ダークネットマーケットの第三者サービス依存性調査：LLMによる検知とシグネチャ生成

内野 彰紀[1,a)]　インミンパパ[2]　吉岡 克成[3]

概要：
ダークネットマーケットは違法薬物，偽造品，マルウェア，金融詐欺ツールなどの取引に利用されており，その匿名性の高さから，従来の運営者特定やインフラ追跡に基づくテイクダウン手法には限界があった．一方で，これらのマーケットは e-commerce（EC）サイトとして機能するために，仮想通貨決済や Web フォントといった第三者サービスに依存するという脆弱性を有する．この依存関係の特定は，サービスプロバイダーへの報告を通じてマーケットの機能停止を促す新たな対策に繋がる．本研究では，大規模言語モデル（LLM）を活用し，既知のダークネットマーケットと第三者サービスの依存関係から第三者サービスの検出シグネチャを自動生成し，未知のマーケットにおける同種の依存性を高精度に検知する手法を提案する．本研究では，LLM が第三者サービスの API 実装方法や仕様に関する資料と，依存が確認されたマーケットおよび依存がないことが確認されたマーケットの HTML および HAR ファイルを分析し，サービス特有の API 呼び出しパターンを捉えた正規表現形式のシグネチャを生成する．提案手法の有効性を検証するため，55 個のダークネットマーケットの観測データと 5 種類の第三者サービス（WooCommerce，jQuery，Google Fonts，Blockonomics，MyCryptoCheckout）から成るデータセットで評価した結果，生成されたシグネチャは未知のデータに対し，平均で 90%を超える F1 スコアを達成した．特に決済サービスである MyCryptoCheckout では，全ての試行において 100%の精度で依存性を特定可能であった．本研究は，LLM を用いることで，従来は手動分析に依存していたシグネチャ生成を自動化し，ダークネットマーケットが持つ第三者サービスへの依存性という脆弱性を効率的に特定することができる．

キーワード：第三者サービス依存性, シグネチャ生成, LLM 活用

Detection of Third-Party Dependencies in Darknet Markets via LLM-Based Signature Generation

Akinori Uchino[1,a)]　Yin Minn Pa Pa[2]　Katsunari Yoshioka[3]

Abstract:
Darknet markets are used for trading illicit goods such as drugs, counterfeit items, malware, and financial fraud tools. Due to their high level of anonymity, traditional takedown approaches that rely on identifying operators or tracking infrastructure have significant limitations. On the other hand, because these markets function as e-commerce (EC) sites, they possess a vulnerability in their reliance on third-party services such as cryptocurrency payment processors and web fonts. Identifying these dependencies can enable new countermeasures, such as prompting service providers to suspend market functionality through responsible reporting.In this study, we leverage large language models (LLMs) to automatically generate detection signatures for third-party services based on known darknet markets and their dependencies. Our method analyzes both technical documentation on API implementations and specifications, as well as HTML and HAR files collected from markets where dependencies have been observed, in order to generate regular-expression-based signatures that capture service-specific API call patterns.To validate the effectiveness of the proposed approach, we evaluated it using a dataset consisting of 55 darknet markets and five third-party services (WooCommerce, jQuery, Google Fonts, Blockonomics, and MyCryptoCheckout). The results show that the generated signatures achieved an average F1-score of over 90% on unseen data. Notably, for the payment service MyCryptoCheckout, all trials achieved 100% accuracy in identifying dependencies.This research demonstrates that by employing LLMs, the process of generating detection signatures—previously dependent on manual analysis—can be automated, enabling the efficient identification of vulnerabilities arising from the reliance of darknet markets on third-party services.

Keywords: Third-Party Service Dependencies, Signature Generation, Application of LLM

1. はじめに

ダークネットマーケットは，Tor ネットワーク等の匿名化技術を基盤とするダークウェブ上で運営される違法取引プラットフォームであり，違法薬物，偽造品，マルウェア，金融詐欺ツールなどが大規模に取引されている [1]．これらのマーケット運営者・利用者は高い匿名性を追求している一方で，一般的な e-commerce (EC) サイトと同様に，決済処理，顧客サポート，セキュリティ機能などの実装において第三者サービスに依存している点に特徴がある．この依存関係は，第三者サービスプロバイダとの協力により，ダークネットマーケットの機能を停止させる新たな対策の可能性を示唆している．

ダークネットマーケットをテイクダウンするための既存手法では，主に運営者を特定するか，利用しているインフラサービスを特定することに焦点を当てていた [2][3][4][5]．しかし，マーケットの高い匿名性により，これらの特定は困難であり，効果的な対策の実施が制限されてきた．その一方で，これらのマーケットは，EC サイトとしての基本機能を実現するため，決済サービスや Web フォントサービスなどの第三者サービスへの依存する傾向がある．この依存関係は，ダークネットマーケットの運営における重要な脆弱性として捉えることができる．すなわち，これらの第三者サービスへの依存状態を特定し，サービスプロバイダーに悪用の事実を報告することでダークネットマーケットの不正な活動を停止させるなどの対策につながる可能性がある．しかしながら，第三者サービスへの依存関係を正確に検証するためには，各サービスの技術仕様の理解と，マーケットサイトにおける API 呼び出しパターンの詳細な分析が必要となる．現状では，このような検証作業を効率的に実施する手法は確立されていない．

