

明 細 書

発明の名称：

ウェブアプリケーションの脆弱性を検査するためのシステム、方法、及びプログラム

技術分野

[0001] 本発明は、ウェブアプリケーションの脆弱性を検査するためのシステム、方法、及びプログラムに関するものである。

背景技術

[0002] 従来、アプリケーションの脆弱性を、ソースコードの静的解析によって検出することが行われている（例えば、下記特許文献1を参照）。こうした静的解析では、典型的には、ソースコード全体のデータフローの検査が行われ、悪意のある入力から脆弱性の発現までのフローが再現される。

先行技術文献

特許文献

[0003] 特許文献1：特開2019-003309号公報

発明の概要

発明が解決しようとする課題

[0004] しかしながら、上述したような静的解析は、言語及びフレームワーク等の環境毎に異なる実装とする必要があるため、多様な環境で構築される様々なアプリケーションの脆弱性の検査に一律に適用することが困難であった。

[0005] 本発明の実施形態は、様々なアプリケーションの脆弱性の検査を可能とすることを目的の一つとする。本発明の実施形態の他の目的は、本明細書全体を参照することにより明らかとなる。

課題を解決するための手段

[0006] 本発明の一実施形態に係るシステムは、1又は複数のコンピュータプロセッサを備え、ウェブアプリケーションの脆弱性を検査するためのシステムで

あって、前記 1 又は複数のコンピュータプロセッサは、前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第 1 指示情報を生成するステップと、前記第 1 指示情報を第 1 大規模言語モデルに対して入力し、前記第 1 大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第 2 指示情報を生成するステップと、前記第 2 指示情報を第 2 大規模言語モデルに対して入力し、前記第 2 大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を実行する。

[0007] 本発明の一実施形態に係る方法は、1 又は複数のコンピュータによって実行され、ウェブアプリケーションの脆弱性を検査するための方法であって、前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第 1 指示情報を生成するステップと、前記第 1 指示情報を第 1 大規模言語モデルに対して入力し、前記第 1 大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第 2 指示情報を生成するステップと、前記第 2 指示情報を第 2 大規模言語モデルに対して入力し、前記第 2 大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を備える。

[0008] 本発明の一実施形態に係るプログラムは、ウェブアプリケーションの脆弱

性を検査するためのプログラムであって、1又は複数のコンピュータに、前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第1指示情報を生成するステップと、前記第1指示情報を第1大規模言語モデルに対して入力し、前記第1大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第2指示情報を生成するステップと、前記第2指示情報を第2大規模言語モデルに対して入力し、前記第2大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を実行させる。

発明の効果

[0009] 本発明の様々な実施形態は、様々なアプリケーションの脆弱性の検査を可能とする。

図面の簡単な説明

[0010] [図1]本発明の一実施形態に係る脆弱性検査サーバ10を含むネットワークの構成を概略的に示す構成図。

[図2]検査ルール情報テーブル152が管理する情報を例示する図。

[図3]検査管理テーブル154が管理する情報を例示する図。

[図4]検査詳細管理テーブル156が管理する情報を例示する図。

[図5]検査画面50を例示する図。

[図6]検査ルール指定画面60を例示する図。

[図7]脆弱性検査を行う際にサーバ10が実行する処理を例示するフローチャート。

[図8]エンドポイント情報取得用のプロンプトを例示する図。

[図9]エンドポイント情報取得用のプロンプトが入力された大規模言語モデル

からの応答を例示する図。

[図10]仕様情報取得用のプロンプトを例示する図。

[図11]仕様情報取得用のプロンプトが入力された大規模言語モデルからの応答を例示する図。

[図12]脆弱性情報取得用のプロンプトを例示する図。

[図13]脆弱性情報取得用のプロンプトが入力された大規模言語モデルからの応答を例示する図。

[図14]仕様情報抽出用のプロンプトを例示する図。

[図15]仕様情報抽出用のプロンプトが入力された大規模言語モデルからの応答を例示する図。

発明を実施するための形態

[0011] 以下、図面を参照しながら、本発明の実施形態について説明する。各図面において、同一の又は類似する構成要素に対しては同一の参照符号が付され得る。

[0012] 図1は、本発明の一実施形態に係る脆弱性検査サーバ10を含むネットワークの構成を概略的に示す構成図である。サーバ10は、図示するように、インターネット等の通信ネットワーク20を介してユーザ端末30と通信可能に接続されている。図1においては、1つのユーザ端末30のみが図示されているが、サーバ10は、複数のユーザ端末30と通信可能に接続されている。脆弱性検査サーバ10は、ユーザ端末30を操作するユーザに対して、ウェブアプリケーションの脆弱性の静的解析を行う脆弱性検査サービスを提供する。脆弱性検査サーバ10は、本発明のシステムの全部又は一部を実装する装置の一例である。

