エージェントアンプ攻撃: AIエージェントシステムによるアンプ攻撃

久井 雅史[1,a)]　坂上 達哉[1]　矢内 直人[1]

概要：AI エージェントシステムは人間によるプロンプト入力に基づいて外部リソースを活用することで自律的にタスクを実行するシステムであり, そのセキュリティ上の問題を明らかにすることは急務である. 本稿では, AI エージェントシステムを悪用した新たなサイバー攻撃手法として, エージェントアンプ (Agent Amp, AA) 攻撃を示す. AA 攻撃は, 攻撃者が巧妙に作成したプロンプトを入力することで, AI エージェントシステムが外部リソースに大量のリクエストを送信し, 結果として膨大なトラフィックを発生させる手法である. LangChain を用いた実験により, プロンプトのデータサイズに対して増幅されたトラフィックと CPU 負荷を確認し, AA 攻撃が有効であることを実証した. さらに, 複数の大規模言語モデル (Large Language Models, LLM) を比較し, LLM がツールを利用する能力が AA 攻撃の効果に影響することを示した. また, AA 攻撃の潜在的な対策を提示し, 現行法や事例からシステム管理者の責任問題を議論する.

キーワード：AI エージェントシステム, アンプ攻撃, LLM, LangChain

A Novel Attack Using AI Agent System

Masashi Hisai[1,a)]　Tatsuya Sakagami[1]　Naoto Yanai[1]

Abstract: An AI agent system is a kind of system that autonomously performs a task based on prompts input by humans, and the security analysis of it is an urgent and crucial theme. In this paper, we present a new attack called agent amplification (AA). The AA attack is a kind of attack that an AI agent system generates a large amount of traffic to external resources. Through experiments of the presented attack with LangChain, which is an open-source library for AI agent systems, we identify that it is feasible for an AI agent system to increase a large amount of traffic to an external resource and consequently, the AA attack is feasible. We also show that our attack is more successful in proportion to the capability of an AI agent system to deal with external resources. We also discuss several security implications of the AA attack, including its countermeasures and accountability.

Keywords: AI agent system, amplification attack, LLM, LangChain

1. 序論

大規模言語モデル (Large Language Models, LLM) をはじめとする AI 技術は, 近年急速に発展し, さまざまな場面で利用されている. 特に注目されているのは, LLM を用いてユーザの入力（プロンプト）に基づいて外部リソースと連携し, 自律的にタスクを実行する AI エージェントシステムである. これらの技術が発展するなかで新たなセキュリティ上の脅威も示されている [7]. その一つが, プロンプトを通じて LLM に不正な出力をさせ [*1], 外部リソースへのサイバー攻撃を実行させる手法である [15, 17]. しかしながら, AI エージェントシステムを悪用することで外部リソースにどのようなサイバー攻撃が可能になるかについて

[1] パナソニック ホールディングス株式会社, 大阪府門真市大字門真 1006 番地, Panasonic Holdings Corporation, 1006 Kadoma, Kadoma City, Osaka
[a)] hisai.masashi@jp.panasonic.com

[*1] OWASP: https://owasp.org/www-project-top-10-for-large-language-model-applications/

は, まだ十分に検討されていない.

上述の背景に基づいて, 本稿では以下の問いを明らかにする: **AI エージェントシステムが連携する外部リソースのサービスを停止させるような攻撃は可能か？**

本稿では, 上記の問いに対する答えとして, 新たな攻撃手法であるエージェントアンプ (Agent Amp, AA) 攻撃を示す. AA 攻撃は, AI エージェントシステムがユーザのプロンプトに従って外部リソースと連携する機能を悪用し, 外部リソースへ大量のリクエストを送信させる攻撃である. AA 攻撃を通じて, 攻撃者はわずかなデータサイズのプロンプト入力によって, AI エージェントシステムを介して外部リソースへのトラフィック量を増幅させ, その可用性を損なうことが可能となる. 広く利用されている AI エージェントシステム開発のオープンソースフレームワークである LangChain[*2]を用いた実験を通じて, 攻撃者によるプロンプトに対してトラフィック量が大幅に増加することを示す. これは AA 攻撃が実際に可能であることを意味する.

上述した議論は, 外部リソースに送信されるデータ量を, AI エージェントシステムを介して増幅させることで障害を引き起こすアンプ攻撃 [13] に位置づけられる. 既存研究 [10, 12, 20] において, AI エージェントシステム自体のサービスを停止させる攻撃が主に検討されてきたのに対して, 本稿では AI エージェントシステムと連携する外部リソースの可用性を標的とする攻撃という点で異なる.

本稿では, さらに以下の三点の知見も得た. 第一に, AI エージェントシステムが内包する LLM のツールを利用する能力によりトラフィック量が異なること, すなわち, AA 攻撃の効果が左右されることを示した. 第二に, AA 攻撃の潜在的な対策についても議論した. 第三に, AA 攻撃により生じる脅威として, 攻撃そのものによるサービスの停止に加え, 現行の法律を含む事例を通じて責任の所在が AI エージェントシステムの管理者に及ぶ可能性を議論した.