本研究では，大規模言語モデル（LLM）を活用した第三者サービス依存性の自動検出手法を提案する．提案手法としては 3 つのステップから構成される．第一に，特定したい第三者サービスの API 実装方法や仕様に関する資料を作成する．第二に，LLM を利用して特定したい第三者依存サービスのシグネチャ生成をマーケットのデータや収集したサービスに関する資料を元に行う．第三に，作成したシグネチャを生成に用いたマーケットデータで検証し，正しくシグネチャ生成が行える事を確認する．

[1] 横浜国立大学大学院環境情報学府
Graduate School of Environment and Information Sciences, Yokohama National University
[2] 横浜国立大学大学院先端科学高等研究院
Institute of Advanced Sciences, Yokohama National University
[3] 横浜国立大学大学院環境情報研究院/先端科学高等研究院
Faculty of Environment and Information Sciences, Yokohama National University / Institute of Advanced Sciences, Yokohama National University
[a)] uchino-akinori-pn@ynu.jp

評価では，提案手法の方法で作成したシグネチャを実際のマーケットデータに対して適用し，シグネチャの性能検証を行った．具体的には，ダークネットマーケットで実際に利用される第三者サービスである WooCommerce[6], jQuery[7], Google Fonts[8], Blockonomics[9], MyCryptoCheckout[10] を特定するためのシグネチャ生成を行い，評価を行った．なお，本研究では検知対象の第三者サービスに依存する 1 つのマーケットデータから作成したシグネチャ，2 つのマーケットデータから作成したシグネチャ，3 つのマーケットデータから作成したシグネチャをそれぞれ最大 10 個ずつ作成した．なおこれらのマーケットデータに加えて検知対象の第三者サービスに依存しないマーケットデータも入力として用いた．評価方法としては，依存するマーケットを正しく検知し、非依存のマーケットを検出しないことを検証した．この検証は、Accuracy, Precision, Recall, F1 Score の 4 つの指標を用いて行った．未知のマーケットデータで平均で 90%を超える F1 スコアを達成した．

本研究の主要な貢献は以下の通りである：

- ダークネットマーケットにおける既知の第三者サービス依存関係を検出するシグネチャの自動生成手法を提案した．
- 5 種類の第三者サービス依存性を手動で分析し，正解データセットを作成し，シグネチャの生成を行った．
- 正解データセットを用いて提案手法を評価し，作成されたシグネチャが未知のダークネットマーケットにおける既知の第三者サービス依存性を検知するために有効であることを示した．

2. 関連研究

ダークネットマーケット (DNM) の規模推定・生態系分析: DNM の取引規模，参加者行動，マーケット運営の継続性に関する研究は継続的に報告されている．Christin による初期の定量分析はシルクロードを対象に売上・商品カテゴリ・出品者行動を長期観測し，DNM が持続的な経済圏であることを示した [11]．Soska と Christin は複数 DNM の消長を継続観測し，テイクダウン・Exit scam・競合関係が市場ダイナミクスに与える影響を示した [12]．さらに，Kruithof らは DNM の規模をマクロ統計的に推定し，違法薬物市場のオンライン移行が実空間の供給網に与える影響を考察した [13]．これらは本研究の「対象領域の重要性（規模・持続性）」「継続観測における変動性」を裏付ける基盤文献である．

DNM のテイクダウンとインフラ・依存関係の追跡: DNM を止める技術的手段としては，運営者同定，ホスティング／CDN／DDoS 保護等のインフラ露出，サプライチェーン依存の活用がある．Biryukov らは Tor Hidden Service のトラフィック相関・指紋化可能性を示し [2]，Kwon らはガード中毒（Guard relay manipulation）により隠しサー

ビスの Deanonymization リスクを実証した [3]．Chaabane らは Tor 上の Web サービスの測定によりアップストリーム依存（外部資源参照やゲートウェイ）を検出した [4]．Edman と Syverson はディレイ測定とトラフィックパターンでの相関手法を整理し [5]．本研究は，アプリケーション層の第三者サービス依存（決済，ライブラリ，フォント取得等）に着目する点で補完的である．

Web における第三者サービス依存とサプライチェーンリスク: Web 追跡・広告エコシステムの計測研究は，第三者ドメインの広範な浸透と相互依存を実証してきた．Lerner らはトラッキングドメインの巨大グラフを計測し，長い依存連鎖と集中度（少数の大手プロバイダ）を示した．また，[14]．Thomas らは外部スクリプト・広告コードが EC サイトのセキュリティに与える影響（Magecart 型攻撃など）を示した [15]．これらの手法は，大規模クロールで HTML ／ JS ／ネットワークアーティファクトを収集し，正規表現や静的解析，ドメイン分類により依存を同定している．本研究は，LLM でサービス固有パターンからシグネチャを合成し，HAR（HTTP Archive）を含む多様なアーティファクトを横断マッチングする点に新規性がある．