[0013] まず、脆弱性検査サーバ10のハードウェア構成について説明する。脆弱性検査サーバ10は、一般的なコンピュータとして構成されており、図1に示すように、コンピュータプロセッサ11と、メインメモリ12と、入出力I/F13と、通信I/F14と、ストレージ（記憶装置）15とを備え、これらの各構成要素が図示しないバス等を介して電氣的に接続されている。

- [0014] コンピュータプロセッサ 11 は、CPU 又は GPU 等として構成され、ストレージ 15 等に記憶されている様々なプログラムをメインメモリ 12 に読み込んで、当該プログラムに含まれる各種の命令を実行する。メインメモリ 12 は、例えば、DRAM 等によって構成される。
- [0015] 入出力 I/F 13 は、操作者等との間で情報をやり取りするための各種の入出力装置を含む。入出力 I/F 13 は、例えば、キーボード、ポインティングデバイス（例えば、マウス、タッチパネル等）等の情報入力装置、マイクロフォン等の音声入力装置、カメラ等の画像入力装置を含む。また、入出力 I/F 13 は、ディスプレイ等の画像出力装置、スピーカー等の音声出力装置を含む。
- [0016] 通信 I/F 14 は、ネットワークアダプタ等のハードウェア、各種の通信用ソフトウェア、又はこれらの組み合わせとして実装され、通信ネットワーク 20 等を介した有線又は無線の通信を実現できるように構成されている。
- [0017] ストレージ 15 は、例えば、磁気ディスク又はフラッシュメモリ等によって構成される。ストレージ 15 は、オペレーティングシステムを含む様々なプログラム及び各種データ等を記憶する。例えば、ストレージ 15 は、図 1 に示すように、脆弱性検査に適用可能な検査ルールに関する情報を管理する検査ルール情報テーブル 152 と、個別の検査に関する情報を管理する検査管理テーブル 154 と、個別の検査の詳細に関する情報を管理する検査詳細管理テーブル 156 とを有する。これらのテーブルは、その一部が 1 つのテーブルに統合されてもよいし、複数のテーブルに分割されてもよい。
- [0018] また、例えば、ストレージ 15 は、本発明の一実施形態に係るサーバ側プログラム 40 を記憶する。当該プログラム 40 は、サーバ 10 を、脆弱性検査サービスを提供するためのシステムの全部又は一部として機能させるためのプログラムである。サーバ側プログラム 40 の少なくとも一部は、後述する端末側プログラム 42 を介して、ユーザ端末 30 側において実行されるように構成され得る。
- [0019] 本実施形態において、脆弱性検査サーバ 10 は、それぞれが上述したハー

ドウェア構成を有する複数のコンピュータを用いて構成され得る。例えば、サーバ１０は、複数のサーバ装置によって構成される。

[0020] このように構成された脆弱性検査サーバ１０は、ウェブサーバ及びアプリケーションサーバとしての機能を有するように構成することができ、ユーザ端末３０からの要求に応答して各種の処理を実行し、当該処理の結果に応じた画面データ（例えば、HTMLデータ）及び制御データ等をユーザ端末３０に対して送信する。ユーザ端末３０では、受信したデータに基づくウェブページ又はその他の画面が出力される。

[0021] 次に、ユーザ端末３０のハードウェア構成について説明する。ユーザ端末３０は、一般的なコンピュータとして構成されており、図１に示すように、コンピュータプロセッサ３１と、メインメモリ３２と、入出力Ｉ／Ｆ３３と、通信Ｉ／Ｆ３４と、ストレージ（記憶装置）３５とを備え、これらの各構成要素が図示しないバス等を介して電氣的に接続されている。

