情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

リクエストパラメータ変換による Webアプリケーション脆弱性診断ツールの 精度向上に関する研究

情報セキュリティ大学院大学
大久保研究室 木村正太朗

2022/02/19

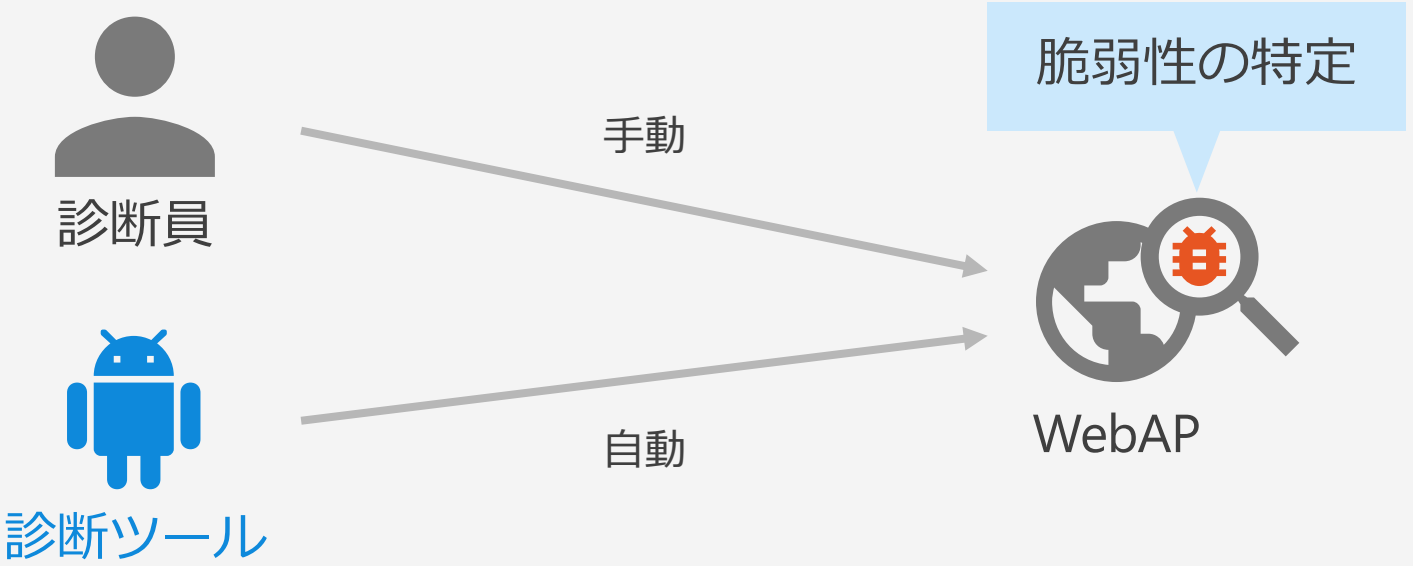
全 57 ページ

目次

目次

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

はじめに



Webアプリケーション脆弱性診断 (以降、Web診断と表記)

- Webアプリケーション (以降、WebAPと表記) の脆弱性を修正するにあたり隠れた脆弱性を特定するために行う
- 一般的に自動化のためツール (以降、診断ツールと表記) が使用される

はじめに

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

診断ツール

- ✓ 自動化された定型的な診断
- ✗ ロジックに係る複雑な診断

診断員

- ✓ サイトの特性に応じた診断
- ✗ 時間・人員等のリソース限界

共に実施することで相互補完が可能

診断ツールが脆弱性を正しく検知できることが前提

経験上、診断ツールが偽陰性を発生させている
（定型的なものも含む）

本研究の概要

目的

- 診断ツールが偽陰性を発生させる原因の特定
- 判明した原因に対する解決策の提案

調査と検証実験・考察

- 実際の Web 診断のデータを基にした調査と実験を実施
- 一般的でないパラメータ(後述)を認識できない問題を指摘

提案手法と評価・考察

- プロキシによりパラメータを変換する手法を提案
- 診断ツールを支援することで診断可能パラメータを拡張

目次

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

パラメータの定義

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

HTTP リクエスト (一例)

```
POST /test?urlparam=aaa HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/97.0.4692.99 Safari/537.36 Edg/97.0.1072.69
Accept: */*
Referer: http://www.example.com
Content-Length: 13
Content-Type: application/x-www-form-urlencoded

formparam=bbb
```

青字は全てパラメータである

XSS* や SQL インジェクション等の脆弱性の診断は、HTTP リクエストのうち
WebAP によって処理される可能性のある変数 に対して診断を行う

▼

本研究の中では下線部を パラメータ という

* XSS : クロスサイトスクリプティング

一般的なパラメータの定義

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

- WebAP がユーザに入力を求める場合、一般的に form タグが使用される
- form タグに設定された属性値に応じて3種類のパラメータ形式が用いられる

HTTP リクエスト

```
POST /test?urlparam=aaa HTTP/1.1
(snip)
Content-Type: application/x-www-form-
urlencoded

formparam=bbb
```

HTTP リクエスト

```
POST /test HTTP/1.1
(snip)
Content-Type: multipart/form-data;
boundary=--------12345678

--------12345678
Content-Disposition: name="file"
filename="sample.txt"

This is a text file.
--------12345678-
```

フォームパラメータ

URLパラメータ

Multipartパラメータ

本研究の中ではこれら3つを 一般的なパラメータ という

* (snip) は一部省略していることを示している

目次

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

関連研究

研究	種類	概要
Doupé et al. [1]	性能評価	独自のWebAPを開発し、性能評価
Rennhard et al. [2]	テストカバレッジ向上	プロキシベースのアプローチ
Pellegrino et al. [3]	テストカバレッジ向上	JavaScriptプログラムの動的解析
Suteva et al. [4]	性能評価	入力ベクトルについても調査有り
Amankwah et al. [5]	性能評価	偽陰性発生なしとする結果
Mburano et al. [6]	性能評価	脆弱性によって性能が異なるとする結果