本研究との位置付け: 先行研究は，(i) DNM の継続観測と規模推定，(ii) インフラ層の露出やトラフィック相関による Deanonymization，(iii) 一般 Web での第三者依存とサプライチェーンリスク計測に大別できる．本研究は，これらを統合し，サービス公式文書／実観測データ（HTML ＋ HAR）から LLM で特徴を抽出してシグネチャを自動生成し，DNM 特有の第三者サービス依存（WooCommerce, jQuery，Google Fonts，Blockonomics，MyCryptoCheckout）をスケーラブルに検出する点で新規性を有する．特に，観測容易なアプリケーション層の依存を攻撃面・介入点として定式化し，少数サンプルからの一般化性能を定量評価した点が貢献である．

3. 提案手法

本研究では，ダークネットマーケットの HTML や HAR ファイルといったマーケットデータを対象に，大規模言語モデル（LLM）を用いて特定の第三者サービスへの依存性を識別するためのシグネチャ生成手法を提案する．本章では例として，サービス S を検出するためのシグネチャ生成手法を示す．まず，サービス S に依存しているマーケットデータと依存していないマーケットデータを入力データとして利用する．この入力データに基づくシグネチャ生成手順は，図 1 に示すように 3 つのステップから構成される．

3.1 技術資料の作成

サービス S の API 実装方法や仕様に関する資料 (以後，技術資料) を作成する．このプロセスでは，主にサービスの公式情報源から具体的かつ技術的な詳細情報を効率的に収集することに焦点を当てる．技術資料作成の目的は，API を実際に利用するために不可欠な技術的詳細情報を網羅的に取得することである．抽象的な説明や一般的な概要ではなく，具体的な技術情報に焦点を当てる．収集すべき主要な項目は以下の通りである．

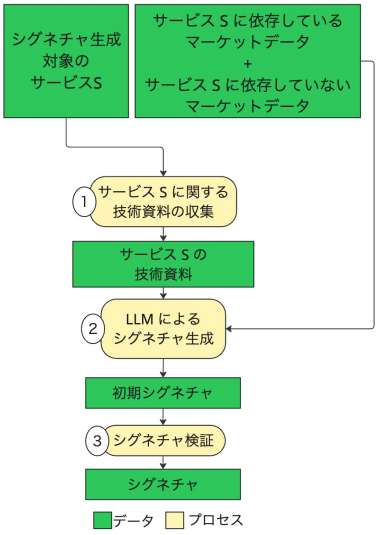

図 1 シグネチャ生成プロセス

- **メイン Web サイト URL**：サービスの公式 Web サイトの URL.
- **公式資料 URL**：サービスの公式資料の URL.
- **API 資料 URL**：API に関する詳細情報が記載された資料の URL(利用可能な場合).
- **認証方式**：API アクセスに必要となる認証方式 (API キー，OAuth，トークンなど).
- **API レスポンス形式**：API がデータを返す際の形式 (JSON，XML など).
- **共通 HTTP ヘッダー**：API リクエストまたはレスポンスで頻繁に使用される HTTP ヘッダー.
- **エラーレスポンスパターン**：エラー発生時に返されるレスポンスの共通パターン.

この技術資料は，後続のステップで LLM がシグネチャを生成するための重要な参照情報として活用される．

3.2 LLM によるシグネチャ生成

本ステップでは，LLM（大規模言語モデル）を活用して，収集したデータから第三者依存サービスを特定するためのシグネチャ（正規表現）を生成する．このプロセスは，マーケットから収集したマーケットデータ (サービス S に依存しているマーケット，依存していないマーケットどちらも利用する) と，3.1 節のステップで作成した技術資料を LLM への入力として用いることで実現される．

LLM に与えられるタスクは，これらの入力データを総合的に分析し，特定のサービスに固有の技術的指標（例：第三者依存サービスの URL API ，特定のヘッダー情報な

ど）を利用してサービスを特定するシグネチャの生成を行うことである．

LLM は，入力された膨大なデータの中から，サービス S に関する技術資料，検知対象のマーケットデータと検知対象でないマーケットデータを元にサービスを識別するための痕跡を多角的に探索する．これらの分析結果に基づき，サービスを確実に検出可能な正規表現パターンが生成される．最終的な出力として，これらのパターンを OR 演算子で組み合わせた初期シグネチャが得られる．

3.3 シグネチャ検証

作成した初期シグネチャを，生成に用いたマーケットデータに適用し，サービス S に依存しているマーケットデータに対しては検知が成功し，非依存のマーケットデータに対しては検知が発生しないことを検証する．この検証を通じて得られたシグネチャを，第三者サービス依存性を特定するためのシグネチャとする．また，意図しない動作が確認された場合には，3.2 節の手順を再度実行し，シグネチャの再生成を行う．

これらの手法を用いる事で，第三者サービスの特定を行うシグネチャの生成を行う事が可能となる．

4. 実験

本章では，3 章の手法を利用した具体的なシグネチャを生成方法をまとめる．なお，3 章の入力データを作成するために事前調査を行い，55 のダークネットマーケットに関するホームページ HTML ファイル，チェックアウトページ HAR ファイルを収集し (以後，マーケットデータとする)，それぞれのダークネットマーケットに依存している第三者依存サービスの特定を行っており，実験においては事前調査を行って特定した第三者依存サービスの中から，55 件のダークネットマーケットにおいて依存している件数が多かった 5 つのサービスを特定する対象とした．具体的には，WooCommerce，jQuery，Google Fonts，Blockonomics の 5 つのサービスを対象とした．