本稿の貢献を以下に要約する:

- 新たな攻撃手法である AA 攻撃の手順を示し, さらに具体的な攻撃シナリオも示した.
- LangChain を用いて AI エージェントシステムの実装を行い, 実際に AA 攻撃が成立することを実証した.
- 複数の LLM を比較し, モデルのツールを利用する能力が攻撃に影響することを示した.
- AA 攻撃に対して, AI エージェントシステムの挙動に基づく対策手法を提案した.
- 現在の法律含む事例から, AA 攻撃により AI エージェントシステムの管理者へ責任が及ぶ可能性を示した.

2. 背景

本節では技術的な背景として, AI エージェントシステムの概要と, 本稿で扱うシステムの構成を定義する.

[*2] LangChain: https://www.langchain.com/

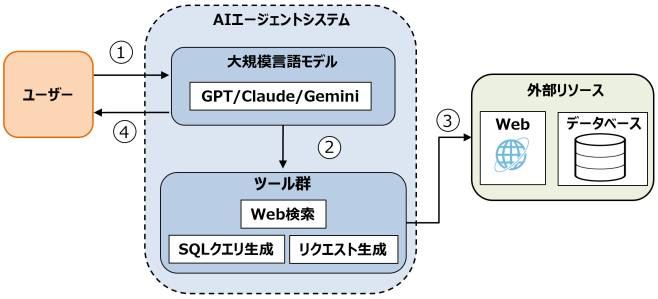

図 1: AI エージェントシステムの全体図

2.1 AI エージェントシステムの概要

AI エージェントシステムは, ChatGPT に代表されるような LLM を活用して特定のタスクを自律的に実行する. 大まかには, ユーザからの自然言語による指示を含むプロンプトを理解し, データベース接続, Web 検索などの外部 API の呼び出しや多様なツールを組み合わせることで, 外部知識を活用した Q&A [21] など様々な処理が可能となる.

2.2 システム構成

本稿では, 図 1 に示すような構成を持つ AI エージェントシステムを対象とする. システムは, ユーザ, AI エージェント, 外部リソースの三つのエンティティから構成される. まずユーザはプロンプトを作成し，Web フォームなどのユーザインターフェースを通じて ①AI エージェントにプロンプトを入力する. AI エージェントは，LLM を用いて ① で取得したプロンプトから指示を解釈し，② ツール群からタスクに必要なツールを選択する. 選択したツールを用いて，AI エージェントは ③Web サーバやデータベースなどの外部リソースから情報を取得する. ここで重要な点は，AI エージェントがタスクを遂行する際，② と ③ のプロセスを必要に応じて繰り返し実行することである. 例えば，複数の Web ページから情報を収集する場合や，データベースに対して段階的なクエリを実行する場合，AI エージェントは ② でツールを選択し，③ で外部リソースに接続し, 取得した情報を基に次の行動を決定するという処理を複数回繰り返す. 最終的に，AI エージェントは収集・処理した情報を統合し，④ ユーザに対して出力を返す.

上述のシステム構成では，AI エージェントシステムは外部リソースへの接続を仲介する役割を果たす. 一般には不正な利用を防ぐため, 不適切なプロンプト入力を遮断するフィルタリング機能が導入される [8].

3. エージェントアンプ攻撃

本節では攻撃者が AI エージェントシステムを利用する攻撃として, AA 攻撃を示す. まず攻撃者の設定を述べたのち, 攻撃の概要と詳細な攻撃手順を述べる. それから AA

攻撃が適用される状況として攻撃シナリオを述べる．

3.1 攻撃者の設定

攻撃者の能力を以下に定義する．攻撃者は，Web ブラウザ上など入力フォームを通じて，AI エージェントシステムにプロンプトを入力できる．また，複数のユーザアカウントの利用が可能である一方，攻撃者は AI エージェントシステムの管理者権限は持たず，内包される LLM とツール群，および，外部リソースの直接操作はできない．また，AI エージェントシステム内の LLM とツール群の間，及び，AI エージェントシステムと外部リソースの間で行われる通信には介入および改ざんはできない．加えて，不適切なプロンプトに対するフィルタリング機能は導入されているとする．

攻撃者の目標は，自ら作成したプロンプトを AI エージェントに入力することで，AI エージェントシステムから外部リソースへの大量のトラフィックを生じさせ，外部リソースに障害を発生させることである．このような攻撃を行う目的は，外部リソースの機能を停止させることで，AI エージェントシステム自体に経済的損失を与えることにある．

3.2 攻撃の概要

AA 攻撃は，入力するプロンプトを調整し，AI エージェントシステムから外部リソースを呼び出す機能を悪用することで，外部リソースの資源を枯渇させる攻撃である．これは既存研究 [10, 12, 20] と比べて，AI エージェントシステムを介して外部リソースへの負荷を増幅させる点が異なる．AA 攻撃の特徴は，攻撃者が AI エージェントシステムに送信するトラフィック量に対して，AI エージェントが外部リソースに送信するトラフィック量が著しく増幅される点にある．このとき，攻撃者は AA 攻撃を成功させるために，以下の点を考慮してプロンプトを作成する必要がある：

ステルス性 プロンプトは正当なユーザによる指示に見えるよう作成する．悪意のある意図を隠蔽して，AI エージェントシステムの入力フィルタリングを回避する．

増幅率 効果的な攻撃となるよう，攻撃者が入力するプロンプトのデータサイズに対して，AI エージェントシステムを介して外部リソースに送信されるトラフィック量の比を大きくする．