- クロール性能やテストカバレッジの重要性を訴えるものが多い
  - ? 診断ツールは対象ページを診断できれば、100% 検出できるのか？
  - → 何かしらの原因により偽陰性が発生するはずと考えられる
- 偽陰性が発生していても原因ついて言及している文献は少ない

関連研究との比較

性能評価の観点

既存研究	● ベンチマーク用WebAPに対して実施を用いた評価 ● ファイルアップロードができないという事実のみ
本研究	● 実際のWeb診断のデータを基にした より現実的な評価 ● ファイルアップロードができていてもファイル名に対して診断できないという事実も示し、より詳細

提案するツールの観点（ツールの目的）

既存研究 (Rennhard et al.)	● エンドポイントの認識や認証済みスキャンの信頼性向上
本研究	● 認識できないパラメータに対しても診断可能とすること

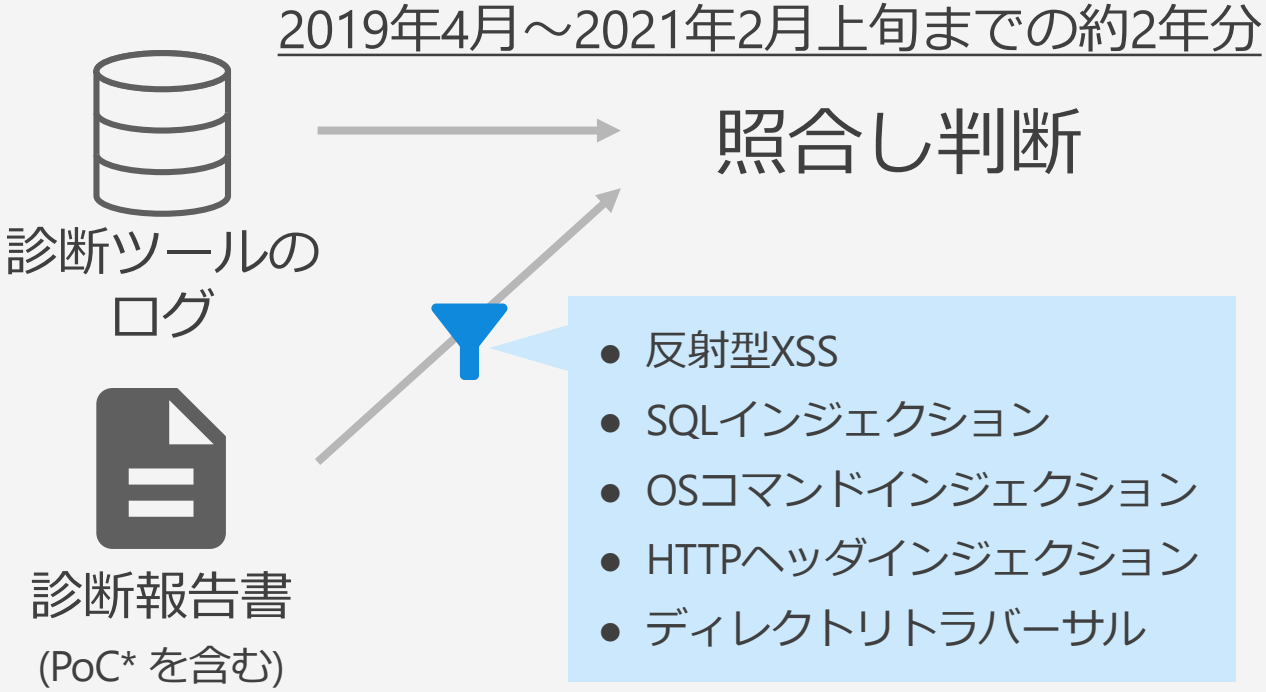
目次

調査1

目的

診断ツールが偽陰性を発生させる原因を特定すること

方法



- 診断ツールを
  使用しなかった事例
- 診断ツールが偽陰性を
  発生させた事例
- 脆弱性があった場合に
  正しく検知できないと
  思われる挙動

これらを収集

* 脆弱性があることを証明するための手順や攻撃コードを含むもの

調査1の結果

問題	概要
認証	診断ツールが認証状態になれず、認証後のページが診断不可
検知ロジック不備	HTML属性値やイベントハンドラに対するXSSの検知が不十分
パラメータ認識不足	ファイル名やパラメータ内の別形式のパラメータへ診断不可

凡例：　先行研究では指摘不十分　　先行研究で指摘済み

ただし本調査では1つのツールによる結果のみであり

調査したツール特有の結果 か 一般的な結果 かが不明

▼

別途 検証実験が必要

調査2

パラメータ認識の不足によって正しく診断できていない可能性

一般的に評価に使われるベンチマーク用WebAPはどうなっている？

脆弱性が埋め込まれている箇所を調査

WebAP	Version/Commit	URL
Bodgeit	1.4.0	https://github.com/psiinon/bodgeit
Damn Vulnerable Web Application (DVWA)	2.0.1	https://github.com/digininja/DVWA/
WackoPicko	065cb92	https://github.com/adamdoupe/WackoPicko
WebGoat	8.1.0	https://github.com/WebGoat/WebGoat/

表. 調査対象としたベンチマーク用WebAP

調査2の結果

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

調査した全てのWebAPにおいて、URLパラメータやフォームデータを使用

診断ツールの評価によく使用されるWebAPは
一般的なパラメータのみに脆弱性を埋め込んでいる

診断ツールが持つ診断パターン（文字列）や検知ロジックによって
脆弱性を検出できるかどうかは評価することができるが、
多様な入力パラメータに対して診断できているかは評価できていない