4.1 技術資料の作成

技術資料の作成には，LLM がブラウザ画面を理解し操作することを可能とする Python ライブラリである Browser-use[16] を利用した．本手法では，このライブラリに与えるプロンプトを，特定サービスの技術情報を効率的に収集できるよう設計した．

収集するデータとしては 3.1 節記載のある情報の収集を行った．プロンプトは，目的，入力パラメータ，タスクリスト，実行手順，情報収集の指針といった主要な要素から構成される．その目的は，指定したサービスに関する技術情報を収集することである．入力パラメータとして，シグネチャ生成対象となるサービスの名称（`service_name`）を指定する．タスクリストは，収集すべき具体的な情報項目を定めており，実行手順は，タスクを開始するための具体的な指示を与える．特に，「‘Focus on finding concrete technical information rather than general descriptions‘」という指示を設けることで，抽象的な説明ではなく，具体的な技術情報に焦点を当てるよう誘導している．

プロンプトが要求するタスクは，第三者依存サービスを実際に利用するために不可欠な技術的詳細情報の収集に集約される．具体的には，メイン Web サイト URL，公式資料 URL，API 資料 URL，認証方式，API レスポンス形式，共通 HTTP ヘッダー，エラーレスポンスパターンの計 7 項目である．これらの収集情報がリスト形式で簡潔かつ明確に提示されるよう，出力形式を統一した．

4.2 LLM によるシグネチャ生成・検証

本実験ではシグネチャの生成プロセスと検証プロセスを同時に行い，シグネチャ生成においては OpenAI 社が提供する gpt-4.1-2025-04-14[17] を利用した．しかし，現行の LLM が持つコンテキストウィンドウの上限（本実験では gpt-4.1-2025-04-14 の 1,047,576 トークンを前提とする）を考慮し，効率的かつ高精度なシグネチャ生成を行った．

シグネチャ生成の対象とするサービスは，前述の通り WooCommerce，jQuery，Google Fonts，Blockonomics の 5 つとした．

本研究においては，事前調査において収集した各マーケットと各マーケットに依存している第三者サービスの依存性を人手で確認した結果を正解データとして利用した．シグネチャ生成に用いる入力データの組み合わせを以下の 3 種類のパターンに分類した．各パターンにおいて，依存マーケットデータは正解データセットからランダムに選択した．

- **パターン 1 (Single)**：依存マーケットデータ 1 件と非依存マーケットデータ 1 件.
- **パターン 2 (Double)**：依存マーケットデータ 2 件と非依存マーケットデータ 1 件.
- **パターン 3 (Triple)**：依存マーケットデータ 3 件と非依存マーケットデータ 1 件.

各サービスにつき 10 回ずつ試行し，合計 30 個のシグネチャを生成した．

4.2.1 初期シグネチャの生成と検証

まず，各サービスの依存性が確認されているマーケットデータを用いて，初期シグネチャを生成した．この段階の目的は，対象サービスを確実に検出する正規表現を構築することである．

依存マーケットデータは，事前に人間が依存性を確認した Web サイトの HTML ファイルおよび HAR ファイルである．プロンプトには，サービス名と技術資料，そして HTML ファイルまたは HAR ファイルをペアで入力した．

具体的には，以下の 2 種類の入力形式で処理を行った．

- **HTML 用プロンプト**：このプロンプトでは，与えられた HTML ファイルを分析し，特定のサービスの依存関係を検出するためのシグネチャを生成するよう LLM に指示する．サービス名に加えて，スクリプトソース，CSS クラス，データ属性，フォーム関連要素，URL，変数名など，HTML に特有の技術的指標を分析対象として指定した．出力は，単一行の正規表現パターンのみとした．
- **HAR 用プロンプト**：このプロンプトでは，与えられた HAR ファイルを分析し，特定のサービスの依存関係を検出するためのシグネチャを生成する．サービス名と技術資料を基に，リクエスト URL，ヘッダー，クッキー，POST データ，API エンドポイントなど，通信データに特有の技術的指標を分析するよう指示した．出力は，HTML 用プロンプトと同様に，単一行の正規表現パターンのみとした．

複数の依存マーケットデータが存在する場合には，それぞれのデータに対して同様の処理を行い，得られた複数のシグネチャを統合プロンプトに入力して最終的なシグネチャを作成した．また，シグネチャ生成時に LLM が参照したマーケットデータ部分を保存し，そのコードが元データに実際に含まれているかを機械的に確認した．このデータは，次ステップのシグネチャ最適化に利用する．

作成された初期シグネチャは，生成に用いた依存マーケットデータに適用し，対象サービスを 100 %の精度で検出するかを検証した．この検証で検出精度が 100 %に満たない場合，再度シグネチャ生成のプロセスを実施した．