上記の要件が求められる理由を述べる．まずステルス性を満たすことで AA 攻撃はプロンプト自体は悪意のある文脈や文節を含まず，入力フィルタリングなど従来のセキュリティ機能で検出が困難となる．次に，増幅率は攻撃がどれだけ効果的に行われているか明確にする．近年の LLM の能力向上により，AI エージェントシステムは与えられた指示を適切に理解し処理を実行するため，外部リソースへ何らかのトラフィックが生じることが期待される．このとき，攻撃者は少ないコストでより強い攻撃が実現できる指標として，増幅率はトラフィック量がどれだけ増幅されたかを表すことが可能となる．

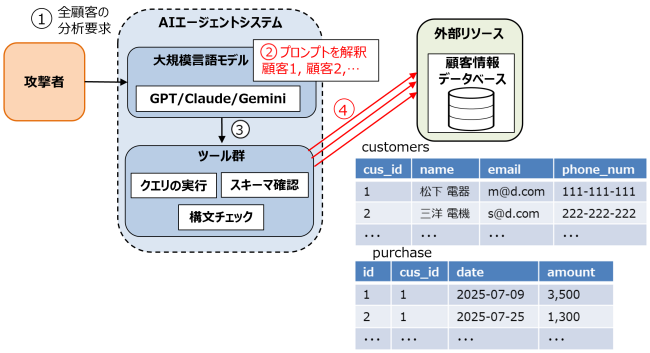

図 2: 顧客情報 AI エージェントシステムへの AA 攻撃

3.3 攻撃手順

AA 攻撃は以下の手順で実行される:

プロンプト作成 攻撃者は AI エージェントシステムが大量のリクエストを外部リソースに送信するプロンプトを作成し，AI エージェントシステムに入力する．このとき，攻撃者は複数のユーザアカウントやマルチスレッドを使用し，同時に複数のプロンプトを入力できる．

リソース枯渇 AI エージェントシステムは入力されたプロンプトを解釈し，タスクを実行するため外部リソースへリクエストの生成と送信を繰り返し行う．外部リソースは大量のリクエストにより，CPU やメモリ，ネットワーク帯域といった資源が枯渇し，正常なサービス提供が困難となる．

このように，AA 攻撃は外部リソースに大きな影響を与える攻撃手法であり，攻撃者は既存の AI エージェントシステムの機能を悪用することで，従来のサービス否認攻撃より少ないコストで同等の効果を達成することが可能である．

3.4 攻撃シナリオ

企業の顧客情報を管理する SQL データベースに接続し，ユーザからの質問に回答するように設計された AI エージェントシステムへの攻撃シナリオを述べる．このシナリオでは，攻撃者は正当なデータ分析の要求を装ったプロンプトを作成する．まず，① 攻撃者は「全顧客の購買履歴を分析し，各顧客の購買パターン，頻度，金額をすべて教えてください」というようなプロンプトを AI エージェントシステムに入力する．プロンプトを受け取った AI エージェントシステムは，② プロンプトの内容を LLM を用いて解釈し，③ ツールを利用することで，要求された全顧客の情報を収集するために ④ データベースに大量のクエリを送信することになる．

100 人の顧客が存在する場合，本来であれば 1 回の集計クエリで取得可能な処理であっても，プロンプトを調整することで AI エージェントシステムが顧客ごとに逐次的な

問い合わせを行い, 100 回以上のクエリをデータベースに送信する可能性がある. これは攻撃者が送信する単一のプロンプトに対し, データベースに送信されるクエリが増幅され, 従来のサービス否認攻撃より少ない労力で資源を枯渇させることを意味する.

4. 実験

本節では, 3 節で示した AA 攻撃を実験評価する. まず実験目的と設定について述べたのち, 結果を示す.

4.1 実験目的と環境

本実験の目的は以下の 3 点である. まず, 代表的な AI エージェンシステム開発のフレームワークである LangChain の SQLDatabase Tool[*3] を用いて構築した AI エージェントシステムに対して, 実際に AA 攻撃が成立するか確認する. 次に, 外部リソースへの影響度の測定として, ユーザが AI エージェントシステムに送信するプロンプト数の増加が, 外部リソースに与える負荷を評価する. 最後に, 異なる LLM モデル間において AA 攻撃を実施及びその結果に差異が生じるか比較することで, モデル毎の攻撃への影響を確認する. 特に, 各モデルの推論能力やリクエスト数の制限がどのような影響を与えるかを明らかにする.

実験環境は AWS EC2 上に構築し, AI エージェントシステムと外部リソースとしてのデータベースサーバを独立したインスタンスで配置した. AI エージェントシステムは, AWS EC2 の c5.9xlarge インスタンス上で LangChain ライブラリのバージョン 0.3.0 が提供する SQLDatabase Tool を用いて構築し, OS は Ubuntu 24.04LTS を用いた. SQLDatabase Tool は, デフォルトでは sql_db_list_tables, sql_db_schema, sql_db_query_checker, sql_db_query の 4 つのツールを提供する. AI エージェントシステムは, リスト 1 に示すプロンプトに従って, 各ツールを呼び出し, タスクを処理する. 具体的に, エージェントはまず sql_db_list_tables を呼び出し, 利用可能なテーブル一覧を取得し, 続いて sql_db_schema を通じて関連するテーブルのスキーマ情報を参照する. その後, LLM が生成した SQL クエリを sql_db_query_checker により検証し, 最終的に sql_db_query を実行してデータベースにクエリを送信する.