認識できるパラメータを測る検証実験 を実施

目次

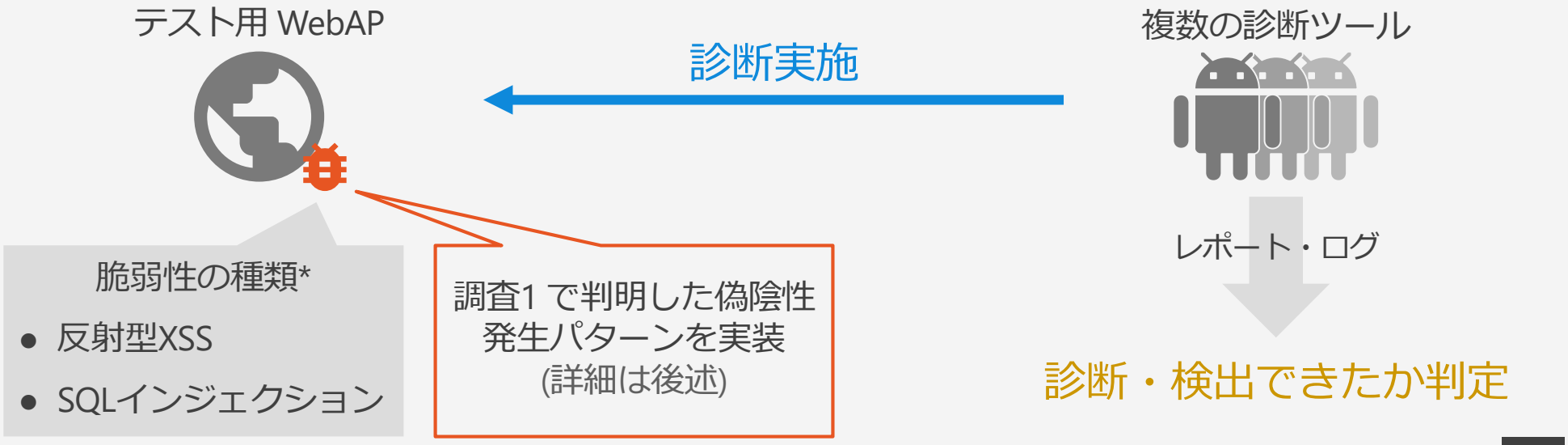
調査結果の検証

目的

診断ツールが多様なパラメータに対して診断できているかを検証する

方法

1. テスト用 WebAP を作成　2. 複数の診断ツールで診断　3. 結果から判定



* 調査で偽陰性が確認された脆弱性のみを対象とした

調査結果の検証

使用した診断ツール

診断ツール	Version	URL	Docker 利用
OWASP ZAP	2.10.0	https://www.zaproxy.org/	
Vega	1.0-devel-130	https://subgraph.com/vega/index.en.html	
Sqlmap	1.5.6.2#dev	https://sqlmap.org/	○
Wapiti	3.0.5	https://pypi.org/project/wapiti3/	○
Arachni	v1.5.1-0.5.12	https://www.arachni-scanner.com/	

診断方法

すべての通信を手動で記録	OWASP ZAP, Vega, Sqlmap*
診断ツールのクロールを実施	Wapiti, Arachni

* プロキシ機能はないため、OWASP ZAP で記録したログを基にリクエストを生成

調査結果の検証

テスト用 WebAP に実装した脆弱性

タイプ	XSS	SQLi	説明
一般的パラメータ	○	○	URLパラメータや x-www-form-urlencoded 方式のパラメータ
イベントハンドラ	○		onclick 等のイベントハンドラへ反射するXSS
User-Agent		○	User-Agent をログイン処理時に記録する想定
Referer	○		Referer ヘッダの値が反射するXSS
Content-Disposition	○	○	ファイルアップロード時の filename 属性
Raw JSON	○	○	Content-Type: application/json でボディにJSON文字列
URL-Encoded JSON	○	○	Raw JSON をURLエンコードした x-www-form-urlencoded 方式
Multipart JSON	○	○	Raw JSON を multipart/form-data 形式のパラメータにセット

- 全てのタイプで3段階の難易度と脆弱性のないものを設置（偽陽性も検出）
- 簡単にクロールできるよう、全てページはリンクで接続

検証結果

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	-	-
Referer XSS	◎	◎		-	-
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△	-	-	△	-
JSON XSS	×	-			
JSON SQLi	×	-	△		
URL-Encoded JSON XSS	-	-			
URL-Encoded JSON SQLi	-	-	△		
Multipart JSON XSS	-	-			
Multipart JSON SQLi	-	-	-		

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

検証結果の考察

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
JSON XSS	-	-			
JSON SQLi	×	-	△		
URL-Encoded JSON XSS	-	-			
URL-Encoded JSON SQLi	-	-	△		
Multipart JSON XSS	-	-			
Multipart JSON SQLi	-	-	-		

イベントハンドラに反射する XSS では全て偽陽性が発生

→ 調査とは真逆の結果であり、一般的ではなかった

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

検証結果の考察

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
JSON XSS	-	-			
JSON SQLi	×	-	△		
URL-Encoded JSON XSS	-	-			
URL-Encoded JSON SQLi	-	-	△		
Multipart JSON XSS	-	-			
Multipart JSON SQLi	-	-	-		

診断ツールによっても大きく異なるが
パラメータとして認識できず正しく診断できないケース
が多く存在した

→ 偽陰性発生の原因の1つと考えられる

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

目次

提案手法

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

```
POST /test?urlparam=aaa HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-
urlencoded
Referer: http://www.example.com/

formparam=bbb&mod_referer=ccc
```

```
POST /test?urlparam=aaa HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-
urlencoded
Referer: ccc

formparam=bbb&mod_referer=ccc
```

凡例：診断できる　診断できない　追加パラメータ

診断ツールを支援するプロキシ

- 診断できるパラメータを追加で認識させ、診断可能なパラメータを増やす
- 追加したパラメータの値を診断できないパラメータに置換

技術的課題

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

```
POST /test?urlparam=aaa HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-
urlencoded
Referer: http://www.example.com/

formparam=bbb&mod_referer=ccc
```

```
POST /test?urlparam=aaa HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-
urlencoded
Referer: ccc

formparam=bbb&mod_referer=ccc
```