4.2.2 初期シグネチャの最適化と検証

初期シグネチャは，依存データのみを対象としているため，意図しないデータ（以下，非依存マーケットデータ）に対しても誤って検出してしまう過剰適合（オーバーフィッティング）の恐れがある．そこで，このステップでは非依存マーケットデータを組み合わせて，シグネチャの最適化を実施した．非依存マーケットデータは，事前に人間が依存性がないことを確認した Web サイトの HTML ファイルおよび HAR ファイルである．

具体的には，初期シグネチャを生成する際に保存しておいた参照マーケットデータと，非依存マーケットデータを LLM に入力し，依存と非依存を明確に識別できるようにシグネチャの改良を指示した．このプロセスにより，シグネチャは不要なパターンを除外するように最適化される．

最適化されたシグネチャは，シグネチャ最適化の際に用いた非依存マーケットデータに適用し，検出しないことを確認した．この条件を満たさない場合は，最適化のプロセスを繰り返し実施した．

以上により，本手法は LLM のコンテキストウィンドウの制限を回避しつつ，大規模かつ多様なダークネットマーケットデータを対象とした第三者サービス依存性検出のための高精度シグネチャ生成を可能にする．さらに，依存・非依存データ双方を利用することで，過剰適合を抑制し，汎用性の高い検出ルールを構築することができる．

5. 結果

本章では，4 章で実験を行った内容に関しての結果をまとめる．

5.1 技術資料

4.1 節で述べた手法に従い，本実験で特定対象とした 5 つのサービス（WooCommerce，jQuery，Google Fonts，Blockonomics，MyCryptoCheckout）それぞれについて技術資料を作成した．これらの資料は，各サービスが提供する API や機能に関する，網羅的かつ具体的な技術情報を含んでいる．

例えば，WooCommerce の資料には，サービスの全体概要，メイン Web サイトや開発者向け資料，API リファレンスの URL といったユニークな識別子に関する情報が記述されている．さらに，API キー認証や OAuth 1.0a といった認証方式，‘Content-Type: application/json‘や ‘Accept: application/json‘といった一般的な HTTP ヘッダー，そして ‘”error”: ”Invalid API Key”‘のようなエラーレスポンスパターンも詳細にまとめられている．この資料は，LLM が WooCommerce を識別するためのキーとなる情報源として機能する．

同様に，MyCryptoCheckout の資料には，‘https://mycryptocheckout.com‘といった公式サイト URL，‘https://mycryptocheckout.com/doc/developers/‘といった開発者向け資料 URL，および ‘https://bitbucket.org/mycryptocheckout/api/src/master/‘といった API リファレンス URL が含まれている．認証方式として，クライアントとサーバー間で秘密鍵を用いる認証がサポートされていることや，例として ‘Authorization: Bearer SECRET_KEY ‘ヘッダーや ‘?secret_key=SECRET_KEY ‘といったクエリパラメータが挙げられている．また，‘”error”: ”Invalid secret key”‘といったエラーメッセージや，‘401 Unauthorized‘といった HTTP ステータスコードも記録されている．

このように，作成された各技術資料は，特定のサービスに特有の技術的特徴を体系的に整理したものであり，後続のシグネチャ生成プロセスにおいて，LLM が適切なパターンを抽出するための重要な参照情報となった．詳細な資料は，GitHub にて公開している [18]．サービスごとに 1 つの技術資料を作成し，作成した資料の内容に関して正しい URL を指定して，正しいデータが集まっているかを人間が手動でチェックを行った．

5.2 シグネチャ生成

表 1 は，本研究で生成したシグネチャの例を示している．各シグネチャは，特定の第三者サービスを識別するための特徴的なリクエストパターンやリソース参照を反映しており，正規表現として記述されている．例えば，WooCommerce のシグネチャではチェックアウト処理や商品追加に関するパスが含まれ，Google Fonts のシグネチャでは外部フォントの読み込みリクエストが検知可能となっている．このように，サービスごとの利用特性を反映したシグネチャを生成することで，ダークネットマーケットにおける依存関係を効率的に特定できる．

本実験では，5 つのサービスでそれぞれ 30 個のシグネチャを生成した．シグネチャを生成する上で再生成する必要のあるシグネチャも存在したが，全て 5 回以内の生成で検証の条件を満たすシグネチャ生成を行うことができた．

6. 評価

6.1 シグネチャの評価方法

本章では，提案手法の有効性を確認するため，4.2 節で生成した各サービス 30 個ずつのシグネチャの検出性能を評価した．シグネチャの評価は，シグネチャ生成に利用しなかったマーケットデータに対して適用することで実施した．この評価は，シグネチャが未知のデータに対しても汎用的に機能するかを確認することを目的としている．

これは，異なるランダムなサンプルセットからシグネチャを生成することで，生成されるシグネチャの安定性と汎用性を評価するためである．なお，紙面の都合上，全てのシグネチャの評価結果を掲載することは困難なため，各作成方法で最も高い性能を示したシグネチャの結果 (highest score) と，最も低い性能を示したシグネチャの結果 (lowest score) を抜粋して記述する．評価指標としては，以下の 4 つの指標を算出した．

