个渗透测试目标的基准

搭建DVWA环境

- **DVWA介绍**：包含数据 TOP10的所有攻击漏洞的练习环境，分别为 Low，Medium，High，Impossible，级别越高，安全防护越严格，渗透难度越大



- **攻击模块**：包括不限于十个经典攻击模块，分别如下

| 攻击名称                   | 特点                            | 攻击方式   |
|------------------------|-------------------------------|--|
| 暴力破解Brute_Force        | 通过大量尝试破解认证凭证，依赖弱口令或默认配置       | 使用工具（如Hydra、Burp Suite）进行字典攻击或穷举                               |
| 命令行注入Command_Injection | 注入操作系统命令，直接控制服务器              | 输入拼接恶意命令（如；rm -rf / 或   cat /etc/passwd ）                      |
| 跨站请求伪造CSRF             | 利用用户已认证的会话，诱导执行非预期操作          | 构造恶意链接或表单（如  ） |
| 文件包含File_Inclusion     | 动态包含文件时未限制路径，分LFI（本地）和RFI（远程） | LFI： ../../etc/passwd ； RFI： 包含远程恶意脚本（需 allow_url_include 开启）  |

| 攻击名称                       | 特点                         | 攻击方式  |
|----------------------------|----------------------------|---|
| 文件上传File_Upload            | 上传恶意文件到服务器并执行              | 绕过扩展名检查（如 .php5 ）、伪造文件头（图片含PHP代码）                       |
| 不安全的验证码Insecure_CAPTCHA    | 验证码可被绕过或自动化破解，使暴力攻击有效      | 前端绕过（直接提交表单）、OCR识别简单验证码                                 |
| SQL注入SQL_Injection         | 通过输入篡改SQL查询，直接操作数据库        | 输入 ' OR '1'='1 绕过登录； UNION SELECT 窃取数据                  |
| SQL盲注SQL_Injection_(Blind) | 无直接回显，通过布尔或时间延迟推断数据        | 布尔盲注： AND 1=1 ； 时间盲注： IF(1=1,SLEEP(5),0)                |
| 反射型跨站脚本XSS_(Reflected)     | 恶意脚本通过URL参数注入并即时反射，需用户点击触发 | 构造含JavaScript的URL（如 ?search=<script>alert(1)</script> ） |
| 存储型跨站脚本XSS_(Stored)        | 恶意脚本存储到服务器（如评论），所有访问者自动执行  | 提交含脚本的内容（如 <script>stealCookie()</script> ）             |

- 反射型跨站脚本XSS\_(Reflected)：low级别的代码只是判断了name参数是否为空，如果不为空的话就直接打印出来，并没有对name参数做任何过滤和检查，没用进行任何的对XSS攻击的防御措施，存在非常明显的XSS漏洞，用户输入什么都会被执行；

家

指示

设置 / 重置数据库

蛮力

命令注入

CSRF

文件包含

文件上传

不安全的 CAPTCHA

SQL 注入

SQL 注入 (盲目)

弱会话 ID

XSS (多姆)

XSS (反射)

XSS (存储)

CSP 旁路

JavaScript (英语)

Authorisation Bypass

打开 HTTP 重定向

密码学

应用程序接口

DVWA 安全性

## 漏洞：反射型跨站脚本（XSS）

你叫什么名字? lx

提交

Hello lx

### 更多信息

- <https://owasp.org/www-community/attacks/xss/>
- <https://owasp.org/www-community/xss-filter-evasion-cheatsheet>
- [https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <https://www.cgisecurity.com/xss-faq.html>
- <https://www.scriptalert1.com/>

Cross Site Scripting (XSS) | OWASP

https://owasp.org/www-community/attacks/xss/#

OWASP

PROJECTS CHAPTERS EVENTS ABOUT

## Cross Site Scripting (XSS)

**Author:** KirstenS

**Contributor(s):** Jim Manico, Jeff Williams, Dave Wichers, Adar Weidman, Roman, Alan Jex, Andrew Smith, Jeff Knutson, Imifos, Erez Yalon, Kingthorin, Vikas Khanna. Grant Ongers

### Overview

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different

测试：用alert进行弹窗测试验证是否存在XSS，输入<script>alert('hack')</script> 查看返回结果

- 测试：用alert进行弹窗测试验证是否存在XSS，输入<script>alert('hack')</script> 查看返回结果

127.0.0.1 显示

hack

确定

- 结论：成功弹窗，说明存在XSS漏洞并且可利用，如获取cookie：编写一个 cookie.php 文档用于获取页面的 cookie，放置在指定目录下

| phpstudy_pro | WWW             | DVWA     | vulnerabilities | xss_r |  |  |
|--------------|-----------------|----------|-----------------|-------|--|--|
| 名称           | 修改日期            | 类型       | 大小              |       |  |  |
| help         | 2023/3/7 11:05  | 文件夹      |                 |       |  |  |
| source       | 2023/3/7 11:05  | 文件夹      |                 |       |  |  |
| cookie.php   | 2023/4/7 11:30  | PHP 源文件  | 1 KB            |       |  |  |
| cookie.txt   | 2023/4/7 11:31  | Text 源文件 | 1 KB            |       |  |  |
| index.php    | 2023/2/25 19:07 | PHP 源文件  | 2 KB            |       |  |  |