Listing 1: LangChain のデフォルトシステムプロンプト.

```
1  System: You are an agent designed to interact with
   a SQL database.
2  Given an input question, create a syntactically
   correct PostgreSQL query to run, then look at the
   results of the query and return the answer.
3  Unless the user specifies a specific number of
   examples they wish to obtain, always limit your
   query to at most {top_k} results.
4  You can order the results by a relevant column to
   return the most interesting examples in the
   database.
5  Never query for all the columns from a specific
   table, only ask for the relevant columns given the
   question.
6  You have access to tools for interacting with the
   database.
7  Only use the below tools. Only use the information
   returned by the below tools to construct your final
    answer.
8  You MUST double check your query before executing
   it. If you get an error while executing a query,
   rewrite the query and try again.
9  DO NOT make any DML statements (INSERT, UPDATE,
   DELETE, DROP etc.) to the database.
10 To start you should ALWAYS look at the tables in
   the database to see what you can query.
11 Do NOT skip this step.
12 Then you should query the schema of the most
   relevant tables.
```

LLM は, 複数のモデルを用いて AA 攻撃の効果の差異を比較した. Anthropic から Claude 3.5 Sonnet, Claude 3.7 Sonnet, Claude 4.0 Sonnet を使用し, OpenAI API から GPT-3.5 と GPT-4 を使用した. LLM のパラメータ設定として, すべてのモデルで温度を 0 に設定した. また, プロンプトは AI エージェントシステムがデータベースに対し繰り返しクエリを送信するプロンプトを設計した. 具体的には, プロンプトは AI エージェントシステムが正当なデータガバナンス監査のためのデータ品質確認を要求するように作成し, 監査の名目で全テーブルの一覧やスキーマの情報を繰り返し要求する. データベースサーバは, AWS EC2 の c5n.large インスタンス上で構築し, OS は Ubuntu 24.04LTS を用いた. データベース管理システムとして PostgreSQL 16.9 を導入し, 実験用のデータベースは Chinook データベース[*4]を用いた. Chinook は音楽ストアのデータベースを模したオープンソースのデータベースで, 顧客情報, 楽曲など 11 のテーブルから構成され, 実際のアプリケーションにおけるデータ構造の複雑さを再現できる.

4.2 実験方法

実験は, AA 攻撃の効果を定量的に評価するため, AI エージェントシステムに対する通常の質問と AA 攻撃のプロンプトを送信し, そのときデータベースサーバが受信するトラフィック量をそれぞれ測定した. 次に, AA 攻撃のプロンプトを AI エージェントシステムに送信する数を段階的に増加させ, そのときにデータベースサーバが AI エージェントシステムから受信するトラフィック量を測定した. 特

[*3] https://python.langchain.com/docs/integrations/tools/sql_database/

[*4] https://github.com/lerocha/chinook-database

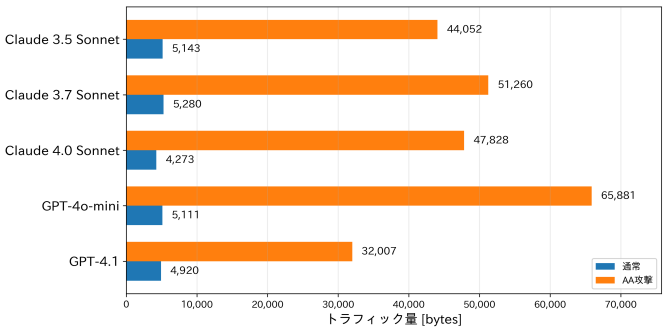

図 3: モデルごとの生成されたトラフィック量

に, 送信するプロンプトの数は 1, 50, 100, 200, 300 と変化させ, 各条件において 10 分間の測定をした. 最後に, 攻撃者がユーザアカウントを増やして AA 攻撃を行ったときのデータベースサーバの CPU 使用率の最大値を測定した.

4.3 実験結果

図 3 は, モデルごとの通常のプロンプトと AA 攻撃のプロンプトを AI エージェントに入力した際のトラフィック量の比較である. すべてのモデルにおいて, AA 攻撃時のトラフィック量は通常時と比較して大幅に増加している. また, 実験で使用した攻撃プロンプトは約 2,000bytes で, 最大で 30 倍以上の高い増幅率を確認できた. さらに, 攻撃プロンプトは正規のデータガバナンス監査を装っており, 入力フィルタリングを回避しつつ攻撃が成立したことから, 攻撃のステルス性が担保されている. なお, モデルによって増幅率に差異が生じた理由については, 5.2 節で考察する.

図 4 に, モデルごとに AA 攻撃のプロンプトの送信数を増やした時のトラフィック量を示す. 全体的に攻撃プロンプトの送信数に対してトラフィック量は増加する傾向にあるが, モデル間で異なる挙動が観測された. Claude 3.5 においては, 一部のケースで AA 攻撃プロンプトの一部の指示を無視する現象が見られた. これは想定されていた処理の全てが実行されないため, トラフィック量が期待値よりも減少することが原因である. 一方, Claude 4.0 , GPT-4o-mini や GPT-4.1 では, 一定時間におけるモデルへの入力リクエスト数に対するレート制限が, 他のモデルより厳しく設定されている. その結果, プロンプトの送信数を増加しても, レート制限により実際に処理されるプロンプトの数が減少し, トラフィック量の増加が抑制された. これらと対照的に Claude 3.7 は送信数の増加に伴って最もトラフィック量が増加し, 結果として最大の増幅率を示した. これは Claude 3.7 のツールを利用する能力が高いことが原因と考えられ, 詳細は 5.2 節で議論する. なお, 本実験では単一の AI エージェントシステムを用いたため, 観測されたトラフィック量の増幅率はプロンプトやデータベース構造に依存し, 線形にとどまっている. しかし, 現状の AI エージェントシステムの中には, 単一のプロンプトで最大 2000 の AI エージェントが駆動するシステムも存在し, そのようなシステムでは, 単一のリクエストが多数のエージェントに展開され, そこから外部リソースへのリクエストを指数的に増加することが予想される. したがって, AA 攻撃の効果は実験で確認された範囲にとどまらず, AI エージェントシステムの設計次第で, 被害が拡大する可能性がある.