[0022] コンピュータプロセッサ３１は、CPU又はGPU等として構成され、ストレージ３５等に記憶されている様々なプログラムをメインメモリ３２に読み込んで、当該プログラムに含まれる各種の命令を実行する。メインメモリ３２は、例えば、DRAM等によって構成される。

[0023] 入出力Ｉ／Ｆ３３は、操作者等との間で情報をやり取りするための各種の入出力装置を含む。入出力Ｉ／Ｆ３３は、例えば、キーボード、ポインティングデバイス（例えば、マウス、タッチパネル等）等の情報入力装置、マイクロフォン等の音声入力装置、カメラ等の画像入力装置を含む。また、入出力Ｉ／Ｆ３３は、ディスプレイ等の画像出力装置、スピーカー等の音声出力装置を含む。

[0024] 通信Ｉ／Ｆ３４は、ネットワークアダプタ等のハードウェア、各種の通信用ソフトウェア、及びこれらの組み合わせとして実装され、通信ネットワーク２０等を介した有線又は無線の通信を実現できるように構成されている。

[0025] ストレージ３５は、例えば、磁気ディスク又はフラッシュメモリ等によって構成される。ストレージ３５は、オペレーティングシステムを含む様々な

プログラム及び各種データ等を記憶する。ストレージ35が記憶するプログラムは、アプリケーションマーケット等からダウンロードされてインストールされ得る。また、ストレージ35は、上述した端末側プログラム42を記憶する。当該プログラム42は、ウェブブラウザ、又は、その他のアプリケーション（例えば、本実施形態の脆弱性検査サービス用の端末側アプリケーション等）として構成され、上述したように、サーバ側プログラム40の少なくとも一部を実行するように構成され得る。

[0026] 本実施形態において、ユーザ端末30は、スマートフォン、タブレット端末、又はパーソナルコンピュータ等として構成され得る。

[0027] このように構成されたユーザ端末30を操作するユーザは、ストレージ35等にインストールされている端末側プログラム42を介したサーバ10との通信を実行することによって、当該サーバ10が提供する脆弱性検査サービスを利用することができる。

[0028] 次に、このように構成された脆弱性検査サーバ10が有する機能について説明する。サーバ10のコンピュータプロセッサ11は、図1に示すように、メインメモリ12に読み込まれたプログラム（例えば、サーバ側プログラム40の少なくとも一部）に含まれる命令を実行することによって、管理機能制御部112、及び、検査制御部114として機能するように構成されている。

[0029] 管理機能制御部112は、脆弱性検査サービスの管理機能の制御に関する様々な処理を実行するように構成されている。例えば、管理機能制御部112は、当該管理機能に関する様々な画面の画面データ及び制御データ等をユーザ端末30に対して送信し、ユーザ端末30において出力される当該画面を介したユーザによる操作入力に応答して様々な処理を実行し、当該処理の結果に応じた画面データ及び制御データ等をユーザ端末30に対して送信する。管理機能制御部112によって制御される管理機能は、例えば、ログイン処理（ユーザ認証）、課金制御、及び、ユーザアカウントの管理等を含む。

- [0030] 検査制御部 114 は、脆弱性の検査の制御に関する様々な処理を実行するように構成されている。例えば、検査制御部 114 は、当該検査を制御するための様々な画面の画面データ及び制御データ等をユーザ端末 30 に対して送信し、ユーザ端末 30 において出力される当該画面を介したユーザによる操作入力に応答して様々な処理を実行し、当該処理の結果に応じた画面データ及び制御データ等をユーザ端末 30 に対して送信する。
- [0031] 本実施形態において、検査制御部 114 は、ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報を、当該ウェブアプリケーションのソースコードに基づいて出力することを指示するエンドポイント情報取得用の第 1 指示情報（プロンプト）を生成するように構成されている。本実施形態において、エンドポイント情報は、対応するエンドポイントにおいて実行される処理に関する処理情報（例えば、HTTP メソッド、及び／又は、コード）を含む。
- [0032] エンドポイントは、API エンドポイントと呼ばれることがあり、クライアントが特定の機能又はデータにアクセスするための入口となり、外部からの攻撃の入力がある箇所・単位であるとも言える。クライアントは、HTTP リクエストを送信することによって、エンドポイントからのレスポンスを得ることができる。エンドポイント情報は、エンドポイントに対応するパスを含み得る。
- [0033] また、検査制御部 114 は、生成した第 1 指示情報を第 1 大規模言語モデル（LLM）に対して入力し、当該第 1 大規模言語モデルから出力されるエンドポイント情報を取得するように構成されている。大規模言語モデルは、多量のデータを用いて訓練された、自然言語処理における機械学習モデルであり、例えば、米国 OpenAI 社が提供する GPT、若しくは、米国 Google 社が提供する BERT 又は PaLM 等が適用され得る。例えば、指示情報は、こうした大規模言語モデルに対応する API を介して入力される。
- [0034] また、検査制御部 114 は、対応するエンドポイントの仕様が自然言語で