課題1

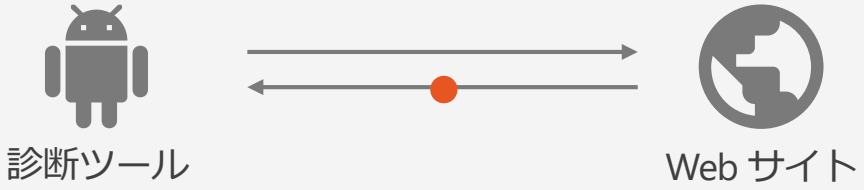
診断ツールに追加でパラメータを認識させ、それに対する正しい応答を記憶させる必要がある

課題2

追加したパラメータの値を、本来のリクエストの形式に合わせて置換する必要がある

課題1 追加パラメータの認識

クローラー型



- 診断ツールが Web サイトを巡回
- レスポンスに応じてリクエスト生成

レスポンスを改変し
新たなパラメータを認識

ローカルプロキシ型

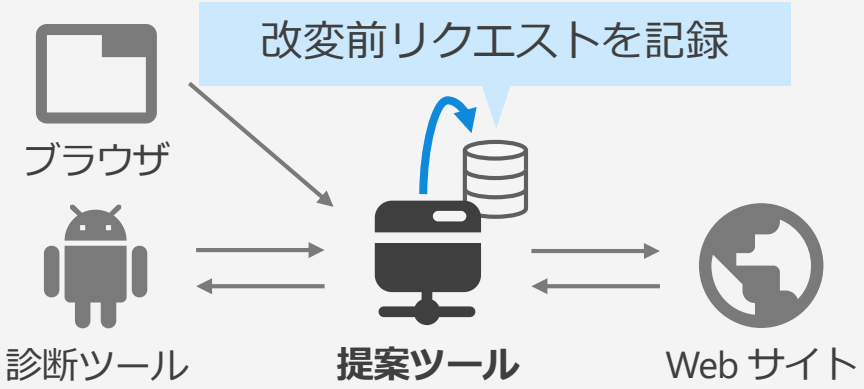
● 改変ポイント

- 診断ツールをブラウザの
  プロキシサーバに設定し通信を記録

ブラウザからのリクエスト
に新たなパラメータを追加

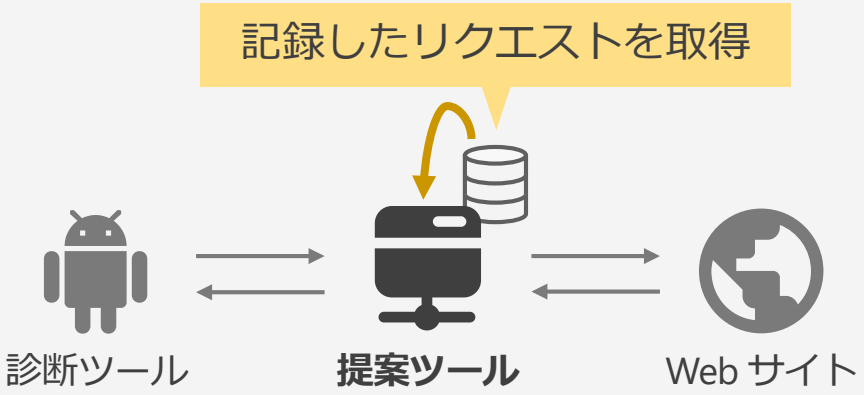
課題2 パラメータの置換

初回アクセス



- 新しいエンドポイントに対する
  リクエストを提案ツールで記録
- 診断ツールに追加のパラメータを
  認識させるための動作を行う

2回目以降のアクセス



- 記録済みのリクエストを取得
- 取得したリクエストの形式に
  合うようにパラメータを置換

提案手法の実装