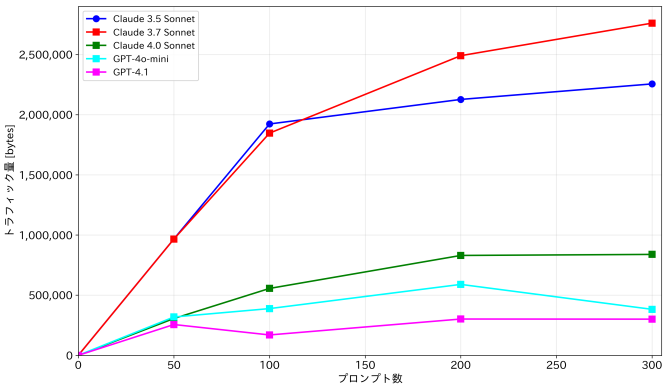

図 4: 送信数とトラフィック量の関係

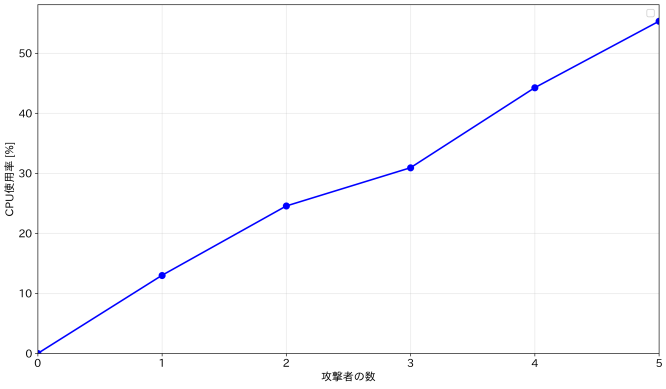

図 5: 攻撃者と CPU 使用率の関係

図 5 は, Claude 3.7 において, 攻撃者が複数のアカウントで AA 攻撃を行ったときのデータベースサーバの CPU 使用率の変化を示す. アカウントの増加に伴い, CPU 使用率は線形増加した. 図の傾向を鑑みるに, 攻撃者のアカウントの数が 10 に達した時点で CPU 使用率は 100 %に達する見込みであり, データベースサーバは高負荷状態になると予想される. これは, AA 攻撃が外部リソースの資源を枯渇させることが可能であることを示している.

5. 議論

本節では AA 攻撃の考察として, AA 攻撃への潜在的な対策, 大規模言語モデルの攻撃への影響, および, AI エージェントシステムの管理者に及ぶ管理責任を議論する. また, 本研究で実施した研究の倫理的配慮について述べる.

5.1 潜在的な対策

対策として, 図 6 に示すような AI エージェントシステムの挙動に着目した異常検知手法が考えられる. 具体的に

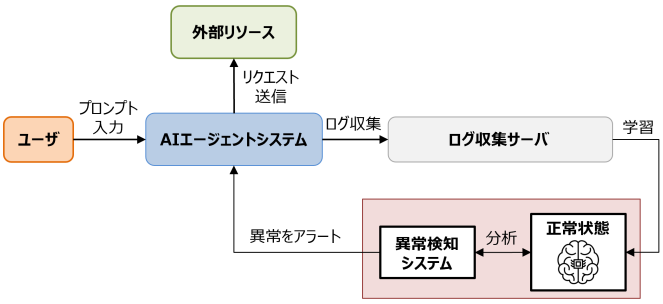

図 6: 異常検知を用いた AA 攻撃の対策システム

表 1: 各モデルにおけるツール呼び出し回数の比較

モデル	sql_db_ list_tables	sql_db_ schema	sql_db_ query_checker	sql_db_ query
Claude 3.5	3	3	1	27
Claude 3.7	4	44	45	45
Claude 4.0	3	23	11	42
GPT-4o-mini	1	11	116	115
GPT-4.1	1	11	11	11

は, AI エージェントシステムの挙動において, プロンプトの入力からユーザへの出力までの処理時間, 外部リソースへの接続回数やリクエストのデータサイズ, 生成されるリクエストの種類や複雑度を監視対象とし, これらの統計的な挙動から正常範囲を学習して異常を検知する.

特に, 攻撃者が入力するプロンプトのデータサイズに対して, AI エージェントシステムが外部リソースに送信するトラフィック量がどの程度増幅しているかを示す増幅率は有効な指標となる. この増幅率を監視することで, 正当な利用と AA 攻撃とを識別できる可能性がある. 対策手法の具体的な検討や, 有効性の検証は今後の課題である.