- **True Positive (TP)**：実際にサービスに依存しており，かつシグネチャが正しく検出したもの．
- **True Negative (TN)**：実際にはサービスに依存しておらず，かつシグネチャが正しく検出しないもの．
- **False Positive (FP)**：実際にはサービスに依存しておらず，シグネチャが誤って検出してしまったもの．
- **False Negative (FN)**：実際にサービスに依存しているにもかかわらず，シグネチャが検出できなかったもの．

これらの基本指標に基づき，Accuracy（正解率），Precision（適合率），Recall（再現率），F1 Score（F 値）を算出した．なお，本実験の全評価データおよび生成されたシグネチャは，公開リポジトリにて参照可能である．

6.2 評価の結果

本節では，各サービスに対して生成されたシグネチャの検出性能を示す．

6.2.1 WooCommerce

WooCommerce に対するシグネチャの評価結果を表 2 に示す．‘Single_HighScore‘，‘Double_HighScore‘，‘Triple_HighScore‘の各パターンにおいて，Accuracy，Precision，Recall，F1 Score はいずれも高い値を示した．特に Precision は全て 1.00 であり，誤検出（FP）が一件も発生していないことがわかる．これは，生成されたシグネチャが非常に厳密な条件でサービスを特定していることを示唆する．一方，‘Single_LowScore‘や ‘Triple_LowScore‘では，FN が若干発生しており，検出漏れが認められる．

6.2.2 jQuery

jQuery の評価結果を表 3 に示す．このサービスは，Web サイト内に広く普及しているため，シグネチャの生成と検証が他のサービスに比べて複雑となる可能性がある．‘Triple_HighScore‘パターンでは，TP，TN，FP，FN がいずれも完璧な結果（1.000）を示しており，このモデルが jQuery を完全に特定できていることを証明している．‘Double_HighScore‘も高い性能を示している一方，‘Single_HighScore‘や ‘LowScore‘系列ではわずかに誤検出（FP）や検出漏れ（FN）が発生しており，特定のプロンプト設定やデータセットの組み合わせが性能に影響を及ぼしていることがわかる．

6.2.3 Google Fonts

Google Fonts の評価結果を表 4 に示す．このサービスは，Web フォントの提供という限定的な用途であるため，特定の URL パターンに依存することが多い．‘Double_HighScore‘と ‘Triple_HighScore‘は，それぞれ Accuracy と F1 Score が 0.98 と非常に高い値を示しており，ほぼ完璧な検出性能を達成した．これらの結果から，本手法が，特定の技術的依存性が明確なサービスに対しても有効であることが示される．

6.2.4 Blockonomics

Blockonomics の評価結果を表 5 に示す．このサービスは，暗号資産決済に特化しており，その技術的痕跡が明確であるため，高い検出性能が期待される．‘Single_HighScore‘と ‘Double_HighScore‘は，TP，TN，FP，FN が全て完璧な結果を示しており，Accuracy，Precision，Recall，F1 Score が全て 1.000 であった．これは，生成されたシグネチャが，Blockonomics の依存性を完璧に特定できたことを意味する．

6.2.5 MyCryptoCheckout

MyCryptoCheckout の評価結果を 6 に示す．このサービスも Blockonomics と同様に，暗号資産決済に特化しているため，非常に明確な技術的痕跡を持つ．‘Single_HighScore‘，‘Double_HighScore‘，‘Triple_HighScore‘のすべてのパターンにおいて，TP，TN，FP，FN が完璧な結果を示し，すべての指標が 1.000 であった．これは，本手法が，特定の

表 1 本研究で生成したシグネチャの例

サービス名	シグネチャ例
WooCommerce	`woocommerce/|wc-ajax=|/wp-json/wc/|/checkout/($|?)|data-product-id=|/?add-to-cart=|/product/|/my-account/|/wp-content/plugins/woocommerce`
jQuery	`(?:jquery(?:.|[]))?:(?:min.)?js(?:[?"’])?|jquery.(?:scrolly|poptrox|qrcode)(?:.min)?.js|code.jquery.com|ajax.googleapis.com/ajax/libs/jquery|jquery.ui)`
Google Fonts	`<link^>[]>+href=["’]https://fonts.googleapis.com/css(?:?|["’])`
Blockonomics	`blockonomics(?:-bitcoin-payments|logo.png|_|-)|pay_(?:button|widget).js|/api/(?:button|merchant_order)|wss://blockonomics.co|/pay-url/`
MyCryptoCheckout	`/wp-content/plugins/mycryptocheckout/|id=[’’]mycryptocheckout-(checkout_data|css|js|web3(-js)?)[’’]|data-mycryptocheout_checkout_data=.js|code.jquery.com|ajax.googleapis.com/ajax/libs/jquery|jquery.ui)`

表 2 WooCommerce に対する各パターンの検出性能

Pattern ID	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Single_HighScore	28	24	0	1	0.964	1.000	0.966	0.98
Single_LowScore	25	24	0	4	0.909	1.000	0.862	0.93
Double_HighScore	27	24	0	1	0.964	1.000	0.964	0.98
Double_LowScore	26	24	0	2	0.945	1.000	0.929	0.96
Triple_HighScore	26	24	0	1	0.982	1.000	0.963	0.98
Triple_LowScore	24	24	0	3	0.927	1.000	0.889	0.94
F1 Ave	-	-	-	-	-	-	-	0.97