記述された仕様情報を取得するように構成されている。

- [0035] また、検査制御部 114 は、エンドポイントの脆弱性に関する脆弱性情報を、対応するエンドポイント情報及び仕様情報に基づいて出力することを指示する脆弱性情報取得用の第 2 指示情報（プロンプト）を生成し、生成した第 2 指示情報を第 2 大規模言語モデルに対して入力し、当該第 2 大規模言語モデルから出力される脆弱性情報を取得するように構成されている。
- [0036] このように、本実施形態における脆弱性検査サーバ 10 は、大規模言語モデルを用いて、ウェブアプリケーションにおける複数のエンドポイントの各々において実行される処理情報を含むエンドポイント情報をソースコードに基づいて取得すると共に、当該エンドポイントの脆弱性情報を、対応するエンドポイント情報及び仕様情報に基づいて取得し、こうした仕組みは、多様な環境で構築される様々なアプリケーションの脆弱性の検査に一律に適用され得る。つまり、サーバ 10 は、様々なアプリケーションの脆弱性の検査を可能とする。
- [0037] また、脆弱性検査サーバ 10 は、自然言語で記述された仕様情報に基づいて脆弱性情報を取得するから、ビジネスロジックに基づく脆弱性（例えば、権限管理の問題等の脆弱性等）の検知が可能となり得る。
- [0038] 本実施形態において、エンドポイント情報を取得するために用いる第 1 大規模言語モデル、及び、脆弱性情報を取得するために用いる第 2 大規模言語モデルは、同一の大規模言語モデルであってもよいし、異なる大規模言語モデルであってもよい。
- [0039] 本実施形態において、仕様情報の取得は、様々な方法で実現され得る。例えば、検査制御部 114 は、ストレージ 15 等において予め記憶されている仕様情報（例えば、ウェブアプリケーションの管理者等によって事前に提供される。）を取得する。
- [0040] また、仕様情報が、大規模言語モデルを用いて取得されるようにしてもよい。つまり、検査制御部 114 は、仕様情報をエンドポイント情報に基づいて出力することを指示する仕様情報取得用の第 3 指示情報を生成し、当該第

3 指示情報を第3大規模言語モデルに対して入力し、当該第3大規模言語モデルから出力される仕様情報を取得するように構成され得る。当該第3大規模言語モデルは、第1大規模言語モデル及び／又は第2大規模言語モデルと同一の大規模言語モデルであってもよいし、異なる大規模言語モデルであってもよい。こうした構成は、ウェブアプリケーションのソースコードに基づくエンドポイントの仕様情報の取得を可能とする。

[0041] 本実施形態において、脆弱性情報を取得するための第2指示情報は、予め定められた脆弱性の検査ルールに従って脆弱性情報を出力することを指示するように構成され得る。例えば、検査制御部114は、検査ルール情報テーブル152において管理されている検査ルール（例えば、脆弱性検査サービスの提供者によって事前に登録される。）を取得し、当該検査ルールに従って脆弱性情報を出力することを指示する第2指示情報を生成するように構成される。こうした構成は、大規模言語モデルの脆弱性の判定精度の向上、大規模言語モデルにとって未学習の情報（例えば、新しい技術や認知度の低い脆弱性等）に基づく検査ルールの適用、及び、特定の条件を満たした場合に脆弱と判断する等のカスタマイズの実現を可能とし得る。