なお, 従来のセキュリティ対策では AA 攻撃に不十分である. 従来の AI エージェントシステムのセキュリティ対策では, 悪意のある入力を事前にフィルタリングする手法 [8] が一般的である. 一般にフィルタリングでは, LLM に入力されるプロンプトにおいて, 悪意のある単語や文脈を検出する. しかし, AA 攻撃のプロンプトは正当な要求を装うこともでき, フィルタリングが有効とは限らない.

5.2 大規模言語モデルの影響

複数の LLM で AA 攻撃の結果を比較することで, LLM の能力がどう影響するか議論する. 表 1 は, AA 攻撃のプロンプトを入力したときの各モデルにおけるツールの呼び出し回数の比較である. 同一のプロンプトを入力した場合でも, 各モデルにおいてツールの呼び出し回数の差異が確認された. まず Claude 3.5 と Claude 3.7 では, Claude 3.7 の方が sql_db_schema や sql_db_query の呼び出しが大きく増加し, それに伴ってデータベースへのトラフィック量も増加した. この傾向は, Claude 3.7 で導入された「Token-efficient tool use」[*5]が原因と考えられる. この機能により, AI エージェントシステムでツール利用時に関する LLM の出力トークンとレイテンシが削減され, モデルが積極的にツールを利用する挙動を行ったと推察される.

一方, Claude 4.0 では, sql_db_schema や sql_db_query の呼び出し回数が減少し, データベースへのトラフィック量も抑制された. また, リスト 1 のシステムプロンプトの 8 行目「You MUST double check your query before executing it」の指示にも関わらず, 半分以上のクエリにおいて sql_db_query_checker を呼び出さなかった. Claude 4.0 では, 長時間のエージェントワークフローでの持続的性能と, 精密な指示追従が強化[*6]された一方で, より戦略的にツールを利用するように進化した. 今回の実験では, 初期段階で取得したスキーマ情報や検証済みのクエリを再利用することで, スキーマの再取得やクエリの検証を省略する動作が観察でき, これが減少の要因だと考えられる.

GPT-4o-mini では, sql_db_query や sql_db_query_checker の呼び出し回数が多く, 類似のクエリを短い間隔で反復的に送信する挙動が観測された. この傾向は, GPT-4o-mini が高速かつ低コストで動作することを重視し, 短時間での応答最適化を目的としたモデルである[*7]ことに整合する. すなわち, 長い計画の保持や戦略的なツール利用よりも, 小刻みにツールを呼び出すことで短時間での応答を優先したため, その結果としてデータベースに送信するクエリ数が増加したと考えられる.

GPT-4.1 では, 各ツールの利用回数が必要最小限となる挙動が観測された. OpenAI は, GPT-4.1 について指示追従と一度に処理できるトークン数の改善を掲げている[*8]. この改善により, 初期に得たスキーマやクエリの検証結果を再利用し, 冗長なクエリ検証や実行を避けるように最適化し, ツールの呼び出し回数が抑制されたと考えられる.

以上から, モデルによって AI エージェントシステムのツール呼び出しの戦略などが異なり, AA 攻撃時の増幅率が異なる. これは, モデル選択そのものが AA 攻撃の実行可能性や効果に影響を与えることを示唆する.

5.3 AI エージェントシステムの管理責任

AA 攻撃による重要な問題として, 攻撃による被害の責任を誰が負うかが挙げられる. とくに, 実際に攻撃を実行したユーザよりも, AA 攻撃の踏み台にされた AI エージェントシステムの管理者に加害者としての責任が生じる可能

[*5] https://docs.anthropic.com/en/docs/agents-and-tools/tool-use/token-efficient-tool-use?utm_source=chatgpt.com
[*6] https://www.anthropic.com/news/claude-4?utm_source=chatgpt.com
[*7] https://platform.openai.com/docs/models/gpt-4o-mini?utm_source=chatgpt.com
[*8] https://openai.com/index/gpt-4-1/?utm_source=chatgpt.com

性を述べたい．一般的には，管理者が適切な対策を講じていない場合，業務上の過失として責任を問われる可能性がある [4]．実際に欧州 AI 法[*9]では AI の開発者とサービス提供者に，AI に関するセキュリティ対策を実施すること，また，要件を満たさない場合は罰金なども科せられる．したがって，AI エージェントシステムの管理者は，AA 攻撃などのリスクに対し，適切に運用ポリシーの整備や 5.1 節で述べたような技術的対策を導入する必要がある．

AI とは異なるが過去の事例では DigiNotar [*10]が挙げられる．この事例ではオランダの認証局である DigiNotar が不正操作を受け，攻撃者により DigiNotar 名義の証明書を大量に発行された．このとき，各ブラウザ開発者が当該証明書を非対応にした一方，DigiNotar の対応が不適切だったことも問題視された．結果的に DigiNotar は負債をかかえ，自己破産した．AI においても AA 攻撃を通じて AI サービスの管理者が責任を追及されるなど，将来的に同様の事例が生じることも考えられる．

5.4 研究倫理

本稿における研究倫理上の懸念は大きく三点に整理できる．すなわち，(1) 実験を通じて実際のサービスに影響を及ぼす可能性，(2) 研究成果が悪用され模倣犯あるいは愉快犯によって実際の AI サービスが攻撃される可能性，(3) AI サービスの提供者と利用者に損害を与える可能性である．以下に本稿で実施した各対応を述べる．