表 3 jQuery に対する各パターンの検出性能

Pattern ID	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Single_HighScore	37	12	2	2	0.909	0.949	0.949	0.95
Single_LowScore	28	14	0	11	0.782	1.000	0.718	0.84
Double_HighScore	38	11	3	0	0.927	0.927	1.000	0.96
Double_LowScore	29	14	0	9	0.818	1.000	0.763	0.87
Triple_HighScore	37	14	0	0	1.000	1.000	1.000	1.00
Triple_LowScore	35	14	0	2	0.945	1.000	0.946	0.97
F1 Ave	-	-	-	-	-	-	-	0.94

表 4 Google Fonts に対する各パターンの検出性能

Pattern ID	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Single_HighScore	26	25	0	2	0.945	1.000	0.929	0.96
Single_LowScore	19	25	0	9	0.818	1.000	0.679	0.81
Double_HighScore	26	25	0	1	0.982	1.000	0.963	0.98
Double_LowScore	23	25	0	4	0.909	1.000	0.852	0.92
Triple_HighScore	25	25	0	1	0.982	1.000	0.962	0.98
Triple_LowScore	22	25	0	4	0.909	1.000	0.846	0.92
F1 Ave	-	-	-	-	-	-	-	0.94

表 5 Blockonomics に対する各パターンの検出性能

Pattern ID	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Single_HighScore	11	42	0	0	1.000	1.000	1.000	1.00
Single_LowScore	9	40	2	2	0.925	0.818	0.818	0.82
Double_HighScore	10	42	0	0	1.000	1.000	1.000	1.00
Double_LowScore	10	40	2	0	0.962	0.833	1.000	0.91
Triple_HighScore	9	40	2	0	0.961	0.818	1.000	0.90
Triple_LowScore	8	39	0	4	0.922	1.000	0.667	0.80
F1 Ave	-	-	-	-	-	-	-	0.90

Web サイトに依存する決済サービスを非常に高い精度で特定できることを証明している．

7. 考察

本章では，提案手法による評価結果を詳細に分析する．特に，大規模言語モデル（LLM）がシグネチャ生成においてどのように機能したのか，サンプル数の増減が性能にどのような影響を与えたのかを多角的に検証し，本研究の学術的・実用的な意義を明らかにする．

表 6 MyCryptoCheckout に対する各パターンの検出性能

Pattern ID	TP	TN	FP	FN	Accuracy	Precision	Recall	F1 Score
Single_HighScore	6	47	0	0	1.00	1.00	1.00	1.00
Double_HighScore	5	47	0	0	1.00	1.00	1.00	1.00
Triple_HighScore	4	47	0	0	1.00	1.00	1.00	1.00
F1 Ave	-	-	-	-	-	-	-	1.00

7.1 シグネチャの特性とサービス間の性能差

本研究では 5 つのサービスに関してシグネチャの作成を行った．サービスごとに 30 個のシグネチャを作成し，それぞれで評価を行った．結果的には全てのシグネチャで精度が 90%を超える結果となった．これは，LLM が第三者サービスの技術資料と依存・非依存のマーケットデータを参考に作成したシグネチャが有効的であることを示唆している．

特に，MyCryptoCheckout は全ての試行で完璧な検出精度（Accuracy=1.000）を達成した．これは，同サービスが仮想通貨決済ゲートウェイという性質上，特定の API エンドポイントや，固有の JavaScript ファイル名（例：‘mycryptocheckout.js‘）など，他サービスと重複しにくいユニークな特徴を持つためと考えられる．LLM は，このような明確な特徴を少数のサンプルからでも正確に抽出できることが判明した．

一方で，jQuery や WooCommerce の一部のパターンでは，他のサービスやパターンと比較して低い性能を示した．これは以下の 2 つの要因に起因すると考えられる．一方で，jQuery の Single_8 では FN が 11 件と他よりも精度が低く，見逃し件数が多かった．原因として考えられるのは，jQuery は極めて汎用的な JavaScript ライブラリであり，そのロード方法や利用パターンが多岐にわたる．例えば，CDN からのロード，ローカルファイルとしてバンドル，または特定のプラグイン内でのみ利用されるなど，実装が多様であるため，単一のサンプルからではその全てのバリエーションを網羅するシグネチャを生成するのは困難

であったと考えられる．

これらの結果から，LLM を利用したシグネチャの生成は高い精度を出すことができる．特定したいサービスの固有の振る舞い（API 呼び出しなど）やユニークな記述が存在することによって 100%の精度を出すことができるケースがある一方で，汎用的で多様な実装方法を持つ第三者サービスに関しては，少量の件数ではあるものの見逃しや誤検知を発生させてしまう事が判明した．