[0042] 次に、このような機能を有する本実施形態の脆弱性検査サーバ10の一態様としての具体例について説明する。まず、この例において、各テーブルが管理する情報について説明する。図2は、この例において、検査ルール情報テーブル152が管理する情報を例示する。この例における検査ルール情報テーブル152は、脆弱性検査に適用可能な検査ルールに関する情報を管理し、図示するように、個別の検査ルールを識別する「検査ルールID」に対応付けて、「ルール名」、検査ルールが自然言語で記述された「検査ルール内容」等の情報を管理する。こうした検査ルールは、この例では、脆弱性検査サービスの提供者によって事前に登録される。検査ルールは、例えば、以下のようなルールを含む。

ルール1：コード内で「badFunction」の引数に「userId

」が指定されていたら「権限管理の不備」として検出すること

ルール2：コード内で「`secureFunction`」を利用して文字出力をしている場合は「`Cross-Site Scripting`」として検知しないこと

ルール3：`escapeX`関数はSQL Injection対策用の関数のため、文字を正しくエスケープされるために文字列結合されていてもSQL Injectionの脆弱性がないと判定すること

[0043] 図3は、この例において、検査管理テーブル154が管理する情報を例示する。この例における検査管理テーブル154は、脆弱性の検査に関する情報を管理し、図示するように、個別の検査を識別する「検査ID」に対応付けて、検査を行うユーザに関する情報、及び、検査日時等を含む「基本情報」、検査対象のウェブアプリケーションのソースコードが記述されたソースコードファイルが格納されているパスである「ソースコードファイル格納パス」、当該ソースコードファイルのファイル名である「ソースコードファイル名」、適用される検査ルールに関する情報である「適用検査ルール情報」等の情報を管理する。適用検査ルール情報は、適用される各検査ルールを識別する検査ルールIDを含む。

[0044] 図4は、この例において、検査詳細管理テーブル156が管理する情報を例示する。この例における検査詳細管理テーブル156は、検査におけるエンドポイント単位の情報を管理し、図示するように、個別の検査を識別する「検査ID」、及び、当該検査において検査対象のウェブアプリケーションの個別のエンドポイントを識別する「エンドポイントID」の組合せに対応付けて、エンドポイントにおいて実行される処理に対応する「HTTPメソッド」、当該エンドポイントを特定する「パス」、当該エンドポイントにおいて実行される処理に対応する「コード」、当該エンドポイントにおける仕

様を自然言語で記述した情報である「仕様情報」、当該エンドポイントにおける脆弱性検査の結果である「脆弱性検査結果」等の情報を管理する。

[0045] 以上、この例において、各テーブルが管理する情報について説明した。次に、この例において、脆弱性検査サーバ10が実行する処理、及び、ユーザ端末30において出力される画面等について説明する。

[0046] 図5は、ユーザ端末30において出力される検査画面50を例示する。当該画面50は、脆弱性検査サービスのユーザが、ソースコードの静的解析による脆弱性の検査を行うため画面であり、図示するように、「ソースコードファイル指定」というテキストが付加された第1ボタン52と、「検査ルール指定」というテキストが付加された第2ボタン54と、「検査開始」というテキストが付加された第3ボタン56とを有する。

[0047] 第1ボタン52は、脆弱性を検査するウェブアプリケーションのソースコードが記述されたファイルを指定するためのオブジェクトである。当該ボタン52が選択されると、特定のファイルを指定するための図示しない画面が出力され、ユーザは、当該画面を介して、ソースコードが記述されたファイルを指定することができる。当該ファイルが指定されると、検査管理テーブル154において、ソースコードファイル格納パス、及び、ソースコードファイル名が登録される。なお、ソースコードが記述されたファイルとして、複数のファイルを指定できるようにしてもよい。

[0048] 第2ボタン54は、脆弱性検査に適用する検査ルールを指定するためのオブジェクトである。当該ボタン54が選択されると、図6に例示する検査ルール指定画面60が出力される。当該画面60は、図示するように、検査ルールを指定するための検査ルール指定領域62と、設定ボタン64と、戻るボタン66とを有する。

[0049] 検査ルール指定領域62は、脆弱性検査において適用可能な複数の検査ルール（検査ルール情報テーブル152において管理されている。）の各々のルール名を一覧表示し、各ルール名の左側には、チェックボックス621が配置されている。ユーザは、脆弱性検査に適用する検査ルールに対応するチ

チェックボックス 6 2 1 を選択状態とした上で、設定ボタン 6 4 を選択する。
当該ボタン 6 4 が選択されると、検査管理テーブル 1 5 4 において、適用検査ルール情報が登録される。