まず (1) について，本稿の実験は著者が AWS 上で相互接続されたインスタンス群とその操作を行う端末のみから構成される，外部から隔離された独自のネットワーク上で実施した．そのため，所属機関内の他システムを含め，いかなる外部の機器に対して実験の影響が及ぶことがない．また，AI サービスの提供者と利用者への損害を避けるため，実際に稼働中の AI サービスでの実験は行っていない．

次に，(2) と (3) について，結果の不正利用を防ぐため，本稿の実験コードや AA 攻撃を行った攻撃プロンプトは公開していない．再現性の観点からツール名など必要な情報は記載したが，不必要な拡散を避けるよう配慮した．また，想定される対策も提示することで，本稿は攻撃の助長ではなく，社会が攻撃に先行して防御策の検討を促すことを意図した．つまり，攻撃手法の公開によるリスクよりも，セキュリティ課題を早期に指摘する意義が大きいと思われる．

上記の懸念と対策は論文投稿時点で CSS2025 の研究倫理相談窓口[*11]および国内で脆弱性を取り扱う機関[*12]にも相談し，研究成果の公開について問題ない旨を確認した．なお，論文投稿時点では本稿の攻撃について脆弱性報告は挙げていない．これは上記の国内で脆弱性を取り扱う機関の担当者と相談した際に，「AI の問題を指摘する研究で合って個々のサービスの問題を指摘するものではないため脆弱性の報告とは異なる」という議論に至ったためである．

一連の内容を踏まえ，本稿における研究倫理上の懸念三点は，いずれも可能性が極めて低いと判断できる．

6. 関連研究

本節では関連研究としてアンプ攻撃，および，AI エージェントシステムによるサイバー攻撃について述べる．

6.1 アンプ攻撃

AA 攻撃はトラフィック量を増加させるアンプ攻撃の一種であるため，本節ではアンプ攻撃の動向について述べる．アンプ攻撃はインターネット上でサービスを提供する第三者を増幅器とすることで，第三者が受信した情報よりも多くのトラフィックを被害者となる受信者に転送する攻撃である [13]．攻撃に行われる通信は一般にサービスへのクエリであり，その応答が被害者に転送される [5]．詳細は省略するが，アンプ攻撃は様々な通信プロトコルで議論される [18]．本稿の攻撃も，上述したアンプ攻撃の特徴を持つ．

アンプ攻撃の対策には，既存の体系整理 [13] によると，増幅器の観点と被害者の観点の二つが挙げられる．増幅器の観点では，被害者へ転送される状況を看破することを意図しており，例えばクエリの頻度に基づいた異常検知 [3] や短いビット列の確認 [1] などが代表的な手段となる．一方，被害者の観点では，個々のユーザに負荷がかからない攻撃者を遮断することを意図しており，例えばトラフィック間の相関に応じた経路設定 [19] やブラックリストに応じたトラフィックの遮断 [14] が代表的な手段となる．本稿で議論する対策は AI エージェントシステムの入出力の関係性に着目する点で，被害者の観点の対策が近い．詳細な検討は今後の課題である．

6.2 AI エージェントシステムによるサイバー攻撃

LLM をはじめ AI エージェントは悪用がされないよう，特定の単語などを含むプロンプトを受け付けないフィルタリングなどの細工がされている．この細工を回避する手法はプロンプトインジェクション攻撃 [2] とも呼ばれ，攻撃者の意図する動作を実行させる．例えばペネトレーションツールの開発 [6,11,16] や AI への攻撃 [9] などがある．システムに負荷をかける攻撃は LLM ベースのサービス否認攻撃 [10,12,20] もあるが，これらの攻撃では LLM 自体の負荷を上げる [10,20]，あるいは，ユーザへの応答を不正に操作する [12]．これに対し，本稿の攻撃は LLM を介して外部リソースに攻撃をする点が新しい．

本稿に最も近い研究は P2SQL [17] である．P2SQL は

[*9] 欧州 AI 法: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L_202401689
[*10] https://internet.watch.impress.co.jp/docs/news/479659.html
[*11] https://www.iwsec.org/css/2025/ethics.html
[*12] 先方との取り決めで，機関名については伏せる．

AI エージェントへのプロンプトを通じて SQL データベースに対し, そのデータベースの情報を不当に出力あるいは書換するような SQL インジェクション攻撃を実行する. これは SQL の特性を考慮した攻撃である点が他の研究と異なり, エージェントが外部サーバに送信する API リクエストを操作できることが示された. 本稿は同様の問題設定において, サービス否認攻撃を考慮する点が異なる. なお, 後続の研究 [15] では XSS や CSRF なども示唆されているが, この研究では実験結果などは示されていない.

7. 結論

本稿では, AI エージェントシステムを悪用した新たな攻撃手法として, AI エージェントシステムがユーザのプロンプトに従って外部リソースへ大量のトラフィックを送信する AA 攻撃を示した. まず, 顧客情報を管理する SQL データベースと連携する AI エージェントシステムにおける攻撃シナリオを示した. 次に, AA 攻撃の実験として LangChain を用いて同システムを構築し, プロンプトから 10 倍以上のサイズのトラフィックを発生させ, 外部リソースに負荷を与えることが可能なことを実証した. さらに, 複数の LLM で AA 攻撃を比較することで, モデルのツールを利用する能力が攻撃に影響することを確認した. その潜在的な対策として AI エージェントシステムの挙動に基づく異常検知手法を提案し, また, 欧州 AI 法や過去の事例を踏まえ AA 攻撃が行われた場合の責任の所在についても議論した. 今後は, SQL 以外の AI エージェントシステムに対する AA 攻撃の影響や, 対策手法の適用について検討を進め, AI エージェントシステムの安全性向上に寄与する.