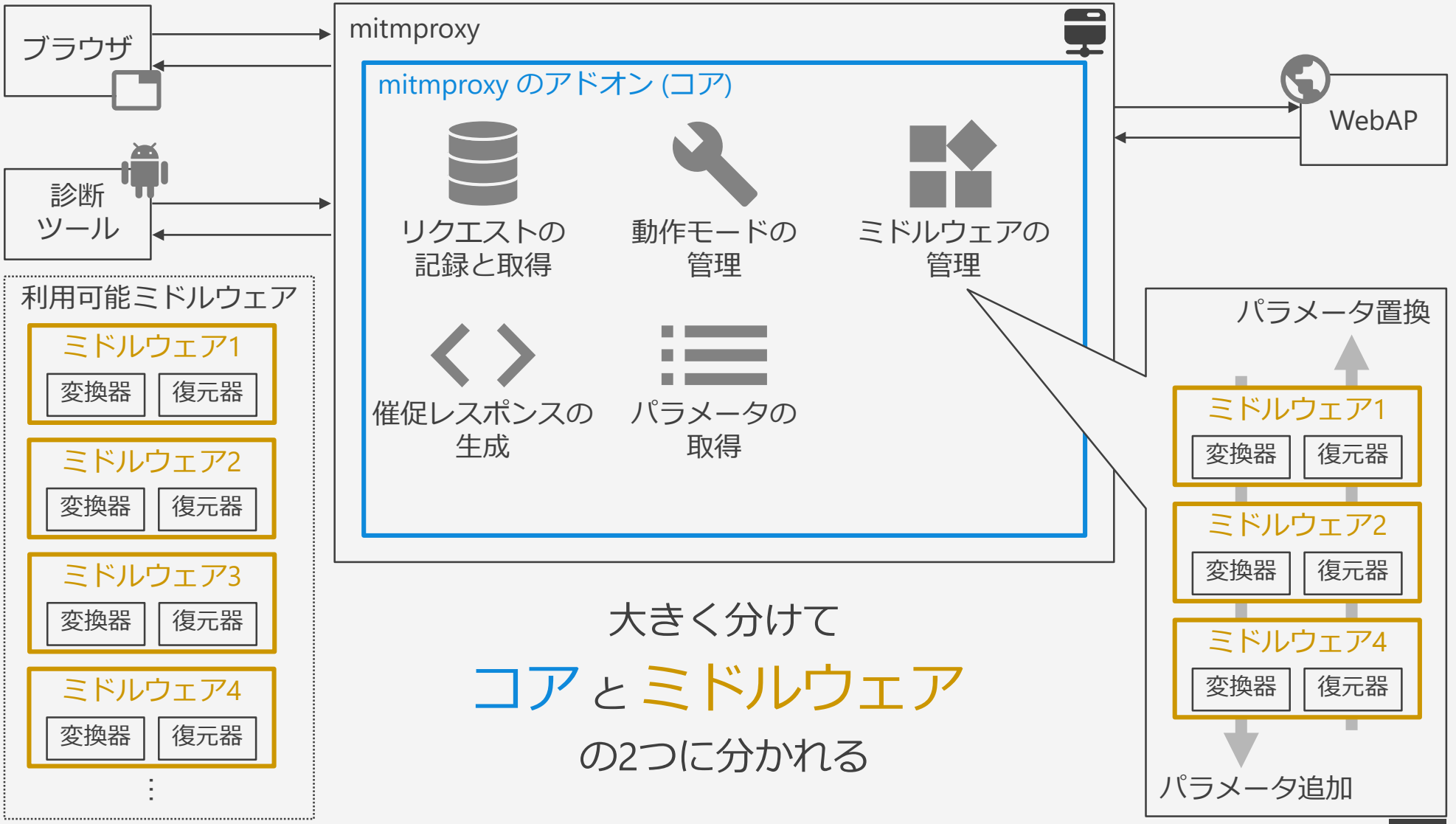
使用したソフトウェア 等

名前	バージョン	概要	URL
Python	3.9.7	プログラミング言語	https://www.python.org/
mitmproxy	7.0.4	Python製プロキシOSS	https://mitmproxy.org/

- mitmproxy では様々な操作が可能な API を提供しておりアドオン開発が可能
- リクエストやレスポンスに応じた柔軟な操作が必要であったことから採用

mitmproxy の アドオンとして動作 するツールを作成

提案ツールの概念

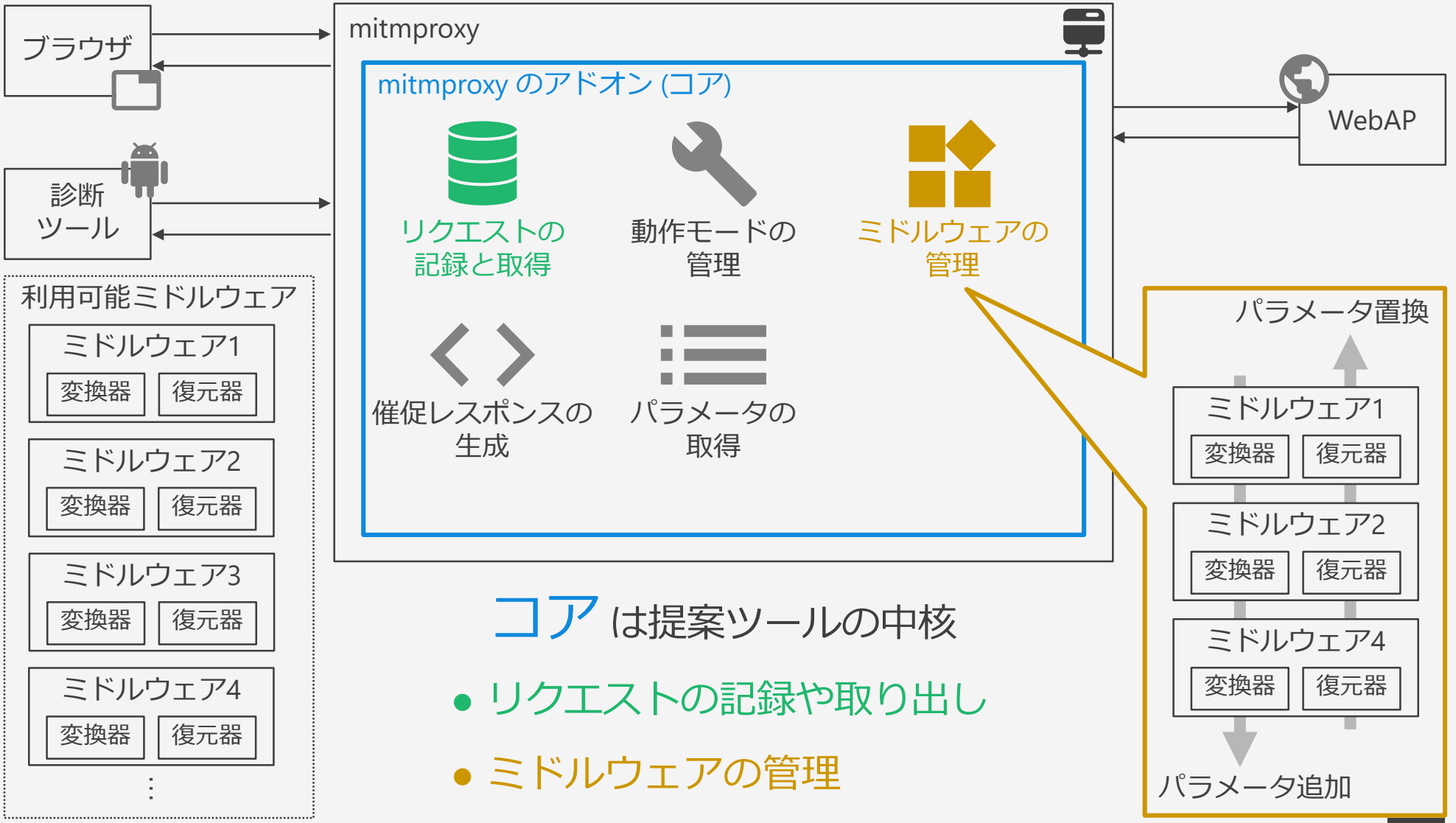


大きく分けて

コア と ミドルウェア

の2つに分かれる

提案ツールの概念



コア は提案ツールの中核

- リクエストの記録や取り出し
- ミドルウェアの管理

提案ツールの概念

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

適用するミドルウェアを切り替えることで
提案手法を適用するパラメータを変更可能

(この図では、1～4 のミドルウェアのうち3以外を適用)