[0050] 第 3 ボタン 5 6 は、脆弱性検査を開始するためのオブジェクトである。図 7 は、当該ボタン 5 6 の選択に応じて（つまり、脆弱性検査を行う際に）、サーバ 1 0 が実行する処理を例示するフローチャートである。サーバ 1 0 は、まず、図示するように、エンドポイント情報取得用のプロンプト（第 1 指示情報）を生成する（ステップ S 1 0 0）。具体的には、ウェブアプリケーションのソースコードに基づいて、エンドポイント情報の取得を指示するプロンプトが生成される。

[0051] 図 8 は、エンドポイント情報取得用のプロンプトを例示する。当該プロンプトは、図示するように、ウェブアプリケーションのソースコードから、エンドポイントのパスと実行される関数・呼び出し先を抽出することを指示しており、具体的には、エンドポイント単位の情報として、HTTP メソッド（method）、パス（path）、及び、コード（code）を抽出することを指示している。ソースコードの内容は、検査管理テーブル 1 5 4 において管理されているソースコードファイル格納パス、及び、ソースコードファイル名によって特定されるソースコードファイルから取得される。

[0052] 図 7 のフローチャートに戻り、続いて、サーバ 1 0 は、エンドポイント情報を取得して登録する（ステップ S 1 1 0）。具体的には、エンドポイント情報取得用のプロンプトを大規模言語モデルに対して入力し、当該大規模言語モデルから出力される複数のエンドポイントの各々のエンドポイント情報（HTTP メソッド、パス、及び、コード）を検査詳細管理テーブル 1 5 6 に登録する。

[0053] 図 9 は、図 8 に例示したエンドポイント情報取得用のプロンプトが入力された大規模言語モデルからの応答を例示する。図示するように、ウェブアプリケーションのソースコードに基づいて、当該ウェブアプリケーションにおける複数のエンドポイントの各々のエンドポイント情報として、HTTP メ

ソッド (method)、パス (path)、及び、コード (code) が抽出されている。

[0054] 図7のフローチャートに戻り、続いて、サーバ10は、対象のエンドポイントについて仕様情報取得用のプロンプト（第3指示情報）を生成する（ステップS120）。具体的には、複数のエンドポイントの中から順に未処理のエンドポイントが対象のエンドポイントとして特定され、当該対象のエンドポイントについての仕様情報取得用のプロンプトが生成される。当該プロンプトは、エンドポイント情報に基づいて、仕様情報の取得を指示するプロンプトである。

[0055] 図10は、仕様情報取得用のプロンプトを例示する。当該プロンプトは、図示するように、エンドポイント情報から、エンドポイントの仕様を記述することを指示しており、具体的には、エンドポイントにおいて実行されるHTTPメソッド (method)、及び、エンドポイントを特定するパス (path) に加えて、エンドポイントの仕様解説 (description) を記述することを指示している。エンドポイント情報（HTTPメソッド、パス、及び、コード）は、検査詳細管理テーブル156から取得される。

[0056] 図7のフローチャートに戻り、続いて、サーバ10は、仕様情報を取得して登録する（ステップS130）。具体的には、仕様情報取得用のプロンプトを大規模言語モデルに対して入力し、当該大規模言語モデルから出力される仕様情報を検査詳細管理テーブル156に登録する。この例では、仕様情報は、エンドポイント情報の取得に用いた上述した大規模言語モデルと同一の大規模言語モデルを用いて取得される。なお、本実施形態の他の例では、仕様情報は、エンドポイント情報の取得に用いた大規模言語モデルとは異なる大規模言語モデルを用いて取得され得る。

[0057] 図11は、図10に例示したエンドポイント情報取得用のプロンプトが入力された大規模言語モデルからの応答を例示する。図示するように、エンドポイント情報に基づいて、エンドポイントの仕様が自然言語（この例では、日本語）で記述されている。