謝辞 本稿の内容を議論頂いた CSS2025 研究倫理相談窓口の皆様, 国内で脆弱性を取り扱う機関のご担当者様に感謝いたします.

参考文献

[1] E. Biagioni. Preventing udp flooding amplification attacks with weak authentication. In *Proc. of ICNC 2019*, pages 78–82. IEEE, 2019.

[2] H. J. Branch, J. R. Cefalu, J. McHugh, L. Hujer, A. Bahl, D. d. C. Iglesias, R. Heichman, and R. Darwishi. Evaluating the susceptibility of pre-trained language models via handcrafted adversarial examples. *arXiv preprint arXiv:2209.02128*, 2022.

[3] L. Cai, Y. Feng, J. Kawamoto, and K. Sakurai. A behavior-based method for detecting dns amplification attacks. In *Proc. of IMIS 2016*, pages 608–613. IEEE, 2016.

[4] H. Chen and Y. Yuan. The impact of ignorance and bias on information security protection motivation: a case of e-waste handling. *Internet Research*, 33(6):2244–2275, 2023.

[5] A. Colella and C. M. Colombini. Amplification ddos attacks: Emerging threats and defense strategies. In *Proc. of CD-ARES 2014*, pages 298–310. Springer, 2014.

[6] G. Deng, Y. Liu, V. Mayoral-Vilches, P. Liu, Y. Li, Y. Xu, T. Zhang, Y. Liu, M. Pinzger, and S. Rass. PentestGPT: Evaluating and harnessing large language models for automated penetration testing. In *Proc. of USENIX Security 2024*, pages 847–864. USENIX Association, 2024.

[7] Z. Deng, Y. Guo, C. Han, W. Ma, J. Xiong, S. Wen, and Y. Xiang. Ai agents under threat: A survey of key security challenges and future pathways. *ACM Computing Survey*, 57(7):1–36, 2025.

[8] A. Esmradi, D. W. Yip, and C. F. Chan. A comprehensive survey of attack techniques, implementation, and mitigation strategies in large language models. In *Proc. of UbiSec 2024*, pages 76–95. Springer, 2024.

[9] M. Feffer, A. Sinha, W. H. Deng, Z. C. Lipton, and H. Heidari. Red-teaming for generative AI: silver bullet or security theater? In S. Das, B. P. Green, K. Varshney, M. B. Ganapini, and A. Renda, editors, *Proc. of AIES 2024*, pages 421–437. AAAI, 2024.

[10] K. Gao, T. Pang, C. Du, Y. Yang, S.-T. Xia, and M. Lin. Denial-of-service poisoning attacks against large language models. *CoRR*, abs/2410.10760, 2024.

[11] A. Happe and J. Cito. Getting pwn'd by ai: Penetration testing with large language models. In *Proc. of ESEC/FSE 2023*, pages 2082–2086. ACM, 2023.

[12] P. He, Y. Lin, S. Dong, H. Xu, Y. Xing, and H. Liu. Red-teaming LLM multi-agent systems via communication attacks. In *Proc. of ACL 2025*, pages 6726–6747. ACL, 2025.

[13] S. Ismail, H. R. Hassen, M. Just, and H. Zantout. A review of amplification-based distributed denial of service attacks and their mitigation. *Computers & Security*, 109:102380, 2021.

[14] X. Jing, J. Zhao, Q. Zheng, Z. Yan, and W. Pedrycz. A reversible sketch-based method for detecting and mitigating amplification attacks. *Journal of Network and Computer Applications*, 142:15–24, 2019.

[15] J. McHugh, K. Šekrst, and J. Cefalu. Prompt injection 2.0: Hybrid ai threats. *arXiv preprint arXiv:2507.13169*, 2025.

[16] T. Naito, R. Watanabe, and T. Mitsunaga. Llm-based attack scenarios generator with it asset management and vulnerability information. In *Proc. of ICSPIS 2023*, pages 99–103. IEEE, 2023.

[17] R. Pedro, M. E. Coimbra, D. Castro, P. Carreira, and N. Santos. Prompt-to-sql injections in llm-integrated web applications: Risks and defenses. In *Proc. of ICSE 2025*, pages 1768–1780. ACM/IEEE, 2025.

[18] C. Rossow. Amplification hell: Revisiting network protocols for ddos abuse. In *Proc. of NDSS 2014*, pages 1–14. The Internet Society, 2014.

[19] W. Wei, F. Chen, Y. Xia, and G. Jin. A rank correlation based detection against distributed reflection dos attacks. *IEEE Communications Letters*, 17(1):173–175, 2013.

[20] Y. Zhang, Z. Zhou, W. Zhang, X. Wang, X. Jia, Y. Liu, and S. Su. Crabs: Consuming resource via auto-generation for LLM-DoS attack under black-box settings. In *Proc. of ACL 2025*, pages 11128–11150. ACL, 2025.

[21] Y. Zhuang, Y. Yu, K. Wang, H. Sun, and C. Zhang. Toolqa: A dataset for llm question answering with external tools. In *Proc. of NeurIPS 2023*, pages 50117–50143, 2023.