ミドルウェア

ミドルウェアとは

変換器と復元器の2つの機能を持ち、パラメータの置換を担うモジュール

(青字部分については後述)

実装したミドルウェア

リクエストヘッダ	Referer、User-Agent、Cookie ヘッダの値を置換
ファイル名	Multipart 形式の Content-Disposition 内の filename 引数を置換
JSON	JSON 形式のリクエストとパラメータ内の JSON 文字列を置換

追加で定義することで

診断ツールやWebAPの特性に応じた柔軟なパラメータ置換が可能

変換器と復元器のイメージ

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

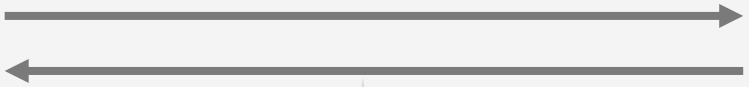
変換器

凡例：

ミドルウェア名
未適用　適用済

```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
User-Agent: hoge
Referer: https://example.com/

param1=value1
```

診断ツール



提案ツール

管理しているミドルウェア

User-Agent
変換器　復元器

Referer
変換器　復元器

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8;
(snip...)

<body>
  <form action="testP~~" action="post">
    <input type="text" name="param1" value="value1">


  </form>
</body>
```

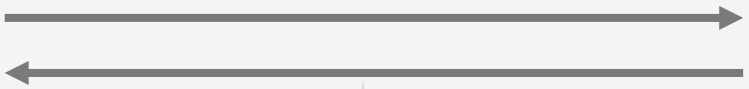
変換器と復元器のイメージ

変換器

凡例：

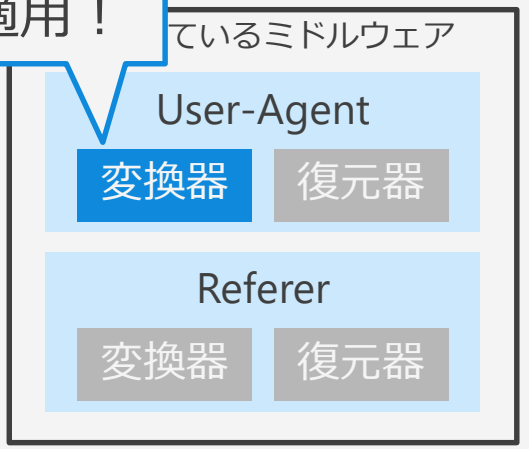

```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
User-Agent: hoge
Referer: https://example.com/

param1=value1
```

診断ツール



提案ツール

適用！




```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8;
(snip...)

<body>
  <form action="testP~~" action="post">
    <input type="text" name="param1" value="value1">
    <input type="text" name="h_useragent" value="hoge">

  </form>
</body>
```

適用！

変換器と復元器のイメージ

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

変換器

凡例：


```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
User-Agent: hoge
Referer: https://example.com/

param1=value1
```


```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8;
(snip...)

<body>
  <form action="testP~~" action="post">
    <input type="text" name="param1" value="value1">
    <input type="text" name="h_useragent" value="hoge">
    <input type="text" name="h_referer" value="https...">
  </form>
</body>
```

適用！

変換器と復元器のイメージ

復元器

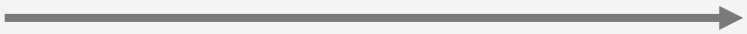
一定規則を満たすときに
復元器を使用

```
POST /testP~~ HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
User-Agent: hoge
Referer: https://example.com/

param1=value1&h_useragent=foo&h_referer=bar
```

凡例：

ミドルウェア名
未適用　適用済

診断ツール



提案ツール

管理しているミドルウェア

User-Agent
変換器　復元器

Referer
変換器　復元器

```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
User-Agent: hoge
Referer: https://example.com/

param1=value1&h_useragent=foo&h_referer=bar
```


Web サイト

※ URL の修正はミドルウェアではなくコアで実施

変換器と復元器のイメージ

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

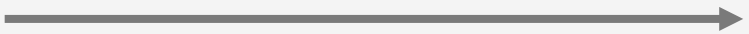
復元器

```
POST /testP~~ HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
User-Agent: hoge
Referer: https://example.com/

param1=value1&h_useragent=foo&h_referer=bar
```

凡例：

ミドルウェア名
未適用　適用済

診断ツール



提案ツール

管理している

適用！

User-Agent
変換器　復元器

Referer
変換器　復元器

```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
User-Agent: hoge
Referer: bar

param1=value1&h_useragent=foo~~&h_referer=bar~~
```

適用！

Web サイト

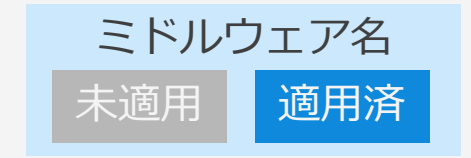
※ URL の修正はミドルウェアではなくコアで実施

変換器と復元器のイメージ

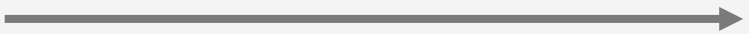
復元器

```
POST /testP~~ HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
User-Agent: hoge
Referer: https://example.com/

param1=value1&h_useragent=foo&h_referer=bar
```

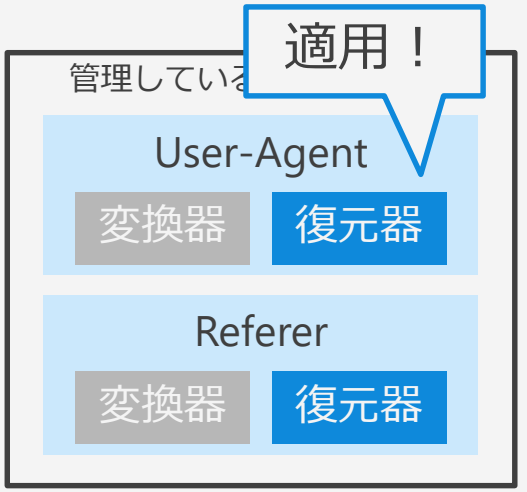
凡例：



診断ツール



提案ツール