- [0058] 図7のフローチャートに戻り、続いて、サーバ10は、対象のエンドポイントについて脆弱性情報取得用のプロンプト（第2指示情報）を生成する（ステップS140）。具体的には、仕様情報及び脆弱性検査ルールに基づいてエンドポイントの脆弱性情報の取得を指示するプロンプトが生成される。
- [0059] 図12は、脆弱性情報取得用のプロンプトを例示する。当該プロンプトは、図示するように、エンドポイントの仕様情報及びコードから、ユーザによって指定されている脆弱性検査ルールに従って、当該エンドポイントの脆弱性を指摘する（脆弱性名、脆弱性の説明、及び、該当コードを取得する。）ことを指示している。エンドポイントの仕様情報及びコードは、検査詳細管理テーブル156から取得される。また、ユーザによって指定されている脆弱性検査ルールは、検査管理テーブル154から取得される。
- [0060] 図7のフローチャートに戻り、続いて、サーバ10は、脆弱性情報を取得して登録する（ステップS150）。具体的には、脆弱性情報取得用のプロンプトを大規模言語モデルに対して入力し、当該大規模言語モデルから出力される脆弱性情報を含む脆弱性検査結果を検査詳細管理テーブル156に登録する。この例では、脆弱性情報は、エンドポイント情報及び仕様情報の取得に用いた上述した大規模言語モデルと同一の大規模言語モデルを用いて取得される。なお、本実施形態の他の例では、脆弱性情報は、エンドポイント情報及び／又は仕様情報の取得に用いた大規模言語モデルとは異なる大規模言語モデルを用いて取得され得る。
- [0061] 図13は、図12に例示した脆弱性情報取得用のプロンプトが入力された大規模言語モデルからの応答を例示する。図示するように、エンドポイントの仕様情報及びコードから、当該エンドポイントの脆弱性（セッション管理の脆弱性、及び、パスワードの平文保存）が指摘されている。なお、図12に例示したプロンプトにおいて、脆弱性検査ルールとしてのSQLインジェクションの判定除外ルールの記載がない（当該ルールが適用されない）場合には、SQLインジェクションの脆弱性が検知され得る。
- [0062] 図7のフローチャートに戻り、続いて、サーバ10は、未処理のエンドポ

イントがある場合には（ステップS 1 6 0においてYES）、ステップS 1 2 0へと戻り、次のエンドポイントを対象として、仕様情報取得用プロンプトの生成、仕様情報の取得及び登録、脆弱性情報取得用プロンプトの生成、並びに、脆弱性情報の取得及び登録を行う（ステップS 1 2 0～S 1 5 0）。こうした一連の処理は、未処理のエンドポイントがなくなるまで（ステップS 1 6 0においてNO）、繰り返される。

[0063] 上述した例では、エンドポイントの仕様情報が、大規模言語モデルを用いて取得されるようにしたが、当該仕様情報が、ユーザから個別に提供されるようにしてもよい。この場合、ユーザによって提供された、ウェブアプリケーションの仕様情報が記述されたファイル（例えば、検査画面5 0において、当該ファイルを指定するためのボタンが配置される。）から、エンドポイントの仕様情報が取得される。さらに、この場合、ファイルに記述された仕様情報からのエンドポイント単位の仕様情報の抽出を、大規模言語モデルを用いて行うようにしてもよい（この場合のプロンプト及び応答を図1 4、図1 5にそれぞれ例示する。）。

[0064] 以上説明した本実施形態に係る脆弱性検査サーバ1 0は、大規模言語モデルを用いて、ウェブアプリケーションにおける複数のエンドポイントの各々において実行される処理情報を含むエンドポイント情報をソースコードに基づいて取得すると共に、当該エンドポイントの脆弱性情報を、対応するエンドポイント情報及び仕様情報に基づいて取得し、こうした仕組みは、多様な環境で構築される様々なアプリケーションの脆弱性の検査に一律に適用され得る。つまり、サーバ1 0は、様々なアプリケーションの脆弱性の検査を可能とする。

[0065] 本発明の他の実施形態において、上述した実施形態における脆弱性検査サーバ1 0が有する機能の一部又は全部は、脆弱性検査サーバ1 0及びユーザ端末3 0が協働することによって実現され、又は、ユーザ端末3 0によって実現され得る。つまり、本発明のシステムは、脆弱性検査サーバ1 0によって構成される他、脆弱性検査サーバ1 0及びユーザ端末3 0によって構成