7.2 サンプル数の影響とシグネチャの頑健性

本研究は，シグネチャ生成に用いる依存マーケットデータの数を 1 件から 3 件と変えることで，LLM が学習するサンプル数によって精度が向上するかどうかに関しても検証を行った．結果的にはサンプル数の増加がシグネチャの精度向上に大きく関連していないということが判明した．シグネチャを生成するための元データの件数による精度のばらつきよりも複数のパターンでのばらつきの方が大きかったため，シグネチャを生成するためのデータ数よりもシグネチャを生成するための元データの方が重要であるということが考えられる．

特に，MyCryptoCheckout のように，もともと特徴が明確なサービスにおいては，Single パターンでもすでに完璧な性能を発揮しており，サンプル数の増加が性能に与える影響は限定的であった．本研究の結果は，少数のサンプルからでも実用的なシグネチャ生成が可能であるという提案手法の主張を裏付けるものであり，データ数が少量であっても有効的なシグネチャを生成することができることを証明した．

8. おわりに

本研究は，LLM が依存・非依存のマーケットデータと技術資料を統合的に分析し，高精度な検出シグネチャを自動生成できることを実証した．この成果は，従来の手動で第三者サービスを特定する手法よりも効率的に高い精度で第三者サービスを特定することができる手法であることを示唆している．

謝辞 本研究の一部はＮＥＤＯ（国立研究開発法人新エネルギー・産業技術総合開発機構）の委託事業「経済安全保障重要技術育成プログラム／先進的サイバー防御機能・分析能力強化」（JPNP24003）によるものである．青砥 陸氏のご協力に感謝申し上げる．

参考文献

[1] N. Christin. Traveling the silk road: a measurement analysis of a large anonymous online marketplace. In Proceedings of the 22nd International Conference on World Wide Web, pp. 213–224, 2013.

[2] A. Biryukov, I. Pustogarov, and R. Weinmann. Trawling for tor hidden services: Detection, measurement, deanonymization. In 2013 IEEE Symposium on Security and Privacy (S&P), pp. 80–94, 2013.

[3] A. Kwon, M. AlSabah, M. Juarez, G. Acar, V. Shmatikov, C. Diaz, and N. Hopper. Circuit fingerprinting attacks: Passive deanonymization of tor hidden services. In Proceedings of the 24th Network and Distributed System Security Symposium (NDSS), 2015.

[4] A. Chaabane, P. Manils, and M. Kaafar. Digging into anonymous traffic: A deep analysis of tor bridges. In 2010 10th Annual Conference on Privacy, Security and Trust (PST), pp. 1–8, 2010.

[5] M. Edman and P. Syverson. AS-awareness in tor path selection. In Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS), pp. 380–389, 2009.

[6] WooCommerce. WooCommerce: The most-trusted ecommerce platform. `https://github.com/browser-use/browser-use`. Accessed: 2025-08-01.

[7] The jQuery Foundation. jQuery: A fast, small, and feature-rich javascript library. `https://jquery.com/`. Accessed: 2025-08-01.

[8] Google Fonts Team. Google Fonts: Making the web more beautiful, fast, and open through great typography. `https://fonts.google.com/`. Accessed: 2025-08-01.

[9] Blockonomics. Blockonomics: Blockonomics helps you track and accept crypto payments. `https://www.blockonomics.co/`. Accessed: 2025-08-01.

[10] MyCryptoCheckout. MyCryptoCheckout: Accept cryptocurrency payments for wordpress. `https://mycryptocheckout.com/`. Accessed: 2025-08-01.

[11] N. Christin. Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In Proceedings of the 22nd International World Wide Web Conference (WWW), pp. 213–224, Rio de Janeiro, Brazil, 2013.

[12] K. Soska and N. Christin. Measuring the longitudinal evolution of the online anonymous marketplace ecosystem. In Proceedings of the 24th USENIX Security Symposium, pp. 33–48, 2015.

[13] K. Kruithof, J. Aldridge, D. Décary-Hétu, M. Sim, A. Dujso, and E. Hoorens. Internet-facilitated drugs trade: An analysis of the size, scope and the role of the netherlands. Technical report, RAND Europe, 2016.

[14] A. Lerner, A. Simpson, T. Kohno, and F. Roesner. Internet jones and the raiders of the lost trackers: An archaeological study of web tracking from 1996 to 2016. In Proceedings of the 25th International World Wide Web Conference (WWW), pp. 1231–1241, 2016.

[15] K. Thomas, E. Bursztein, C. Grier, G. Ho, N. Jagpal, A. Kapravelos, D. McCoy, A. Nappa, V. Paxson, P. Pearce, N. Provos, and M. Rajab. Ad injection at scale: Assessing deceptive advertisement modifications. In 2015 IEEE Symposium on Security and Privacy (S&P), pp. 151–167, 2015.

[16] browser-use contributors. browser-use: A library for llm-driven browser automation. `https://github.com/browser-use/browser-use`. Accessed: 2025-07-31.

[17] OpenAI. OpenAI: Model - openai api. `https://platform.openai.com/docs/models/gpt-4.1`. Accessed: 2025-08-17.

[18] GitHub. GitHub: — ylab-organization-1/dnm_result. `https://github.com/ylab-organization-1/dnm_result`. Accessed: 2025-08-20.