```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
User-Agent: foo
Referer: bar

param1=value1&h_useragent=foo&h_referer=bar
```

適用！

Web サイト

※ URL の修正はミドルウェアではなくコアで実施

変換器と復元器のイメージ

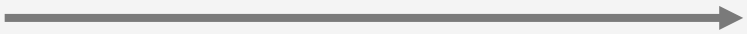
復元器

凡例：

ミドルウェア名
未適用　適用済

```
POST /testP~~ HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
User-Agent: hoge
Referer: https://example.com/

param1=value1&h_useragent=foo&h_referer=bar
```

診断ツール



提案ツール

管理しているミドルウェア

User-Agent
変換器　復元器

Referer
変換器　復元器

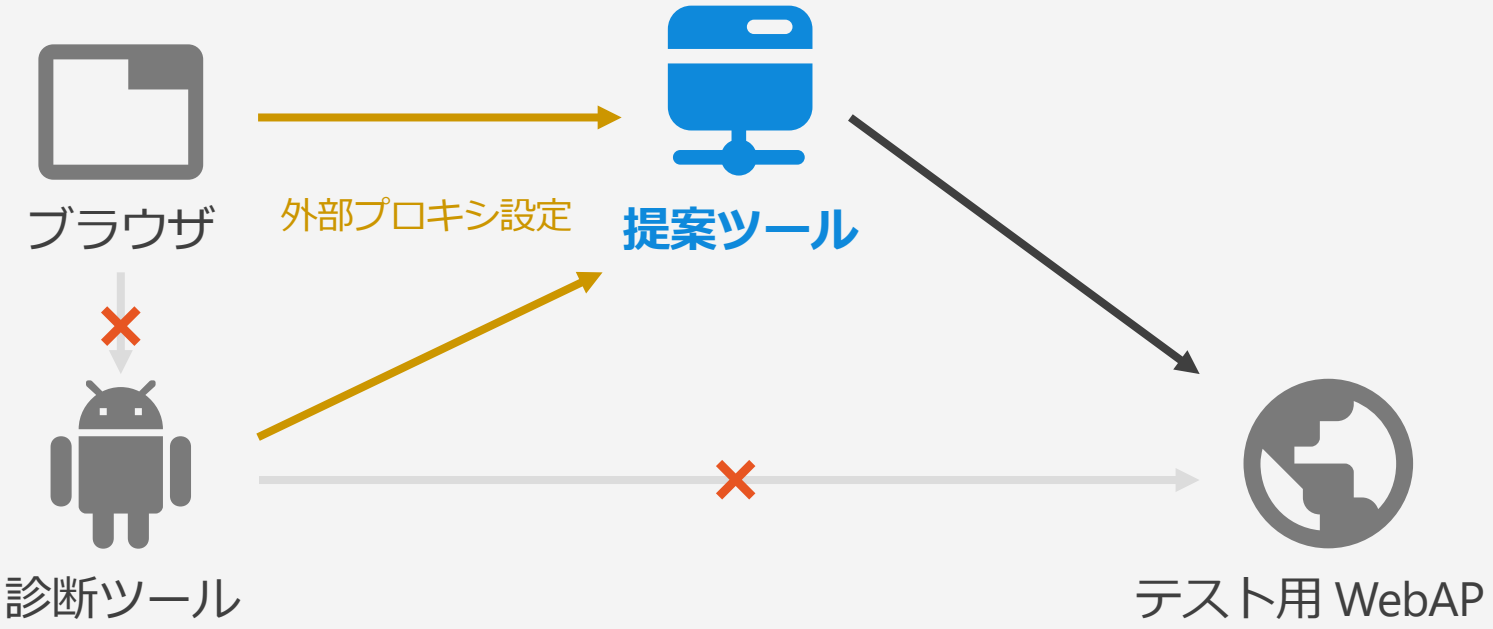
```
POST /test HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 13
User-Agent: foo
Referer: bar