され、又は、ユーザ端末30によって構成され得る。

- [0066] 本明細書で説明された処理及び手順は、明示的に説明されたもの以外にも、ソフトウェア、ハードウェアまたはこれらの任意の組み合わせによって実現される。例えば、本明細書で説明される処理及び手順は、集積回路、揮発性メモリ、不揮発性メモリ、磁気ディスク等の媒体に、当該処理及び手順に相当するロジックを実装することによって実現される。また、本明細書で説明された処理及び手順は、当該処理・手順に相当するコンピュータプログラムとして実装し、各種のコンピュータに実行させることが可能である。
- [0067] 本明細書中で説明された処理及び手順が単一の装置、ソフトウェア、コンポーネント、モジュールによって実行される旨が説明されたとしても、そのような処理または手順は複数の装置、複数のソフトウェア、複数のコンポーネント、及び／又は複数のモジュールによって実行され得る。また、本明細書において説明されたソフトウェアおよびハードウェアの要素は、それらをより少ない構成要素に統合して、またはより多い構成要素に分解することによって実現することも可能である。
- [0068] 本明細書において、発明の構成要素が単数もしくは複数のいずれか一方として説明された場合、又は、単数もしくは複数のいずれとも限定せずに説明された場合であっても、文脈上別に解すべき場合を除き、当該構成要素は単数又は複数のいずれであってもよい。

請求の範囲

- [請求項1] 1 又は複数のコンピュータプロセッサを備え、ウェブアプリケーションの脆弱性を検査するためのシステムであって、前記 1 又は複数のコンピュータプロセッサは、
- 前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第 1 指示情報を生成するステップと、
- 前記第 1 指示情報を第 1 大規模言語モデルに対して入力し、前記第 1 大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、
- 前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、
- 前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第 2 指示情報を生成するステップと、
- 前記第 2 指示情報を第 2 大規模言語モデルに対して入力し、前記第 2 大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を実行する、
- システム。
- [請求項2] 前記第 2 大規模言語モデルは、前記第 1 大規模言語モデルと同一の大規模言語モデルである、
- 請求項 1 のシステム。
- [請求項3] 前記処理情報は、前記エンドポイントにおいて実行される HTTP メソッド及び／又はコードを含む、
- 請求項 1 のシステム。
- [請求項4] 前記 1 又は複数のコンピュータプロセッサは、さらに、前記仕様情

報を前記エンドポイント情報に基づいて出力することを指示する第3指示情報を生成するステップを実行し、

前記仕様情報を取得するステップは、前記第3指示情報を第3大規模言語モデルに対して入力し、前記第3大規模言語モデルから出力される前記仕様情報を取得する、

請求項1のシステム。

[請求項5] 前記第3大規模言語モデルは、前記第1大規模言語モデル及び／又は前記第2大規模言語モデルと同一の大規模言語モデルである、

請求項4のシステム。

[請求項6] 前記第2指示情報は、予め定められた脆弱性の検査ルールに従って前記脆弱性情報を出力することを指示する、

請求項1のシステム。

[請求項7] 1又は複数のコンピュータによって実行され、ウェブアプリケーションの脆弱性を検査するための方法であって、

前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第1指示情報を生成するステップと、

前記第1指示情報を第1大規模言語モデルに対して入力し、前記第1大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、

前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、

前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第2指示情報を生成するステップと、

前記第2指示情報を第2大規模言語モデルに対して入力し、前記第

2 大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を備える、

方法。

[請求項8]

ウェブアプリケーションの脆弱性を検査するためのプログラムであって、1又は複数のコンピュータに、

前記ウェブアプリケーションにおける複数のエンドポイントの各々に関するエンドポイント情報であって、前記エンドポイントにおいて実行される処理に関する処理情報を含む前記エンドポイント情報を、前記ウェブアプリケーションのソースコードに基づいて出力することを指示する第1指示情報を生成するステップと、

前記第1指示情報を第1大規模言語モデルに対して入力し、前記第1大規模言語モデルから出力される前記エンドポイント情報を取得するステップと、

前記エンドポイントの仕様が自然言語で記述された仕様情報を取得するステップと、

前記エンドポイントの脆弱性に関する脆弱性情報を、前記エンドポイント情報及び前記仕様情報に基づいて出力することを指示する第2指示情報を生成するステップと、

前記第2指示情報を第2大規模言語モデルに対して入力し、前記第2大規模言語モデルから出力される前記脆弱性情報を取得するステップと、を実行させる、

プログラム。