param1=value1
```


Web サイト

※ URL の修正はミドルウェアではなくコアで実施

目次

提案手法の実験と評価



実験方法と実験環境

- 使用した診断ツール、テスト用 WebAP は検証実験で実施した条件と同じ
- 提案ツールを使用するためブラウザと診断ツールの外部プロキシ設定を変更

提案手法の実験と評価

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	△	△
Referer XSS	◎	◎		◎	◎
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△	-	◎	△	-
JSON XSS	×	-			
JSON SQLi	×	-	△		
URL-Encoded JSON XSS	×	-			
URL-Encoded JSON SQLi	×	-	△		
Multipart JSON XSS	×	-			
Multipart JSON SQLi	×	-	△		

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

提案手法の実験と評価

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	△	△
Referer XSS	◎	◎		◎	◎
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△	-	◎	△	-
JSON XSS	×				
JSON SQLi	×				
URL-Encoded JSON XSS	×				
URL-Encoded JSON SQLi	×				
Multipart JSON XSS	×	-			
Multipart JSON SQLi	×	-	△		

OWASP ZAP と Wapiti にでは
診断を実施した項目について
用意した全てのパターンに対して
診断可能となった。

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

提案手法の実験と評価

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	△	△
Referer XSS	◎	◎		◎	◎
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△				
JSON XSS	×				
JSON SQLi	×				
URL-Encoded JSON XSS	×				
URL-Encoded JSON SQLi	×	-	△		
Multipart JSON XSS	×	-			
Multipart JSON SQLi	×	-	△		

SQLインジェクションの検知率が悪い。
診断時の設定で強度を "超" にしたところ検知されるようになったが偽陽性も発生

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

提案手法の実験と評価

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	△	△
Referer XSS	◎	◎		◎	◎
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△	-	◎	△	-
JSON XSS	×				
JSON SQLi	×				
URL-Encoded JSON XSS	×				
URL-Encoded JSON SQLi	×				
Multipart JSON XSS	×	-			
Multipart JSON SQLi	×	-	△		

Vega と Arachni については
Multipart 形式のリクエストが発行できず
ファイル名に対する診断ができなかった

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

提案手法の実験と評価

タイプ	ZAP	Vega	Sqlmap	Wapiti	Arachni
一般的なパラメータXSS	◎	◎		◎	◎
一般的なパラメータSQLi	×	△	◎	△	△
イベントハンドラXSS	○	○		○	○
User-Agent SQLi	×	◎	◎	△	△
Referer XSS	◎	◎		◎	◎
Content-Disposition XSS	◎	-		◎	-
Content-Disposition SQLi	△	-	◎	△	-
JSON XSS	×				
JSON SQLi	×				
URL-Encoded JSON XSS	×				
URL-Encoded JSON SQLi	×				
Multipart JSON XSS	×				
Multipart JSON SQLi	×				

提案ツールを使用していなかったときに検知できていた脆弱性が、提案ツールの使用によって検知できなくなった事象は確認できなかった。

→ ツール使用による新たな偽陰性は発生しなかった。

- … 診断せず　× … 検知せず　△ … 一部検知せず　○ … 全て検知（FPあり）　◎ … 全て検知　* 網掛けは実施せず

提案手法の実験と評価

影響確認のための実験

提案手法を用いることにより、送信リクエスト数の増大や
プロキシのオーバーヘッドによって、処理時間が長くなる

提案手法の使用有無による
送信リクエスト数 と **診断時間** の比較を実施

(参考) 取得方法

診断ツールが診断終了時に
表示するリクエスト数を取得

以下の方法のいずれか

- 最初と最後のリクエストの送信時間の差分
- 診断ツールが診断終了時に表示する処理時間
- time コマンドの結果

提案手法の実験と評価

提案ツール有無による比較

ツール名	送信リクエスト数		診断時間	
	提案ツール無	提案ツール有	提案ツール無	提案ツール有
OWASP ZAP	10,962	46,144	0:16:42	1:04:35
Vega	-	-	0:17:09	0:16:59
Wapiti	11,980	103,277	0:03:53	0:33:31
Arachni	112,219	381,710	0:21:59	1:54:22
Sqlmal (平均*)	11,082	18,573	0:07:24	0:22:40

- Vega は送信リクエスト数を表示する機能がなく取得できなかった
- パラメータの追加ができておらず、単に提案ツールを経由しただけだった

* Sqlmap はエンドポイント毎に測定したため、それらの平均値を記載

提案手法の実験と評価

提案ツール有無による比較

ツール名	送信リクエスト数		診断時間	
	提案ツール無	提案ツール有	提案ツール無	提案ツール有
OWASP ZAP	10,962	46,144	0:16:42	1:04:35
Vega	-	-	0:17:09	0:16:59
Wapiti	11,980	103,277	0:03:53	0:33:31
Arachni	112,219	381,710	0:21:59	1:54:22
Sqlmal (平均*)	11,082	18,573	0:07:24	0:22:40

x8.6 x8.6 x3.4 x5.2

- クローラー型ではエンドポイントの追加を行ったことでリクエスト数が増大
- Wapiti では診断時間の増加が送信リクエスト数に比例
- Arachni では高速な処理に遅延が発生したことで診断時間がさらに増加

* Sqlmap はエンドポイント毎に測定したため、それらの平均値を記載

提案手法の実験と評価

提案ツール有無による比較

ツール名	送信リクエスト数		診断時間	
	提案ツール無	提案ツール有	提案ツール無	提案ツール有
OWASP ZAP	10,962	46,144	0:16:42	1:04:35
Vega	-	-	0:17:09	0:16:59
Wapiti	11,980	103,277	0:03:53	0:33:31
Arachni	112,219	381,710	0:21:59	1:54:22
Sqlmal (平均*)	11,082	18,573	0:07:24	0:22:40

x4.2

x3.9

- ローカルプロキシ型でもパラメータ数が増えたことでリクエスト数が増大
- OWASP ZAP では診断時間の増加率が送信リクエストの増加率より小さい
- 提案ツールを経由することによるオーバーヘッドはあまり大きくない

* Sqlmap はエンドポイント毎に測定したため、それらの平均値を記載

目次

情報セキュリティ大学院大学
INSTITUTE of INFORMATION SECURITY

考察

提案手法の有効性

- 診断ツールの内部構造を変更せずに、元々の診断ツールの検知精度を低下させることなく、診断可能なパラメータを拡張することができた
- 本研究で取り扱ったツールの他にも、外部プロキシの設定をすることができる機能があれば、動作可能であると考えている

提案手法を使用することによる影響

- 送信リクエスト数が増加することによって診断時間の増加する
  → GETメソッドではパラメータ置換をしないなどの工夫が必要

考察

提案手法の限界

- 診断ツールが対応していない形式のリクエストについては効果を発揮しない
  → Multipart 形式に対応していない Vega や Arachni では提案手法を用いても
  ファイル名に対しては診断できなかった
  → 診断ツールの使用時にはこうした対応状況を確認することが重要
- 意図的に追加したパラメータに対して診断するため、レポートの内容が
  実際には存在しないパラメータに対するものになってしまう
  → 特にクローラー型では実際に存在しないエンドポイントに対する
  レポート結果となってしまう

目次

まとめと今後の課題

まとめ

- 一般的でないパラメータに対して診断ツールが正しく診断できない可能性について、実際のデータを用いた調査と検証実験を行った
- 解決策として診断ツールと WebAP 間で動作するプロキシを提案し、その有効性を実験によって示した

今後の課題

- HTTPS を用いた場合への対応や影響調査
- JSON 形式を用いた XSS に対しても診断ができるような仕組みづくり